Funded by
the European Union

University of Craiova

University of Plovdiv
"Paisii Hilendarski"

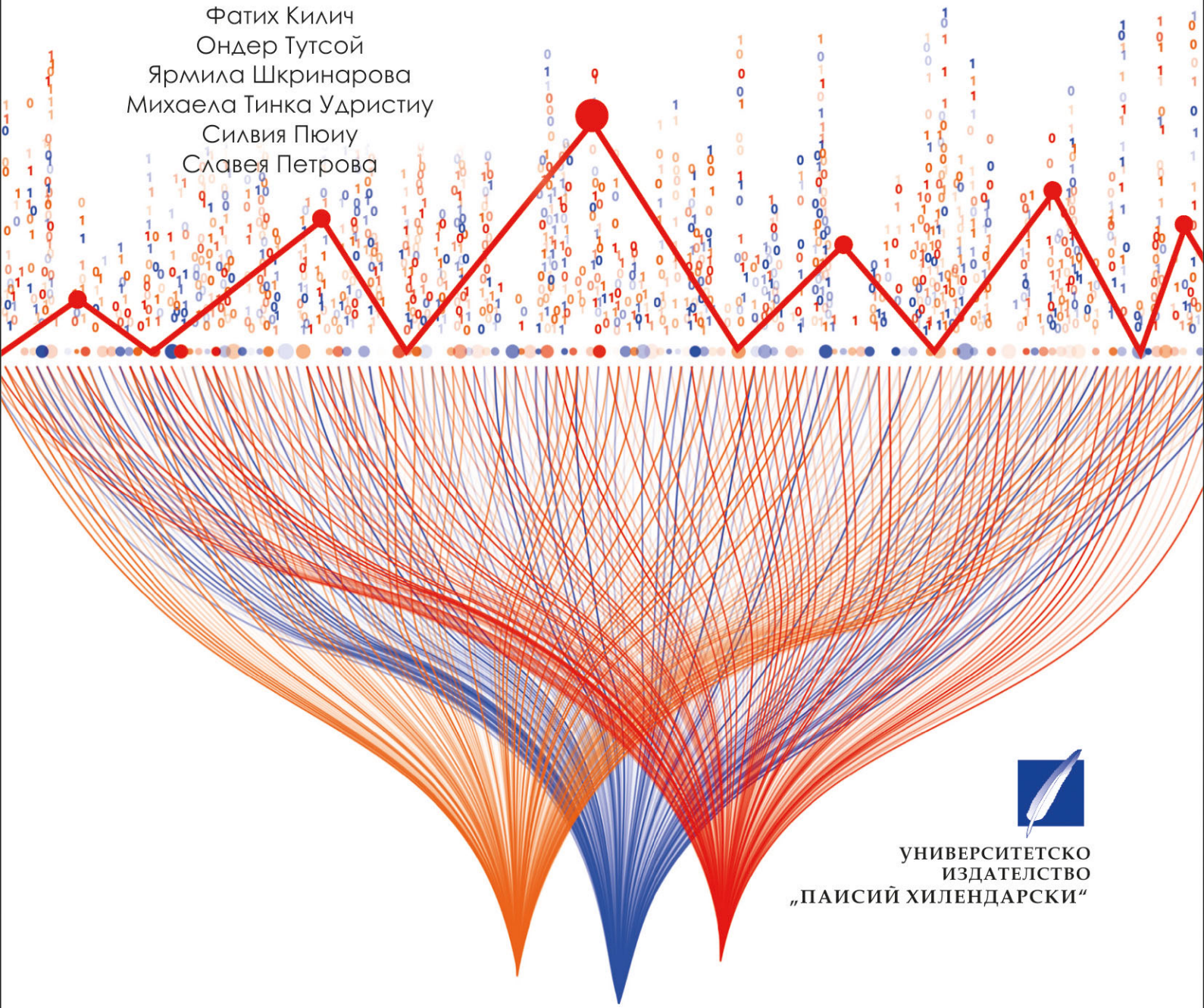
Adana Alparslan Türkeş
Science and Technology
University

UNIVERZITA
MATEJA BELA
Matej Bel University,
Banská Bystrica

УСЪВЪРШЕНСТВАНИ ТЕХНОЛОГИИ ЗА ОБРАБОТКА И АНАЛИЗ НА ГОЛЕМИ МАСИВИ ОТ ДАННИ

Автори
Михаела Тинка Удристиу
Адам Дудаш
Алжбета Михаликова
Фатих Килич
Ондер Тутсой
Ярмила Шкринарова
Михаела Тинка Удристиу
Силвия Пюиу
Славея Петрова

Редактор
Адам Дудаш



УНИВЕРСИТЕТСКО
ИЗДАТЕЛСТВО
„ПАИСИЙ ХИЛЕНДАРСКИ“

Проект – Erasmus+ 2021-1-RO01-KA220-HED-000030286 със заглавие:
Прилагане на някои съвременни технологии в преподаването и научните изследвания във връзка със замърсяването на въздуха

УСЪВЪРШЕНСТВАНИ ТЕХНОЛОГИИ ЗА ОБРАБОТКА И АНАЛИЗ НА ГОЛЕМИ МАСИВИ ОТ ДАННИ

УНИВЕРСИТЕТСКО ИЗДАТЕЛСТВО
„ПАИСИЙ ХИЛЕНДАРСКИ“

Автори:

Михаела Тинка Удристиу (Въведение)

Адам Дудаш (Секции 1-4, Приложение А, форматиране)

Алжбета Михаликова (Секции 5-8, Приложения А и Б)

Фатих Килич (Секция 9)

Ондер Тутсой (Секция 10)

Ярмила Шкринарова (Секция 11)

Михаела Тинка Удристиу, Силвия Пюиу (Приложение В)

Славея Петрова (Приложение Г)

ISBN 978-619-7663-80-8

СЪДЪРЖАНИЕ

	ВЪВЕДЕНИЕ	5
1	ДАННИ И ТЕХНИТЕ СВОЙСТВА	7
2	ОБРАБОТКА И АНАЛИЗ НА ДАННИ	14
3	МЕТОДИ ЗА ИЗГОТВЯНЕ НА ИЗВАДКИ ОТ ДАННИ	24
4	ОСНОВИ НА ИЗСЛЕДОВАТЕЛСКИЯ АНАЛИЗ НА ДАННИ	34
5	РАЗМИТИ (FUZZY) НАБОРИ	67
6	РАЗМИТИ РАЗСЪЖДЕНИЯ	80
7	ИЗПОЛЗВАНЕ НА МЕТОДА СУГЕНО ЗА КЛАСИФИКАЦИЯ НА ДАННИ	84
8	ИЗПОЛЗВАНЕ НА МЕТОДА НА СУГЕНО ЗА АПРОКСИМАЦИЯ НА ДАННИ	90
9	ВЪВЕДЕНИЕ В ОПТИМИЗАЦИЯТА	99
10	ЕДНОСЛОЙНА НЕВРОННА МРЕЖА	108
11	ВНЕДРЯВАНЕ НА НЕВРОННА МРЕЖА	119
12	ПРИЛОЖЕНИЯ	147
	ЛИТЕРАТУРНИ ИЗТОЧНИЦИ	186

ВЪВЕДЕНИЕ

Този наръчник е резултат от проект № 2021-1-RO01-KA220-HED-000030286 по програма „Еразъм+“, озаглавен „Прилагане на някои съвременни технологии в обучението и научните изследвания във връзка със замърсяването на въздуха“. Четирима партньори (Университетът „Матей Бел“ в Банска Бистрица, Словакия, Университетът в Крайова, Румъния, Пловдивският университет „Паисий Хилендарски“, България, и Университетът за наука и технологии в Адана, Турция) работиха заедно за постигането на този резултат. Той има за цел да помогне на преподавателите по STEM да подобрят уменията на студентите за работа с данни.

Ние сме претоварени от информацията, която има около нас. В днешно време е необходимо да знаем как да обработваме данните, като извличаме релевантна информация за всяка цел. Във всяка секунда компютрите, сензорните мрежи и сателитите събират милиони стойности за физически величини и параметри. Базите данни съхраняват и организират данни и информация, като подобряват качеството на данните. Повече от всякога информацията е сила; от този момент учениците от STEM трябва да се научат как да работят с данни. Компаниите изискват висшето образование да осигурява висококвалифицирани висшисти, способни да решават проблеми въз основа на информацията, предоставена от базите данни, или с помощта на специализирани програми или алгоритми. В университетите студентите по STEM трябва да изучават как се събират, анализират и интерпретират набори от данни. Също така те трябва да разбират да правят класификации, приближения и оценки на данни. И накрая, пазарът на труда изисква от завършилите STEM да предвиждат как се развиват процесите в пространството и времето или да вземат решения. Машинното обучение и изкуственият интелект са стандартни термини в ежедневния речник на учениците.

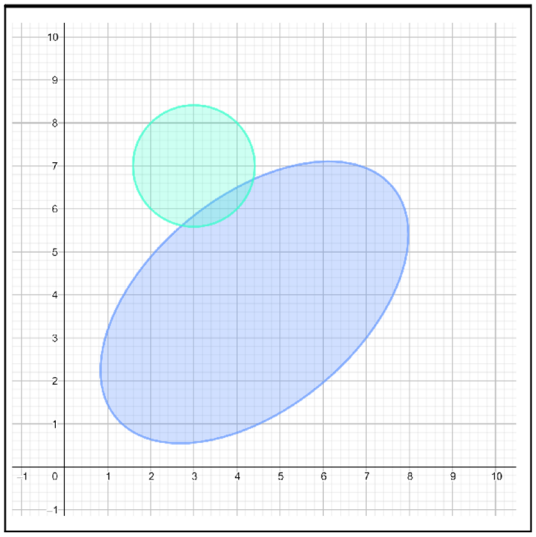
Настоящият наръчник се състои от единадесет раздела, приложения и препратки. Първата част е посветена на различните видове данни, техните свойства, методите за вземане на проби от данни и начините за обработка и анализ на данни. В следващите раздели е разгледан един от най-значимите проблеми, свързани с големите масиви от данни, а именно анализът на данните. При анализа на големи масиви от данни е необходимо да се знае как да се използват подходящи методи за статистически анализ, визуализация на данни и други проучвателни, прогнозни и оценъчни методи. Различни раздели се фокусират върху подходи като ма-

шинно обучение, размити изводи и система от невронни мрежи. Приложението съдържа описание на набора от данни Iris, примери за решения на някои проблеми, набори от данни за изменението на климата или замърсяването на въздуха и информация за въздействието на замърсяването на въздуха върху човешкото здраве. Примерна учебна програма за курс по „Съвременни технологии за обработка и анализ на големи масиви от данни“ затваря този наръчник.

1

ДАННИ И ТЕХНИТЕ СВОЙСТВА

Тази част от ръководството е написана от Адам Дудаш от Департамента по компютърни науки, Факултет по природни науки, Университет „Матей Бел“ в Банска Бистрица, Словакия.



	A	B	C	D
1	5.1	3.5	1.4	0.2
2	4.9	3.0	1.4	0.2
3	4.7	3.2	1.3	0.2
4	4.6	3.1	1.5	0.2
5	5.0	3.6	1.4	0.2
6	5.4	3.9	1.7	0.4
7	4.6	3.4	1.4	0.3
8	5.0	3.4	1.5	0.2
9	4.4	2.9	1.4	0.2
10	4.9	3.1	1.5	0.1
11	5.4	3.7	1.5	0.2
12	4.8	3.4	1.6	0.2
13	4.8	3.0	1.4	0.1
14	4.3	3.0	1.1	0.1
15	5.8	4.0	1.2	0.2
16	5.7	4.4	1.5	0.4
17	5.4	3.9	1.3	0.4
18	5.1	3.5	1.4	0.3

48° 44' 10.597" N 19° 8' 46.291" E

Този наръчник е насочен към демонстриране на прости методи за анализ на данни с помощта на техники от компютърните науки, като изкуствен интелект, машинно обучение или невронни мрежи. В следващия раздел на работата ще разгледаме основните термини и понятия в областта на данните, техните свойства, обработка и анализ.

Данните са технически, статистически, икономически или друг вид бележки или информация, които могат да бъдат обработени с помощта на технически средства – в нашия случай тези технически средства са компютри. Ние подхождаме към данните като към обекти, които се интегрират и споделят в рамките на системата:

- **Интегриране на данни** – данните могат да бъдат поставени в няколко файла, така че дублирането да бъде сведено до минимум, а достъпът до данните от няколко файла да бъде осъществен едновременно.
- **Споделяне на данни** – всеки обект на данни може да бъде споделен от множество потребители (многократно и едновременно).

Най-важното свойство на данните е **устойчивостта** – устойчиви данни са тези, които съществуват дори след прекратяване на програмата. Като се има предвид, че входните данни могат да бъдат трансформирани в постоянни, а изходните данни могат да бъдат трансформирани от постоянни данни, входни данни или получени от тях. Данните, които могат да бъдат изведени от други данни, не трябва да бъдат устойчиви (увеличаваме разходите, свързани с работата на системата) – но понякога това е необходимо.

Важно е да се реши как ще бъдат представени данните в записите в съответствие с определените типове (като се вземе предвид възможно най-ефективното съхранение). Най-типичните **видове данни** са:

- **Числени данни –** могат да се съхраняват по различни начини (двоичен, символен, полулогаритмичен ...), често е необходимо да се определи броят на необходимите битове/байтове за числото.
- **Стрингове** – могат да се съхраняват в различни набори от символи (ACII, UNICODE, EBDIC ...).
- **Enumerators (Изброяване)** – използване на символни кодове вместо низове (напр. А вместо отличен ...).
- **Единици** – трябва да се адаптират към конкретната ситуация (безсмислица: разстоянието на полета се измерва в милиметри).

Тези реализации на типове данни не са много интересни за нас в контекста на анализа на данни. Като цяло ще говорим за два типа данни, които се различават по своето съдържание:

- **Количествени данни, състоящи се от числови стойности** (височина, разстояние, брой ...). Такива данни могат да се използват директно в математически модели, което е от решаващо значение от гледна точка на анализа на данни, основан на методите за машинно обучение.
- **Категорийните данни се състоят от езикови обозначения** на свойства (пол, цвят, вид ...), което предполага необходимостта от специфични методи за анализ на данните. Някои категорийни данни могат да бъдат кодирани в количествени, но такава операция не винаги е смислена. Пример за такова кодиране може да бъде нещо като: АКО полът е мъжки, то полът е 1, АКО полът е женски, то полът е 2 и т.н. Това донякъде има смисъл, но има някои въпроси, които не насърчават подобно кодиране:

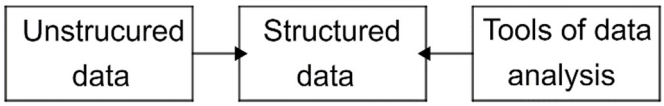
  Тъй като 2 – 1 = 1, дали жена – мъж = мъж?

  Каква е максималната стойност на пола?

Най-важното нещо преди анализа на данните е те да бъдат правилно структурирани. **Разграничаваме три вида данни от гледна точка на структурата**:

- **Структурирани данни** – данни, съхранявани под формата на таблици или файл, в които е възможно да се идентифицират едни и същи свойства в един и същи ред (колони) за всеки регистриран обект (ред). Най-често използваните формати за структурирани данни са csv, excel файл, обикновен текстов файл или SQL база данни.
- **Полуструктурирани данни** – данните са структурирани, но формата им не е фиксирана. Следователно можем да определим едни и същи свойства за всеки обект (ред), но тези свойства може да не са записани всички за всеки запис или да не са в един и същи ред за всички записи (колоните не могат да бъдат идентифицирани). Това е типично за данни от набор от сензори, от мобилни приложения и други подобни. Форматите, използвани за този тип данни, са XML, JSON или MongoDB.
- **Многоструктурирани/неструктурирани данни –** необработени данни в различни формати. Например необработени данни от сензори, уеб логове, данни от социални мрежи, аудио, видео, изображения, 3D модели, координати и други подобни.

При работа с неструктурирани данни обикновено използваме следния работен процес:



Пример: Преобразуване на неструктурирани данни в структурирани (таблица). Нека имаме събрана основна информация за трима ученици:

1. Мартин, мъж, 28.6.1983, година на обучение 2, 40 годишен

2. Джейн, 1994-9-13., Ж, 1. Година на обучение, 29 годишна

3. Мириам, жена, 05.04 1992, втора година на обучение, възраст: 31

Информацията е в непоследователна форма – редът на отделните характеристики на хората е различен за всеки от учениците, а форматът на отделните свойства също е различен. Данните съдържат и лесно изчислими данни, които не е необходимо да се съхраняват (възраст). Ето защо, когато съхраняваме данните в структурирана форма (таблица), трябва да уеднаквим реда и формата на всички свойства (напр. данни във формат ГГГГ-ММ-ДД).

Име	***Дата на раждане***	***Година на следване***	***Пол***
Мартин	*1983-6-28*	*2*	*М*
Джейн	*1994-9-13*	*1*	*Ж*
Мириам	*1992-4-5*	*2*	*Ж*

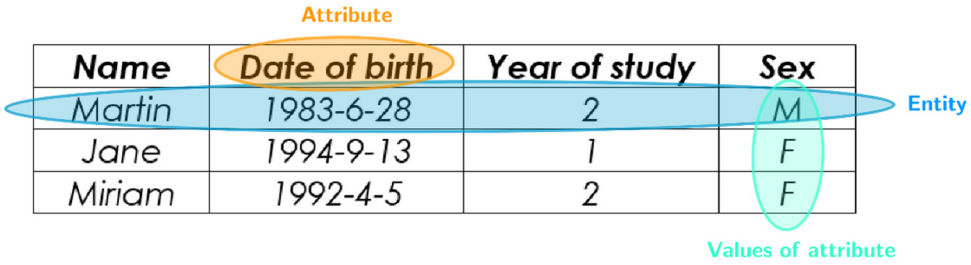
В контекста на този наръчник ще използваме различни наименования за едни и същи обекти в наборите от данни, затова по-долу представяме кратко обяснение на термините.

Субект, обект или запис е обект от реалния свят, който е способен на самостоятелно съществуване и е ясно разграничен от други обекти.

Атрибут или свойство е функция, задаваща стойност на същност, която определя някакво съществено свойство на същността (например ръст, възраст ...).

Наборът от данни или таблицата е съвкупност от единици, състоящи се от един и същ набор от атрибути**.**

Структурирането на събраните данни е основен метод за обработка на данни (вж. раздел 2).



1.1. НЯКОЛКО ДУМИ ПО ТЕМАТА ЗА BIG DATA

По принцип винаги е по-добре да разполагаме с много данни, отколкото с малко (винаги можем да изхвърлим някои записи). Можем да наречем данните големи, ако обработката и анализът им не са възможни с конвенционални инструменти в практически срок. Разбира се, въпросите могат да бъдат: *Какво е практическото време? Какво е конвенционален инструмент?* Затова използваме определението за големи данни, като изброяваме техните свойства.

Данните обикновено се наричат големи данни (Big Data), когато те придобиват свойства, наречени 3V-та (броят на тези „V-та“ се увеличава с течение на времето, като в някои литературни източници 5V-та е най-основният модел на разглеждане на големите данни):

- **Обем на данните** – Обемът на данните, които могат да бъдат получени от различни източници, прави немислимо използването на прости модели на релационни бази данни. Не сме в състояние да представим данните с проста таблица или набор от таблици и да работим на една машина. Ето защо необходимостта от разработване на по-усъвършенствана компютърна инфраструктура и прилагане на оптимизирани алгоритми нараства по своята значимост. Тази система трябва да прилага високопроизводителни,

разпределени и облачни изчислителни принципи, нерелационни бази данни, които са в състояние да съхраняват хомогенни и силно свързани помежду си данни, и модели на изкуствен интелект както за обработка, така и за анализ на данните.

- **Разнообразие на данните** – Тъй като наборът от данни, които се разглеждат в случай на реални проблеми, свързани с големи обеми данни, не е хомогенен, трябва да можем да работим с голям брой типове и формати файлове, например обикновени текстови документи, аудиофайлове, видеофайлове, координати или компютърни модели с повече от две измерения. Повечето от тези видове данни не могат да се съхраняват в релационни бази данни и изискват голямо количество изчислителна мощност и пространство за съхранение, за да бъдат успешно и удобно съхранени и обработени за по-нататъшно използване.
- **Скорост на движение на данните** – Големите масиви от данни често са „живи“ (или динамични) масиви от данни – масиви от данни, които се променят с течение на времето. Тази промяна на данните с течение на времето се случва във всички системи, които използват т. нар. източници на данни от околната среда – източници, които са винаги активни и събират данни, като например сензори от интернет на нещата, които измерват прогресията на един и същ набор от стойности. Тази активност на данните и тяхната обработка, съхранение и анализ създават потоци от данни, които постъпват в системата. Това води до едно от най-важните изисквания към системите за големи данни – възможност за събиране, съхраняване, обработка и анализ на данни в (почти) реално време.

В допълнение към този проблем с живите данни можем да идентифицираме проблема с комбинацията от динамична част от набора от данни и статична част от набора от данни, което предизвиква няколко проблемни събития в системата.

Освен с обема, разнообразието и скоростта на данните, броят на източниците работи и с достоверността и стойността:

- **Достоверност на данните** – Тъй като големите данни често се използват при вземането на решения, които засягат редица реално съществуващи субекти, съществува сериозна необходимост от надеждни и сигурни данни. Необходимо е да се разгледа метрика, която измерва доверието ни в данните. Това е важно не само при вземането на решения, но и при реалността на данните – генерирането на големи данни не е трудно и следователно може лесно да се използва като част от атаки, които целят да претоварят целевата система.
- **Стойност на данните** – както беше споменато по-горе, големите данни могат да се използват (и се използват) за вземане на решения. Стойността на данните се увеличава с увеличаването на

обема на данните, свързани с предмета на изследване, или с прецизността на потенциала за оценка/предвиждане, свързан с данните. Тази стойност може да бъде от парично, бизнес, човешко, изследователско или друго значение.

Тези свойства на големите данни пораждат редица **проблеми, свързани с тяхната обработка и анализ**.

Първият от тези проблеми е свързан с **размера на самите данни.** Техният размер е важен не само в контекста на пространството в паметта, което е необходимо за съхраняването на самите данни, но и от гледна точка на търсенето в данните и анализа на тези данни. При работа с такива данни е необходимо да се използват високопроизводителни, разпределени или облачни изчислителни методи в съчетание с алгоритми за изкуствен интелект от областите на машинното обучение, размитите системи за извод и невронните мрежи за получаване на знания от този вид данни.

Освен че тези данни са големи по размер, те често се състоят от **разнородни дялове, които могат да се различават в няколко аспекта** – размерност на данните, състав на данните, структура на данните, но също така и използвани измервания. Тази несъгласуваност е следствие от факта, че масивите от големи данни често се събират от няколко несъвместими източника в едно хранилище. Поради това е необходимо дяловете с данни да бъдат преформатирани (или по-точно отделните формати на данните да бъдат донякъде унифицирани). Това е представено като поредица от прости задачи, които трябва да бъдат изпълнени върху данните (като например преобразуване на измерванията, ако е необходимо), но също така и задачи, които са по-сложни, например идентифициране на отклонения и липсващи стойности. В случай на липсващи стойности може да бъдат предприети действия за изчисляване на липсващите стойности – могат да се използват методи за машинно обучение и невронни мрежи за оценка на стойностите или класификация на данните.

Проблемът, тясно свързан с хетерогенността на големите масиви от данни, се състои в **жизнеността на тези масиви от данни**. В случай, че измерваме данни с помощта на източници на данни от околната среда (като например сензорни мрежи) и тези измервания се извършват на достатъчно голям брой сензори през достатъчно малки интервали от време, ние създаваме потоци от данни, които трябва да бъдат обработени и подготвени за анализ в системата. Ето защо тази система трябва да може да обработва и анализира набори от данни, които се променят във времето.

Един от най-значимите проблеми, свързани с големите масиви от данни, е **анализът на данните**. Анализът трябва да бъде подпомогнат от високопроизводителни, разпределени или облачни изчисления, правилна декомпозиция на проблема и изчислителни модели за машинно обучение,

размити и невронни мрежи. Докато анализираме големите масиви от данни, можем да използваме методи за статистически анализ, визуализация на данните и други методи за проучвателен анализ на данните или прогнозен и оценъчен анализ на данните с помощта на подходи за машинно обучение, размита система за извод на невронни мрежи (виж Раздел 2).

1.2. ОБЩИ ПРОБЛЕМИ В НАБОРИТЕ ОТ ДАННИ

Съществуват някои често срещани проблеми, свързани с данните, които не бяха описани по-горе – конкретно в проблемите, свързани с големите данни.

Както беше споменато по-горе, обемът на данните непрекъснато нараства, което означава, че анализът на тези съхранявани и обработвани данни отнема повече време. Анализът обаче е същността на самото съхранение на данни и затова е невъзможно да бъде избегнат. Това води след себе си необходимостта от методи и процедури, които ни позволяват да придобиваме знания и да подпомагаме вземането на решения в контекста на голям набор от данни.

Наборите от данни, предназначени за нуждите на конкретни задачи, често се създават чрез комбиниране на данни от няколко източника. Тези източници могат да се характеризират с разнообразие във форматите и състава на отделните единици данни, поради което се нуждаем от начин за събиране и обединяване на такива разнообразни данни за нуждите на по-нататъшния анализ. С тази точка е свързан още един проблем – тъй като данните идват от различни източници, може да възникне ситуация, при която отделни записи ще си противоречат (или няма да са съвместими помежду си).

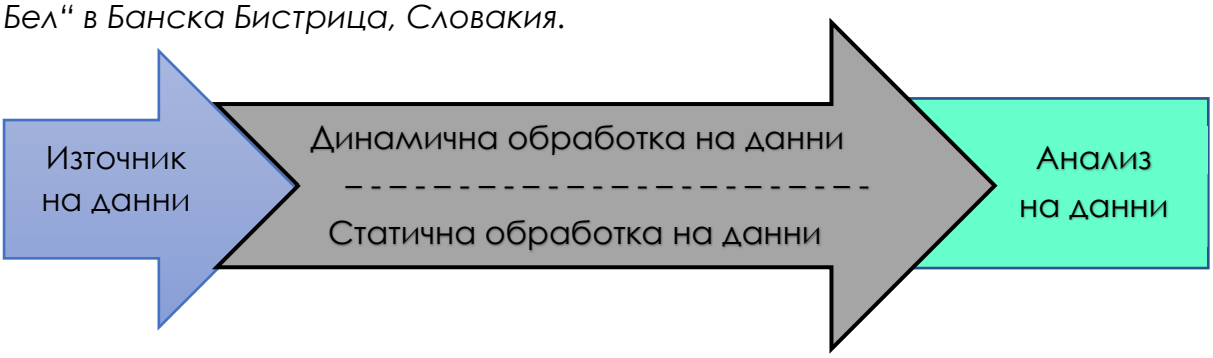
Сигурността на данните е голям проблем, въпреки че в контекста на този наръчник тя не е важна за нас. Не всеки източник на данни е сигурен и може дори да не е в съответствие с политиката на компанията, която иска да го използва. Като цяло е необходимо да се обърне внимание на създаването на оторизация и автентификация, наблюдението на потребителите, които работят с данни, сигурността на необработените и придобитите данни и защитата на комуникацията, т.е. предаването на данни.

И накрая, проблеми, които са особено важни в контекста на създаването на прогнози или оценки, са липсващите данни и отклоненията в данните. И двата проблема са естествени – в случая на липсващи стойности в данните има ситуации, при които една от необходимите измерени стойности в набора от данни липсва, а в случая на отклоненията това е естествена ситуация, при която някои измерени стойности се намират далеч извън тялото на набора от данни. Тези два проблема са основното съдържание на Раздел 2 от настоящия наръчник.

2

ОБРАБОТКА И АНАЛИЗ НА ДАННИ

Тази част от ръководството е написана от Адам Дудаш *от Департамента по компютърни науки, Факултет по природни науки, Университет „Матей Бел“ в Банска Бистрица, Словакия.*



При работа с данни, поне от гледна точка на този наръчник, можем да определим две основни дейности – обработка на данни и анализ на данни. Основната ни цел е да представим информация и примери за анализ на данни, който не може да бъде извършен ефективно без данни, които са подходящи като входни данни за този процес. Подобна подготовка на данните до форма, подходяща за безпроблемен анализ, се нарича обработка на данни. Необходимостта от обработка и анализ на данни произтича от няколко характеристики на съвременните данни:

- **Източници на данни** – В днешно време често работим с набори от данни, които са създадени чрез комбиниране на по-малки набори от данни, събрани от различни източници. Създадените по този начин набори от данни могат да произхождат от различни бази данни, от различни сензори в една мрежа, но също така и от комбинация от тези два подхода. Наборите от данни, които са съставени от по-малки части, носят със себе си съвсем естествени проблеми, свързани с този тип структура на данните:

  – хомогенизиране на структурата на данните;
  – работа с липсващи стойности;
  – работа с отклонения.

- **Статична обработка на данни** – в контекста на съвременните системи възприемаме два вида данни, които могат да се появят в дадената система или които можем да обработваме в системата – първият вид данни са статични данни. Статичните данни са данни, които не се променят с течение на времето. Често срещан подход

за обработка на такива данни е т. нар. пакетна обработка. По подразбиране тези партиди от задачи включват зареждане на файл, обработка на файла и записване на резултата в нов файл без ръчна намеса от страна на потребителя.

- **Динамична обработка на данни** – ако системата използва източници на данни от околната среда (например постоянно активен сензор или набор от сензори), тя трябва да може да улавя и съхранява тези данни във време, близко до реалното. Няма значение какъв тип памет се използва за това – тя трябва да поддържа мащабиране поради големия обем данни. Такива динамично променящи се набори от данни наричаме също потоци от данни, които трябва да можем да обработваме, филтрираме, агрегираме и по друг начин да подготвяме за анализ. Обработените по този начин данни след това се изпращат за анализ. Проблемът за динамичната обработка на данни е извън обхвата на този наръчник, но той е доста важна част от съвременните системи за данни.
- **Анализ на данни** – ние възприемаме анализа на данни като дейност по придобиване на знания за нуждите на по-доброто вземане на решения в контекста на избрана проблемна област, възможността за прогнозиране на стойности въз основа на събрани данни или оценка на неизмервани данни. Във втората част на този раздел описваме видовете анализ на данни и основните проблеми при анализа на данни.

Този раздел от наръчника е посветен на обработката на данни и избрани проблеми, свързани с това действие. Втората част на раздела е посветена на въведение в анализа на данни, което след това се разглежда в дълбочина в следващите раздели на представения наръчник.

2.1. ОБРАБОТКА НА ДАННИ

Невинаги имаме възможност да работим с набор от данни, който е готов за директен анализ на данните, често (особено когато става въпрос за нашите собствени данни) това са буквално събрани набори от информация. Ето защо е необходимо **данните да бъдат изчистени** и **форматирани**, преди да бъдат анализирани.

Бележка за този процес – обработката на данните и всички стъпки, описани в този раздел на текста, трябва винаги да се извършват върху копие на оригиналния набор от данни, а не върху самия набор от данни. Също така в идеалния случай трябва да използваме методи, които са систематични и повторяеми. В края на краищата не искаме да загубим трудно спечелените си данни.

Вътрешна съгласуваност на набора от данни

Както беше споменато в началото на този раздел от наръчника, фактът, че съвременните набори от данни се създават чрез комбиниране на няколко по-малки набори от данни, създава проблеми, свързани с вътрешната съгласуваност на самите набори от данни. Тази несъгласуваност може да се възприеме на две нива – несъгласуваност на самите данни и несъгласуваност на структурата на набора от данни.

По подразбиране няколко типични проблема могат да причинят непоследователност на данните:

- **Преобразуване на единици –** когато се комбинират два набора от данни, които използват различни единици за измерване на стойностите на атрибутите (например сантиметри и милиметри), е необходимо да се унифицира мерната единица. Необходимо е също така да се унифицират набори от данни, измерени на континенти, които не използват едни и същи мерни единици – например, за измерване на една и съща величина в Европа ще се използват сантиметри, а в САЩ – инчове.
- **Числени преобразувания** – Цифровите стойности, отбелязани описателно, трябва да се превърнат в числа. Тази област на необходими преобразувания включва и доста типични проблеми с определянето на единици в рамките на стойността на атрибута.
- **Конвертиране на имена** – при записването на имената на физически лица е необходимо да се уеднакви начинът на записване на имената и фамилните имена. Най-големият проблем в случай на набори от данни, използващи атрибути на имена от различни континенти, са знаците с ударение (напр. š, č, ä).
- **Преобразуване на дата и час** – в случай на анализи, съдържащи информация за времето, е необходимо да се унифицира форматът на записване на времето и особено на датата в дадения набор от данни.
- **Финансови и валутни превръщания** – стойностите на атрибутите, посочени в различни валути, трябва да бъдат унифицирани до една от валутите, които вече присъстват в набора от данни.

Вторият случай е **несъответствие на структурата на набора от данни**. Идеалният метод за съхраняване на данни за по-нататъшен анализ е методът, описан в раздел 1 на настоящия наръчник – съхраняване на данни под формата на таблица. Това обаче невинаги е възможно да се постигне по прост начин – проблемът в този случай ще бъде главно в липсващите стойности.

Набор от данни, който съдържа липсващи стойности, е проблематичен за анализиране с помощта на стандартни инструменти, но също така и с помощта на всякакви софтуерни инструменти. Клетките на въображаемата таблица, в които липсва стойност, се запълват с NULL стойности, които не могат да бъдат оценени статистически и в същото време не могат да се приемат като стойности (тъй като 0 ≠ NULL). Поради това е необходимо да се справим с този вид проблем по определен начин.

Липсващи и повредени данни

За целите на настоящия наръчник данните се разглеждат като измервания на реални свойства. Тези измервания се влияят от два фактора – инструмента за събиране на данни и метода за обработка на данните. И при двата фактора може да възникне проблем, чиято последица е **загуба или повреждане на данни**. Ако има проблем с инструмента за събиране на данни (изгоряла част от сензора, загубени записи след прекъсване на работата на сървъра и т.н.), говорим за загуба на данни, които не могат да бъдат възстановени. Обратното е загубата или повреждането на данни по време на тяхната обработка. Ако разполагаме с необработени данни, не е проблем да коригираме грешката – този вид загуба или повреда на данни наричаме **артефакт**.

Ако наборът ни от данни не е пълен, е необходимо да се идентифицират липсващите стойности и след това да се компенсират по подходящ начин. Проблемът е, че някои от липсващите стойности може дори да не съществуват, пример за това може да бъде стойност за атрибут, който съдържа часа на пристигане на определено място в ситуация, в която все още не сме пристигнали на това място.

Начините за работа с липсващи стойности, когато липсват първични данни, могат да бъдат разделени на няколко вида компенсации:

- **Замяна на липсваща стойност с друга стойност** (0 (-1) безсмислица) – при този подход всяка липсваща стойност (NULL) се заменя с избрана, специална стойност. Този подход не се препоръчва – заместващите стойности често могат да се приемат за правилни и ще бъдат неправилно интерпретирани при анализа на набора от данни. Ако например нямаме определена стойност за заплатата на служител, не я заменяме със стойност 0 или -1, тъй като служителят не работи безплатно или не плаща за това, че може да дойде на работа.
- **Премахване на непълни единици** – малко по-добър случай в сравнение с предишния би могъл да бъде подходът, при който премахваме всеки непълен запис от набора от данни. Този подход е добър, ако разполагаме с достатъчно данни, но все пак може да доведе до необективни резултати.

- **Изчисляване на липсващи стойности (приписване/импликация)** – в случай че трябва да използваме записи, които съдържат липсващи стойности, можем да изчислим тези стойности, като използваме един от методите по-долу. Наричаме този подход импликация на стойности.
    - Присвояване чрез евристичен подход – ако знаем достатъчно за набора от данни и връзките в него, би трябвало да можем да оценим стойността на някои атрибути.
    - Присвояване чрез средна стойност на атрибута – този метод замества липсващите стойности със средната стойност за дадения атрибут. Използването на такава стойност е изгодно по няколко причини, най-важната от които е, че средните стойности на атрибутите не са силни в нито една от двете посоки и поради това оказват слабо влияние върху прогностичния потенциал в набора от данни. Не винаги обаче е подходящо липсващите стойности да се заменят със средната стойност на дадения атрибут. За средна заплата този подход би бил подходящ, но за средна дата на пристигане на дадено място няма смисъл.
    - Присвояване чрез случайна стойност на атрибута – за липсващата стойност избираме случайна стойност на дадения атрибут, която сме записали в масива от данни.
    - Изчисляване чрез методи за машинно обучение – най-усъвършенстваният подход за изчисляване на липсващи данни е използването на методи за машинно обучение. Тези методи обаче не могат да се използват за всяка съвкупност от данни – или по-точно, не е възможно да се използват ефективно за всяка съвкупност от данни. Методите за машинно обучение работят въз основа на корелациите между отделните стойности в набора от данни и ако тези корелации са слаби или не съществуват, оценките на стойностите на набора от данни ще бъдат неточни. Този подход е описан по-подробно, като се започне от Раздел 4 на настоящия наръчник.

Изключения

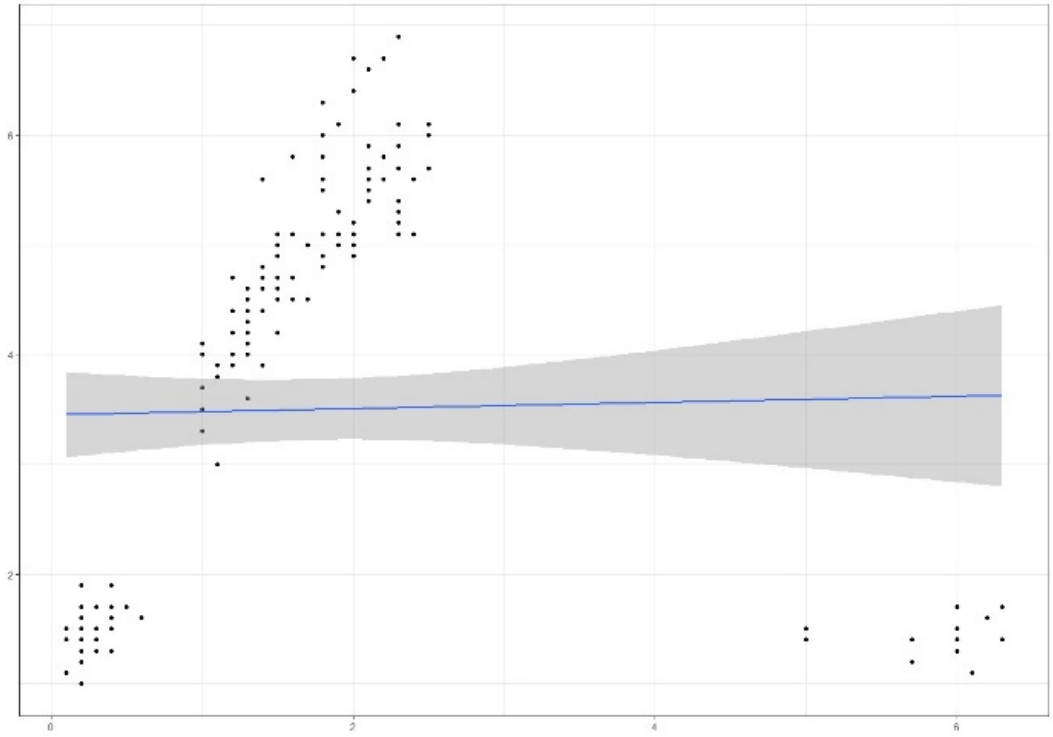
Изключенията са стойности, които се намират извън основната част на набора от данни. При нормално разпределена съвкупност от данни вероятността за поява на дадена стойност в съвкупността от данни намалява с разстоянието от средната стойност на дадената съвкупност от данни. Проблемът обаче възниква при набори от данни с необичайно разпределение.

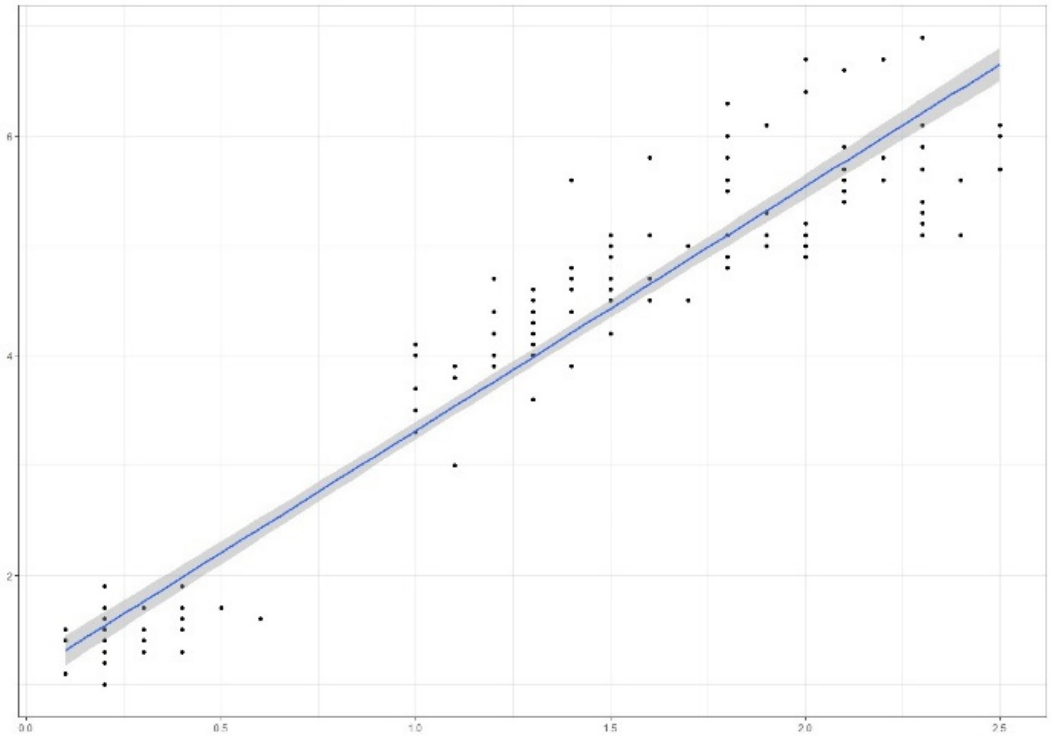
Отклоненията възникват по няколко начина:

- грешка при измерването,
- печатна грешка при обработката на данните,
- недостоверна информация, която може допълнително да посочи недостоверността на други записи.

Често обаче става дума за реална стойност, която се отклонява от стандартните ситуации (например периоди на замърсяване на въздуха), така че е необходимо да се анализира записът като цяло.

Проблемът с отклоненията възниква, когато се опитвате да правите **обобщения** въз основа на данни, които съдържат отклонения. На фигурата по-долу се вижда опит за описание на дадената съвкупност от данни с помощта на права линия. От лявата страна има набор от данни, съдържащ 162 записа, от които 12 са разположени значително извън тялото на набора от данни. В този случай виждаме, че синята линия, която би трябвало да минава през центъра на набора от данни, напълно го пропуска с изключение на една точка. Вдясно виждаме същата съвкупност от данни след отстраняване на дадените дванадесет отклонения. Резултатът от обобщаването е много по-задоволителен в този случай.





Ако искаме да направим някакво **обобщение** на масива от данни, отклоненията ще действат като смущаващ елемент и затова се препоръчва да не се вземат предвид такива стойности на атрибутите (и записите, които ги съдържат), дори ако са верни. Както може да се види на горната фигура – искаме да опишем набора от данни с помощта на линия (всъщност линейна функция). В случая на лявата подфигура линията се отклонява поради наличието на отклонения (долният десен ъгъл на разглежданото пространство). След премахването на тези отклонения можем да видим драстично увеличение на точността на това обобщение дясната подфигура).

2.2. АНАЛИЗ НА ДАННИ

Анализът на данни е действие, чиято цел е **да се получат полезни знания от данните**, за да се подпомогне вземането на информирани решения относно проблема, прогнозирането на събития и поведението на избрани обекти въз основа на обработените данни. Разпознаваме няколко вида анализ на данни, но в рамките на този наръчник ще се интересуваме само от три основни, **най-често използвани вида:**

- **Описателен и диагностичен анализ на данни** – най-простият (и същевременно най-често използваният) метод за анализ на набори от данни. Дескриптивният анализ се фокусира върху правенето на изводи (или получаването на знания) от данните. Той се използва най-често в контекста на описанието на набора от данни и измер-

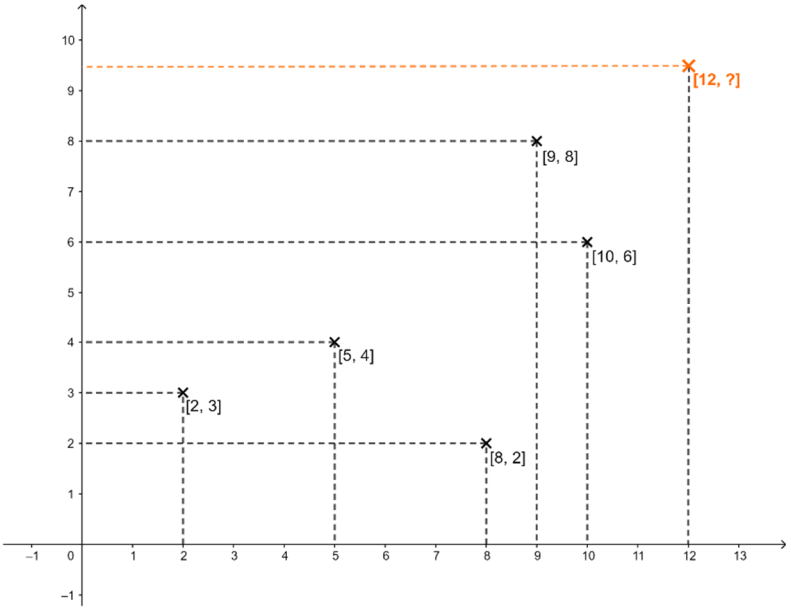
ването на основните свойства, които наборът от данни описва – например изпълнението на плановете в организацията. Диагностичният анализ е насочен към изясняване на причините за настъпване на събитията, идентифицирани при описателния анализ. Диагностичният анализ често се използва, тъй като създава връзки между данните и може да се използва за идентифициране на повтарящи се модели в поведението на обектите с данни. Този вид анализ на данни се основава на създаването на подробна информация, която впоследствие може да се използва многократно при решаването на подобни проблеми.

- **Проучвателен анализ на данни** – най-естественият вид анализ за хората е проучвателният анализ на данни. Той е насочен към анализ на данни чрез проучване, най-често с помощта на визуализация на данни. Този анализ е ефективен в контекста на идентифицирането на модели и зависимости в данните, но е важен и от гледна точка на представянето на резултатите от други анализи. В допълнение към визуалната страна на проучвателния анализ на данни, тук включваме и действия, свързани с опростяване на набора от данни или представяне на набора от данни – например намаляване на размерността, операция, при която проектираме n-измерен набор от данни в m-измерен набор от данни, докато $m < n$.

- **Прогнозен анализ на данни** – прогнозният анализ е продължение на гореспоменатите видове анализ. Неговата цел е да се използват събраните данни за създаване на логически прогнози за резултатите от събитията или за предсказване и оценка на стойности, които всъщност не сме измерили. При този вид анализ на данни се използват методи за моделиране, основани на статистиката, което води до необходимостта от използване на компютърни технологии за създаване на модели за прогнозиране. Имайте предвид, че прогнозите, които са резултат от моделите, създадени по време на прогностичния анализ, са само оценки за дадения набор от данни и следователно тяхната точност пряко зависи от качеството на дадените данни.

Всички тези видове анализ на данни обикновено работят само с два основни типа проблеми за решаване – регресионен проблем и проблем за класификация. Следващата част от този раздел е посветена на описанието на тези два проблема.

Проблем с регресията

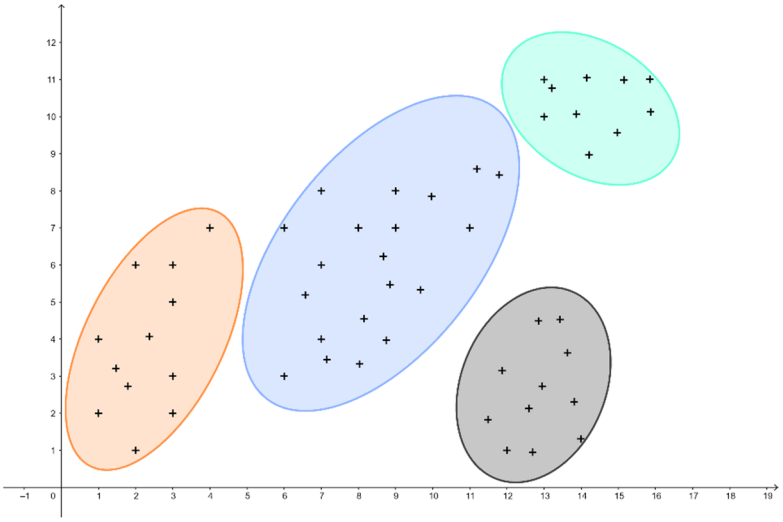


На фигурата по-горе виждаме набор от данни, който съдържа пет точки, определени от стойностите на два атрибута – тези стойности се измерват по осите x и y, затова ще ги наричаме стойности на атрибутите x и y. Тогава този набор от данни се състои от точки [2, 3], [5, 4], [8, 2], [9, 8] и [10, 6].

Регресионната задача в този случай ще бъде задачата да се оцени реалната стойност на признака $y \in R$, ако знаем стойността на x и модела от предишните (в този случай) пет точки. Следователно имаме съдържаща единица стойност за атрибута $x = 12$ и неизвестна стойност за атрибута y, която трябва да изчислим.

Най-общо можем да определим този тип задачи като оценяване или предсказване на числената стойност на променливата y въз основа на стойността на променливата x, където $x, y \in R$.

Проблем с класификацията



Общото описание на този проблем може да изглежда по следния начин: При даден модел *x* и пространството *X*, преценете коя стойност на свързания атрибут *y* ∈ {*1*,..., *n*} ще бъде придобита от модела *x*.

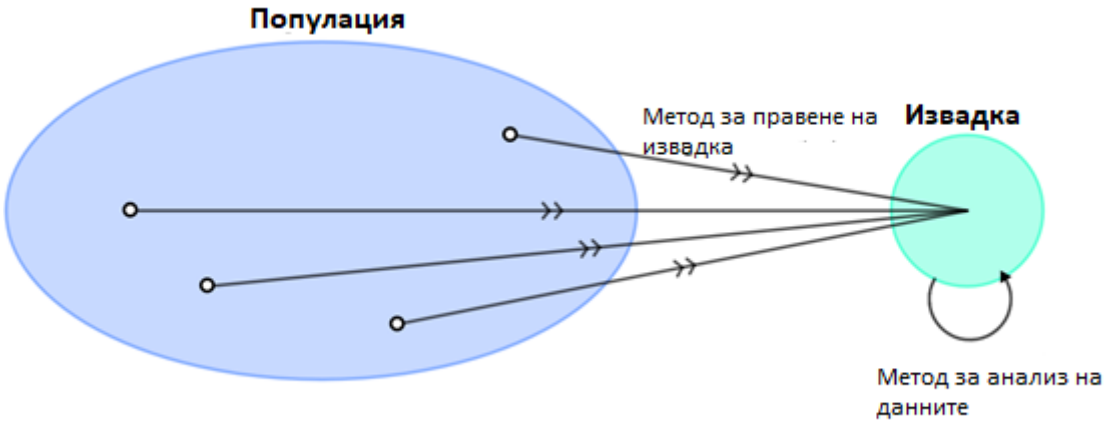
Подобно описание е малко неясно, по-разбираемо е, че в проблема за класификацията ние разпределяме същностите в набор от предварително определени класове от индивиди, които са сходни в определен смисъл в рамките на един клас. Като цяло можем да определим три вида процедури за класификация:

- **йерархична класификация,** при която самите класове се класифицират, като процесът се повтаря на различни нива, за да се образува дърво;
- **разделяне,** при което класовете са взаимно изключващи се, като по този начин се формира разделяне на множеството от същности;
- **групиране,** при което класовете или групите могат да се припокриват, а една група и нейното допълнение се разглеждат като различни видове класове.

3

МЕТОДИ ЗА ИЗГОТВЯНЕ НА ИЗВАДКИ ОТ ДАННИ

Тази част от ръководството е написана от Адам Дудаш *от Департамента по компютърни науки, Факултет по природни науки, Университет „Матей Бел“ в Банска Бистрица, Словакия.*



Извадката може да се определи като избор на част от популацията (набор от данни), която е най-представителна за нея, така че да може да се използва за анализ и получаване на знания за популацията. Техниката, използвана при вземането на проби от популацията, се нарича метод на извадката.

По този начин извадката се определя като част или **фракция от дадена популация**, избрана от популацията по такъв начин, че да могат да се направят изводи за популацията, докато популацията е съвкупността от всички изследвани лица или обекти.

Известни са няколко добре познати метода за вземане на проби. Най-често ги разделяме на **две групи** – вероятностни и не-вероятностни методи за вземане на проби. Важно е обаче да се каже, че видът, който ще се използва при подбора на извадката, зависи изцяло от решавания проблем. Най-общо обаче можем да кажем, че:

- **методите, които не са свързани с вероятността,** зависят от лицето, което съставя извадката, така че е много лесно да се получат резултати, които дадено лице би очаквало (дори те да не са верни за цялата популация).
- **вероятностните методи** повече или по-малко избягват този проблем.

3.1. Не-вероятностни методи за изготвяне на извадки

Подборът на извадка от населението зависи главно от преценката на лицето, което я съставя – поради това тези методи могат да доведат до изкривяване на някои стойности в сравнение с населението. Някои методи на невероятностна извадка дори зависят само от удобството на лицето, което съставя извадката – например методът на извадката, наречен „извадка по удобство“, при който членовете на популацията се избират въз основа на удобството на съставителя. По подобен начин, при метода, наречен извадка по преценка, извадката се съставя въз основа на преценката на съставителя – например въз основа на непознаването на данните от лицето, което съставя извадката.

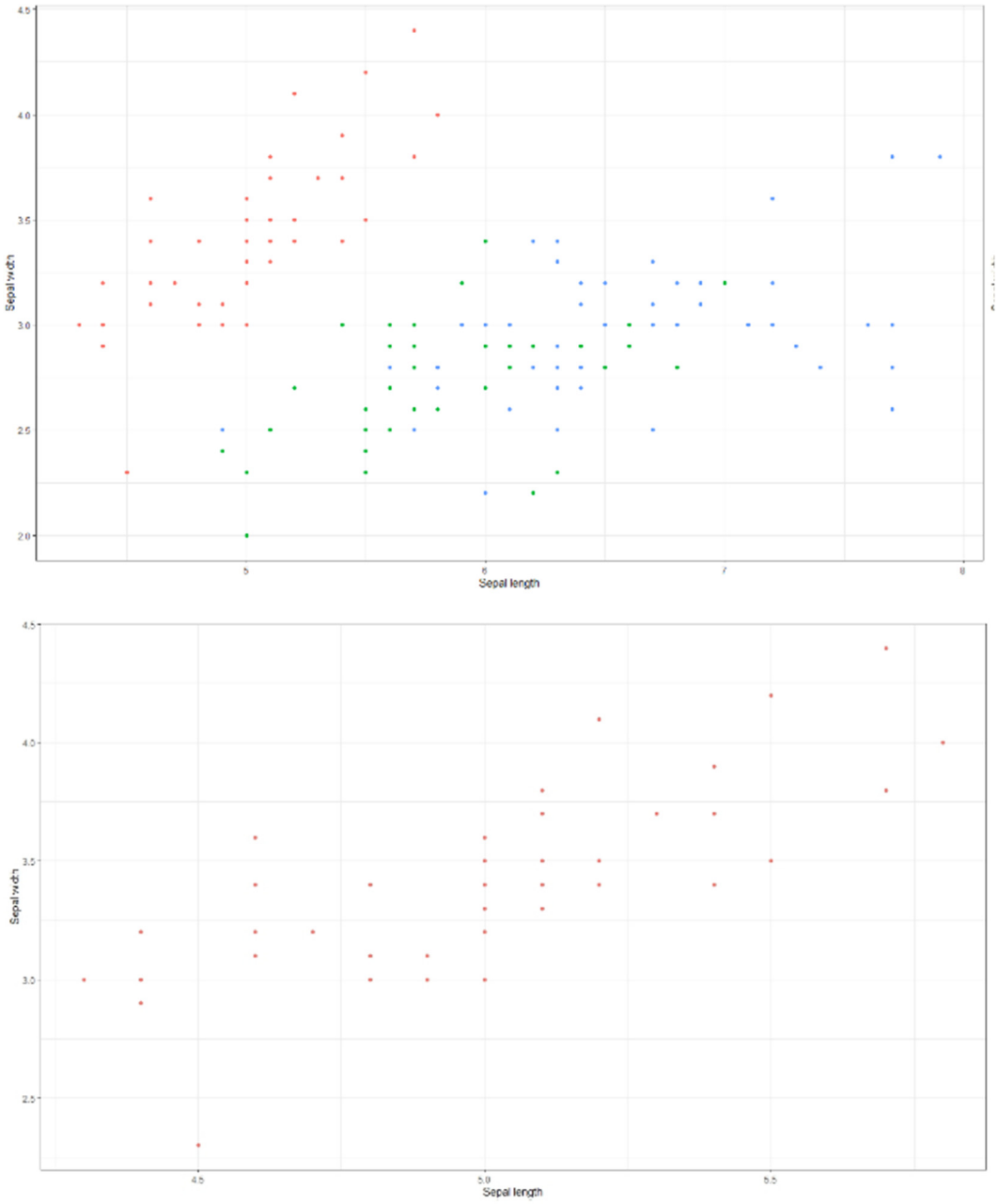
Редица методи за невероятностна извадка са много прости и представляват просто вид извадка на здравия разум. Ето защо в този наръчник предлагаме по-подробно описание само на един от методите за невероятностна извадка.

Метод на целева извадка

При този метод на извадка лицето, което прави извадката, избира членове на популацията с конкретна цел, за която се прави извадката. Тъй като всички членове на популацията нямат еднакъв шанс да бъдат включени в извадката, говорим за непроизводствен метод на извадката.

Пример за такава извадка може да бъде необходимостта от изготвяне на анализ на студентите от третата година на обучение в областта на компютърните науки, т.е. да се създаде извадка от студентите от третата година на обучение в областта на компютърните науки от популацията на всички студенти от всички години във всички области на обучение. Очевидно е, че няма да искаме да включим в извадката студенти от първи, втори, четвърти или пети курс. По същия начин няма да включим в извадката студенти, които се обучават в области като приложна математика, биология или съдебна химия.

За да опишем резултатите от отделните методи за вземане на проби от този момент на главата, ще използваме проби от набора от данни Iris, който е описан в Приложение А на настоящия наръчник. Задачата за метода на целенасочената извадка може да бъде следната: *Необходимо е да се анализират стойностите на дължината и ширината на чашката за един конкретен вид цвете – Ирис* (*Iris Setosa*).



На фигурата е представено сравнение между стойностите на дължината на чашката и ширината на чашката в а) пълния набор от данни за ириса на лявата подфигура (всеки клас на цвета на ириса е отбелязан със собствен цвят) и б) извадка от набора от данни, състояща се от един клас Ириси – Iris setosa.

3.2. Вероятностни методи за вземане на проби

С това наименование се обозначават методите, при които всички членове на разглежданата съвкупност имат равни шансове да бъдат избрани като част от извадката. Тези методи предотвратяват (или намаляват) отклонението от страна на лицето, което прави извадката, при добавянето на обекти към извадката, което беше споменато в раздела за невероят-

ностните методи. Съществуват различни видове вероятностни методи за подбор на извадки, които се използват в различни ситуации за подбор на извадки от различни популации.

Тези методи изискват от изследователя да познава разглежданата съвкупност и да знае какъв метод на извадка да използва и как да го използва във всяка възникнала ситуация.

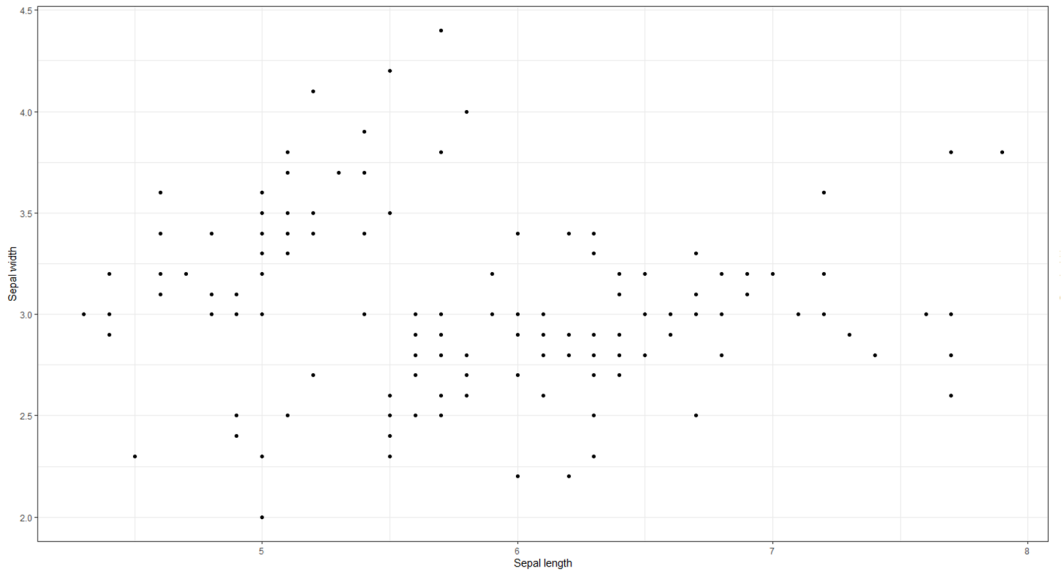
Съществуват редица вероятностни методи за вземане на проби – многоетапна извадка, клъстерна извадка, систематична извадка и др. Ще се спрем на четири метода за вероятностна извадка, които са прости и приложими в широк кръг от решавани задачи.

