POKROČILÉ TECHNOLÓGIE SPRACOVANIA A ANALÝZY VEĽKÝCH DÁT

EDITOR

Adam DUDÁŠ

Autori

Mihaela Tinca Udriştioiu

Adam Dudáš

Alžbeta Michalíková

Fatih Kilic

Onder Tutsoy

Jarmila Škrinárová

Silvia Puiu

Slaveya Petrova

Táto publikácia bola financovaná Európskou komisiou v rámci projektu

Erasmus+ Uplatňovanie niektorých pokročilých technológií vo výučbe a výskume v súvislosti so skúmaním znečistenia ovzdušia

Kód projektu: 2021-1-RO01-KA220-HED-00003028

Podpora Európskej komisie na vydanie tejto publikácie nepredstavuje schválenie jej obsahu, ktorý vyjadruje len názory autorov, a národná agentúra a Európska komisia nenesú zodpovednosť za akékoľvek použitie informácií v nej obsiahnutých.

Financované Európskou úniou

University of Craiova

University of Plovdiv "Paisii Hilendarski"

Adana Alparslan Türkeş Science and Technology University

Matej Bel University Banská Bystrica

ISBN
978-625-399-465-5

Book Title
Pokročilé technológie spracovania a analýzy veľkých dát

Editor
Adam DUDÁŠ
ORCID iD: 0000-0001-5517-9464

Project manager
Mihaela Tinca UDRISTIOIU
ORCID iD: 0000-0002-5811-5930

Publishing Coordinator
Yasin DİLMEN

Page and Cover Design
Akademisyen Dizgi Ünitesi

Publisher Certificate Number
47518

Printing and Binding
Vadi Matbaacılık

Bisac Code
BUS070030

DOI
10.37609/akya.2892

Library ID Card
Tınca Udrıstıoıu, Mihaela and others.
Pokročilé technológie spracovania a analızy veľkıch dát / Mihaela Tınca Udrıstıoıu, Adam Dudaš, Alžbeta Michalikova [and others] ; editör : Adam Dudas.
Ankara : Akademisyen Yayınevi Kitabevi, 2023.
172 page. : figure, table. ; 195x275 mm.
Includes Bibliography.
ISBN 9786253994655
1. Information Technology.

GENERAL DISTRIBUTION
Akademisyen Kitabevi A.Ş.
Halk Sokak 5 / A Yenişehir / Ankara
Tel: 0312 431 16 33
siparis@akademisyen.com

www.akademisyen.com

OBSAH

ÚVOD

Táto učebnica predstavuje jeden z výsledkov dosiahnutých v rámci Erasmus+ projektu číslo 2021-1-RO01-KA220-HED-000030286 s názvom "Uplatňovanie niektorých pokročilých technológií vo výučbe a výskume v súvislosti so skúmaním znečistenia ovzdušia". Na dosiahnutí tohto cieľa spolupracovali štyri partnerské organizácie: Univerzita Mateja Bela v Banskej Bystrici zo Slovenska, Univerzita v Craiove z Rumunska, Univerzita Paisij Chilendarského v Plovdive z Bulharska a Adanská univerzita vied a technológií z Turecka. Cieľom učebnice je pomôcť inštruktorom STEM predmetov zlepšiť zručnosti študentov pri práci s dátami.

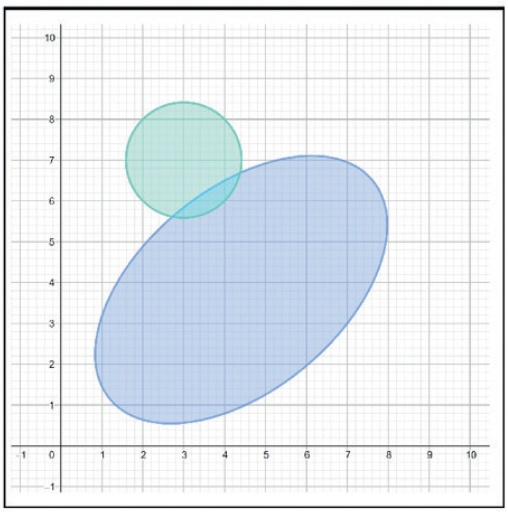
Sme zahltení informáciami, ktoré sú okolo nás. Pre získanie informácií relevantných pre každý zvolený cieľ skúmania je v dnešnej dobe potrebné vedieť spracovávať dáta. Počítače, senzorové siete a satelity každú sekundu zhromažďujú milióny hodnôt rôznych fyzikálnych alebo iných veličín a parametrov. Databázy uchovávajú a organizujú údaje a informácie, čím zlepšujú kvalitu dát. Keďže dôležitosť informácií rastie viac ako kedykoľvek predtým, študenti STEM predmetov sa musia naučiť pracovať s údajmi. Moderné spoločnosti vyžadujú vysokoškolské vzdelanie, aby poskytovali vysokokvalifikovaných absolventov schopných riešiť problémy na základe informácií získaných zo špecializovaných databáz alebo pomocou programov či algoritmov. Na univerzitách by STEM študenti mali študovať, ako sa množiny dát zhromažďujú, analyzujú a interpretujú – činnosti, ktoré pomôžu pri klasifikácii a aproximácii údajov a pri vytváraní kvalitných odhadov. Nakoniec, trh práce žiada absolventov STEM, aby vytvárali predpovede toho, ako sa procesy vyvíjajú v priestore a čase, alebo aby robili dôležité rozhodnutia. Strojové učenie sa a umelá inteligencia sú štandardné pojmy v každodennej slovnej zásobe študentov.

Táto učebnica obsahuje jedenásť častí, niekoľko príloh a zoznam literatúry relevantnej pre opisované oblasti analýzy dát. Prvá časť učebnice je zameraná na rôzne typy dát, ich vlastnosti, metódy vzorkovania dát a spôsob spracovania a analýzy dát. Nasledujúce časti približujú jeden z najvýznamnejších procesov súvisiacich s veľkými dátami, analýzu dát. Pri analýze veľkých dát je potrebné vedieť používať vhodné metódy štatistickej analýzy, vizualizáciu dát a ďalšie exploratívne, prediktívne a odhadovacie metódy. Jednotlivé sekcie učebnice sa zameriavajú na prístupy, ako je strojové učenie sa, fuzzy inferencia a systémy využívajúce neurónové siete. Prílohy učebnice obsahujú opis datasetu Iris, príklady riešení niektorých problémov, opis datasetov o klimatických zmenách alebo znečistení ovzdušia a informácie o vplyve znečistenia ovzdušia na ľudské zdravie. Príručku uzatvára príklad učebného plánu pre kurz "Pokročilé technológie spracovania a analýzy veľkých dát".

KAPITOLA 1

DÁTA A ICH VLASTNOSTI

Autorom tejto časti učebnice je Adam Dudáš z Katedry informatiky, Fakulty prírodných vied, Univerzity Mateja Bela v Banskej Bystrici zo Slovenska.



	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

Táto učebnica je zameraná na demonštráciu jednoduchých metód analýzy dát, ktoré využívajú techniky z oblasti informatiky akými sú umelá inteligencia, strojové učenie sa alebo neurónové siete. V tejto časti učebnice sa budeme venovať základným pojmom a konceptom, ktoré sa týkajú dát, ich vlastností, ich spracovaniu a – v neposlednom rade – ich analýze.

Dáta sú technické, štatistické, ekonomické alebo iné správy a informácie, ktoré môžu byť spracované pomocou technických prostriedkov. V našom prípade, sú týmito technickými prostriedkami počítače. K dátam pristupujeme ako k objektom, ktoré sú v rámci systému integrované a zdieľané:

- **Integrácia dát** – dáta môžu byť uložené v niekoľkých súboroch takým spôsobom, aby bola minimalizovaná duplicita dát samotných a zároveň aby bolo možné k viacerým súborom pristupovať súčasne.
- **Zdieľanie dát** – každý dátový objekt môže byť zdieľaný niekoľkými používateľmi (opakovane a súčasne).

Najdôležitejšou vlastnosťou dát je však ich **perzistencia** – perzistentné dáta sú dáta, ktoré v systéme existujú aj po ukončení programu. Je zrejmé, že vstupné dáta môžu byť transformované na perzistentné dáta a výstupné dáta môžu byť transformované z perzistentných, vstupných alebo inak vypočítaných dát. Dáta, ktoré je možné vypočítať z iných dát by nemali byť perzistentné (takýmto prístupom by sme zvýšili náklady na prácu systému) – niekedy je to však nevyhnutné.

Keďže je pri zbieraní dátových celkov potrebné efektívne ukladanie dát do systému, je dôležité rozhodnúť ako budú dáta v jednotlivých záznamoch reprezentované z pohľadu špecifického dátového typu. Najtypickejšími **dátovými typmi** sú:

- **Numerické dáta** – môžu byť uložené niekoľkými spôsobmi (binárne, znaky, semi-logaritmická forma, ...). Pri tomto dátovom type je často potrebné definovať počet potrebných bitov/bytov pre uloženie čísla.
- **Reťazce** – bývajú ukladané pomocou rôznych znakových sád (ASCII, UNICODE, EBDIC, ...).
- **Počítadlá** – označujú používanie znakových kódov namiesto reťazcov (napríklad A namiesto výborne, B namiesto veľmi dobre, ...).
- **Jednotky** – dôležité je spomenúť aj jednotky, v ktorých sú dáta merané. Tie by mali byť prispôsobené konkrétnym situáciám (nezmyslom by bolo meranie letovej vzdialenosti v milimetroch).

Tieto implementácie dátových typov pre nás z pohľadu analýzy dát nie sú príliš zaujímavé. Vo všeobecnosti budeme v rámci tejto učebnice hovoriť o dvoch typoch dát, ktoré sú navzájom odlíšiteľné svojim obsahom:

- **Kvantitatívne dáta obsahujúce numerické hodnoty** (výška, vzdialenosť, číslo, ...). Tento typ dát môže byť priamo využitý v matematických modeloch, ktoré sú kritické pri analýze dát prostredníctvom metód strojového učenia sa.
- **Kategorické dáta obsahujúce lingvistické opisy** vlastností (pohlavie, farba, druh, ...), ktoré implikujú potrebu špecificky navrhnutých metód analýzy dát. Niektoré kategorické dáta môžu byť zakódované do kvantitatívnych, ale takáto operácia nie je vždy zmysluplná. Príkladom takéhoto kódovania by mohol byť prípad, kde *AK pohlavie = muž, POTOM pohlavie = 1* a *AK pohlavie = žena, POTOM pohlavie = 2* a tak ďalej. Takéto kódovanie je do určitej miery logické, ale tiež z neho vyplýva niekoľko otázok, ktoré od neho odrádzajú:

Keďže 2 – 1 = 1, je žena – muž = muž?

Aká je maximálna hodnota pohlavia?

Pred samotným procesom analýzy dát je najdôležitejšie, aby bola skúmaná dátová množina vhodne štruktúrovaná. Z pohľadu štruktúry rozlišujeme **tri typy dátových celkov**:

- **Štruktúrované dáta** – dáta uložené vo forme tabuliek alebo súborov, v ktorých je možné identifikovať rovnaké vlastnosti v rovnakom poradí (stĺpce) pre každý zaznamenaný objekt (riadok) Najčastejšie používaný formát štruktúrovaných dát je CSV, súbor programu Excel, jednoduchý text alebo SQL databáza.

- **Semi-štruktúrované dáta** – dáta sú štruktúrované, ale ich tvar nie je fixovaný. Môžeme teda determinovať rovnaké vlastnosti pre každý zo zaznamenaných objektov (riadkov), ale tieto vlastnosti nemusia byť zaznamenané pre každý z objektov alebo môžu byť zaznamenané v rôznom poradí (v súbore nie je možné identifikovať stĺpce). Toto je typické pre dáta zbierané pomocou siete navzájom nezávislých senzorov alebo mobilných aplikácií. Medzi formáty využívané pri tom type dát radíme XML, JSON alebo MongoDB.
- **Multi-štruktúrované** alebo **neštruktúrované dáta** – surové dáta rôznych formátov. Napríklad surové dáta tešúce zo senzorov, web logy, dáta zo sociálnych sietí, audio, video, 3D modely, súradnice a podobne.

Pri práci s neštruktúrovanými dátami často používame nasledovný pracovný tok:

Príklad: Konverzia z neštruktúrovanej dátovej formy do štruktúrovaných dát (tabuľka). Majme zbierku základných informácií o troch študentoch:

1. *Martin, muž, 28.6.1983, ročník štúdia: 2, 40 rokov.*
2. *Jana, 1994-9-13., Ž, 1. rok štúdia, 29 r.*
3. *Miriam, žena, Piaty apríl 1992, druhý rok štúdia, vek: 31*

Tieto informácie sú v nekonzistentnej forme – poradie a formát jednotlivých charakteristík je rozdielny pre každého z uvedených študentov. Táto zbierka dát o troch študentoch tiež obsahuje veľmi jednoducho vypočítateľné dáta, ktoré nie je potrebné ukladať (napríklad vek študenta). Práve preto pri ukladaní týchto dát do štruktúrovanej formy (tabuľky) potrebujeme zjednotiť poradie a formát všetkých vlastností (napríklad dátumy vo formáte RRRR-MM-DD).

Meno	Dátum narodenia	Rok štúdia	Pohlavie
Martin	*1983-6-28*	*2*	*M*
Jane	*1994-9-13*	*1*	*F*
Miriam	*1992-4-5*	*2*	*F*

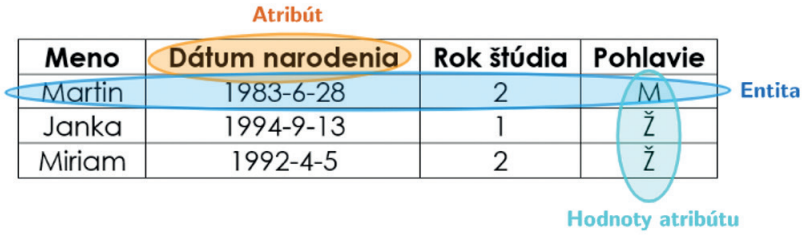
V kontexte tejto učebnice budeme používať rôzne názvy pre rovnaké objekty v dátových množinách. Preto na tomto mieste učebnice ponúkame stručný prehľad a vysvetlenie najzákladnejších z týchto pojmov.

Entita, objekt alebo záznam je objekt reálneho sveta, ktorý je schopný nezávislej existencie a je jednoznačne odlíšiteľný od ostatných objektov.

Atribút alebo vlastnosť je funkcia priraďujúca hodnotu k entite, ktorá determinuje nejakú esenciálnu vlastnosť entity (napríklad výška, vek, ...).

Dataset alebo tabuľka je množina entít pozostávajúcich z množiny rovnakých atribútov.

Štrukturalizácia zozbieraných dát je základnou metódou spracovania dát (pozri Sekciu 2).



1.1 NIEKOĽKO SLOV K TÉME VEĽKÝCH DÁT

Principiálne je vždy lepšie mať dát priveľa než primálo (v takomto prípade môžeme stále niektoré záznamy z datasetu vyhodiť). Dáta môžeme volať veľkými v prípade, že ich spracovanie a analýza nie je možná s využitím konvenčných nástrojov v praktickom čase. Samozrejme, pri takejto definícii vyvstáva niekoľko základných otázok: *Aký je to praktický čas? Čo je považované za konvenčný nástroj?* Odpovede na obe tieto otázky sa v čase menia a preto je častejšie používaná definícia veľkých dát pomocou vymenovania ich vlastností.

Dáta sú často označované ako veľké v prípade, že nadobúdajú vlastnosti označované ako 3V (počet týchto "V" časom rastie, v niektorých literárnych zdrojoch je uvedených 5V ako základný model pohľadu na veľké dáta):

- **Objem dát** (z anglického *Volume of the data*) – to, koľko dát je v moderných systémoch generované robí použitie štandardných (relačných) databázových modelov nemysliteľným. V prípade veľkých dát nie sme schopní dataset reprezentovať pomocou jednej jednoduchej tabuľky a pracovať na jednom stroji, z čoho vyplýva aj nárast potreby vývoja sofistikovanej výpočtovej infraštruktúry a implementácie optimalizovaných algoritmov. Systém pre veľké dáta potrebuje implementovať princípy vysokovýkonných, distribuovaných a cloudových výpočtových systémov, nerelačné databázy, do ktorých je možné ukladať heterogénne a logicky prepojené dáta a modely umelej inteligencie využiteľné v spracovaní a analýze dáta.
- **Rozmanitosť dát** (z anglického *Variety of the data*) – keďže množina dát, o ktorej môžeme v reálnych problémoch zahrňujúcich veľké dáta uvažovať, je zriedka homogénna, systém musí byť schopný práce s niekoľkými typmi a formátmi súborov, napríklad jednoduché textové dokumenty, audio súbory, video súbory, súradnice alebo počítačové modely berúce do úvahy dve a viac dátových dimenzií. Keďže väčšina z týchto typov dát vyžaduje na ich pohodlné uloženie a spracovanie veľké množstvo výpočtovej sily a úložného priestoru, nie je možné ukladať ich v relačných databázach.
- **Rýchlosť dát** (z anglického *Velocity of the data*) – Veľké datasety sú často živé (tiež označované ako dynamické) – datasety, ktoré sa menia v priebehu času. K takejto zmene zloženia datasetu v priebehu času dochádza vo všetkých systémoch, do ktorých sú zakomponované takzvané ambientné dátové zdroje – zdroje, ktoré sú vždy aktívne a zbierajú dáta, napríklad senzory zapojené do Internetu vecí, ktoré sledujú zmeny hodnôt niekoľkých meraných veličín. Živosť dát a ich spracovanie, ukladanie a analýza tvorí toky dát, ktoré plynú do systému. Tento fakt so sebou prináša jednu z najdôležitejších požiadaviek na systémy pre veľké dáta – schopnosť zbierať, ukladať, spracovávať a analyzovať dáta v (takmer) skutočnom čase.