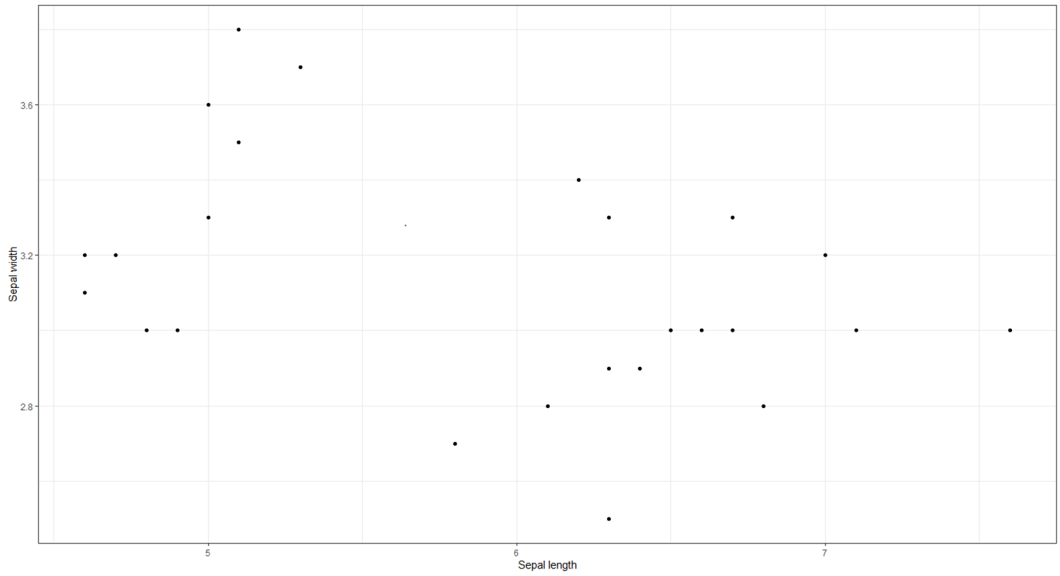
Метод на проста случайна извадка

Този метод се основава на случаен подбор на индивиди от популацията. С други думи, от всяка популация се избира определена извадка без математически модел или логическо решение. Тъй като всеки индивид (запис) има еднакъв шанс да стане част от извадката, този метод е най-представителният от вероятностните методи на извадката.

Методът на простата случайна извадка има само един входящ параметър – желания размер на извадката.

Пример: Нашата популация (вляво) съдържа 150 индивида (записи) и ние избираме на случаен принцип 25 представители от нея (вдясно) – тази съвкупност представлява проста случайна извадка за нас.

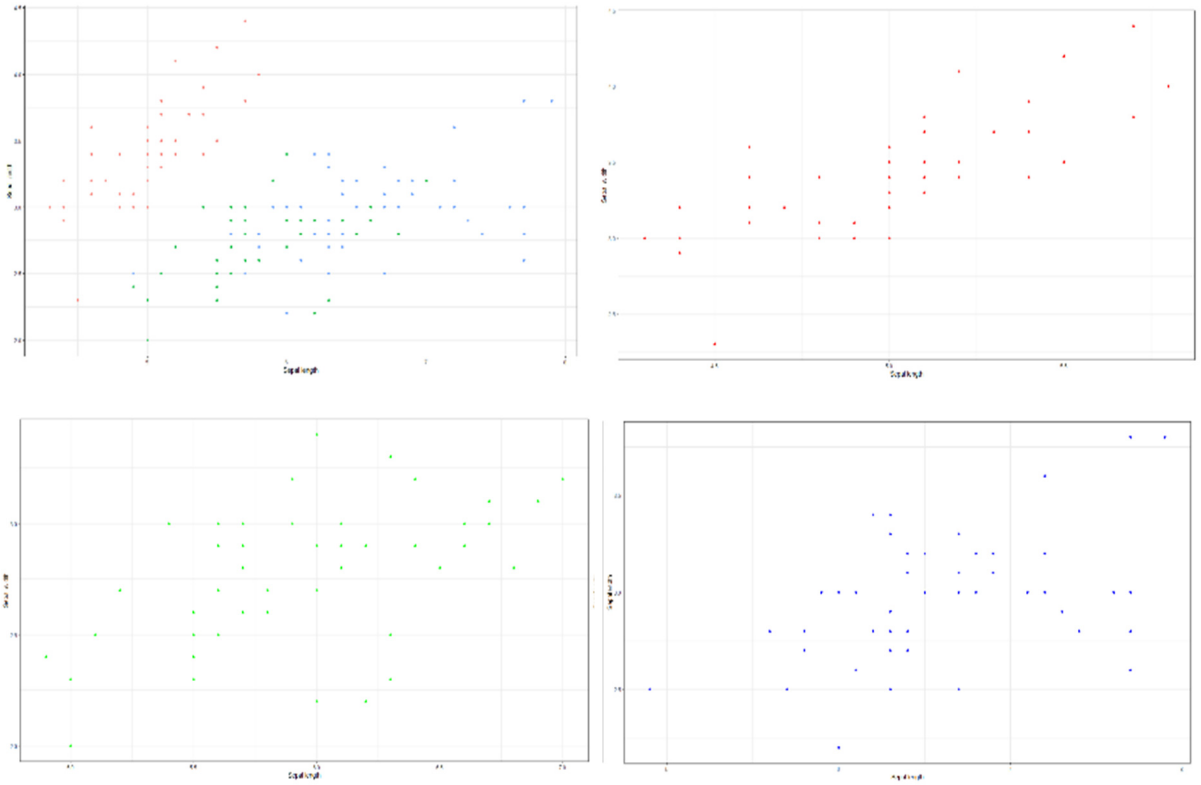




Метод на клъстерна извадка

Метод, при който целият набор от данни се разделя на части или клъстери. Клъстерите се идентифицират и включват в извадката въз основа на определен признак, като най-често този признак е категоричен, например цвят на косата, пол и т.н. Методът е приложим с цел създаване на извадки, подходящи за анализ на вече съществуващи подмножества в данните.

Методът на клъстерната извадка има само един входящ параметър – атрибут, който трябва да се използва за клъстеризиране на данните.



Пример: Разделихме нашия набор от данни (горния ред) според атрибута class, който приема три стойности – Iris setosa, Iris versicolor и Iris virginica. Като използваме метода на клъстерната извадка, можем да създадем три извадки (долният ред на фигурата), които могат да се използват, за да се анализират характеристиките на индивидите от дадените класове. Очевидно е, че извадка2 (долният, средният ред) не е подходяща за правене на заключения за цялата популация, а само за подмножеството на популацията, чийто атрибут на класа има същата стойност като извадка2. Подходящо използване на този метод е например изготвянето на статистически анализ, описващ отделните клъстери, който ще позволи сравняването на характеристиките на класовете цветя на Ириса.

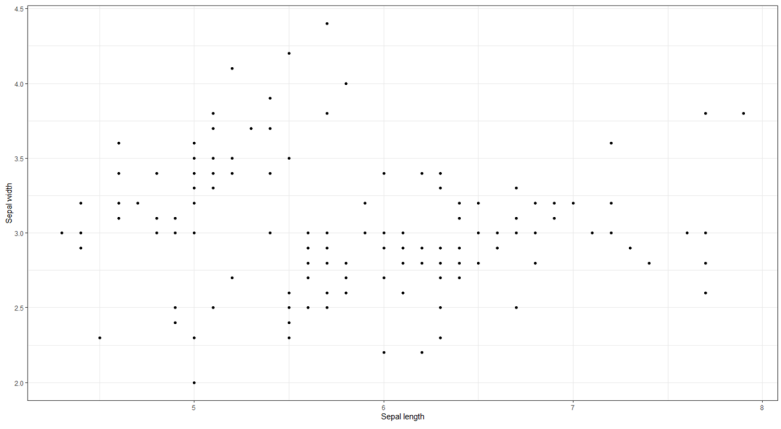
Систематичен метод за вземане на проби

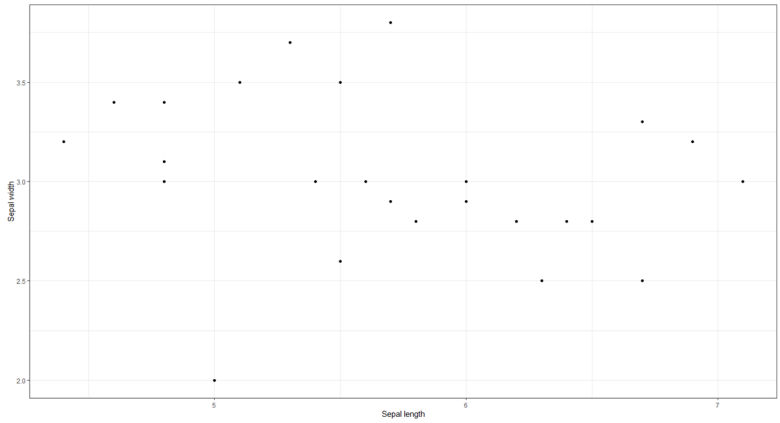
Този метод се използва за подбор на извадка от населението на равни интервали от време. Този вид метод на извадка има предварително определен обхват и поради това е най-малко трудоемката техника на извадка.

Методът на систематичната извадка има два или три входни параметъра:

- избрана начална точка за създаване на извадката (първото лице, което принадлежи към извадката);
- интервал, през който индивидите се добавят към извадката, което създава подразбиращ се размер на извадката,
- или интервал, през който се добавят индивиди към извадката, и размер на извадката, която създаваме.

Пример: Тъй като с метода на простата случайна извадка избрахме 25 лица, които представляваха използваната от нас популация, искаме да използваме и метода на систематичната извадка, за да създадем извадка от 25 лица. Първоначалната съвкупност се състои от 150 представители и тъй като 150/25 = 6, ще изберем всяко шесто лице (в случай че започнем от първия запис в съвкупността от данни). На фигурата по-долу можем да видим цялата съвкупност (вляво) и извадката (вдясно), състояща се от 25 индивида, които са избрани чрез описаната по-горе процедура.





Метод на стратифицирана извадка

При метода на стратифицираната извадка целият набор от данни се разделя на по-малки обособени групи, които представляват цялата популация. В сравнение с метода на клъстерната извадка този метод създава групи в данните, като използва новоопределена граница по един от атрибутите, присъстващи в първоначалната съвкупност от данни. Методът на клъстерната извадка не създава тези граници, а използва един от (категорийните) атрибути за определяне на групите в данните.

Пример: Извадките, създадени по метода на стратифицираната извадка на фигурата по-долу, могат да бъдат дефинирани като интервали, определени по атрибута „Дължина на чашелистчето“. Всяка извадка е различна, но във всяка извадка има представители, чиято стойност на дължината на сепарето е сходна от гледна точка на избрания метод. В нашия случай разделихме извадките на 1 cm от най-малката до най-голямата:

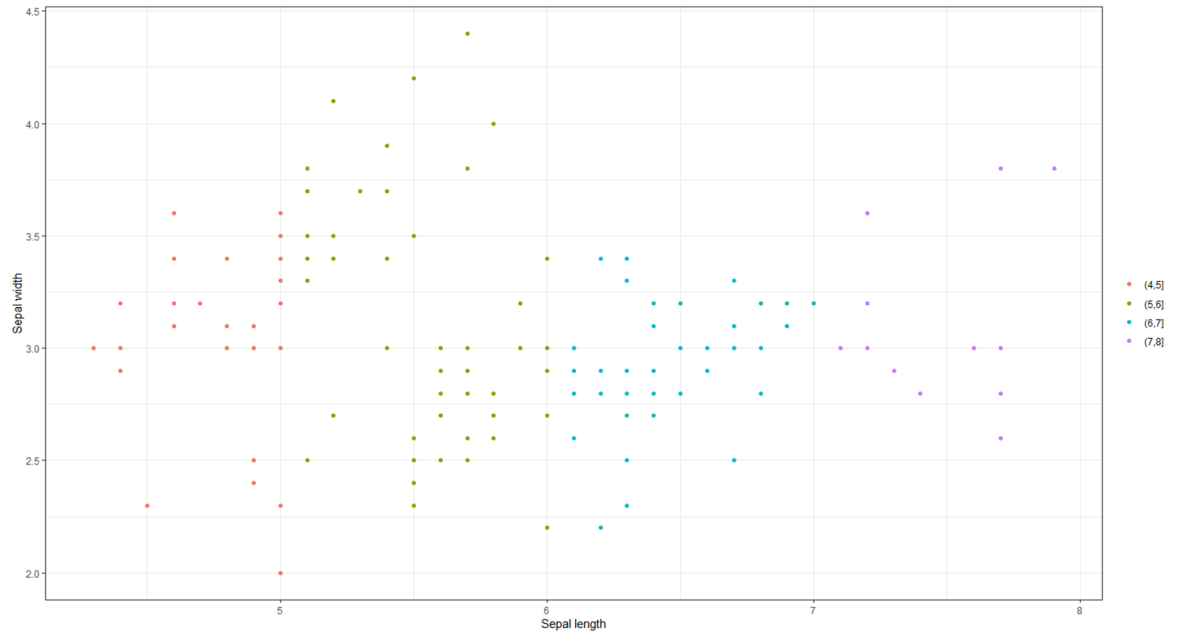
$$\text{дължина на чашелистчето} \in (4, 5]$$

$$\text{дължина на чашелистчето} \in (5, 6]$$

$$\text{дължина на чашелистчето} \in (6, 7]$$

$$\text{дължина на чашелистчето} \in (7, 8]$$

Така фигурата съдържа четири извадки, разделени по цвят – извадка1 е отбелязана в червено, извадка2 е отбелязана в зелено и т.н.



3.3. Няколко думи по темата за качеството на извадките

Правилното получаване на извадка е една от необходимите техники, използвани при работа с големи данни (виж Раздел 1.1 за справка). С гореспосочените методи се създават извадки, чието качество може да се оцени от няколко гледни точки. За целите на настоящия наръчник представяме само два критерия за описание на качеството на извадките, като само един от тях е наистина критичен за обикновените потребители (има се предвид предимно потребители извън областта на компютърните науки):

- **Бързина на събиране на извадките** – в съвременния свят използваме модерен софтуер, който често съдържа оптимизирани функции и пакети. Такива функции включват методи за вземане на проби, чиято реализация в избрания инструмент е преминала през оптимизации и методи, които са повишили ефективността на дадената функция (това е почти гарантирано). Ако е необходимо да се създаде извадка от стандартен голям набор от данни (не от типа big data), потребителят няма да се сблъска с проблема с недостатъчната производителност на системата, което би довело до продължително създаване на извадка (или, във фатални случаи, до невъзможност да се създаде извадка). Когато обаче работим с истински големи масиви от данни, стандартната система престава да бъде достатъчно ефективна. От собствения си опит можем да дадем пример за създаване на извадка върху набор от данни (популация) с размер сто милиона записа, като всеки запис съдържаше шестнадесет атрибута (имайте предвид, че това не е толкова голям набор от данни). Когато използвахме функцията за метода на систематичната извадка с входен параметър 4

(за създаване на извадка с размер 25% от популацията) в езика R на стандартен потребителски компютър, не успяхме да създадем извадка. Този проблем може да бъде решен по няколко начина, най-разпространеният от които е използването на високопроизводителни изчисления и методи за изчисления в облак.

- **Представителност на извадката** – въпрос, който е по-важен за оценката на качеството на извадката, отколкото скоростта на събиране на извадката, е качеството на нейната способност да описва популацията, от която е създадена. Както и в предишния случай, очевидно е, че подобен показател няма да бъде универсален – както беше посочено в описанието на метода на клъстерната извадка по-горе, извадка от един клъстер (една специфична подгрупа от данни) не е подходяща за изготвяне на заключение за цялата популация. В случаите, когато е подходящо да се сравнят характеристиките на извадката и на популацията, можем да процедираме по няколко начина, в зависимост от целите си:
  - **Статистическо описание на извадката –** когато искаме да опишем данни с малък брой стойности, можем да изчислим критични статистически показатели. От тези числени стойности можем да извлечем знания, подходящи за по-нататъшна работа с данните.
  - **Визуализация на извадката –** големите данни са известни със сложността на визуализацията си, поради което е добре да се създаде представителна извадка, която съдържа по-малко елементи и по този начин се визуализира по-лесно.
  - **Анализ на прогностичния потенциал на извадката –** ако целта ни е да изградим модели за прогнозиране или оценяване, основани на машинно обучение, е подходящо да анализираме прогностичния потенциал на отделните атрибути, като използваме методи като корелационен анализ или използваме модели на дървета на решенията.

Всички тези подходи са описани по-подробно в раздел 4 на настоящия наръчник.

3.4. Няколко думи по темата за размера на извадката

В случай че трябва да съставим извадка от дадена популация, може да възникне проблемът **какъв трябва да бъде размерът на тази извадка, за да се постигнат желаните резултати** – за да можем точно да извлечем

необходимите ни знания. Отговорът на този въпрос зависи от използвания метод и от нашите цели:

- В случай на методи за вземане на проби, като например клъстерна или стратифицирана извадка, отговорът се дава от **използвания метод**. При тези методи се създават извадки, чийто размер се определя от появата на определена стойност в данните и следователно в този случай не е стандартно да се разглежда размер на извадката, различен от размера на клъстера, определен от метода.

- В други случаи, особено при случайната извадка, е необходимо да се използва **модел, който определя размера на извадката, подходящ за нашите нужди**. Този модел е определен по подразбиране за два вида популации – популации с ограничен брой елементи или популация без ограничение на броя на елементите. За нашите нужди ще разгледаме по-естествения от тези варианти – популация с ограничен брой елементи:

$$\bar{n} = \frac{\frac{z^2 \bar{p}(1-\bar{p})}{\varepsilon^2}}{1 + \frac{z^2 \bar{p}(1-\bar{p})}{\varepsilon^2 N}}$$

където

  - $\bar{n}$ е размерът на извадката,
  - z е т.нар. коефициент на доверителност, който отразява нивото на доверителност, най-често 90%, 95% или 99%, като стийностите на коефициента на доверителност са представени в следната таблица:

Ниво на доверителност	z-стойност
90%	1.65
95%	1.96
99%	2.58

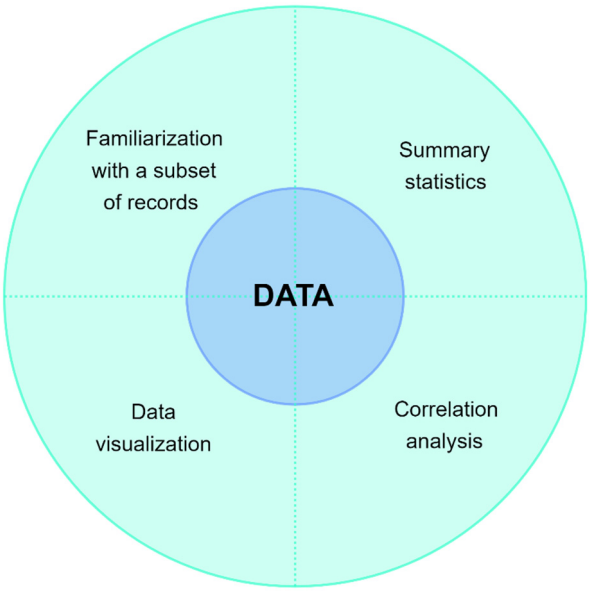
    Тази таблица е типичен пример за таблици със z-стойности, които съдържат предварително изчислени стойности на z-коефициента и могат да бъдат търсени онлайн за други стойности на доверителните нива.

  - $\bar{p}$ е дял от популацията – процент (или част) от популацията, свързан с изследвания проблем (стандартната стойност за неизвестна популация е p = 0.5).
  - ε е допустимата грешка, зададена от потребителя.
  - *N* е размерът (обемът) на използваната популация.

Най-лесният начин да се изчисли размерът на извадката е да се използват свободно достъпни онлайн калкулатори за изчисляване на размера на извадката, които работят на принципите, изброени по-горе.

4 ОСНОВИ НА ИЗСЛЕДОВАТЕЛСКИЯ АНАЛИЗ НА ДАННИ

Тази част от ръководството е написана от Адам Дудаш *от Департамента по компютърни науки, Факултет по природни науки, Университет „Матей Бел“ в Банска Бистрица, Словакия.*



Както беше споменато в предишните раздели на този наръчник, в процеса на анализ на данните можем да работим с цялата съвкупност от данни или с извадка, създадена с помощта на методите, посочени в раздел 3 на този текст. За основния, описателен анализ на данни, използваме методите на описателната статистика, които предоставят инструменти за улавяне на характеристиките на набор или извадка от данни. Методите на описателната статистика се основават на методи за агрегиране за представяне на подмножества от данни, например средна стойност на атрибута, минимум, честота или сума на стойностите. Такива методи могат да бъдат наречени агрегиращи като методи за редуциране на данни.

Когато анализираме данните с помощта на методите на описателната статистика, използваме основно следните три понятия:

- **Мерки за централна тенденция,** с които се търсят центрове на данни, около които данните са групирани или разпределени.
- **Мерките за вариабилност** са мерки, които описват разпределението на данните в разглежданото пространство, т.е. колко далеч са отделните измервания от центъра, определен с помощта на мерките за централна тенденция.

- **Корелационният анализ** се основава на изчисляването на коефициенти, които описват потенциала за прогнозиране между отделните атрибути в набора от данни. Тези коефициенти са от съществено значение при създаването на модели за машинно обучение, но също така и при визуализирането на данни, което е съществена част от този раздел на наръчника.

В този раздел на наръчника ще се сблъскаме с първата версия на анализа на данни, която е много естествена от човешка гледна точка – **Проучвателен анализ на данни (ПАД).** Както подсказва името, това е анализ на данни, при който се използва проучване на данни с цел да се открият модели и тенденции в дадена популация или извадка. В най-основната си форма този вид анализ се извършва чрез визуално изследване и затова методите за визуализация на данни ще бъдат важна част от такъв анализ.

За да разберем кои части от нашата съвкупност от данни са подходящи за визуализиране и кои не, използваме основните знания, придобити чрез методите на описателната статистика.

4.1. Основни статистически методи

От гледна точка на основните статистически методи можем да определим методите, с които измерваме „централността“ в данните – търсим центрове на данни, около които данните са групирани или гъсто разпределени. Най-разпространените от тези методи са:

- **Средното** е средната стойност на елементите на масива. Средната стойност е подходяща за характеризиране на симетрично разпределени данни без отклонения (напр. височина, тегло). Симетрично разпределени данни са тези, при които броят на елементите на масива от данни трябва да е сходен преди и след средната граница, в идеалния случай еднакъв. Формулата за изчисляване на средната стойност е следната:

$$\mu_A = \frac{\sum_{i=1}^{n} Ai}{n}$$

  където μ_A е средната стойност на атрибута *A*, *n* е броят на субектите, съдържащи атрибута *A*, а *Ai* е *i*-тата стойност на този атрибут.

- **Медианата** е стойност, разположена в средата на подредения масив. Медианата е изключително симетрична, което означава, че преди нея има същия брой елементи, както и след нея. Изключение от това правило са масивите от данни, които съдържат четен брой елементи (тогава избираме една от двете по средата стойности като медиана – в разумен масив от данни те трябва да са доста близки една до друга). За разлика от средната стойност, медиа-

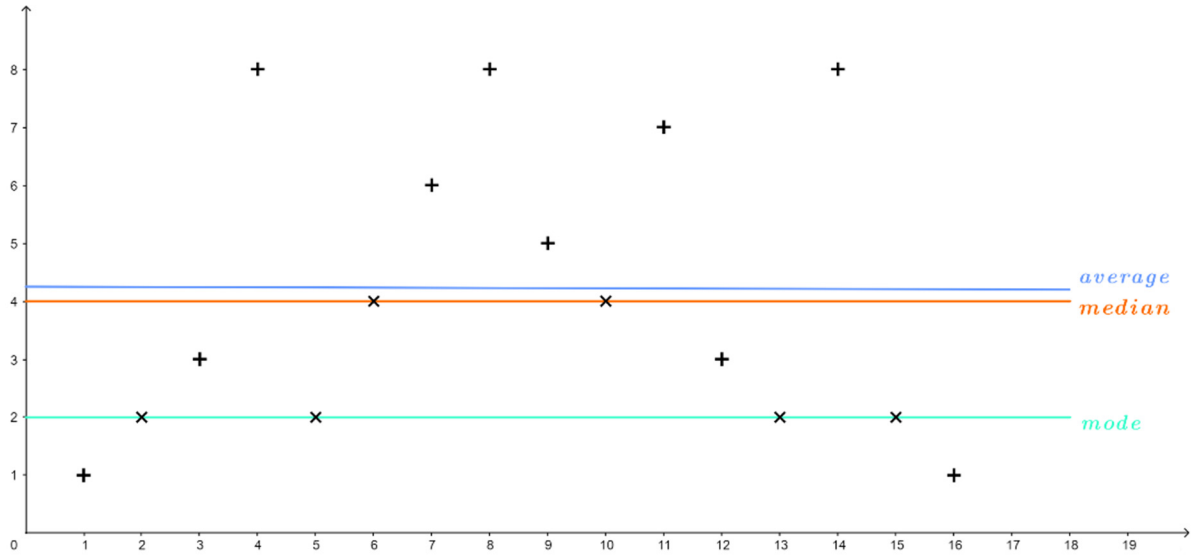
ната е реална стойност на даден атрибут, така че е по-подходяща, ако данните съдържат отклонения и са асиметрично разпределени (например заплатите на служителите в определена област). Медианата се изчислява с помощта на следната зависимост:

$$median_A = \frac{(n+1)}{2} \textit{ ти елемент на подредения масив } A$$

където *n* е броят на субектите, съдържащи атрибута *A*.

- **Модата** е най-често срещаният елемент в атрибута. Тази мярка обаче е трудна за използване и не е точна при повечето аналитични задачи. Пример за това може да бъде режимът за посочените заплати, който в повечето случаи ще бъде равен на 0, тъй като точно 0 получават повечето хора – безработни, деца, пенсионери.

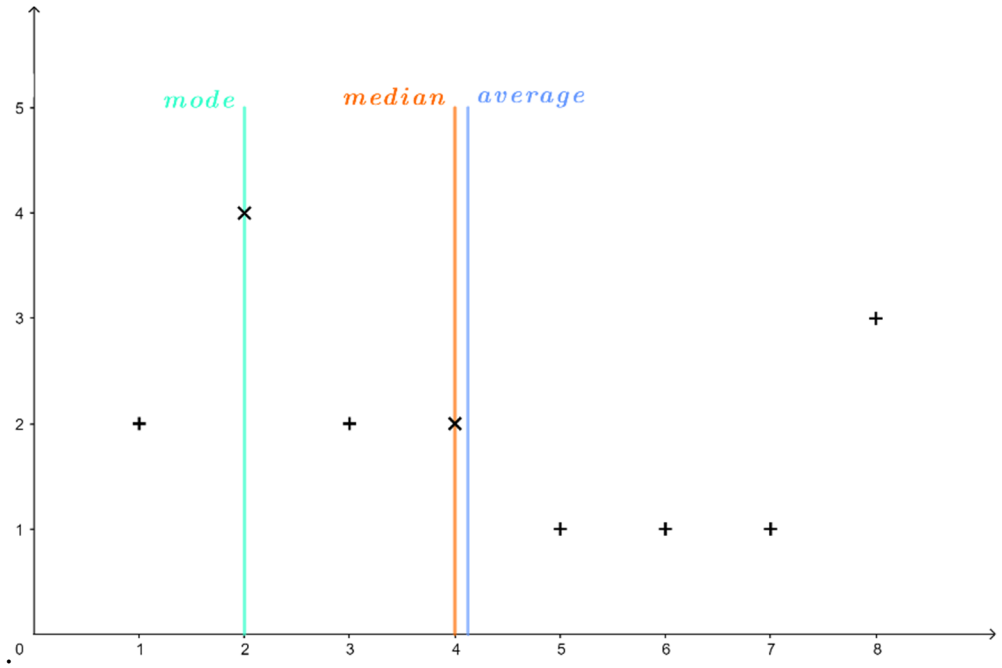
На следващата фигура предлагаме визуализация на мерките за централност за обикновен набор от данни. Можем да видим доста типично поведение на средните и медианните стойности при данни, които са нормално разпределени в разглежданото пространство – т.е. тези стойности са доста близки една до друга. Стойността на модата е трудно предвидима, тъй като тя е най-често срещаната стойност на елемента на атрибута (следователно може да бъде висока, може да бъде ниска, може да бъде някъде по средата).



В допълнение към тези стандартни мерки **честотата на стойностите** на даден атрибут е от голямо значение. Под названието **честотно разпределение** разбираме списък, таблица или графика, които показват честотата на поява на различни изходи в извадка (набор от данни). Всеки запис в таблицата съдържа честотата (или броя) на срещане на стойности в дадена група или интервал. Пример за такова честотно разпределение за простата съвкупност от данни, използвана по-горе, изглежда по следния начин:

стойност	честота
1	2
2	4
3	2
4	2
5	1
6	1
7	1
8	3

Тази таблица може да се съпостави с **графиката** по-долу. Подобна визуализация на графиките на честотите е от съществено значение, особено от гледна точка на опознаването на данните и евентуалното откриване на отклонения



Другата страна на монетата при стандартните статистически мерки е разсейването на данните в разглежданото пространство. Най-разпространената мярка за разсейване е така нареченото **стандартно отклонение** (σ), което се определя като сума от квадратичните разлики между отделните елементи на признака и е средната стойност:

$$\sigma_A = \frac{\sqrt{\sum_{i=1}^{n}(A_i - \mu_A)^2}}{n-1}$$

където μ_A е средната стойност на атрибута A, n е броят на елементите, съдържащи атрибута A, а A_i е i-тата стойност на този атрибут.

Мярка, подобна на стандартното отклонение, е **дисперсията**, изчислена като:

$$V = \sigma^2$$

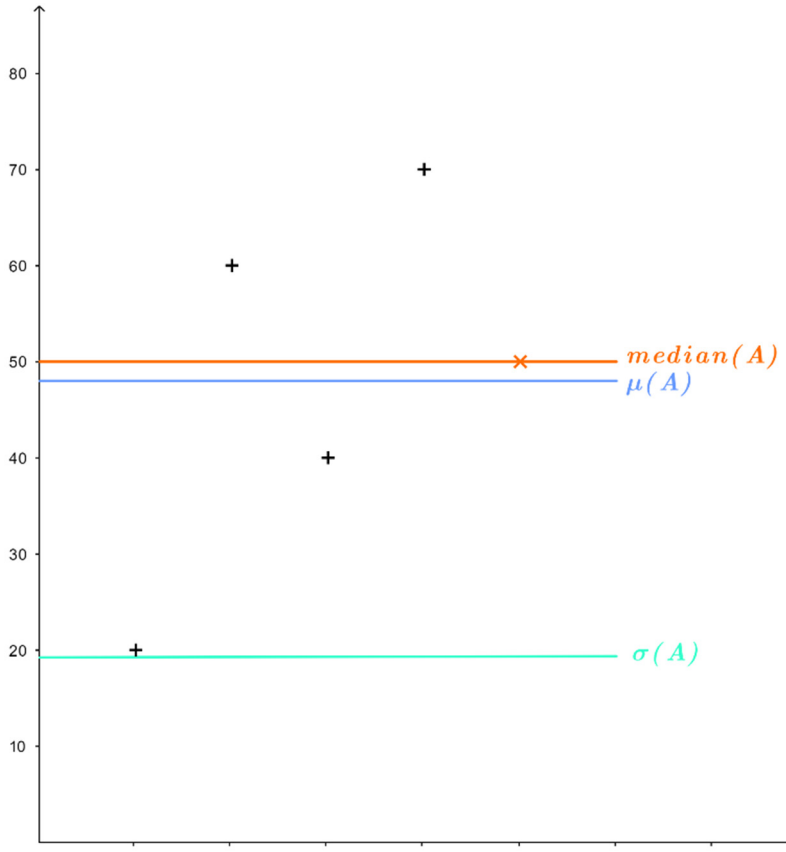
Пример: Нека имаме следния прост набор от данни, състоящ се от един атрибут с пет измервания A = [20, 60, 40, 70, 50]. Нека да изчислим средната стойност, медианата и стандартното отклонение за тази съвкупност от данни.

$$\mu_A = \frac{\sum_{i=1}^{n} Ai}{n} = \frac{20 + 60 + 40 + 70 + 50}{5} = \frac{240}{5} = \mathbf{48}$$

$$median_A = \frac{(n+1)}{2} = \frac{6}{2} = 3\,\textit{ти елемент от подредения набор}$$
$$\rightarrow [20, 40, 50, 60, 70] = \mathbf{50}$$

$$\sigma_A = \frac{\sqrt{\sum_{i=1}^{n}(A_i - \mu_A)^2}}{n-1}$$
$$= \frac{\sqrt{(20-48)^2 + (60-48)^2 + (40-48)^2 + (70-48)^2 + (50-48)^2}}{4} = \frac{\sqrt{1480}}{4}$$
$$= \sqrt{370} \approx \mathbf{19.235}$$

Това стандартно отклонение е сравнително високо, което е естествено, тъй като самият набор от данни е много разсеян. Визуализацията на тези измервания е показана на фигурата по-долу.



Методите за изчисляване на централността и разсейването служат за описание на набора от данни с помощта на малък брой стойности. Такъв подход за описание на набор от данни може да се нарече и **описание чрез обединяване**.

Пример: Нека имаме атрибут A = [1, 2, 3, 8, 2, 4, 6, 8, 5, 4, 7, 3, 2, 8, 2, 1] (представен в началото на Раздел 4.1). Можем да опишем тази съвкупност от данни с помощта на три обобщени стойности, например (min(A), μ(A), max(A)), следователно A = (1, 4.125, 8).

Последният показател от простите статистически показатели е **разпределението на набора данни в пространството**. Тази метрика се характеризира с използването на средната стойност на атрибута и стандартното отклонение. При нормално разпределена съвкупност от данни поне $(1 - 1/k^2)$-та от точките лежат на разстояние kσ или по-малко от средната стойност. Такава съвкупност от данни не съдържа отклонения и е отличен кандидат за методите за машинно обучение и анализ на данни с помощта на изкуствен интелект.

Пример: За познатия ни атрибут A = [20, 60, 40, 70, 50] можем да изчислим разпределението по следния начин:

$\mu_A = 48$

$\sigma_A \approx 19.235$

$2\sigma = 2 * 19.235 \approx 38.47$

Виждаме, че поне 3 от стойностите трябва да са отдалечени на наймного 38.47 единици от средната стойност (48). Това е вярно за всички измервания на атрибута A.

4.2. Корелационен анализ

Основните статистически стойности, описани в предходния раздел, са важни показатели, с които можем да опишем дадените данни. От гледна точка на целите на анализа на данните обаче така наречения корелационен анализ е много по-силен показател.

В случай че нашата съвкупност от данни съдържа повече от един цифров атрибут, можем да измерим корелацията между подмножествата от два елемента на тази съвкупност от данни. Нека имаме два атрибута от набор от данни A – A_1, A_2. Тези атрибути корелират помежду си, когато атрибутът A_1 има **прогностичен потенциал** за атрибута A_2. Такъв прогностичен потенциал говори за **наличието на тенденции и модели** в набора от данни и за възможността за изграждане на аналитични модели, които работят с данните.

Измерваме корелацията на две променливи, като използваме **коефициента на корелация** $r(A_1, A_2)$, който показва доколко признакът A_1 е функция на признака A_2 и обратно. Този коефициент на корелация може да приема стойности от интервала [-1, 1], като:

- **1** показва **пълна корелация** на два атрибута, с други думи, когато стойността на атрибута A_1 се увеличава, стойността на атрибута A_2 също се увеличава. Ако е налице пълна корелация между стойностите на две променливи, говорим за силен прогностичен потенциал и следователно тези атрибути са подходящи за взаимно прогнозиране.
- **0** показва най-лошата ситуация от гледна точка на корелацията на две стойности, която наричаме **липса на корелация.** В случай че коефициентът на корелация между два атрибута е близък или равен на 0, това са независими стойности, които са неизползваеми от гледна точка на изграждането на аналитични модели.
- **-1** е обратното на пълната корелация, която наричаме **антикорелация.** В този случай можем да определим тенденция, при която с увеличаване на стойността на атрибута A_1 стойността на атрибута A_2 намалява или обратно. Както и в случая на пълната корелация, това е задоволително условие за изграждане на аналитични модели.

Използваме два стандартни метода за анализ на корелациите и измерване на коефициентите на корелация – разбира се, има много повече такива методи. За нашите цели ще се съсредоточим върху коефициента на корелация на Пирсън и коефициента на рангова корелация на Спирман.

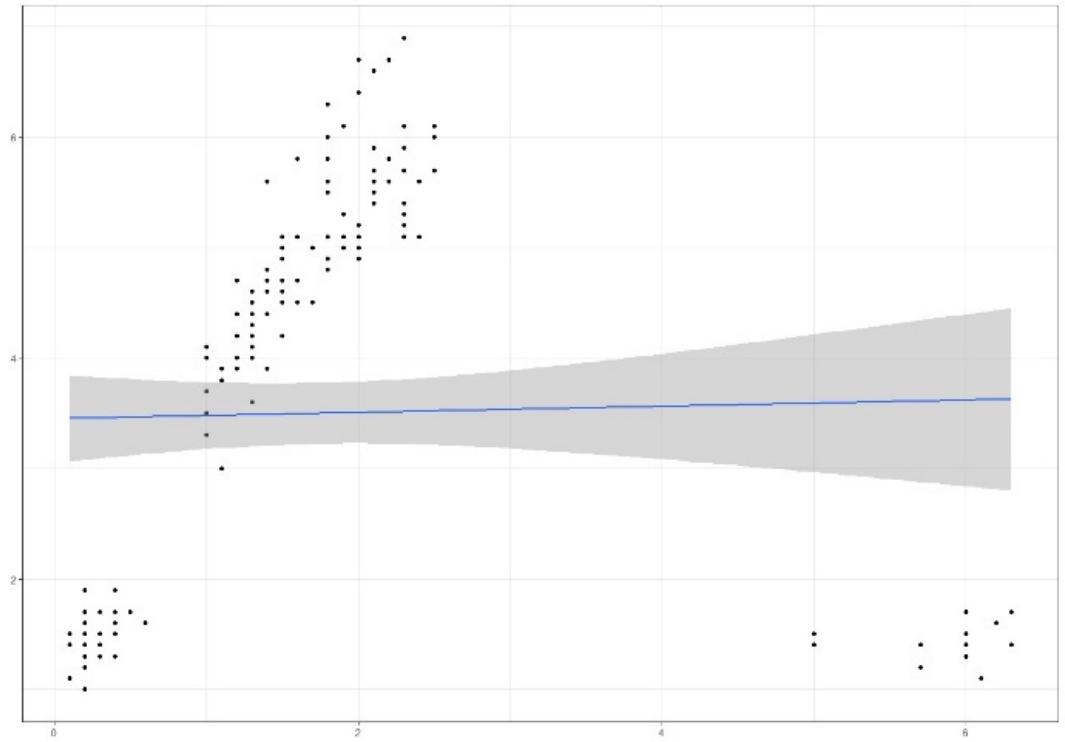
Коефициент на корелация на Пирсън

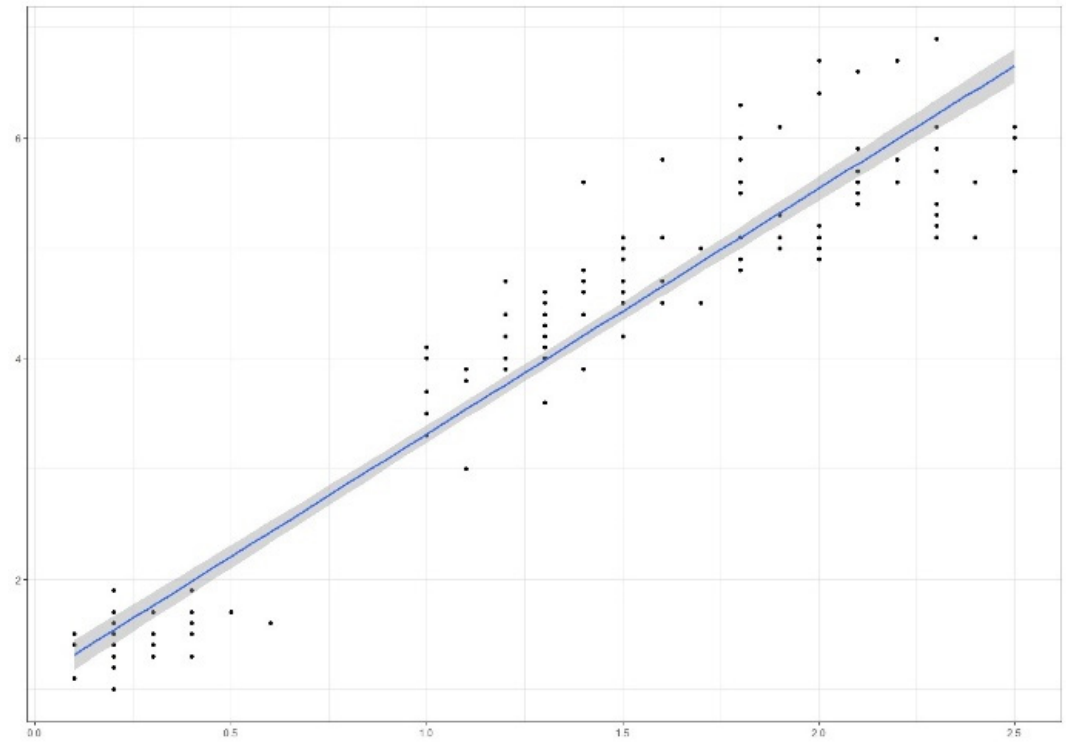
Първият и най-силен коефициент, който се използва за измерване на корелацията между два атрибута на набора от данни, е коефициентът на корелация на Пирсън. Този коефициент е насочен към **линейно предсказване на стойности** и за връзката между атрибутите *A* и *B* се описва със следната зависимост:

$$r = \frac{\sum_{i=1}^{n}(A_i - \mu(A))(B_i - \mu(B))}{\sqrt{\sum_{i=1}^{n}(A_i - \mu(A))^2}\sqrt{\sum_{i=0}^{n}(B_i - \mu(B))^2}}$$

където $\mu(A)$ е средната стойност на атрибута *A*, аналогично $\mu(B)$ е средната стойност на атрибута *B*, *n* е броят на измерванията (вертикален размер на набора от данни).

Тази очевидна зависимост от средната стойност води до най-големия недостатък на коефициента на корелация на Пирсън – чувствителност към отклонения (както е показано на фигурата по-горе).





Използвайки коефициента на корелация на Пирсън, търсим линия, която описва стойностите на дадените атрибути. На изображението вляво виждаме визуализация на сравнението на стойностите на два атрибута от една съвкупност от данни, в която има отклонения (долу, вдясно). Виждаме също, че линията, с която сме напаснали данните, ги пропуска напълно и значително (с изключение на една точка от данни) – от това можем да заключим, че коефициентът на корелация на Пирсън не е подходящ за измер-

ване на прогностичния потенциал в такава съвкупност от данни. В дясната част представяме същите атрибути на набора от данни след премахване на отклоненията. Виждаме, че в този случай линията е описателна за тенденциите, присъстващи в данните.

Следователно **коефициентът на корелация на Пирсън може да се използва, когато** атрибут *A* и *B* съдържат:

- линейни връзки,
- нормално (Гаусово) разпределение,
- няма отклонения.

Коефициент на корелация на Спирман

Като начин за справяне с **набори от данни, които съдържат нелинейни връзки** с отклонения, можем да използваме друг вид коефициент на корелация – по-конкретно коефициента на рангова корелация на Спирман. Този метод за измерване на корелацията между атрибутите създава **йерархия** (ранг) на стойностите на отделните атрибути за своята функционалност.

Пример: Нека имаме атрибут A = [a_0 = 4, a_1 = 8, a_2 = 2, a_3 = 6]. Горе-споменатата йерархия или класиране ще изглежда по следния начин:

Тъй като $a_1 > a_3 > a_0 > a_2$ ранг(a_1) = 1, ранг(a_2) = 4 и т. н.

По този начин измерваме **монотонността на стойностите** в рамките на атрибута и следователно можем да кажем, че ранговият корелационен коефициент на Спирмън е най-подходящ за набори от данни с монотонни връзки между атрибутите – ако стойността на единия атрибут се увеличава, стойността на другия никога не намалява или обратното. От друга страна, не се препоръчва използването на този вид корелационен коефициент, ако в набора от данни има повтарящи се стойности (което означава един и същ ранг). Този ефект отслабва с увеличаване на размера на набора от данни.

Коефициентът на рангова корелация на Спирман се изчислява по следния начин:

$$\rho = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n^2-1)}$$

където d = ранг(a_i) − ранг(b_i) и n е броят на субектите в разглежданите атрибути.

Пример: Предупреждение – следният пример е популярен сред учениците, те могат да запомнят примера повече, отколкото самите идеи за корелация. Нека имаме два атрибута – цена на кебап и разстояние на заведението за кебап от университета.

Индекс	*Разстояние в метри*	*Цена в евро*
1	*10*	*4*
2	*70*	*3.50*
3	*85*	*3.30*
4	*100*	*3.20*
5	*130*	*3.80*
6	*195*	*2.90*
7	*215*	*3.10*
8	*300*	*3.90*
9	*420*	*3.15*
10	*505*	*3*

Първо, изчисляваме стойността на коефициента на корелация на Спирман. Този метод изисква създаване на ранг за двата атрибута и изчисляване на стойностите на d и d^2, както следва:

Индекс	*Разстояние в метри*	*Ранг (разстояние)*	*Цена в евро*	*Ранг (цена)*	*d*	*d^2*
1	*10*	*10*	*4*	*1*	*9*	*81*
2	*70*	*9*	*3.50*	*4*	*5*	*25*
3	*85*	*8*	*3.30*	*5*	*3*	*9*
4	*100*	*7*	*3.20*	*6*	*1*	*1*
5	*130*	*6*	*3.80*	*3*	*3*	*9*
6	*195*	*5*	*2.90*	*10*	*-5*	*25*
7	*215*	*4*	*3.10*	*8*	*-4*	*16*
8	*300*	*3*	*3.90*	*2*	*1*	*1*
9	*420*	*2*	*3.15*	*7*	*-5*	*25*
10	*505*	*1*	*3*	*9*	*-8*	*64*

Следователно стойностите от тази таблица могат да бъдат добавени към връзката за изчисляване на коефициента на рангова корелация на Спирман:

$$\sum d^2 = 256$$

$$n = 10$$

$$\rho = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n^2-1)} = 1 - \frac{6 * 256}{10(100-1)} = 1 - \frac{1536}{990} = 1 - 1.55 = -0.55$$

И така, установихме, че между тези два атрибута има корелация от -0.55, което може да се нарече умерено силна антикорелация.

Нека изчислим и стойността на коефициента на корелация на Пирсън. За да изчислим този вид коефициент, трябва да определим средната

стойност на разстоянието μ(разстояние) и средната стойност на цената на кебапа μ(цена):

- *μ(разстояние) = 203, означен по-нататък като μ(d)*
- *μ(цена) = 3.39, означен по-нататък като μ(p)*

Таблицата ни ще съдържа още колони, но всички те са само предварителни изчисления на частите, необходими за крайното съотношение на корелационния коефициент на Пиърсън[1]*:*

индекс	d	p	d − μ(d)	p − μ(p)	(d − μ(d))²	(p − μ(p))²	(d − μ(d)) (p − μ(p))
1	*10*	*4*	-193	0.61	37 249	0.3721	-117.73
2	*70*	*3.50*	-133	0.11	17 689	0.0121	-14.63
3	*85*	*3.30*	-118	-0.09	13 924	0.0081	10.62
4	*100*	*3.20*	-103	-0.19	10 609	0.0361	19.57
5	*130*	*3.80*	-73	0.41	5 329	0.1681	-29.93
6	*195*	*2.90*	-8	-0.49	64	0.2401	3.92
7	*215*	*3.10*	12	-0.29	144	0.0841	-3.48
8	*300*	*3.90*	97	0.51	9 409	0.2601	49.47
9	*420*	*3.15*	217	-0.24	47 089	0.0576	-52.08
10	*505*	*3*	302	-0.39	91 204	0.1521	-117.78

Така че можем да изчислим коефициента на корелация на Пиърсън, както следва:

$$\sum((\mathrm{d} - \mu(\mathrm{d}))^2) = 232\ 710$$

$$\sum((\square - \mu(\square))^2) = 1.3905$$

$$\sum((\mathrm{d} - \mu(\mathrm{d}))\,(\square - \mu(\square))) = -252.05$$

$$\mathrm{r} = \frac{\sum_{i=1}^{n}(A_i - \mu(A))(B_i - \mu(B))}{\sqrt{\sum_{i=1}^{n}(A_i - \mu(A))^2}\sqrt{\sum_{i=0}^{n}(B_i - \mu(B))^2}} = \frac{-252.05}{\sqrt{232\ 710}\sqrt{1.3905}} \approx \frac{-252.05}{569.232} \approx -0.44$$

И така, установихме, че между тези два признака има корелация на Пиърсън от -0.44, което може да се нарече умерено силна антикорелация. За повече информация относно тълкуването на резултатите от коефициента на корелация вижте края на този раздел на наръчника.

[1] [1]Поради липса на място в представената таблица използваме d за разстоянието и p за цената на кебапа.

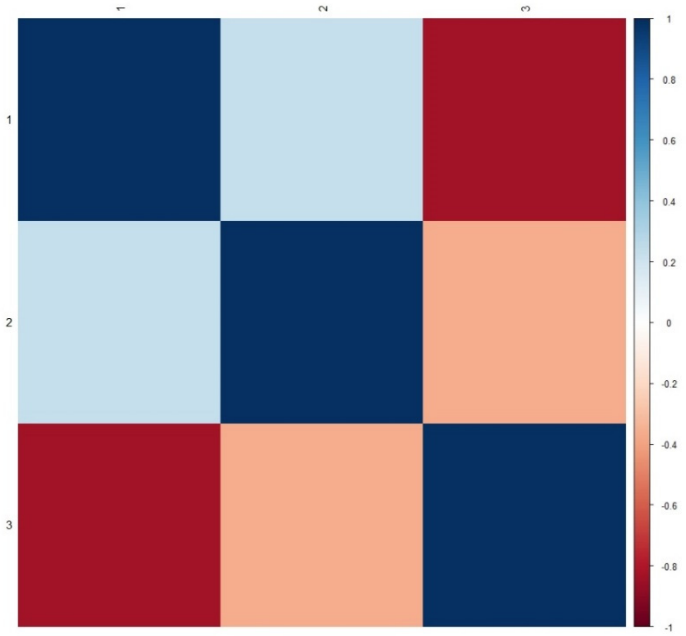
Корелационна матрица и корелационна топлинна карта

Наборите от данни рядко съдържат само два атрибута, което води до необходимостта от определяне на коефициентите на корелация между всички техни атрибути. За тези цели използваме корелационна матрица – таблица, съдържаща коефициенти на корелация, определени между всички възможни двойки атрибути в набора от данни. В таблицата по-долу виждаме коефициентите на корелация, измерен между атрибутите A_1, A_2, A_3 (матрица на корелацията), като можем да видим, че $r(A_1, A_2)$ = 0.238, $r(A_1,A_3)$ = -0.834 и т.н.

	A_1	A_2	A_3
A_1	1	0.238	-0.834
A_2	0.238	1	-0.362
A_3	-0.834	-0.362	1

Тази матрица има две **естествени свойства** – тя е симетрична по диагонала и този диагонал винаги съдържа стойностите на коефициента на корелация, равни на 1 – корелацията на атрибута A_1 със самия себе си винаги е $r(A_1, A_1)$ = 1, независимо от използвания метод, което също е естествено, тъй като стойността на атрибута A_1 напълно зависи от стойността на атрибута A_1.

Такъв метод на корелационен анализ е подходящ само за определен брой атрибути в изследваната съвкупност от данни. Очевидно е, че за набор от данни, съдържащ десетки атрибути, такава матрица би била объркваща и трудна за четене. Поради това тя често се заменя с така наречената **корелационна топлинна карта** или **корелационна диаграма**. За корелационната матрица, представена по-горе, топлинната карта може да бъде конструирана по следния начин:

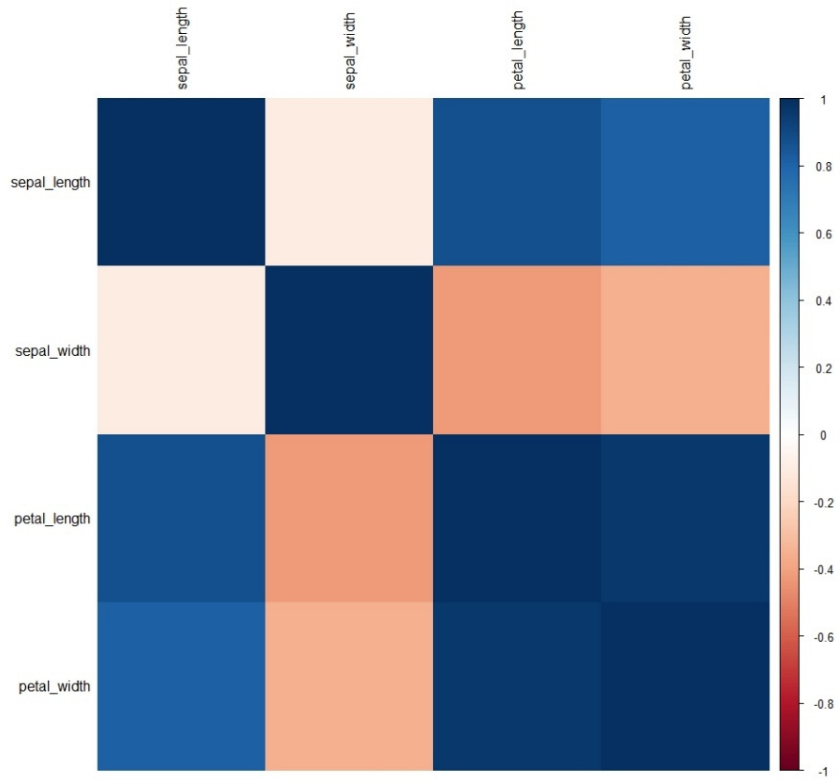


Такава корелационна топлинна карта е просто проекция на корелационната матрица в цветна мрежа, в която цветът на полето се определя от стойността на корелационния коефициент за дадена двойка атрибути. За по-добра четливост в топлинната карта на корелацията е посочена скалата (вдясно), съдържаща интервала от възможни стойности за коефициента на корелация. Вместо да търсим числа, близки до крайните стойности на интервала [1,-1] в корелационната матрица, в корелационната топлинна карта търсим тъмночервени или тъмносини полета на мрежата, които показват същото свойство и са по-лесни за идентифициране за повечето хора.

Пример: Нека вземем набора от данни Iris, който е описан в Приложение А към настоящия наръчник. Този набор от данни съдържа пет атрибута, измерени върху 150 субекти, четири от които са цифрови атрибути, а петият съдържа лингвистична стойност, указваща класа за дадения субект. Като част от корелационния анализ не е възможно да се работи с лингвистични атрибути, така че ще разгледаме само набора от данни с размер 150×4. Корелационната матрица на Пирсън на (донякъде орязаната) съвкупност от данни за Iris съдържа следните стойности:

	дължина на чашелистчетата	ширина на чашелистчетата	дължина на венчелистчетата	ширина на венчелистчетата
дължина на чашелистчетата	1	-0.1093692	0.8717542	0.8179536
ширина на чашелистчетата	-0.1093692	1	-0.4205161	-0.3565441
дължина на венчелистчетата	0.8717542	-0.4205161	1	0.9627571
ширина на венчелистчетата	0.8179536	-0.3565441	0.9627571	1

Разбира се, тази съвкупност от данни не съдържа голям брой атрибути, така че създаването на корелационна топлинна карта не е необходимо за извършване на корелационен анализ. Въпреки това обаче представяме топлинната карта, като демонстрация на проекцията на стойностите на корелационния коефициент в цветовата скала, представена корелационната топлинна карта.



Интерпретиране на резултатите от корелационните коефициенти

Коефициентът на корелация показва доколко е възможно да се предскажат стойностите на атрибута A_2 в избраната извадка от данни въз основа на атрибута A_1. Колкото по-близо е стойността на този коефициент до крайните стойности на разглеждания интервал (т.е. до стойността 1 или -1), толкова по-подходящ е даденият атрибут A_1 за прогнозиране на стойностите на незадължителния атрибут A_2.

Очевидно е, че стойност **0 е най-лошият случай** за корелацията между два атрибута. В такъв случай между стойностите на тези атрибути не съществува връзка, която би могла да се използва в случай на изграждане на математически модели върху набора от данни, с който работим.

Литературата се различава малко в мнението си за степента на приемливост на стойностите на корелацията – с изключение на това, че колкото по-висока е корелацията, толкова по-добре. Като цяло говорим за два атрибута като силно корелирани, когато стойността на коефициента на корелация, измерен между тях, достига стойности, **по-високи от 0.8**. Налице е силна антикорелация между два атрибута, ако коефициентът на корелация достига стойности, **по-ниски от -0.8**. Тази граница на приемливост на потенциала за прогнозиране може да **бъде смекчена по-близо до стойностите 0.7 или -0.7**, но повече не се препоръчва.

4.3. Проучвателен анализ на данни и визуализация на данни

Проучвателният анализ на данни (обикновено наричан ПАД) е метод за анализ на данни, при който се използва проучване на данни с цел да се открият модели и тенденции в дадена популация или извадка. В най-основната си форма този вид анализ се извършва чрез визуално изследване на данните. Преди самата визуализация е необходимо да се предприемат няколко стъпки, които ще бъдат от полза при по-нататъшното търсене на скритото в данните знание:

- **Запознаване с набора от данни –** необходимо е да можем да отговорим на няколко въпроса, свързани с дадения набор от данни, преди да го анализираме:
  - **Кой, кога и защо е съставил набора от данни?** Този въпрос е от съществено значение от гледна точка на релевантността, актуалността и полезността на набора от данни. Ако наборът от данни е съставен от експерти в областта, той е по-подходящ, отколкото ако е съставен от начинаещ, който е измервал данни с евтин сензор от потребителски клас. Ако наборът от данни е бил съставен преди 93 години, възможно е данните да не са актуални, измерванията да са по-малко точни, отколкото бихме могли да измерим днес, и т.н. Наборът от данни също така е съставен с конкретна цел и поради това не е универсален (може да не е подходящ за всички задачи).
  - **Колко голям е този набор от данни?** Под размер на набора от данни разбираме броя на същностите и атрибутите, измерени в набора от данни. В случай че разполагаме с набор от данни, който е твърде голям, за да можем да работим удобно с него (виж Раздел 1), трябва да изберем извадка от него въз основа на принципите, споменати в предишния раздел на настоящото ръководство. Противоположният проблем – твърде малка съвкупност от данни – е много по-голям проблем. Съществуват обаче алгоритми, които могат да работят с малки набори от данни, и подходи, които генерират нови единици въз основа на съществуващи, наречени „свръхпроизводство“(oversampling).
  - **Какъв е съставът на масивите от данни?** Този въпрос е тясно свързан с причината за съставяне на набора от данни. Важно е да се прегледат всички атрибути на набора от данни и да се разбере тяхното предназначение. Също така е необходимо да се съсредоточите върху това дали данните, записани в дадения атрибут, са числови или категорични и в какъв диапазон се движат стойностите на отделните атрибути.

- **Изчисляване на обобщени статистически данни** – за всеки атрибут е препоръчително да се съберат основни обобщени статистически данни. Препоръчителните стойности са екстремум (минимум, максимум), медиана или средна стойност, стандартно отклонение и други. Това е много важна и информативна стъпка, при която получаваме средните стойности на дадените атрибути, техните най-малки и най-големи стойности и можем да опишем допълнително отделните атрибути чрез агрегиране.
- **Извършване на корелационен анализ** – за всяка съвкупност от данни е препоръчително да се състави матрица или топлинна карта на корелационните коефициенти. Тази матрица измерва корелациите между стойностите на всички атрибути на набора от данни и по този начин ни показва колко трудно ще ни бъде да изградим математически модели върху дадения набор от данни. Корелационният анализ помага и за идентифициране на атрибути и подмножества на набора от данни, подходящи за визуализация.

Следващата стъпка от изследователския анализ на данни е същинската визуализация на набора от данни. Става дума обаче за ефективна визуализация на данни въз основа на няколко принципа, представени в следващата част на този наръчник.

Ефективна визуализация на данни

Наричаме визуализацията на данни ефективна в случай, че:

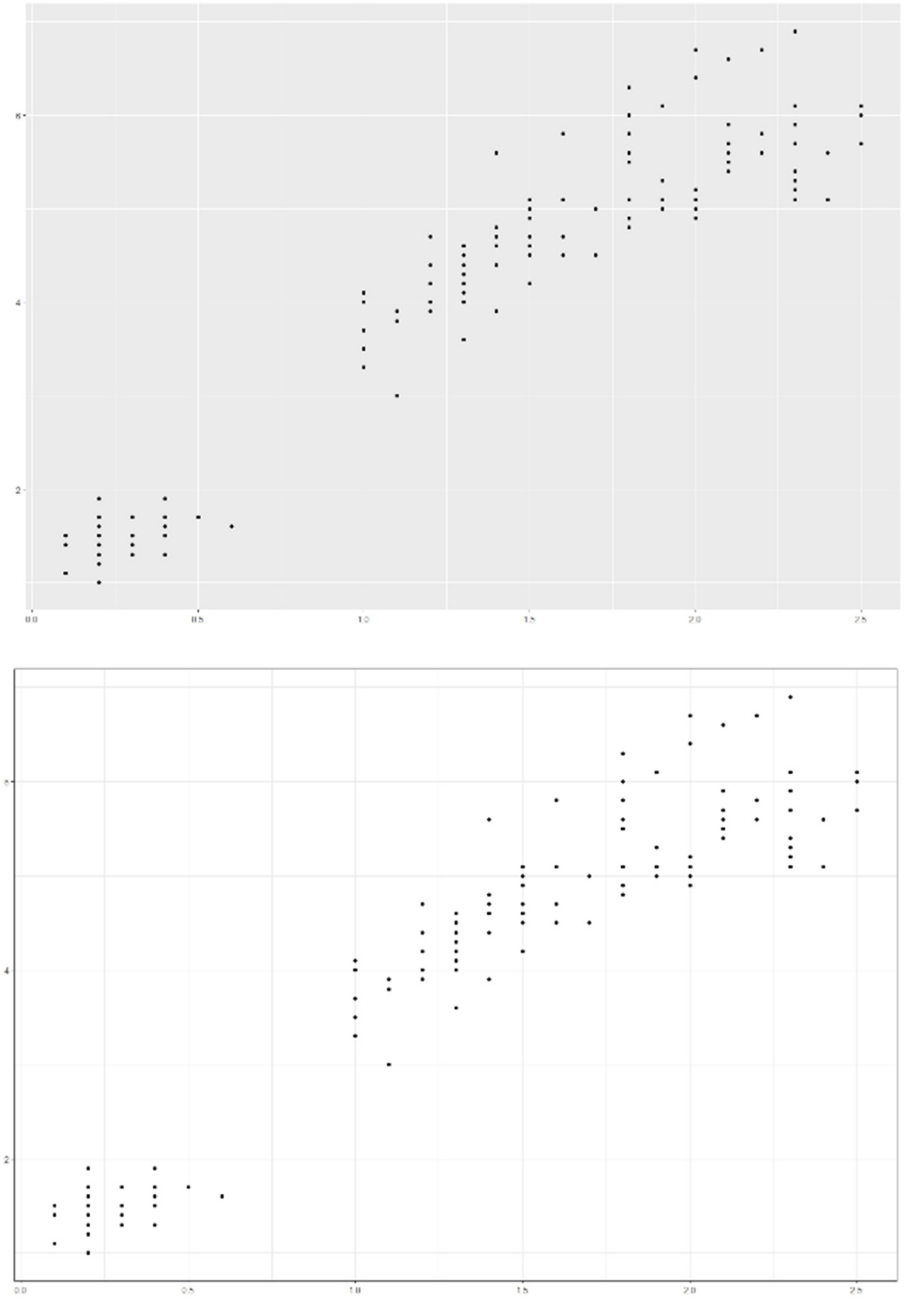
- визуализираме правилните данни;
- по правилния начин;
- използваме правилния тип графика.

Тези три точки са съвсем естествени. Данните, които могат да се нарекат подходящи за визуализация от гледна точка на анализа на данни, са тези, които носят някакво знание – най-често прогностичен потенциал. Както вече знаем от предишния раздел на наръчника, можем лесно да търсим прогностичен потенциал в наборите от данни, като използваме методите на корелационния анализ, и следователно частите от набора от данни, подходящи за визуализация, ще бъдат тези, в които сме установили силни корелации или антикорелации между стойностите на атрибутите (вж. раздел Интерпретиране на резултатите от корелационните коефициенти).

В нашия случай под **правилен начин за визуализация на данни** разбираме елиминирането на два сравнително често срещани проблема при визуализацията на набори от данни:

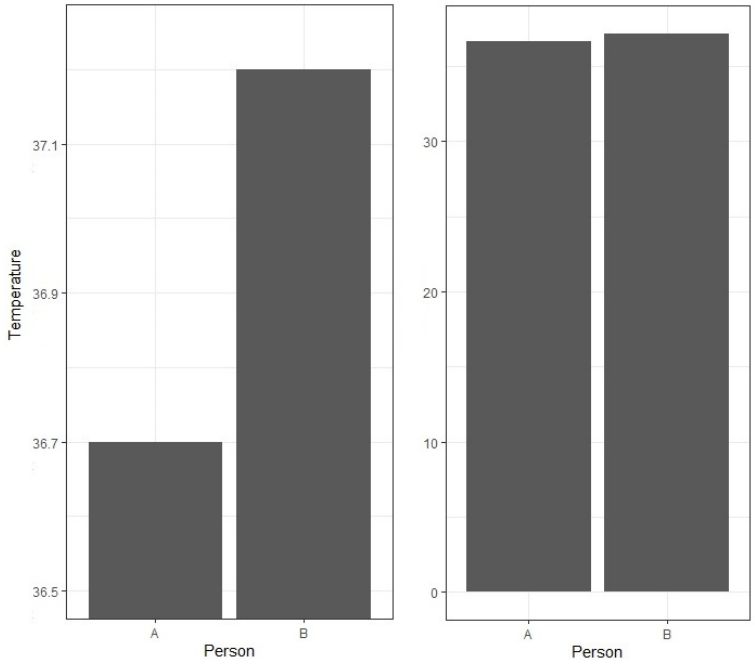
- **Максимизиране на съотношението между използвания цвят и данните –** тъй като искаме да визуализираме данните, в идеалния случай графиката ще съдържа минимум други графични еле-

менти (например фонов цвят, отличителна мрежа и т.н.). Такова максимизиране на съотношението между цвят и данни е от съществено значение, особено когато се визуализират големи набори от точки, които потенциално могат да се слеят или да бъдат изобразени от много малки точки (или друг тип обекти). Фигурата по-долу съдържа стандартна графика на точки, начертана на езика R (вляво), и модификация на тази графика, така че данните да са по-видими.

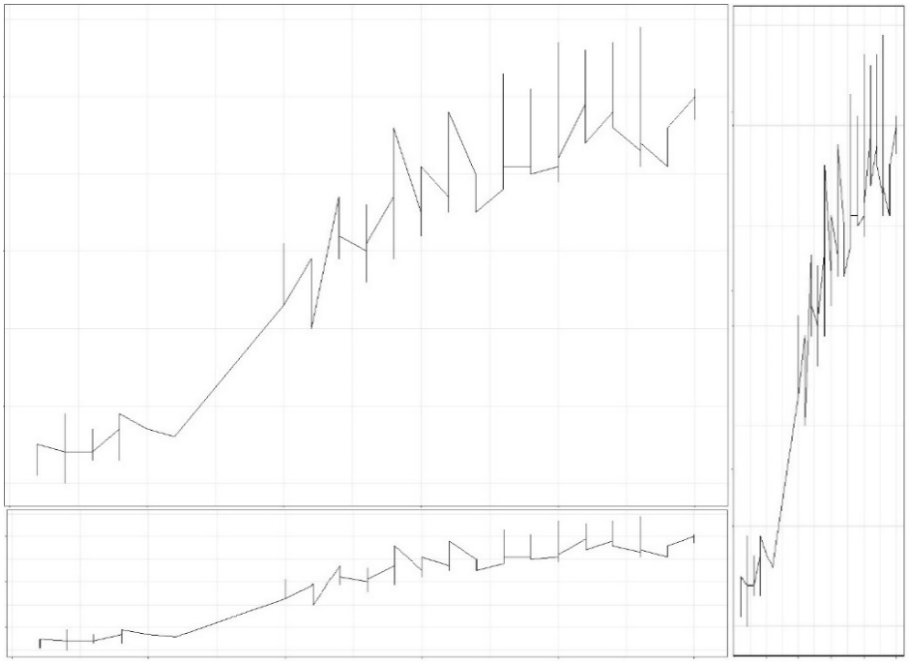


- **Премахване на изкривяванията –** при визуалното представяне на данните и последващото им тълкуване често се появяват изкривявания поради неправилно задаване на осите или поради изкривяване на самия файл с изображение. На фигурите по-долу са предста-

вени и двата проблема. Първата стълбовидна графика съдържа сравнение на температурата на двама души (A и B). Обърнете внимание, че стойностите на лявата графика изглеждат значително различни, докато стойностите на дясната са много сходни, въпреки факта, че това са две идентични измервания, представени с различни настройки на осите. На лявата фигура представяме оста y само от стойността 36.5 до 37.3 градуса, а на дясната графика представяме стойностите на y от 0 до 40 градуса. Това е типичен фактор за объркване при визуализацията на данни.



Вторият проблем с изкривяването е изкривяването на самото изображение.

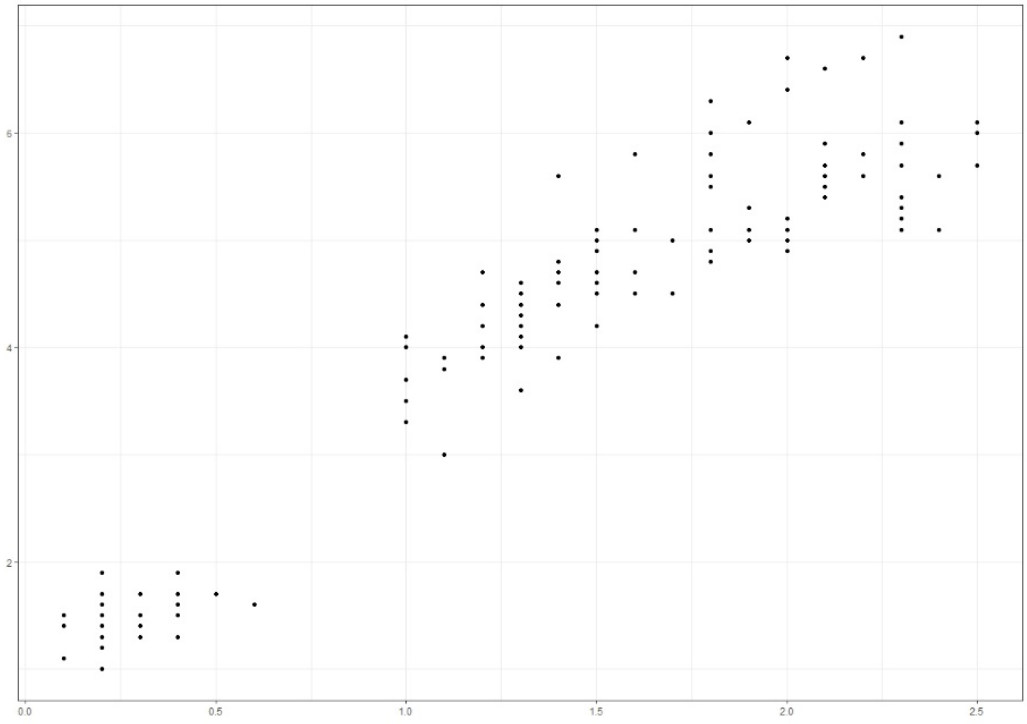


На изображението по-горе можем да видим една графика, представена в три съотношения на формата на графиката. Ясно е, че изображението в долната и дясната част е неподходящо поради изкривяването на действителната форма на тенденцията, представена от дадената графика, и следователно идеалното съотношение за тези данни ще бъде горното, ляво изображение на графиката – близко до стандартното съотношение на изображението на повечето съвременни проектори – 16:9.

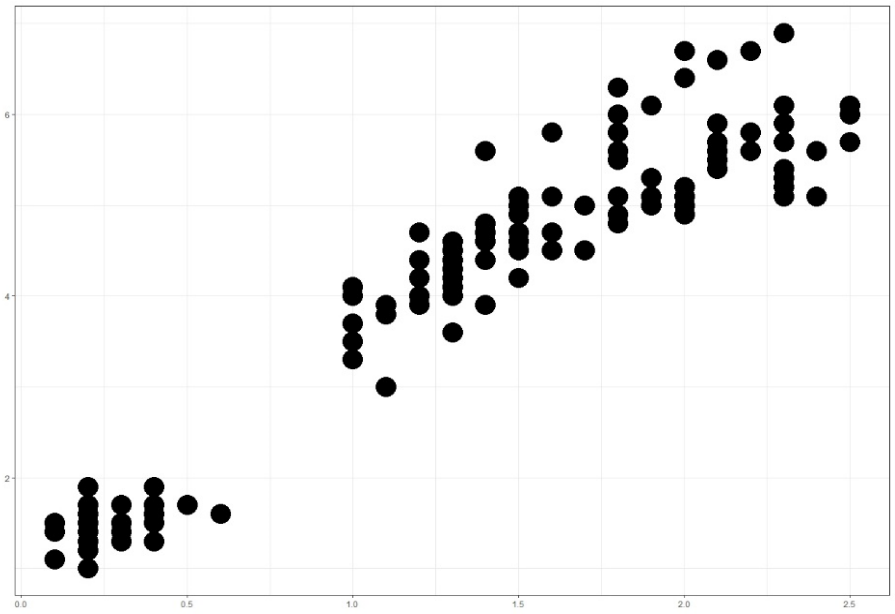
Последният много важен елемент за ефективността на визуализацията е използването на правилния тип графика. По принцип са популярни четири вида графики – точкови, линейни, кръгови и стълбовидни. Всеки от тези видове графики е подходящ за различни цели и има различни предимства и недостатъци. В този раздел на наръчника ще се спрем само на двата най-разпространени метода за визуализация на данни – точкови и линейни графики.

Точкови графики

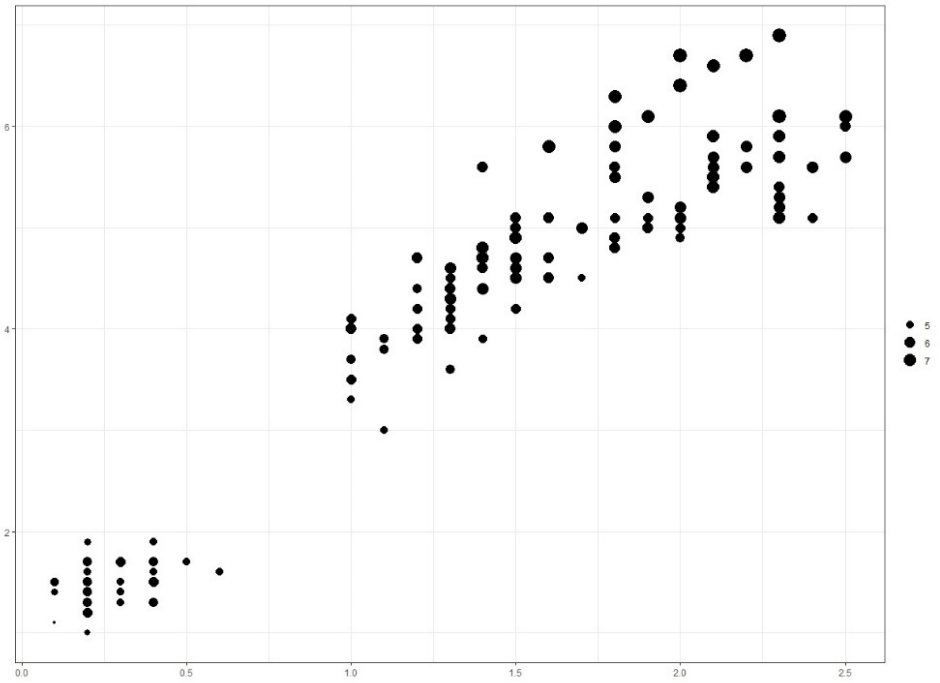
Точковите графики се използват за визуализиране на връзката между два (или повече) атрибута на набор от данни с помощта на точки. Стандартният начин за визуализация е да се сравнят стойностите на два атрибута в равнина:



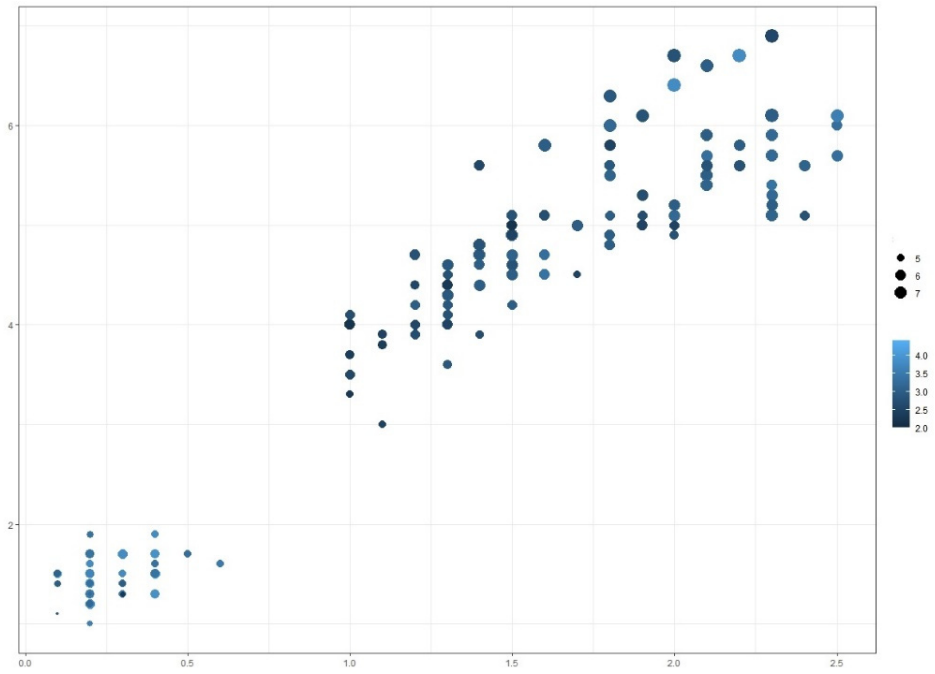
При такъв подход към визуализацията на данни трябва да обърнем внимание на размера на точката в графиката. Изображението по-долу показва пренасищане на графиката, причинено от голям размер на точката; подобен проблем възниква при голям брой точки, които са разположени близо една до друга.



Размерът на точките обаче може да се използва в контекста на визуализацията на данни за предаване на допълнителна информация. Ако зададем размер на точката, пряко пропорционален на размера на третия атрибут в набора от данни, можем да визуализираме връзката между три атрибута от набора от данни.

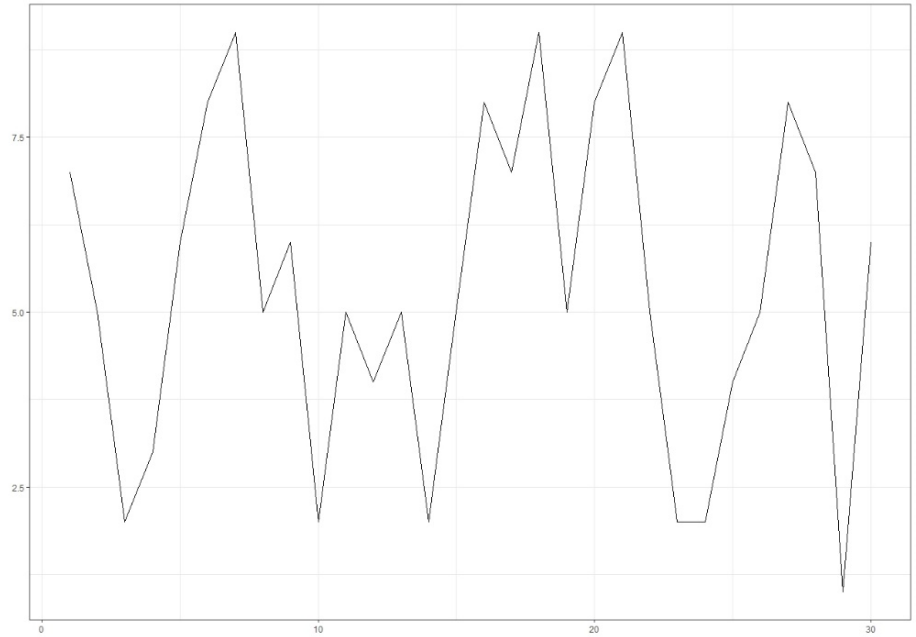


Можем да разширим тази концепция с други свойства на точките, например с техния цвят (изброени по-долу), и така да постигнем визуализация на връзката между четири атрибута на набора от данни. Този метод на визуализация обаче е труден за използване при голям брой точки или атрибути. По принцип не се препоръчва визуализирането на повече от три или четири атрибута в една равнина.

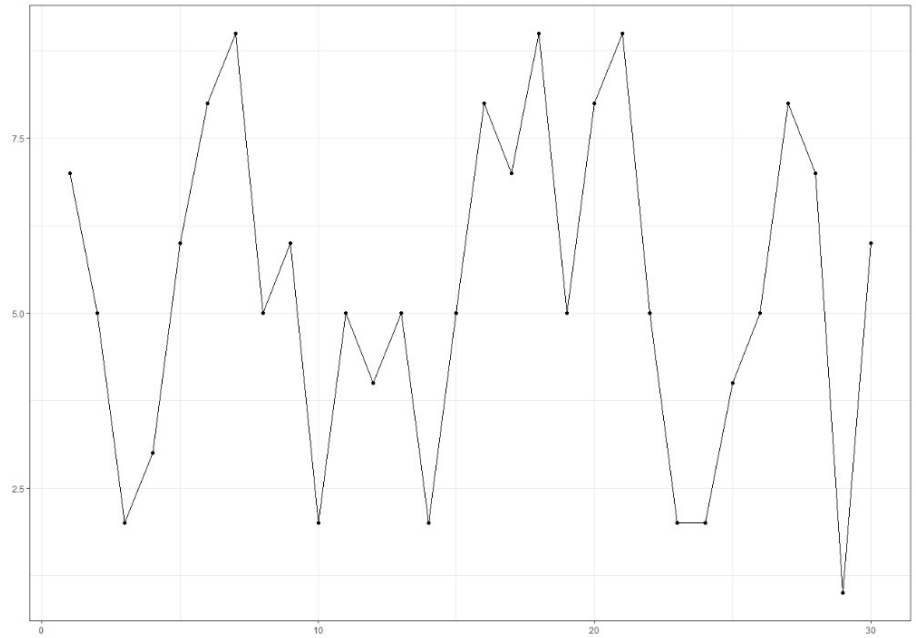


Линейни графики

Линейните графики се използват за визуализиране на хода на стойността на един атрибут във времето или за визуализиране на колебанията на стойността на даден атрибут в зависимост от друг атрибут. Важно е да се отбележи, че линиите, които присъстват в линейните графики, са приближение на точките – преобразуване на дискретни точки от данни в непрекъснати линии и поради това на някои места може да са неточни. В сравнение с точковите графики линейните графики са трудни за използване за категорични (лингвистични) данни.

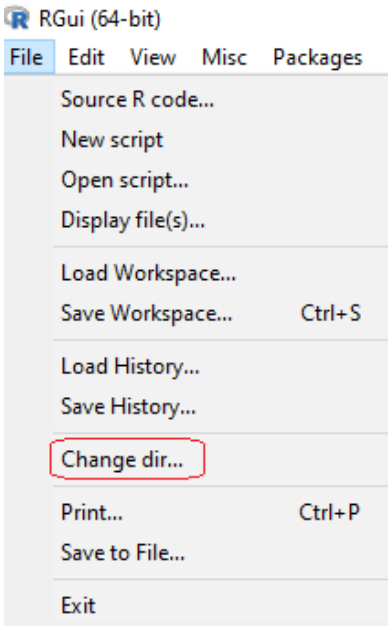


Тъй като линиите са приблизителни стойности на точките от данни, препоръчва се да се визуализират линията и точките, на които се основава линията. Това намалява двусмислеността на коректността на стойностите на атрибутите.



4.4. Изследователски анализ на данни в практиката

Тази част от наръчника е посветена на практическото приложение на методите за анализ на проучвателни данни на езика R. Примерите, дадени по-долу, са приложени във версия R *4.3.0*, но всички тези концепции и команди присъстват във всички версии на всички програмни инструменти, подходящи за анализ на данни.



Първо, трябва да променим работната директория на сесията към директорията, в която се съхраняват нашите структурирани данни – това може да стане с помощта на горната лента на програмата, където: *File → Change dir → set the working directory*. За целите на представянето на метода за анализ на проучвателни данни ще използваме набора от данни Iris, описан в Приложение А.

След като сменихме работната директория, можем да започнем да зареждаме набори от данни в R. Тази операция може да се извърши по няколко начина, като представяме (от наша гледна точка) най-простия – командите *read.table* и *read.csv*, които работят по един и същи начин за различни типове входни файлове. Командата под формата *read.csv* се използва за файлове с разширение *.csv*, които са най-често срещаните, структурирани входни данни за инструментите за анализ на данни. Формата *read.table* може да се използва в случай на входни файлове във формат *.txt* или *.data*.

```
read.table("title“, header=T/F, sep="symbol")
read.csv("title“, header=T/F, sep="symbol")
```

където *title* е името на файла, в който се съхраняват нашите данни, заедно с разширението на файла, частта header на командата показва дали файлът с входни данни има заглавие (*T* за истина /true/) или не (*F* за невярно/false/), а частта *sep* на командата очаква символа, с който се разделят стойностите на атрибутите във файла.

За да можем да използваме заредения набор от данни по-нататък в програмата, трябва да го запазим под избрано заглавие, например *title_of _data*:

```
title_of_data <- read.table("title“, header=T/F, sep="symbol")
```

Пример за такова зареждане на файл с данни, съхраняван под заглавието *our_data.data*, може да изглежда по следния начин. Във втората дадена команда записваме нашия набор от данни под заглавие *data*:

```
read.table("our_data.data“, header=T, sep=“,")
data <- read.table("our_data.data“, header=T, sep=“,")
```

Проучвателен анализ на данни – Стъпка 1 – Запознаване с даден набор от данни

Като част от опознаването на набора от данни можем да извършим няколко много прости операции. Първата от тях е да изведем цялата съвкупност от данни в конзолата на инструмента R, като използваме *title_of _data*. Това обаче не е практично за големи масиви от данни, които съдържат хиляди единици, и затова представяме втората версия на командата за изписване на единиците от масива от данни – *head*, която изписва определения брой единици от началото на масива от данни в конзолата.

```
title_of_data
head(title_of_data, number_of_entities)
```

Пример за тази концепция може да бъде изписването на целия набор от данни, съхраняван под данните за името, или изписването на първите пет единици от този файл.

```
data
head(data, 5)
```

Резултатът от тази команда на езика R ще бъде псевдотаблица в следния формат:

```
> data <- read.table("iris.data", header = T, sep = ",")
> head(data, 5)
  sepal_length sepal_width petal_length petal_width       class
1          5.1         3.5          1.4         0.2 Iris-setosa
2          4.9         3.0          1.4         0.2 Iris-setosa
3          4.7         3.2          1.3         0.2 Iris-setosa
4          4.6         3.1          1.5         0.2 Iris-setosa
5          5.0         3.6          1.4         0.2 Iris-setosa
```

След основно запознаване с набора от данни, неговите атрибути и стойностите в тях можем да пристъпим към изчисляване на мерките за централност и променливост. Всички тези функции се извеждат от английската версия на името на отделните функции (напр. *sd* за стандартно отклонение) и техният вход е само един от атрибутите на набора от данни, записан във формата

title_of _data$title_of _attribute.

Най-универсалната от тези команди е функцията за обобщаване, която измерва *минимума*, *първия квартил*, *медианата*, *средната стойност*, *третия квартил* и *максимума* за всички атрибути от набора от данни.

```
mean(title_of_data$attribute_title)
median(title_of_data$attribute_title)
min/max/sum(title_of_data$attribute_title)
sd(title_of_data$attribute_title)
summary(title_of_data)
```

Пример за такъв анализ на статистическите свойства на данните е използването на следните команди:

```
summary(data)
sd(data$attribute_title)
```

Изходът на тези функции, изпълнени върху набора от данни на Iris, се състои от следния набор от стойности:

```
> summary(data)
  sepal_length    sepal_width     petal_length    petal_width          class
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   Length:150
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   Class :character
 Median :5.800   Median :3.000   Median :4.350   Median :1.300   Mode  :character
 Mean   :5.843   Mean   :3.054   Mean   :3.759   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
> sd(data$sepal_length)
[1] 0.8280661
```

Проучвателен анализ на данните – Стъпка 2 – Корелационен анализ

Както беше споменато в предишните раздели на този наръчник, една от най-съществените части на проучвателния анализ на данни е корелационният анализ. Основната форма на командата за корелационен анализ е изчисляването на коефициента на корелация между два атрибута на набора от данни с помощта на функцията cor. В синтаксиса на командата, представен по-долу, можем да видим, че типът на коефициента на корелация, който искаме да изчислим за данните, може да бъде определен с помощта на параметъра *method = type_of _correlation*. По подразбиране в тази команда се използва коефициентът на корелация на Пирсън.

```
cor(title_of_data$attribute_title_1, title_of_data$attribute_title_2)

cor(title_of_data$attribute_title_1,  title_of_data$attribute_title_2,
method = „pearson")

cor(title_of_data$attribute_title_1, title_of_data$attribute_title_2,
method = „spearman")
```

Като част от анализа на набора от данни обаче искаме да разгледаме всички връзки между всички атрибути на набора от данни и затова можем да създадем матрица на корелациите:

```
cor(title_of_data)
cor(title_of_data, method = „spearman")
```

Корелационният анализ на набора от данни Iris може да се извърши с помощта на следните прости команди:

```
cor(data[, 1:4])
cor(data[, 1:4], method = „spearman")
```

Забележка: Тъй като наборът от данни на Iris съдържа един атрибут, чиито стойности са езикови (клас на атрибута), а корелационната матрица е съставена само от числови стойности, функцията cor трябва да има за вход само първите четири (числови) атрибута. Постигаме това, като избираме колони 1:4 от набор от данни, наречен data: data[, 1:4].

```
> cor(data[,1:4])
             sepal_length sepal_width petal_length petal_width
sepal_length    1.0000000  -0.1093692    0.8717542   0.8179536
sepal_width    -0.1093692   1.0000000   -0.4205161  -0.3565441
petal_length    0.8717542  -0.4205161    1.0000000   0.9627571
petal_width     0.8179536  -0.3565441    0.9627571   1.0000000
> cor(data[,1:4], method = "spearman")
             sepal_length sepal_width petal_length petal_width
sepal_length    1.0000000  -0.1594565    0.8813864   0.8344207
sepal_width    -0.1594565   1.0000000   -0.3034206  -0.2775111
petal_length    0.8813864  -0.3034206    1.0000000   0.9360034
petal_width     0.8344207  -0.2775111    0.9360034   1.0000000
```

Както е споменато в Раздел 4.2, за големи набори от данни е препоръчително да се използва корелационна топлинна карта. За да можем да използваме този метод за визуализация, трябва да инсталираме пакет от функции на езика R, който съдържа функция за визуализация на корела-

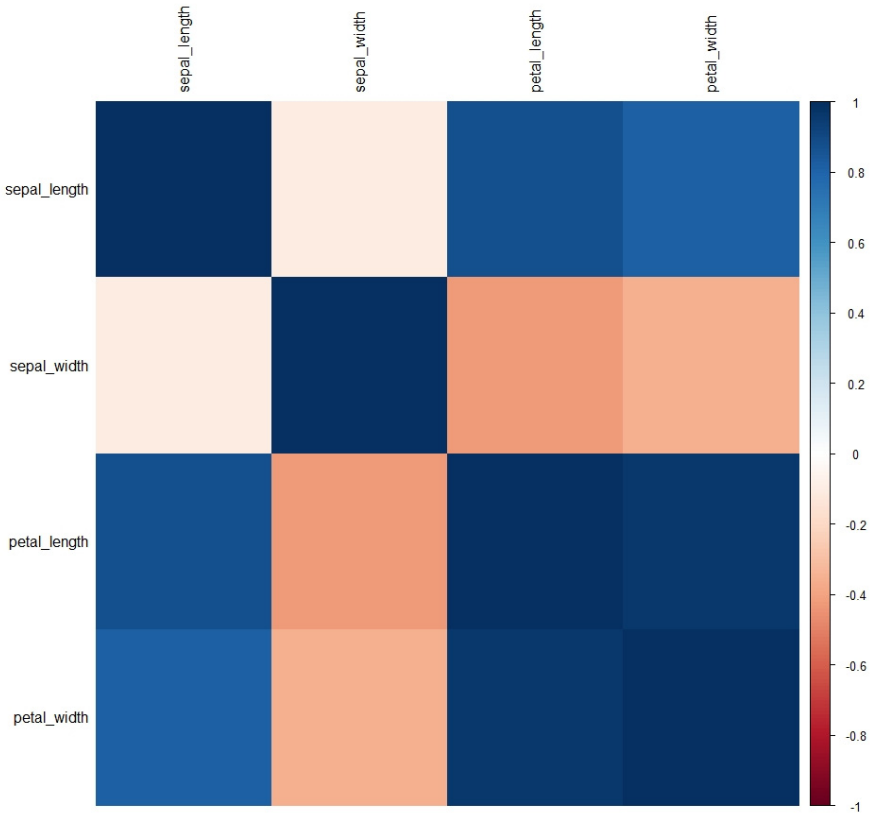
ционна топлинна карта, наречена *corrplot*. След като инсталираме този пакет, просто го зареждаме с помощта на функцията *require* и след това създаваме корелационна топлинна карта:

```
install.packages("corrplot")
require(corrplot)
corrplot(cor(data), method = „color")
```

Корелационната матрица и топлинната карта за набора от данни на Iris ни показват взаимоотношенията, които могат да се използват при по-нататъшен анализ на данните. По-конкретно се интересуваме от всички взаимоотношения между атрибутите, при които който и да е вид корелационен коефициент е по-висок от 0.8 или по-нисък от -0.8:

- ρ(дължина на чашелистчетата, дължина на венчелистчетата) ≈ 0.87
- ρ(дължина на чашелистчетата, дължина на венчелистчетата) ≈ 0.82
- ρ(дължина на чашелистчетата, дължина на венчелистчетата) ≈ 0.94

Тези връзки между атрибутите заслужават да бъдат визуализирани.



Проучвателен анализ на данни – Стъпка 3 – Визуализация на данните

След като анализираме корелациите, сме готови да визуализираме двойки атрибути, при които сме забелязали силна корелация или антикорелация. Преди самата визуализация обаче трябва да инсталираме пакет, използван за целите на визуализацията:

```
install.packages("ggplot2")
require(ggplot2)
```

Пакетът *ggplot2* е един от най-популярните пакети за визуализация на данни, който съдържа функции за изчертаване на точкови графики, линейни графики, стълбовидни графики и много други. В този раздел на наръчника ще подберем няколко прости примера за тези функции.

Точкови диаграми

Най-основната и същевременно най-мощната визуализация на данни е точкова графика. В рамките на езика R и пакета *ggplot2* използваме функцията *ggplot* за конструиране на всякакъв вид графики, като функцията очаква няколко стойности като вход. За най-простите графики тези входни данни са:

- Заглавие на набора от данни, с който работим (в нашия случай това заглавие е *data*).
- Разделът *aes*, произлизащ от английската дума aestetics, очаква информация за поне една ос. Тази информация се представя под формата axis_title (x или y) = *title_of _attribute* представен от оста.
- Вид на графиката.

По принцип синтаксисът на командите за точкови графики съдържа присвояване на два атрибута към осите на графиката и част от командата + *geom_point()*. Обобщеният синтаксис за този тип команди е представен по-долу:

```
ggplot(title_of_data, aes(x = title_of_attribute_1, y =
title_of_attribute_2)) + geom_point()
```

Можем да променим цвета на точките, като използваме разширението на раздела на командата + *geom_point()*, като добавим втори раздел aes, валиден само за самите точки, а именно + *geom_point*(*aes*(*color*

= „*color name*")). Като алтернатива на посочването на цвят можем да променим цвета на точките в рамките на една графика, като вместо името на цвета посочим името на атрибута от набора от данни, с който работим, в секцията на командата + *geom_point()*. По този начин постигаме визуализация в две измерения (атрибути) с допълнително измерение, маркирано от цвета на точките, който се променя въз основа на стойностите на избрания атрибут. За да се придържаме към принципите за ефективна визуализация на данни, представени в предишния подраздел, добавяме в края на командата опцията + *theme_bw()*, която осигурява бял фон под самата графика, като по този начин максимално се увеличава съотношението между данните и цвета, използвани в графиката.

```
ggplot(title_of_data, aes(x = title_of_attribute_1,
y = title_of_attribute_2))) + geom_point(aes(color = „color"))
+ theme_bw()

ggplot(title_of_data, aes(x = title_of_attribute_1, y =
title_of_attribute_2))) + geom_point(aes(color = title_of_attribute_3)) +
theme_bw()
```

Прост пример за този подход може да се направи, както е представено по-долу. Въвеждаме и две допълнителни понятия:

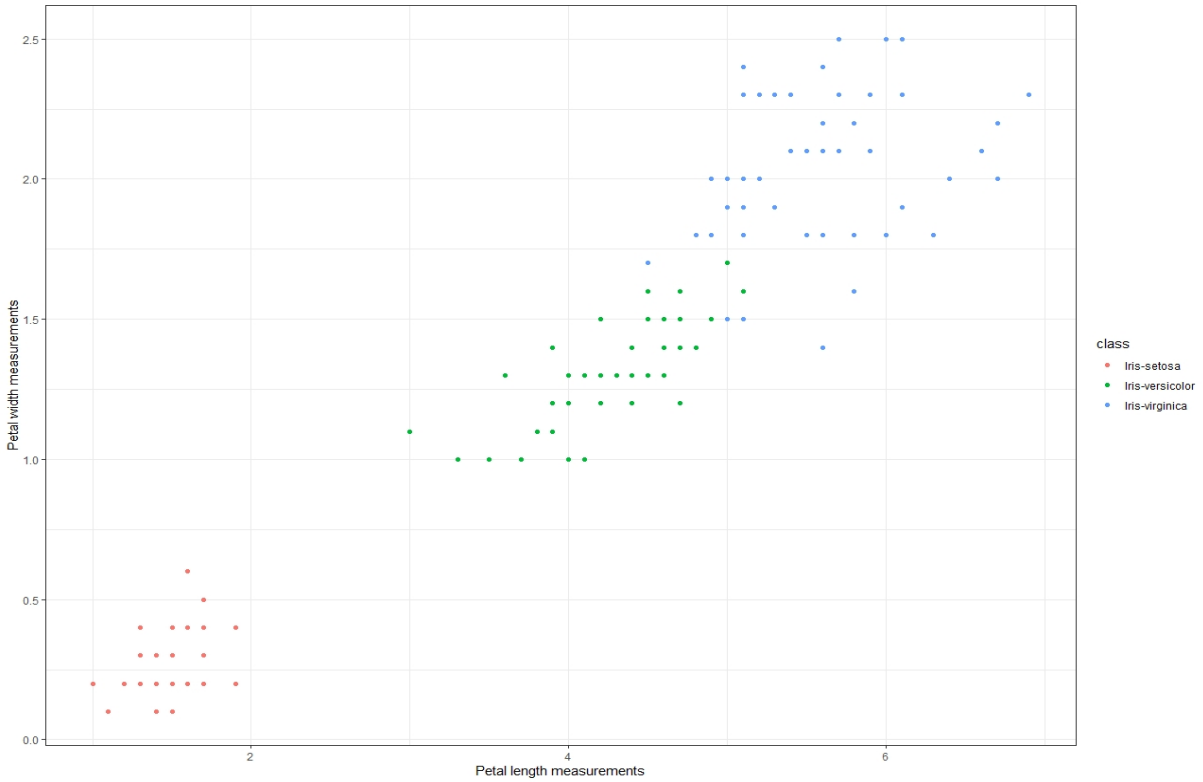
- Създадената графика може да бъде запазена под избрано име, например graph1, както е показано по-долу.
- Можем да добавим други части на командата към запазената по този начин графика. В примера по-долу използваме + *xlab()* и + *ylab()*, за да добавим етикети за осите x и y. След това графиката се визуализира чрез извикване на нейното име в конзолата.

```
graph1 <- ggplot(data, aes(x= atr_1, y = atr_2))
+ geom_point(aes(color = class) + theme_bw()

graph1 <- graph1 + xlab("X parameter") + ylab("Y parameter")

graph1
```

Следващият код за набор от данни Iris:

```
> require(ggplot2)
> graph1 <- ggplot(data, aes(x = petal_length, y = petal_width)) + geom_point(aes(color = class)) + theme_bw()
> graph1 <- graph1 + xlab("Petal length measurements") + ylab("Petal width measurements")
> graph1
```

създава следната фигура:



Линейни графики

Друг често срещан вид графики е линейната графика, която се използва за визуализиране на хода на стойността на един атрибут във времето или за визуализиране на колебанията на стойността на даден атрибут в зависимост от друг атрибут. Синтаксисът на командата за линейна графика в пакета *ggplot2* не се различава съществено от предишните примери, представени в този раздел на наръчника. Единствената разлика е видът на геометрията, която се използва при изчертаването на графиката – в случая с линейните графики тя е + *geom_line()*. С помощта на опцията *linetype* в секцията на командата *geom_line()* можем да променим и вида на линията, която се използва при изчертаването на данните.

```
ggplot(title_of_data, aes(x=attribute_title_1, y=attribute_title_2))
+ geom_line()

ggplot(title_of_data, aes(x=attribute_title_1, y=attribute_title_2))
+ geom_line(linetype = „dashed")
```

```
ggplot(title_of_data, aes(x=attribute_title_1, y=attribute_title_2))
+ geom_line(linetype = „twodashed")

ggplot(title_of_data, aes(x=attribute_title_1, y=attribute_title_2))
+ geom_line(linetype = „dotted")
```

Подобно на точковите графики, можем лесно да променяме цвета на геометрията – в този случай на линията – с помощта на опцията цвят, която може да се комбинира с всички типове линии в графиката.

```
ggplot(title_of_data, aes(x=attribute_title_1, y=attribute_title_2))
+ geom_line(color = „color")

ggplot(title_of_data, aes(x=attribute_title_1, y=attribute_title_2))
+ geom_line(linetype = „type“, color = „color")
```

Тъй като линейната графика винаги е приближение на точките с данни, е подходящо да се визуализират самите точки заедно с линейната графика. Това може да се направи много просто чрез комбиниране на геометрията на линията и точката, както следва:

```
ggplot(title_of_data, aes(x=attribute_title_1, y=attribute_title_2))
+ geom_line() + geom_point()
```

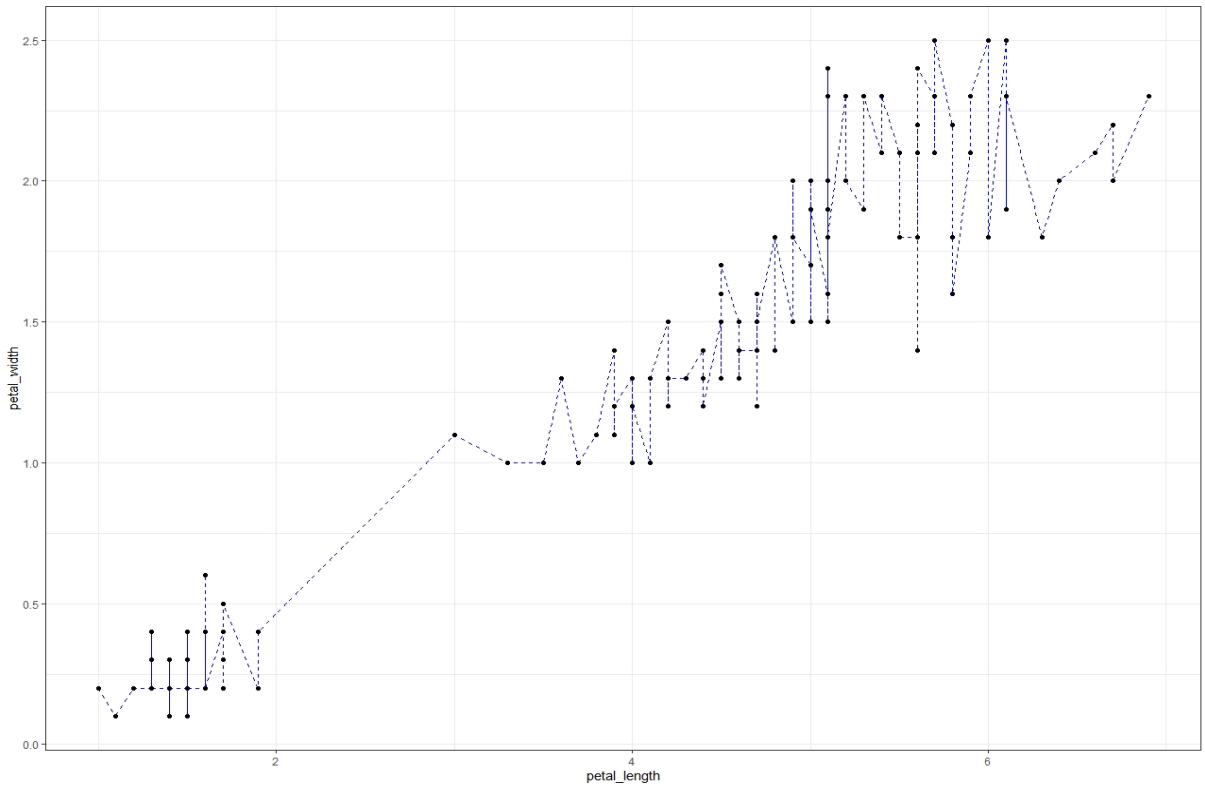
Разбираемо е, че е възможна комбинация от всички тези варианти:

```
ggplot(data, aes(x=attribute_title_1, y=attribute_title_2))
+ geom_line(linetype = „dashed“, color = „blue") + geom_point()
```

Следващият код за набор от данни Iris:

```
> ggplot(data, aes(x = petal_length, y = petal_width)) + geom_line(linetype = "dashed", color = "blue") +
+ theme_bw() + geom_point()
```

генерира следната фигура:



Истинската сила на линейните графики се крие в способността им да визуализират сравнението между стойностите на даден атрибут и набор от атрибути – с други думи, възможността за начертаване на повече от една линия. В примера със синтаксиса по-долу виждаме, че ядрото на функцията *ggplot* съдържа само един атрибут (този, който е поставен на оста x), а ние използваме оста y, за да визуализираме стойностите на attribute2 и attribute3. Тази визуализация се извършва чрез отделни части от кода + *geom_line()*. Представяме и редица комбинации от гореспоменатите опции с този подход.

```
ggplot(title_of_data, aes(x=attribute_title_1))
+ geom_line(aes(y= attribute_title_2))
+ geom_line(aes(y= attribute_title_3))
```

```
ggplot(title_of_data, aes(x=attribute_title_1))
+ geom_line(aes(y= attribute_title_2), color = „color")
+ geom_line(aes(y= attribute_title_3), color = „color")

ggplot(title_of_data, aes(x=attribute_title_1))
+ geom_line(aes(y= attribute_title_2), linetype = „type“, color = „color")
+ geom_line(aes(y= attribute_title_3), linetype = „type“, color = „color")
```

Нека вземем набора от данни за ирис, в който измерихме силни корелации между три атрибута – дължина на чашелистче, дължина на венчелистче и ширина на венчелистче. Колебанията на дължината и ширината на венчелистчетата в зависимост от дължината на чашелистчетата могат да се визуализират с помощта на две линии, които ще бъдат разделени по техния вид или цвят. В нашия случай:

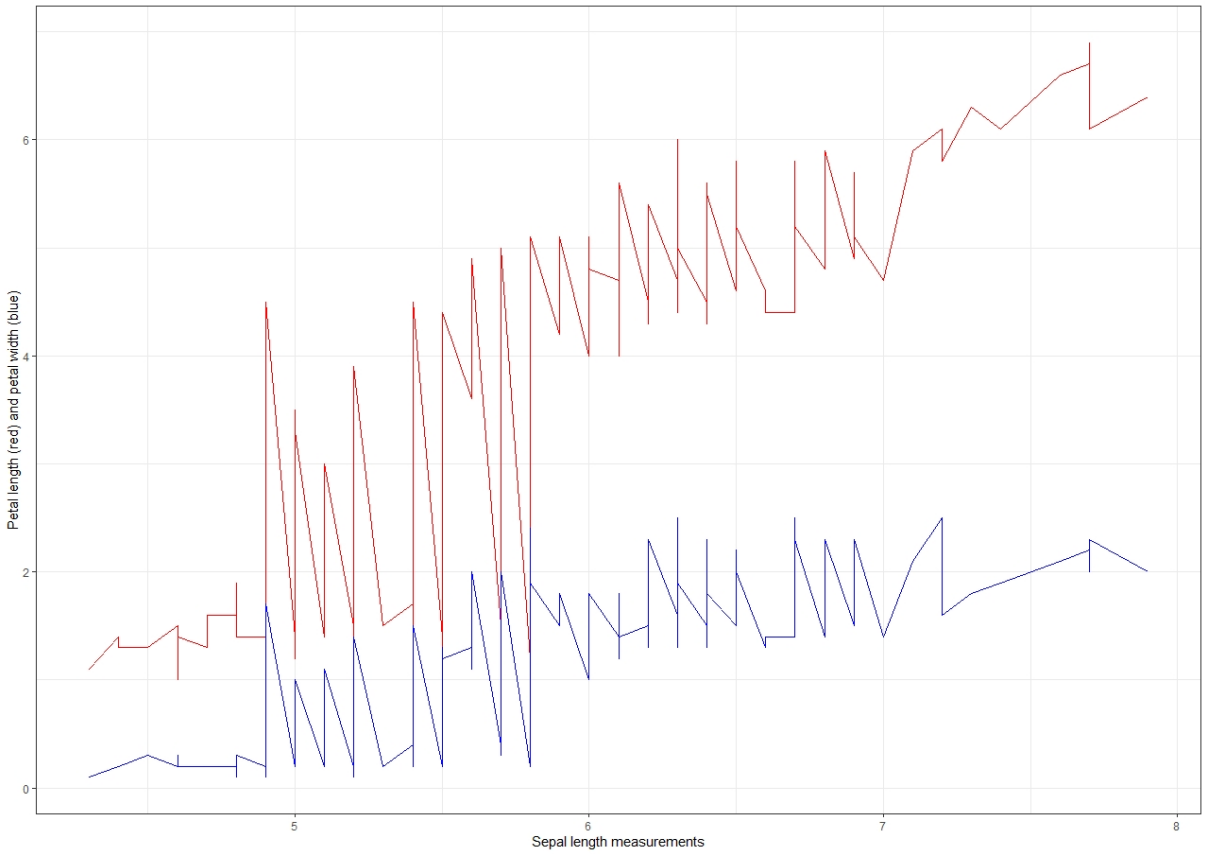
- колебанието на стойностите на дължината на венчелистчетата в зависимост от дължината на чашелистчетата се визуализира с помощта на червена линия,
- колебанието на стойностите на ширината на венчелистчетата в зависимост от дължината на чашелистчетата се визуализира със синя линия.

```
ggplot(data, aes(x=attribute_title_1)) + geom_line(aes
(y= attribute_title_2), linetype = „dotted“, color = „red")
+ geom_line(aes(y= attribute_title_3), color = „blue")
```

Следващият код за набор от данни Iris:

```
> ggplot(data, aes(x = sepal_length)) + geom_line(aes(y = petal_length), color = "red") +
+ geom_line(aes(y = petal_width), color = "blue") + theme_bw() + xlab( "Sepal length measurements") +
+ ylab("Petal length (red) and petal width (blue)")
```

създава следната фигура:



Такава визуализация на данни може да се използва за търсене на тенденции и модели в данните, които след това могат да се използват в по-сложни подходи за анализ на данни, представени в други раздели на този наръчник. Проучвателният анализ на данни има един недостатък – трудно е да се използва при наистина големи масиви от данни, които изискват намаляване на размерността и някои други техники, за да се анализират правилно.

5

РАЗМИТИ (FUZZY) НАБОРИ

Тази част от ръководството е написана от Алжбета Михаликова *от Департамента по компютърни науки, Факултет по природни науки, Университет „Матей Бел“ в Банска Бистрица, Словакия.*

В ежедневието си често използваме неопределени изрази като млад човек, малко сол, малко вдясно, силен вятър, висока температура, ниска цена... Тези изрази нямат точни граници. Те не са ясно дефинирани. Можем да ги наречем **неясни** или **размити**. Първата идея за математическо моделиране чрез размити понятия може да се открие в статията на Лотфи А. Задех [1]:

ZADEH, L. A.: Fuzzy sets. Information and control. Volume 8, pp. 338-353, 1965.

Използването на размити множества се нарича оперирање с естествения език (оперира се не само с числа, но и с други термини на човешката реч). В същото време е разбираемо, че различните хора възприемат различни понятия по различен начин. Основната част от следващия текст се основава на университетския учебник [2] **Размити набори в информатиката** (на словашки език), който е предназначен за студентите от специалност „Приложна информатика“ в катедрата по компютърни науки, Факултет по приложни науки, Университет „Матей Бел“ в Банска Бистрица, Словакия.

Защо да използвате размита (fuzzy) логика?

– Идеята на размитата логика е лесна за разбиране;

– тя е гъвкава система, която е толерантна към неточни данни;

– тя може да работи с опита на експертите;

– тя може да моделира нелинейна система с всякаква сложност;

– може да се използва в стандартно техническо оборудване.

Използване на размити множества

- Експертни системи;
- разпознаване и класификация на обекти;
- теория на управлението и регулирането;
- системи за бази данни;
- математическо моделиране;
- напоследък – обяснимо и невронни мрежи.

Области на приложение на размитите множества

Винаги, когато неопределеността е включена в изчислението. Те често се използват в устройства, които не представляват скъпи уреди от икономическа гледна точка, например **електронни домакински уреди** (перални машини, микровълнови печки, прахосмукачки, самобръсначки, манометри, ...). Можем да ги срещнем и в **сложни и икономически**, както и **устройства за интензивни изчисления**, например

- управление на метрото в Япония – (град Сендай – от 1988 г.) [3];
- управление на доменни пещи (контрол на температурата, която може да се управлява по-ефективно, отколкото с конвенционални регулатори);
- управление на атомни електроцентрали [4]; ...

Пример: Нека имаме обява за работна позиция, в която се изисква възрастта на кандидатите да е в интервала 20-30 години. Нека опишем това множество!

- Какво представлява популацията?
- Можете ли да опишете това множество чрез неговата характерна функция?
- Може ли човек, който утре ще навърши 31 години, да отговори на тази обява?

Какво представлява Вселената?

Това множество обикновено се обозначава с буквата X. Вселената трябва да бъде всеки интервал с постижими стойности. Например в нашата задача тя може да бъде $\mathbb{X} = \langle 0, \infty)$.

Можете ли да опишете това множество чрез неговата характеристична функция?

Характеристичната функция е функция, която придава число 1 на елементите, които принадлежат към разглежданото множество, и от друга страна, придава число 0 на елементите, които не принадлежат към разглежданото множество. Тази функция обикновено се обозначава с буквата χ. За нашия пример функцията има следния запис

$$\chi_A: \mathbb{X} \to \{0,1\} \qquad \chi_A(x) = \begin{cases} 1, & \text{if } 20 \leq x \leq 30, \\ 0, & \text{if } 0 \leq x < 20 \text{ or } x > 30. \end{cases}$$

Може ли човек, който утре ще навърши 31 години, да отговори на тази обява?

НЕ! – защото в момента, в който някой прочете отговора му, той вече няма да отговаря на изискваното условие.

Пример: Нека имаме подобна ситуация: В една обява има изискване, че фирмата търси млади хора.

- Променила ли се е ситуацията в сравнение с предишния пример?
- Каква е популацията?
- Как можем да опишем тази съвкупност?
- Може ли човек, който утре ще има 31-ви рожден ден, да отговори на тази обява?

Променила ли се е ситуацията в сравнение с предишния пример?

ДА! – множеството на младите хора представлява **размито множество.** Няма рязка граница за възрастта на хората, които принадлежат към това множество!

Какво представлява популацията?

Популацията на размитото множество трябва да бъде всеки интервал с достижими стойности. Тя може да бъде същото множество, каквото е било при класическото множество, напр. $\mathbb{X} = \langle 0, \infty)$.

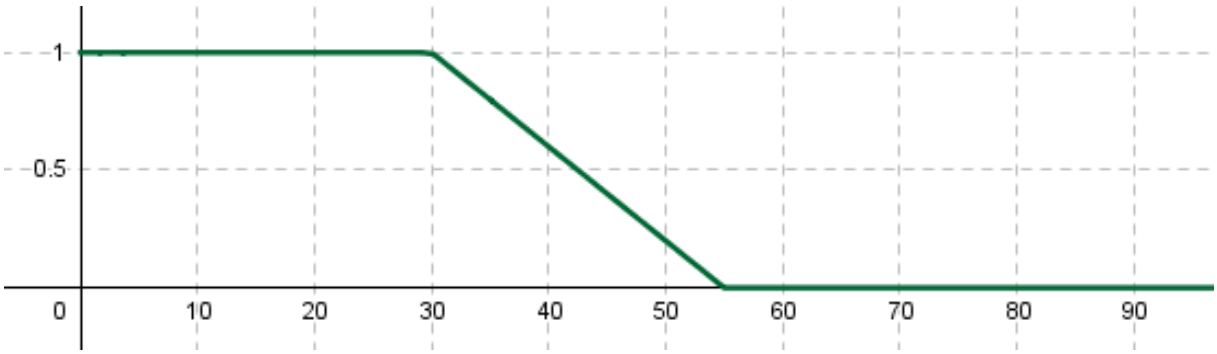
Как можем да опишем този набор?

Например 20-годишният човек със сигурност е млад, следователно му се приписва степен на младост, равна на 1. По същия начин 30-годишният човек може да се счита за млад със сигурност, следователно му се приписва степен на младост, равна на 1, но на 35-годишния можем да припишем степен на младост 0,8, ... За да опишем размитите множества, използваме т.нар. **Функции на принадлежност**. Те се означават с буквата μ. С помощта на тази функция трябва да присвоим някаква стойност от

единичния интервал (например от интервала ⟨0,1⟩ на всеки елемент на вселената. Първо, нека разгледаме стойностите на възрастта, която разгледахме точно като възраст на млад човек. Като пример, тези стойности могат да бъдат от интервала ⟨0, 30⟩. Функцията на принадлежност придава на тези стойности стойност, равна на 1. Сега нека разгледаме стойностите на възрастите, които считахме за точно не младежка възраст. Пример за такива стойности могат да бъдат стойности, по-големи от 55. На тези стойности функцията за членство присвоява стойност, равна на 0. За стойностите от интервала (30, 55⟩ очакваме, че стойностите на принадлежност към множеството на младите хора ще се понижават последователно от 1 до 0 (вж. Фигура 1).

Означете размитото множество от млади хора с B. Предписанието на описаната функция е

$$\mu_B: \mathbb{X} \to \langle 0, 1\rangle \qquad \mu_B(x) = \begin{cases} 1, & \text{if } x \in \langle 0, 30\rangle, \\ \frac{1}{25}(55 - x), & \text{if } x \in (30, 55\rangle, \\ 0, & \text{if } x > 55. \end{cases}$$



Фигура 1: Функция на принадлежност на размитото множество „Млади хора“.

Бележки:

Термините размито множество и функция на принадлежност често се считат за еквивалентни.

Стойността, която се приписва на определена входна стойност, се нарича **степен на принадлежност** или **принадлежностна степен**.

Нека размитото множество, определено по формула (1), описва множеството от млади хора. Можем ли да определим степента на принадлежност на хората, които са на 20, 35 и 45 години? Като използваме формула (1), получаваме

μ_B (20) = 1, т.е., 20 годишен човек е млад със сигурност;

μ_B (35) = 0.8, т.е., 35 години човек е млад със степен 0.8;

μ_B (40) = 0.6, т.е., 40 години човек е млад със степен 0.6.

Може ли човек, който утре ще навърши 31 години, да отговори на тази обява?

ДА! – Защото неговата степен на принадлежност към размитото множество μ_B е равна на 0.96 (тъй като $\mu_B(31) = 0.96$). Тази стойност представлява висока степен на принадлежност към размитото множество на младите хора.

Бележка:

Съществуват няколко вида функции на принадлежност на размити множества. Ще покажем някои от тях в следващия пример.

Пример: Нека моделираме размитото множество C от реални числа, което представлява понятието „приблизително 7“.

- *Какво представлява популацията?*
- *Как можем да опишем свойствата на тази съвкупност?*

С използването на този термин „приблизително 7“ можем да си представим изречения като

Навън е около 7 градуса по Целзий.

или

Похарчих приблизително 7 евро в магазина.

Какво представлява популацията?

Като популация обикновено разглеждаме най-голямото възможно множество. В този пример това може да бъде цялото множество от реални числа, напр. $\mathbb{X} = \mathbb{R}$.

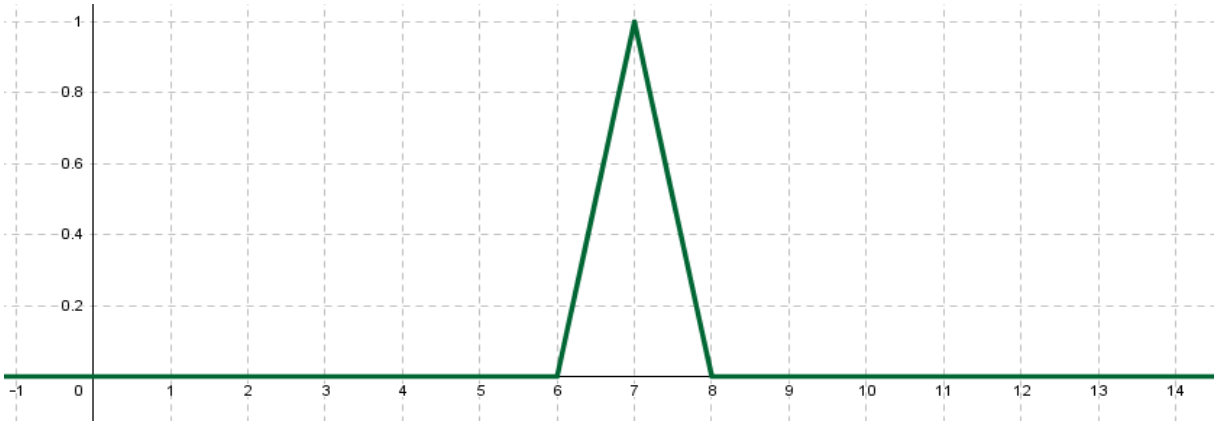
Как можем да опишем свойствата на това множество?

Нека обозначим това множество като множеството C. За функцията на принадлежност на това множество, μ_C, може да са налице две условия:

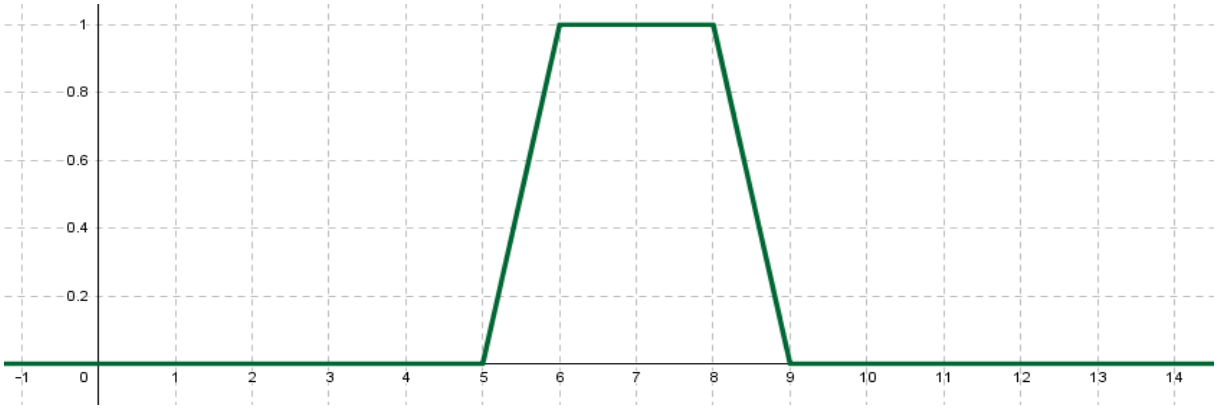
μ_C (7) = 1,

с увеличаване на разликата $|x - 7|$ стойностите му трябва да намаляват до нула.