Okrem problémov, ktoré sú spôsobené živými dátami, môžeme identifikovať problémy vyplývajúce z kombinácie dynamických a statických častí datasetov, čo spôsobuje niekoľko problematických udalostí v systémoch.

Spolu s vlastnosťami veľkých dát ako sú objem, rozmanitosť a rýchlosť množstvo literárnych prameňov uvádza aj spoľahlivosť a hodnotu dát:

- **Spoľahlivosť dát** (z anglického *Veracity of the data*) – keďže veľké dáta sú často využívané v kontexte rozhodovacích procesov, ktoré berú do úvahy veľké množstvo entít reálneho sveta, je kritické, aby boli tieto entity vierohodné a spoľahlivé. V tomto prípade je potrebné pracovať s metrikami, ktoré merajú našu dôveru v dáta, čo je podstatné nie len v prípade rozhodovacích procesov ale tiež v prípade skutočnosti dát – generovanie umelých množín veľkých dát nie je zložité a preto môže byť použité ako forma útoku s cieľom preťaženia cieľového systému.
- **Hodnota dát** (z anlglického *Value of the data*) – ako bolo spomenuté vyššie, veľké dáta môžu (a sú) používané v procesoch rozhodovania sa. Hodnota dát stúpa spolu s ich veľkosťou v prípade, že sa takáto množina dát týka jedného problému alebo oblasti. Pri veľkých dátových množinách stúpa aj potenciál využiteľnosti týchto dát pri odhadovaní a predikovaní udalostí. Hodnota dát môže byť vnímaná z finančného, ľudského, výskumného alebo iného pohľadu.

Tieto vlastnosti veľkých dát prinášajú niekoľko **problémov súvisiacich s ich spracovaním a analýzou**.

Prvým z týchto problémov je **samotná veľkosť dát**. Ich veľkosť je dôležitá nielen v súvislosti s pamäťovým priestorom, ktorý je potrebný na ukladanie samotných dát, ale aj z hľadiska vyhľadávania v dátach a analýzy týchto dát. Pri práci s takýmito údajmi je potrebné využívať metódy vysokovýkonného, distribuovaného alebo cloudového počítania v spojení s algoritmami umelej inteligencie z oblasti strojového učenia sa, fuzzy inferenčných systémov a neurónových sietí na získanie poznatkov z takýchto množín dát.

Okrem toho, že tieto dáta sú veľké, sú často zložené z **heterogénnych partícií, ktoré sa môžu líšiť vo viacerých aspektoch** – dimenzionalita dát, zloženie dát, štruktúra dát, ale aj použité jednotky. Táto nekonzistentnosť je dôsledkom skutočnosti, že veľké datasety sa často zhromažďujú z niekoľkých navzájom nekompatibilných zdrojov do jedného úložiska. Preto je potrebné, aby sa dátové partície preformátovali (resp. aby sa jednotlivé dátové formáty zjednotili). Tento proces predstavuje postupnosť jednoduchých úloh, ktoré je potrebné vykonať na dátach (napríklad konverzia jednotiek v prípade potreby), ale aj úlohy, ktoré sú zložitejšie, napr. identifikácia odľahlých a chýbajúcich hodnôt. V prípade chýbajúcich hodnôt je možné vykonať opatrenia na spätné dopočítanie chýbajúcich hodnôt – na odhad hodnôt alebo klasifikáciu údajov možno použiť metódy strojového učenia sa alebo neurónové siete.

Problém úzko súvisiaci s heterogenitou veľkých datasetov spočíva v **živosti týchto súborov dát**. V prípade, že zbierame dáta s využitím ambientných dátových zdrojov (ako sú senzorové siete) a tieto merania sú vykonávané na dostatočne veľkom počte senzorov v dostatočne malých časových intervaloch, vytvárame dátové toky, ktoré je potrebné spracovávať a pripravovať na analýzu v systéme. Preto musí byť tento systém schopný spracovať a analyzovať datasety, ktoré sa časom menia.

Jedným z najvýznamnejších problémov súvisiacich s veľkými množinami dát je **analýza dát**. Analýza musí byť podporená vysokovýkonným, distribuovaným alebo cloudovým výpočtovým systémom, správnou dekompozíciou problému a strojovým učením sa, výpočtovými fuzzy modelmi a neurónovými sieťami. Pri analýze veľkých datasetov môžeme použiť metódy štatistickej analýzy,

vizualizácie dát a ďalšie metódy exploratívnej alebo prediktívnej analýzy dát s využitím strojového učenia sa, fuzzy inferenčných systémov a prístupov neurónových sietí (pozri Sekciu 2).

1.2 BEŽNÉ PROBLÉMY V DÁTOVÝCH MNOŽINÁCH

Existuje niekoľko bežných problémov súvisiacich s dátami, ktoré neboli opísané vyššie – konkrétne v problémoch týkajúcich sa veľkých dát.

Ako už bolo spomenuté vyššie, množstvo dát neustále rastie, čo znamená, že analýza týchto uložených a spracovaných dát trvá dlhšie. Analýza je však podstatou samotného ukladania dát, a preto sa jej nemožno vyhnúť. To so sebou prináša potrebu vývoja metód a postupov, ktoré nám umožňujú získavať znalosti a podporujú rozhodovanie v kontexte veľkých datsetov.

Dátové sady určené pre potreby konkrétnych úloh sú často vytvárné kombináciou dát z viacerých zdrojov. Tieto zdroje sú charakteristické rôznorodosťou formátov a zloženia jednotlivých dátových častí, a preto potrebujeme spôsob, ako takéto rôznorodé dáta zbierať a zjednocovať pre potreby ďalšej analýzy. S týmto bodom sa spája ešte jeden problém – keďže dáta pochádzajú z rôznych zdrojov, môže nastať situácia, keď si jednotlivé záznamy budú protirečiť (alebo nebudú navzájom konzistentné).

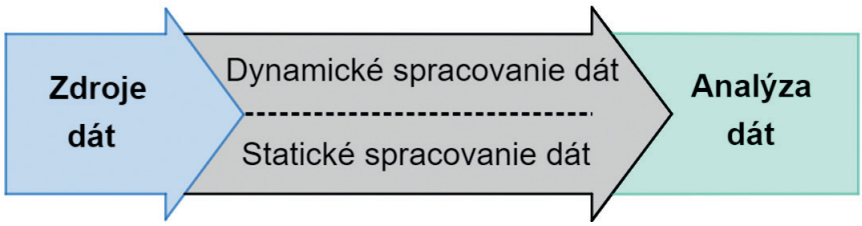
Bezpečnosť veľkých dát je značným problémom, aj keď pre nás v kontexte tejto učebnice nie je podstatná. Nie každý dátový zdroj je bezpečný a nemusí byť ani v súlade s politikou spoločnosti, ktorá ho chce využívať. Vo všeobecnosti je potrebné dbať na vytváranie autorizácie a autentifikácie, sledovanie používateľov, ktorí s dátami pracujú, bezpečnosť nespracovaných a získaných dát a ochranu komunikácie, teda prenosu dát.

Napokon, problémy, ktoré sú obzvlášť významné v kontexte vytvárania predpovedí alebo odhadov, sú chýbajúce dáta a odľahlé hodnoty v dátach. Oba problémy sú prirodzené – v prípade chýbajúcich hodnôt v dátach dochádza k situáciám, keď jedna z požadovaných nameraných hodnôt v dátovom súbore chýba. V prípade odľahlých hodnôt je to prirodzená situácia, keď niektoré namerané hodnoty ležia ďaleko mimo tela datasetu. Tieto dva problémy sú hlavným obsahom Sekcie 2 tejto učebnice.

KAPITOLA 2

SPRACOVANIE A ANALÝZA DÁT

Autorom tejto časti učebnice je Adam Dudáš z Katedry informatiky, Fakulty prírodných vied, Univerzity Mateja Bela v Banskej Bystrici zo Slovenska.



Z pohľadu tejto učebnice môžeme pri práci s dátami identifikovať dve hlavné činnosti – spracovanie dát a ich analýzu. V rámci tohto textu je našim hlavným cieľom prezentovať informácie a príklady analýzy dát, ktorú nie je možné efektívne vykonávať bez dát vhodných ako vstup do tohto procesu. Takáto príprava dát do tvaru vhodného na analyzovanie sa nazýva **spracovanie dát**. Potreba spracovania a analýzy dát vyplýva z niekoľkých charakteristík moderných datasetov:

- **Zdroje dát** – v dnešnej dobe je úplne bežné, že pracujeme s dátovými množinami, ktoré vznikajú spojením menších dátových celkov zozbieraných z rôznych zdrojov. Takto vytvorené dátové sady môžu pochádzať z rôznych databáz, rôznych senzorov v jednej sieti, ale môžu vznikať aj kombináciou týchto dvoch prístupov. Množiny dát, ktoré sa skladajú z menších častí, so sebou prinášajú veľmi prirodzené problémy spojené s týmto typom štruktúry údajov:
  - potreba homogenizácie dátovej štruktúry,
  - práca s chýbajúcimi hodnotami,
  - práca s odľahlými hodnotami.
- **Statické spracovanie dát** – v kontexte moderných systémov vnímame dva typy dát, ktoré sa môžu v danom systéme vyskytovať, resp. ktoré by sme v systéme mohli spracovávať – prvým typom dát sú statické údaje. Dáta nazývame statickými v prípade, že sa v priebehu času nemenia. Bežným prí-

stupom k spracovaniu takýchto dát je takzvané dávkové spracovanie. Štandardne tieto dávky úloh zahŕňajú načítanie súboru, spracovanie súboru a zápis výstupu do nového súboru bez akéhokoľvek manuálneho zásahu používateľa.

- **Dynamické spracovanie dát** – ak systém využíva zdroje okolitých dát (napríklad neustále aktívny senzor alebo sadu senzorov), musí byť schopný zachytiť a uložiť tieto dáta v čase blízkom reálnemu času. Nech sa v systéme používa ľubovoľný typ úložiska , je potrebné aby kvôli veľkému objemu dát podporoval škálovanie. Takéto dynamicky sa meniace datasety nazývame aj dátové toky, ktoré musíme vedieť spracovať, filtrovať, agregovať a inak pripraviť na analýzu. Takto spracované dáta sú následne odoslané na analýzu. Problém dynamického spracovania dát presahuje rámec tejto učebnice, ale je jednou z veľmi podstatných súčastí moderných dátových systémov.
- **Analýza dát** – analýzu dát môžeme vnímať ako činnosť získavania poznatkov pre potreby lepšieho rozhodovania v kontexte vybranej problémovej oblasti, možnosť predikcie hodnôt na základe zozbieraných dát, či odhadu nemeraných dát. V druhej časti tejto kapitoly opisujeme typy analýzy dát a hlavné problémy dátovej analýzy.

Táto časť učebnice je zameraná na spracovanie dát a vybrané problémy týkajúce sa tohto procesu. Druhá časť sekcie je zameraná na úvod do analýzy dát, o ktorej je následne podrobne diskutované v ostatku učebnice.

2.1 SPRACOVNIE DÁT

Nie vždy máme možnosť pracovať s množinou dát, ktorá je pripravená na priamu analýzu. Často (najmä pokiaľ ide o naše vlastné dáta) ide doslova o zozbierané súbory informácií. Preto je potrebné dáta pred analýzou **vyčistiť a naformátovať**.

Poznámka k tomuto procesu – spracovanie dát a všetky kroky opísané v tejto časti textu by sa mali vždy vykonávať na kópii pôvodného datasetu, nie na datasete samotnom. V ideálnom prípade by sme mali používať metódy, ktoré sú systematické a opakovateľné. Nechceme predsa prísť o ťažko nadobudnuté dáta.

Vnútorná konzistencia datasetu

Ako už bolo spomenuté na začiatku tejto časti učebnice, skutočnosť, že moderné množiny údajov vznikajú kombináciou viacerých – menších – dátových celkov, vytvára problémy súvisiace s vnútornou konzistenciou samotných datasetov. Túto nekonzistentnosť možno vnímať v dvoch rovinách – nekonzistentnosť samotných dát a nekonzistentnosť štruktúry datasetu.

Vieme identifikovať niekoľko základných problémov, ktorých následkom je nekonzistentnosť dát:

- **Konverzia jednotiek** – pri kombinácii dvoch množín údajov, ktoré na meranie hodnôt atribútov používajú rôzne jednotky (napríklad centimetre a milimetre), je potrebné jednotku merania zjednotiť. Nevyhnutné je aj zjednotenie datasetov meraných na kontinentoch, ktoré nepoužívajú rovnaké merné systémy – napríklad na meranie rovnakej veličiny sa v Európe použijú centimetre a v USA palce.
- **Číselné konverzie** – číselné hodnoty zaznamenané slovne je potrebné previesť na čísla. Takáto verzia nevyhnutných konverzií zahŕňa aj typické problémy so špecifikovaním jednotiek v rámci hodnoty atribútu.

- **Názvové/menné konverzie** – pri zaznamenávaní mien fyzických osôb alebo konkrétnych objektov je potrebné zjednotiť spôsob zápisu mien a priezvisk či názvov. Najväčším problémom v prípade datasetov využívajúcich atribúty mien z rôznych krajín sú znaky s diakritikou (napr. š, č, ä).
- **Časo-dátumové konverzie** – v prípade analýz obsahujúcich časové informácie je potrebné zjednotiť formát záznamu času, najmä dátum v danom datasete.
- **Finančné a menové konverzie** – hodnoty atribútov uvedené v rôznych menách je potrebné zjednotiť s jednou z mien, ktoré sa už v množine dát nachádzajú.

Druhým prípadom nekonzistencie datasetu je **nekonzistentnosť štruktúry datasetu**. Ideálnou metódou ukladania dát na ďalšiu analýzu je metóda opísaná v časti 1 tejto učebnice – ukladanie údajov vo forme tabuľky. Nie vždy sa to však dá dosiahnuť jednoduchým spôsobom – problémom v tomto prípade budú najmä chýbajúce hodnoty.

Množinu dát, ktorá obsahuje chýbajúce hodnoty, je problematické analyzovať pomocou štandardných nástrojov, ale aj pomocou akýchkoľvek softvérových nástrojov. Bunky pomyselnej tabuľky, v ktorých chýba hodnota, sú vyplnené NULL hodnotami, ktoré nie je možné štatisticky vyhodnotiť a zároveň ich nemožno brať ako hodnoty samotné (keďže 0 ≠ NULL). Preto je potrebné sa s týmto typom problémov určitým spôsobom vysporiadať.

Chýbajúce a poškodené dáta

Na účely tejto učebnice sú dáta považované za merania skutočných vlastností objektov reálneho sveta. Tieto merania sú ovplyvňované dvoma faktormi – nástrojom zberu dát a spôsobom spracovania dát. V prípade oboch faktorov môže nastať problém, ktorého dôsledkom je **strata dát alebo ich poškodenie**. Ak sa vyskytne problém s nástrojom na zber dát (vypálená časť senzoru, stratené záznamy po výpadku servera a pod.), hovoríme o strate dát, ktoré nie je možné rekonštruovať. Opakom je strata alebo poškodenie dát pri ich spracovaní. Ak máme k dispozícii nespracované dáta, oprava chyby nie je problematická – tento typ straty alebo poškodenia dát nazývame **artefakt**.

V prípade, že je dataset, s ktorým pracujeme neúplný, je potrebné identifikovať chýbajúce hodnoty a vhodne ich kompenzovať. Problémom je, že niektoré chýbajúce hodnoty nemusia existovať ani v reálnom svete. Príkladom môže byť hodnota pre atribút, ktorý obsahuje čas príchodu na určené miesto v situácii, keď sme na dané miesto ešte nedorazili.

Spôsoby práce s chýbajúcimi hodnotami v prípade, že nie sú k dispozícii surové údaje, možno rozdeliť do niekoľkých typov:

- **Nahradenie chýbajúcej hodnoty inou hodnotou** (0 / -1 / nonsens) – pri takomto prístupe by sme každú chýbajúcu hodnotu (NULL) nahradili vybranou, špeciálnou hodnotou. Tento prístup sa neodporúča – náhradné hodnoty možno často považovať za skutočné a pri analýze datasetu môžu byť nesprávne interpretované. Napríklad, ak nie je určená hodnota mzdy zamestnanca, nenahrádzame ju hodnotou 0 alebo -1, keďže zamestnanec nepracuje zadarmo alebo neplatí za to, že môže prísť práce.
- **Vylúčenie nekompletných entít z datasetu** – o niečo lepším prípadom v porovnaní s predchádzajúcim prístupom, by mohla byť metóda, kde z datasetu odstránime každý neúplný záznam. Tento prístup je v poriadku, ak máme dostatok údajov, no stále môže viesť k skresleným výsledkom.
- **Dopočítanie chýbajúcich hodnôt (imputácia)** – v prípade, že potrebujeme použiť záznamy, ktoré obsahujú chýbajúce hodnoty, môžeme tieto hodnoty vypočítať pomocou jednej z nižšie uvedených metód. Tento prístup nazývame aj imputácia hodnoty.