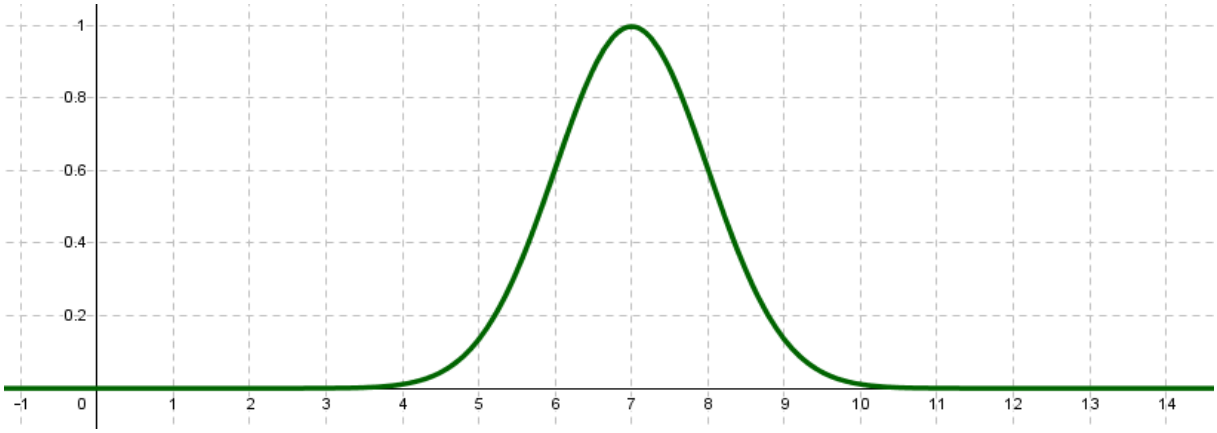
След това някои от възможностите са показани на Фигура 2, Фигура 3 и Фигура 4.



Фигура 2: Триъгълна функция на принадлежност, представяща стойността „приблизително 7“.



Фигура 3: Трапецовидна функция на принадлежност, представяща стойността „приблизително 7“.



Фигура 4: Друга функция на принадлежност, представяща стойността „приблизително 7“.

Видове функции на принадлежност

На фигури 2-4 видяхме, че функцията на принадлежност може да има много различни форми. Ще покажем някои от тях, които са дефинирани в софтуерния инструмент MATLAB.

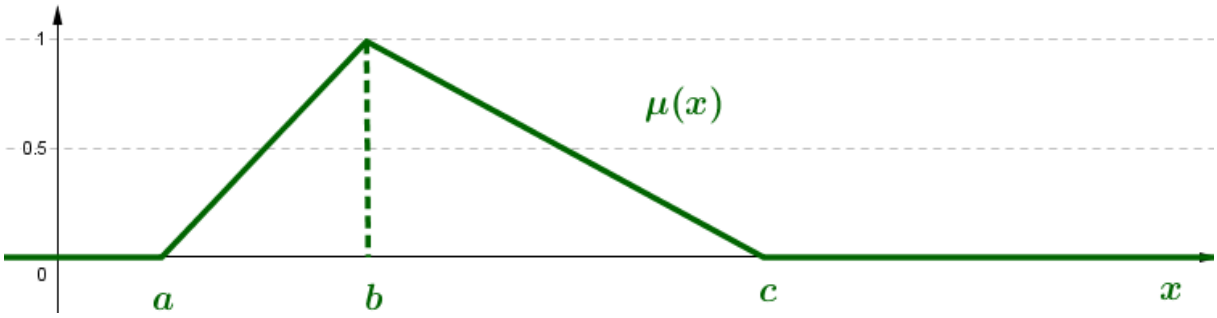
Линейни функции на принадлежност

Линейните функции на принадлежност представляват най-простия вид функции на принадлежност. Те се изграждат с помощта на части от прави линии. Те се разделят на две основни групи:

- *триъгълна;*
- *трапецовидна.*

Триъгълна функция на принадлежност

Триъгълната функция на принадлежност се състои от четири части (вж. фигура 5). Първата част приписва на входните стойности изходна стойност, равна на нула (интервал (-∞, a⟩ на фигура 5). Втората част е линейно нарастваща от стойност 0 до стойност 1 (интервал (a; b⟩ на фигура 5). Третата част е линейно намаляваща от стойност 1 до стойност 0 (интервал (b, c⟩ на фигура 5). Последната част отново присвоява на входните стойности изходна стойност, равна на 0 (интервал (c, ∞) на фигура 5). По принцип тази функция на принадлежност се описва с 3 параметъра a, b, c. В софтуера MATLAB тя се означава като **trimf**, а за параметрите се използва означението ***[a b c]***. Забележете, че триъгълната функция на принадлежност достига изходна стойност, равна на 1, само за един вход (по-конкретно за входна стойност *b*).

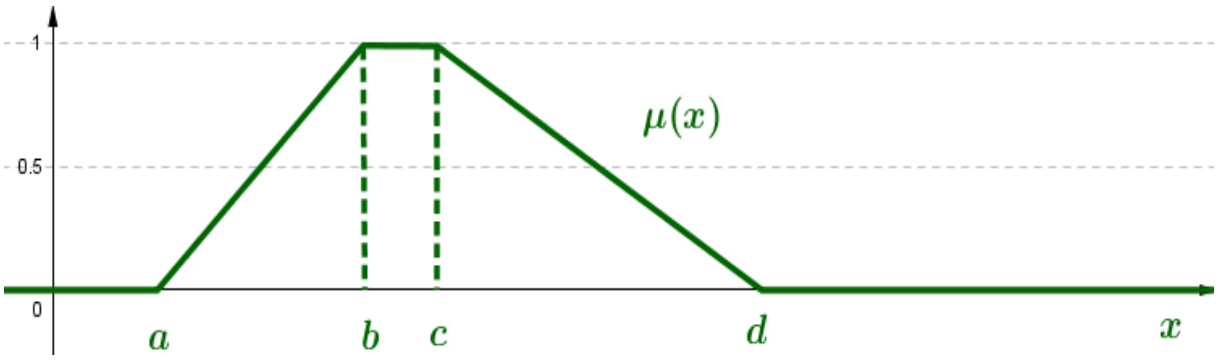


Фигура 5: Обща триъгълна функция на принадлежност.

Трапецовидна функция на принадлежност

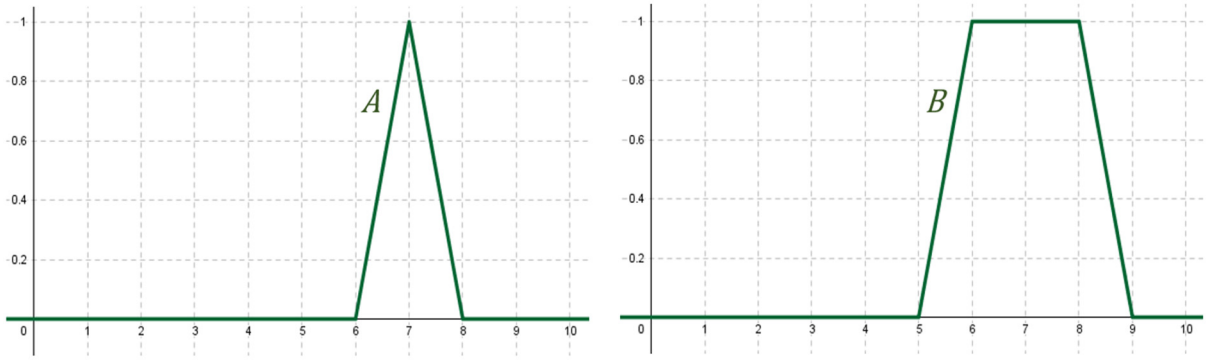
Трапецовидната функция на принадлежност се състои от пет части (вж. фигура 6). За разлика от триъгълната функция на принадлежност, тази функция се състои от интервала от входни стойности, които достигат изходна стойност, равна на 1. В общия случай тази функция на принадлеж-

ност се описва с 4 параметъра a, b, c, d. В софтуера MATLAB тя се означава като **trapmf**, а за параметрите се използва означението ***[a b c d]***.



Фигура 6: Обща трапецовидна функция на принадлежност.

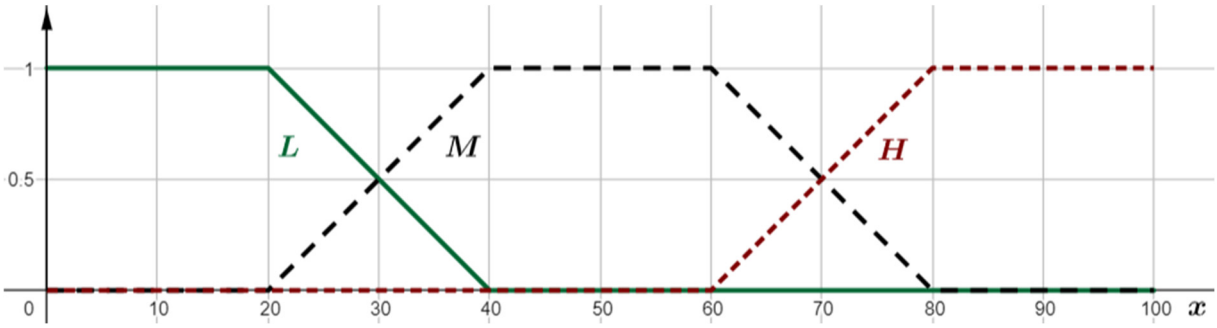
Пример: Запишете в MATLAB-а размитите множества A, B, които са показани на фигура 7:



Фигура 7: Размити множества A и B от примера.

Размитото множество *A* се представя с помощта на триъгълна функция на принадлежност. Нейното записване в програмата MATLAB е *A [6 7 8]*. Размитото множество *B* се представя с помощта на трапецовидна функция на принадлежност. Нейното записване в програмата MATLAB е *B [5 6 8 9]*.

Пример: В реалния живот за някои наблюдавани явления често използваме термините ниско, средно и високо явление. За температурата на водата можем да говорим за ниска, средна и висока температура (вж. фигура 8). Докато терминът средна температура (функция M) би могъл да се опише с трапецовидна функция на принадлежност, както бе споменато в предишния текст, стойностите ниска температура (функция L) и висока температура (функция H) са специфични в този смисъл, че трябва да се опишат с помощта на асиметрични функции на принадлежност. Как можем да запишем предписанието на тези функции в софтуера MATLAB?



Фигура 8: Размити множества L, M и H от примера.

За да опишем някакво размито множество в софтуера MATLAB, можем да използваме и параметри, които не принадлежат към популацията на изследваната променлива. И така, на първата стъпка трябва да определим популацията на понятието „температура на водата“. Тя е $\mathbb{X} = \langle 0, 100 \rangle$. Тогава можем да напишем

$$L\ [-20\ -10\ 20\ 40], \quad M\ [20\ 40\ 60\ 80], \quad H\ [60\ 80\ 110\ 120].$$

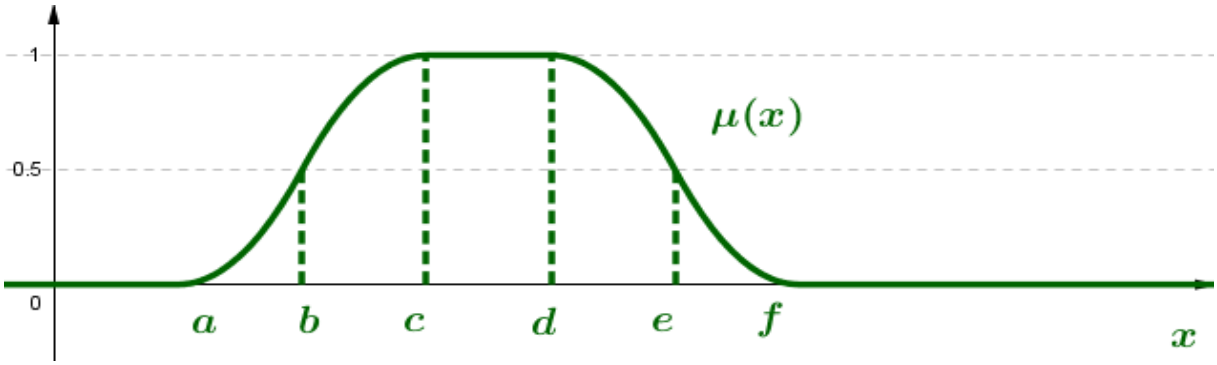
Функции на принадлежност, базирани на полиноми

Този тип функции се изграждат с помощта на полиномни (квадратични) функции. Те се разделят на три основни групи:

- Pi крива;
- S крива
- Z крива.

Функция на принадлежност от тип Pi

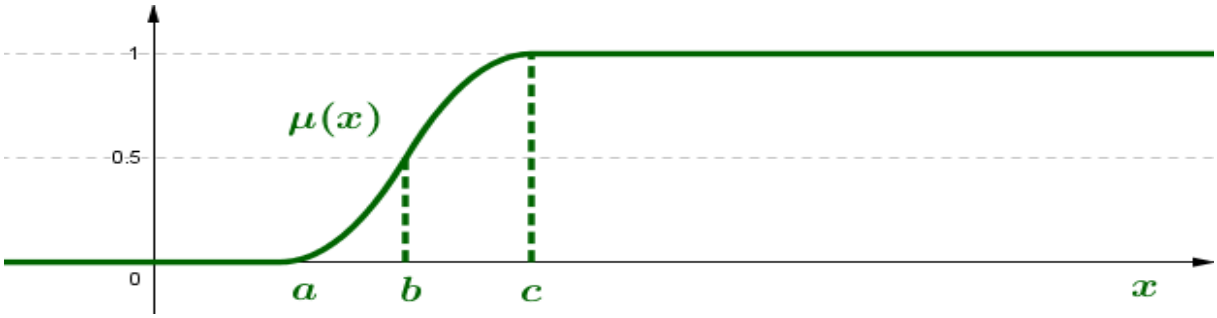
Функцията на принадлежност от тип Pi се определя от 6 параметъра *a,b,c,d,e,f* (вж. фигура 9). Тя има четири части, които се определят от квадратични функции (интервали $\langle a, b \rangle$; $\langle b, c \rangle$; $\langle d, e \rangle$; $\langle e, f \rangle$), две части, в които стойността е 0 (интервали $(-\infty, a \rangle$; (f, ∞)) се присвоява на всяка входна стойност и една част, в която на всяка входна стойност се присвоява стойност 1 (интервал $\langle c, d \rangle$). В софтуера MATLAB той се обозначава като **pimf.**



Фигура 9: Функция на принадлежност от тип Pi.

Функция на принадлежност от тип S

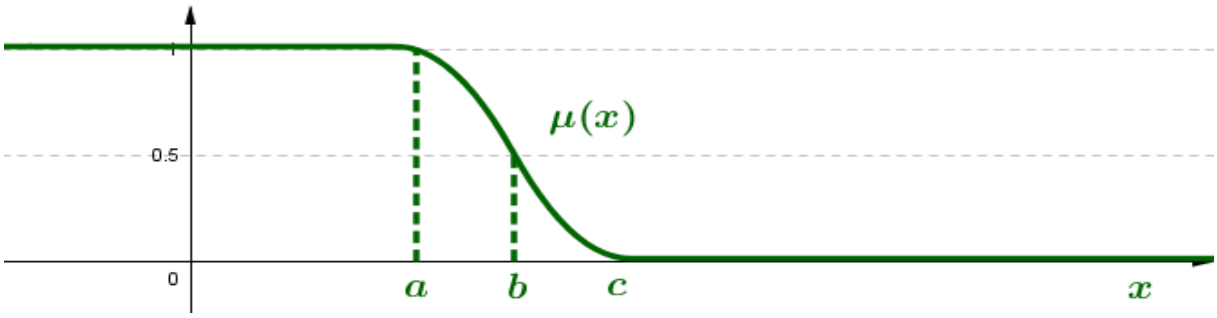
Функцията на принадлежност от тип S се определя от 3 параметъра *a,b,c* (вж. Фигура 10). Има две части от нея, които се определят от квадратични функции (интервали $\langle a, b\rangle$; $\langle b, c\rangle$), една част, в която на всяка входна стойност се присвоява стойност 0 (интервал $(-\infty, a\rangle$), и една част, в която на всяка входна стойност се присвоява стойност 1 (интервал $\langle c, \infty)$). В софтуера MATLAB тя се обозначава като **smf**.



Фигура 10: Функция на принадлежност от тип S

Функция на принадлежност от тип Z

Функцията на принадлежност от тип S се определя от 3 параметъра *a, b, c* (вж. Фигура 11). Тя има две части, които се определят от квадратични функции (интервали $\langle a, b\rangle$; $\langle b, c\rangle$), една част, в която на всяка входна стойност се присвоява стойност 1 (интервали $(-\infty, a\rangle$), и една част, в която на всяка входна стойност се присвоява стойност 0 (интервал $\langle c, \infty)$). В софтуера MATLAB тя се обозначава като **zmf.**



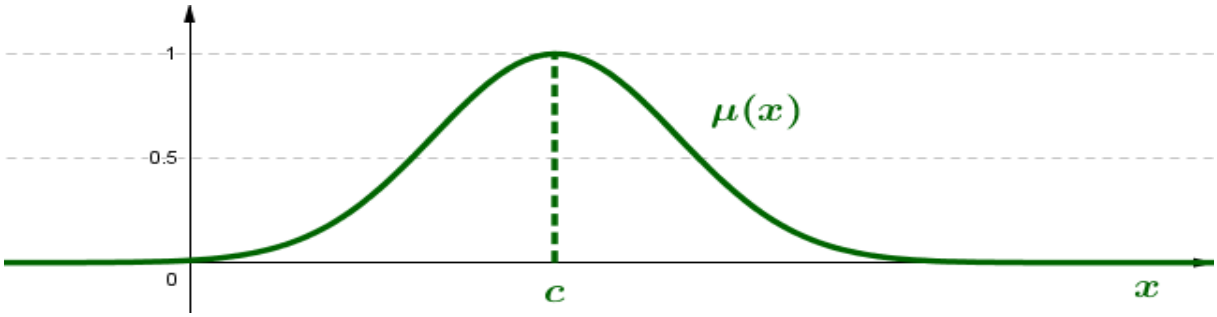
Фигура 11: Функция на принадлежност от тип Z.

Бележка:

Функциите на принадлежност от тип S и Z представляват асиметрични функции на принадлежност. Те могат да се използват за моделиране на ниски и високи стойности на променливите.

Функция, базирана на статистическа основа

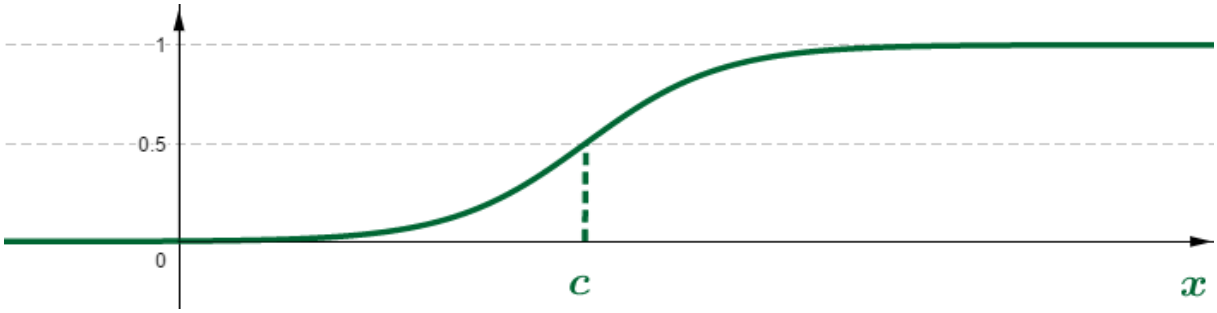
Ако разполагаме с голям набор от данни, можем да ги обработим, като използваме статистически подход. **Гаусовите функции на принадлежност (gaussmf)** се извличат от класическата крива на Гаусовото разпределение, която има два параметъра *c,σ* (вж. Фигура 12), където *c* представлява средната стойност, а *σ* – стандартното отклонение на данните.



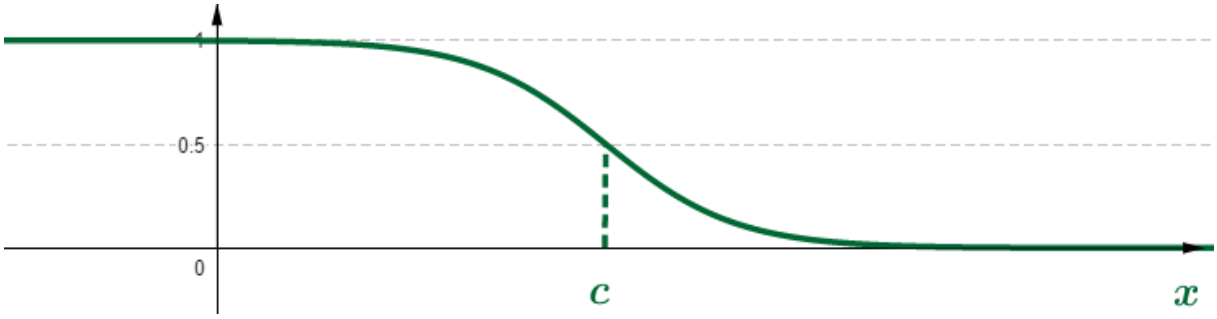
Фигура 12: Гаусова функция на принадлежност.

Сигмоидални функции на членство

Гаусовите функции на принадлежност не могат да се определят като **асиметрични функции на принадлежност**. По тази причина се използват **сигмоидалните функции на принадлежност (sigmf)** с два параметъра *a, c* (вж. Фигура 13 и Фигура 14). След това параметрите *a, c* отново се получават чрез използване на статистически подход.



Фигура 13: Сигмоидална функция на принадлежност, когато a > 0.



Фигура 14: Сигмоидална функция на принадлежност, когато a < 0.

Бележки:

Ще използваме **триъгълни** и **трапецовидни функции на принадлежност**. В реалните приложения често се използват и гаусови и сигмоидални функции на принадлежност, като параметрите им се избират с помощта на статистически анализ на данните.

В реалния живот обикновено **първо искаме от експерта** да опише проблема чрез подходящи функции. След това, **на втория етап**, **обикновено определяме параметрите на функциите** с помощта на (статистическа) обработка на голяма група данни.

За да работим с размити множества, трябва да дефинираме основните операции върху размити множества – пресичане, обединяване и **допълване**. По подобен начин, тъй като има много видове функции на принадлежност, **се дефинират и няколко вида операции върху размити множества**. Ще спомена само така наречените **стандартни операции върху размити множества**, които са предложени от професор Задех.

Определение (стандартно пресичане)

Нека $\mathbb{X}$ е вселената и A, B са размити набори. **Стандартно пресичане на две размити множества** A, B е размитото множество A ∩ B с функция на принадлежност

$$\mu_{A\cap B}(x) = \min\big(\mu_A(x), \mu_B(x)\big).$$

Определение (стандартно обединение)

Нека $\mathbb{X}$ е популация и A, B са размити съвкупности (набори). Стандартно обединение на два размити набора A, B е размит набор A ∪ B с функцията на принадлежност

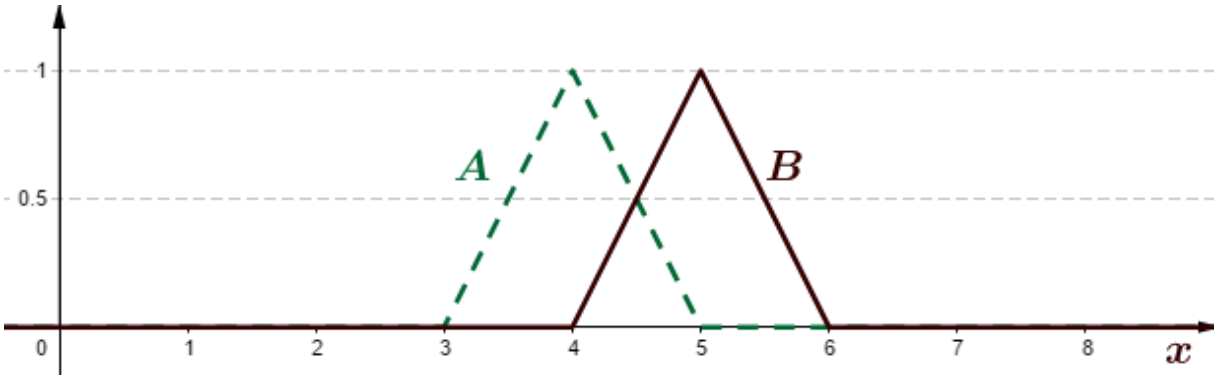
$$\mu_{A\cup B}(x) = \max\big(\mu_A(x), \mu_B(x)\big).$$

Определение (стандартно допълнение)

Нека $\mathbb{X}$ е популация и A е размито множество. **Стандартното допълнение на размитото множество** A е размитото множество $\bar{A}$ с функция на принадлежност

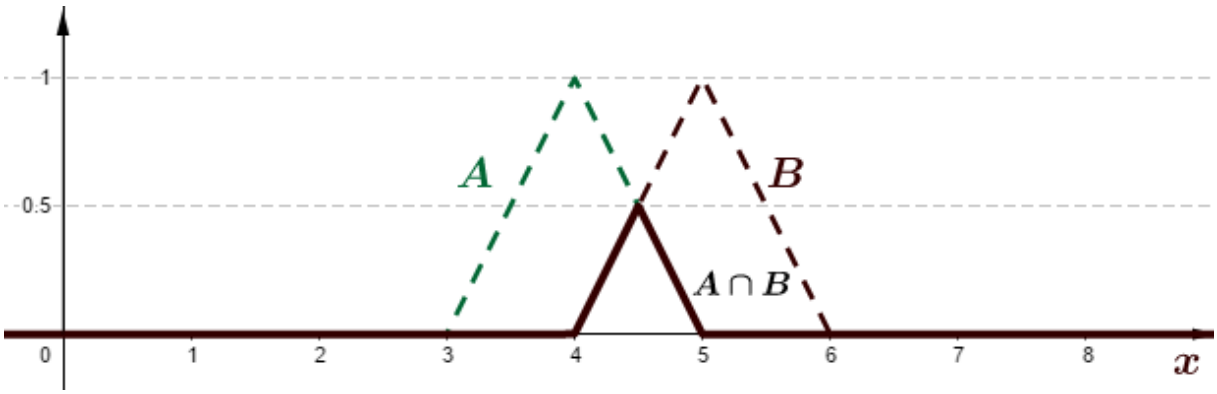
$$\mu_{\bar{A}}(x) = 1 - \mu_A(x).$$

Пример: Има две размити множества A, B, представени на фигура 15. Като използвате предишните дефиниции, определете графично пресечната точка и съюза на размитите множества A, B, а също и допълнението към размитото множество A.



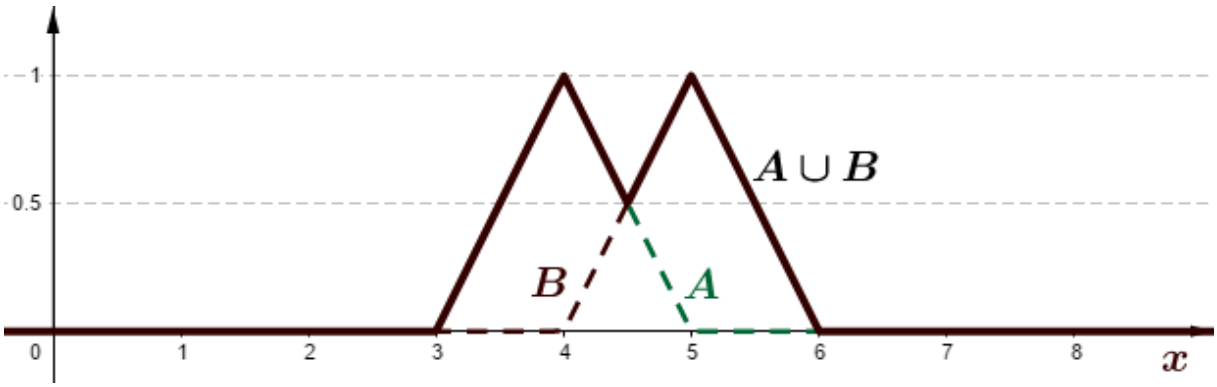
Фигура 15: Размити множества A, B от примера.

Стандартно пресичане на две размити множества *A*, *B* е размитото множество $A \cap B$ с функция на принадлежност $\mu_{A\cap B}(x) = \min(\mu_A(x), \mu_B(x))$. Решението е показано на фигура 16.



Фигура 16: Стандартно пресичане на размити множества A, B от примера.

Стандартно обединение на две размити множества *A*, *B* е размитото множество $A \cup B$ с функция на принадлежност $\mu_{A\cup B}(x) = \max(\mu_A(x), \mu_B(x))$. Решението е показано на фигура 17.



Фигура 17: Стандартно обединение на размити множества *A*, *B* от примера.

6

РАЗМИТИ РАЗСЪЖДЕНИЯ

Тази част от ръководството е написана от Алжбета Михаликова *от Департамента по компютърни науки, Факултет по природни науки, Университет „Матей Бел“ в Банска Бистрица, Словакия.*

Размитото разсъждение е процес, при който извеждаме следствията въз основа на информация, която се състои от неясни термини. Например, в реалния живот често използваме правила като

"Ако навън е студено, ще се облека с топлите дрехи."

Тези правила са резултат от нашите наблюдения, учене, разсъждения и т.н. В размитото разсъждение използваме така наречените IF – THEN размити правила, които имат следната форма:

$\underbrace{IF\ \langle ... \rangle}$ $\underbrace{THEN \langle ... \rangle}$

= антецедент *= последващ*

(предположение) *(заключение)*

За нашите нужди ще модифицираме правилото

"Ако навън е студено, ще се облека с топлите дрехи."

Във формата:

"IF the Temperature outside is low, THEN the Dress is warm. "

"АКО (*IF*) температурата навън е ниска, ТОГАВА (*THEN*) дрехите са топли."

Думите „**Температура**“ и „**Дрехи**“ се наричат **лингвистични променливи,** затова ги пишем с главна буква. Стойностите „**ниско**“ и „**топло**“ се наричат **стойности на лингвистичните променливи**. Лингвистичните променливи, които се намират в антецедентната част, се наричат **входни лингвистични променливи**. Лингвистичните променливи, които са в следствието, се наричат **изходни лингвистични променливи**. Като използваме операциите конюнкция (AND), дизюнкция (OR) и отрицание (NOT), можем да създадем по-сложни правила, например:

"АКО (IF) Температурата навън е ниска И (AND) Облачността е висока, ТОГАВА(THEN) Дрехите са топли."

Ако дефинираме всички необходими правила, получаваме набор такива, който се нарича **база от правила**. Съществуват различни подходи за работа с правилата от базата правила. Ние ще обсъдим и използваме един от тях – **метода на Сугено**.

Метод на Сугено

Автори на този метод са **Т. Такаги**, **М. Сугено** и **Г. Канг** [5]. Те го предлагат през 1985 г. Този метод е предназначен за моделиране на проблеми, при които е възможно да се опише зависимостта между входните и изходните променливи чрез функция, която е нелинейна, но има и части, които са линейни.

Методът на Сугено е използван за първи път при моделирането на проблем с **паркирането на автомобили**. Днес той се използва за **апроксимация на данни** чрез нелинейни функции, **в класификацията**, **регулирането** и **контрола**, **вземане на решения**, **експертните системи**, ...

В метода на Сугено стойностите на **входните променливи** се описват чрез **функции на принадлежност**. Те се проектират от експерта. **Изходните променливи** се **описват с функции**, които могат да бъдат или **константни функции**, или **линейни функции**, или **полиномни функции от всякаква степен.**

Правила на Сугено с постоянни изходни функции – изходната променлива на всяко правило се описва с функция, която е постоянна. Обикновено правилото има формата

$$R_j\colon\ IF\ X_1\ is\ A_{1j}\ \ AND\ \ X_2\ is\ A_{2j}\ \ AND\ \ \dots\ AND\ \ X_n\ is\ A_{nj}, THEN\ Y\ is\ b_j.$$

Правила на Сугено с линейни изходни функции – изходната променлива на всяко правило се описва с линейна функция. Обикновено правилото има формата

$$R_j\colon\ IF\ X_1\ is\ A_{1j}\ \ AND\ \ \dots\ AND\ \ X_n\ is\ A_{nj},\qquad THEN\ Y\ is\ a_{1j}x_1 + \cdots + a_{nj}x_n + b_j,$$

където $a_{1j}, \dots, a_{nj}, b_j$ са реални числа.

Правила на Сугено с полиномни изходни функции – изходната променлива на всяко правило се описва с полиномна функция от произволна степен.

$$R_j\colon\ IF\ X_1\ is\ A_{1j}\ \ AND\ \ \dots\ AND\ \ X_n\ is\ A_{nj}, THEN\ Y\ is\ a_{1j}{x_1}^{m_1} + \cdots + a_{nj}{x_n}^{m_n} + b_j,$$

където $a_{1j}, \dots, a_{nj}, b_j$ са реални числа и $m_1, \dots, m_n$ са естествени числа.

Пример: Правило на Сугено с постоянна изходна функция

Трябваше да оценим учениците с помощта на метода на Сугено. Той може да бъде описан чрез правила от типа

$$R\colon\ IF\ Presentation\ is\ high\ AND\ Test\ points\ value\ is\ high,$$

$$THEN\ Evaluation\ is\ equal\ to\ 1\ (=A).$$

Пример: Правила на Сугено с линейни изходни функции

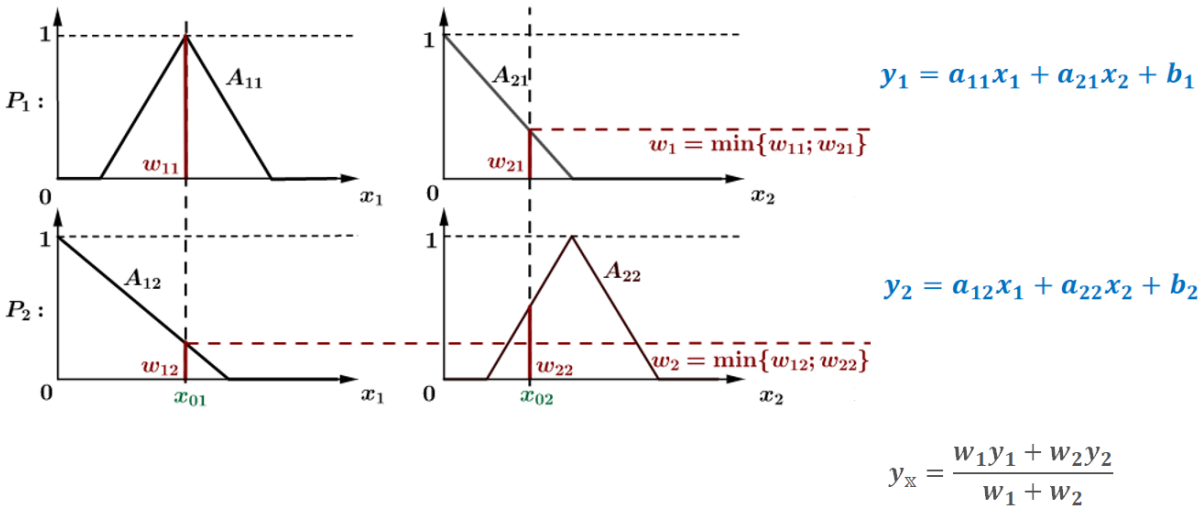
Знаем, че за някои стойности (например за малки стойности) на положението на автомобила, автомобилът ще се движи по права линия, чието предписание може лесно да се определи. Правилото има следния вид

$$R\colon\ IF\ Position\ value\ is\ low, THEN\ Line\ prescription\ is\ 3.25x + 2.5\ .$$

Нека имаме базата правила с k правила. Нека има $\mathbb{x} = (x_1, x_2, \dots, x_n)$. Нека всяко правило има изходна функция във вида: $y_j = a_{1j}{x_1}^{m_1} + a_{2j}{x_2}^{m_2} + \cdots + a_{nj}{x_n}^{m_n} + b_j$. След това крайният резултат y_x се изчислява по формулата

$$y_{\mathbb{x}} = \frac{\sum_{j=1}^{k} w_j y_j}{\sum_{j=1}^{k} w_j}$$

където w_j е теглото на *j*-то правило (виж Фигура 18).



Фигура 18: Краен резултат при метода на Сугено с две входни променливи и две правила.

Бележка относно начина на получаване на теглата:

Нека имаме правило R_j с *n* входни променливи $(x_1, x_2, \dots, x_n)$. Първо, изчисляваме теглата w_{ij} като пресечна точка между стойността на измере-

ната входна величина x_i и съответната функция на принадлежност A_{ij}. Второ, изчисляваме теглото w_j, като използваме следната формула

$$w_j = \min_i w_{ij} .$$

Как правилно да проектираме „размити“ правила?

- *Можем да поискаме от експерт да опише знанията си, като използваме съответните функции.*
- *Можем да определим стойностите на параметрите на функциите, като обработим голямо количество известни данни.*

Този метод има няколко наименования, например **Метод на Сугено**, **Такаги-Судено размита система за изводи, Такаги-Судено регулатор**,... Тези имена представляват един и същ метод. Използваното име е свързано с областта, в която се използва методът.

Ще демонстрираме използването на метода Сугено в две различни области

- *при класифицирането на данни;*
- *в апроксимацията на данни.*

И в двата случая ние ще бъдем експертите, които ще разработват правилата на дадената система [2], [6], [7]. За създаване на стойностите на **входните лингвистични променливи** ще използваме най-простите видове функции на принадлежност – **линейни функции на принадлежност**. За създаване на стойностите на **изходните лингвистични променливи** ще използваме **константни функции за класификация** и **линейни функции за апроксимация**.

7

ИЗПОЛЗВАНЕ НА МЕТОДА СУГЕНО ЗА КЛАСИФИКАЦИЯ НА ДАННИ

Тази част от ръководството е написана от Алжбета Михаликова от Департамента по компютърни науки, Факултет по природни науки, Университет „Матей Бел“ в Банска Бистрица, Словакия.

В тази част на наръчника ще работим с набора от **данни IRIS** (виж Приложение А). Припомнете си, че наборът от данни за ириси се състои от 150 проби на цветя на ириси. За всяко цвете разполагаме с **4 основни атрибута** – дължината и ширината на чашелистчетата и дължината и ширината на венчелистчетата, в сантиметри (или милиметри). От друга страна, можем да класифицираме цветовете в три класа, съответстващи на три вида ириси (**Iris Setosa, Iris Virginica** и **Iris Versicolor**).

В тази част ще комбинираме обработката на данните в софтуер Excel и софтуер MATLAB.

Пример: Класифицирайте данните от масива от данни Iris в подходящ брой класове с помощта на метода на Сугено. (Решението на този пример може да бъде намерено в Приложение Б.)

Първо, нека се опитаме да отговорим на следните въпроси:

*1. Колко **входни променливи** има в набора от данни Iris?*

*2. Какво ще използваме, за **да опишем входните променливи**?*

*3. Какъв **тип размити функции на принадлежност** ще използваме?*

*4. Какъв ще бъде **изходът**?*

*5. Какво ще използваме, за **да опишем изходните променливи**?*

*6. **Какъв тип правила** ще използваме?*

*7. **Напишете пример** за едно правило!*

Второ, изтеглете набора от данни Iris от уебстраницата и го копирайте във файла на Excel. Нека **отбележим** първите 50 обекта с **червен цвят**, следващите 50 обекта със **син цвят**, а останалите със **зелен цвят**. В Excel създайте четири независими листа и копирайте оцветената таблица във всеки от тях. В първия лист подредете стойностите (от най-малката до най-голямата) според първата колона. По същия начин подредете стой-

ностите във всеки лист според една от колоните. За да моделираме входните променливи, ще използваме **трапецовидни функции**. Определете стойностите на параметрите на входните променливи от тези данни и ги попълнете в следните таблици.

Таблица1: Параметри на входните променливи

ВХОД 1

Име	**Параметри**
Вселена	
Червена	
Синя	
Зелена	

ВХОД 2

Име	**Параметри**
Вселена	
Червена	
Синя	
Зелена	

ВХОД 3

Име	**Параметри**
Вселена	
Червена	
Синя	
Зелена	

ВХОД 4

Име	**Параметри**
Вселена	
Червена	
Синя	
Зелена	

Трето, определяне на стойностите на изходните параметри. Попълнете следната таблица с правилните стойности, ако за изходната лингвистична променлива използваме константни функции.

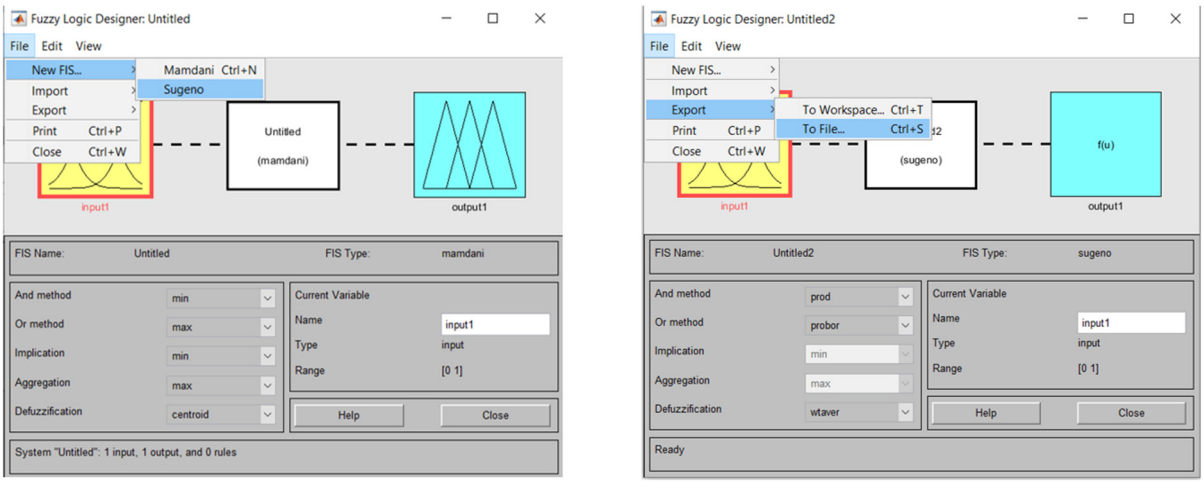
Таблица 2: Параметри на изходната променлива

ИЗХОД:

Име	**Параметри**
Вселена	
Червена	
Синя	
Зелена	

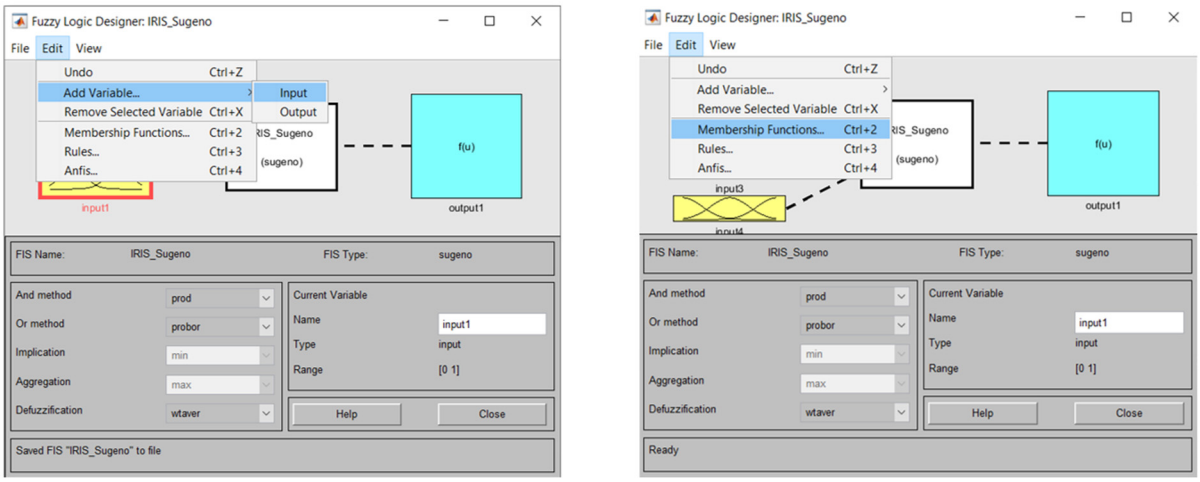
Четвърто, предложете броя на правилата и ги запишете в правилната форма.

Правила:



(а) (б)

Фигура 19: Отваряне на нов Sugeno FIS (а) и преименуване/запазване на FIS (б) в софтуера MATLAB.

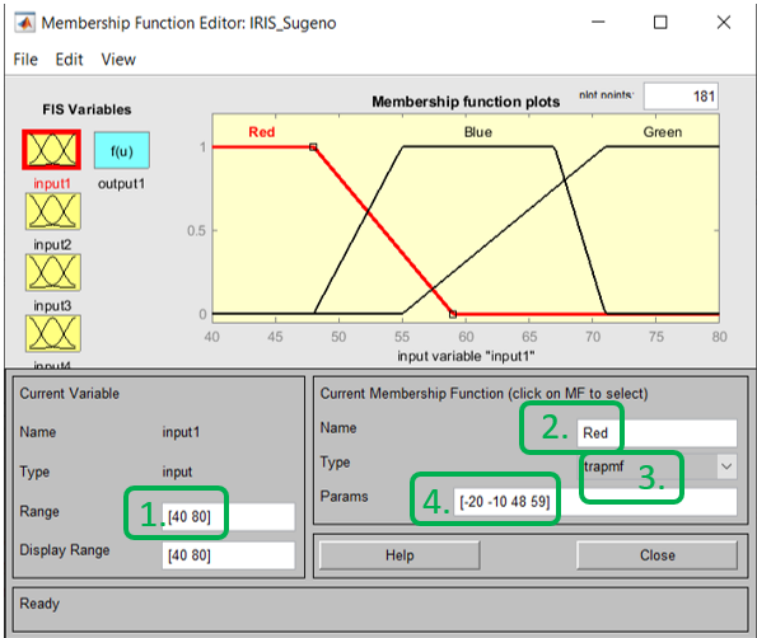


(а) (б)

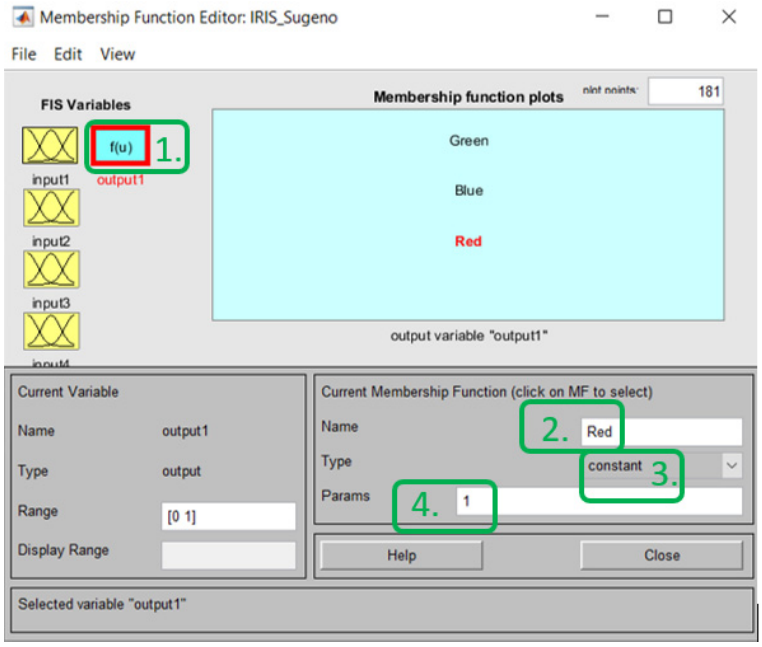
Фигура 20: Добавяне на нови променливи (а) и редактиране на функции на принадлежност (б) в софтуера MATLAB.

Сега ще обработим получените стойности в софтуера MATLAB. Отворете софтуера MATLAB и запишете командата **fuzzy** в командния прозорец. Тази команда отваря Fuzzy Logic Designer (Дизайнер на размита логика). Ще използваме метода на Сугено (Sugeno fuzzy inference system = Sugeno FIS), затова трябва да отворим този тип FIS (виж Фигура 19а). Можем да преименуваме и да запазим тази FIS, например като файл **IRIS_Sugeno** (виж Фигура 19б). Сега трябва да имаме четири входни лингвистични променливи – добавяме три нови входни променливи (виж Фигура 20а) и след това трябва да редактираме параметрите на функциите на принадлеж-

ност (виж Фигура 20б). Сега за всяка входна променлива ще променим обхвата на променливата стъпка по стъпка, ще добавим името на функциите на принадлежност, ще променим вида на функциите на принадлежност и ще добавим параметрите на всяка функция на принадлежност (използвайте Таблица 1). Тези стъпки са показани на фигура 21.



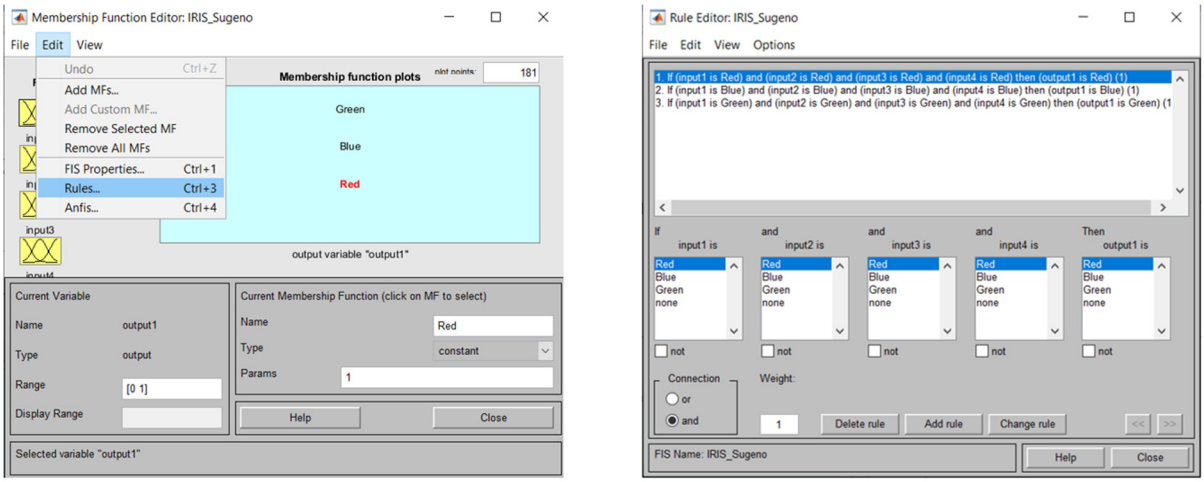
Фигура 21: Промяна на параметрите на входната функция на принадлежност в софтуера MATLAB.



Фигура 22: Промяна на параметрите на изходните стойности в софтуера MATLAB.

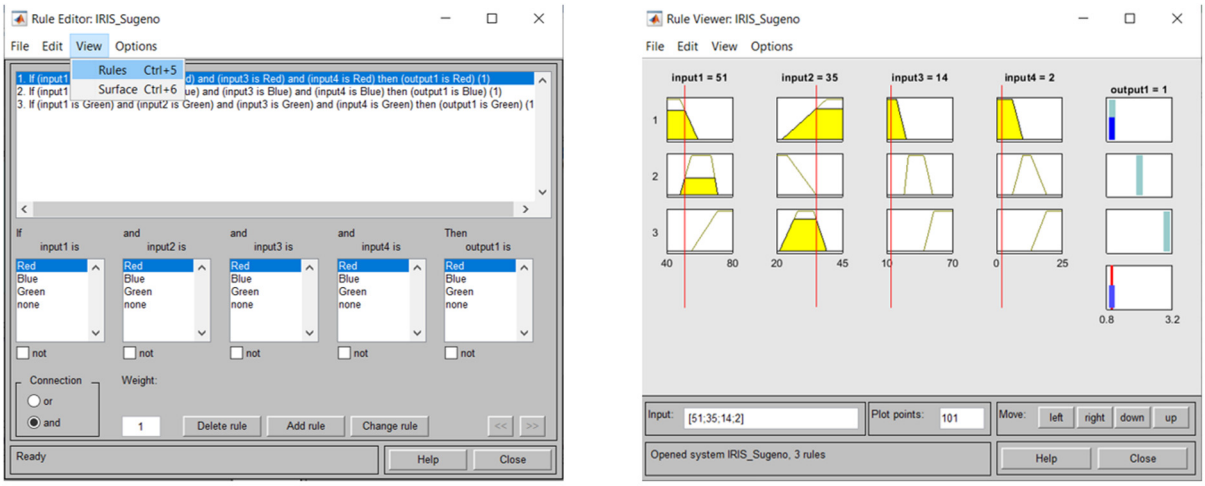
Сега трябва да променим стойностите на изходната променлива. За да редактирате изходната променлива, щракнете два пъти върху синия правоъгълник, наречен output1. Ще получим ново меню за изходната променлива, както е показано на Фигура 22. В това меню ще въведем стойностите от Таблица 2.

Последната стъпка е да създадем правилата на нашата система. Отваряме менюто с правила (виж Фигура 23а) и използваме три прости правила (виж Фигура 23б).



(а) (б)

Фигура 23: Отваряне на менюто с правила (а) и добавяне на правила (б) в софтуера MATLAB.



(а) (б)

Фигура 24: Отваряне на браузъра за правила (а) и добавяне на конкретни стойности към входните параметри (б) в софтуера MATLAB.

Системата ни е готова. Сега можем да оценим резултатите, които системата дава за известни входове. Можем да отворим програмата за преглед на правила (виж Фигура 24а) и да добавим конкретна стойност към всяка входна променлива (виж Фигура 24б). Можем да добавим тези стойности чрез преместване на червените линии в горната част на менюто или чрез промяна на стойностите на параметрите на подредената четворка в долната част на менюто.

Бележки:

Входните стойности, показани на Фигура 24б, принадлежат на първия ред от таблицата с данни за Iris. Както виждаме, системата класифицира обекта с тези входни атрибути в клас 1 (output1 = 1). Това е стойността, която очаквахме като резултат!

Показахме как можем да класифицираме един обект от набора данни Iris. Можем да използваме този подход за всеки ред от таблицата. Разбира се, можем също така да класифицираме всички редове на таблицата в една стъпка, като използваме последователност от команди от командния ред. Можем също така да изчислим процента на успеваемост на класификацията, като използваме дадената FIS. Използвайки параметрите, посочени в Приложение Б, достигаме успеваемост 94.6667%, т.е. 142 цветя от всички 150 цветя са класифицирани правилно.

Успеваемостта на класификацията може да бъде подобрена чрез използването на няколко различни подхода. Например, можем да използваме повече стойности на входните лингвистични променливи и след това да създадем повече правила. В представения пример използвахме по 3 стойности за всяка от входните променливи (**червено – синьо – зелено**). Бихме могли да използваме и 5 стойности на всяка входна променлива, които представляват стойностите **много_малки-малки-средни-високи-много високи** (very_small_value – small_value – middle_value – high_value – very_high_value). След това можем да създадем още правила чрез комбинация от стойностите на тези входни променливи. От друга страна, можем да използваме други методи, които са създадени за оптимизиране на параметрите на стойностите на входните, а също и на изходните променливи. Един от тях е така наречената **ANFIS = Adaptive Neuro-Fuzzy Inference System** (Адаптивна невро-размита система за изводи), която оптимизира параметрите на създадената FIS с помощта на невронна мрежа. Основните познания за невронните мрежи са представени в следващата част на този наръчник.

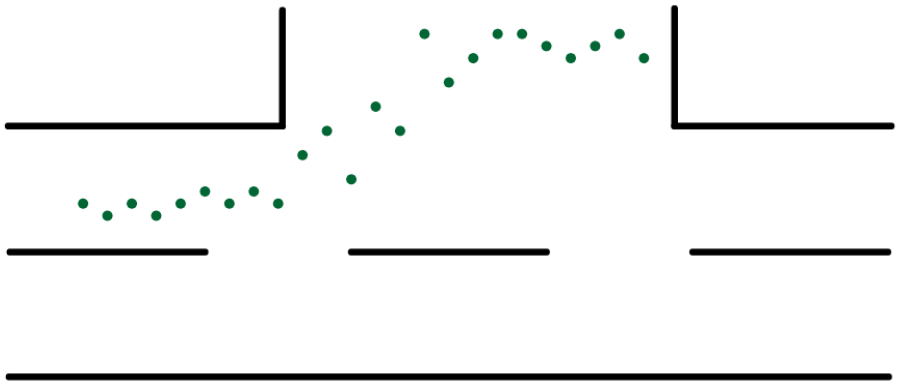
8

ИЗПОЛЗВАНЕ НА МЕТОДА НА СУГЕНО ЗА АПРОКСИМАЦИЯ НА ДАННИ

Тази част от ръководството е написана от Алжбета Михаликова от Департамента по компютърни науки, Факултет по природни науки, Университет „Матей Бел“ в Банска Бистрица, Словакия.

В случай че разполагаме с много данни и трябва да ги обработим, често е полезно да ги апроксимираме с някоя по-проста функция, която дава приблизителна стойност на реалния изход при точни входни стойности. Този процес се нарича **апроксимация**. Методът на Сугено е предназначен за апроксимация на такива данни, които в някои части на областта са линейни (в 2D те могат да бъдат апроксимирани с част от линията), а в останалата част на областта трябва да бъдат апроксимирани с някаква подходяща функция. В тази част на текста ще **апроксимираме** данни, които представят **пътя на автомобила**.

В тази част ще комбинираме обработката на данните в софтуер Excel и софтуер MATLAB.



Фигура 25: Позиция на автомобила в процеса на паркиране.

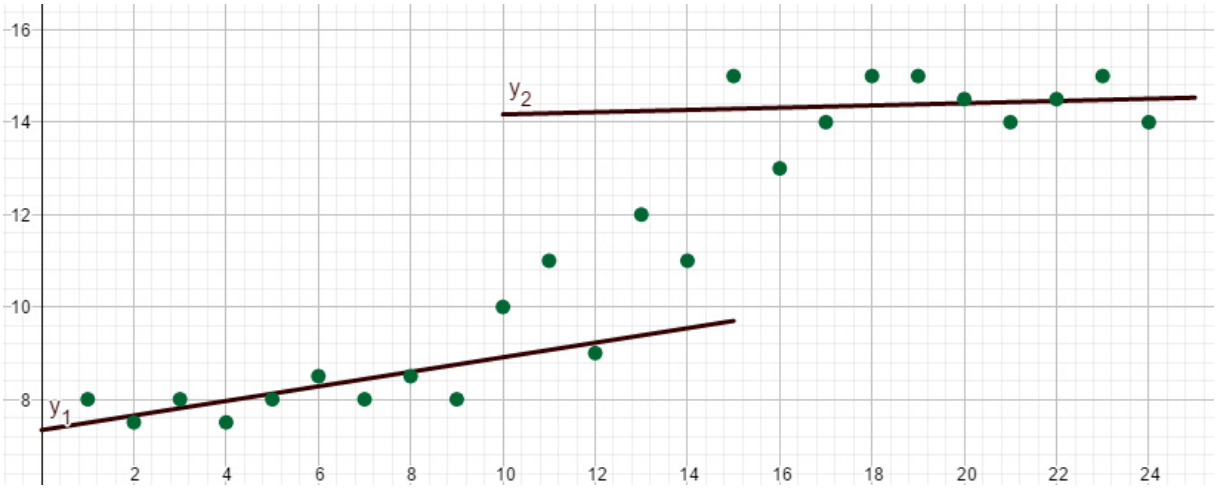
Пример: Да си представим, че разработваме автономен автомобил. Една от задачите, които трябва да решим, е да намерим функцията, която ще описва паркирането на автомобила на определено място. Бихме могли да помолим професионалния шофьор да паркира на определено място няколко пъти и бихме могли да заснемем пътя на автомобила чрез сензори (виж фигура 25).

Решение: Това движение на автомобила може да се опише чрез линии в двете части. Първа част – движение по прав път преди паркиране. Втора част – движение на мястото за паркиране. Движението между тези две части ще бъде апроксимирано по метода на Сугено.

В първата стъпка ще поставим данните си в декартовата координатна система (виж Таблица 3 и Фигура 26). Можем също така да начертаем линиите на праволинейно движение и да ги наречем y_1 и y_2. Както виждаме, има някои точки от данни, които ще допринесат за описанието на една линия (точки със стойност x от интервалите $\langle 1, 10\rangle$ и $\langle 15, 24\rangle$, а също и данни, които ще допринесат за описанието на две линии (точки със стойност x от интервалите $\langle 10, 15\rangle$. Тази информация е важна, когато проектираме функциите на принадлежност на използваното размито множество.

Таблица 3: Координати на приблизителните данни.

x	1	2	3	4	5	6	7	8	9	10	11	12
y	8	7,5	8	7,5	8	8,5	8	8,5	8	10	11	9
x	13	14	15	16	17	18	19	20	21	22	23	24
y	12	11	15	13	14	15	15	15	14	15	15	14

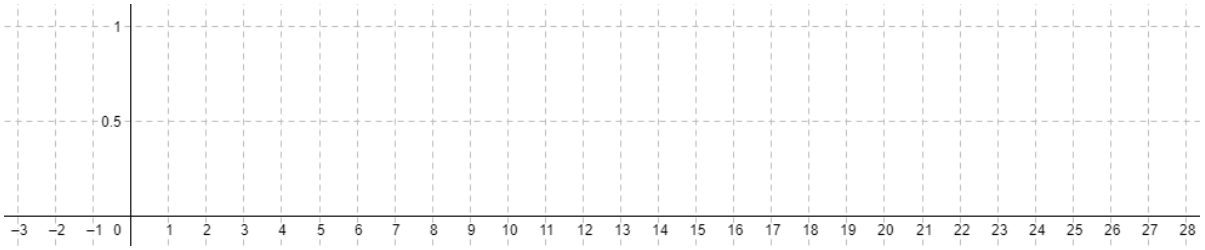


Фигура 26: Поставяне на данните в декартовата координатна система.

Нека да отговорим на следните въпроси:

1. Колко **входни променливи** имаме? Назовете тези променливи!
2. Колко **стойности на входните променливи** ще използваме? Назовете тези стойности на променливите!
3. Какво ще използваме, за **да опишем входните променливи**?
4. Какъв **тип размити функции на принадлежност** ще използваме?

5. Можете ли да **нарисувате** тези **функции на принадлежност**? Използвайте следната мрежа:



6. Можете ли да напишете **вселената, параметрите** на тези функции? Можете ли да **напишете предписанието** на тези функции за софтуера MATLAB?

7. Какъв ще бъде **изходът**?