- Imputácia heuristickým prístupom – v prípade, že o datasete a vzťahoch v ňom vieme dostatočne veľa, mali by sme vedieť odhadnúť hodnotu niektorých atribútov.
- Imputácia priemernou hodnotou atribútu – táto metóda nahrádza chýbajúce hodnoty priemernou hodnotou pre daný atribút. Použitie takejto hodnoty je výhodné z niekoľkých dôvodov, z ktorých najdôležitejší je ten, že priemerné hodnoty atribútov nie sú silné v žiadnom smere, a preto majú malý vplyv na predikčný potenciál v datasete. Nie vždy je však vhodné nahradiť chýbajúce hodnoty priemernou hodnotou daného atribútu. Za priemerný plat by bol tento prístup v poriadku, ale priemerný dátum príchodu na dané miesto nedáva zmysel.
- Imputácia náhodnou hodnotou atribútu – pre chýbajúcu hodnotu zvolíme náhodnú hodnotu daného atribútu, ktorú máme zaznamenanú v datasete.
- Imputácia využívajúca metódy strojového učenia sa – najsofistikovanejším prístupom k výpočtu chýbajúcich údajov je použitie metód strojového učenia sa. Tieto metódy však nemožno použiť s akýmkoľvek datasetom – alebo presnejšie povedané, nie je možné ich efektívne použiť pre každý dataset. Metódy strojového učenia sa fungujú na základe korelácií medzi jednotlivými hodnotami v datasete a ak sú tieto korelácie slabé alebo neexistujúce, odhady hodnôt jednotlivých atribútov budú nepresné. Tento prístup je podrobnejšie opísaný v časti 4 tejto učebnice.

Odľahlé hodnoty

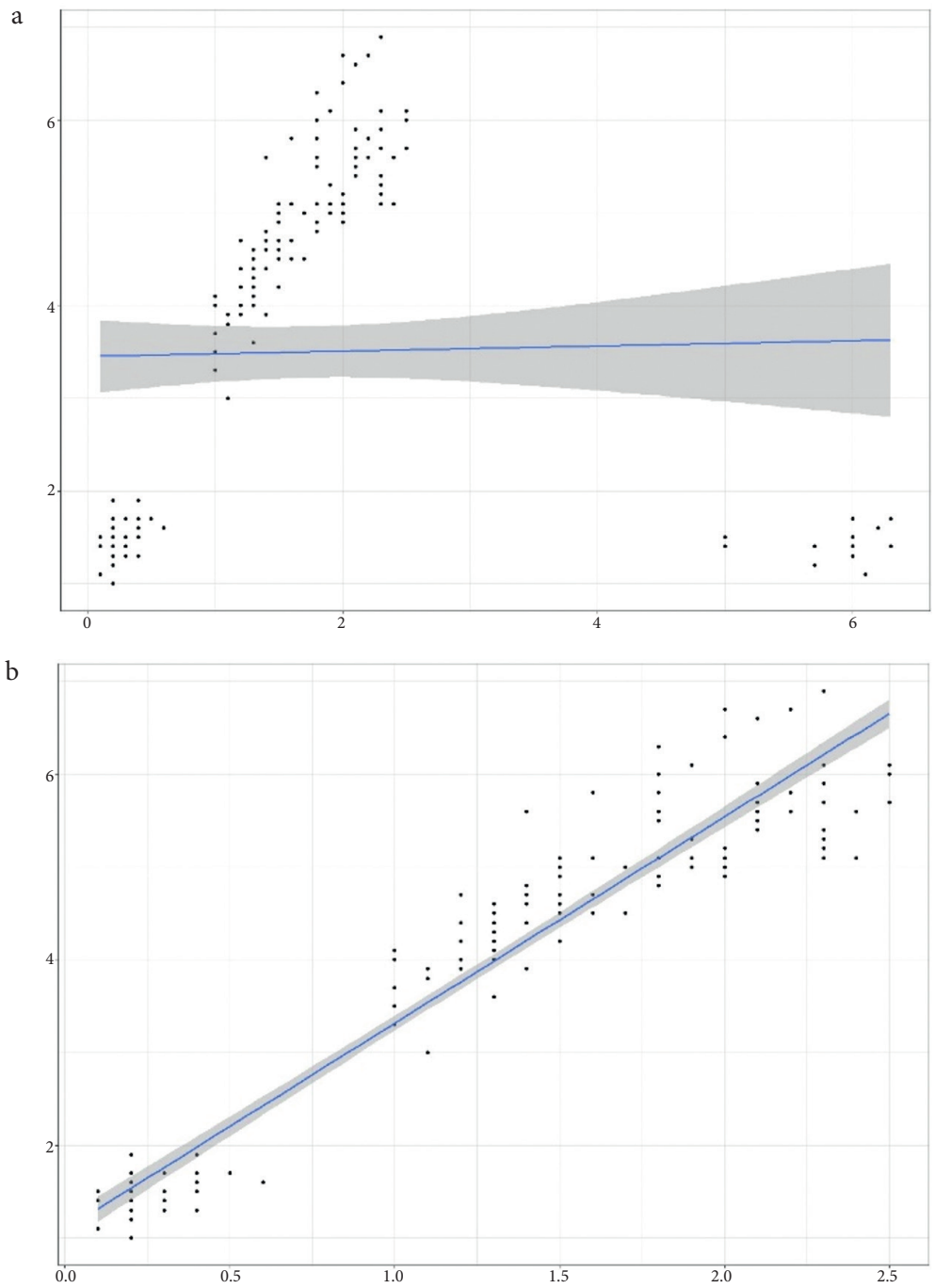
Odľahlé hodnoty sú hodnoty, ktoré sa nachádzajú mimo hlavného tela datasetu. V normálne distribuovanom súbore dát sa pravdepodobnosť výskytu konkrétnej hodnoty v datasete znižuje so vzdialenosťou od priemernej hodnoty daného datasetu. Problém však nastáva pri datasetoch s iným ako normálnym rozložením. **Odľahlé hodnoty vznikajú niekoľkými spôsobmi:**

- chyba pri meraní,
- preklep pri spracovávaní dát,
- nedôveryhodná informácie, ktorá môže nabádať k nedôveryhodnosti celého záznamu, v ktorom sa nachádza.

Často však ide o reálnu hodnotu, ktorá sa odchyľuje od štandardných situácií (napríklad obdobia znečistenia ovzdušia), preto je potrebné analyzovať záznam ako celok.

Problém s odľahlými hodnotami nastáva pri pokuse o **zovšeobecnenie** informácií na základe dát, ktoré obsahujú odľahlé hodnoty. Na obrázku nižšie vidíme pokus opísať daný dataset pomocou priamky. Na ľavej strane obrázku je dataset obsahujúci 162 záznamov, z ktorých 12 je umiestnených výrazne mimo tela datasetu. V tomto prípade vidíme, že modrá čiara, ktorá by mala prechádzať stredom množiny dát, ju až na jeden bod úplne míňa. Vpravo vidíme rovnaký súbor údajov po odstránení daných dvanástich odľahlých hodnôt. Výsledok zovšeobecnenia je v tomto prípade oveľa uspokojivejší.

Ak chceme zovšeobecniť množinu dát, odľahlé hodnoty budú pôsobiť ako **rušivý prvok**, a preto sa odporúča nebrať takéto hodnoty atribútov (a záznamy, ktoré ich obsahujú) do úvahy, aj keď sú správne. Ako je možné vidieť na nasledujúcom obrázku – množinu údajov chceme opísať pomocou línie (v skutočnosti lineárnej funkcie). V prípade ľavého podobrázku sa línia odchýlila v dôsledku prítomnosti odľahlých hodnôt (dolný pravý roh uvažovaného priestoru). Po odstránení týchto odľahlých hodnôt môžeme vidieť drastické zvýšenie presnosti tohto zovšeobecnenia (pravý podobrázok).



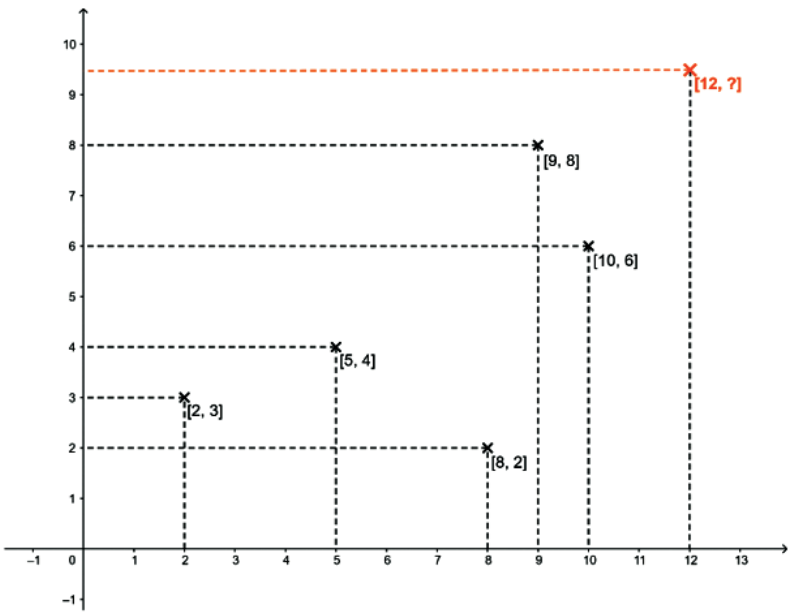
2.2 ANALÝZA DÁT

Analýza dát je činnosť, ktorej cieľom je **získať užitočné znalosti z dát** na podporu informovaného rozhodovania sa o probléme, predikcie udalostí a správania sa vybraných objektov na základe spracovaných údajov. Vieme identifikovať niekoľko typov analýzy dát, no v rámci tejto učebnice nás budú zaujímať len tri základné, **najčastejšie používané typy**:

- **Deskriptívna a diagnostická analýza dát** – najjednoduchšia (a zároveň najčastejšie používaná) metóda analýzy datasetov. Opisná analýza sa zameriava na vyvodzovanie záverov (alebo získavanie znalostí) z dát. Najčastejšie sa používa v kontexte opisu datasetu a merania základných vlastností, ktoré dataset samotný opisuje – napríklad plnenie plánov v organizácii. Diagnostická analýza je zameraná na objasnenie toho, prečo došlo k udalostiam identifikovaným v deskriptívnej analýze. Keďže diagnostická analýza vytvára spojenia medzi jednotlivými hodnotami a preto môže vyť použitá pri identifikácii opakujúcich sa vzorov v správaní sa dátových objektov. Tento typ analýzy dát je založený na vytváraní podrobných informácií, ktoré je možné následne opakovane použiť pri riešení podobných problémov.
- **Exploratívna analýza dát** – pre ľudí je najprirodzenejším typom analýzy exploratívna analýza dát. Zameriava sa na analýzu dát pomocou prieskumu, najčastejšie pomocou vizualizácie dát. Táto analýza je efektívna v kontexte identifikácie vzorcov a závislostí v dátach, ale je dôležitá aj z hľadiska prezentácie výsledkov iných analýz. Okrem vizuálnej stránky prieskumnej analýzy údajov sem zaraďujeme aj činnosti spojené so zjednodušením alebo reprezentáciou datasetu - napríklad redukcia dimenzionality, operácia, pri ktorej premietame n-rozmerný dataset do m-rozmerného datasetu, pričom $m < n$.
- **Prediktívna analýza dát** – prediktívna analýza je rozšírením vyššie uvedených typov analýz. Jej cieľom je použiť zozbierané dáta na vytvorenie logických predpovedí výsledkov udalostí alebo predpovedanie a odhad hodnôt, ktoré v skutočnosti neboli namerané. Pri tomto type analýzy dát sa využívajú metódy modelovania založené na štatistike, čo so sebou prináša potrebu využitia výpočtových technológií na vytváranie predikčných modelov. Je nevyhnutné si uvedomiť, že predikcie, ktoré sú výsledkom modelov vytvorených počas prediktívnej analýzy, sú iba odhadmi pre daný dataset a ich presnosť preto priamo závisí od kvality daných dát.

Všetky tieto typy analýzy údajov bežne pracujú len s dvoma základnými typmi problémov, ktoré je potrebné riešiť – s regresným problémom a klasifikačným problémom. Nasledujúca časť tejto kapitoly je zameraná práve na opis týchto dvoch problémov.

Regresný problém

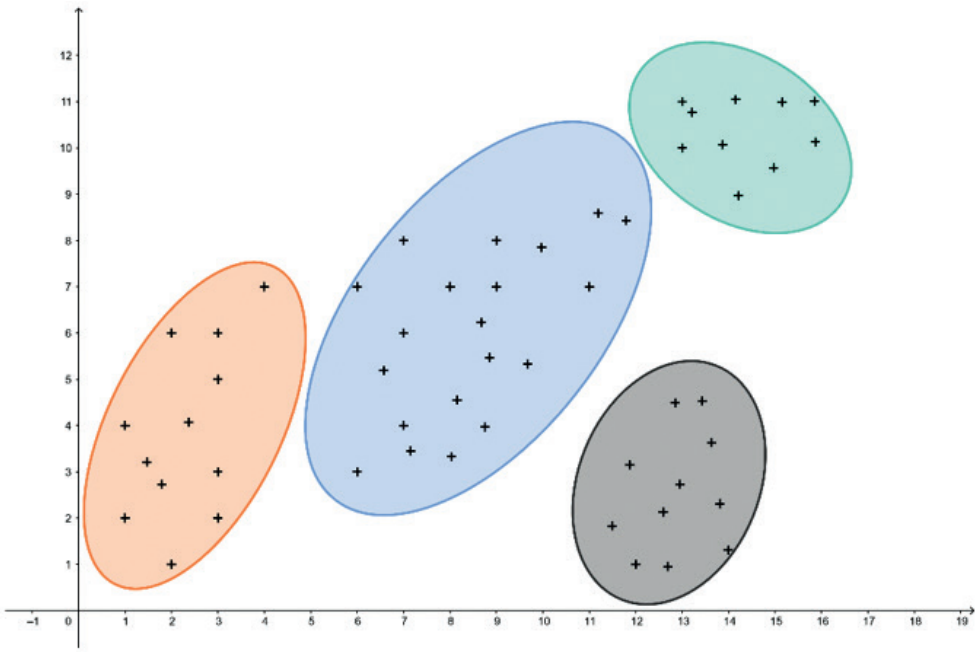


Vo vyššie uvedenom obrázku môžeme vidieť dataset, ktorý obsahuje päť bodov definovaných hodnotami dvoch atribútov – tieto hodnoty sú merané na osiach *x* a *y* a preto ich budeme označovať ako hodnoty atribútu *x* a *y*. Tento dataset pozostáva z bodov [2, 3], [5, 4], [8, 2], [9, 8] a [10, 6].

Regresným problémom v tomto prípade rozumieme úlohu odhadnúť skutočnú hodnotu atribútu $y \in R$, v prípade, že poznáme hodnotu *x* a vzor z predchádzajúcich (v tomto prípade) piatich bodov. Máme teda entitu obsahujúcu hodnotu pre atribút x = 12 a neznámu hodnotu pre atribút *y*, ktorú je potrebné dopočítať.

Vo všeobecnosti môžeme tento typ problému definovať ako odhad alebo predikciu číselnej hodnoty premennej *y* na základe hodnoty premennej *x*, kde $x, y \in R$.

Klasifikačný problém



Všeobecný opis tohto problému môže vyzerať nasledovne: Vzhľadom na vzor *x* a priestor *X* odhadnite, akú hodnotu pridruženého atribútu $y \in \{1,\ldots, n\}$ získa vzor *x*.

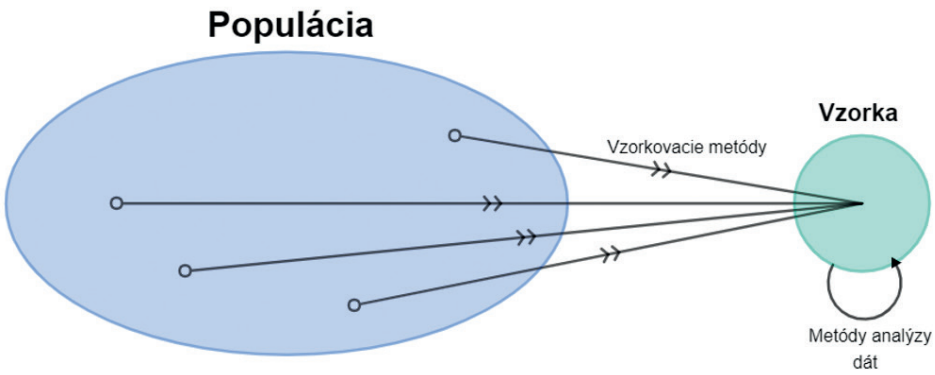
Takýto opis je mierne vágny, zrozumiteľnejšie by mohlo byť, že v probléme klasifikácie priraďujeme entity k množine vopred definovaných tried jednotlivcov, ktorí sú si v určitom zmysle podobní v rámci jednej triedy. Vo všeobecnosti môžeme identifikovať tri typy klasifikačných postupov:

- **Hierarchická klasifikácia**, v ktorej sú triedy samotné klasifikované do skupín a proces je opakovaný na rôznych stupňoch s cieľom vytvoriť stromovú štruktúru.
- **Rozdeľovanie** (z anglického *partitioning*), v ktorom sú jednotlivé triedy disjunktné, čím sú vytvorené partície množiny entít.
- **Zhlukovanie** (z anglického *clumping*), v ktorom je prekryv medzi jednotlivými triedami prípustný, pričom zhluk a jeho doplnok sú vnímané ako dva rôzne typy tried.

KAPITOLA 3

METÓDY VZORKOVANIA DÁT

Autorom tejto časti učebnice je Adam Dudáš z Katedry informatiky, Fakulty prírodných vied, Univerzity Mateja Bela v Banskej Bystrici zo Slovenska.



Vzorkovanie je možné definovať ako výber časti populácie (datasetu), ktorá by mala byť najviac reprezentatívna vzhľadom na tento dataset, aby mohla byť použitá v analýze a získavaní znalostí platných pre populáciu. Technika vzorkovania z populácie je označovaná ako vzorkovacia metóda.

Vzorka je teda definovaná ako **časť alebo zlomok populácie** vybranej takým spôsobom, že je možné urobiť závery o populácii, zatiaľ čo populácia je súhrn všetkých subjektov alebo subjektov, ktoré sú predmetom štúdie.