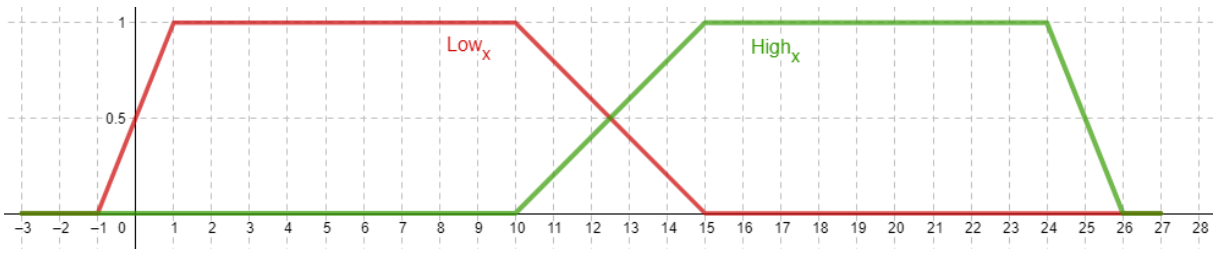
8. Какво ще използваме, за **да опишем изходните променливи**?

9. **Колко правила** ще използваме?

10. **Напишете пример** за едно правило!

Отговори:

Има само една входна променлива – тя може да бъде наречена **Позиция** (на автомобила) **по оста x**. Тази входна променлива има 2 стойности – **ниска стойност на координатата** x и **висока стойност на координатата x**. Ще ги опишем чрез размити множества. Ще използваме **трапецовидни функции на принадлежност**. Те могат да бъдат нарисувани, както е показано на Фигура 27.



Фигура 27: Трапецовидни функции на принадлежност за апроксимация на данни.

Популацията на тези размити функции е $\mathbb{X} = [1, 24]$. За ниската стойност на координатата x имаме параметрите $Low\ x = [-1, 1, 10, 15]$. За висока стойност на координатата x имаме параметри $High\ x = [10, 15, 24, 26]$.

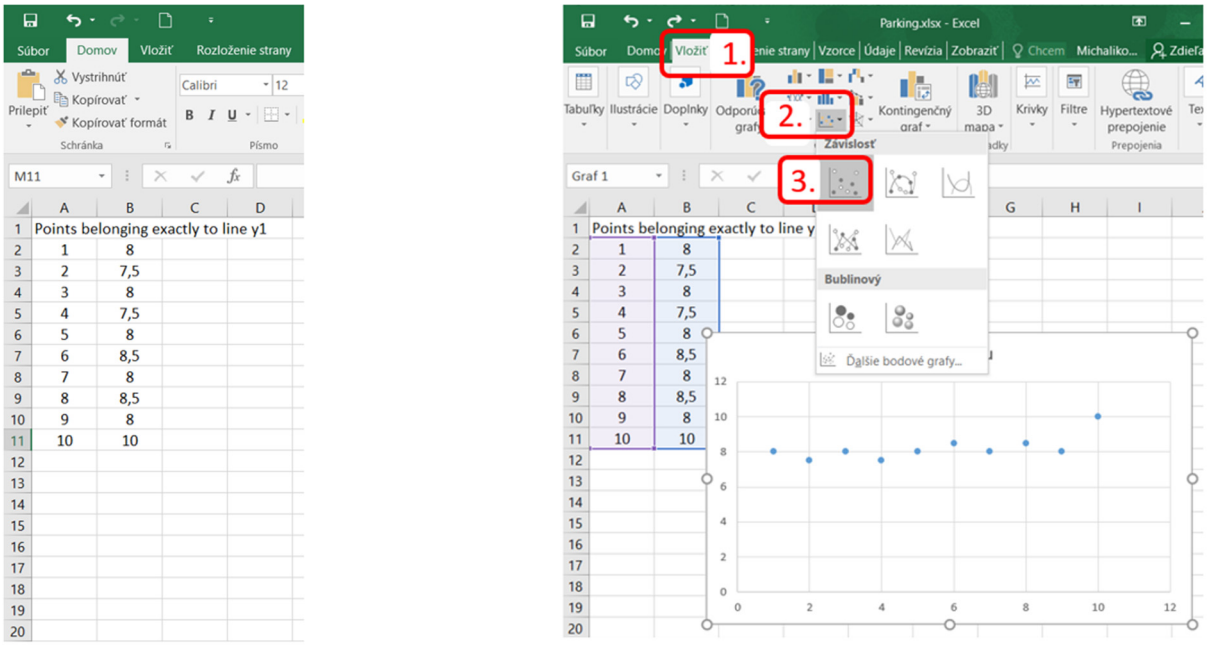
Изходната променлива представлява **Позицията на автомобила по оста y**. Като изходна променлива ще използваме линейна функция (линия). Ще използваме 2 линии y_1 и y_2. За да опишем параметрите на тези

линии, **ще използваме софтуера Excel** (виж по-долу). След това ще имаме **две правила IF-THEN**, които могат да бъдат записани по следния начин:

R1: IF Position of the car on x axis is Low_x, THEN Position of the car on *y* axis is y_1.

R2: IF Position of the car on x axis is High_x, THEN Position of the car on *y* axis is y_2.

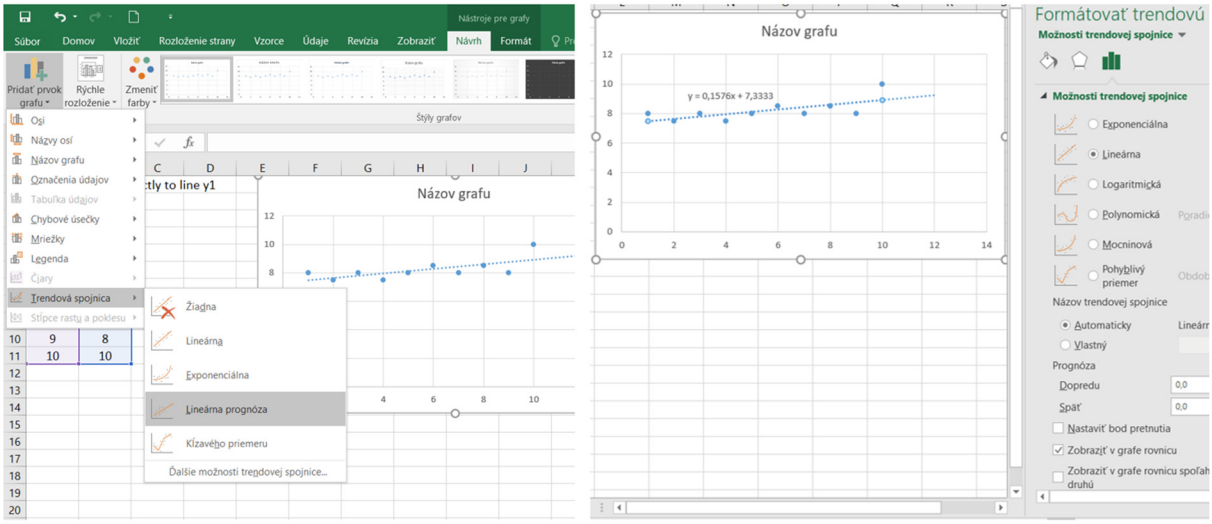
Трябва да изчислим параметрите на линейните функции, които представят изхода на правилата. Тези параметри ще бъдат изчислени от онези стойности на данните, които допринасят за описанието на точно една линия, т.е. за описанието на линията y_1 са допринесли първите 10 точки от данните. Нека ги копираме в Excel – всяка точка в един ред (виж Фигура 28а). След това маркирайте тези точки и използвайте следната последователност от стъпки **Insert** → **Charts** → **Points**.



(а) (б)

Фигура 28: Обработване на входните данни в софтуера Excel.

След това добавете елемента на графиката, както е показано на Фигура 29а, и изведете уравнението на линията (виж Фигура 29б).



(а) (б)

Фигура 29: Показване на уравнението на линията в софтуера Excel.

Получихме параметрите на линията y_1.

Със същата процедура ще получим параметрите на линията y_2. Така $y_1 = 0{,}1576x + 7{,}3333$ и $y_2 = 0{,}0242x + 13{,}927$. В MATLAB описанието е y_1 [0,1576 7,3333] и y_2 [0,0242 13,927].

Таблица 4: Обобщение на входните и изходните параметри за апроксимация на данни.

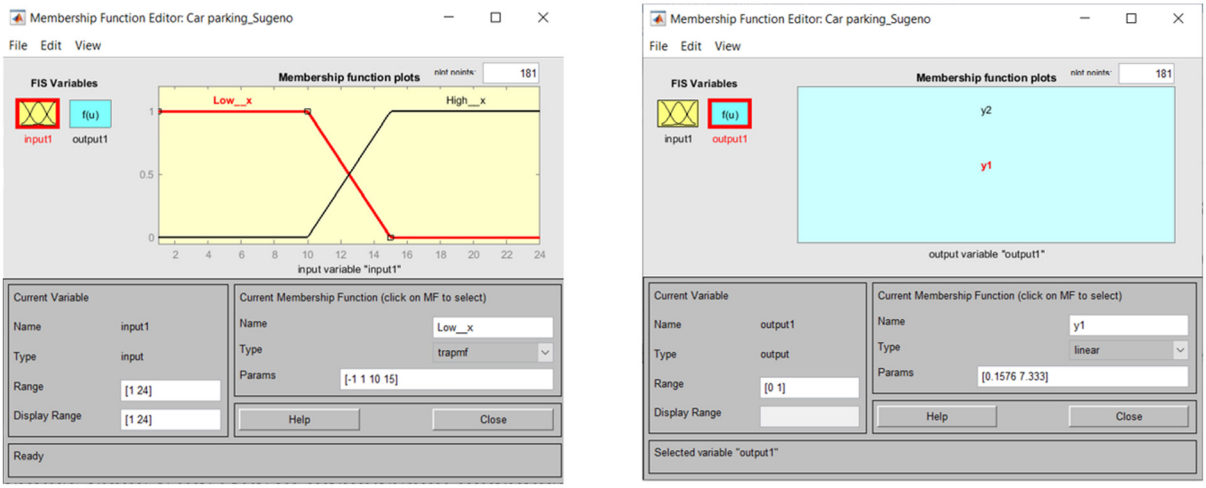
ВХОД: **ИЗХОД:**

Име	**Параметри**	**Име**	**Параметри**
популация	[1, 24]	**популация**	- – -
Ниска стойност на x	[-1, 1, 10, 15]	Ниска стойност на y	[0.1576; 7.3333]
Висока стойност на x	[10, 15,24, 26]	Висока стойност на y	[0.0242; 13.927]

Сега вече имаме всички параметри и можем да създадем FIS (ФИС) от тип Сугено в софтуера MATLAB. Първите стъпки са подобни на тези, които бяха споменати в подраздел 3 (класификация на данните). Отворете софтуера MATLAB и в командния прозорец напишете командата **fuzzy**. Тази команда отваря Fuzzy Logic Designer (Дизайнер на размита логика). Ние ще използваме метода на Сугено (Sugeno fuzzy inference system = Sugeno FIS), затова трябва да отворим този тип FIS (виж Фигура 19а). Бихме могли да преименуваме и запазим тази FIS (виж Фигура 19б) например като файл **Parking_Sugeno**.

Имаме само **една входна лингвистична променлива**. За тази променлива имаме само **две функции на принадлежност**. За да премахнете една от тях, просто щракнете върху една от функциите на графиката и използвайте Delete на клавиатурата. След това редактирайте параметрите на функциите на принадлежност (използвайте стойностите от таблица 4). Окончателната конфигурация за входните функции на принадлежност е показана на Фигура 30а.

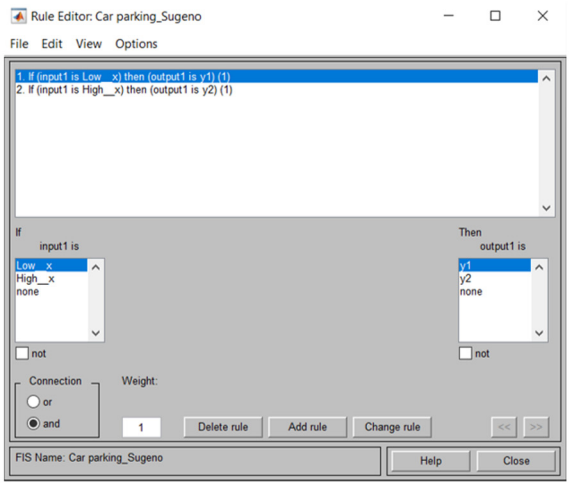
Сега трябва да променим стойностите на изходната променлива. За да редактираме изходната променлива, отново щракваме два пъти върху синия правоъгълник, наречен output1. Ще получим менюто за изходната променлива. По същия начин, както при създаването на предишния FIS, попълваме всички стойности (от Таблица 4). Не забравяйте, че в този ФИС **типът на изходната функция е линеен** (виж Фигура 30б).



(а) (б)

Фигура 30: Конфигуриране на входните и изходните променливи в софтуера MATLAB.

Последната стъпка е да създадем правилата на нашата система. Ще използваме две прости правила (виж Фигура 31а). Системата ни е готова. Сега можем да оценим резултатите, които системата дава за известни входове. Можем да отворим програмата за преглед на правила и да добавим към входната променлива нейната специфична стойност (виж Фигура 31б).

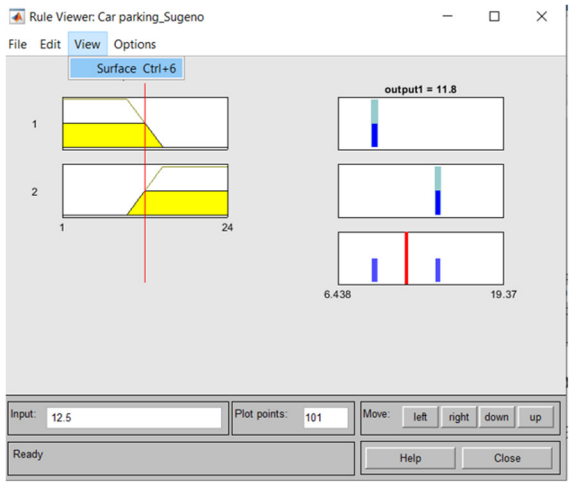


(а)

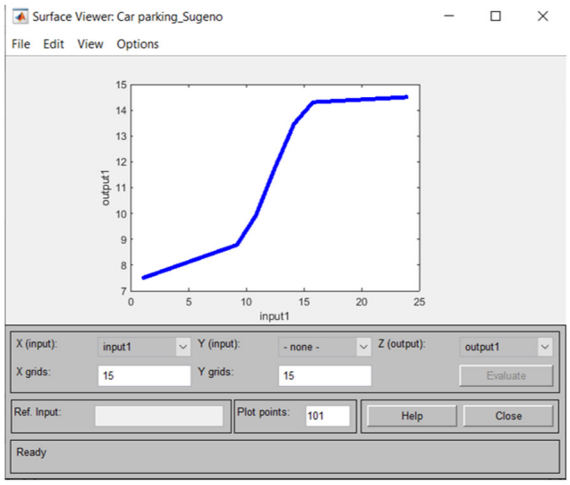
(б)

Фигура 31: Конфигуриране на правилата и оценка на резултатите в софтуера MATLAB.

Има и възможност да се покаже функцията, която създаваме с помощта на този ФИС (виж Фигура 32а).



(а)



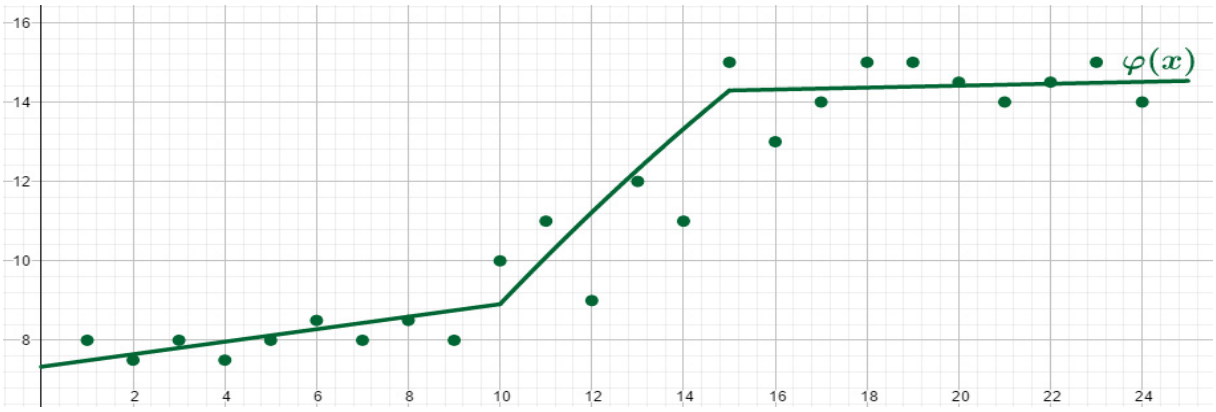
(б)

Фигура 32: Отваряне на Surface Viewer и крайната функция на създадения ФИС в софтуера MATLAB.

Бележки:

Входната стойност, показана на Фигура 31б, е равна на 8. Както виждаме, системата дава изходна стойност 8.59 за тази входна стойност. Реалната стойност (виж Таблица 3) е 8.5. Следователно получената стойност представлява добро приближение за тази точка.

Можем да сравним оригиналните (реални) данни с получената функция. Съществуват два основни подхода за сравняване (оценяване) на резултатите. Първият е графичен. Вторият е чрез изчисляване на грешката на създадената система. На фигура 33 е представено **графично сравнение** на реалните данни и получената функция.



Фигура 33: Графично сравнение на реалните данни и получената функция.

Показахме как можем да получим приблизителна изходна стойност към една конкретна входна стойност. Разбира се, можем да апроксимираме всички входове от таблица 3 в една стъпка с помощта на поредица от команди от командния ред. Можем също така да изчислим получената грешка. Съществуват повече видове грешки, които могат да бъдат изчислени. Най-широко използваната грешка е така наречената **Средна квадратна грешка** – Ср.кв.гр.(Mean Square Error – MSE). Тази грешка се изчислява по формулата

$$MSE = \frac{1}{n}\sum_{i=1}^{n}[f(x_i) - \varphi(x_i)]^2 ,$$

където *n* представлява броят на входните данни, стойностите на $f(x_i)$ представляват реалните изходи, а $\varphi(x_i)$ представлява изходите, изчислени от ФИС. Тази стойност на Ср.кв.гр. е подходяща за използване, ако искаме да сравним два или повече различни подхода. Тогава най-малката стойност представлява по-добрия подход. За нашата система достигнахме стойност на Ср.кв.гр., равна на 0.7263.

Качеството на апроксимацията може да бъде подобрено чрез използването на няколко различни подхода. Например, можем да използваме повече функции на принадлежност и след това да създадем повече правила. В представения пример използвахме 2 функции на принадлежност (**Low_x – High_x**). Можем да използваме 3 стойности на входната променлива, които представляват стойностите **Low_x – Medium_x – High_x**. След това можем да намерим предписанието на 3 реда и да създадем

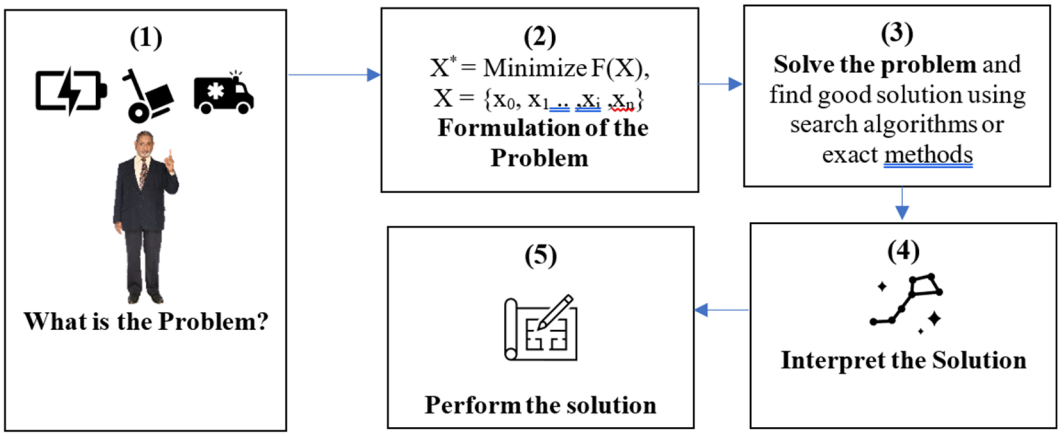
3 правила чрез комбинация от входните и изходните стойности. Когато имаме голям набор от входни данни, можем да използваме и други видове функции на принадлежност (те бяха споменати в подраздел 2 на този раздел). Тогава определянето на параметрите на функциите може да бъде дадено от статистическото разпределение на данните. От друга страна, отново можем да използваме друг метод, който е предназначен за оптимизиране на параметрите на стойностите на входните, а също и на изходните променливи.

9

ВЪВЕДЕНИЕ В ОПТИМИЗАЦИЯТА

Тази част от ръководството е написана от Фатих Килич от Университета по наука и технологии, Адана, Турция.

В много области на науката се решава оптимизационен проблем за търсене на оптимално решение в пространството за търсене (всички възможни решения), като се използват математически и евристични методи. Съществуват различни оптимизационни проблеми, като например инженерни, финансови, медицински и производствени проблеми. На Фигура 1 са показани основните етапи на решаване на оптимизационни проблеми. В първата стъпка вземащите решения искат да решат оптимизационен проблем, за да подобрят настоящите системи или да предложат нови системи. Например искаме да разположим болницата в най-добрата позиция, като вземем предвид търсенето и потенциалните пациенти. На второ място този проблем трябва да бъде математически формулиран като структура на решението, цел и ограничения. Структурата на решението се състои от променливи на решението. Променливите на решението са възможните позиции на кандидат-болниците за този проблем. Целевата функция измерва качеството на решението, което трябва да се оцени сред кандидат-решенията. Целевата функция може да бъде сумата от разстоянията между болниците и потенциалните пациенти за примерния проблем. Всички решения могат да бъдат изпълними или неизпълними решения поради предварително зададени ограничения. Тези ограничения се определят от специалиста. За този проблем най-малко една болница може да се намира в подрайон, който е търсен от заинтересованите страни. В три стъпки се прилагат добре познати методи за намиране на добри решения. Тези методи генерират оптимално решение или добри решения, близки до оптималното решение. Заинтересованите страни интерпретират решенията и правят незначителни корекции на решението, ако това е необходимо. Накрая решението се реализира.



Фигура 1 – Основни стъпки при решаването на оптимизационни задачи.

От математическа гледна точка всяка оптимизация може да се обясни по следния начин:

$$\max/\min_{x \in F \subseteq S} f(x),$$

където x показва набор от променливи за вземане на решение, *F* съдържа възможни решения, *S* представлява пространството на решенията, а *f(x)* показва целевата функция. max/min има за цел да намери максималната и минималната стойност на *f(x)*.

Можем да формулираме ограничения и обхват на данните и променливите. Даден е следният пример:

$$\sum_{j}^{n} x_j < \mathrm{b}$$

$$x_j \in \{0,1\} for\ j = 1 \ldots . n$$

където x_j може да бъде 0 или 1 за всяко *j*, а сумата на всички x елементи, по-малка от *b*.

Задачите, които се опитват да намерят непрекъснати променливи, се класифицират като непрекъснати оптимизационни задачи. В таблица 1 са показани добре познати проблеми за непрекъсната оптимизация. Решението (X) се състои от D-мерни реални стойности. Всяко измерение е между предварително зададени минимални и максимални числа. Измерението е броят на променливите на решението. Те се използват, за да се демонстрира тяхната ефективност при въвеждането на алгоритми за оптимизация.

Таблица 1 – Едномодални функции.

Размер-ност	Диапазон	Уравнение
5	[-100, 100]	$F_1(x) = \sum_{i=1}^{n} x_i^2$
	[-10, 10]	$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$
	[-100, 100]	$F_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j \right)^2$
	[-100, 100]	$F_4(x) = \max_i \{ \lvert x_i \rvert,\ 1 \le i \le n \}$
	[-30, 30]	$F_5(x) = \sum_{i=1}^{n-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + (x_i - 1)^2 \right]$
	[-100, 100]	$F_6(x) = \sum_{i=1}^{n} (\lvert x_i + 0.5 \rvert)^2$
	[-1.28, 1.28]	$F_7(x) = \sum_{i=1}^{n} i x_i^4 + \text{random}[0, 1]$

9.1. Алгоритми за локално търсене

Алгоритмите за локално търсене (LSA) се използват за решаване на оптимизационни задачи в компютърните науки и свързаните с тях изчислителни науки. Тези алгоритми се определят като обобщени евристични алгоритми за търсене и могат да се прилагат за различни оптимизационни задачи след формулиране на проблемите.

Обикновено LSA се занимават с едно решение, за да създадат по-добро решение във всеки един момент. Добре познати алгоритми за локално търсене са симулираното отгряване, търсенето по Табу, изкачването по хълм и търсенето по променливи съседи.

Алгоритъм 1 показва основните стъпки на алгоритъма за изкачване по хълм (HC).

Algorithm 1: Hill Climbing

```
1  currentSolution = Generate initial solution
2  Evaluate currentSolution
3  iteration =0
4  while (!Stop contitions)
5        NeighbourSolution = Movement(currentSolution)
6        If NeighbourSolution is better than currentSolution
7              currentSolution = NeighbourSolution
8        end if
9        iteration = iteration+1
```

В първата стъпка началното решение се генерира на случаен принцип и се присвоява на текущото решение. Например, *X= [60.15, -50.07, 10.08, -80.01, 17.59]* за функцията *F1* в Таблица 1. Всеки елемент от този век-

тор е между -100 и +100 и е избран на случаен принцип. Второ, генерира се съседното решение, като се използва текущото решение и функцията за малка модификация във всяка итерация. Избират се случайни индексни числа и избраният елемент се променя за вектора *X*. Ако съседното решение е по-добро от текущото решение, тогава текущото решение се актуализира със съседното решение. Итерациите се извършват до удовлетворяване на текущото решение или до достигане на максималния брой итерации.

9.2. Еволюционно изчисление

Еволюционното изчисление (ЕС) е популярен алгоритъм за оптимизация и популация, който имитира биологичната еволюция, като възпроизвеждане, рекомбинация, мутация, селекция и оцеляване на индивидите. Въведени са различни варианти на ЕК, като се използват процесите на биологичната еволюция. Генетичният алгоритъм е изобретен от Джон Холанд (1962), докато Еволюционните стратегии са изобретени от Инго Рехенберг (1965).

Стъпките на една типична ЕК са дадени в Алгоритъм 2.

Algorithm 2: Evolutionary Computation	
1	`Population = Generate randomize initial solutions`
2	`iteration =0`
3	`while (!Stop conditions)`
4	`  Fitness values are computed for each individual in Population`
5	`  Individuals are selected as parents in Population based on fitness values`
6	`  Crossover and mutation are performed to generate offspring`
7	`  Update Population according to new offspring and their fitness values`
8	`  iteration = iteration+1`

В първите етапи популацията се избира на случаен принцип с предварително определен размер. Популацията се състои от решения. Всеки индивид представлява едно решение. Втората стъпка представлява набор от итеративни процеси. Във втората стъпка се изчислява стойността на фитнес за всеки индивид, като се използва целева функция. Родителите се избират въз основа на тяхната фитнес способност или различни техники. Кръстосването и мутацията са процеси на възпроизвеждане за генериране на нови решения. За следващото поколение процесът на подбор се извършва, като се използват стойностите на фитнес на индивидите. Тези стъпки се повтарят, докато се изпълнят условията за спиране.

Оператор за кръстосване (Crossover)

Операторът за кръстосване е за обмен на информация между хромозомите на двама избрани родители, за да се генерират две нови потомства. Този оператор е важен оператор за изследване в ЕС. Съществуват различни общи техники за кръстосване, като например едноточково, многоточково, равномерно кръстосване, както и специфични за проблема техники за кръстосване (за комбинаторни оптимизационни проблеми), като рекомбинация на ръбове, кръстосване на няколко родители с частично картографиране и кръстосване на базата на ред. Този оператор се извършва в зависимост от вероятността за кръстосване.

В Таблица 2 е представен пример за оператор за кръстосване в една точка. Родители 1 и 2 са избрани индивиди, като двете решения са показани съответно с курсив и подчертаване на първия и втория ред. Точката на прекъсване се избира на случаен принцип и родителите се разделят на две части за всеки индивид. Генерират се деца 1 и 2, които да разменят вторите части на родителите и да вземат същите първи части на родителите.

Таблица 2 – Пример за оператор за кръстосване с една точка.

	X_1	X_2	X_3	X_4	X_5
Родител 1	*60.15*	*-50.07*	*10.08*	*-80.01*	*17.59*
Родител 2	40.22	30.08	20.09	-20.05	60.85
Дете 1	*60.15*	*-50.07*	20.09	-20.05	60.85
Дете 2	40.22	30.08	*10.08*	*-80.01*	*17.59*

Оператор за мутация

Използва се оператор за мутация, за да се осигури разнообразие в популацията. Операторът за мутация модифицира родител, за да се получи потомство. Избира се произволна позиция от решението и съответният ген или бит се променя, за да се изпълни операторът за мутация. Съществуват различни оператори за мутация. Един от операторите за мутация е едромащабна мутация, която актуализира едновременно няколко позиции на индивида.

В Таблица 3 е представена извадка от мутации. X_3 се избира произволно и се използва методът на флип-флопа, като новата стойност на X_3 трябва да бъде 0.

Таблица 3 – Пример за оператор за мутация.

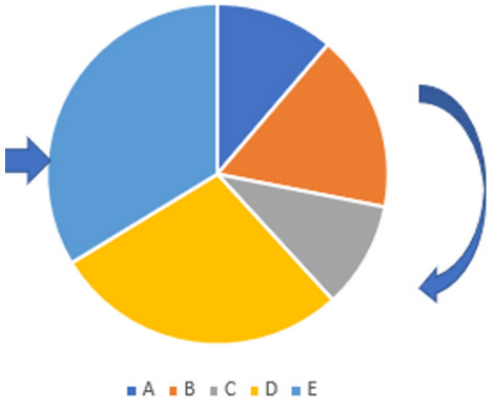
	X_1	X_2	X_3	X_4	X_5
Индивидуален	1	0	***1***	0	1
Нов	1	0	***0***	0	1

Стратегии за подбор

Стратегиите за селекция се използват за увеличаване на вероятността за оцеляване на индивиди и потомство с по-висока фитнес способност в следващото поколение и за подбор на родители. Селекцията на рулетка и турнирната селекция са популярни стратегии за селекция.

Избор на колело за рулетка: Кръговото колело се състои от n пити, където n е броят на решенията в популацията. Всяко решение получава част от пая въз основа на неговата стойност на годност. Избира се точка от периферията на колелото и то се завърта.

Solutions	Fitness Values
A	10
B	15
C	9
D	25
E	30



Фигура 2 – Извадка от популацията.

Избор на турнира: При този подход „турнирна, k“ селекция, k индивида се избират на случаен принцип от популацията и един от тях с най-добра фитнес стойност, използвайки турнир.

9.3. Решаване на проблема с „раницата“

В задачата за раницата един пакет с набор от предмети с тегла и стойности иска да бъде поставен в раницата с максимална обща стойност. Таблица 4 показва набор от тестови данни за проблема с раницата.

Таблица 4 – Набор от тестови данни за задачата „раницата“.

	Продукт 1	Продукт 2	Продукт 3	Продукт 4	Продукт 5	Продукт 6	Продукт 7
Тегло	30	20	10	45	15	33	25
Стойност	10	5	30	16	50	13	13
Пример на решение	1	0	1	1	0	1	1

Използваме следната терминология, параметри и променливи на решението.

Бележки:

j: item index, $j \in \{1 \ldots J\}, J \; is \; number \; of \; items$

Параметри:

v_j: value of item j

w_j: weight of item j

W: maximum capacity of the knapsack

Променливи на решението:

$$x_j = \begin{cases} 1, & if \; item \; j \; is \; selected \; into \; the \; knapsack \\ 0, & otherwise \end{cases}$$

$$\text{Maximize } F = \sum_{j=1}^{J} v_j x_j$$

$$\text{subject to } = \sum_{j=1}^{J} w_j x_j < W$$

Код на функцията за пригодност:

```
function Fit = MyFitness(x)
    global wSet vSet maxCapacity;
    sumV = sum(x(1,:).* vSet);
    sumW = sum(x(1,:).* wSet);
    if sumW <= maxCapacity
        Fit= sumV;
    else
        Fit = 0;
    end
```

Код на генетичния алгоритъм:

```
clc;
clear;
close all;

global nItem wSet vSet maxCapacity;
wSet = [30, 20, 10, 35, 15, 33, 25, 25, 25, 15, 25,54]; % weights of each
item
vSet = [10, 5, 30, 16, 50, 13, 13, 23, 14, 52, 10,50]; % value of each item
maxCapacity = 120;
nItem = size(wSet,2);
FitnessFunction = @(x) MyFitness(x);
WeighFunction = @(x) MyFitnessW(x);
popSize = 20;
```

```
maxIter = 50;

muProbility = 0.2;
individual.Solution = [];
individual.FitnessValue = [];
individual.Weight = [];
population = repmat(individual, popSize, 1);
round(rand(1,nItem));
for i = 1:popSize
    population(i).Solution = round(rand(1,nItem));
    population(i).FitnessValue = FitnessFunction(population(i).Solution);
    population(i).Weight = WeighFunction(population(i).Solution);
end

% Sort Population
FitnessValues = [population.FitnessValue];
[FitnessValues, SortOrder] = sort(FitnessValues,'descend');
population = population(SortOrder);

BestSol = population(1);
BestFitness = zeros(maxIter, 1);
TournamentSize=3;

for t = 1:maxIter
    % Crossover operator
    populationCrossover = repmat(individual, popSize/2, 2);
    for j = 1:popSize/2
        i1 = TournamentSelection(population, TournamentSize);
        i2 = TournamentSelection(population, TournamentSize);

        p1 = population(i1);
        p2 = population(i2);

        % Perform Crossover
        [populationCrossover(j, 1).Solution, populationCrossover(j,
2).Solution] = Crossover(p1.Solution, p2.Solution);

        % Evaluate Offsprings
        populationCrossover(j, 1).FitnessValue =
FitnessFunction(populationCrossover(j, 1).Solution);
        populationCrossover(j, 2).FitnessValue =
FitnessFunction(populationCrossover(j, 2).Solution);

        populationCrossover(j, 1).Weight =
WeighFunction(populationCrossover(j, 1).Solution);
```

```
        populationCrossover(j, 2).Weight =
WeighFunction(populationCrossover(j, 2).Solution);
    end

    populationCrossover = populationCrossover(:);

    % Mutation operator
    mutPop =0;
    populationMutation = repmat(individual, 0,1);
    for j = 1:popSize
        p = population(i);

        if (rand < muProbility)
            mutPop=mutPop+1;
            k= randi(nItem);
            p.Solution(k) = 1- p.Solution(k);
            p.FitnessValue = FitnessFunction(p.Solution);
            p.Weight = WeighFunction(p.Solution);

            populationMutation(mutPop) = p;
        end
    end

    populationMutation = populationMutation(:);

    population = [population
         populationCrossover
         populationMutation];

    FitnessValues = [population.FitnessValue];
    [FitnessValues, SortOrder] = sort(FitnessValues,'descend');
    population = population(SortOrder);

    population = population(1:popSize);
    FitnessValues = FitnessValues(1:popSize);

    BestSol = population(1);

    BestFitness(t) = BestSol.FitnessValue;

    disp(['Generation : ' num2str(t) ': Best Fitness value = '
num2str(BestFitness(t))]);
end
plot(1:maxIter,BestFitness);
```

10

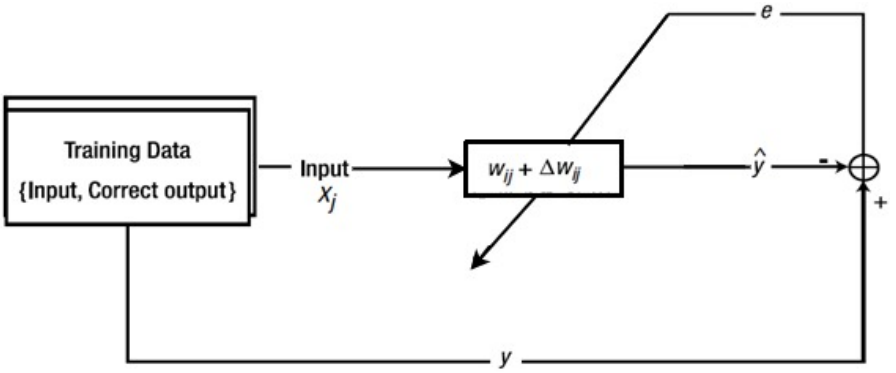
ЕДНОСЛОЙНА НЕВРОННА МРЕЖА

Тази част от ръководството е написана от Ондер Тутсой от Университета по наука и технологии, Адана, Турция.

Невронните мрежи (НМ) съхраняват информация под формата на тегла, научени от гледна точка на контролираното (разпознаване на образи) или неконтролираното (апроксимация на функции) обучение. НМ по същество са непараметрични подходи за моделиране, използвани за приблизително представяне на реални системи. Поради това техният аналитичен (задълбочен и строг математически) анализ е предизвикателство. За да се обучат НМ, теглата трябва да се актуализират въз основа на информацията, предоставена чрез входовете. Систематичният подход, използван за актуализиране на теглата, се нарича правило за обучение, което използва предоставената входна информация. По същество то съпоставя входната информация с изходната. Тъй като обучението е единственият начин за НМ да съхраняват и запомнят информацията систематично, правилото за обучение е жизненоважен компонент на процеса на обучение, разгледан по-долу.

10.1. Правило Делта

Правилото делта е представително правило за обучение на еднослойните НМ. Процесът на обучение на еднослойно НМ може да бъде показан на фигурата по-долу.



Фигура 1: Блок схема на процеса на обучение на НМ с един слой.

Важно е да се отбележи, че еднослойният НМ може да бъде с един вход и един изход (SISO), с един вход и няколко изхода (SIMO), с един вход и няколко изхода (MISO) или с няколко входа и няколко изхода (MIMO). Броят на входовете и изходите варира в зависимост от характера на проблема за обучение. Обърнете внимание, че свързаната динамика може да се изучава само от НМ, които имат повече от един вход или изход. Въпреки това, ако проблемът за обучение не е свързан, но проблемът за обучение на НМ е конструиран като свързан, тогава ефективността на НМ ще намалее. Ето защо първоначално трябва да се анализира характерът на входните данни и въз основа на получените познания за входните данни да се конструират НМ.

Псевдокод за правило за делта обучение за *m* входа и *n* изхода в невронна мрежа:

1. Вход: Данни за обучението $x \in \mathbb{R}^{m \times l}$, където l е дължината на всеки m брой на входящите данни, маркиран изход $y \in \mathbb{R}^{l \times n}$, където n е номерът на изхода, произволно инициализирана матрица/вектор на неизвестните параметри $w \in \mathbb{R}^{m \times n}$, очаквана продукция $\hat{y}_o \in \mathbb{R}^{n \times l}$, наситени и оценени $\hat{y} \in \mathbb{R}^{n \times l}$, грешката при обучението $e \in \mathbb{R}^{l \times n}$, скоростта на изучаване на параметъра $0 < \eta \le 2 / x(:,1)^T x(:,1)$, броят на множествените симулации simMultiple, матрица за съхранение w_s , грешка при запазване e_s , праг за спиране на грешката e_t , съхраняване на изчисления резултат $\hat{y}_s$.

2. Изход: Крайната стойност на обучените параметри w, грешка за запазване при обучението e_s , съхраняване на резултата $\hat{y}_s$

3. for *i* to simMultiple

4. for *j* to *l*

5. 1. Изчисляване на приблизителния изходен поток $\hat{y}_o$

6. $\hat{y}_o(:, j) = w^T x(:, j)$

7. 2. Прилагане на праг на изхода σ (ако е необходимо)

8. $\hat{y}(:, j) = \sigma(\hat{y}_o(:, j))$

9. 3. Определяне на грешката

10. $e(:, j) = y - \hat{y}(:, j)$

11. 4. Актуализиране и съхраняване на неизвестния параметър

12. $w \to w + \eta e(:, j)\, x(:, j)^T$

13. $w_s = \left[w_s ; reshape(w_s, 1, []) \right]$

14. end *j*

15. Съхраняване на грешката и изход на насищане

16. $e_s = \left[e_s ; reshape(e, 1, []) \right]$

17. $\hat{y}_s = \left[\hat{y}_s ; reshape(\hat{y}, 1, []) \right]$

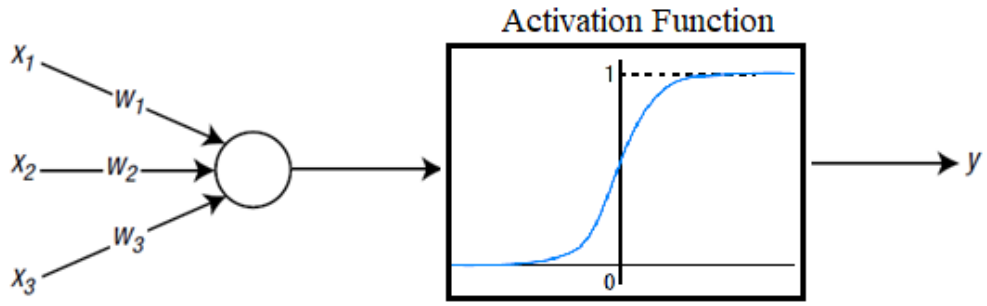
18. If $e(:, j) < e_t$ then

19. break

20. end if

21. end *i*

Правилото делта актуализира неизвестните параметри итеративно, вместо да ги решава наведнъж. Това е вид цифров итеративен метод, използващ спускане по градиент. Спускането по градиент започва от началната стойност и се движи към решението. Името му произлиза от поведението му, при което той търси решението, сякаш топката се търкаля по хълма по най-стръмния път. В тази аналогия позицията на топката е случайният резултат от модела, а дъното е решението. Заслужава да се отбележи, че итеративният метод за спускане по градиент не може да пусне топката до дъното само с едно хвърляне. Целият процес се повтаря, тъй като преквалификацията на модела със същите данни може да подобри модела.



Фигура 2 – NN, която се състои от три входни и един изходен възел.

Пример: Правилото делта

Разгледайте НМ, който се състои от три входни възела и един изходен възел, както е показано на предходната фигура.

Както се вижда от Фигура 2, за активационната функция на изходния възел се използва сигмоидната функция. Имаме четири точки с данни за обучение, както е показано в следната таблица.

Таблица 1 – Точки с данни за обучение на OR изход с етикети.

{0,0,1,**0**}
{0,1,1,**1**}
{1,0,1,**1**}
{1,1,1,**1**}

Тъй като те се използват за обучение с наблюдение, всяка точка от данни се състои от двойка вход – коректен изход. Последният удебелен номер на всеки набор от данни е правилният изход. Това е задача на OR Gate, като последната стойност на входа е отклонението от 1.

Тъй като е еднослоен и съдържа прости данни за обучение, кодът не е сложен. След като следвате кода, ще разберете ясно поведението на обучението на NN.

Съответният код се изпълнява по следния начин:

*Първоначално параметрите на обучението се определят, както в следната функция **trainPar**:*

```
function trainPar = trainParameters()
% Training input data where the last value of 1 represents the bias
trainPar.x = [0 0 1; 0 1 1; 1 0 1; 1 1 1]';
% Labelled output data
trainPar.y = [0 1 1 1]';
% Randomly intialized unknown parameters
trainPar.w = rand(size(trainPar.x,1),size(trainPar.y,2));
% Intitialize the estimated output
trainPar.yo_hat = zeros(size(trainPar.x,2),size(trainPar.y,2));
% Intitialize the estimated output
trainPar.y_hat = zeros(size(trainPar.x,2),size(trainPar.y,2));
% Intitialize the error
trainPar.e = zeros(size(trainPar.x,2),size(trainPar.y,2));
% Intitialize the learning rate
trainPar.mu = zeros(size(trainPar.y));
% Learning rate upper scaling
trainPar.mur = 2;
```

```
% Stopping error threshold
trainPar.et = 0.001;
% The number of the multiple trainings
trainPar.simMultiple = 1000;
% The output saturation function upper limit (sigmoid)
trainPar.satUppper = 1;
end
```

След определянето на съответните параметри на обучението, за процеса на обучение се използва следната функция.

```
% This m-file trains a single layer NN for the OR problem
% Upload the training parameters
trainPar = trainParameters();
% Upload the allocated error
e = trainPar.e;
% Upload the allocated estimated output
yo_hat = trainPar.yo_hat;
% Upload the allocated ouput with threshold
y_hat = trainPar.y_hat;
% Upload the allocated unknown parameter
w = trainPar.w;
% Upload the allocated learning rate
mu = trainPar.mu;
% Introduce the store matrix for the unknown parameter
ws = [];
% Introduce the stora matrix for the error
es = [];
% Introduce the store matrix for the estimated output
ys_hat = [];
for i=1:trainPar.simMultiple
    for j=1:size(trainPar.x,2)
        % Calculate the estimated current output
        yo_hat(j,:) = w'*trainPar.x(:,j);
        % Apply a threshold for the estimated output
        y_hat(j,:) = satOutput(yo_hat(j,:),trainPar);
        % Determine the instant error
        e(j,:) = trainPar.y(j,:) – y_hat(j,:);
        % Update the learning rate
        mu(i,j) =trainPar.mur/(trainPar.x(:,j)'*trainPar.x(:,j));
        % Update the unknown parameter vector/matrix
        w = w + mu(i,j)*e(j,:)*trainPar.x(:,j);
        % Store the unknown parameter vector/matrix
        ws = [ws;reshape(w,1,[])];
    end
```

```
        % Store the error history
        es = [es;reshape(e,1,[])];
        % Store the estimated output
        ys_hat = [ys_hat;reshape(y_hat,1,[])];
end
```

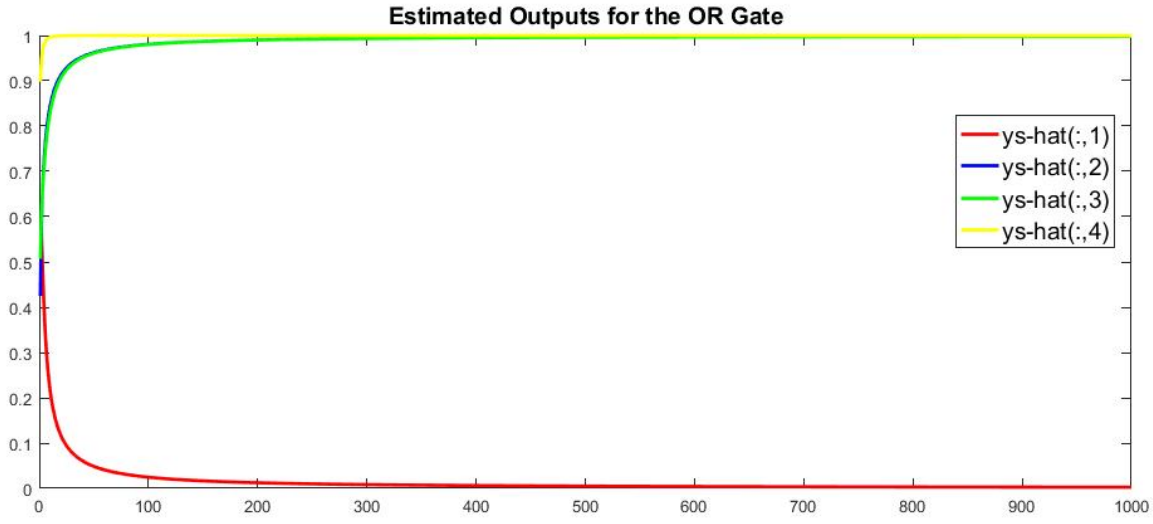
където функцията **satOutput** е създадена за сигмоидната активираща функция, както е посочено по-долу:

```
function y_sat = satOutput(y_unsat, trainPar)
      y_sat = trainPar.satUppper / (1 + exp(-y_unsat));
end
```

*След това оценените резултати **ys_hat** за всеки набор от входове се нанасят на графиката, като се използва следният блок-код:*

```
figure(1),
plot(1:length(ys_hat),ys_hat(:,1),'r','LineWidth',2),
hold on,
plot(1:length(ys_hat),ys_hat(:,2),'b','LineWidth',2),
plot(1:length(ys_hat),ys_hat(:,3),'g','LineWidth',2),
plot(1:length(ys_hat),ys_hat(:,4),'y','LineWidth',2),
hold off
title('Estimated Outputs for the OR Gate')
```

При изпълнението на този код се получава следната фигура:



Фигура 3 – Очаквани резултати за изходите на OR изход.

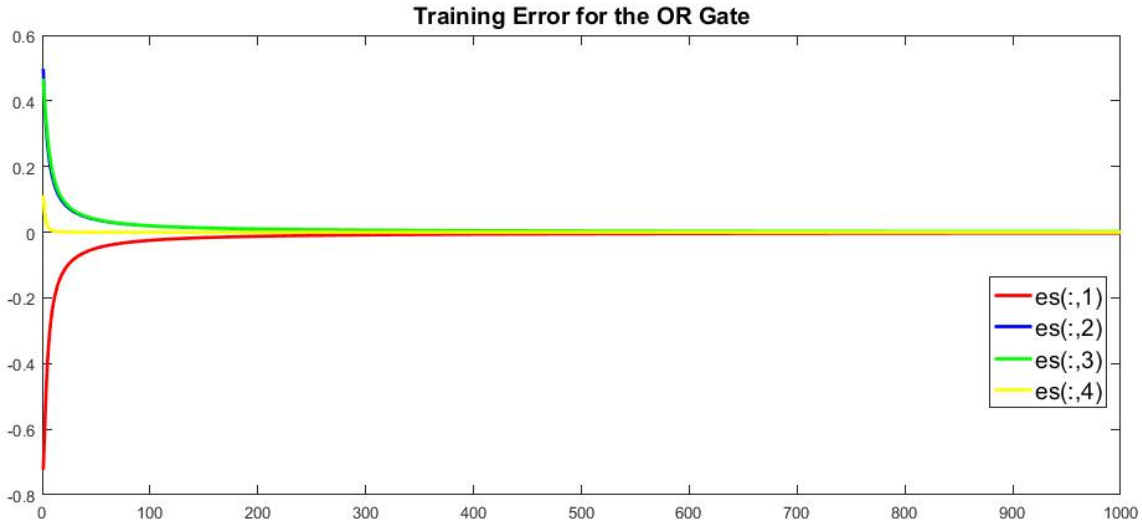
Изпълнението на този код дава следните стойности. Тези изходни стойности са много близки до правилните изходни стойности на целевата стойност y. Следователно можем да заключим, че НМ е бил правилно обучен да учи OR изход.

$$\begin{bmatrix} 0.0025 \\ 0.9980 \\ 0.9980 \\ 1.0000 \end{bmatrix} \Leftrightarrow \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Построяване на грешката при обучение,

```
figure(),
plot(1:length(es),es(:,1),'r','LineWidth',2),
hold on,
plot(1:length(es),es(:,2),'b','LineWidth',2),
plot(1:length(es),es(:,3),'g','LineWidth',2),
plot(1:length(es),es(:,4),'y','LineWidth',2),
hold off
title('Training Error for the OR Gate')
```

се използва блок-код и определеният резултат се представя като:

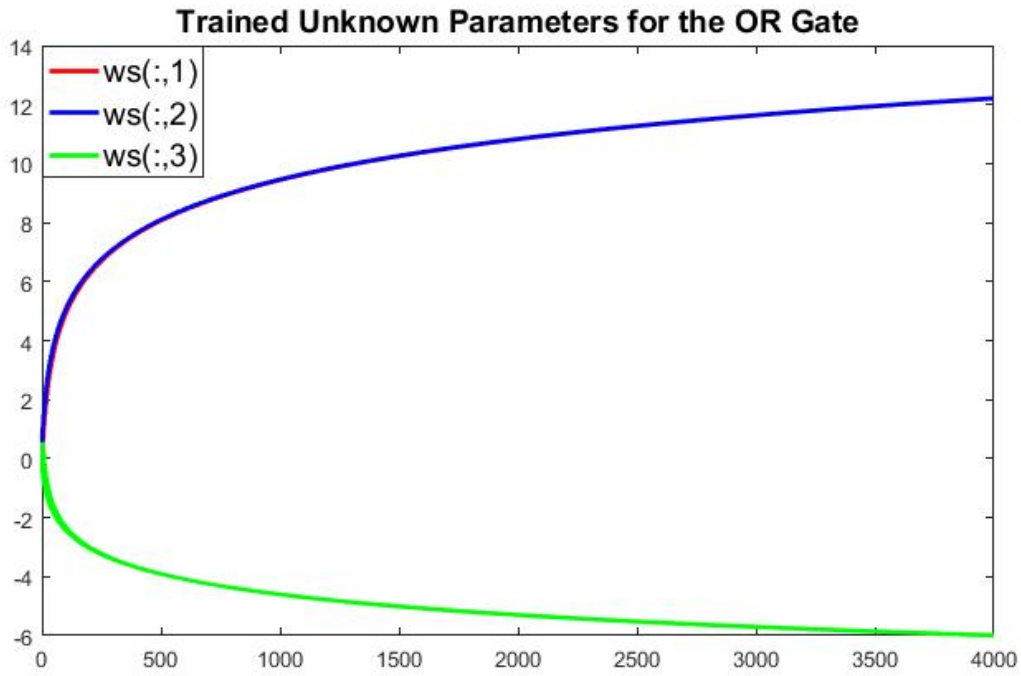


Фигура 4 – Резултати от грешките при обучението за OR изход.

Както се вижда от Фигура 4, грешката се доближава до нула съответно за съответните точки от данни на OR изход.

Накрая, обучените неизвестни параметри се нанасят и показват чрез следния блок-код:

```
figure(),
plot(1:length(ws),ws(:,1),'r','LineWidth',2),
hold on,
plot(1:length(ws),ws(:,2),'b','LineWidth',2),
plot(1:length(ws),ws(:,3),'g','LineWidth',2),
plot(1:length(ws),ws(:,4),'y','LineWidth',2),
hold off
title('Trained Unknown Parameters for the OR Gate')
```



Фигура 5: Обучени неизвестни параметри за OR изход.

Както ясно се вижда на Фигура 5, са нанесени само 3 „обучени“ неизвестни параметъра. Това се дължи на умножаването на матриците в следващия блок код една с друга.

```
trainPar.w = rand(size(trainPar.x,1),size(trainPar.y,2));
```

*където **size(trainPar.x)** е 3x4, а **size(trainPar.y)** е 4x1. Така ще бъде определен вектор 3x1. Всъщност е съвсем просто да се начертаят всички обучени неизвестни параметри тук. След всички тези обяснения, моля, направете необходимата актуализация в кода.*

10.2. Ограничения на еднослойната невронна мрежа

В този раздел е представена критичната причина, поради която еднослойнната невронна мрежа (НМ) трябваше да се превърне в многослойна НМ. Ще се опитаме да покажем това чрез конкретен случай. Да разгледаме същата НМ, която беше разгледана в предишния раздел. Тя се състои от три входни възела и един изходен възел, а функцията на активиране на изходния възел е сигмоидна функция. Да предположим, че имаме четири точки с данни за обучение, както е показано по-долу.

Таблица 2 – XOR изход на данни за обучение с етикети.

{0,0,1,**0**}
{0,1,1,**1**}
{1,0,1,**1**}
{1,1,1,**0**}

Както е посочено в Таблица 2, това е задача за XOR изход, в която последната стойност на входа е отклонението от 1. Тя се различава от раздела „Правило делта“ по това, че вторият и четвъртият правилни изходи се превключват, докато входовете остават същите. Е, разликата е едва забележима.

Тъй като разглеждаме една и съща невронна мрежа, можем да я обучим, като използваме функцията **trainPar** от „Пример: Правилото делта“, с изключение на това, че има различни стойности за y, както бе споменато по-горе. Преди да изпълните кода, блокът за код на маркираните изходни данни във функцията **trainPar** се актуализира, както следва.

```
% Labelled output data
trainPar.y = [0 1 1 0]';
```

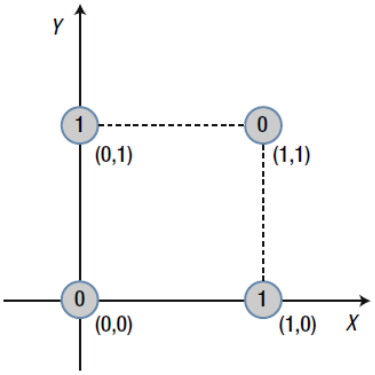
При изпълнение на този код ще се появят следните стойности, които се състоят от изхода на обучения NN, съответстващ на данните за обучение. Можем да ги сравним с правилните изходи, дадени от „y“, както следва:

$$\begin{bmatrix} 0.5297 \\ 0.5000 \\ 0.4703 \\ 0.4409 \end{bmatrix} \Leftrightarrow \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Както може да се види от определеното уравнение, получихме два напълно различни набора. Обучението на НМ за по-дълъг период от време не прави разлика.

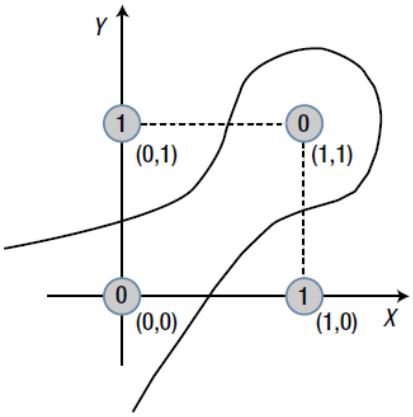
Какво всъщност се случи?

Илюстрирането на данните за обучение може да помогне за изясняване на този проблем. Нека интерпретираме трите стойности на входните данни съответно като координати X, Y и Z. Тъй като третата стойност (координатата Z) е фиксирана като 1, данните от обучението могат да бъдат визуализирани върху равнина, както е показано на следващата фигура.



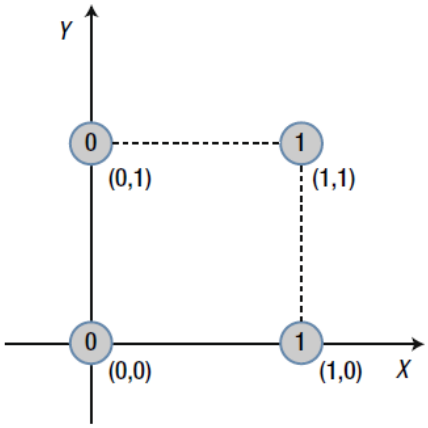
Фигура 6 – Интерпретиране на трите стойности на входните данни като координати X, Y и Z.

Стойностите 0 и 1 в кръгчетата са правилните изходи, определени за всяка точка. Едно нещо, което трябва да се забележи от тази фигура, е, че не можем да разделим областите на 0 и 1 с права линия. Можем обаче да ги разделим със сложна крива, както е показано на следващата фигура. За този тип задачи се казва, че са линейно неделими.



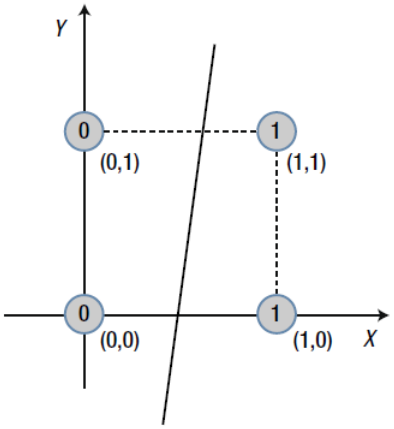
Фигура 7 – Разделяне на 0 и 1 със сложна крива (линейно неразделима).

При същия процес данните за обучение от „Пример: Правило делта“ в равнината X-Y изглежда така:



Фигура 8: Данни за обучение по Правило делта.

В този случай лесно може да се намери права гранична линия, която разделя областите 0 и 1. Това е линейно разделима задача, както е показано на следващата фигура:



Фигура 9: Проблем с линейно разделяне.

Казано по-просто, еднослойната НМ може да решава само линейно разделими проблеми. Това е така, защото еднослойната НМ е модел, който линейно разделя пространството на входните данни. За да преодолеем това ограничение на еднослойната НМ, се нуждаем от повече слоеве в мрежата. Тази необходимост е довела до появата на многослойната NN, която може да постигне това, което еднослойната НМ не може. Само имайте предвид, че еднослойната НМ е приложима за специфични видове проблеми. Многослойната НМ няма такива ограничения. Моля, вижте препратките по-долу за повече подробности.

11

ВНЕДРЯВАНЕ НА НЕВРОННА МРЕЖА

Тази част от ръководството е написана от Ярмила Шкринарова от Департамента по компютърни науки, Факултет по природни науки, Университет „Матей Бел“ в Банска Бистрица, Словакия.

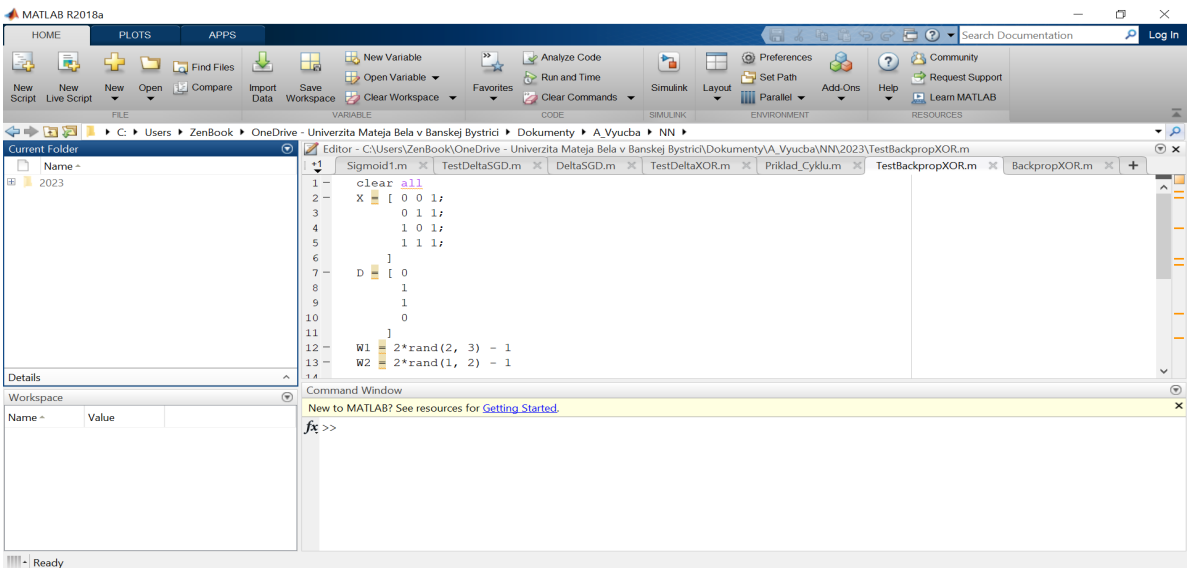
MATLAB е език от високо ниво и интерактивна среда за числени изчисления, визуализация и програмиране за:

- анализ на данни;
- разработване на алгоритми;
- създаване на модели и приложения.

Целта на този раздел е да научите как да работите с Matlab и как да създавате прости невронни мрежи. Представяме методология и три примера за невронни мрежи в графична среда на Matlab. Примерите са ориентирани към задачи за създаване на фитинг функция и класификация.

11.1. Кратко въведение в MATLAB – MAtrix LABoratory

Под лентата с инструменти областта е разделена на четири прозореца, които са предназначени за навигация (придвижване в структурата на директориите), редактиране на изпълними скриптове, показване на работното пространство и команден прозорец (виж Фигура 1).



Фигура 1: Редактор, команден прозорец, работно пространство, навигатор.

Първо ще се научим да работим в командния прозорец, където записваме командите след знака „>>“ (виж Фигура 2).

Фигура 2: Команден прозорец.

Примери за прости изчисления, работа с променливи, вектори и матрици

Изчисления с **променливи –** примери от 1 до 4 могат да се упражняват последователно директно в командния прозорец:

Пример 1	Пример 2	Пример 3	Пример 4
>> 12 + 34 ans = 46	**>> a = 5, b = a^2** a = 5 b = 25	**>> (101+79) /(47 - 17)** ans = 6	**>> 15*12** ans = 180

Векторът е едноизмерен масив от елементи. Обикновено записваме отделните елементи на векторите в квадратни скоби и ги разделяме със запетая или интервал. Обърнете внимание, че записваме бележката след знака %. В следващите части на тази глава ще работим с невронни мрежи и за обучението на невронните мрежи са ни необходими входни и целеви данни. Ако мрежата има един вход, входните данни са едноизмерен масив от елементи (вектор). Ако мрежата има един изход, целевите данни също са едноизмерен масив от елементи [3].

Пример 5:	Пример 6:	Пример 7:	Пример 8:
>> **u1=[1 2 3 4]** %вектор-ред u1 = 1 2 3 4	**>> u2=[1 2 1 2]** %вектор-ред u2 = 1 2 1 2	**>> u1.*u2** %скаларно произведение на два вектора отговор = 1 4 3 8	>> **v=[-1; -7; -3]** %колона-вектор v = -1 -7 -3

Пример 9:	Пример 10:	Пример 11:	Пример 12:
>> w=[1 7 -2]' %транспониран вектор w = 1 7 -2	**>> 6:2:12** % За да генерираме обикновен вектор, определяме първия и последния елемент на вектора и стъпката. отговор = 6 8 10 12	**>> m=15:-3:0** m = 15 12 9 6 3 0	**>> x=12** x = 12 **>> z=[x, 2*x, 3*x]** z = 12 24 36

Пример 13:	Пример 14:
>> W=2*rand(1,3)-1 W = 0.9298 -0.6848 0.9412	**>> x2=linspace(-1, 4, 8)** % -1 до 4 е интервалът и 8 е броят на елементите x2 = -1.0000 -0.2857 0.4286 1.1429 1.8571 2.5714 3.2857 4.0000

За да използваме повече от един вход или цел в невронните мрежи, трябва да подготвим данни под формата на двуизмерни полета. В MATLAB двуизмерните полета се представят чрез **матрици**. Затова ще упражним работата с матрици:

Пример 15:	Пример 16:
>> A=[1 -1 2 -3; 3 0 4 5; 3.2, 5 -6 12] %матрица A = 1.0000 -1.0000 2.0000 -3.0000 3.0000 0 4.0000 5.0000 3.2000 5.0000 -6.0000 12.0000	**>> O=[]** %празна матрица O = []

Пример 17:	Пример 18:
>> **B=[A; u1]** % Разширяване на матрицата с 1 ред (вектор u1). B = 1.0000 -1.0000 2.0000 -3.0000 3.0000 0 4.0000 5.0000 3.2000 5.0000 -6.0000 12.0000 1.0000 2.0000 3.0000 4.0000	**>> C=[A, v]** % Разширяване на матрицата с 1 колона (вектор v). C = 1.0000 -1.0000 2.0000 -3.0000 -1.0000 3.0000 0 4.0000 5.0000 -7.0000 3.2000 5.0000 -6.0000 12.0000 -3.0000

Пример 19:	Пример 20:
>> Z=zeros(2,5) %Създаване на нулева матрица с размер 2 реда на 5 колони. Z = 0 0 0 0 0 0 0 0 0 0	**>> O1=ones(3,4)** %Създаване на единична матрица с размер 3 реда на 4 колкони. O1 = 1 1 1 1 1 1 1 1 1 1 1 1

Пример 21:	Пример 22:
>> A=[1 -1 2 -3; 3 0 4 5; 3.2, 5 -6 12] A = 1.0000 -1.0000 2.0000 -3.0000 3.0000 0 4.0000 5.0000 3.2000 5.0000 -6.0000 12.0000 **>> A(2, :)** %Изписване на втория ред на матрицата A отговор = 3 0 4 5 **>> A(:, 3)** % Изписване на третата колона на матрицата A отговор = 2 4 -6	**>> I=eye(5,8)** %Създаване на диагонална матрица с размер 5 реда на 8 колони. I = 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0

Пример 23:

```
>> R1=rand(3,5)
% Създаване на случайна матрица с размер 3 реда на 5 колони със стойности в диапазона от 0 до 1
R1 =
 0.1419  0.7922  0.0357  0.6787  0.3922
 0.4218  0.9595  0.8491  0.7577  0.6555
 0.9157  0.6557  0.9340  0.7431  0.1712
```

Пример 24:

```
>> R2=randn(4)
% матрица със случайни елементи – стандартно разпределение
R2 =
    0.8884   -2.9443    1.3703    0.3192
   -1.1471    1.4384   -1.7115    0.3129
   -1.0689    0.3252   -0.1022   -0.8649
   -0.8095   -0.7549   -0.2414   -0.0301
```

Пример 25:

```
>> A=[1 5 0; -1 2 3; 1 2 1]
A =
     1     5     0
    -1     2     3
     1     2     1
>> c=2
c =
     2
% умножение на матрица по вектор
>> D=A*c
D =
     2    10     0
    -2     4     6
     2     4     2
```

Пример 26:

```
>> B=[1 2 3; 1 2 3; 1 2 3]
B =
     1     2     3
     1     2     3
     1     2     3

>> E=B*D % умножение на матрици
E =
     4    30    18
     4    30    18
     4    30    18
```

Пример 27:

```
>> A =[2 3; 0 10]
B =[1 0; -3 5]
%умножение на матрици

A =
     2     3
     0    10
B =
     1     0
    -3     5
>> C=A*B
C =
    -7    15
   -30    50
```

Пример 28:

```
%скаларно умножение на матрици, матриците A и B са от предишния пример

>> C=A.*B

C =

     2     0
     0    50
```

Често използваме данни, които съдържат много елементи. Затова е практично да запишем тези данни във **файл**, за да можем да ги използваме по-късно. Всички данни, с които сме работили в MATLAB, се съхраняват в работното пространство и могат да се видят в долния ляв прозорец. Можем да покажем информация за съдържанието на работното пространство, като използваме примери 29 и 30.

Пример 29:	Пример 30:
>> who %списък на работното пространство само с имената на променливите, векторите и матриците Вашите променливи са: A B C O ans u v w x z	**>> whos** %работно пространство Име Размер Байтове Клас Атрибути A 3x4 96 double B 4x4 128 double C 3x5 120 double O 0x0 0 double ans 1x1 8 double u1 1x4 32 double u2 1x4 32 double v 3x1 24 double w 3x1 24 double x 1x1 8 double z 1x3 24 double

Можем да запишем всички променливи, вектори и матрици от **работното пространство** във файла (виж пример 31) или само избрани променливи, вектори и матрици (виж пример 32).

Пример 31:	Пример 32:
>> save data %запазване на всички данни във файл data.dat	**>> save data1 u1 u2 v** %запазване на променливите u1, u2 и v в data1.mat

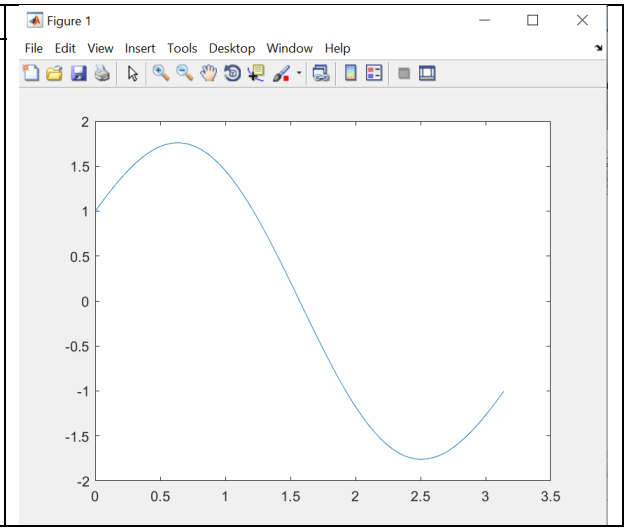
Данните, съхранявани във файлове, могат да бъдат заредени по всяко време в работното пространство на инструмента MATLAB, за да се работи с тях. Вижте примери 33 и 34.

Пример 33:	Пример 34:
>> **load data1** % прочитане на всички данни, записани в data1.mat	>> **load data.dat -MAT** % прочитане на всички променливи, записани в data.dat

Когато **чертаете** хода на дадена функция, броят на елементите по оста x трябва да е същият като този по оста y. Можем да видим начертаната функция на картинката в пример 35. Функцията се изчертава в нов прозорец, който може да се редактира – да се вмъкнат имена на оси, заглавия и др.

Пример 35:

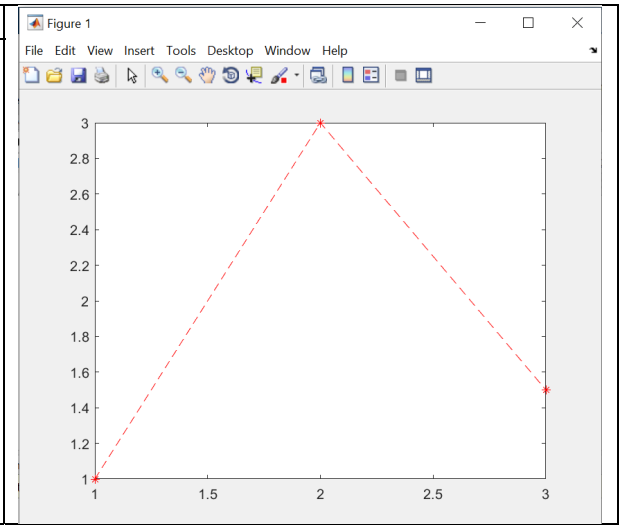
```
>> x=linspace(0, pi);
%чертае функцията
>> y=sin(2*x)+cos(x) ;
>> plot(x,y)
```



Както **цветът**, така и **типът на линията** могат да се променят с командата plot. Вижте пример 36.

Example 36:

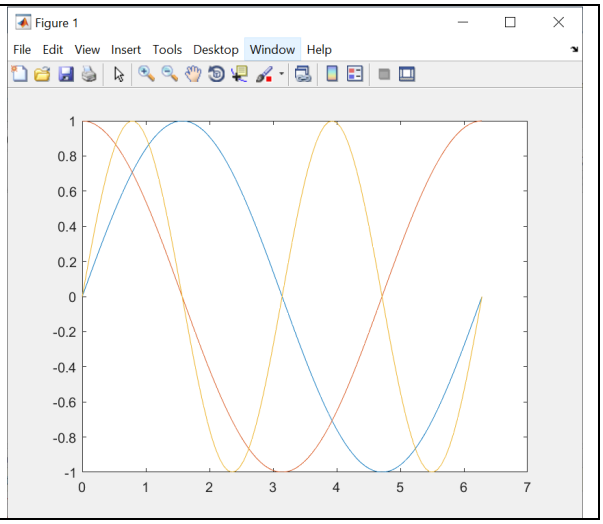
```
>> u=[1 2 3] ;
>> v=[1 3 1.5] ;
>> plot(u,v, 'r--*')
```



Възможно е също така да се **начертаят няколко функции** в едно изображение. Вижте пример 37.

Пример 37:

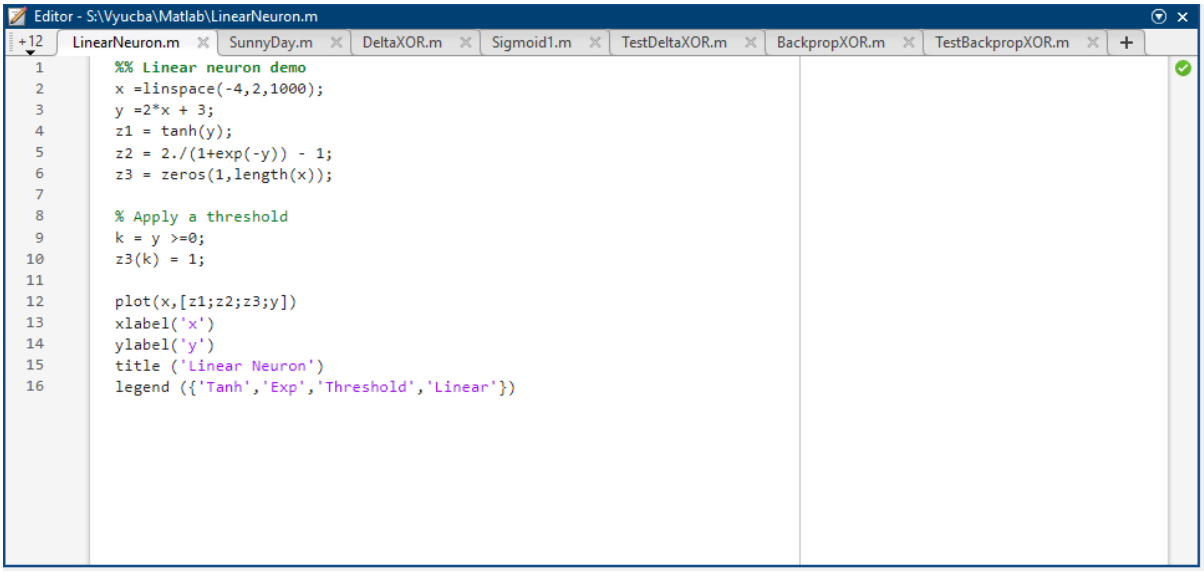
```
>> x=linspace(0, 2*pi);
>> y1=sin(x);
>> y2=cos(x);
>> y3=sin(2*x);
>> plot(x, y1, x, y2, x, y3)
```



За илюстрация ще покажем прост програмен пример (пример 38), в който отделните редове на матрицата X се изписват последователно в цикъл. Изходът от програмата е в дясната колона.

Пример 38:	
>> clear all X = [0 0 1; 0 1 1; 1 0 1; 1 1 1;] ; N = 4; %отпечатва винаги един ред от матрицата X for k = 1:N x = X(k, :) end	x = 0 0 1 x = 0 1 1 x = 1 0 1 x = 1 1 1

В предишните раздели показахме как се пишат команди в командния ред. Това не е практично, ако трябва да напишем много команди. Пишем такива последователни команди в текстов редактор и ги записваме във файл с разширение „m“. Така създаваме **скрипт**, който се отваря и изпълнява в средата на MATLAB. Командите, които пишем в скрипта, трябва първо да се тестват в командния ред на MATLAB, за да се избегнат грешки.



Фигура 3: Пример за M-файл в прозореца на редактора на MATLAB.

11.2. Реализиране на невронни мрежи в MATLAB

В реалния свят често се случва да можем да правим различни измервания, но да не можем да опишем поведението на дадена система с прост математически модел. Това означава, че имаме измерените стойности на входовете на системата и съответните изходи, но не можем да изчислим изходите въз основа на входовете. За тази цел използваме невронни мрежи, за да научим връзката между входовете (от определен интервал от стойности) и изходите или да класифицираме входовете в определени групи. Добре обучената невронна мрежа може да даде правилни изходи за различни входни стойности (от един и същи интервал).

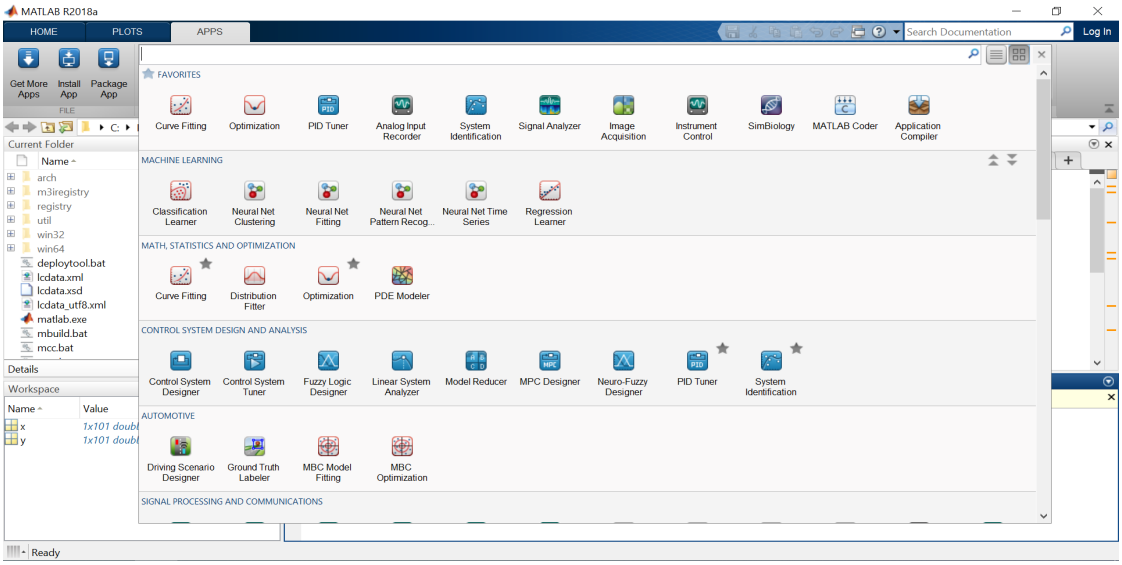
Този подраздел има за цел да определи **методологията** за създаване на невронни мрежи в средата на MATLAB. Впоследствие ще решим прости **примери**, които да ни помогнат да разберем процеса на създаване на невронни мрежи.

Методология за създаване на невронни мрежи в графичната среда на MATLAB

Ще опишем методологията в отделни стъпки:

Стъпка 1: Подготовка на данните. Обикновено се нуждаем от два набора от данни (входящ набор и съответстващ му изходен набор). Ако имаме една извадка от данни, където на един вход съответства един очакван изход (цел), тогава двата набора от данни имат еднакъв размер, т.е. 1 ред х брой колони (извадки).

Стъпка 2: Изберете подходящо приложение от раздела APPS в средата на MATLAB. Например, от категорията Machine Learning (Машинно обучение) изберете приложението Neutral Net Fitting (Фигура 4).



Фигура 4: Приложения, които са част от средата на MATLAB.

Стъпка 3: Зареждаме входните данни от набора от данни и целевите данни от набора от данни (тези, които подготвихме в стъпка 1).

Стъпка 4: Въвеждаме съотношението, в което данните трябва да бъдат разделени на три набора за обучение, валидиране и тестване, например 70%, 15% и 15%.

Стъпка 5: Ще проектираме архитектурата на мрежата. Броят на мрежовите входове и изходи се задава автоматично в зависимост от входните и целевите данни. Необходимо е да се зададе броят на невроните в скритите слоеве. Например, да предположим, че проектираме многослойна перцепторна мрежа, която има 2 скрити слоя. В този случай трябва да зададем броя на невроните за първия и втория скрит слой.

Стъпка 6: Избор на алгоритъм за обучение. Избираме един от готовите алгоритми за обучение, например Левенберг-Маркардт, Байесова регуларизация или мащабиран конюгиран градиент.

Стъпка 7: Започваме процеса на обучение на мрежата. Трябва да се отбележи, че някои стойности са предварително зададени в приложението. Например броят на епохите на обучение може да се зададе на 1000, а точността на обучението се изразява чрез MSE и R. Средната квадратична грешка (MSE) е средната квадратична разлика между изходите на мрежата след и целите преди процеса на обучение. Нашата цел е да получим най-малките стойности на грешките. Нулева стойност означава липса на грешка. Регресията (R) изразява измерената корелация между изходите и целите. Стойност на R от 1 означава тясна корелация, а 0 – липса на корелация, или с други думи, налице е случайна връзка.

Стъпка 8: Процесът на обучение на мрежата приключва, ако постигнатата точност на обучение е достатъчна за нас. В противен случай трябва да се промени архитектурата на мрежата (броят на скритите слоеве и броят на невроните в тях) или да се промени алгоритъмът за обучение, или да се промени броят на обучаващите епохи на мрежата, ако това е възможно. Това означава, че трябва да повторим процедурата от стъпка 5. Трябва да се има предвид, че големият брой обучителни епохи може да доведе до т.нар. преобучение на мрежата.

Пример за създаване на проста функция на невронна мрежа

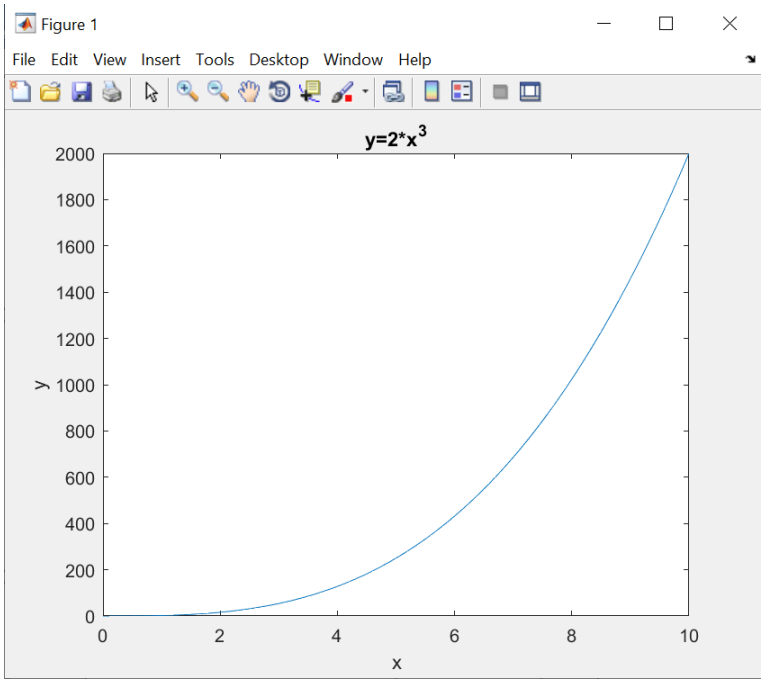
В този пример ще покажем как една невронна мрежа научава стойността на дадена функция. Следваме методологията, представена в Раздел 2.1 на тази глава.

Стъпка 1: За улеснение няма да използваме измерените данни, а ще създадем входните и изходните данни в MATLAB. Използвайки команди в MATLAB, ще създадем два набора от данни (вижте кода по-долу). Първият набор от данни съдържа входни данни с име data1.mat съдържа стойности

на входните данни, а вторият набор от данни е с име data2.mat съдържа стойности на целите, т.е. очакваните изходи след обучението на мрежата. Елементите на входовете и целите са подредени в такъв ред, че отделните входни елементи да съответстват на съответните целеви елементи [2].

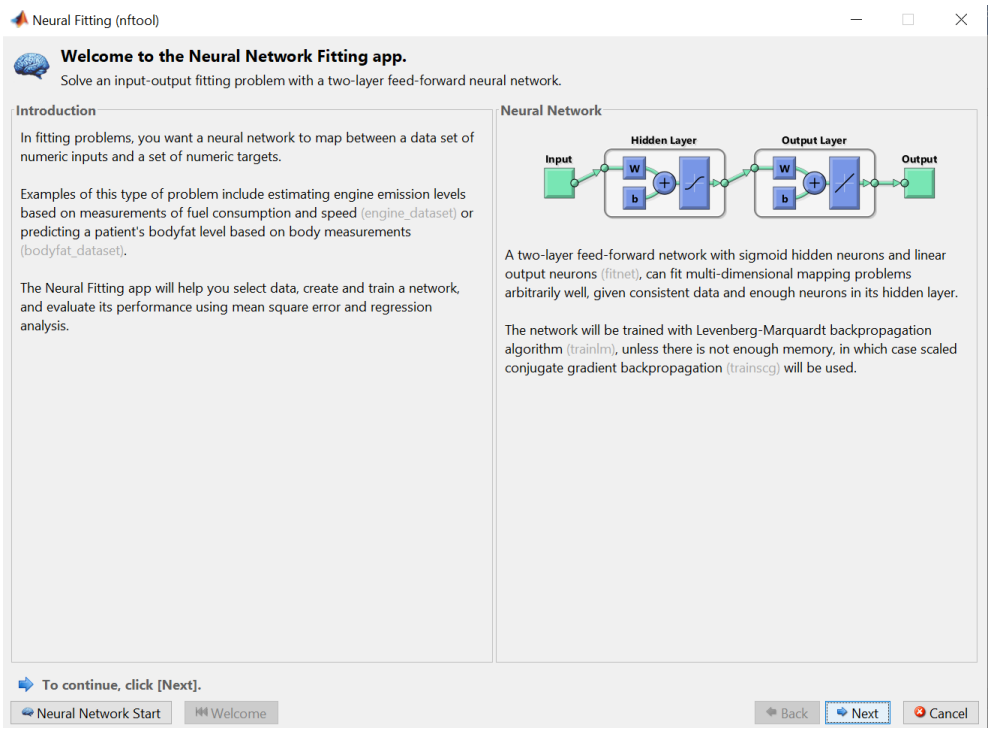
```
>> x=0:0.1:10
>> y=2*x.^3
>> plot(x,y)
>> save data1 x
>> save data2 y
```

След изпълнение на командите от Листинг 1 се изписват стойностите на векторите x и y, изчертава се графиката на функцията (вж. Фигура 5) и наборите от данни се записват във файловете.



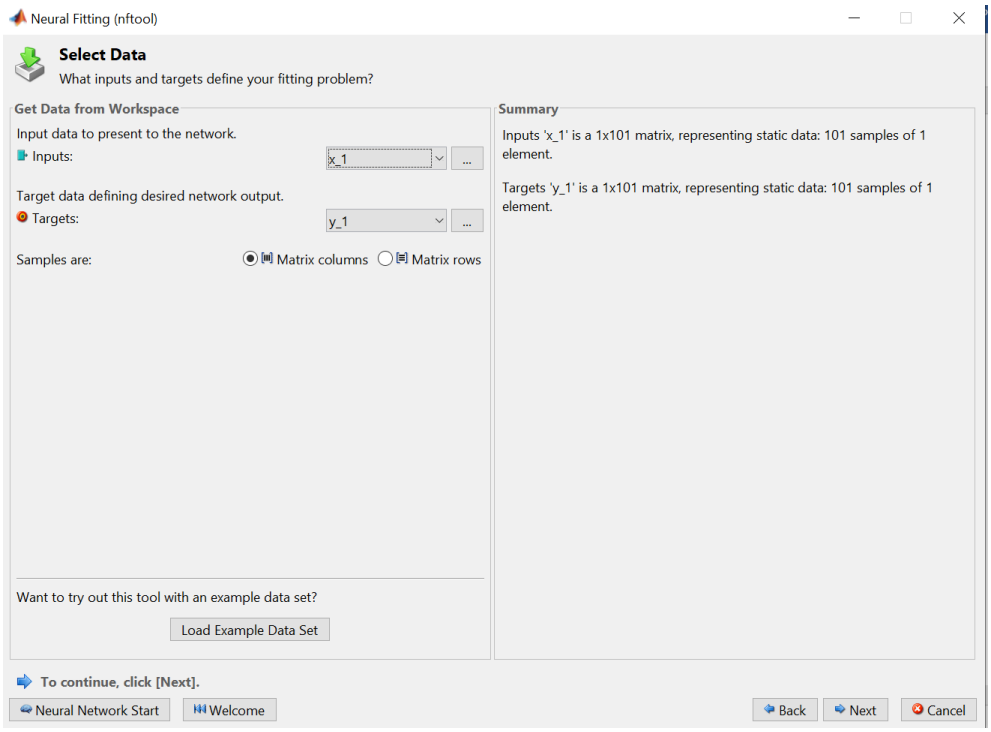
Фигура 5: Графика на функцията $y = 2x^3$.

Стъпка 2: В средата на MATLAB избираме подходящо приложение от раздела APPS. От категорията Machine Learning (Машинно обучение) избираме приложението Neutral Net Fitting (Подбор на неутрална мрежа). Нека стартираме приложението Neutral Net Fitting (виж фигура 6). Навигираме в приложението с помощта на бутона Next (Напред).



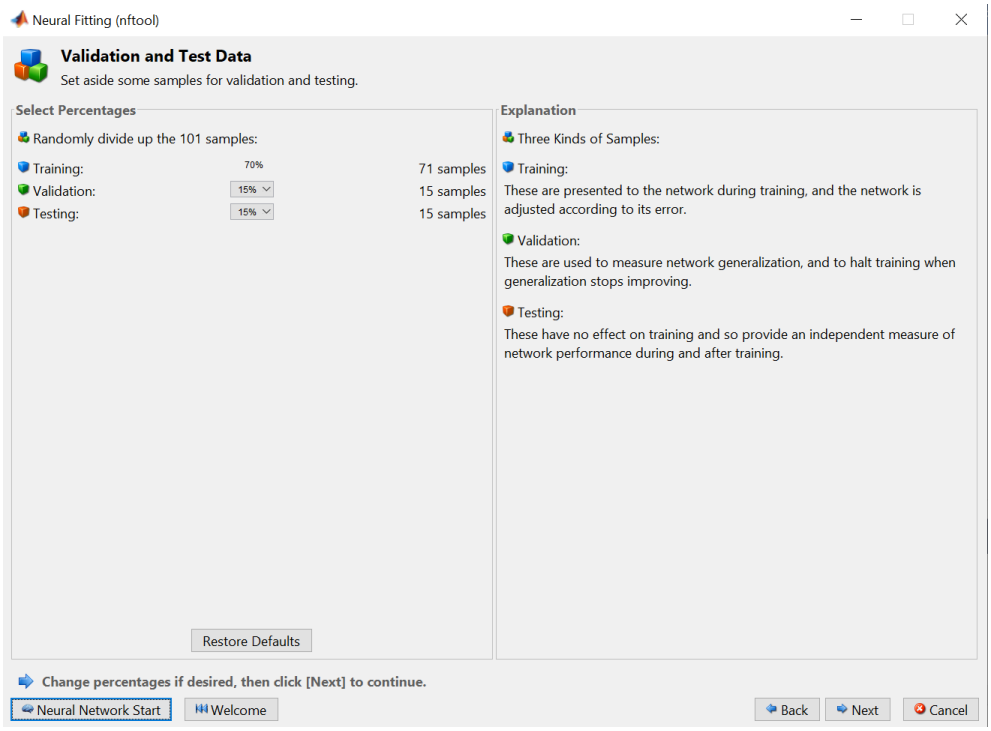
Фигура 6: Настройване на невронна мрежа в MATLAB.

Стъпка 3: Зареждаме набори от входни и целеви данни от подготвени файлове (виж Фигура 7, вляво). Тъй като имаме еднакъв брой входни и целеви данни, гарантираме, че файловете са с еднакъв размер (виж Фигура 7, дясно).



Фигура 7: Метод за зареждане на входни и целеви стойности.

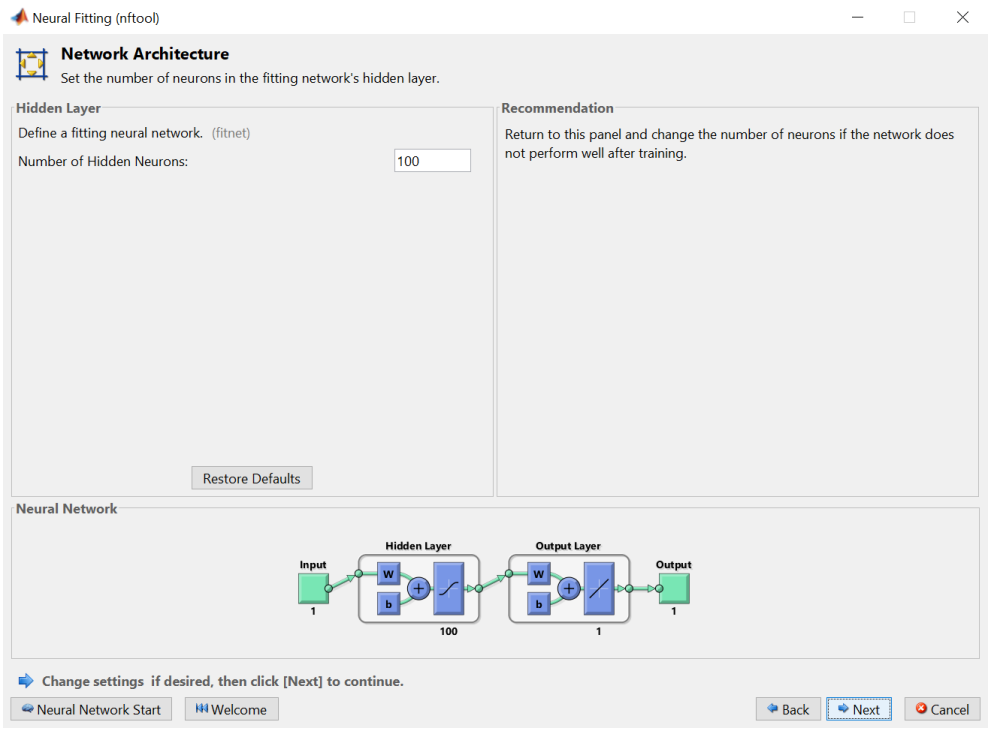
Стъпка 4: Въвеждаме съотношението, в което данните трябва да бъдат разделени на три множества за обучение, валидиране и тестване. В нашия случай това са 70 % от данните за обучение на мрежата, 15 % за валидиране в рамките на процеса на обучение на мрежата и 15 % за тестване (виж Фигура 8).



Фигура 8: Метод на разпределение на извадките от данни в решената задача.

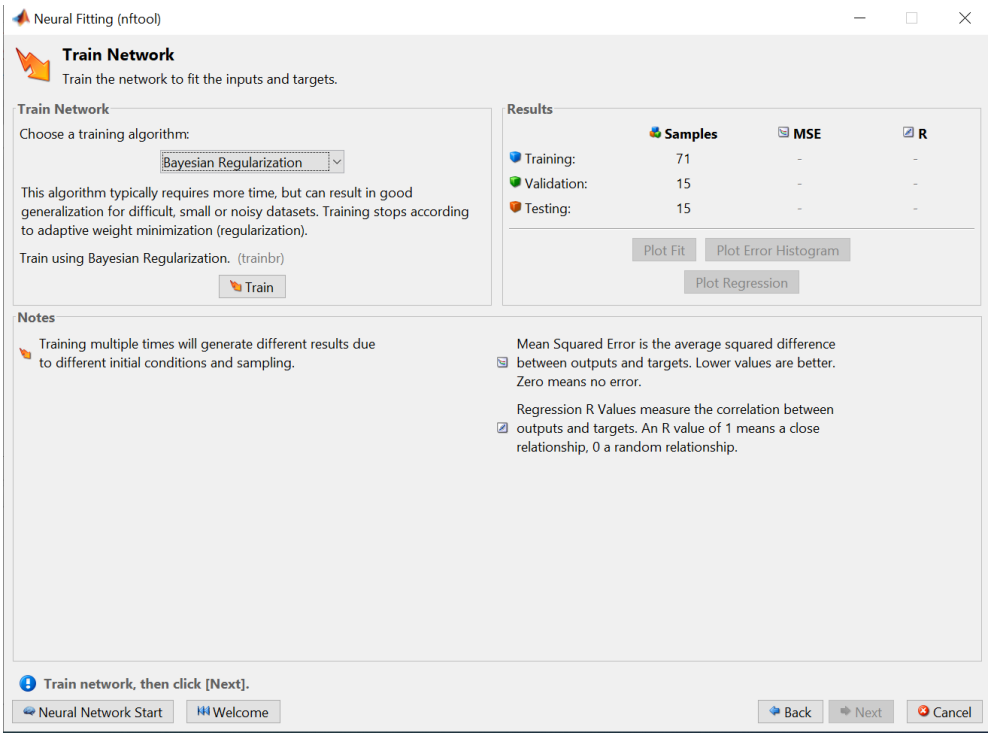
Стъпка 5: Ще проектираме архитектурата на мрежата. Броят на входовете и изходите на мрежата е зададен автоматично в зависимост от входните и изходните данни, така че нашата мрежа има един вход и една цел (вж. Фигура 9).

Изходният слой има само един неврон, защото имаме един изход от мрежата. Броят на невроните в изходния слой също е зададен автоматично. В нашия случай имаме само един скрит слой, тъй като използваме приложението Neutral Net Fitting. Задаваме броя на невроните в скрития слой на 100. Ако мрежата не усвои достатъчно точно връзката между входовете и целите, можем да се върнем към настройката на архитектурата на мрежата и да променим броя на скритите неврони.



Фигура 9: Дизайн на мрежовата архитектура за нашата задача.

Стъпка 6: Избор на алгоритъм за обучение. Можем да изберем един от трите алгоритъма за обучение: Левенберг-Маркардт, Байесова регуларизация или Мащабен конюгиран градиент. Ние ще изберем алгоритъма на Байесовата регуларизация (виж Фигура 10).



Фигура 10: Избор на алгоритъм за обучение.

Стъпка 7: Започваме процеса на обучение на мрежата, като натискаме бутона Train (Обучение). Броят на епохите на обучение е зададен на 1000 и можем да проследим процеса на обучение, както е показано на Фигура 11.

Фигура 11: Напредък в изучаването на мрежата.

Точността на обучението се изразява чрез MSE и R (виж Фигура 12).

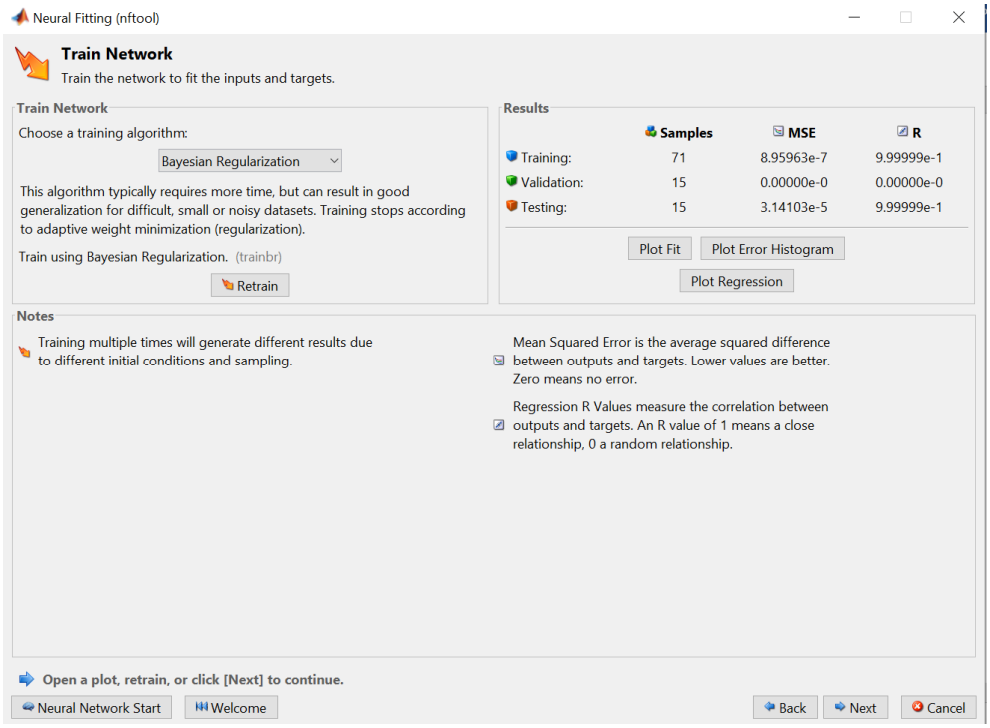
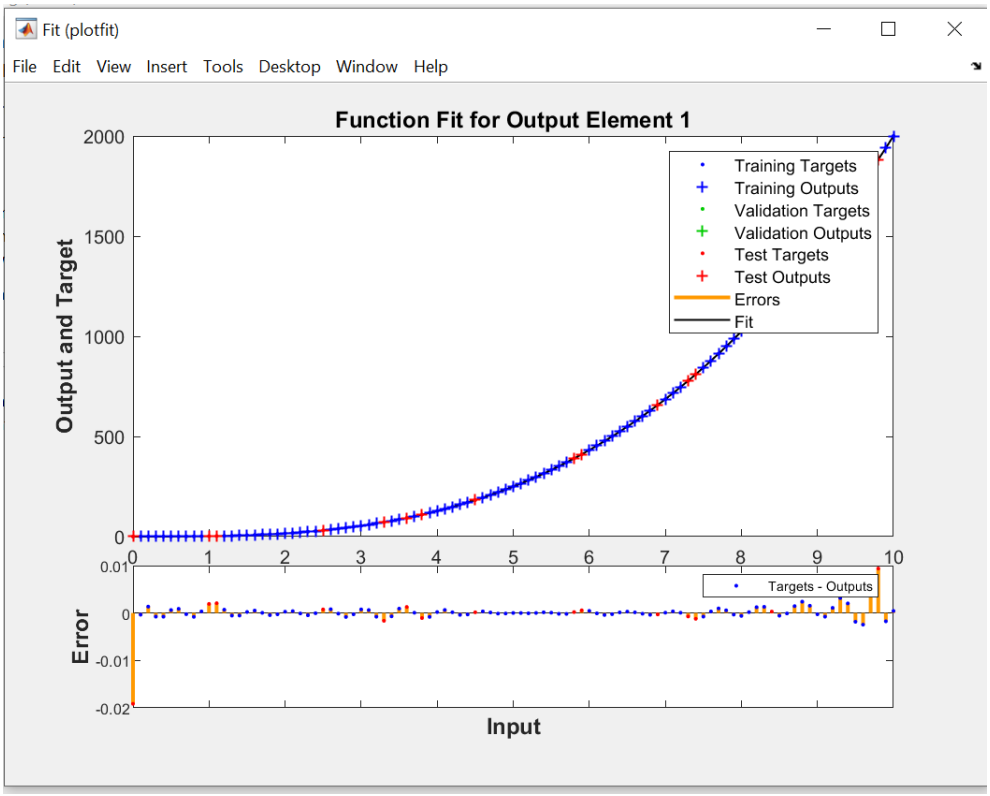


Figure 12: Errors and correlations after the network learning and testing process

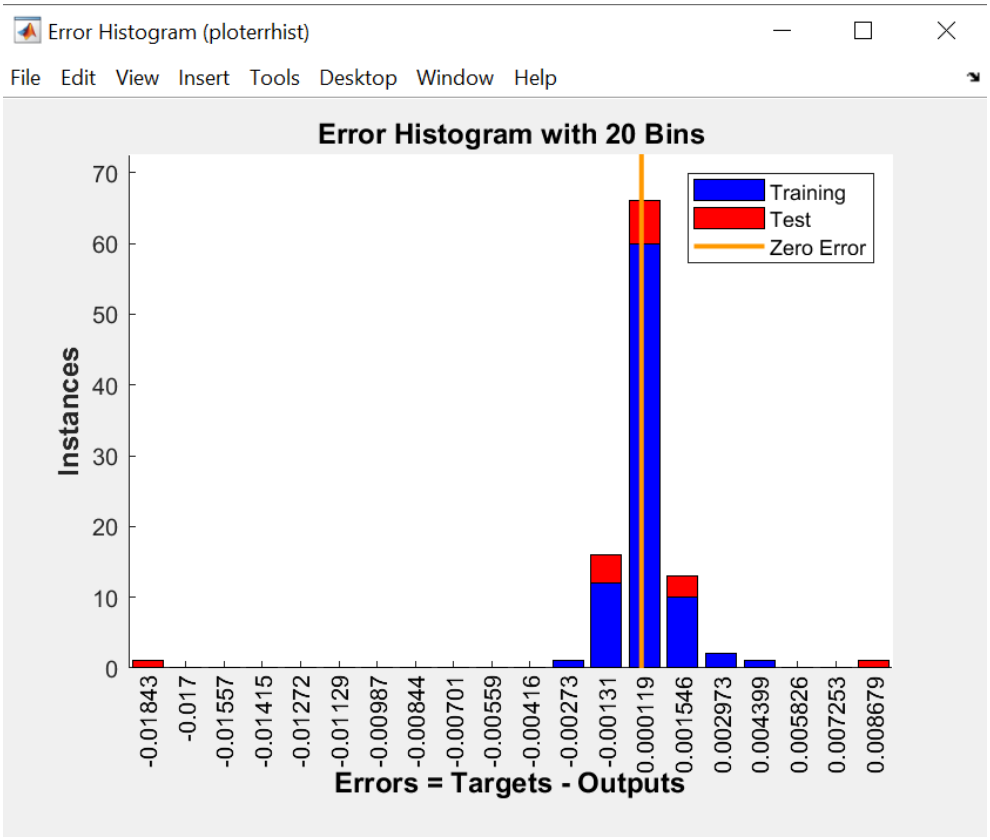
Средната квадратична грешка (MSE) е средната квадратична разлика между изходите на мрежата след и целите преди процеса на обучение. Целта е да се получат най-малки стойности на грешките. Нулева стойност означава липса на грешка. Можем да видим, че мрежата е научила нашия признак с грешка MSE $8.95*10^{-7}$, което е незначителна грешка. Тестването на мрежата потвърди, че мрежата е научила правилно с ниска грешка MSE от $3.14*10^{-5}$. Регресията (R) изразява измерената корелация между изходите и целите. Стойност на R от 1 означава тясна корелация, а 0 – липса на корелация или наличие на случайна връзка. Изчисляването на корелациите след обучението на мрежата и тестването на мрежата достигна стойност 1, което потвърждава, че мрежата е научила много добре връзката между входовете и изходите на мрежата.

Стъпка 8: Можем да заключим, че постигнатата точност на обучение и тестване на мрежата е достатъчна и процесът на обучение на мрежата приключва. Затова можем да видим по-подробно стойностите и напредъка на научената функция, ако последователно натиснем бутоните Plot Fit (Построяване на фит), Plot Error Histogram (Построяване на хистограма на грешките) и Plot Regression (Построяване на регресия) (виж Фигура 12). След натискане на Plot Fit (Построяване на фитнеса) знаем хода на стойностите на целите и изходите на мрежата в зависимост от входовете след обучението и тестването на мрежата (виж Фигура 13).



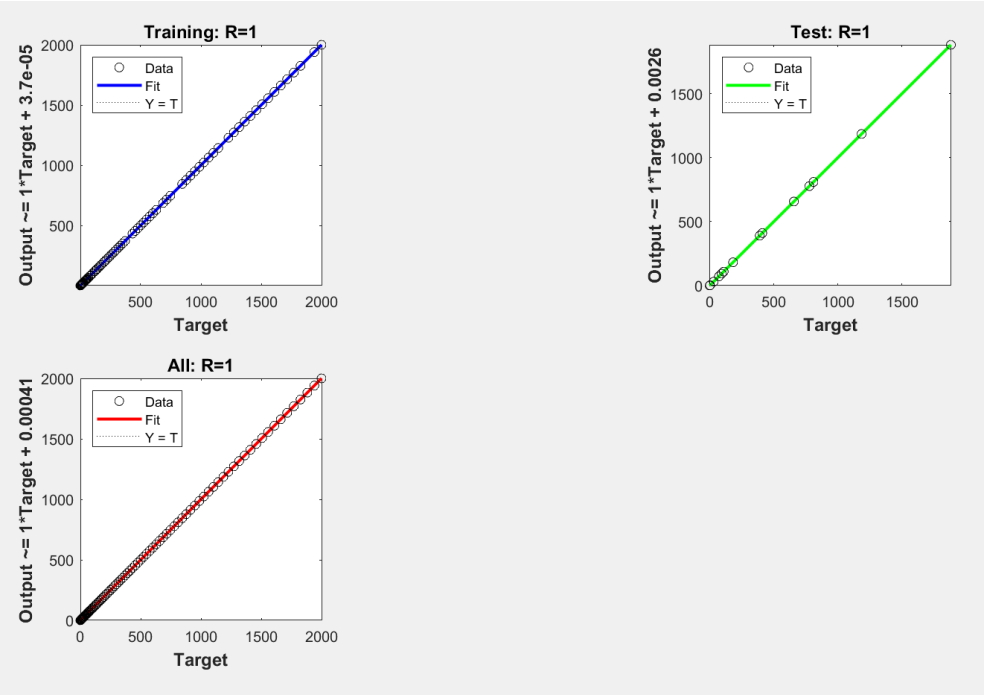
Фигура 13: Напредък на целите и резултатите на мрежата в зависимост от входните данни след процеса на обучение и тестване на мрежата.

След като натиснете **Plot Error Histogram** (Построяване на хистограма на грешките), можете да видите стойностите на грешките и тяхната честота (виж Фигура 14). В този случай абсолютната грешка е разликата между целевата стойност и изхода на мрежата за определен вход на мрежата. Можем да видим, че най-често се среща грешката 0.000119.



Фигура 14: Стойности на абсолютните грешки и тяхната честота.

След натискане на бутона **Plot Regression** (Построяване на регресия) виждаме стойностите на корелация между целевите стойности и изходните стойности в процеса на обучение, процеса на тестване и двата процеса (виж Фигура 15). Изчисляването на корелациите след обучението на мрежата и тестването на мрежата достига стойност 1, което потвърждава, че мрежата е научила много добре връзката между входовете и изходите на мрежата.



Фигура 15: Корелации между целевите стойности и изходните стойности в процеса на обучение, в процеса на тестване и в двата процеса.

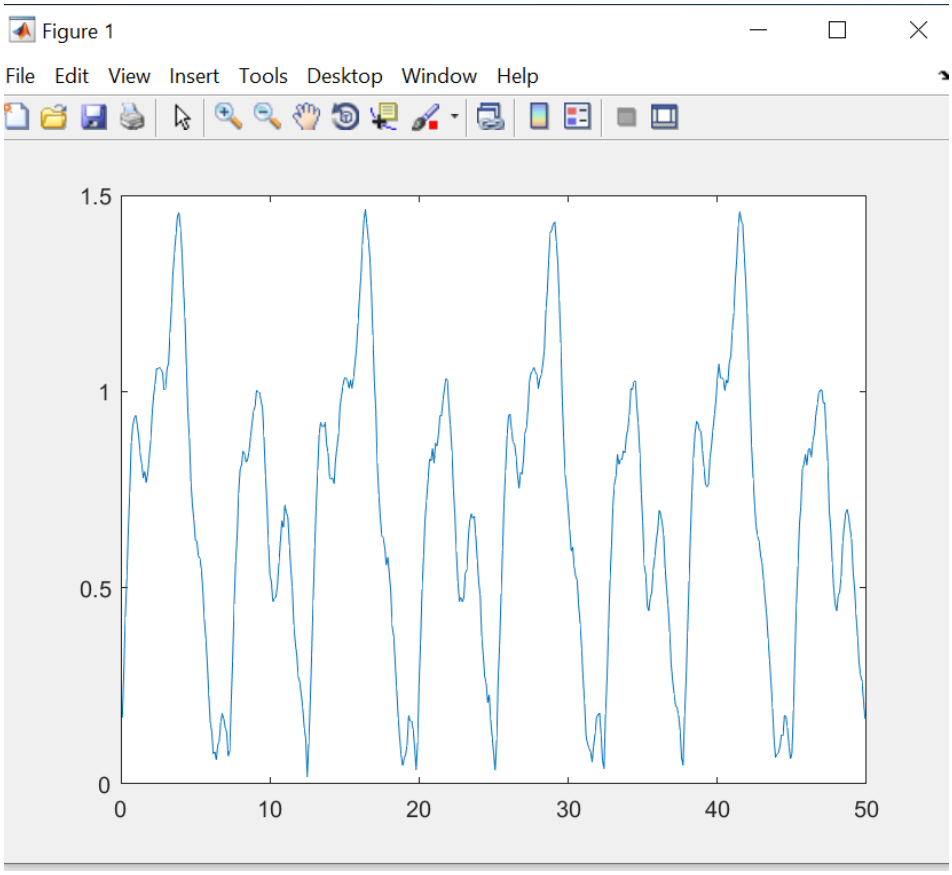
Пример за функция за напасване на невронната мрежа към измерените стойности

Този пример има за цел да научи невронната мрежа на стойностите, които получихме чрез измерване. Следваме методологията, представена в Раздел 2.1 на тази глава.

Стъпка 1: Подготовка на данните. Имаме 500 измерени стойности на данни, съхранени във файла data4 (виж Фигура 16). За по-голяма яснота начертаваме оста x от стойността 0.1 до стойността 50 със стъпка от 0.1. Вижте кода по-долу:

x=0.1:0.1:50

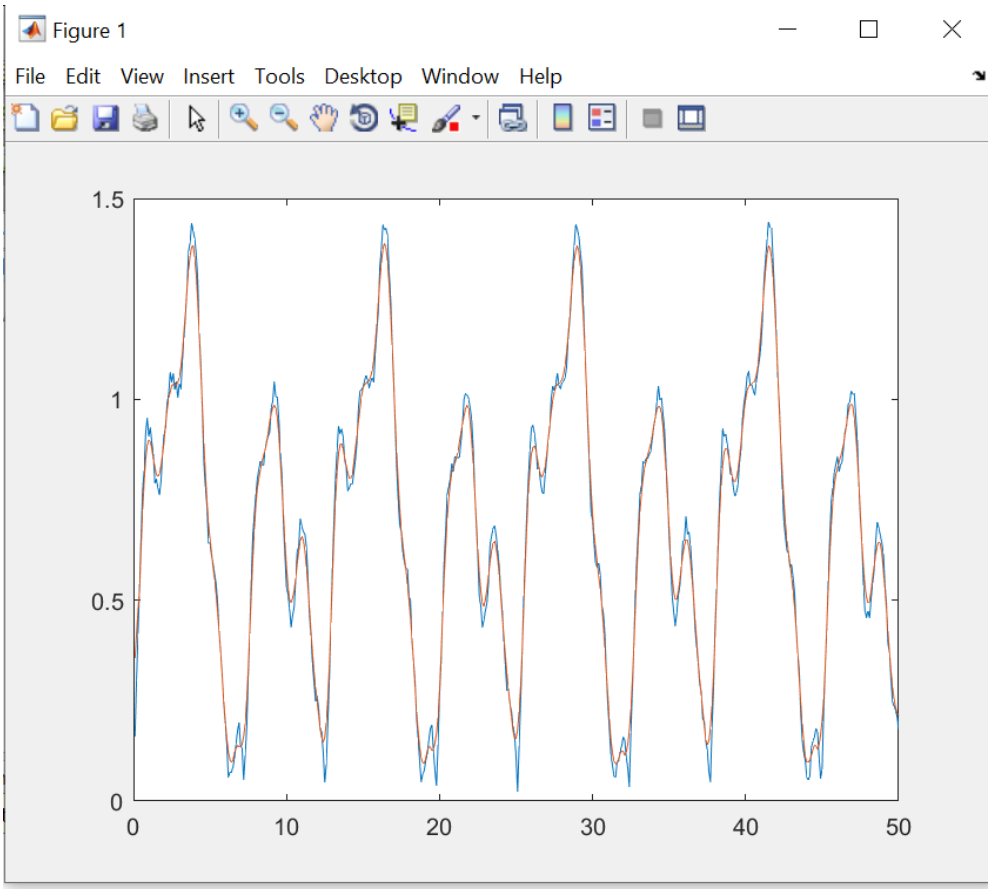
```
save data3 x
load data4 y
plot(x,y)
```



Фигура 16: Измерени стойности.

Понякога измерените данни са зашумени и затова трябва да се коригират [1]. Можем да използваме движеща се средна стойност, която изглажда данните, за да може невронната мрежа да ги научи. Подвижната средна стойност постепенно преминава през всички изгладени стойности и заменя текущата стойност със средната стойност. Прозорецът се състои от текущата стойност и определен брой стойности преди и след текущата стойност. Сега ще покажем как да изгладим нашите измерени данни, съхранени във файла data4.mat. Ще разширим нашия списък 2 с команди, като изгладим измерените данни, използвайки пълзяща средна стойност с прозорец, който е широк девет стойности. Ще запишем модифицираните данни във вектора m във файла data5.mat и ще ги използваме като цели при обучението на мрежата (вижте кода по-долу) и Фигура 17.

```
x=0.1:0.1:50
save data3 x
load data4 y
plot(x,y)
m = movmean(y,7)
plot(x,y,x,m)
save data5 m
```



Фигура 17: Измерените стойности са в син цвят, а изгладените данни – в червен.

Стъпка 2: Избираме приложението Neutral Net Fitting в средата на MATLAB.

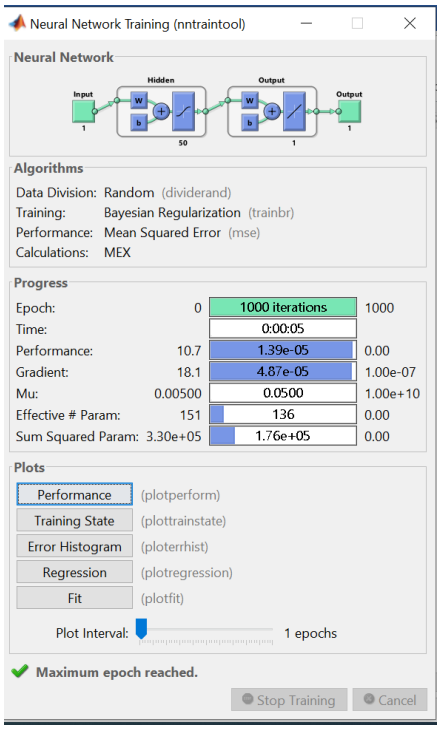
Стъпка 3: Зареждаме данните в приложението. Файлът data3.mat съдържа входните данни, а файлът data5.mat съдържа целите.

Стъпка 4: Оставяме съотношението, както в предишния пример.

Стъпка 5: Ще проектираме архитектурата на мрежата. Избираме 50 неврона в скрития слой.

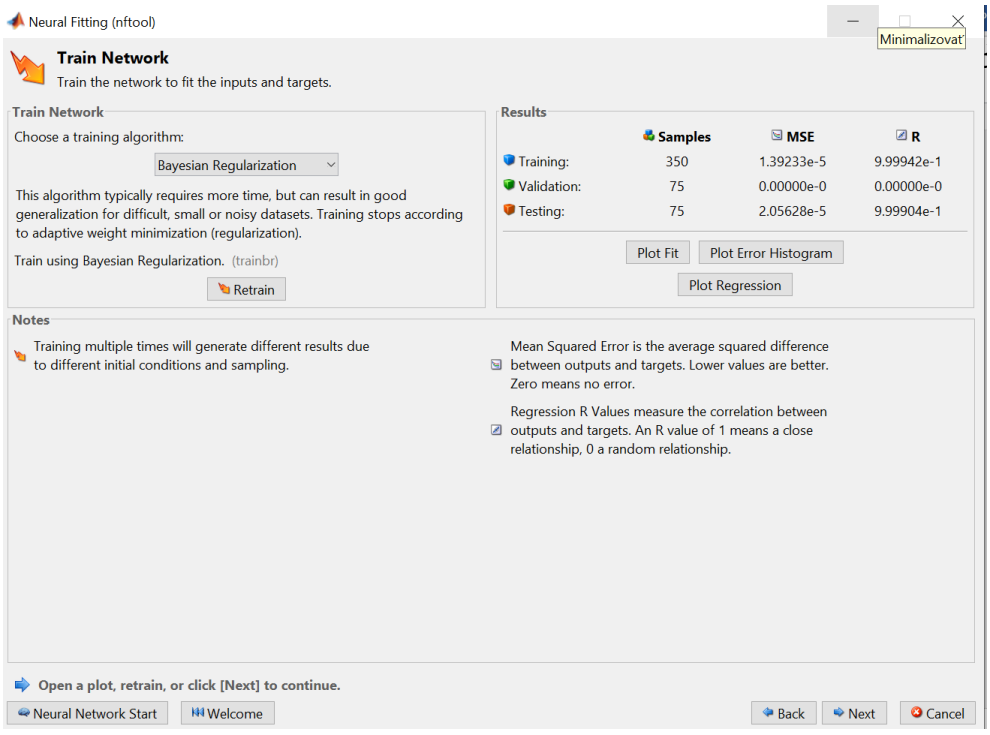
Стъпка 6: Избираме алгоритъма за обучение с Байесова регуларизация.

Стъпка 7: Започваме процеса на обучение (виж Фигура 18).



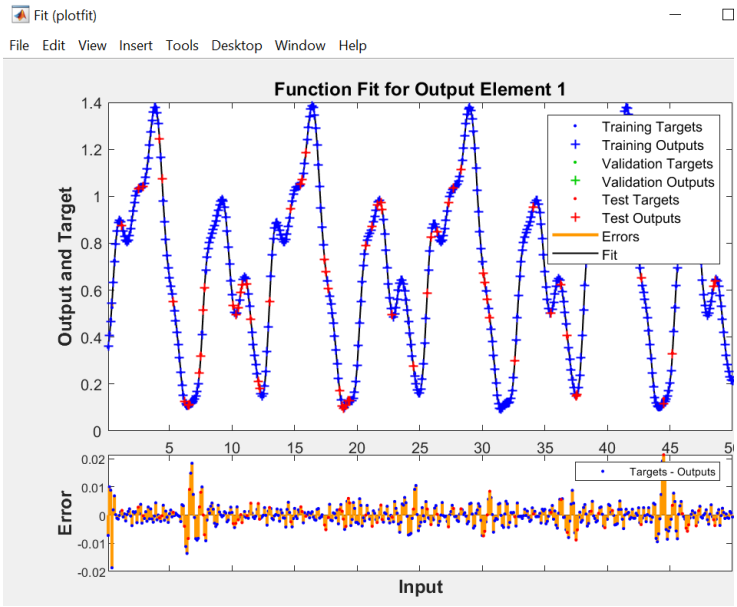
Фигура 18: Напредък в изучаването на мрежата.

Стъпка 8: Нека разгледаме по-отблизо резултатите от обучението на мрежата (Фигура 19). Стигаме до заключението, че мрежата е научила правилните стойности въз основа на грешката при обучението от $1.39x10^{-5}$ и грешката при тестването на мрежата от $1.39x10^{-5}$.