Poznáme niekoľko známych metód odberu vzoriek. Najčastejšie ich delíme **do dvoch skupín** – pravdepodobnostné a nepravdepodobnostné metódy vzorkovania. Je však dôležité povedať, že typ vzorkovania, ktorý sa má byť použitý pri výbere vzorky, závisí výlučne od riešeného problému. Vo všeobecnosti však môžeme povedať, že:

- **Nepravdepodobnostné metódy** sú závislé od osoby, ktorá vzorku vytvára – je zrejmé, že práve preto je jednoduché získať výsledky, ktoré táto osoba očakáva (aj keď nemusia byť pravdivé pre celú populáciu).
- **Pravdepodobnostné metódy** sa viac-menej vyhýbajú tomuto problému.

3.1 NEPRAVDEPODOBNOSTNÉ METÓDY VZORKOVANIA

Výber vzorky z populácie závisí najmä od úsudku človeka, ktorý vzorku zostavuje – preto tieto metódy môžu viesť k skresleniu niektorých hodnôt v porovnaní s populáciou. Niektoré metódy nepravdepodobnostného vzorkovania dokonca závisia len od pohodlnosti osoby, ktorá vzorku zostavuje – príkladom môže byť metóda vzorkovania s názvom Pohodlné vzorkovanie (z anglického *Convenience sampling*), kde sa členovia populácie vyberajú na základe toho, čo zostavovateľovi najviac vyhovuje. Podobne v metóde nazývanej Vzorkovanie úsudkom (z anglického *Judgement sampling*) sa vzorka zostavuje na základe úsudku zostavovateľa – napríklad na základe mimodátových znalostí osoby, ktorá vzorku zostavuje.

Niekoľko nepravdepodobnostných metód vzorkovania je veľmi jednoduchých a sú len akýmsi vzorkovaním využívajúcim sedliacky rozum. Preto v tejto učebnici ponúkame podrobnejší opis len jednej z nepravdepodobnostných metód výberu vzoriek.

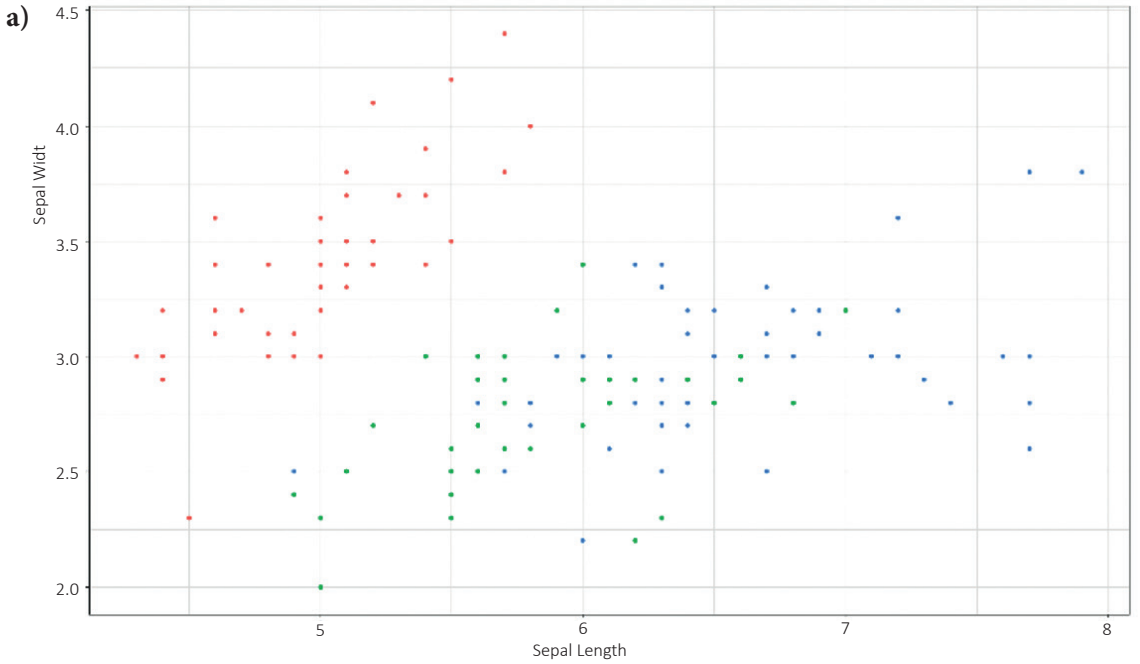
Vzorkovanie so zámerom

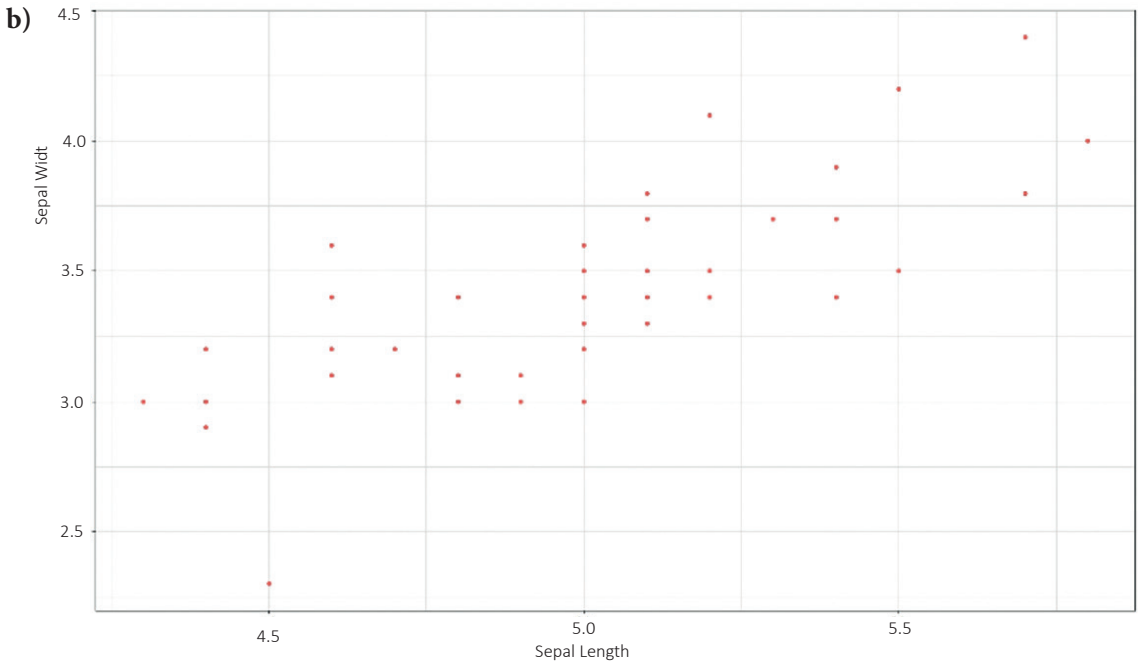
Pri tejto metóde vzorkovania si osoba, ktorá vzorku kompiluje, vyberá členov populácie s konkrétnym účelom, pre ktorý je vzorka zbieraná. Keďže všetci príslušníci populácie nemajú rovnakú šancu na zaradenie do vzorky, hovoríme o nepravdepodobnostnej metóde výberu.

Príkladom takéhoto výberu môže byť potreba zostaviť analýzu študentov tretieho ročníka študijného odboru informatika, teda vytvoriť vzorku študentov tretieho ročníka študentov informatiky z populácie všetkých študentov všetkých ročníkov vo všetkých študijných odborov. Je zrejmé, že do vzorky nebudeme chcieť zahrnúť žiakov prvého, druhého, štvrtého alebo piateho ročníka. Rovnako do vzorky nezaradíme študentov, ktorí študujú v odboroch ako aplikovaná matematika, biológia, či forenzná chémia.

Aby sme opísali výsledky jednotlivých metód odberu vzoriek, budeme od tohto bodu v kapitole používať dataset Iris, ktorý je popísaný v *prílohe A* tejto učebnice. Úloha pre metódu vzorkovania so zámerom by mohla byť nasledovná: *Je potrebné analyzovať hodnoty dĺžky a šírky sepal lístkov pre jeden konkrétny druh kvetu - Iris Setosa.*

Obrázok predstavuje porovnanie medzi hodnotami dĺžky a šírky sepal lístkov v a) kompletnom datasete Iris na ľavom podobrázku (každá trieda kvetu je označená vlastnou farbou) a b) vzorke datasetu pozostávajúceho z jednej triedy - Iris setosa.





3.2 PRAVDEPODOBNOSTNÉ METÓDY VZORKOVANIA

Týmto názvom označujeme metódy, pri ktorých majú všetci členovia uvažovanej populácie rovnaké šance byť zaradení do tvorenej vzorky. Tieto metódy zabraňujú (alebo znižujú) skresleniu výsledkov, ktoré môže byť spôsobené autorom pri pridávaní objektov do vzorky. Existujú rôzne typy pravdepodobnostných metód vzorkovania, ktoré sa používajú v rôznych situáciách na výber vzoriek z rôznych populácií.

Tieto metódy vyžadujú, aby výskumník poznal uvažovanú populáciu a vhodnú metódu vzorkovania, ktorú má použiť, a ako ju použiť v každej situácii, s ktorou sa stretne.

Existuje niekoľko pravdepodobnostných metód vzorkovania – viacstupňové vzorkovanie, zhlukové vzorkovanie, systematické vzorkovanie a podobne. V rámci tejto časti učebnice sa zameriavame na štyri pravdepodobnostné metódy vzorkovania, ktoré sú jednoduché a použiteľné na široké spektrum riešených problémov.

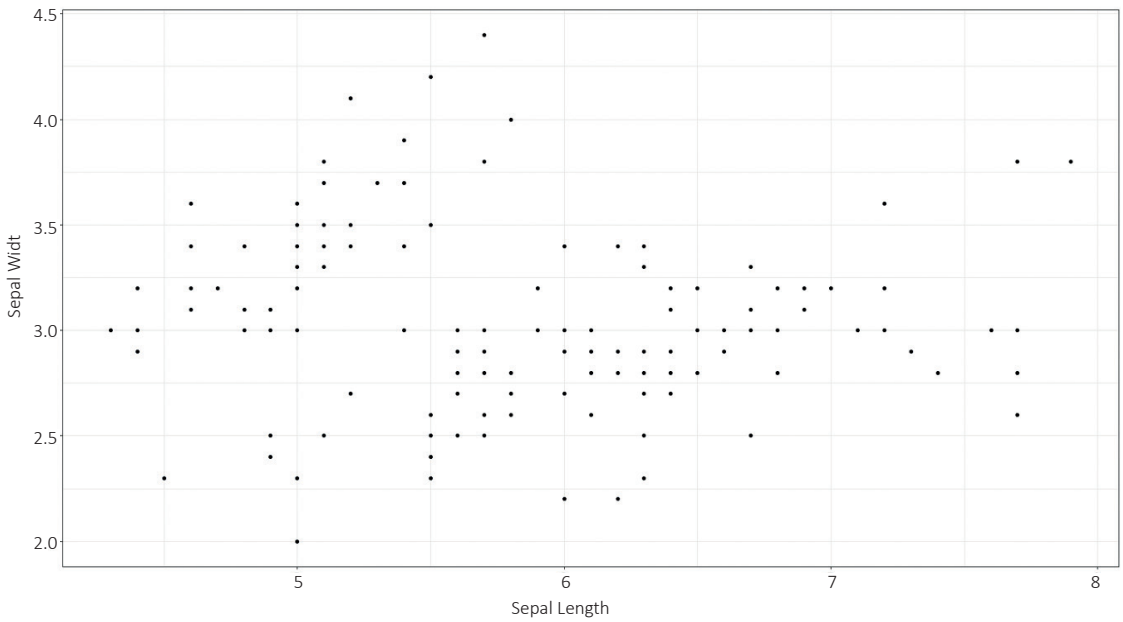
Jednoduché náhodné vzorkovanie

Táto metóda je založená na náhodnom výbere jedincov z populácie. Inými slovami, určitá vzorka je vybraná z akejkoľvek populácie bez akéhokoľvek matematického modelu alebo logického rozhodovania. Keďže každý jednotlivec (záznam) má rovnakú šancu stať sa súčasťou vzorky, je táto metóda najreprezentatívnejšou z pravdepodobnostných metód vzorkovania.

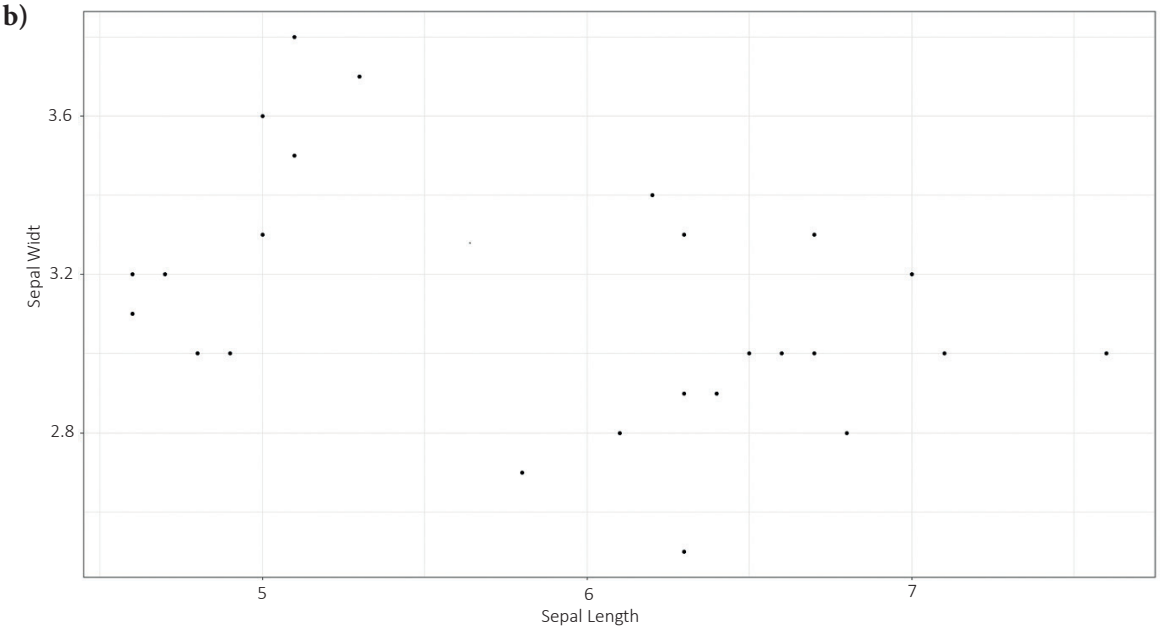
Jednoduchá metóda náhodného výberu vzoriek má len jeden vstupný parameter – požadovanú veľkosť vzorky.

Príklad: Naša populácia (vľavo) obsahuje 150 jedincov (záznamov) a náhodne z nej vyberieme 25 zástupcov (vpravo) - tento súbor pre nás predstavuje jednoduchú náhodnú vzorku.

a)



b)

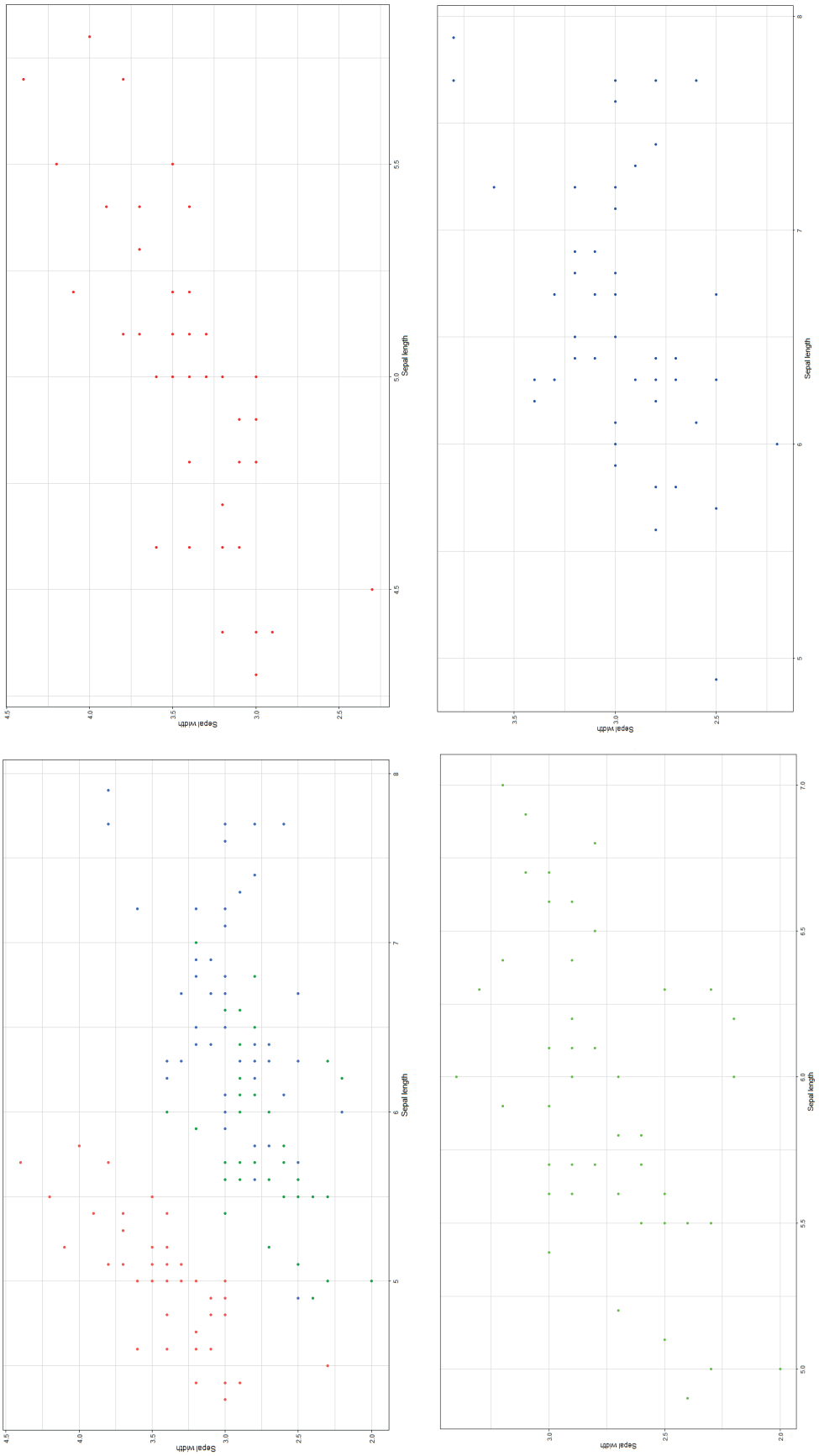


Zhlukové vzorkovanie

Metóda, pri ktorej je celý dataset rozdelený do sekcií alebo zhlukov. Zhluky sú identifikované a zahrnuté do vzorky na základe určitého atribútu, najčastejšie atribútu kategorického, ako farba vlasov, pohlavie a podobne. Táto metóda je použiteľná na vytvorenie vzoriek vhodných na analýzu už existujúcich podmnožín v údajoch.