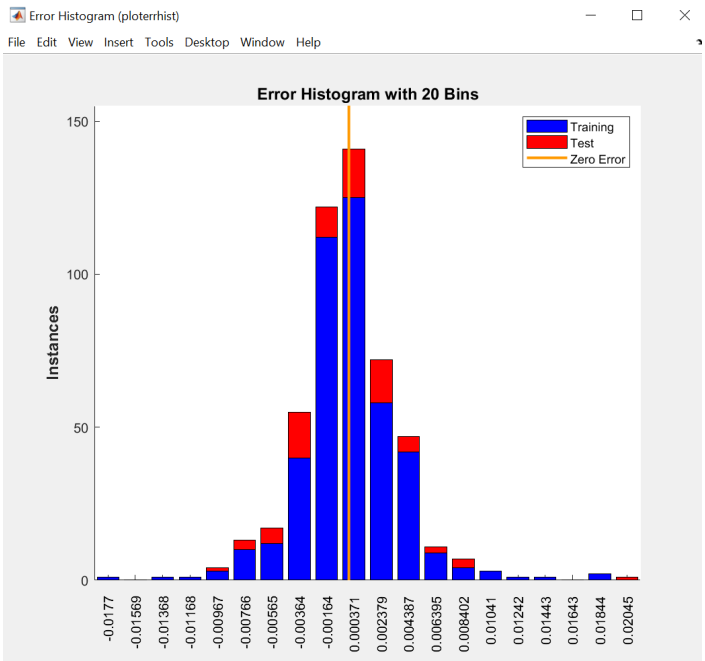
Фигура 19: Показване на грешки при обучението.

На фигура 20 се вижда развитието на стойностите на целите и изходите на мрежата в зависимост от входните данни след обучение и тестване на мрежата. Научените стойности се припокриват с целевите стойности. В долната част на изображението виждаме показаните грешки, които са минимални.



Фигура 20: Резултати от обучението в мрежата – цели за напредък и усвоени стойности.

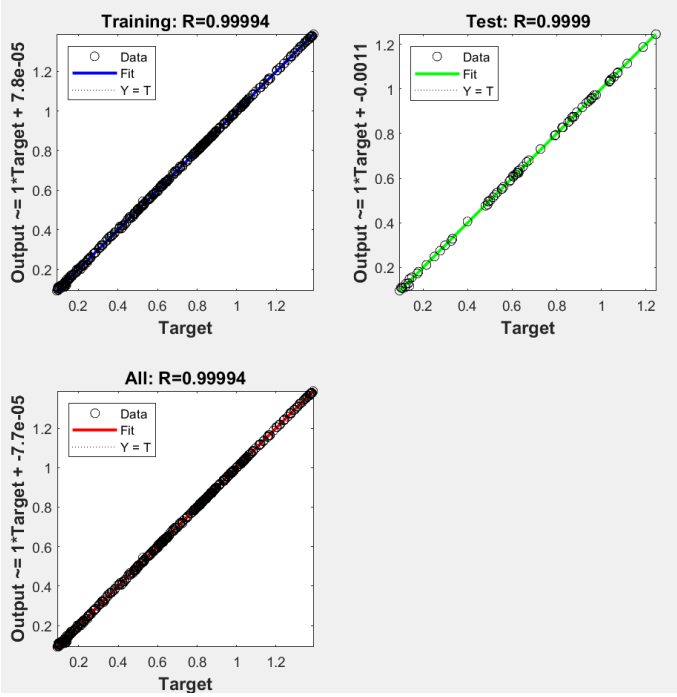
На хистограмата на фигура 21 се виждат размерът и честотата на грешките, което отново доказва, че грешките са минимални.



Фигура 21: Резултати от обучението на мрежата – хистограма – размер и честота на грешките.

Показаните регресии (виж Фигура 22), които достигат стойност 1, от гледна точка на обучените, тестваните и всички входове еднакво показват, че имаме много добре обучена мрежа.

Като цяло можем да заключим, че постигнатата точност на обучението и тестването на мрежата е достатъчна и процесът на обучение на мрежата приключва.



Фигура 22: Резултати от мрежовото обучение – регресионно изчисление.

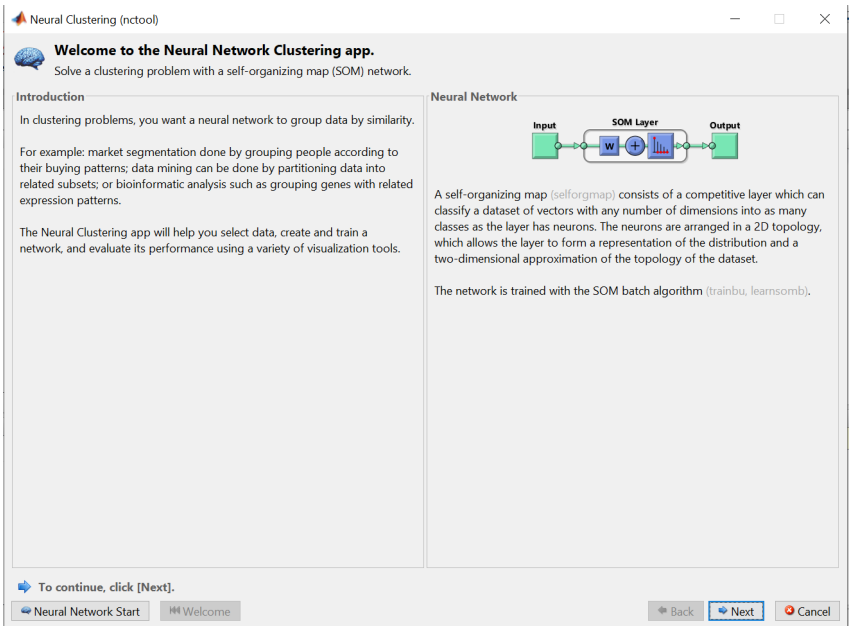
Пример за клъстериране на данни чрез невронна мрежа със самоорганизираща се карта

В този пример решаваме добре познатата задача за класифициране на цветя ирис. Ще използваме набора от данни IRIS и ще опишем решен пример, който е част от средата на Matlab. Цветовете на ириса могат да бъдат описани с помощта на 4 параметъра, а стойностите (дължина на чашелистчето, ширина на чашелистчето, дължина на венчелистчето и ширина на венчелистчето) в набора от данни са дадени в сантиметри. Следователно всяко цвете се характеризира с 4 елемента. Нашата задача е да създадем такава невронна мрежа със самоорганизираща се карта, която да класифицира типовете цветя на ириса в класове, така че сходните типове да се намират в група, близка един до друг. Картата се създава въз основа на сходството на образците, а научената невронна мрежа може да класифицира дори непознати образци [4].

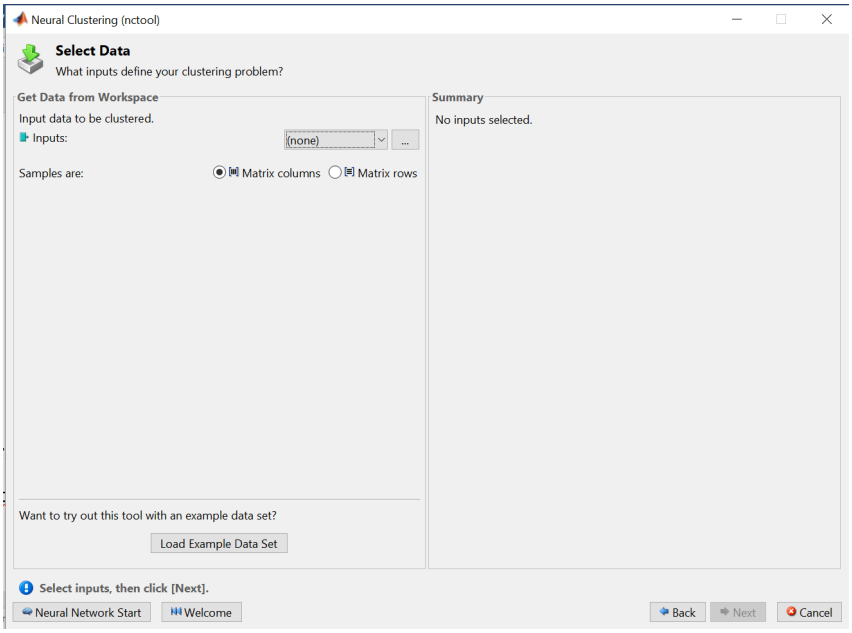
Продължаваме по методологията за реализация на невронни мрежи в графичната среда на MATLAB.

Стъпка 1: Прескачаме тази стъпка. Ще използваме готов набор от данни, наличен в MATLAB, и ще опишем този набор от данни подробно в стъпка 3.

Стъпка 2: В средата на MATLAB изберете подходящото приложение от раздела APPS, от категорията Machine Learning (Машинно обучение) изберете приложението Neutral Net Clustering и стартирайте приложението. Това приложение ще ни помогне да създадем невронна мрежа със самоорганизираща се карта. Вижте Фигура 23.

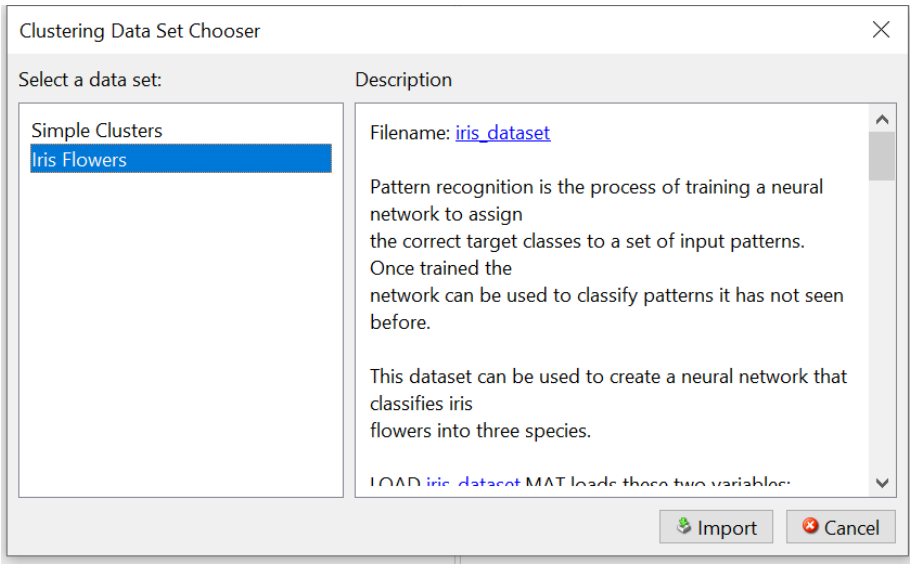


Фигура 23: Приложение за клъстериране на невронни мрежи.

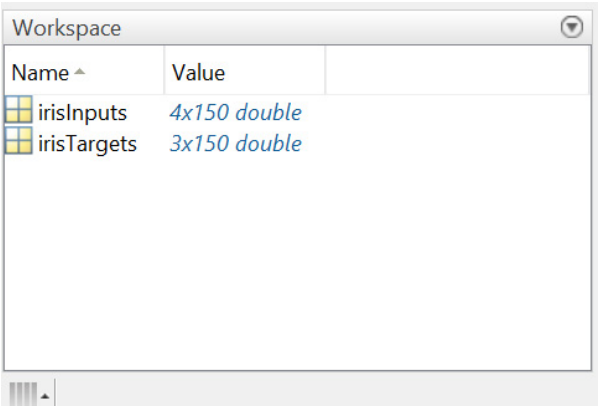


Фигура 24: Зареждане на подготвен набор от данни.

Стъпка 3: Зареждаме наборите от данни. Вижте Фигура 24. Тъй като искаме да използваме подготвения набор от данни IRIS (виж Приложение А), натискаме Load Example Data Set (Зареждане на примерен набор от данни), избираме Iris Flowers (Цветя на ириса) и импортираме данните. Вижте Фигура 25.



Фигура 25: Импортиране на набор от данни за цветя от ирис (Iris).



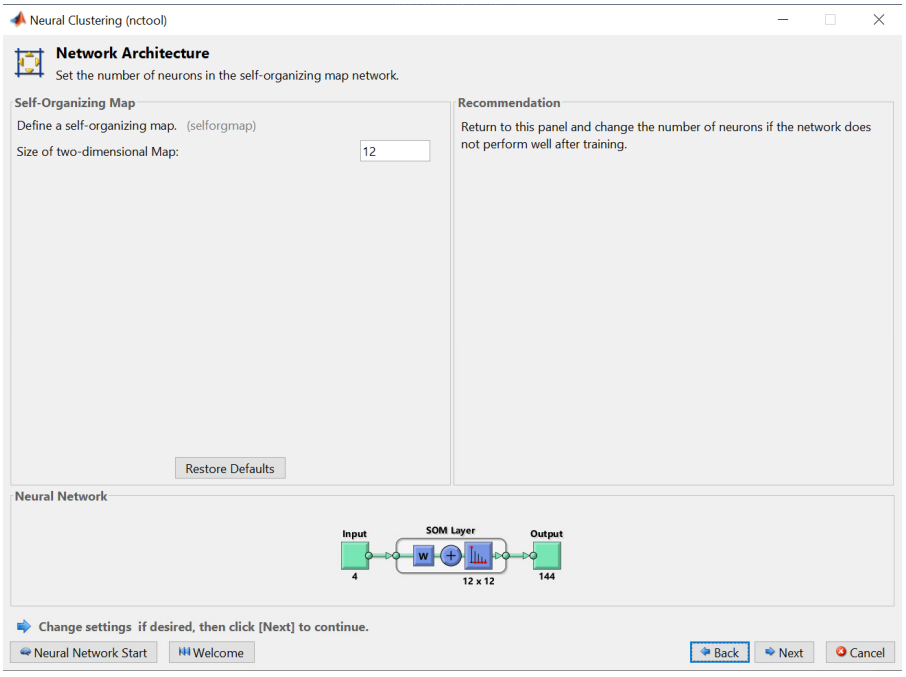
Фигура 26: Работно пространство на MATLAB след зареждане на набора от данни за цветя от ирис.

След зареждане на набора от данни матриците irisInputs и irisTargets се показват в прозореца на работното пространство на MATLAB, виж Фигура 26. Виждаме, че матрицата irisInputs има размери 4 реда x 150 колони. Четири параметъра описват едно цвете; следователно една колона представлява една проба от цвете. Входният набор от данни съдържа 150 проби на цветя. Матрицата irisTargets определя класификацията на всяка входна проба в един от 3 класа. Ще запишем двете матрици в командния прозорец поотделно, за да разберем по-добре данните. Образците на данните могат да се видят в кодовете по-долу:

```
>> irisInputs
irisInputs =
  Columns 1 through 11
    5.1000    4.9000    4.7000    4.6000    5.0000    5.4000    4.6000    5.0000    4.4000    4.9000    5.4000
    3.5000    3.0000    3.2000    3.1000    3.6000    3.9000    3.4000    3.4000    2.9000    3.1000    3.7000
    1.4000    1.4000    1.3000    1.5000    1.4000    1.7000    1.4000    1.5000    1.4000    1.5000    1.5000
    0.2000    0.2000    0.2000    0.2000    0.2000    0.4000    0.3000    0.2000    0.2000    0.1000    0.2000


>> irisInputs
irisTargets =

  Columns 1 through 18

     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0

  Columns 19 through 36

     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0

  Columns 37 through 54

     1     1     1     1     1     1     1     1     1     1     1     1     1     1     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     1     1     1     1
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
```

Стъпка 4: Разделянето на извадките на тренировъчни и тестови е предварително зададено и затова не е необходимо да го въвеждате в това приложение.

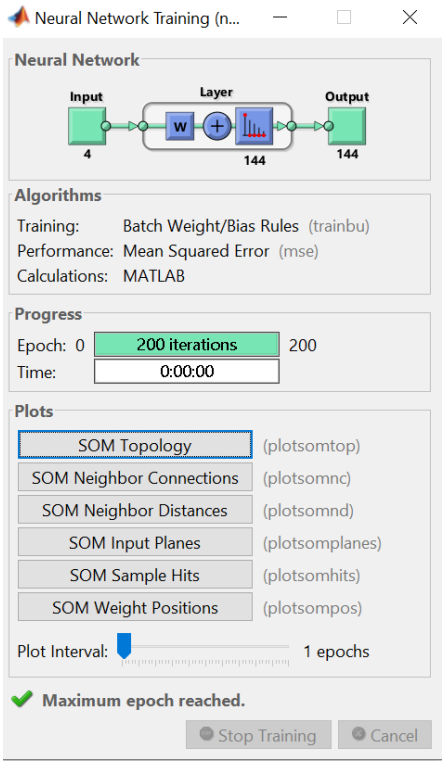
Стъпка 5: Ще проектираме архитектурата на мрежата. В този случай трябва да се въведе броят на невроните на слоя SOM. Един слой представлява двуизмерен квадратен масив. Следователно, когато зададем размер на масива 12, се създава двумерен масив от 12x12 елемента. Вижте Фигура 27. Тогава картата на изхода на мрежата ще бъде 12x12, т.е. 144 елемента. Предварително зададената топология на изходната карта е шестоъгълна.



Фигура 27: Дизайн на мрежовата архитектура.

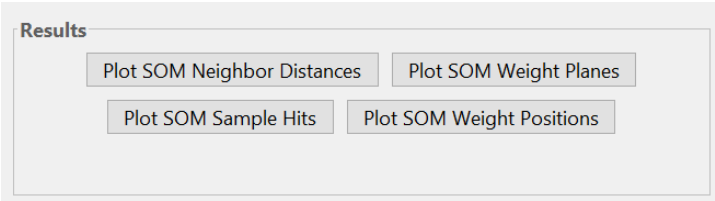
Стъпка 6: Избор на алгоритъм за обучение. Приложението разполага с предварително дефиниран алгоритъм SOM, така че прескачаме тази стъпка.

Стъпка 7: Започваме процеса на обучение на мрежата, като натискаме бутона Train (Обучение). Броят на епохите на обучение е зададен на 200 и можем да проследим процеса на обучение, както е показано на фигура 28.

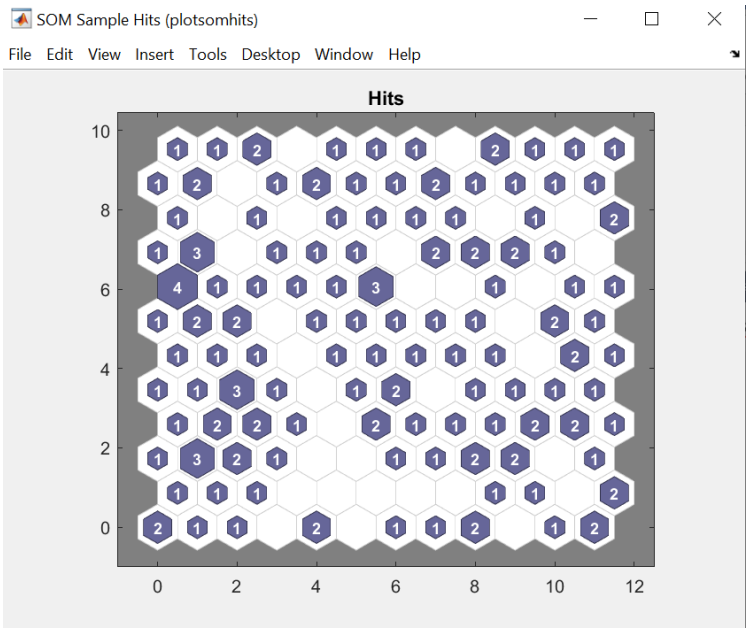


Фигура 28: Напредък в обучението на мрежата.

Стъпка 8: Резултатите от обучението на мрежата се показват с помощта на четири бутона (виж Фигура 29). Първият резултат е класификацията на класовете за всяко цвете, а Plot som sample hits (виж Фигура 30) показва броя на цветята във всеки клас. Областите от неврони с по-високи стойности представляват класове от сходно често представяни цветя. Обратно, областите с малки стойности означават цветя с по-рядко присъствие.

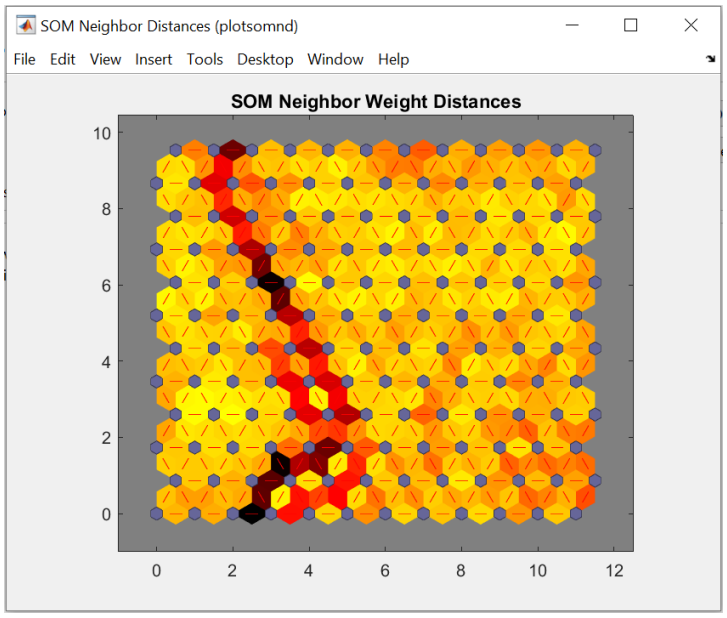


Фигура 29: Резултати от обучението на мрежата.



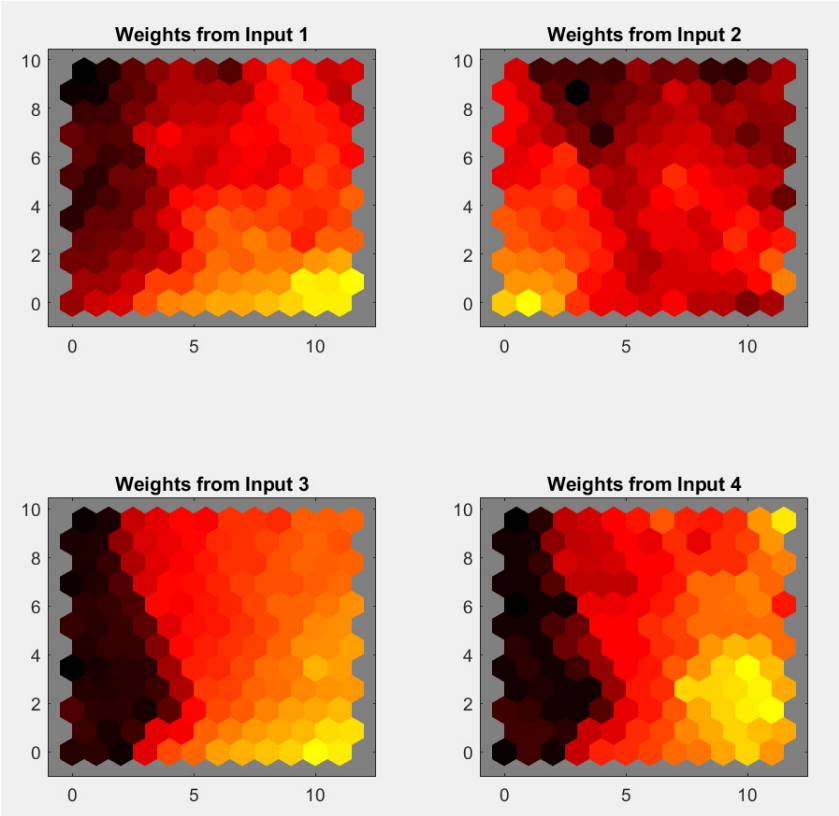
Фигура 30: Резултати от мрежовото обучение – брой цветя във всеки клас.

Резултатът, който показваме с помощта на Plot SOM Neighbor Distances, изразява евклидовото разстояние на класа неврони от техните съседи. Групите неврони, които образуват ярки връзки, означават голямо сходство на цветята във входното множество. Обратно, тъмно-светлите връзки представляват култивирани области с по-малко цветя или области без цветя. Вижте Фигура 31. Тъмните граници (стави) разделят големи области от входното пространство и показват, че цветята в отделните области имат различни характеристики.



Фигура 31: Резултати от обучението на мрежата – изобилие и класове цветя във входното пространство.

Получените тегла на мрежата от гледна точка на четирите входни характеристики на цветята ще бъдат показани с помощта на тегловните равнини на Plot SOM. Вижте Фигура 32.



Фигура 32: Резултати от обучението на мрежата – карти на теглото за отделните входове на мрежата.

Теглата свързват всеки вход с всеки от 144-те изходни неврона на мрежата. Тъмните цветове представляват по-големи тегла. Входовете, които имат един и същ цвят на картата, са силно корелирани.

12

ПРИЛОЖЕНИЯ

Този раздел съдържа приложения към работата, представена в основната част на наръчника. Има **четири приложения**, които съдържат различни данни и информация за упражнения, свързани с наръчника и курса, в който може да се използва наръчникът, а именно:

- **Приложение А** се отнася до описанието на набора от данни Iris, използван в примерите от раздели 3-8.
- **Приложение Б** съдържа примери за решения на задачи, представени в раздел 7.
- **Приложение В** се фокусира върху представянето на избрани набори от данни за замърсяването на въздуха и изменението на климата, които могат да се използват като източник за анализ на данни.
- **Приложение Г** описва въздействието на замърсяването на въздуха върху човешкото здраве.
- **Приложение Д** съдържа предложена учебната програма за курс, в който може да се използва наръчникът.

A

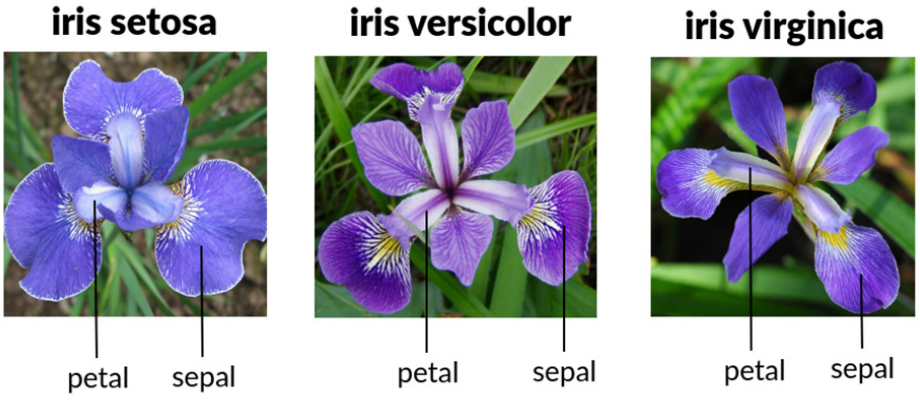
КРАТКО ОПИСАНИЕ НА НАБОРА ОТ ДАННИ IRIS

Тази част от ръководството е написана от Алжбета Михаликова и Адам Дудаш от Департамента по компютърни науки, Факултет по природни науки, Университет „Матей Бел“ в Банска Бистрица, Словакия.

Наборът от данни IRIS е един от най-често използваните набори от данни в областта на анализа на данни, работата с модели за прогнозиране и настройката на алгоритми, предназначени за обработка на данни.

Този набор от данни е създаден от Едгер Андерсен и е представен за първи път в контекста на анализа на данни в публикацията Фишер, Р. А. „*The use of multiple measurements in taxonomic problems*“ *Annual Eugenics*, 1936. Наборът от данни се състои от **пет атрибута**, измерени върху **150 индивида** от цветето ирис (следователно наборът от данни е с размер 150×5). Цветето се състои от шест листа, структурирани в два цикъла, в които:

- чашелистчетата образуват вътрешния кръг на цветето;
- венчелистчетата образуват външния кръг на цветето.



Наборът от данни за ирисите е съставен чрез измерване на две стойности за всеки тип листа (ширина и дължина), което създава **четири цифрови атрибута**:

- дължина на чашелистчетата и ширина на чашелистчетата, измерени в сантиметри или милиметри,
- дължина на венчелистчетата и ширина на венчелистчетата, измерени в сантиметри или милиметри.

Петият атрибут на набора от данни е **класът** с категорична стойност или понякога **вид**, който разделя единиците от набора от данни на три класа:

- iris setosa,
- iris versicolor,
- iris virginica.

Всеки от тези класове е представен в набора от данни равномерно – от **50 субекти**. Представяме пример за по един примерен обект от всеки клас от набора данни Iris:

субект	Дължина на чаше-листчетата	Ширина на чашелистче-тата	Дължина на венчелистче-тата	Ширина на венче-листчета	Клас
1	5.1	3.5	1.4	0.2	setosa
2	7.0	3.2	4.7	1.7	versicolor
3	6.3	3.3	6.0	2.5	virginica

Работа с набора от данни IRIS

Наборът от данни IRIS е толкова стандартизиран, че повечето инструменти за обработка и анализ на данни имат вътрешна команда, която може да се използва за зареждане на този набор от данни.

Например в езика R вместо името на файла с данни използвайте само *iris*.

Пример: При въвеждане на името на заредената съвкупност от данни в R получаваме конзолен изход, съдържащ всички атрибути и същности на съвкупността от данни. В случая с набора от данни iris можем просто да въведем iris (без да е необходимо зареждане на набора от данни).

```
> iris
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1            5.1         3.5          1.4         0.2     setosa
2            4.9         3.0          1.4         0.2     setosa
3            4.7         3.2          1.3         0.2     setosa
4            4.6         3.1          1.5         0.2     setosa
5            5.0         3.6          1.4         0.2     setosa
```

В случай че работите с инструмент, който не разполага с набор от данни за ириса по този начин, е възможно свободно да изтеглите набора от данни, например на адрес:

https://archive.ics.uci.edu/ml/datasets/iris

Б РЕШЕНИЯ НА ВЪПРОСИ ЗА „РАЗМИТА“ КЛАСИФИКАЦИЯ

Тази част от ръководството е написана от Алжбета Михаликова от Департамента по компютърни науки, Факултет по природни науки, Университет „Матей Бел“ в Банска Бистрица, Словакия.

С помощта на метода на Сугено класифицирайте данните от набора от данни Iris в подходящ брой класове.

Решение:

Отговорете на следните въпроси:

1. Колко **входни променливи** има в набора от данни IRIS?

Имаме четири входни променливи в набора от данни IRIS.

2. Какво ще използваме, **за да опишем входните променливи**?

За да опишем входните променливи, ще използваме размити функции на принадлежност.

3. Какъв тип **размити функции на принадлежност** ще използваме?

Ще използваме трапецовидни функции на принадлежност.

4. Какъв ще бъде **резултатът**?

Резултатът (изходът) ще бъде конкретният клас, към който принадлежат отделните цветя на ириса (редове от таблицата/обекти).

5. Какво ще използваме, за да **опишем изходните променливи**?

За да опишем изходните променливи, ще използваме константни функции (константи).

6. **Какъв тип правила** ще използваме?

Ще използваме правилата на Сугено IF-THEN.

7. **Напишете пример** за едно правило!

If Input1 is small and Input2 is small and Input3 is middle and Input4 is high, then Output is class1 (or Iris_Setosa).

Определете стойностите на параметрите на входните променливи от тези данни и ги попълнете в следните таблици.

Таблица B.1: Параметри на входните променливи.

ВХОД 1

Име	**Параметри**
Популация	[40; 80]
Червена	[-20, -10, 48, 59]
Синя	[48, 55, 67, 71]
Зелена	[55, 71, 81, 90]

ВХОД 2

Име	**Параметри**
Вселена	[20; 45]
Червена	[22, 39, 46, 50]
Синя	[0, 10, 24, 35]
Зелена	[21, 28, 34, 39]

ВХОД 3

Име	**Параметри**
Вселена	[10; 70]
Червена	[0, 5, 19, 28]
Синя	[26, 30, 44, 52]
Зелена	[44, 53, 75, 80]

ВХОД 4

Име	**Параметри**
Вселена	[0; 25]
Червена	[-10, -5, 6, 10]
Синя	[6, 10, 13, 19]
Зелена	[13, 19, 30, 35]

Определяне на стойностите на изходните параметри. Попълнете следната таблица с правилните стойности, ако за **изходната лингвистична променлива** използваме **константни функции**.

Таблица B.2: Параметри на изходните променливи.

Изход:

Име	**Параметър**
Вселена	**[1 3]**
Червена	1
Синя	2
Зелена	3

Design the number of rules and write them in the correct form.

Правила:

1. Ако вход1 е червен, а вход2 е червен, а вход3 е червен, а вход4 е червен, то изходът е червен.
2. Ако вход1 е син, вход2 е син, вход3 е син и вход4 е син, тогава изходът е син.
3. Ако вход1 е зелен и вход 2 е зелен и вход 3 е зелен и вход 4 е зелен, тогава изходът е зелен.

B

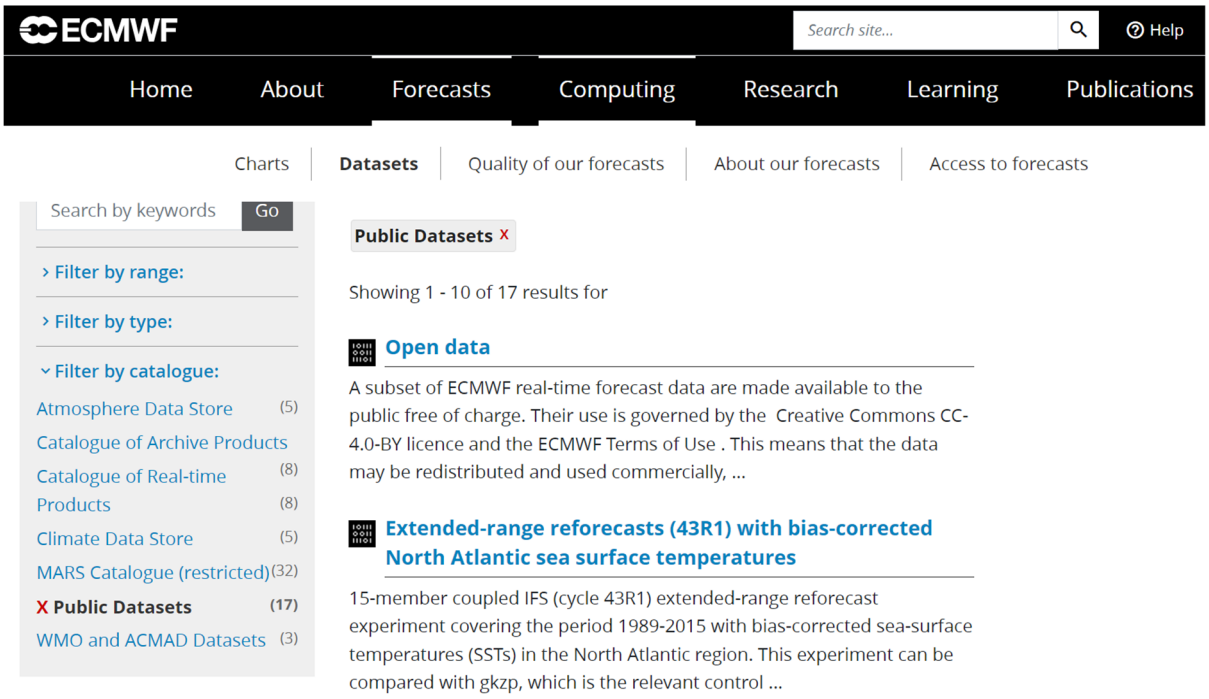
КРАТКО ОПИСАНИЕ НА НАБОРИТЕ ОТ ДАННИ ЗА ИЗМЕНЕНИЕТО НА КЛИМАТА

Тази част от ръководството е написана от Михаела Тинка Удристиу от Департамента по физика, Факултет по науки, и Силвия Пуиу, от Департамената по мениджмънт, маркетинг и бизнес администрация, Факултет по икономика и бизнес администрация, Университет на Крайова, Румъния.

Научните изследвания могат да се развиват по-лесно и по-бързо, когато хората имат достъп до информация с отворен код. При наличието на интернет на една ръка разстояние трябва да знаем къде да търсим точни, актуални и надеждни източници на информация. Ето защо ролята на базите данни е толкова важна. Данните са структурирани и обикновено по начин, който може лесно да се трансформира и обработва според нуждите на изследователя или потребителя.

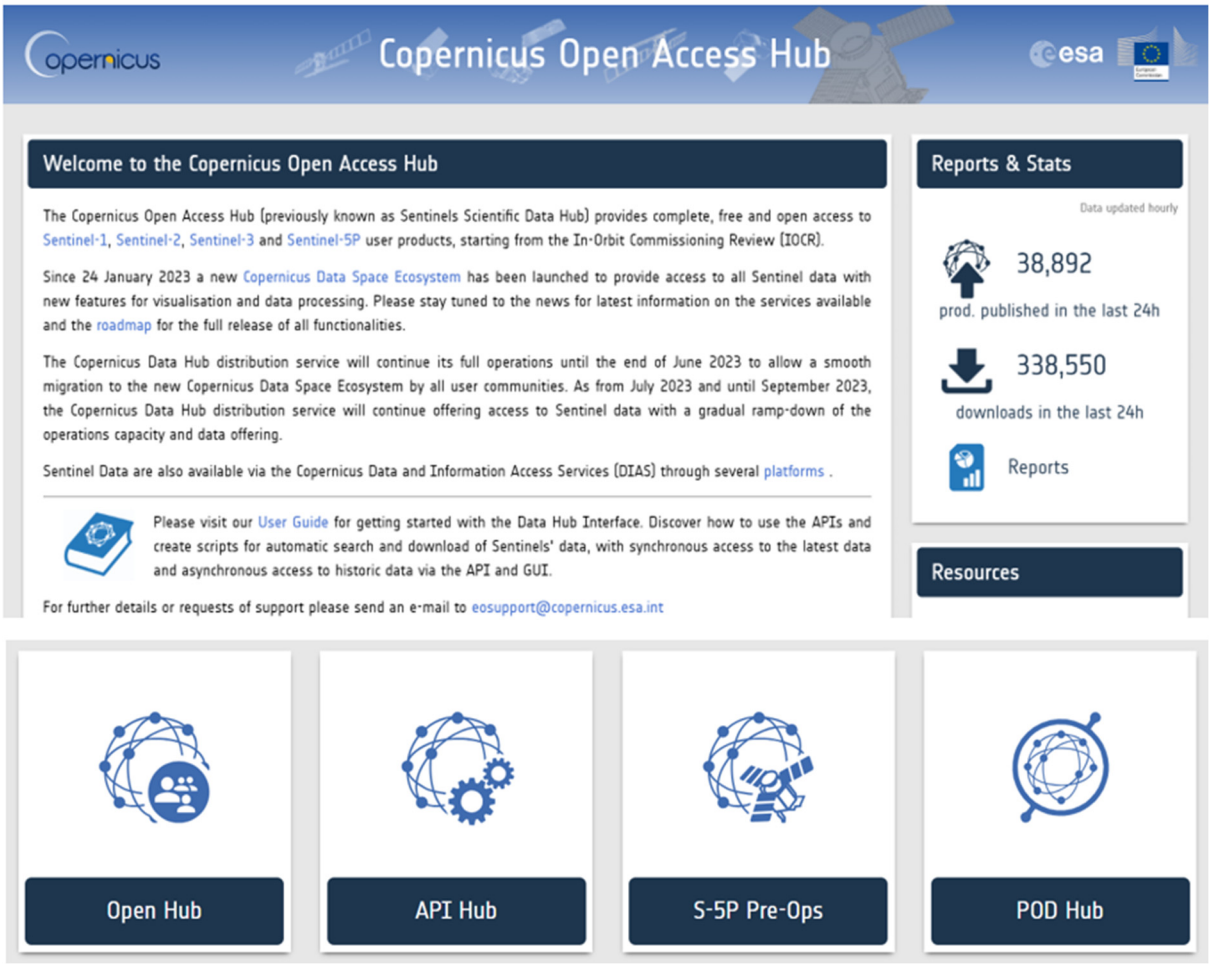
Европейската комисия е създала няколко данни с отворен код за изменението на климата. Лесно е да се изтеглят набори от данни; по-трудно е да се обработват и анализират (проучвателен и прогнозен анализ) набори от данни и да се използват различни алгоритми за създаване на математически модели. Copernicus, European Climate Assessment and Dataset (Европейска оценка на климата и набор от данни за климата), Climate Explorer (Изследовател на климата) и Indecis са само някои от тях. Нуждаем се от данни, за да наблюдаваме изменението на климата и да прогнозираме времето, да наблюдаваме чувствителността на климата към различни параметри, да създаваме различни сценарии и да наблюдаваме развитието на някои процеси в краткосрочен и дългосрочен план. Ще бъде направено кратко представяне на някои бази данни.

***Европейският център за средносрочни метеорологични прогнози (ECMWF)** обработва данни от около 90 спътникови инструмента в ежедневните си оперативни дейности по асимилиране на данни и мониторинг. Около 800 милиона наблюдения дневно и 60 милиона наблюдения с контролирано качество са на разположение ежедневно за използване в Интегрираната система за прогнозиране, като повечето от тях са сателитни измервания. ECMWF също така се възползва от всички налични наблюдения от неспътникови източници, включително наземни и самолетни доклади.*



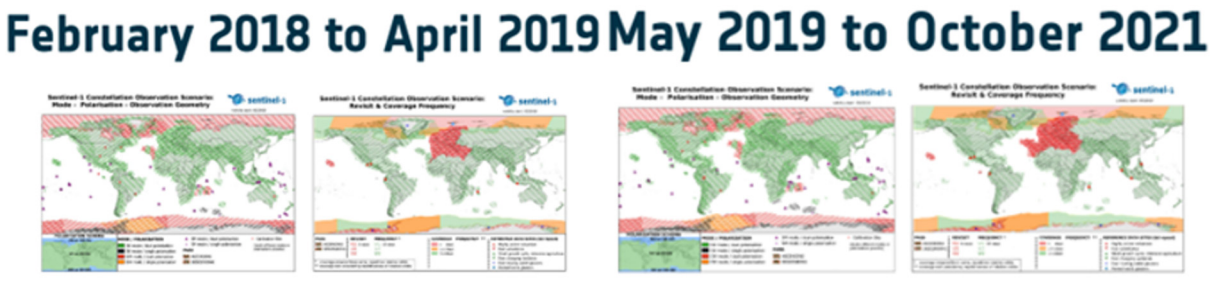
Фигура В.1 – Раздел „Публичен набор от данни“ от ECMWF (източник: https://www.ecmwf.int/en/forecasts/datasets/search)

Коперник (Copernicus) е компонентът за наблюдение на Земята на космическата програма на ЕС, управлявана от Европейската комисия. Тя се осъществява в партньорство с държавите-членки на ЕС, Европейската космическа агенция (ЕКА), Европейската организация за използване на метеорологични спътници (EUMETSAT), Европейския център за средносрочни прогнози за времето (ECMWF), Съвместния изследователски център (JRC), Европейската агенция за околна среда (EAOC), Европейската агенция за морска безопасност (EMSA), Frontex, SatCen и Mercator Océan. Copernicus съдържа набори от климатични данни от различни източници (повторни анализи, сателитни продукти, климатични прогнози). Базата данни на Copernicus е един от най-често използваните набори от данни за изменението на климата, като се работи с модели за прогнозиране и се настройват алгоритми за обработка на данни. Тя разполага със спътници (SENTINELS 1-6) с ясно определени мисии.



Фигура B.2 – Изображение на интерфейса на центъра за отворен достъп Коперник (източник: *https://scihub.copernicus.eu/*)

SENTINEL-1 има два спътника в полярна орбита и работи 24 часа от 24. Той използва радарни изображения със синтетична апертура в С-обхвата, за да събира изображения независимо от времето.



Фигура B.3 – Изображение, предоставено от SENTINEL 1 за два времеви интервала (източник: *https://sentinels.copernicus.eu/web/sentinel/missions/sentinel-1/observation-scenario/archive*)

SENTINEL-2 се състои от два полярно орбитиращи спътника в една и съща слънчевосинхронна орбита, разположени на 180° един от друг. Те следят промените в условията на земната повърхност. Голямата ширина на обхвата (290 км) и дългото време за повторна проверка (10 дни на екватора с един спътник и пет дни с два спътника при условия на безоблачност, което води до 2-3 дни на средни ширини) ще помогнат за наблюдение на промените на земната повърхност. Продуктите на SENTINEL-2 са на разположение на потребителите и са изброени в таблиците с видовете продукти. Някои продукти са предназначени само за експерти (лъчения от горната част на атмосферата в геометрията на сензорите), а други – за всички потребители (отражения от горната част на атмосферата в картографска геометрия и атмосферно коригирани отражения на повърхността в същата геометрия). Пилотните продукти се генерират при поискване. Съществуват две категории: Хармонизирани повърхностни отражения на SENTINEL-2+Landsat-8/9 в картографска геометрия.

SENTINEL-3 извършва измервания на топографията на морската повърхност, температурата на морската и сухоземната повърхност и цвета на океанската и сухоземната повърхност в подкрепа на системите за океански прогнози, мониторинг на околната среда и климата.

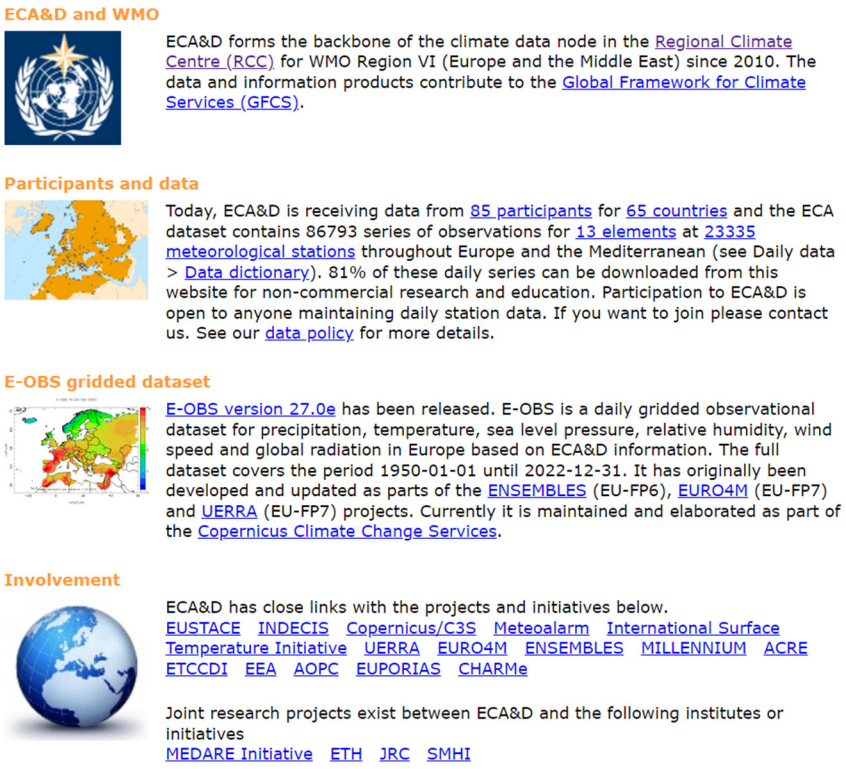
SENTINEL-4 следи ключови газове и аерозоли за качеството на въздуха над Европа, като подпомага Службата за мониторинг на атмосферата на Коперник (CAMS) с бързо време за преразглеждане. Спектрално и радиометрично калибрираната и геолокализирана земна радиация и спектрално и радиометрично калибрираната слънчева радиация са достъпни като параметри за всички потребители, но параметрите за обработка на данните, калибрирането и диагностичните данни за инструмента са достъпни само за експертни потребители.

SENTINEL-5 е спектрометрична система с висока разделителна способност, работеща в ултравиолетовия до късовълновия инфрачервен диапазон със седем различни спектрални ленти: UV-1 (270-300 nm), UV-2 (300-370 nm), VIS (370-500 nm), NIR-1 (685-710 nm), NIR-2 (745-773 nm), SWIR-1 (1590-1675 nm) и SWIR-3 (2305-2385 nm). Инструментът ще бъде разположен на спътника MetOp-SG A. Sentinel-5 е фокусиран върху качеството на въздуха и взаимодействието между състава и климата, като основните продукти от данни са O_3, NO_2, SO_2, HCHO, CHOCHO и аерозоли. Sentinel-5 предоставя качествени параметри за CO, CH_4 и стратосферния O_3 с ежедневно глобално покритие за приложения в областта на климата, качеството на въздуха и озона/повърхностния ултравиолетов спектър.

SENTINEL-5P извършва атмосферни измервания с висока пространствено-времева разделителна способност за качество на въздуха, озон и ултравиолетова радиация, както и за мониторинг и прогнозиране на климата.

Copernicus SENTINEL-6 Майкъл Фрайлих е фокусирана върху повишаването на морското равнище поради изменението на климата и е следващата референтна мисия за радиолокационна алтиметрия, която ще разшири наследството от измервания на височината на морската повърхност поне до 2030 г.

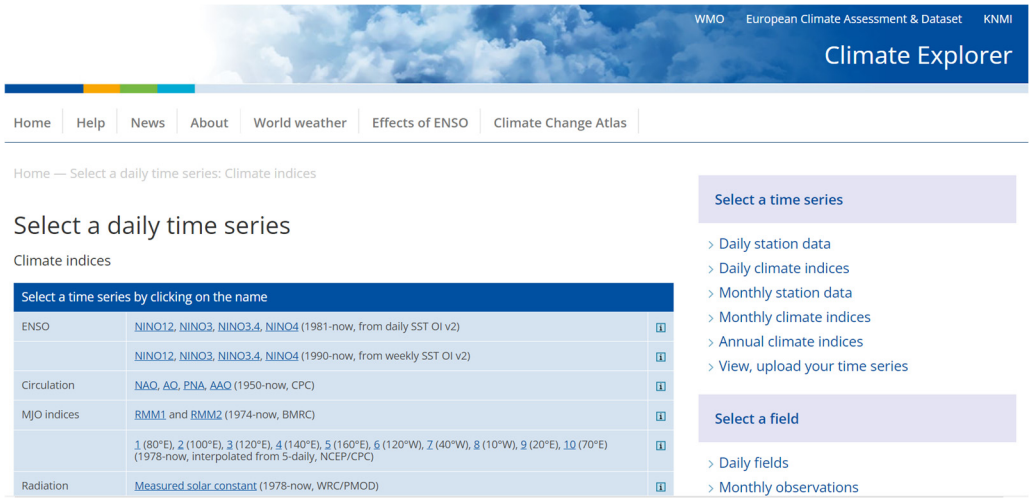
Друга важна база данни е **ECA&D**, която съдържа наблюдения от метеорологични станции и набори от данни, получени от тях на европейско ниво; обикновено те се използват като референтни данни. Този сайт съдържа информация относно промените в екстремните метеорологични и климатични явления и ежедневните набори от данни, необходими за наблюдение и анализ на тези екстремни явления.



Фигура B.4 – Интерфейсът на ECA&D и WMO
(източник: *https://www.ecad.eu*)

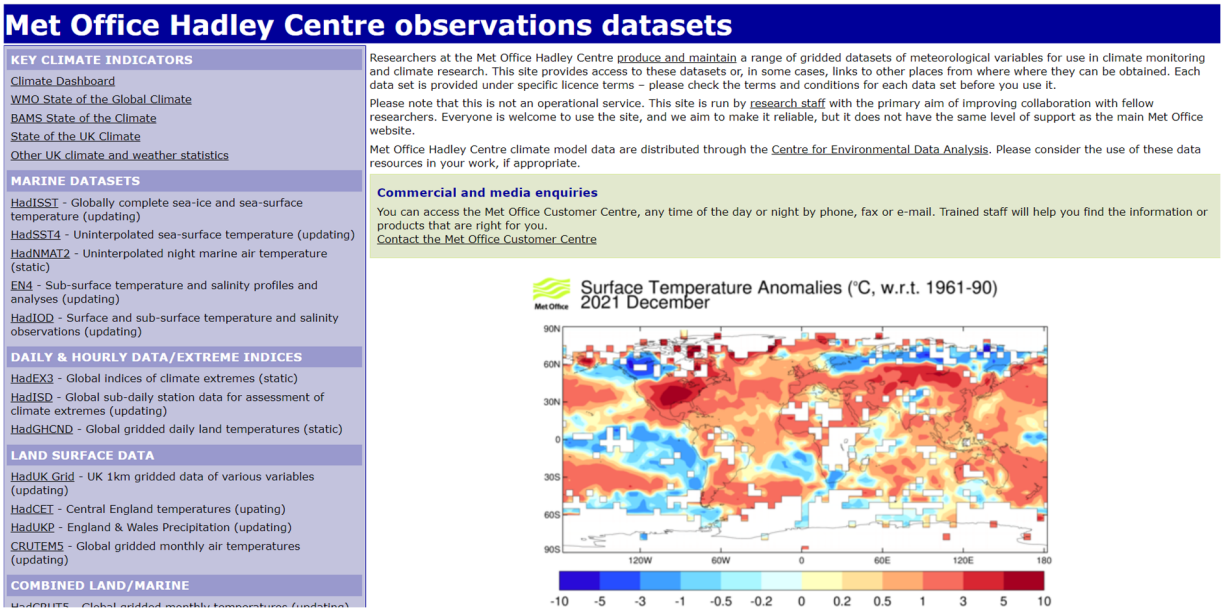
KNMI Climate Explorer е друга база данни, която съдържа клас климатични данни (времеви редове или полеви данни) от повторни анализи и климатични модели, включително климатични прогнози; предимството ѝ е по-приятелският интерфейс (включително графични изображения). По тази причина тя представлява добър образователен инструмент. Потребителите могат да изтеглят времеви серии за дневни и месечни данни от станции и климатични индекси. На годишно ниво са налични само годишни климатични индекси. Информацията може да бъде изтеглена, като се избере поле, като например дневни полета, месечни наблюдения, ме-

сечни полета от повторен анализ, месечни и сезонни исторически реконструкции, месечни сезонни хиндкастове, месечни изпълнения на сценария CMIP3+, месечни изпълнения на сценария CMIP5, годишни екстремуми на CMIP5, месечни изпълнения на сценария CMIP6, месечни изпълнения на сценария CORDEX, атрибутивни изпълнения.



Фигура B.5 – Отпечатване на екрана на KNMI Climate Explorer (източник: *https://climexp.knmi.nl/selectdailyindex.cgi?id=someone@somewhere*)

Центърът Hadley Met Office предоставя набори от данни за метеорологични променливи.



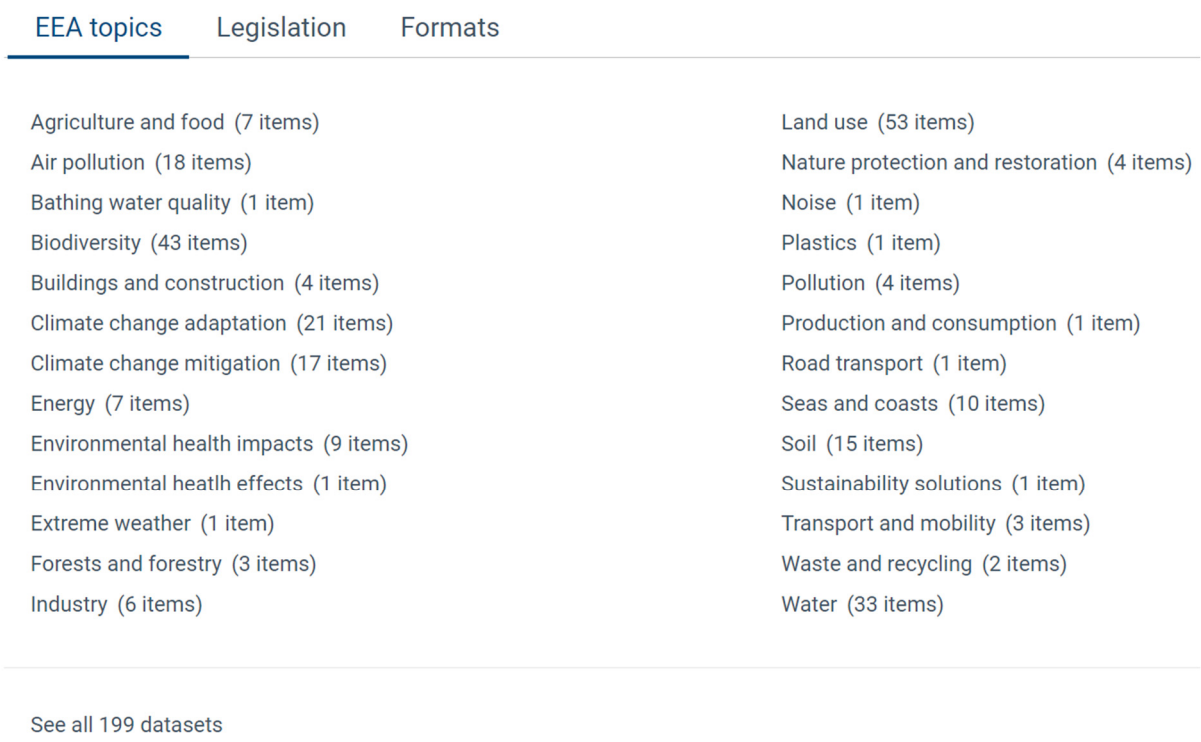
Фигура B.6 – Интерфейсът на центъра Hadley на метеорологичната служба (източник: *https://www.metoffice.gov.uk/hadobs/index.html*)

Тази информация се използва за мониторинг на климата и за изследвания на климата. Класовете са представени от ключови климатични показатели, набори от морски данни, дневни и часови данни/екстремни индекси, данни за земната повърхност, комбинирани данни за земната/морската повърхност, данни за налягането, данни за горните слоеве на въздуха, еднократни данни, придружаващи статии в списания, и постари набори от данни.

Indecis съдържа данни за климата в областта на селското стопанство, намаляването на риска от бедствия, енергетиката, здравеопазването, водите и туризма (http://indecis.eu/indices.php). Тук има само климатични индекси – много, с различни приложения; платформата разполага с дефиниции за климатичните индекси, с графично представяне като карта и поредица от данни в една точка, плюс изтегляне. Това е и добър образователен инструмент. Тя съдържа дневни данни от станции, данни от станции с контролирано качество, хомогенизирани данни от станции, възстановени данни от станции и мрежови версии на индексите.

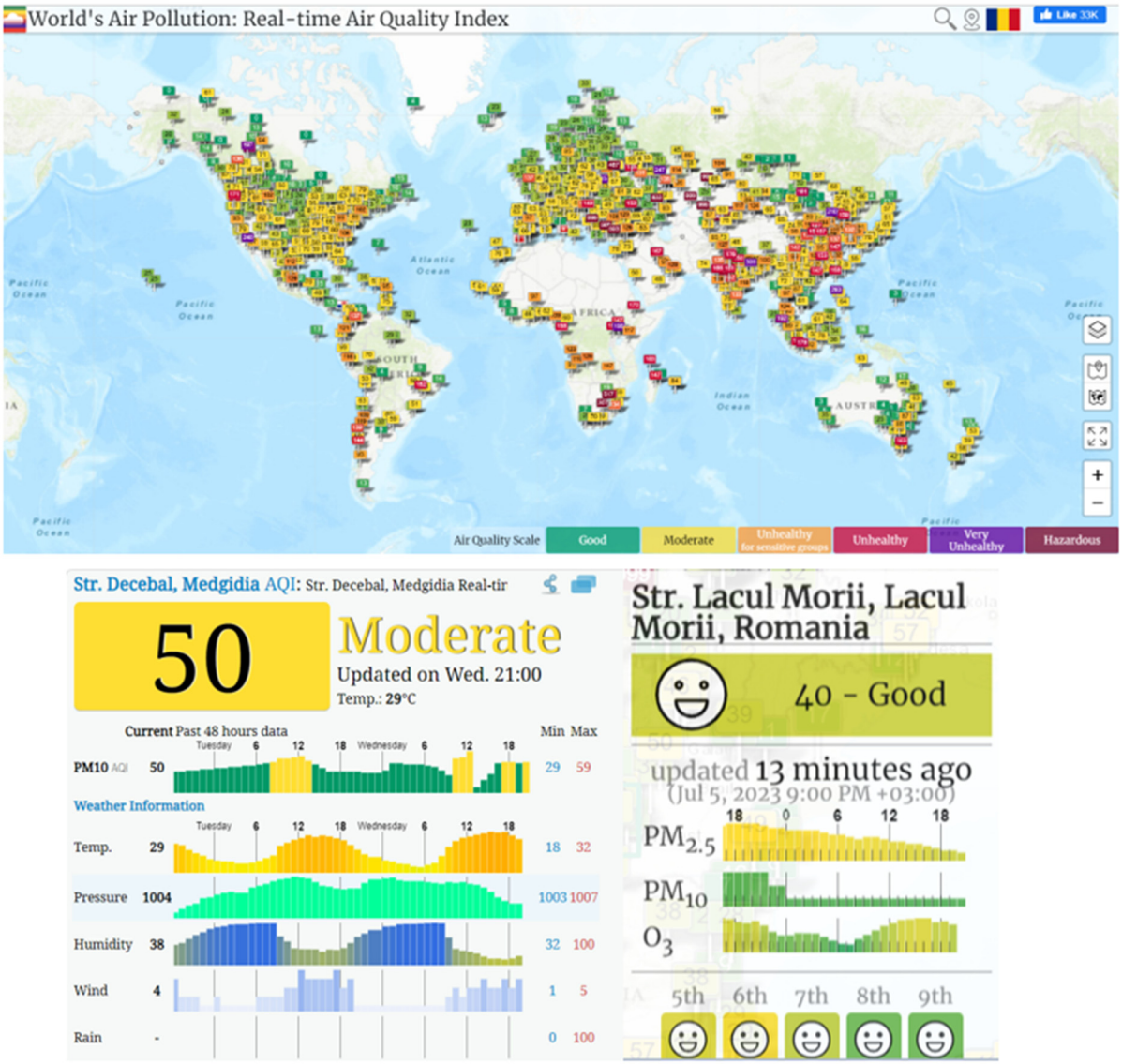
Фигура B.7 – Класове данни, които могат да се изтеглят от Indecis (източник: *https://www.ecad.eu/dailydata/predefinedseries.php*)

Европейската агенция по околна среда (European Environment Agency) разполага с данни с отворен код за качеството на въздуха.



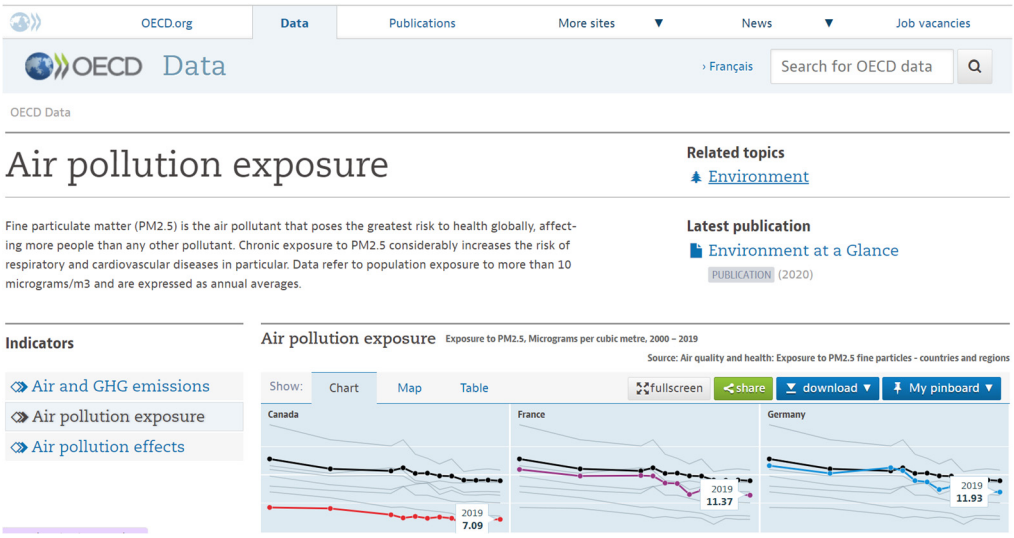
Фигура В.8 – Набори от данни, предоставени от Европейската агенция за околната среда (източник: *https://www.eea.europa.eu/themes/air/explore-air-pollution-data*)

World's Air Pollution (**Замърсяването на въздуха в света**) съдържа сензори от националните агенции за околна среда и дава информация за индекса на качеството на въздуха в реално време.



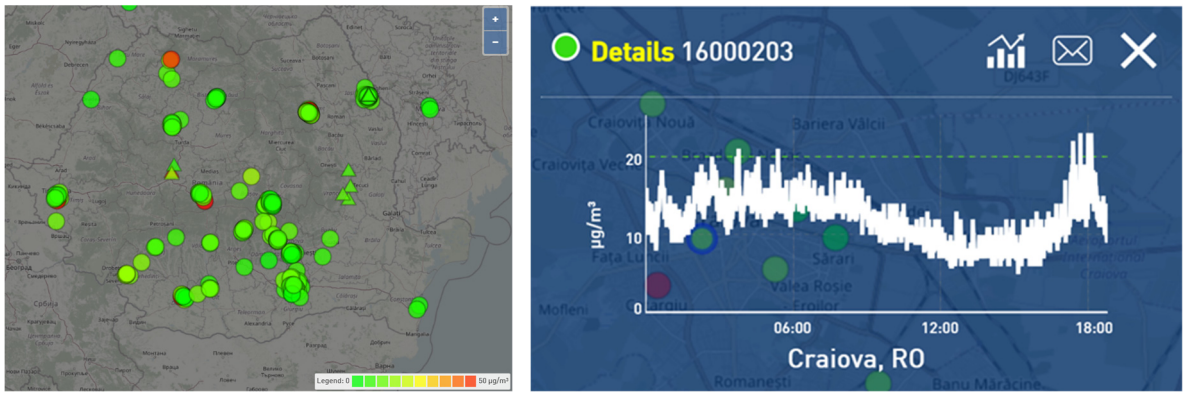
Фигура В.9 – Картата на сензорите за качество на въздуха по света и повече информация при кликване върху сензор (източник: *https://waqi.info/*)

OECD съдържа информация за показатели като емисии във въздуха и GHC, излагане на замърсяване на въздуха и последици от замърсяването на въздуха под формата на диаграми, карти или таблици, за да може всеки лесно да визуализира развитието на данните във времето. С едно кликване ще се отворят данни, организирани в диаграми, карти и таблици.



Фигура B.10 – OECD интерфейс (източник: *https://data.oecd.org/air/air-pollution-exposure.htm*)

Съществуват и граждански научни инициативи и управлявани от общността сензорни мрежи. Тези мрежи съдържат евтини сензори, които следят качеството на въздуха от гражданите в техните общности и имат отлично покритие на голяма част от територията на Европа. Някои от тези мрежи са изградени от доброволци в рамките на някои проекти с образователна цел. uRADMonitor® е такъв пример в Румъния. Мрежата осигурява свободен достъп до данни в реално време. Администраторите могат да предоставят исторически данни при поискване. Инициативите за гражданска наука насърчават прозрачността и отчетността в мониторинга на околната среда. Други примери са следните: Мрежата от обществени сензори за въздух (CAIRSENSE), мрежата Smart Citizen®, Обществената лаборатория за отворени технологии и наука или мрежата Public Lab, инициативата Eye on Earth, Global Learning and Observations to Benefit the Environment (GLOBE), HabitatMap©, проектът за обществен мониторинг на въздуха в окръг Империал и програмата Citizen Weather Observer Program (CWOP).

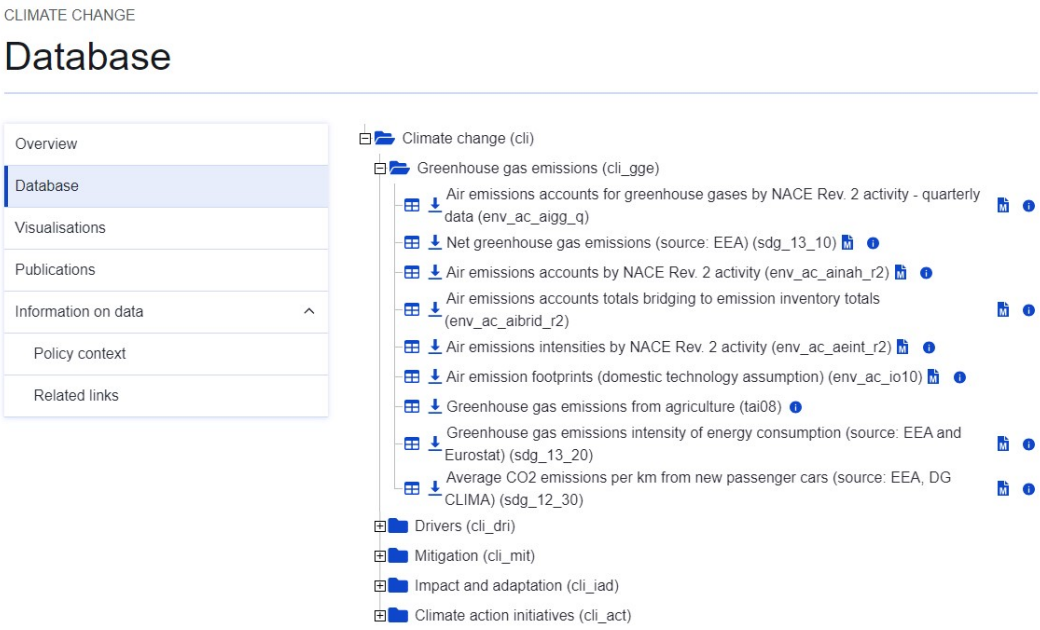


Фигура B.11 Екран на uRADMonitor® мрежата (източник: https://www.uradmonitor.com/)

Статистическите бази данни спомагат за напредъка на изследванията, като предоставят важни данни за множество променливи и по-дълги периоди. Това е от основно значение за изготвяне на заключения, създаване, прогнозиране или смекчаване на сценарии. Например, ако се нуждаем от данни за изменението на климата, като например за нивата на замърсяване, разполагаме с множество бази данни с отворен код, до които можем да получим достъп и да ги използваме за нашите цели. Решенията се основават на входящи данни; изборът ни е добър само ако разполагаме с правилната информация. Затова е важно да избираме надеждни източници на информация, например такива от известни национални и международни организации.

Един от тези източници е базата данни на Евростат, която предоставя статистически данни за много аспекти от интерес за европейските страни. Една от причините, поради които можем да сме сигурни в тази база данни, е нейната дълга история (70 години) и фактът, че тя е под егидата на Европейския съюз.

Нека да дадем пример как можем да получим достъп до данни за изменението на климата, като използваме базата данни на Евростат. Можем да влезем директно в уебсайта и да потърсим данни за изменението на климата или да направим това с помощта на търсачка. Ако отидем на адрес https://ec.europa.eu/eurostat/web/climate-change/database, можем да намерим множество данни за нашата цел, както е показано на фиг. B.12.

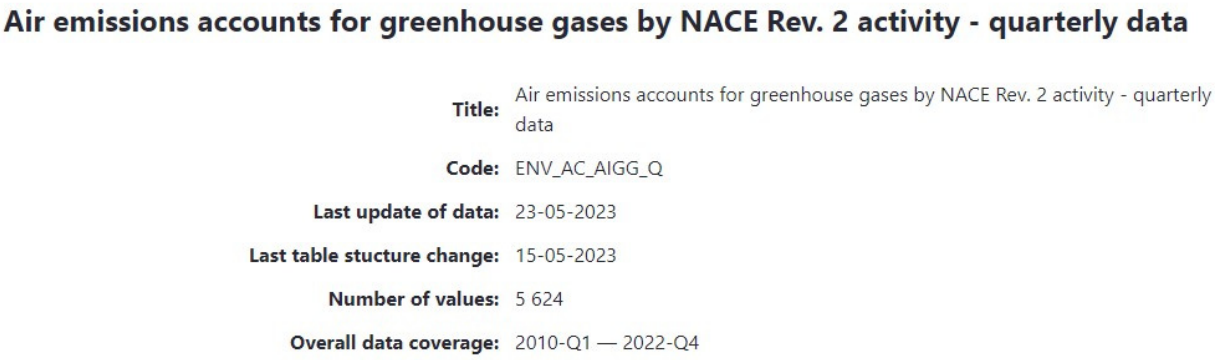


Фигура В.12 – Отпечатване на екран от уебсайта на Евростат относно информацията за изменението на климата
(източник: *https://ec.europa.eu/eurostat/web/climate-change/database*)

Базата данни за изменението на климата е структурирана в много папки: емисии на парникови газове; фактори за изменението на климата; смекчаване на последиците от изменението на климата; въздействие и адаптация; и инициативи за действие в областта на климата. Във всяка от тях има данни, които потребителят може да изтегли. Информацията е безплатна и достъпна за всички в няколко формата.

В папка „Емисии на парникови газове“ има няколко данни. Ако отидем на първата от тях – Отчет за емисиите на парникови газове във въздуха (тримесечни данни), можем да намерим повече информация, ако натиснем десния бутон. Прозорецът, който се показва, е този на Фигура В.12. Така можем да видим, че данните са налични за 13 години, от 2010 г. до 2022 г., и са актуализирани през май 2023 г.

Ако искаме да разберем повече за факторите, които водят до изменение на климата, избираме втората папка и на фигура В.13 виждаме, че има данни за всички важни фактори, като например енергия, транспорт, промишлени процеси, отпадъци, селско стопанство и земеползване, промени в земеползването и горско стопанство. За „Енергия“ можем да изтеглим данни относно крайното потребление на енергия, крайното потребление на енергия на глава от населението, крайното потребление на енергия по сектори и др.



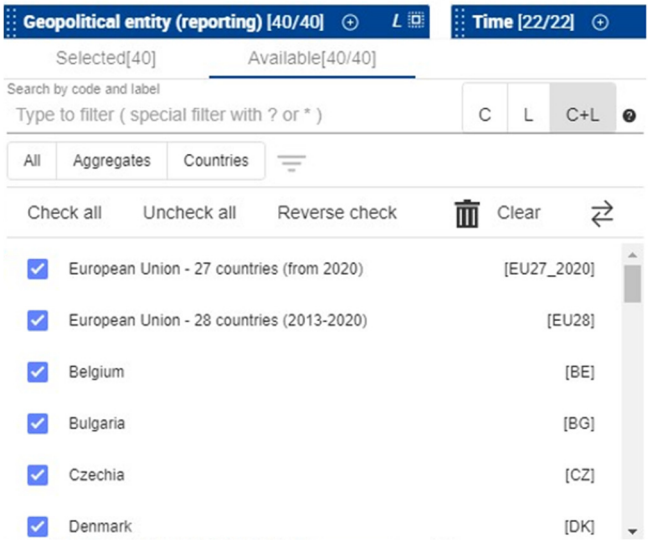
Фигура B.13 – Екран след натискане на бутона за информация (източник: https://ec.europa.eu/eurostat/web/climate-change/database)

Важно е да се изберат данните, от които потребителят се нуждае, за да постигне целите си. Данните са сурови и необработени, така че потребителят може да използва няколко инструмента, за да обработи данните и да забележи тенденция, да предвиди някои сценарии и да предостави резултатите, до които е стигнал. Предприятията, физическите лица, правителствата и други отговорни лица ще използват тези резултати, за да предотвратят или подобрят някои аспекти.

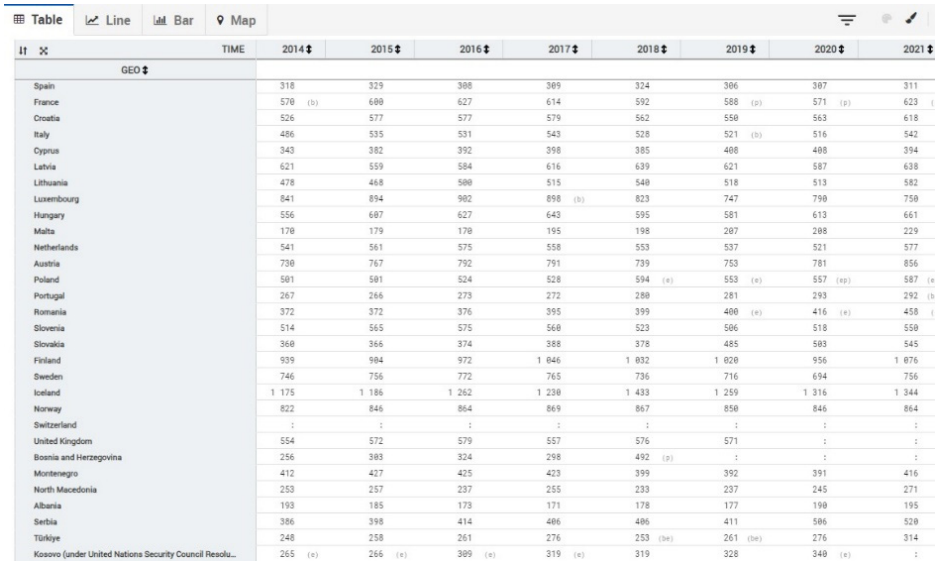
Фигура B.14 – Екран за разпечатване на информация от Евростат относно причините за изменението на климата (източник: https://ec.europa.eu/eurostat/web/climate-change/database)

Сега нека видим как изглежда информацията, ако искаме да проверим крайното потребление на енергия в домакинствата на глава от населението. На фигура B.11 е показан код в скоби в близост до този показател: SDG 7. Всъщност това е препратка към седмата цел за устойчиво развитие от Програмата до 2030 г. на ООН. Тя се отнася до „Достъпна и чиста енергия".

Ако кликнем върху първата икона, която прилича на таблица, можем да прочетем обяснения за индикатора, но също така можем да изберем формата на данните (таблица, линия, стълб, карта) и променливите, от които се нуждаем (държави и години) – Фиг. B.14 и B.15.

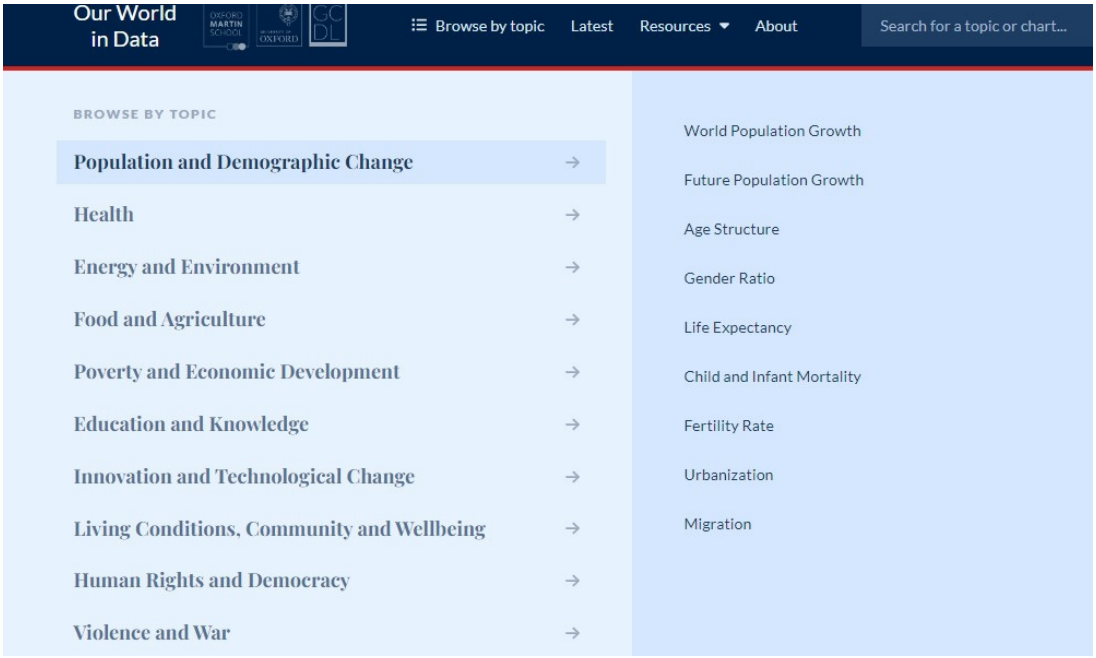


Фигура B.15 – Екран за филтрите, които могат да се използват за данните от Евростат (източник: *https://ec.europa.eu/eurostat/web/climate-change/database*)



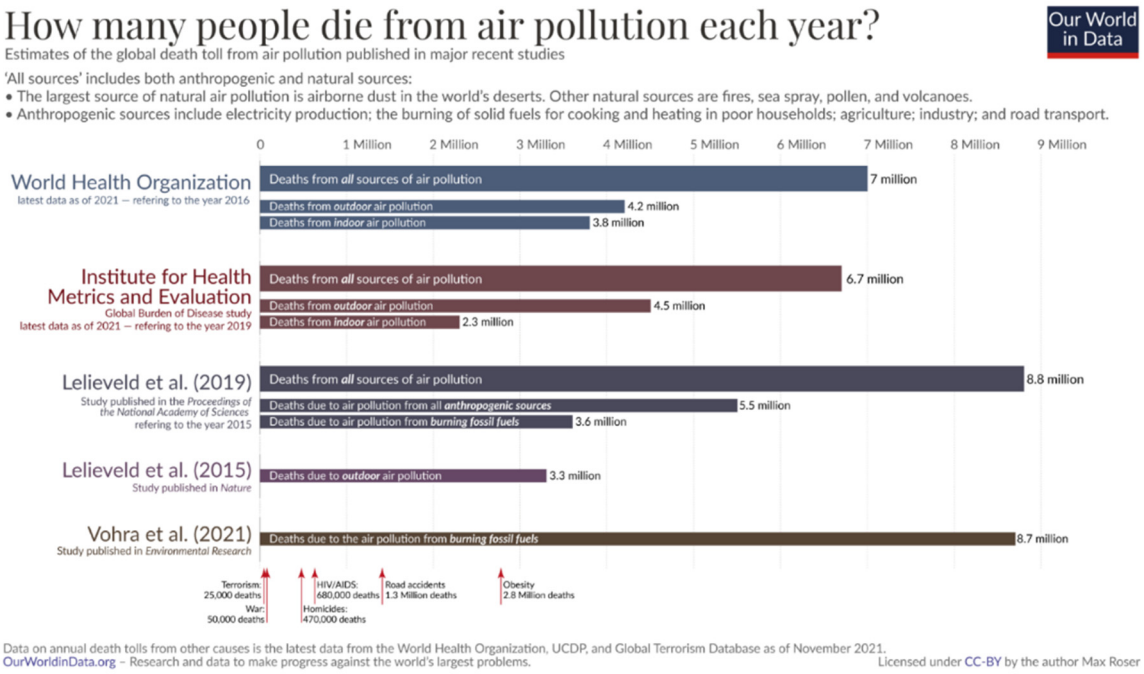
Фигура B.16 – Екран на показваната информация, ако изберем формата на таблицата (източник: *https://ec.europa.eu/eurostat/web/climate-change/database*)

Освен Евростат изследователите могат да използват и други бази данни с отворен код. Можем да споменем Our World in Data, чиято основна цел, посочена на техния уебсайт (https://ourworldindata.org/), е: те публикуват „изследвания и данни, за да постигнат напредък в борбата с най-големите световни проблеми, които се отнасят до динамиката на населението, енергията и околната среда, здравето, храната, бедността, образованието, условията на живот, правата на човека, технологичните промени, насилието и войната. Организацията е с нестопанска цел, но е много цитирана в литературния обзор и в медиите.

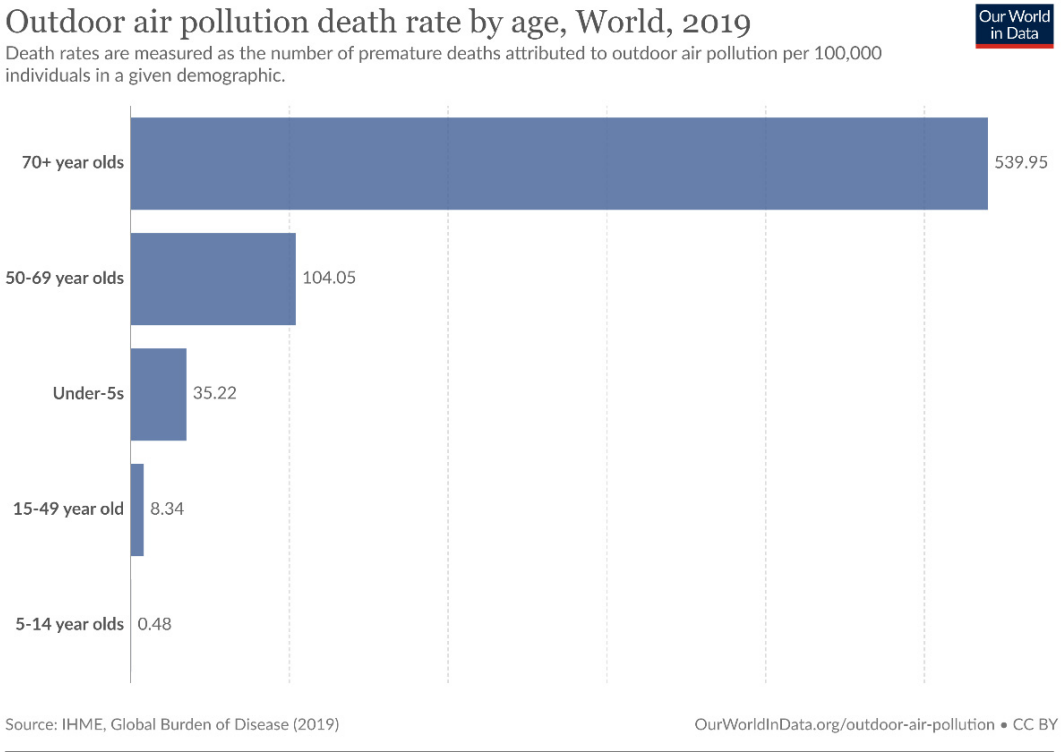


Фигура B.17 – Екран от Our World in Data във връзка с темите, разгледани в тази публикация (източник: https://ourworldindata.org/)

Ако се интересуваме от замърсяването на въздуха, избираме Енергия и околна среда и можем да изберем замърсяване на въздуха на открито или на закрито. Този уебсайт предлага статии и необработени статистически данни за вашите изследвания или дейности. Така можете да откриете колко смъртни случая в световен мащаб могат да се припишат на замърсяването на въздуха (Фиг. B.17). Можете също така да откриете степента на замърсяване на въздуха на открито по възраст и да изтеглите данните като таблица или диаграма (Фиг. B.18).



Фигура B.18 – Смъртност от замърсяване на въздуха в световен мащаб (източник: https://ourworldindata.org/data-review-air-pollution-deaths)



Фигура B.19 – Смъртност от замърсяване на въздуха по възраст (източник: https://ourworldindata.org/outdoor-air-pollution#outdoor-air-pollution-deaths-by-age)

Г

ВЪЗДЕЙСТВИЕ НА ЗАМЪРСЯВАНЕТО НА ВЪЗДУХА ВЪРХУ ЧОВЕШКОТО ЗДРАВЕ

Тази част от наръчника е написана от Славея Петрова от катедра „Екология и опазване на околната среда“, Биологически факултет, Пловдивски университет „Паисий Хилендарски“, България.

Замърсяването на въздуха е замърсяването на вътрешната или външната среда с химически, физични или биологични агенти, които променят естествените характеристики на атмосферата.

Горивните устройства в домакинствата, моторните превозни средства и промишлените предприятия са често срещани източници на замърсяване на въздуха. Замърсителите, които предизвикват сериозна загриженост за общественото здраве, включват прахови частици (ПЧ), въглероден оксид (CO), озон (O_3), азотен диоксид (NO_2) и серен диоксид (SO_2).

Според Европейската агенция по околна среда (ЕАОС) всеки един от замърсителите на въздуха може да бъде свързан с различен източник:

- Потреблението на енергия в жилищни, търговски и институционални сгради е основният източник на прахови частици през 2020 г. Преработващата и добивната промишленост и селското стопанство също са значителни източници на фини ПЧ10. Между 2005 г. и 2020 г. се наблюдава низходяща тенденция по отношение на емисиите на прахови частици (ПЧ10 и ПЧ2.5) – те са намалели съответно с 30 % и 32 %.
- През 2020 г. селското стопанство е основният източник на амоняк (94% от общите емисии) и метан (56%). Емисиите на амоняк са намалели само с 8% от 2005 г. до 2020 г. Това е най-ниското процентно намаление от всички замърсители.
- Пътният транспорт е основният източник на азотни оксиди през 2020 г. – 37% от емисиите. Между 2005 г. и 2020 г. е установено значително намаляване на емисиите на азотни оксиди с до 48 %.
- Секторът на енергоснабдяването е основният източник на серен диоксид, отговорен за 41% от емисиите през 2020 г. Емисиите на серен диоксид са намалели със 79% между 2005 и 2020 г.

- Основните източници на емисии на тежки метали през 2020 г. са производствените и добивните отрасли и секторът на енергоснабдяването. Между 2005 г. и 2020 г. най-голямо намаление на емисиите е установено за никел (64 %) и арсен (62 %).

Г.1. Видове замърсители и рискове за здравето

Прахови частици (ПЧ)

Финитe прахови частици (ФПЧ) са често срещан косвен показател за замърсяването на въздуха. Основните компоненти на фракциите на ФПЧ са сулфати, нитрати, амоняк, натриев хлорид, черен въглерод, минерален прах и вода.

Рисковете за здравето, свързани с праховите частици с диаметър по-малък от 10 и 2.5 микрона (ПЧ10 и ПЧ2.5), са особено добре документирани. ПЧ могат да проникнат дълбоко в белите дробове и да попаднат в кръвообращението, причинявайки сърдечносъдови (исхемична болест на сърцето), мозъчносъдови (инсулт) и респираторни въздействия. Както дългосрочното, така и краткосрочното излагане на ПЧ е свързано със заболеваемост и смъртност от сърдечносъдови и респираторни заболявания. Дългосрочната експозиция е свързана с неблагоприятни перинатални резултати и рак на белия дроб

Въглероден оксид (CO)

Въглеродният оксид е безцветен, без мирис и без вкус токсичен газ, който се получава при непълното изгаряне на въглеродни горива като дърва, бензин, дървени въглища, природен газ и керосин. Въглеродният оксид дифундира през тъканите на белите дробове и навлиза в кръвния поток, като затруднява свързването на кислорода с клетките на организма. Тази липса на кислород уврежда тъканите и клетките. Излагането на въглероден оксид може да причини затруднено дишане, изтощение, замаяност и други грипоподобни симптоми. Излагането на високи нива на въглероден оксид може да бъде смъртоносно

Озон (O_3)

Озонът на нивото на земята е една от основните съставки на фотохимичния смог. Той се появява в резултат на реакция с газове при наличие на слънчева светлина. Струва си да се спомене, че озонът може да се генерира от домакински уреди, като например преносими почистващи машини за въздух. Излагането на прекомерно количество озон може да причини проблеми, като например затруднено дишане, да предизвика астма, да намали функцията на белите дробове и да доведе до белодробни заболявания.

Азотен диоксид (NO_2)

NO_2 е газ, който обикновено се отделя при изгарянето на горива в транспортния и промишления сектор. Домакинските източници на азотни оксиди (NOx) включват оборудване, което изгаря горива, като пещи, камини, газови печки и фурни. Излагането на азотен диоксид може да раздразни дихателните пътища и да влоши респираторните заболявания.

Серен диоксид (SO_2)

SO_2 е безцветен газ с остра миризма, който се получава при изгарянето на изкопаеми горива (въглища и нефт) и при топенето на минерални руди, съдържащи сяра. Излагането на SO_2 се свързва с хоспитализации за астма и посещения в спешното отделение.

Полицикличи ароматни въглеводороди (ПАВ)

Полицикличните ароматни въглеводороди (ПАВ) се съдържат в атмосферата под формата на частици. Те са група химични вещества, образувани предимно от непълното изгаряне на органични вещества (напр. готвене на месо) и изкопаеми горива в коксови пещи, дизелови двигатели и печки за изгаряне на дърва. Те могат да присъстват и в тютюневия дим. Краткотрайното излагане може да раздразни очите и дихателните пътища. Дългосрочното излагане на ПАВ е свързано с рак на белия дроб.

Г.2 Глобални насоки на СЗО за качеството на въздуха

От 1987 г. насам СЗО периодично издава здравни насоки за качеството на въздуха, за да помогне на правителствата и гражданското общество да намалят излагането на хората на замърсяване на въздуха и неговите неблагоприятни последици. Основната цел е да се предложат количествени здравни препоръки за управление на качеството на въздуха, изразени като дългосрочни или краткосрочни концентрации за няколко основни замърсители на въздуха. Превишаването на нивата на препоръчителните норми за качество на въздуха (ПНКВ) е свързано със сериозни рискове за общественото здраве. Тези насоки не са правно обвързващи стандарти и нямат задължителен характер. Въпреки това те предоставят на държавите – членки на СЗО, инструмент, основан на доказателства, който те могат да приложат в националните програми за намаляване на нивата на замърсителите на въздуха, за да се намали огромната тежест за здравето от излагането на замърсяване на въздуха в световен мащаб.

Таблица Г.1 – Препоръчителни насоки за качеството на въздуха за всеки замърсител [5]

Замърсител	Ориентировъчна стойност	Средно време	Препратка към насоките
$ПЧ_{2.5}$	5 μg/m³	годишна	СЗО 2021
	15 μg/m³	24-часова	
$ПЧ_{10}$	15 μg/m³	годишна	СЗО 2021
	45 μg/m³	24-часова	
Въглероден оксид (CO)	4 μg/m³	24-часова	СЗО 2021
Азотен диоксид (NO_2)	10 μg/m³	годишна	СЗО 2021
	25 μg/m³	24-часова	
Серен диоксид (SO_2)	40 μg/m³	24-часова	СЗО 2021
Формалдехид	0.1 μg/m³	30-минутна	СЗО 2010
Полициклични ароматни въглеводороди	8.7×10^{-5} per ng/m³		СЗО 2010
Радон	100 Bq/m³		СЗО 2010
Олово	0.5 μg/m³	годишна	СЗО Регионален офис за Европа, 2000

Г.3. Проучвания на въздействието на замърсяването на атмосферния въздух върху здравето

През последното столетие засиленото изгаряне на изкопаеми горива и непрекъснатото интензифициране на трафика са причина за прогресивното изменение на състава на атмосферата, което се отразява отрицателно на качеството на живот.

Един от основните аспекти е влиянието на отработените газове от превозните средства върху здравето, към което децата са особено чувствителни. Отработените газове съдържат над 200 вида замърсители, някои от които са: CO_2, NO_x, CO, SO_x, въглеводороди с ниско молекулно тегло, алдехиди (формалдехид, ацеталдехид, акролеин), бензен, 1,3 бутадиен, полициклични въглеводороди, частици с окислителна компонента (еле-

ментарен въглерод, адсорбирани ароматни въглеводороди, малки количества сулфати, нитрати, метали и други елементи) и др.

Въпреки че намалената икономическа активност по време на рецесията доведе до намаляване на емисиите в атмосферата, като цяло се счита, че автомобилният транспорт в Европа е отговорен за вредните нива на замърсители на въздуха и за една четвърт от емисиите на парникови газове в Европейския съюз. Стандартите „Евро" за превозните средства постигнаха известен успех, но не доведоха до значително намаляване на NO_2.

Замърсителите на въздуха, като въглероден оксид (CO), серен диоксид (SO_2), азотни оксиди (NO_x), летливи органични съединения (ЛОС), озон (O_3), тежки метали и прахови частици (ПЧ2.5 и ПЧ10), се различават по своя химичен състав, реакционни свойства, време на разпадане и способност да се разпространяват на големи или малки разстояния. Замърсяването на въздуха на открито е основен здравен проблем на околната среда, който засяга всички в страните с ниски, средни и високи доходи, тъй като може да причини респираторни и други заболявания и е важен източник на заболеваемост и смъртност [9]. Тези ефекти на замърсителите на въздуха върху човешкото здраве и механизмът им на действие ще бъдат разгледани накратко по-долу.

Замърсяването на въздуха има остри и хронични ефекти върху човешкото здраве, като засяга няколко различни системи и органи. То варира от леко дразнене на горните дихателни пътища до хронични респираторни и сърдечни заболявания, рак на белия дроб, остри респираторни инфекции при децата и хроничен бронхит при възрастните, влошаване на вече съществуващи сърдечни и белодробни заболявания или астматични пристъпи. Освен това краткосрочните и дългосрочните експозиции са свързани и с преждевременна смъртност и намалена продължителност на живота.

СЗО изчислява, че през 2019 г. около 37% от преждевременните смъртни случаи, свързани със замърсяването на въздуха на открито, се дължат на исхемична болест на сърцето и инсулт, 18% и 23% от смъртните случаи се дължат съответно на хронична обструктивна белодробна болест и остри инфекции на долните дихателни пътища, а 11% от смъртните случаи се дължат на рак на дихателните пътища. Замърсяването на атмосферния въздух (на открито) в градовете и селските райони се оценява като причина за 4.2 милиона преждевременни смъртни случая в света годишно през 2019 г. Тази смъртност се дължи на излагането на фини прахови частици, които причиняват сърдечносъдови и респираторни заболявания и рак.

В целия ЕС често се срещат нива на замърсяване на въздуха, които са по-високи от последните препоръки на СЗО. Все пак има признаци на подобрение, но по-долу са посочени някои факти:

- През 2021 г. 97% от градското население е било изложено на концентрации на фини прахови частици, надвишаващи препоръчителното ниво, определено от Световната здравна организация.
- Всяка година се смята, че замърсяването на въздуха в страните членки на ЕАОС и в страните, които си сътрудничат с нея, причинява смъртта на над 1200 души на възраст под 18 години [10].
- Данните от 2021 г. показват, че в Централна и Източна Европа и Италия са отчетени най-високите концентрации на прахови частици, което се дължи главно на изгарянето на твърди горива за битово отопление и използването им в промишлеността.
- Всички страни от ЕС докладваха за нива на озон и азотен диоксид, надвишаващи препоръчителните нива, определени от Световната здравна организация.
- Всяка година около 275 000 случая на преждевременна смърт се дължат на фини прахови частици, а 64 000 – на азотен диоксид (NO_2).
- Като цяло 97% от градското население на ЕС е било изложено на нива на фини прахови частици над последните насоки, определени от СЗО през 2021 г.

Неблагоприятните последици от излагането на замърсяване на въздуха са глобален проблем за общественото здраве както в развиващите се, така и в развитите страни, тъй като децата и младите хора са особено уязвими към последиците от замърсяването на въздуха.

Епидемиологичните проучвания са най-показателни за оценка на въздействието на замърсяването на въздуха върху здравето. Едни от най-уязвимите, подходящи за изследване контингенти, са децата в предучилищна и ранна училищна възраст, тъй като те прекарват повече време на открито, имат по-висока интензивност на метаболитните процеси и аспирират относително по-голям обем въздух от възрастните. Същевременно те все още не са придобили вредни навици (тютюнопушене, консумация на алкохол и др.) и не са изложени на промишлени рискове. В големия комплекс от негативни здравни ефекти на отработените газове най-ясно се открояват нарушенията във функциите на дихателната, сърдечно-съдовата и имунната система, кръвотворната и др.

Голямо проучване, обхванало деца в предучилищна и ранна училищна възраст в 6 града в Северен Китай, показва силна положителна корелация между респираторните симптоми (кашлица, затруднено дишане, хрипове и храчки) и нивата на общ суспендиран прах, серен диоксид и азот.

Особено внимание се обръща на връзката между азотния диоксид и озона и провокирането или изостряне на респираторни заболявания с обструктивен синдром, на първо място астма. Красноречив пример са случаите, при които дори временно намалената интензивност на автомобилния трафик намалява респираторните симптоми на горните дихателни пътища.

Замърсяването на въздуха оказва влияние върху физическото и умственото развитие в детска възраст и изостря респираторни заболявания като астма и сезонен алергичен ринит (SAR), по-често наричан сенна хрема. Сенната хрема е най-често срещаното хронично заболяване при децата и е най-разпространено сред учениците. Съществуват все повече доказателства, че замърсителите на въздуха, като например озонът (O3), могат да засилят алергенността на полените, което от своя страна може да повлияе на когнитивното развитие.

Г.4. Въздействие на замърсяването на въздуха върху здравето и околната среда

Качеството на въздуха е основен проблем за европейците и е област, в която ЕС е особено активен вече повече от 30 години. Основната цел на ЕС по отношение на качеството на въздуха е „да се постигнат нива на качество на въздуха, които не водят до неприемливи въздействия и рискове за човешкото здраве и околната среда“. Въпросите в годишното проучване Flash Eurobarometer са предназначени да подпомогнат тази дейност, като предоставят по-добра представа за възгледите на европейската общественост относно качеството на въздуха и замърсяването на въздуха.

Проучването на Евробарометър има за цел да проучи:

- нивото на познания за проблемите с качеството на въздуха;
- възприеманата сериозност на проблемите с качеството на въздуха и възприеманите промени в качеството на въздуха през последните десет години;
- възприеманото въздействие на различните сектори и дейности върху качеството на въздуха;
- основните заплахи за качеството на въздуха;
- екологосъобразни енергийни и транспортни възможности;
- индивидуални и други действия за намаляване на проблемите с качеството на въздуха;
- и много други.

Резултатите от проучването през 2022 година показват, че качеството на въздуха все още е сериозна грижа за европейските граждани. Всички необработени данни от проучването са свободно достъпни и могат да се ползват онлайн.

- Въпреки че повечето европейци не се чувстват добре информирани (60%), почти половината от анкетираните смятат, че качеството на въздуха се е влошило през последните десет години (47%).
- Повечето европейци смятат, че здравословни състояния като респираторни заболявания (89 %), астма (88 %) и сърдечносъдови заболявания са сериозни проблеми в техните страни, произтичащи от замърсяването на въздуха. Проучването на Евробарометър разкрива, че гражданите нямат достатъчно информация за проблемите с качеството на въздуха в тяхната страна.
- Повечето европейци все още са слабо информирани за съществуващите стандарти на ЕС за качество на въздуха, тъй като само малка част от анкетираните (27 %) са чували за тях.
- Въпреки това голямото мнозинство от анкетираните (67 %), които са запознати със стандартите на ЕС за качество на въздуха, твърдят, че те трябва да бъдат засилени.

Скринингов въпросник за оценка на възприемането на замърсяването на въздуха и на риска от излагане на замърсяване на въздуха на открито и на закрито

За разработването на въпросника беше използван набор от елементи, базирани на много стандартизирани препоръки за проучвания, в допълнение към механизми от подобни проучвания за защита от замърсяване на въздуха. Елементите бяха внимателно написани, за да се сведе до минимум двусмислието и да се повиши разбираемостта. Общият набор от елементи се състоеше от 25 елемента. Въпросникът е обещаващ инструмент за оценка на нагласите и възприятията на населението към замърсяването на въздуха и риска от излагане на замърсяване на открито и закрито. Този въпросник може да се използва от учени, изследователи, власти и планиращи промоцията на здраве за разработване и прилагане на програми за промоция на защита от замърсяване на въздуха.

Въпросник А – основни въпроси

Моля, прочетете всички въпроси и отговорете, като поставите отметка в квадратчето или дадете кратко обяснение, когато е уместно.

Проучването е анонимно и ви уверяваме, че поверителността на вашите индивидуални отговори ще бъде запазена.

1. Пол

□ male □ female

2. Възраст

□ под 3 години □ 3 – 7 години □ 8 – 14 години
□ 15 – 20 години □ 21 – 30 години □ 31 – 40 години
□ 41 – 50 години □ 51 – 60 години □ над 60 години

3. В кой регион живеете?

Страна ..... Селище.....

4. **Бихте ли казали, че живеете в …"**

□ селски район □ село □ малък град
□ средно голям град □ голям град/община

5. Трудова заетост

□ ученик □ студент □ самостоятелно заети лица
□ служител □ неквалифициран работник
□ без професионална дейност □ отказ на отговор
□ други …….. *Моля, посочете.*

6. Колко души на възраст 15 години или повече живеят във вашето домакинство, включително и вие?

□ 1 □ 2 □ 3 □ 4 □ 5 □ 6 □ друго ..... *Моля, посочете.*

7. Какъв е средният месечен доход на семейството ви (на член)?

□ до 300 евро □ 300 – 600 евро □ 600 – 1000 евро
□ 1000 – 1500 евро □ повече от 1500 евро □ друго ……..

Моля, посочете.

8. С какъв тип битово отопление разполагате?

□ електрически отоплител □ газов отоплител □ климатик □ печка
□ отопление на дърва/пелети □ соларно отопление
□ друго ..... *Моля, посочете.*

9. Доколко сте информирани за проблемите с качеството на въздуха във вашата страна?

□ много добре съм информиран □ добре съм информиран
□ не съм добре информиран □ въобще не съм информиран
□ други ..... *Моля, посочете.*

10. Смятате ли, че през последните 10 години качеството на въздуха във вашата страна е ...?

□ подобрено □ останало същото □ влошено
□ друго ..... *Моля, посочете.*

11. Колко голямо е според вас влиянието на всеки от следните фактори върху качеството на въздуха във вашата страна?

Има ли то голямо въздействие, умерено въздействие, слабо въздействие или няма никакво въздействие?

	голямо въздействие	умерено въздействие	слабо въздействие	никакво въздействие
Използване на енергия от домакинствата (напр. въглища и дърва за отопление на отделните домакинства)				
Селско стопанство – емисии от ферми, торове и изгаряне на селскостопански отпадъци				
Емисии от леки и товарни автомобили				
Емисии от международния транспорт (напр. кораби и самолети)				
Емисии от промишленото производство (стомана, цимент, целулоза, хартия и т.н.) и от електроцентралите, използващи изкопаеми горива				
Ландшафт				
Реки/езера				
Чист въздух				
Друго				

12. Кои три от следните замърсители според вас са основните заплахи за качеството на въздуха във вашата страна?

□ трансгранични емисии от други държави/региони
□ транспортни дейности
□ производство на електроенергия и топлинна енергия
□ природни замърсители (морска сол, пустинен пясък, вулканична пепел).
□ промишлени дейности
□ емисии от отделните домакинства
□ емисии от земеделски стопанства
□ друго ..... *Моля, посочете.*

13. Кои две от следните горивни системи за автомобили смятате за най-екологични от гледна точка на качеството на въздуха?

□ бензин □ дизел □ биогориво
□ хибридни електрически/бензинови автомобили
□ хибридни електрически/дизелови автомобили
□ електрически автомобили
□ друго ..... *Моля, посочете.*

14. Кои две от следните енергийни системи за битово отопление смятате за най-екологични от гледна точка на качеството на въздуха?

□ масло □ газ □ въглища
□ биомаса (дървесина) □ биомаса (пелети)
□ електричество □ централно отопление
□ друго..... *Моля, посочете.*

15. Съществуват различни начини за намаляване на вредните емисии във въздуха. За да намалите тези проблеми, правили ли сте някое от следните неща през последните две години? Моля, изберете всички приложими.

□ Сменили сте отоплителната система на жилището си от такава с по-високи емисии (напр. въглища, нафта или дърва) на такава с по-ниски емисии (напр. природен газ, пелети, електричество)

□ Заменили сте по-старо енергоемко оборудване (бойлер за топла вода, фурна, съдомиялна машина и т.н.) с по-ново, с по-добра енергийна ефективност (напр. A+++ за енергийна ефективност).

□ Често сте използвали обществен транспорт, колоездене или ходене пеша вместо автомобил.

□ Купили сте автомобил с ниски емисии

□ Купили сте продукти с ниски емисии за гориво за открит огън или барбекю (напр. брикети вместо въглища).

□ друго ..... *Моля, посочете.*

16. Бихте ли казали, че следното е много сериозен проблем, доста сериозен проблем, не много сериозен проблем или не сериозен проблем във вашата страна?

	Много сериозен проблем	Доста сериозен проблем	Не много сериозен проблем	Това изобщо не е сериозен проблем
Респираторни заболявания (напр. белодробни заболявания)				
Сърдечно-съдови заболявания (болести на сърцето)				
Астма и алергия				
Подкиселяване (киселинни дъждове, които засягат горите и др.)				
Еутрофикация (увеличаване на количеството органични вещества в екосистемата, например прекомерен растеж на водорасли, който води до измиране на риби в реките или езерата).				

17. Според вас всеки от изброените фактори прави ли твърде много, прави ли приблизително необходимото или не прави достатъчно за насърчаване на доброто качество на въздуха във вашата страна?

	Прекалено много работа	Правилното количество	Не правим достатъчно	Не знам
Домакинства				
Фермери				
Производители на енергия				
Производители на автомобили				
Обществени органи				

18. Според вас как най-добре може да се отговори на предизвикателствата, свързани със замърсяването на въздуха?

□ на местно ниво □ на национално равнище
□ на европейско равнище □ друго .....*Моля, посочете.*

19. Как оценявате цялостното качество на въздуха във вашия град/ село сега в сравнение с миналата година?

□ много по-добре □ малко по-добре
□ приблизително същото □ малко по-лошо
□ много по-лошо □ друго.....

20. Кои са основните причини за замърсяването на въздуха във вашия град? Моля, изберете всички приложими.

□ строителство
□ промишлени източници/производствени съоръжения
□ моторни превозни средства
□ готвене и отопление в домакинството
□ нарастваща употреба на климатици
□ нарастване на населението
□ електроцентрали
□ дим от цигари
□ изхвърляне на отпадъци
□ изгаряне на отпадъци
□ замърсяване от други региони
□ друго .....

21. До каква степен замърсяването на въздуха ви засяга?

□ Дихателна недостатъчност/затруднено дишане
□ Извършване на по-малко дейности на открито
□ Да се грижа повече за кожата си
□ Правим повече, за да сме здрави
□ Усещане за депресия
□ Дразнене на очите/носа/гърлото
□ Кожни проблеми
□ Желание да се преместите в други по-малко замърсени места
□ Заболеваемост от астма
□ Лоша видимост
□ Безпокойство за средата на живот
□ друго ..... *Моля, посочете.*

22. Домът ви се намира в ...?

□ в тих район, със слаб автомобилен трафик
□ в шумен район, с интензивен автомобилен трафик
□ в шумна зона, поради разликата в източника на трафик
□ друго ..... *Моля, посочете.*

23. Усещате ли в дома си миризма на изгорели газове от автомобилния трафик?

□ да, всеки ден □ да, често □ рядко □ друго ..... *Моля, посочете.*

24. До каква степен изпитвате замърсяване от превозни средства (шум, изгорели газове и др.) в дома си?

□ много висока □ средна □ ниска
□ друго..... *Моля, посочете.*

25. Семейството ви има ли проблеми със съня през нощта (събуждане от шума от движението)?

□ да, много често □ често □ рядко
□ друго ..... *Моля, посочете.*

Част В – специален раздел за здравето на децата

Моля, прочетете всички въпроси и отговорете, като поставите отметка в квадратчето или дадете кратко обяснение, когато е уместно.

Проучването е анонимно и ви уверяваме, че поверителността на вашите индивидуални отговори ще бъде запазена.

1. Пол

□ мъж □ жена

2. Възраст

□ под 3 години □ 3 – 7 години □ 8 – 15 години □ над 15 години

3. Тегло на детето

□ при раждане ..... □ в момента .....

4. Възраст на майката при раждането на детето

□ под 20 години □ 21 – 30 години □ 31 – 40 години
□ 41 – 50 години □ над 50 години

5. Колко време е кърмено бебето (в месеци)?

□ под 1 месец □ 1 – 3 месеца □ 3 – 6 месеца
□ 6 – 9 месеца □ 9 – 12 месеца
□ друго ..... *Моля, посочете.*

6. Има ли пушачи в семейството ти? Колко?

□ 1 □ 2 □ 3 □ друго ..... *Посочете дали е майката.*

7. Има ли домашни любимци в дома ви? Колко?

□ да □ не □ друго ..... *Моля, посочете.*

8. Някой от родителите или братята/сестрите има ли алергично заболяване?

□ да □ не □ друго ..... *Моля, посочете.*

9. Вашето дете (респондент) има ли алергично заболяване?

□ да □ не □ друго ..... *Моля, посочете.*

10. Преминало ли е детето ви (респондентът) някакво сериозно заболяване до момента (необходимост от хоспитализация)?

□ да □ не □ друго ..... *Моля, посочете.*

11. Страда ли детето ви от респираторни заболявания (хрема, бронхит, пневмония) по-често от четири пъти годишно?

□ да □ не □ друго ..... *Моля, посочете.*

12. Регистрирали ли сте някои от следните симптоми при Вашето дете (респондент) през последните шест месеца?

	да	не	Не знам
упорита кашлица			
свирене хрипове			
суха кашлица през нощта			
сенна хрема			
пристъпи на затруднено дишане (астма).			
грип или друго заболяване, засягащо дихателната система.			
кръвясали очи (конюнктивит)			

Д

УЧЕБНА ПРОГРАМА

Тази част от наръчника представлява учебна програма за курса „Разширени технологии за обработка на големи данни“. Тази учебна програма е предоставена от екипа на Факултета по природни науки към Университета „Матей Бел“ в Банска Бистрица, Словакия. Този партньор е приложил този курс и всички партньори ще го прилагат по време на устойчивостта на проекта.

Университет: Университет „Матей Бел“, Банска Бистрица, Словакия
Факултет: Факултет по природни науки
Код: DEK FPV/2d-fpv-401
Име на курса: Разширени технологии за обработка на големи масиви от данни в областта на природните науки
Вид, натовареност и методи на образователните дейности: Вид на курса: избираем Препоръчителна натовареност: 2 часа семинари/седмица Метод на обучение: комбиниран Форма на обучение: редовна Брой кредити: 3 Препоръчителен семестър: втори семестър на магистърското обучение
Степен на образование: втора (Магистър)
Предварителни курсове: няма предварителни изисквания
Условия за преминаване и завършване на курса: а) непрекъснато оценяване: активно участие в упражненията, изпълнение на възложените задачи 100 % б) крайна оценка: 0 % Оценяването на предмета е в съответствие с класификационната скала, определена от учебния правилник на УМБ.
Резултати от обучението: Студентите ще придобият знания и умения в следните области: 1. Въведение в обработката и анализа на данни 2. Въведение в основните задачи за анализ на данни – регресия и класификация

3. Въведение в работата с големи данни – методи за вземане на извадки от данни
4. Статистически методи за анализ на данни
5. Основи на изследователския анализ на данни – теория и практика
6. Въведение в „размитите“ множества
7. „Размити“ множества и регресионна задача
8. „Размити“ множества и задачи за класификация
9. Въведение в невронните мрежи

По време на курса студентът ще придобие опит в работата с:

1. Софтуерен инструмент MATLAB
2. Софтуерен инструмент R

ЛИТЕРАТУРНИ ИЗТОЧНИЦИ

Литературни източници за Секции 1 – 4:

- C.J. Date. An Introduction to Database Systems (8th. ed.). Addison-Wesley Longman Publishing Co., 2003. ISBN 978-0-321-19784-9
- Felix Kutsanedzie, Sylvester Achio, Edmund Ameko. Practical Approaches to Measurements, Sampling Techniques and Data Analysis. Science Publishing Group, 2016. ISBN 978-1-940366-58-6.
- William J. Lammers, Pietro Badia. Fundamentals of Behavioral Research Textbook. Online: https://uca.edu/psychology/fundamentals-of-behavioral-research-textbook/
- Jimin Quian et al. Introducing self-organized maps (SOM) as a visualization tool for materials research and education. Results in Materials, Volume 4, 2019, ISSN 2590-048X.
- Naseer Raheem. Big Data: A tutorial-based approach. Chapman and Hall/CRC, 2019. ISBN 978-0-367-67024-5
- Lior Rokach, Oded Maimon. Data mining with decision trees. 2015.
- Steven S. Skiena. The Data Science Design Manual. Springer, 2017. ISBN 978-3-319-55443-3
- Karthik Ramasubramanian, Abhishek Singh. Machine Learning Using R. Springer, 2019. ISBN 978-1-4842-4214-8
- Patrik Očenáš. Parallel and distributed methods of big data sampling (in Slovak). 2023.
- Bianka Modrovičová. Decision trees for sizable graph datasets (in Slovak). 2023.
- Aneta Szoliková. Explorative data analysis in document databases (in Slovak). 2023.
- Adam Dudáš, Bianka Modrovičová. Decision Trees in Proper Edge k-coloring of Cubic Graphs. In Proceedings of 33rd FRUCT conference. 2023.

Литературни източници за Секции 5 – 8:

1. ZADEH, L. A. Fuzzy Sets. In: Information and Control, 8, 1965, 338–353.
2. MICHALÍKOVÁ, A.: Fuzzy množiny v informatike. rec. Mirko Navara, Martin Kalina, Martin Klimo. Belianum. Matej Bel University in Banská Bystrica, 1, 2020, 206 p. ISBN 978-80-557-1707-4
3. Sendai Subway. Japan Visitor [cit. 2023-02-02]. Online: https://www.japanvisitor.com/japan-transport/sendai-subway
4. RUAN D.: Fuzzy Logic Applications in Nuclear Industry. Fuzzy Logic Foundations and Industrial Applications. 1996, 8, ISBN 978-1-4612-8627-1.
5. TAKAGI, T., SUGENO, M. Fuzzy Identifications of Fuzzy Systems and its Applications to Modelling and Control. In: IEEE Transactions on Systems, Man, and Cybernetics, 15(1), 1985, 116–132.
6. ROSS, T. J. Fuzzy Logic with Engineering Applications. John Wiley & Sons, 2005, 585s., ISBN 9780470743768.
7. ZADEH, L. A., The Concept of a Linguistic Variable and its Application to Approximate Reasoning – 1, In: Information Sciences, 8, 1975, 199–249.

Литературни източници за Секции 9 – 10:

- Basic Neural Networks 1 – https://docs.google.com/a/atu.edu.tr/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxpaHNhbnlhc3NpbjJ8Z3g6NGY4MjNjN2Y4ZTdhNWM2MQ
- Basic Neural Networks 2 – http://www.cs.stir.ac.uk/courses/ITNP4B/lectures/
- Basic Neural Networks 3 https://www.cs.bham.ac.uk/~jxb/inn.html
- Basic Neural Network 4 https://www.fer.unizg.hr/en/course/neunet_a/lecture_notes
- Basic Neural Network 5 http://users.monash.edu/~cema/courses/FIT3094/lecturePDFs/

Литературни източници за Секция 11:

1. Paluszek, M., Thomas, S. Matlab machine learning recepies. 2019. Plainsboro, NJ, USA. ISBN-13 (pbk): 978-1-4842-3915-5. DOI 10.1007/978-1-4842-3916-2.
2. Kim, P. MATLAB Deep Learning. With Machine Learning, Neural Networks and Artificial Intelligence. 2017. Apress Korea ISBN-13 (pbk): 978-1-4842-2844-9. DOI 10.1007/978-1-4842-2845-6.

3. Get Started with Matlab. https://www.mathworks.com/help/matlab/getting-started-with-matlab.html

4. Iris Clustering. https://www.mathworks.com/help/deeplearning/ug/iris-clustering.html

Литературни източници за Приложенията:

- Fisher, R. A. (1936) "The use of multiple measurements in taxonomic problems". Annual Eugenics, 7, Part II, pages 179–188
- Gates, G. W. (1972) "The Reduced Nearest Neighbor Rule". IEEE Transactions on Information Theory, May 1972, pages 431–433
- Duda, R. O., Hart, P. E. (1973) Pattern Classification and Scene Analysis. (Q327.D83) John Wiley & Sons. ISBN 0-471-22361-1, page 218
- Dasarathy, B. V. (1980) "Nosing Around the Neighborhood: A New System Structure and Classification Rule for Recognition in Partially Exposed Environments". IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-2, No. 1, pages 67–71
- https://sentinels.copernicus.eu/web/sentinel/missions/sentinel-3/data-products
- https://sentinels.copernicus.eu/web/sentinel/missions/sentinel-4/data-products
- https://climexp.knmi.nl/
- https://www.uradmonitor.com/
- Velea L, Udriștioiu MT, Puiu S, Motișan R, Amarie (2023)D. A Community-Based Sensor Network for Monitoring the Air Quality in Urban Romania. Atmosphere; 14(5):840. https://doi.org/10.3390/atmos14050840
- https://bookdown.org/floriandierickx/bookdown-demo/climate-data-from-models.html#differences-between-climate-projections-predictions-and-scenarios
- https://ec.europa.eu/eurostat/web/climate-change/database
- https://ourworldindata.org/
- https://ourworldindata.org/data-review-air-pollution-deaths
- https://ourworldindata.org/outdoor-air-pollution#outdoor-air-pollution-deaths-by-age
- https://www.who.int/health-topics/air-pollution#tab=tab_1

- https://www.eea.europa.eu/en/topics/in-depth/air-pollution
- https://www.who.int/teams/environment-climate-change-and-health/air-quality-and-health/health-impacts/types-of-pollutants
- https://www.who.int/publications/i/item/9789240034228
- https://apps.who.int/iris/bitstream/handle/10665/345329/9789240034228-eng.pdf
- EEA, 2012, The contribution of transport to air quality, EEA Report no. 10/2012, European Environment Agency.
- EEA. A closer look at urban transport TERM 2013: transport indicators tracking progress towards environmental targets in Europe EEA Report No 11/2013 Copenhagen, ISSN 1725-9177.
- http://dx.doi.org/10.1016/j.envpol.2007.06.012
- https://www.who.int/health-topics/air-pollution#tab=tab_1
- Report no. 05/2022, Air quality in Europe 2022. doi: 10.2800/488115. https://www.eea.europa.eu/publications/air-quality-in-europe-2022
- Xin Zhang, X. Chen, Xiaobo Zhang. The impact of exposure to air pollution on cognitive performance. Proc. Natl. Acad. Sci. Unit. States Am., 115 (2018), pp. 9193–9197, 10.1073/pnas.1809474115
- J. Currie, J. S. G. Zivin, J. Mullins, M. J. Neidell. What do we know about short and long term effects of early life exposure to pollution? NBER Work. Pap., 6 (2013), pp. 217–247, 10.3386/w19571
- Escamilla-Nuñez M-C., Barraza-Villarreal A., Hernandez-Cadena L., Moreno-Macias H., Ramirez-Aguilar M., Sienra-Monge J-J., Cortez-Lugo M., Texcalac J-L., del Rio-Navarro B., Romieu I. Traffic-Related Air Pollution and Respiratory Symptoms Among Asthmatic Children, Resident in Mexico City: The EVA Cohort Study. http://www.medscape.com/viewarticle/585875.
- Juvin P., Fournier T., Boland S. et al. Diesel particles are taken up by alveolar type II tumor cells and alter cytokines secretion. Arch Environ Health. 2002; 57(1):53–60.
- Le Tertre A., S. Medina, E. Samoli et al: Short term effects of particulate air pollution on cardiovascular disease in eight European cities. J. Epidemiol Community Health, 2002; 56, (10):773–9.
- Nordling E., Berglind N., Melén E., Emenius G., Hallberg J., Nyberg F., Pershagen G., Svartengren M., Wickman M., Bellander T. Traffic related air pollution and childhood respiratory symptoms, function and allergies. Epidemiology. 2008; 19(3):401–8.

- Pan G., Zhang S., Feng Y., Takahashi K., Kagawa J., Yu L., Wang P., Liu M., Liu Q., Hou S., Pan B., Li J. Air pollution and children's respiratory symptoms in six cities of Northern China. Respiratory Medicine 2010;104(12):1903–11.
- Richardson E.A., Pearce J., Tunstall H., Mitchell R., Shortt N.K.: Particulate air pollution and health inequalities: a Europe-wide ecological analysis. Int J Health Geogr 2013;12:34
- I. Jáuregui, J. Mullol, I. Dávila, M. Ferrer, J. Bartra, A. Del Cuvillo, J. Montoro, J. Sastre, A. Valero. Allergic rhinitis and school performance. J Investig. Allergol. Clin. Immunol., 19 (2009), pp. 32-39
- D. P. Skoner. Allergic rhinitis: definition, epidemiology, pathophysiology, detection, and diagnosis. J. Allergy Clin. Immunol., 108 (2001), pp. 2-8, 10.1067/mai.2001.115569
- I. Beck, S. Jochner, S. Gilles, M. McIntyre, J.T.M. Buters, C. Schmidt-Weber, H. Behrendt, J. Ring, A. Menzel, C. Traidl-Hoffmann. High environmental ozone levels lead to enhanced allergenicity of birch pollen. PloS One, 8 (2013), 10.1371/journal.pone.0080147
- P. Sturdy, S. Bremner, G. Harper, L. Mayhew, S. Eldridge, J. Eversley, A. Sheikh, S. Hunter, K. Boomla, G. Feder, K. Prescott, C. Griffiths. Impact of asthma on educational attainment in a socioeconomically deprived population: a study linking health, education and social care datasets. PloS One, 7 (2012), pp. 1-8, 10.1371/journal.pone.0043977
- https://europa.eu/eurobarometer/surveys/detail/2660
- https://data.europa.eu/data/datasets/s2660_97_2_sp524_eng?locale=en
- https://www.surveymonkey.com/r/airpollutionperceptionsurvey
- https://apps.who.int/iris/rest/bitstreams/1350812/retrieve
- https://www.ab.gov.tr/files/ardb/evt/Attitudes_of_Europeans_towards_air_quality_2013.pdf

Михаела Тинка Удристиу, Адам Дудаш, Алжбета Михаликова,
Фатих Килич, Ондер Тутсой, Ярмила Шкринарова,
Михаела Тинка Удристиу, Силвия Пюиу, Славея Петрова

Усъвършенствани технологии за обработка и анализ на големи масиви от данни

Българска, първо издание

Предпечатна подготовка: Георги Ташков
Печат и подвързия: Пловдивско университетско издателство

Пловдив, 2023
ISBN 978-619-7663-80-8