Metóda zhlukového vzorkovania má len jeden vstupný parameter – atribút, ktorý by sa mal použiť na zhlukovanie údajov.

Príklad: Našu množinu údajov (horný riadok) sme rozdelili podľa atribútu trieda, ktorý má tri hodnoty – iris setosa, iris versicolor a iris virginica. Metódou zhlukového vzorkovania môžeme vytvoriť tri vzorky (spodný rad obrázkov), ktoré je možné použiť na analýzu charakteristík jedincov daných tried. Je zrejmé, že vzorka2 (spodok, v strede) nie je vhodná na vyvodzovanie záverov o celej populácii, iba o podmnožine populácie, ktorej atribút triedy má rovnakú hodnotu ako vzorka2. Vhodným využitím tejto metódy je napríklad príprava štatistickej analýzy opisujúcej jednotlivé zhluky, ktorá umožní porovnanie charakteristík tried kvetov v datasete.


Sepal width
Sepal length
Sepal width
Sepal length
Sepal width
Sepal length
Sepal width
Sepal length

Systematické vzorkovanie

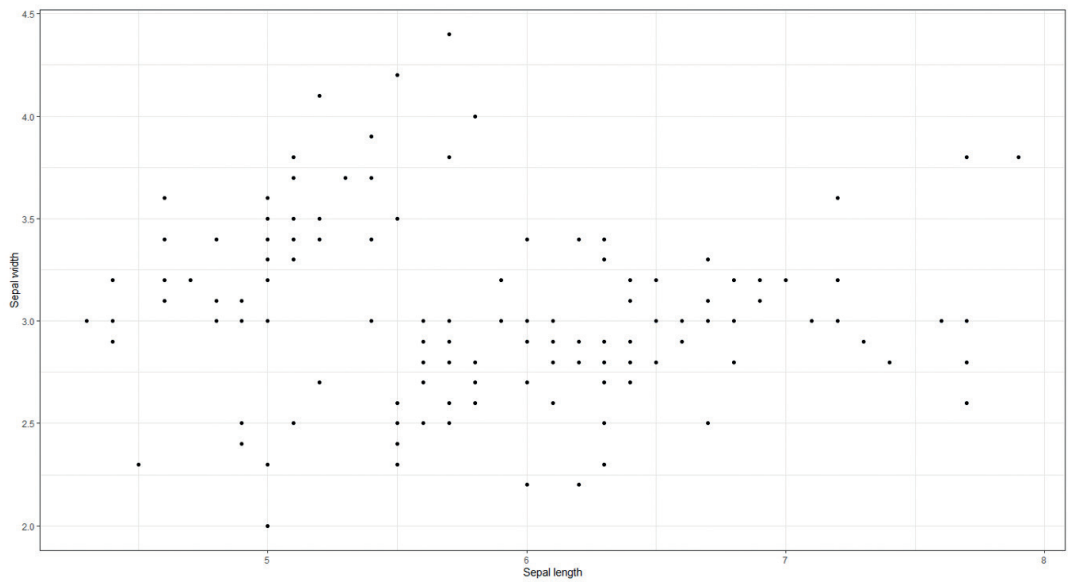
Táto metóda sa používa pri výbere členov vzorky z populácie v pravidelných intervaloch. Tento typ metódy odberu vzoriek má vopred definovaný rozsah, a preto je to technika odberu vzoriek, ktorá je najmenej časovo náročná.

Metóda systematického odberu vzoriek má dva alebo tri vstupné parametre:

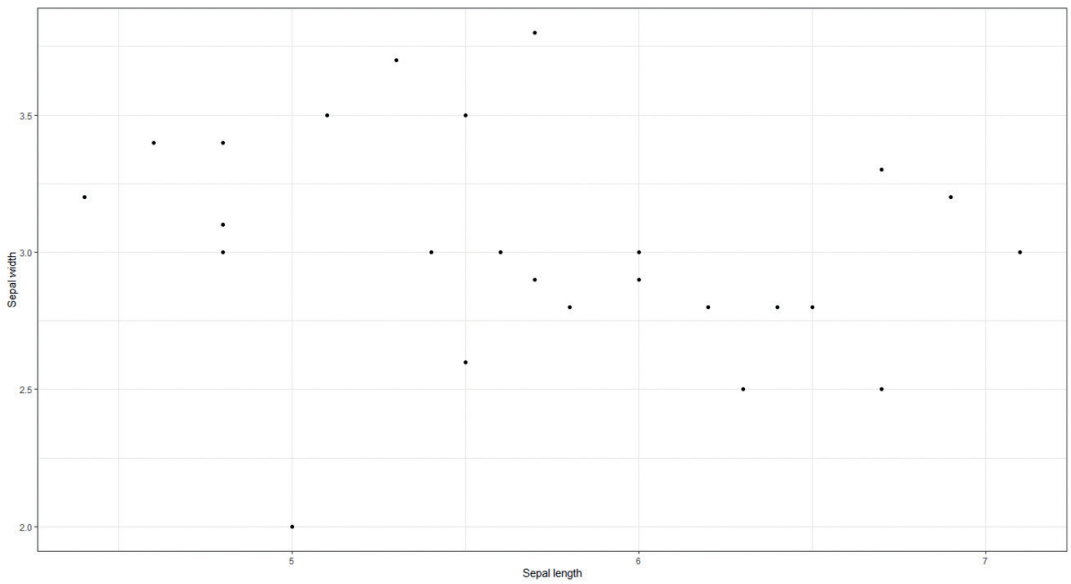
- vybraný východiskový bod pre vytvorenie vzorky (prvý jedinec, ktorý patrí do vzorky),
- interval, v ktorom sú jednotlivci pridávaní do vzorky, čo vytvára implicitnú veľkosť vzorky,
- alebo interval, v ktorom sú jednotlivci pridávaní do vzorky a veľkosť vzorky, ktorú vytvárame.

Príklad: Keďže pomocou metódy jednoduchého náhodného výberu sme vybrali 25 jedincov, ktorí predstavovali populáciu, ktorú sme použili; chceme tiež použiť metódu systematického výberu vzorky na vytvorenie vzorky 25 jedincov. Pôvodný súbor pozostáva zo 150 zástupcov a keďže 150/25 = 6 vyberieme každého šiesteho jedinca (v prípade, že vychádzame z prvého záznamu v datasete). Na obrázku nižšie môžeme vidieť celú populáciu (vľavo) a vzorku (vpravo) pozostávajúcu z 25 jedincov, ktorí boli vybraní vyššie popísaným postupom.

a)



b)



Stratifikované vzorkovanie

Pri stratifikovanej metóde vzorkovania je celý dataset rozdelený do menších oddelených skupín, ktoré predstavujú novú populáciu. V porovnaní s metódou zhlukového vzorkovania táto metóda vytvára skupiny v dátach pomocou novo definovanej hranice jedného z atribútov prítomných v pôvodnom datasete. Metóda zhlukového vzorkovania nevytvára tieto hranice, ale používa jeden z (kategorických) atribútov na identifikáciu skupín v údajoch.

Príklad: Vzorky vytvorené pomocou metódy stratifikovaného vzorkovania na obrázku nižšie možno definovať ako intervaly definované v atribúte sepal_length. Každá vzorka je iná, ale v každej vzorke sú zástupcovia, ktorých hodnota atribútu sepal_length je z pohľadu zvolenej metódy podobná. V našom prípade sme vzorky rozdelili po 1 cm od najmenšej po najväčšiu:

$$sepal\_length \in (4, 5]$$
$$sepal\_length \in (5, 6]$$
$$sepal\_length \in (6, 7]$$
$$sepal\_length \in (7, 8]$$

Obrázok teda obsahuje štyri vzorky oddelené farbou - vzorka1 označená červenou farbou, vzorka2 označená zelenou farbou a podobne.

3.3 PÁR SLOV NA TÉMU KVALİTY VZORKY

Správne vzorkovanie je jednou z nevyhnutných techník používaných pri práci s veľkými údajmi (pre referenciu pozri časť 1.1). Vyššie uvedené metódy vytvárajú vzorky, ktorých kvalitu je možné hodnotiť z viacerých hľadísk. Pre účely tejto učebnice uvádzame iba dva kritériá na opis kvality vzorky a iba jedno z nich je skutočne kritické pre bežných používateľov (hlavne používateľov mimo oblasti informatiky):

- **Rýchlosť odberu vzorky** – v dnešnom svete používame moderný softvér, ktorý často obsahuje optimalizované funkcie a balíky. Medzi takéto funkcie patria aj metódy vzorkovania, ktorých implementácia vo vybranom nástroji prešla optimalizáciami a technikami, ktoré zvýšili efektivitu danej funkcie (toto je takmer zaručené). Ak je potrebné vytvoriť vzorku zo štandardne veľkého súboru dát (nie veľkých dát), používateľ sa nedostane do kontaktu s problémom nedostatočného výkonu systému, ktorý by mohol mať za následok zdĺhavé vytváranie vzorky (alebo vo fatálnych prípadoch, nemožnosť vytvorenia vzorky). Keď však pracujeme so skutočnými veľkými datasetmi, štandardný systém prestáva byť dostatočne efektívny. Z vlastnej skúsenosti môžeme uviesť príklad vytvorenia vzorky na množine údajov (populácii) s veľkosťou sto miliónov záznamov, pričom každý záznam obsahoval šestnásť atribútov (všimnime si, že nejde o tak veľký dataset). Pri použití metódy systematického vzorkovania so vstupným parametrom 4 (pre vytvorenie veľkosti vzorky 25 % populácie) v jazyku R na štandardnom používateľskom počítači sa nám vzorku vytvoriť nepodarilo. Tento problém možno vyriešiť niekoľkými spôsobmi, z ktorých najbežnejším je použitie vysokovýkonných výpočtových metód alebo metód cloudového počítania.
- **Reprezentatívnosť vzorky** – problémom, ktorý je pre hodnotenie kvality vzorky dôležitejší ako rýchlosť zberu vzoriek, je schopnosť vzorky opísať populáciu, z ktorej bola vytvorená. Rovnako ako v predchádzajúcom prípade je zrejmé, že takáto metrika nebude univerzálna - ako bolo uvedené v opise metódy zhlukového vzorkovania vyššie, vzorka jedného zhluku (jednej konkrétnej podmnožiny údajov) nie je vhodná na vyvodenie záveru o celej populácii. V prípade, že je vhodné porovnať charakteristiky vzorky a populácie, môžeme postupovať niekoľkými spôsobmi podľa našich cieľov:
  - **Štatistický opis vzorky** – v prípade, že chceme dáta opísať pomerne malým počtom hodnôt, môžeme vypočítať kritické štatistické metriky. Z týchto číselných hodnôt vieme odvodiť poznatky vhodné pre ďalšiu prácu s dátami.
  - **Vizualizácia vzorky** – veľké dáta sú povestné zložitosťou ich vizualizácie. Preto je dobré vytvoriť reprezentatívnu vzorku, ktorá obsahuje menej jedincov, a preto je jednoduchšie vizualizovateľná.
  - **Analýza predikčného potenciálu vzorky** – ak je naším cieľom zostaviť predikčné alebo odhadovacie modely založené na strojovom učení sa, je vhodné analyzovať predikčný potenciál jednotlivých atribútov pomocou metód ako korelačná analýza alebo pomocou modelu rozhodovacích stromov.

  Všetky tieto prístupy sú podrobnejšie opísané v časti 4 tejto učebnice.

3.4 PÁR SLOV NA TÉMU VEĽKOSTİ VZORKY

V prípade, že potrebujeme zostaviť vzorku z danej populácie, môže nastať problém s identifikáciou potrebnej veľkosti tejto vzorky, pre dosiahnutie požadovaných výsledkov – aby sme vedeli presne odvodiť potrebné znalosti. Odpoveď na túto otázku závisí od použitej metódy a od našich cieľov:

- V prípade metód vzorkovania, ako je zhlukové vzorkovanie alebo stratifikované vzorkovanie, je odpoveď daná použitou metódou. Tieto metódy vytvárajú vzorky, ktorých veľkosť je definovaná výskytom určitej hodnoty v dátach, a preto v tomto prípade nie je štandardne uvažované o inej veľkosti vzorky ako je veľkosť zhluku identifikovaného metódou.
- V iných prípadoch, najmä pri náhodnom výbere, je potrebné použiť model, ktorý identifikuje veľkosť vzorky vhodnú pre naše potreby. Tento model je štandardne definovaný pre dva typy populácií – populácie s obmedzeným počtom jedincov alebo populáciu bez obmedzenia počtu jedincov. Pre naše potreby zvážime prirodzenejšiu z týchto verzií – obmedzenú populáciu:

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

kde

- $\bar{n}$ je veľkosť vzorky,
- z je takzvané z-skóre, ktoré odráža úroveň spoľahlivosti, najčastejšie nastavené na 90%, 95% alebo 99% s prislúchajúcimi koeficientmi z-skóre s hodnotou, aká je uvedená v nasledujúcej tabuľke:

Úroveň spoľahlivosti	z-skóre
90%	1.65
95%	1.96
99%	2.58

Táto tabuľka je typickým príkladom tabuliek z-skóre, ktoré obsahujú vopred vypočítané hodnoty z-skóre. Internet ponúka možnosť vyhľadať podobné tabuľky pre ďalšie hodnoty úrovne spoľahlivosti a prislúchajúce z-skóre.

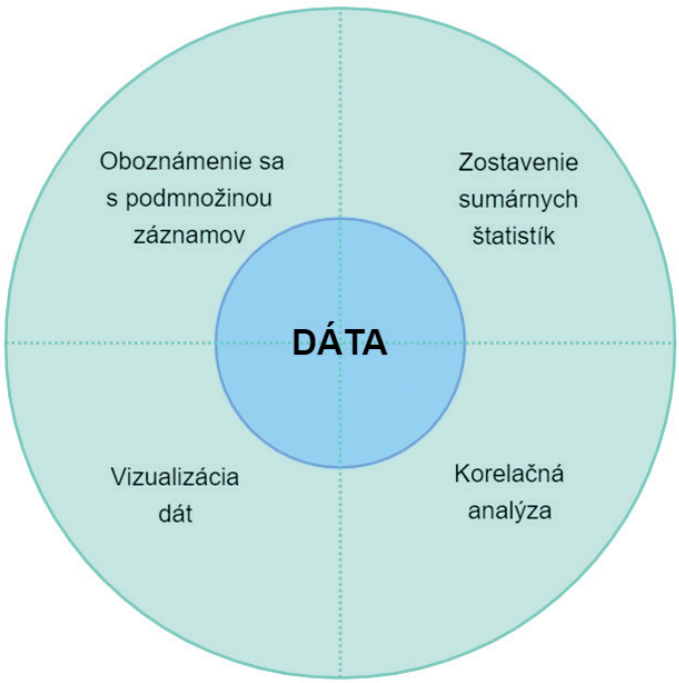
- p (p with the line above) je podiel populácie - percento (alebo zlomok) populácie spojenej so skúmaným problémom (štandardná hodnota pre neznámu populáciu je nastavená na $p = 0.5$).
- ε je hranica chyby nastavená používateľom.
- N je veľkosť použitej populácie.

Najjednoduchším spôsobom výpočtu veľkosti vzorky je použitie voľne dostupných online kalkulačiek veľkosti vzorky (*sample size calculators*), ktoré fungujú na princípoch uvedených vyššie.

KAPITOLA 4

ZÁKLADY EXPLORATÍVNEJ ANALÝZY DÁT

Autorom tejto časti učebnice je Adam Dudáš z Katedry informatiky, Fakulty prírodných vied, Univerzity Mateja Bela v Banskej Bystrici zo Slovenska.



Ako bolo uvedené v predchádzajúcich častiach tejto učebnice, v procese analýzy dát môžeme pracovať s celým datasetom alebo so vzorkou vytvorenou pomocou metód uvedených v časti 3 tohto textu. Na základnú, deskriptívnu analýzu dát používame metódy deskriptívnej štatistiky, ktoré poskytujú nástroje na zachytenie charakteristík datasetu alebo vzorky dát. Metódy deskriptívnej štatistiky sú založené na agregačných metódach na reprezentáciu podmnožín údajov, napríklad priemernej hodnoty atribútu, minima, frekvencie alebo súčtu hodnôt. Takýto prístup môžeme označiť názvom agregácia ako metóda redukcie dát.

Pri analýze dát pomocou metód deskriptívnej štatistiky používame najmä nasledovné tri koncepty:

- **Miery centrálnej tendencie** slúžia na identifikáciu centrálnych bodov v dátach. Tieto body sú identifikované tým, že v ich okolí sú dáta blízko zoskupené alebo distribuované.
- **Miery variability** sú miery, ktoré opisujú rozloženie dát v uvažovanom priestore, t.j. ako ďaleko sú jednotlivé merania od centra identifikovaného pomocou mier centrálnej tendencie.
- **Korelačná analýza** je založená na výpočte koeficientov, ktoré opisujú predikčný potenciál medzi jednotlivými atribútmi v datasete. Tieto koeficienty sú nevyhnutné pri vytváraní modelov strojového učenia sa, ale aj pri vizualizácii dát, ktorá je podstatnou súčasťou tejto časti učebnice.

V tejto časti učebnice opíšeme úvod do prvej verzie analýzy dát, ktorá je z ľudského hľadiska veľmi prirodzená – **Exploratívna Dátová Analýza** (EDA). Ako už samotný názov napovedá, ide o analýzu dát pomocou prieskumu s cieľom nájsť vzory a trendy v danej populácii alebo vzorke. Vo svojej najzákladnejšej forme sa tento typ analýzy vykonáva vizuálnym prieskumom, a preto budú metódy vizualizácie dát dôležitou súčasťou takejto analýzy.

Aby sme vedeli, ktoré časti nášho datasetu je vhodné vizualizovať a ktoré nie, využívame základné poznatky získané prostredníctvom metód deskriptívnej štatistiky.

4.1 ZÁKLADNÉ ŠTATISTICKÉ METÓDY

Z pohľadu základných štatistických metód môžeme identifikovať metódy, ktorými meriame centrálnosť v dátach – hľadáme dátové centrá, okolo ktorých sú dáta zhlukované alebo husto distribuované. Najbežnejšie z týchto metód sú:

- **Priemer** je priemerná hodnota prvkov poľa. Priemer je vhodný na charakterizáciu symetricky rozdelených datasetov bez odľahlých hodnôt (napr. výška, hmotnosť). Symetricky rozdelené dáta sú tie, kde by mal počet prvkov datasetu byť podobný pod a nad strednou hodnotou, ideálne rovnaký. Vzťah na výpočet priemernej hodnoty:

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

kde μ je priemerná hodnota atribútu A, n je počet entít, ktoré obsahujú meranie pre atribút A a A_i je i-ta hodnota tohto atribútu.

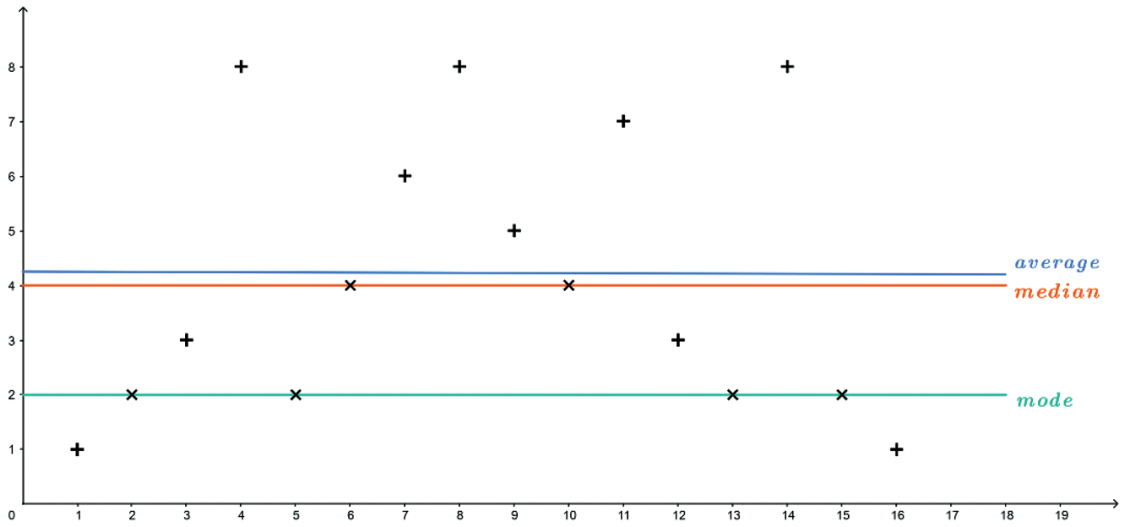
- **Medián** je stredná hodnota zoradeného poľa. Medián je extrémne symetrický, čo znamená, že pod mediánom je rovnaký počet prvkov ako nad ním. Výnimkou z tohto pravidla sú datasety obsahujúce párny počet prvkov (ako medián volíme jednu z dvoch stredných hodnôt– v prirodzene zostavenom datasete by mali byť tieto hodnoty dostatočne blízko seba). Na rozdiel od priemeru je medián reálnou hodnotou atribútu, preto je vhodnejší, ak dáta obsahujú odľahlé hodnoty alebo sú asymetricky distribuované (napríklad mzdy zamestnancov v určitej oblasti). Medián je definovaný nasledovným vzťahom:

$$median_A = \frac{(n+1)}{2} \text{ ty prvok zoradeného poľa } A,$$

kde n je počet entít obsahujúcich hodnotu pre atribút A.

- **Modus** je najčastejšie sa vyskytujúci prvok atribútu. Táto miera je však ťažko použiteľná a vo väčšine analytických úloh nie je relevantná, resp. presná. Príkladom môže byť modus vyššie spomínaných miezd, ktorý by bol vo väčšine prípadov rovný 0, keďže práve 0 zarába väčšina ľudí – nezamestnaní, deti, dôchodcovia a podobne.

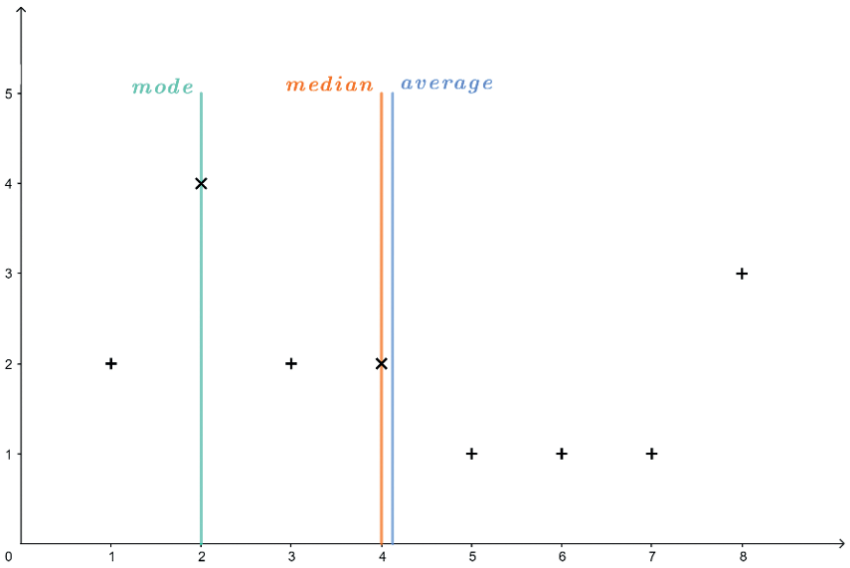
Na nasledujúcom obrázku ponúkame vizualizáciu mier centrality pre jednoduchý dataset. Môžeme pozorovať pomerne typické správanie sa hodnôt priemeru a mediánu, ktoré sú normálne distribuované v uvažovanom priestore – teda tieto hodnoty sú navzájom pomerne blízko seba. Keďže je modus najčastejšie sa vyskytujúca hodnota prvku atribútu, je komplikované ju predpovedať (môže byť vysoká, môže byť nízka, môže byť niekde v strede).



Okrem týchto štandardných mier je veľmi dôležitá **frekvencia hodnôt** v atribúte. Pod názvom **frekvenčné rozdelenie** rozumieme zoznam, tabuľku alebo graf, ktorý zobrazuje frekvenciu výskytu rôznych hodnôt vo vzorke (datasete). Každý záznam v tabuľke obsahuje frekvenciu (alebo počet) výskytov hodnôt v danej skupine alebo intervale. Príklad takéhoto rozdelenia frekvencie pre jednoduchý dataset použitý vyššie vyzerá takto:

hodnota	frekvencia
1	2
2	4
3	2
4	2
5	1
6	1
7	1
8	3

Túto tabuľku možno namapovať do nižšie uvedeného **grafu**. Takáto vizualizácia frekvenčných grafov je nevyhnutná najmä z pohľadu oboznamovania sa s dátami a pri prípadnej detekcii odľahlých hodnôt.



Na druhej strane je pri štandardných štatistických meraniach podstatná aj variabilita údajov v uvažovanom priestore. Najbežnejšou mierou variability je takzvaná **štandardná odchýlka** (σ), ktorá je definovaná ako súčet druhých mocnín rozdielov medzi jednotlivými prvkami atribútu a jeho priemernou hodnotou:

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

kde je priemerná hodnota atribútu A, n je počet entít obsahujúcich atribút A a Ai je hodnota i-teho merania tohto atribútu.

Podobnou mierou je aj **rozptyl** počítaný ako:

$$V = \sigma^2$$

Príklad: Majme nasledovný jednoduchý dataset pozostávajúci z jedného atribútu spiatimi meraniami A = [20, 60, 40, 70, 50]. Vypočítajme priemer, medián a štandardnú odchýlku pre tento dataset.

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

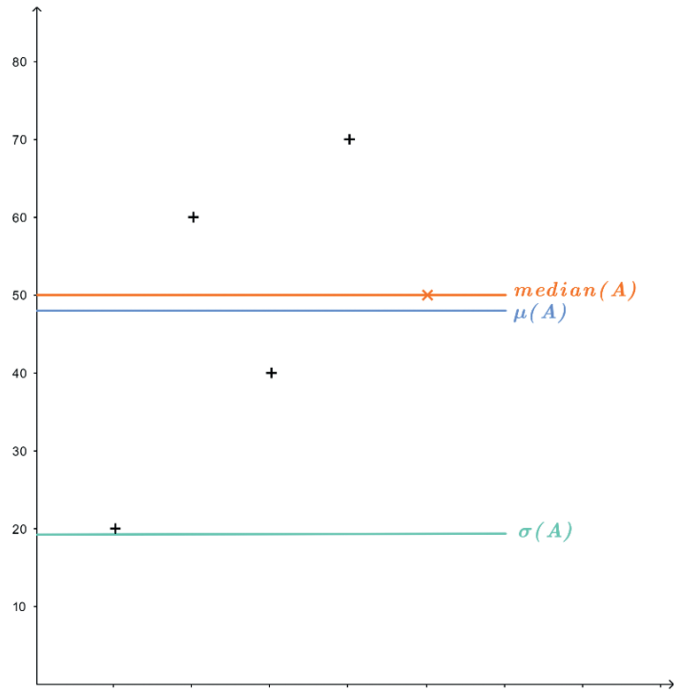
$$median_A = \frac{(n+1)}{2} = \frac{6}{2} = 3\text{-tí}$$ *prvok zoradeného poľa A.* → *[20,40,50,60,70] = 50*

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

$$= \frac{\sqrt{(20-48)^2 + (60-48)^2 + (40-48)^2 + (70-48)^2 + (50-48)^2}}{4} = \frac{\sqrt{1480}}{4}$$

$$\sqrt{370} \approx 19.235$$

Táto štandardná odchýlka je pomerne vysoká, čo je prirodzené, keďže dataset samotný je veľmi rozptýlený. Vizualizácia týchto meraní je uvedená v obrázku nižšie.



Metódy výpočtu mier centrality a variability slúžia na opis datasetu pomocou malej skupiny hodnôt. Takýto prístup k opisu datasetu môžeme nazvať aj **opis agregáciou**.

Príklad: Majme atribút A = [1, 2, 3, 8, 2, 4, 6, 8, 5, 4, 7, 3, 2, 8, 2, 1] (uvedený na začiatku časti 4.1) . Tento dataset môžeme opísať pomocou troch agregovaných hodnôt, napr. (min(A), μ(A), max(A)), a teda A = (1, 4.125, 8).

Posledným ukazovateľom patriacim do jednoduchých štatistických metrík je **distribúcia dát** v priestore. Táto metrika je charakterizovaná použitím priemernej hodnoty a štandardnej odchýlky vybraného atribútu. V normálne distribuovanom datasete sa aspoň $(1 - 1/k^2)$-tina bodov nachádza vo vzdialenosti $k\sigma$ alebo menej od priemeru. Takýto dataset neobsahuje odľahlé hodnoty a je vynikajúcim kandidátom na metódy strojového učenia sa a analýzy dát pomocou metód umelej inteligencie.

Príklad: Pre náš dataset A = [20, 60, 40, 70 ,50] môžeme distribúciu vypočítať nasledovným spôsobom:

$\mu A = 48$

$\sigma A \approx 19.235$

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

Môžeme vidieť, že aspoň tri hodnoty datasetu by mali byť vo vzdialenosti najviac 38.47 od priemernej hodnoty (48). Toto je pravdou pre všetky namerané hodnoty pre atribút A.

4.2 KORELAČNÁ ANALÝZA

Základné štatistické hodnoty uvedené v predchádzajúcej časti textu sú dôležitými ukazovateľmi, pomocou ktorých môžeme opísať dataset, s ktorým pracujeme. Z hľadiska cieľov analýzy dát je však oveľa silnejšou metrikou takzvaná korelačná analýza.

V prípade, že náš dataset obsahuje viac ako jeden číselný atribút, môžeme merať koreláciu medzi dvojprvkovými podmnožinami tohto datasetu. Majme dva atribúty dátovej množiny $A - A_1, A_2$. Tieto atribúty navzájom korelujú, keď má atribút A_1 **prediktívny potenciál** pre atribút A_2. Takýto prediktívny potenciál hovorí o **prítomnosti trendov a vzorov** v datasete a možnosti budovania analytických modelov, ktoré d týmito dátami pracujú.

Koreláciu dvoch premenných meriame pomocou korelačného koeficientu $r(A_1, A_2)$, ktorý vyjadruje, nakoľko je atribút A_1 funkciou atribútu A_2 a naopak. Tento korelačný koeficient môže nadobúdať hodnoty z intervalu $[-1, 1]$, pričom:

- **1** označuje **úplnú koreláciu** dvoch atribútov. Inými slovami, keď narastá hodnota atribútu A_1, narastá aj hodnota atribútu A_2. Úplná korelácia medzi hodnotami dvoch premenných naznačuje prítomnosť silného predikčného potenciálu, z ktorého vyplýva, že tieto atribúty sú vhodné na vzájomnú predikciu.
- **0** označuje z pohľadu korelácie dvoch hodnôt najhoršiu možnú situáciu, ktorú označujeme názvom **nekorelácia**. V prípade, že korelačný koeficient medzi dvoma atribútmi je blízky alebo rovný 0, ide o nezávislé hodnoty, ktoré sú z hľadiska budovania analytických modelov nepoužiteľné.
- **-1** je opakom úplnej korelácie, ktorý nazývame **antikorelácia**. V tomto prípade môžeme identifikovať trend, pri ktorom so zvyšujúcou sa hodnotou atribútu A_1, hodnota atribútu A_2 klesá, alebo naopak. Rovnako ako v prípade úplnej korelácie je to uspokojivá podmienka pre zostavenie analytických modelov.

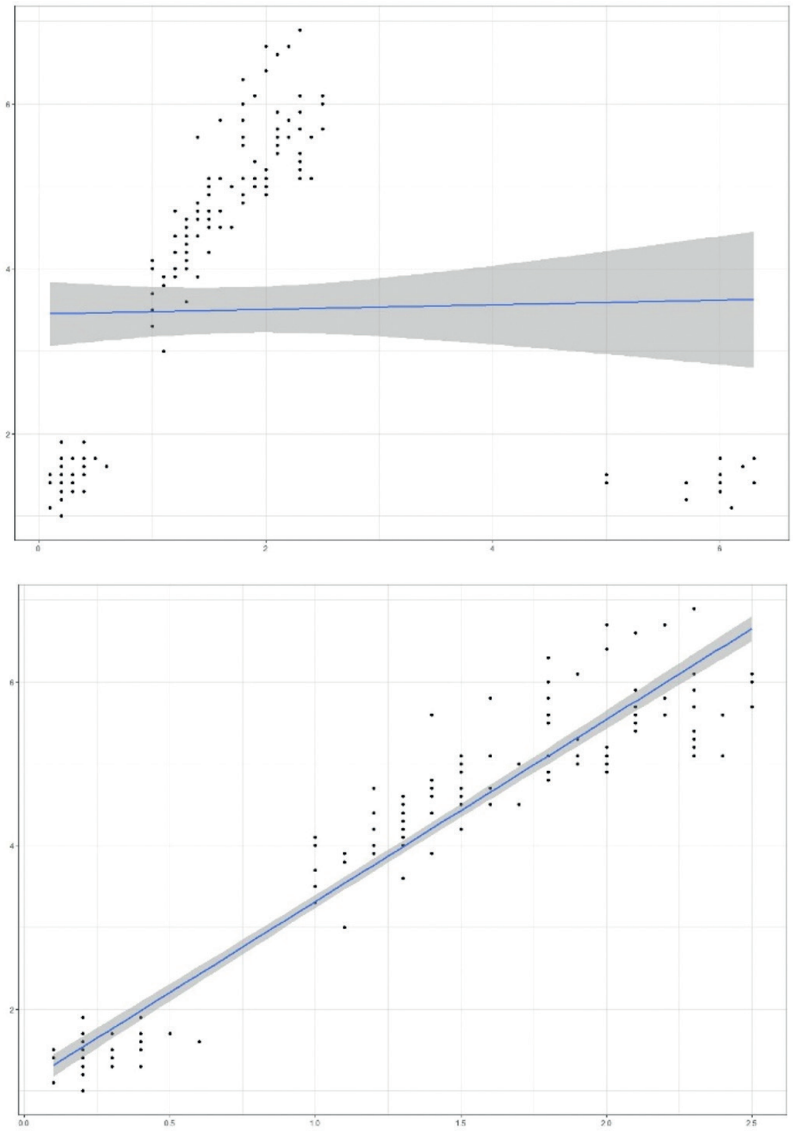
Na analýzu korelácií a meranie korelačných koeficientov používame dve štandardné metódy – aj keď je takýchto metód, samozrejme, viac. Pre naše účely sa zameriame na Pearsonov korelačný koeficient a Spearmanov korelačný koeficient.

Pearsonov korelačný koeficient

Prvý a najsilnejší koeficient, ktorý sa používa na meranie korelácie medzi dvoma atribútmi datasetu, je Pearsonov korelačný koeficient. Tento koeficient je zameraný na **lineárnu predikciu hodnôt** a vzťah medzi atribútmi *A* a *B*. Je opísaný pomocou vzťahu:

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

kde $\mu(A)$ je priemerná hodnota atribútu A, podobne $\mu(B)$ je priemerná hodnota atribútu B a n je počet meraní (vertikálna veľkosť datasetu). Táto zjavná závislosť od priemernej hodnoty prináša najväčšiu nevýhodu Pearsonovho korelačného koeficientu – citlivosť na odľahlé hodnoty (ako je znázornené na obrázku nižšie).



Pomocou Pearsonovho korelačného koeficientu hľadáme líniu, ktorá opisuje hodnoty daných atribútov. Na ľavom podobrázku môžeme vidieť vizualizáciu porovnania hodnôt dvoch atribútov z jedného datasetu, v ktorom sú odľahlé hodnoty (dole, vpravo). Môžeme tiež vidieť, že línia, ktorou sme dáta opísali, ju úplne míňa (až na jeden dátový bod) – z toho môžeme usúdiť, že Pearsonov korelačný

koeficient nie je vhodný na meranie predikčného potenciálu v takomto datasete. Na pravej strane uvádzame rovnaké atribúty datasetu po odstránení odľahlých hodnôt. Vidíme, že v tomto prípade línia opisuje trendy prítomné v dátach.

Práve preto môže byť **Pearsonov korelačný koeficient** použitý keď atribúty *A* a *B* obsahujú:

- lineráne vzťahy,
- normálnu (Gaussovu) distribúciu,
- žiadne odľahlé hodnoty.

Spearmanov korelačný koeficient

Ako spôsob práce s datasetmi, ktoré obsahujú nelineárne vzťahy s odľahlými hodnotami, môžeme použiť iný typ korelačného koeficientu – konkrétne Spearmanov korelačný koeficient. Tento spôsob merania korelácie medzi atribútmi vytvára pre svoju funkčnosť **hierarchiu** (resp. poradie) jednotlivých hodnôt atribútov.

Príklad: Majme atribút A = [a0 = 4, a1 = 8, a2 = 2, a3 = 6]. Vyššie spomínaná hierarchia resp. poradie by v tomto prípade vyzeralo nasledovne:

keďže a1 > a3 > a0 > a2, tak poradie(a1) = 1, poradie(a2) = 4 a tak ďalej.

Týmto spôsobom meriame **monotónnosť hodnôt v rámci atribútu**, a preto môžeme povedať, že Spearmanov koeficient korelácie je najvhodnejší pre datasety s monotónnymi vzťahmi medzi atribútmi – ak sa jedna hodnota atribútu stúpa, druhá nikdy neklesne resp. naopak. Na druhej strane sa tento typ korelačného koeficientu neodporúča používať, ak sa v datasete vyskytujú opakujúce sa hodnoty (znamená hodnoty s rovnakým poradím). Tento efekt sa zmierňuje so zvyšujúcou sa veľkosťou datasetu.

Spearmanov koeficient korelácie je definovaný nasledovne:

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

kde $d = poradie(a_i) - poradie(b_i)$ a n je počet entít, v ktorých sú uvažované atribúty merané.

Príklad: Upozornenie – nasledujúci príklad je medzi študentmi obľúbený; je možné, že si budú pamätať zadanie príkladu viac, ako samotné idey korelácie. Majme dva atribúty – cenu kebabu a vzdialenosť predajne kebabov od univerzity.

Index	Vzdialenosť v metroch	Cena v €
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

Najprv vypočítame hodnotu Spearmanovho koeficientu korelácie. Táto metóda vyžaduje vytvorenie poradia pre oba atribúty a výpočet hodnôt d a d^2 ako:

Index	Vzdialenosť v metroch	Poradie(vzdialenosť)	Cena v €	Poradie(cena)	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

Hodnoty z tejto tabuľky je preto možné pridať do vzťahu pre výpočet Spearmanovho koeficientu korelácie:

$$\sum d^2 = 256$$

$$n = 10$$

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

Zistili sme teda, že medzi týmito dvoma atribútmi existuje korelácia -0,55, čo by sa dalo nazvať stredne silnou antikoreláciou.

Vypočítajme aj hodnotu Pearsonovho korelačného koeficientu. Na výpočet tohto typu koeficientu potrebujeme určiť priemernú hodnotu vzdialenosti μ(vzdialenosť) a priemernú hodnotu ceny kebabu μ(cena):

- *μ(vzdialenosť) = 203, budeme používať označenie μ(v),*
- *μ(cena) = 3.39, budeme používať označenie μ(c).*

Naša tabuľka bude obsahovať viac stĺpcov, ale všetky sú to len predbežné výpočty častí potrebných v konečnom vzťahu Pearsonovho korelačného koeficientu[1]*:*

index	d	p	d – μ(d)	p – μ(p)	$(d - \mu(d))^2$	$(p - \mu(p))^2$	(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

Pearsonov korelačný koeficient teda môžeme vypočítať ako:

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

$$\sum((\mathrm{p} - \mu(\mathrm{p}))^2) = 1.3905$$

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

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

Zistili sme teda, že medzi týmito dvoma atribútmi existuje Pearsonova korelácia rovná -0,44, čo by sme mohli nazvať aj stredne silnou antikoreláciou. Viac o interpretácii výsledkov korelačných koeficientov je možné nájsť na konci tejto časti učebnice.

Korelačná matica a tepelná mapa

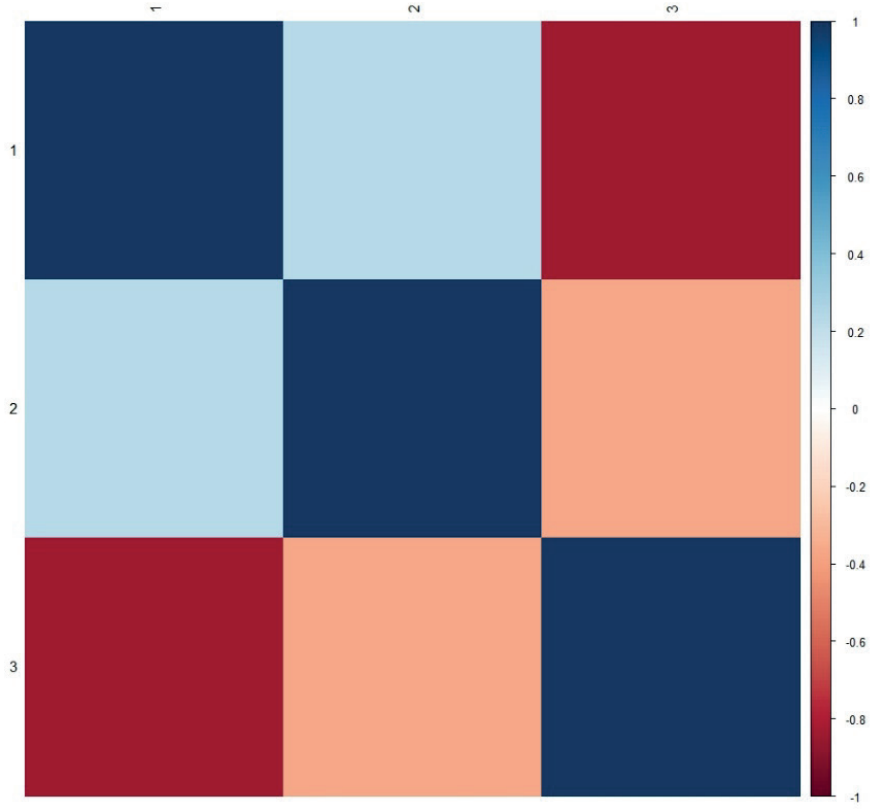
Datasety zriedka obsahujú iba dva atribúty, z čoho vyplýva potreba merať korelačné koeficienty medzi všetkými jeho atribútmi. Na tieto účely používame korelačnú maticu – tabuľku obsahujúcu korelačný koeficient nameraný medzi všetkými možnými pármi atribútov datasetu. V tabuľke nižšie vidíme korelačný koeficient meraný medzi atribútmi A_1, A_2, A_3 (korelačná matica), pričom vidíme, že $r(A_1, A_2) = 0{,}238$, $r(A_1, A_3) = -0{,}834$ a tak ďalej.

[1] Pre nedostatok miesta v predloženej tabuľke používame v pre vzdialenosť a c pre cenu kebabu.

	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

Táto matica má **dve prirodzené vlastnosti** - je symetrická po diagonále a táto diagonála vždy obsahuje hodnoty korelačného koeficientu rovné 1 - korelácia atribútu A_1 so sebou samým je vždy $r(A_1, A_1) = 1$ bez ohľadu na to, aká metóda bola na výpočet použitá, čo je tiež prirodzené, keďže hodnota atribútu A_1 je plne závislá od hodnoty atribútu A_1.

Takáto metóda korelačnej analýzy je vhodná len do istého počtu atribútov v skúmanom datasete. Je zrejmé, že pre dataset obsahujúci desiatky atribútov by takáto matica bola neprehľadná a ťažko čitateľná. Preto sa často nahrádza takzvanou **korelačnou tepelnou mapou** alebo **korelačným grafom**. Pre vyššie uvedenú korelačnú maticu možno tepelnú mapu zostaviť nasledovným spôsobom:

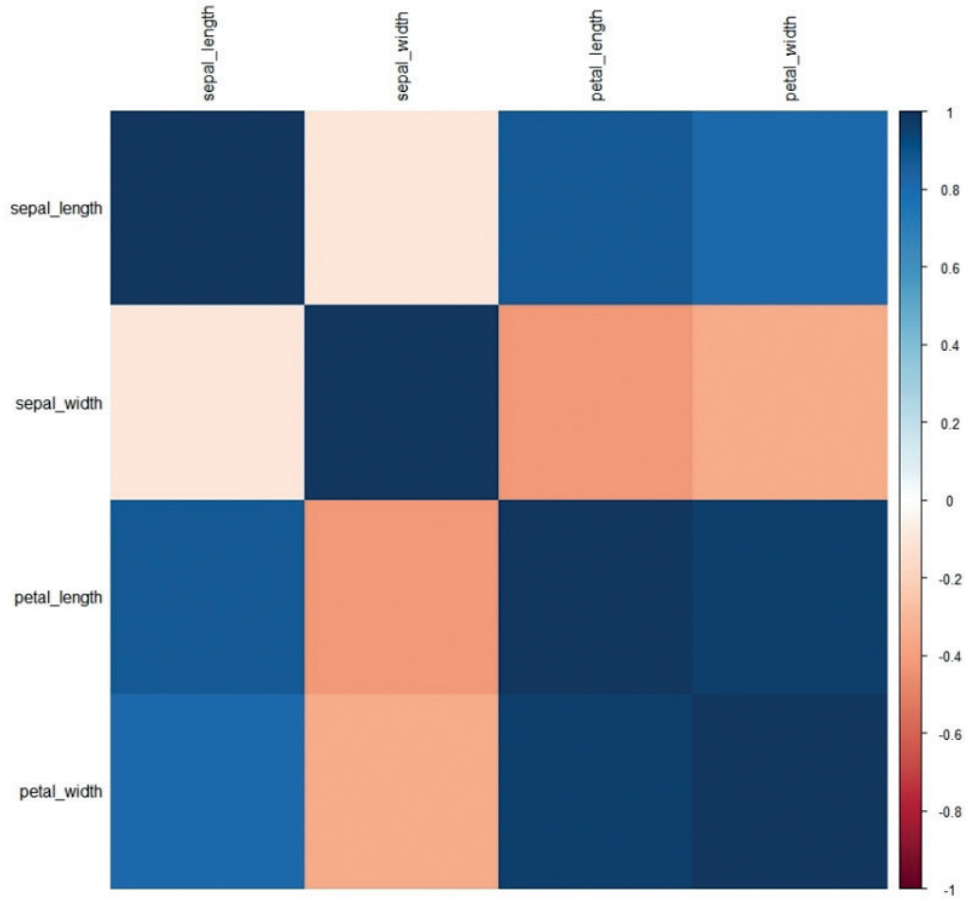


Korelačná tepelná mapa je len jednoduchým premietnutím korelačnej matice do farebnej mriežky, v ktorej je farba poľa definovaná hodnotou korelačného koeficientu pre danú dvojicu atribútov. Pre lepšiu čitateľnosť je ku korelačnej tepelnej mape doplnená mierka (vpravo) obsahujúca interval možných hodnôt pre korelačný koeficient. Namiesto hľadania čísel blízkych extrémom intervalu [1,–1] v korelačnej matici hľadáme v korelačnej tepelnej mape tmavočervené alebo tmavomodré polia mriežky, ktoré označujú rovnakú vlastnosť a sú ľahšie identifikovateľné pre väčšinu ľudí.

Príklad: Majme dataset Iris, ktorý je opísaný v Prílohe A tejto učebnice. Tento dataset obsahuje päť atribútov odmeraných na 150 entitách, z ktorých štyri sú číselné a piaty obsahuje lingvistickú hodnotu označujúcu triedu pre danú entitu (druh kvetu). V rámci korelačnej analýzy nie je možné pracovať s lingvistickými atribútmi, preto budeme brať do úvahy iba dataset o veľkosti 150×4. Pearsonova korelačná matica (mierne orezaného) datasetu Iris obsahuje nasledujúce hodnoty:

	sepal length	**sepal width**	**petal length**	**petal width**
sepal length	1	-0.1093692	0.8717542	0.8179536
sepal width	-0.1093692	1	-0.4205161	-0.3565441
petal length	0.8717542	-0.4205161	1	0.9627571
petal width	0.8179536	-0.3565441	0.9627571	1

Samozrejme, tento dataset neobsahuje veľké množstvo atribútov, takže zostavenie korelačnej tepelnej mapy nie je pre akt korelačnej analýzy v tomto prípade potrebné. Tepelnú mapu však napriek tomu uvádzame ako demonštráciu premietnutia hodnôt korelačných koeficientov do farebnej škály prezentovanej v tepelnej mape.



Interpretácia výsledkov korelačných koeficientov

Korelačný koeficient ukazuje, do akej miery je možné predikovať hodnoty atribútu A_2 vo vybranej vzorke údajov na základe atribútu A_1. Čím je hodnota tohto koeficientu bližšie k extrémom uvažovaného intervalu (teda k hodnote 1 alebo -1), tým je daný atribút A_1 vhodnejší na predikciu hodnôt voliteľného atribútu A_2.

Je zrejmé, že pre koreláciu meranú medzi dvojicou atribútov je **hodnota 0 najhorším prípadom**. V takejto situácii neexistuje medzi hodnotami týchto atribútov vzťah, ktorý by bolo možné použiť v prípade budovania matematických modelov na datasete, s ktorým pracujeme.

Literatúra sa mierne líši v pohľade na úroveň akceptovateľnosti hodnôt korelačných koeficientov – až na to, že čím vyššia korelácia, tým lepšie. Vo všeobecnosti o dvoch atribútoch hovoríme ako o silne korelovaných, keď hodnota korelačného koeficientu nameraná medzi nimi dosahuje hodnoty **vyššie ako 0.8**. Medzi týmito dvoma atribútmi existuje silná antikorelácia, ak korelačný koeficient dosiahne hodnotu **nižšiu ako -0.8**. Táto hranica akceptovateľnosti predikčného potenciálu môže byť **relaxovaná bližšie k hodnotám 0.7 alebo -0.7**, ale viac sa neodporúča.

4.3 EXPLORATÍVNA DÁTOVÁ ANALÝZA A VIZUALIZÁCIA DÁT

Exploratívna dátová analýza (bežne označovaná ako EDA) je metóda analýzy dát, ktorá využíva prieskum dát s cieľom nájsť vzory a trendy v danej populácii alebo vzorke. Vo svojej najzákladnejšej forme sa tento typ analýzy vykonáva vizuálnym skúmaním dát. Pred samotnou vizualizáciou je však potrebné vykonať niekoľko krokov, ktoré budú prínosom pri ďalšom hľadaní znalostí ukrytých v dátach:

- **Oboznámenie sa s datasetom** – pred tým, než vybranú množinu dát analyzujeme by sme mali byť schopní zodpovedať niekoľko základných otázok o tomto datasete:
  - **Kto zostavil dataset, kedy a prečo?** Tento bod je podstatný z hľadiska relevantnosti, aktuálnosti a užitočnosti datasetu. V prípade, že dataset zostavovali odborníci v danej oblasti, je relevantnejší, ako v prípade, keď by bol zostavený začiatočníkom, ktorý meral údaje na lacnom spotrebiteľskom senzore. Ak by bol dataset zostavený pred 93 rokmi, je možné, že údaje by neboli aktuálne, merania by boli menej presné ako by sme boli schopní zmerať dnes a podobne. Dataset je tiež zostavený so špecifickým účelom a teda nie je univerzálny (nemusí byť vhodný pre všetky úlohy).
  - **Aký veľký je dataset?** Veľkosťou datasetu rozumieme počet entít a atribútov nameraných vo vybranom datasete. V prípade, že máme dataset, ktorý je príliš veľký na to, aby sme s ním mohli pohodlne pracovať (pozri časť 1), musíme z neho vybrať vzorku na základe princípov uvedených v predchádzajúcej časti tejto učebnice. Oveľa väčším problémom je opačný problém – príliš malý súbor dát. Existujú však algoritmy, ktoré dokážu pracovať s malými datasetmi a prístupy, ktoré generujú nové entity na základe existujúcich, nazývané oversampling.
  - **Aké je zloženie datasetu?** Tento bod úzko súvisí s dôvodom zostavovania datasetu. Je dôležité prejsť všetkými atribútmi množiny údajov a porozumieť ich účelu. Je potrebné zamerať sa aj na to, či sú dáta zaznamenané v danom atribúte číselné alebo kategorické a v akom rozmedzí sa pohybujú hodnoty jednotlivých atribútov.
- **Výpočet sumárnych štatistík** – pre každý atribút je vhodné zostaviť základnú sumárnu štatistiku. Odporúčané merané hodnoty sú extrémy (min, max), medián alebo priemer, štandardná odchýlka a iné. Ide o veľmi dôležitý a informatívny krok, v ktorom získame stredné hodnoty daných atribútov, ich najmenšie a najväčšie hodnoty a jednotlivé atribúty môžeme ďalej opísať agregáciou.

- **Zostavenie korelačnej analýzy** – pre akýkoľvek dataset je vhodné zostaviť maticu alebo tepelnú mapu korelačných koeficientov. Táto matica meria korelácie medzi hodnotami všetkých atribútov datasetu a ukazuje, aké náročné bude zostavenie matematických modelov na danom datasete. Korelačná analýza tiež pomáha pri identifikácii atribútov a podmnožín datasetu vhodných na vizualizáciu.

Ďalším krokom exploratívnej analýzy dát je samotná vizualizácia datasetu. Hovoríme však o efektívnej vizualizácii dát, ktorá je založená na niekoľkých princípoch uvedených v ďalšej časti tejto učebnice.

Efektívna vizualizácia dát

Vizualizáciu dát nazývame efektívnou v prípade, že:

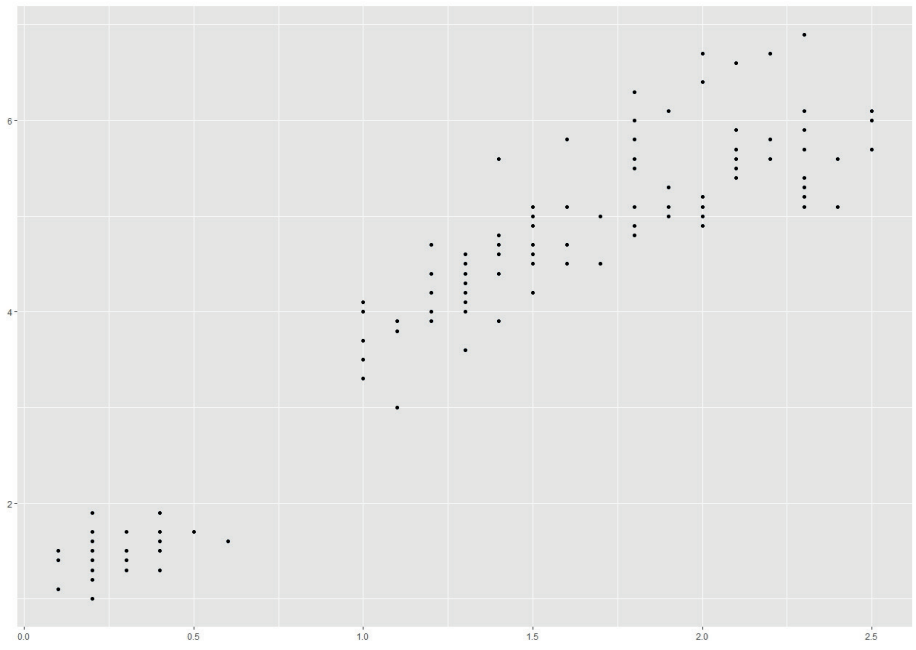
- vizualizujeme správne dáta,
- správnym spôsobom,
- s využitím správneho typu grafu.

Tieto tri body sú celkom prirodzené. Dáta, ktoré by sme mohli nazvať vhodnými na vizualizáciu z hľadiska analýzy dát, sú tie, ktoré nesú nejaké znalosti – najčastejšie prediktívny potenciál. Ako už vieme z predchádzajúcej časti učebnice, predikčný potenciál v datasetoch môžeme jednoducho identifikovať pomocou metód korelačnej analýzy, a preto budú na vizualizáciu vhodné tie časti datasetu, v ktorých sme identifikovali silné korelácie alebo antikorelácie medzi hodnotami atribútov. (pozri časť Interpretácia výsledkov korelačných koeficientov).

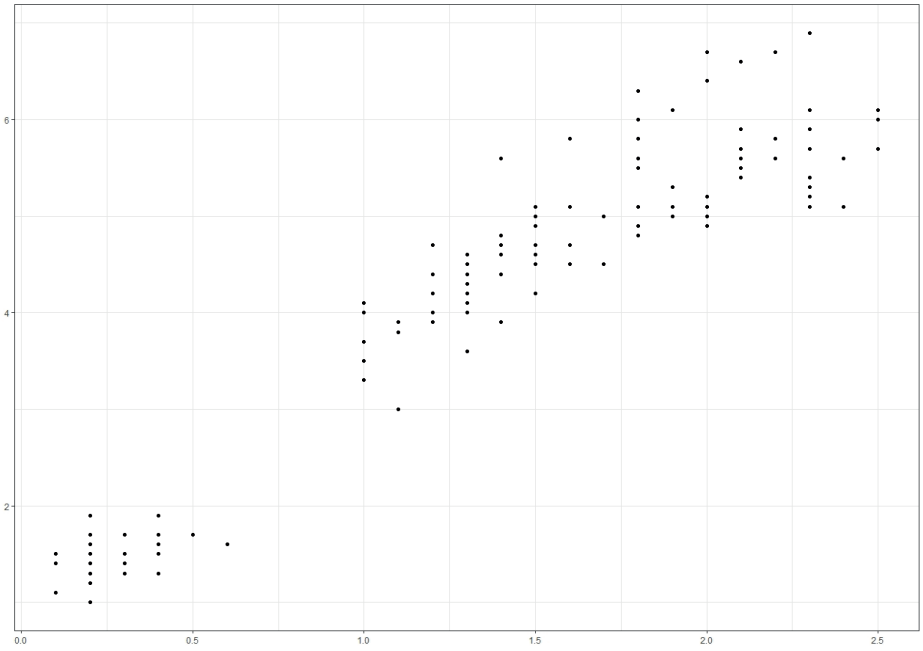
V našom prípade pod správnym spôsobom vizualizácie dát rozumieme elimináciu dvoch pomerne častých problémov vyskytujúcich sa pri vizualizácii datasetov:

- **Maximalizácia pomeru medzi použitou farbou a dátami** – keďže chceme vizualizovať dáta, v ideálnom prípade bude graf obsahovať minimum ďalších grafických prvkov (napríklad farbu pozadia, výraznú mriežku a podobne). Takáto maximalizácia pomeru farby a dát je nevyhnutná najmä pri vizualizácii veľkých datasetov, ktoré môžu byť vizualizované pomocou potenciálne veľmi malých, prekrývajúcich sa bodov (alebo iného typu objektov). Obrázok nižšie prezentuje štandardný bodový graf nakreslený v jazyku R (a) a úpravu tohto grafu tak, aby boli dáta viditeľnejšie (b).

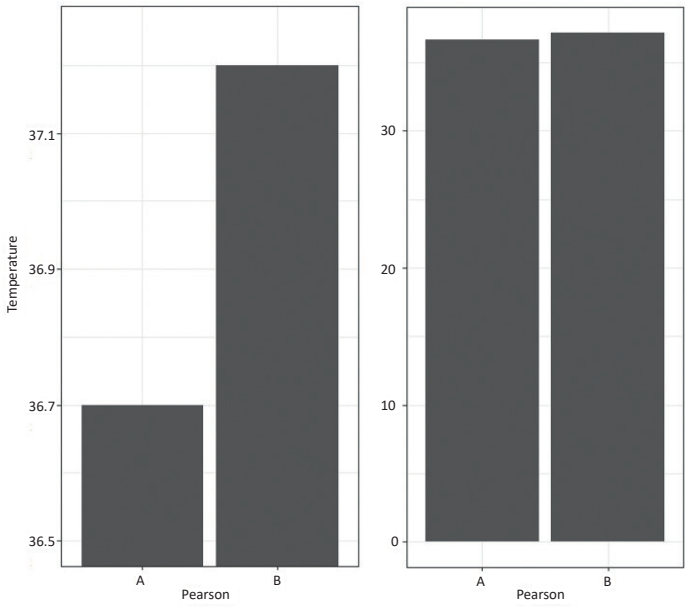
a)



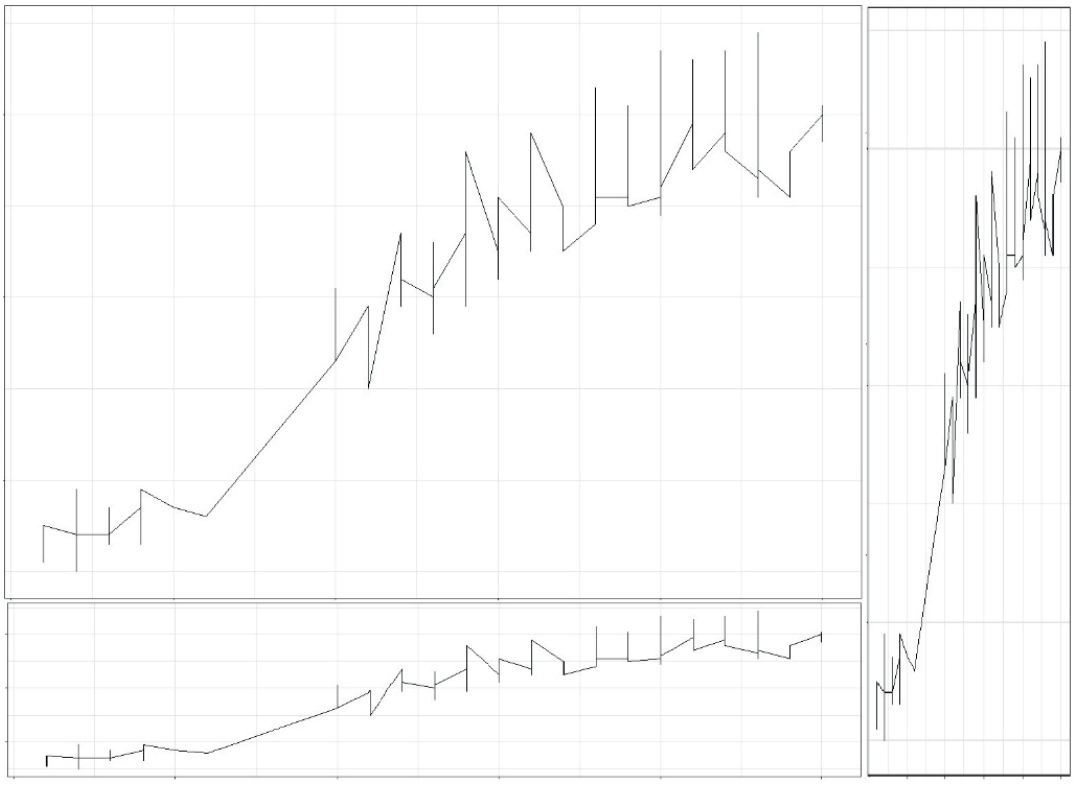
b)



- **Eliminácia skreslení** – pri vizuálnej prezentácii dát a ich následnej interpretácii často dochádza k skresleniam v dôsledku nesprávneho nastavenia osí alebo v dôsledku skreslenia samotného obrazového súboru. Grafy uvedené nižšie predstavujú oba tieto problémy. Prvý stĺpcový graf obsahuje porovnanie telesnej teploty dvoch osôb (A a B). Všimnime si, že hodnoty na ľavom grafe sa zdajú byť výrazne odlišné, zatiaľ čo hodnoty napravo sú veľmi podobné napriek skutočnosti, že ide o dva identické merania prezentované s rôznymi nastaveniami osí. Na ľavom obrázku uvádzame os *y* len od hodnoty 36,5 až 37,3 stupňa. Na pravom grafe uvádzame hodnoty *y* od 0 do 40 stupňov. Toto je typický mýliaci faktor prítomný v mnohých prípadoch vizualizácie dát.



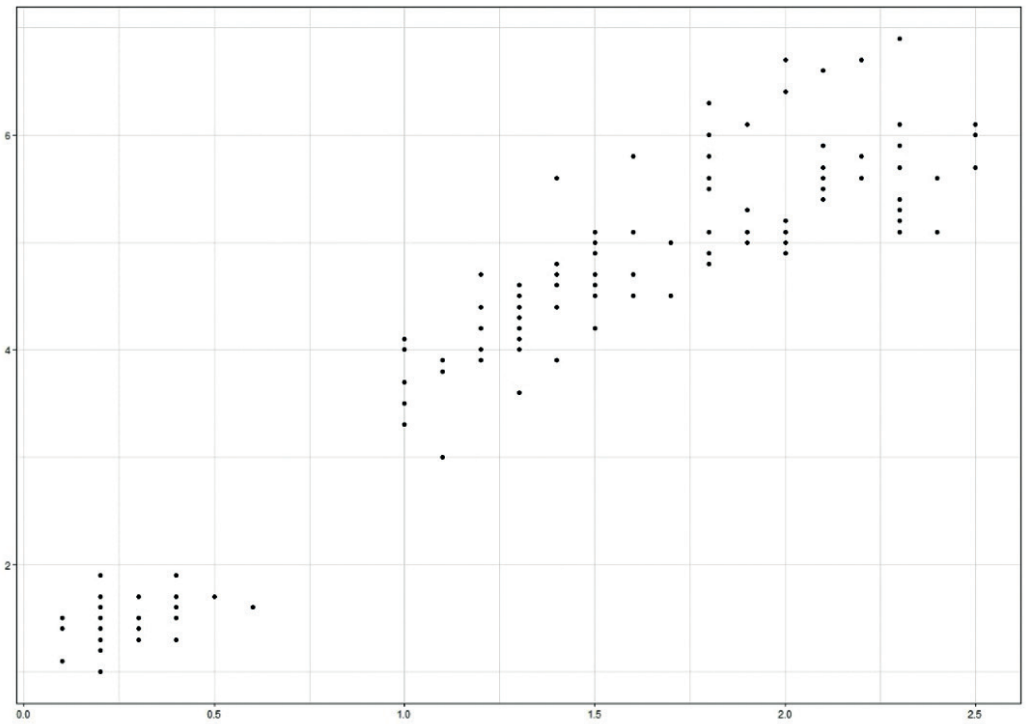
Druhým problémom skreslenia je skreslenie samotného obrazu. Na obrázku nižšie môžeme vidieť jeden graf prezentovaný v troch pomeroch strán obrazu. Je zrejmé, že spodný a pravý obrázok je nevhodný z dôvodu skreslenia skutočného tvaru trendu prezentovaného daným grafom. Práve preto bude ideálnym pomerom pre tieto dáta horný, ľavý graf - blízky štandardu pomer obrazu väčšiny moderných projektorov a monitorov - 16:9.



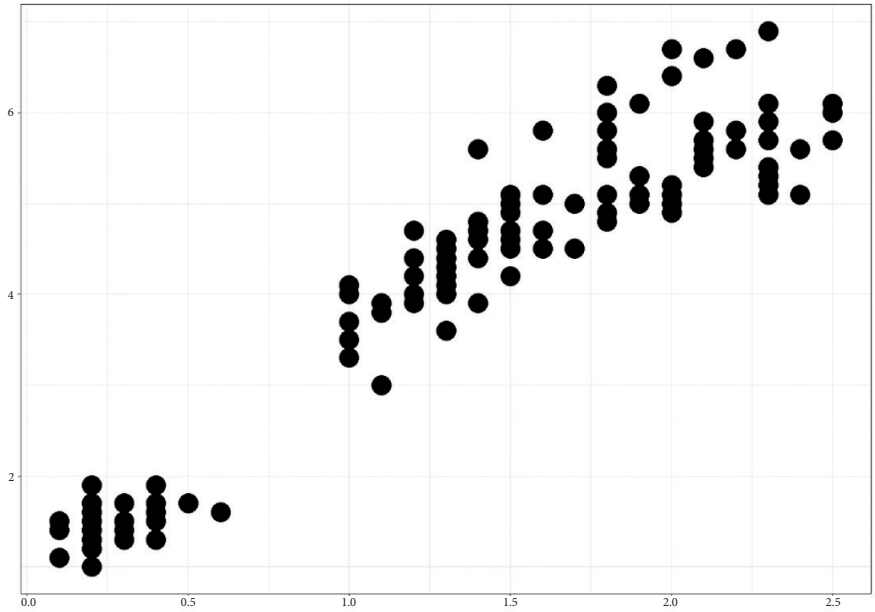
Posledným veľmi dôležitým prvkom efektívnej vizualizácie je použitie správneho typu grafu. Vo všeobecnosti sú populárne štyri typy grafov – bodové grafy, čiarové grafy, koláčové grafy a stĺpcové grafy. Každý z týchto typov grafov je vhodný na iné účely a má rôzne výhody a nevýhody. V tejto časti učebnice sa zameriame len na dva najbežnejšie spôsoby vizualizácie dát – bodové grafy a čiarové grafy.

Bodové grafy

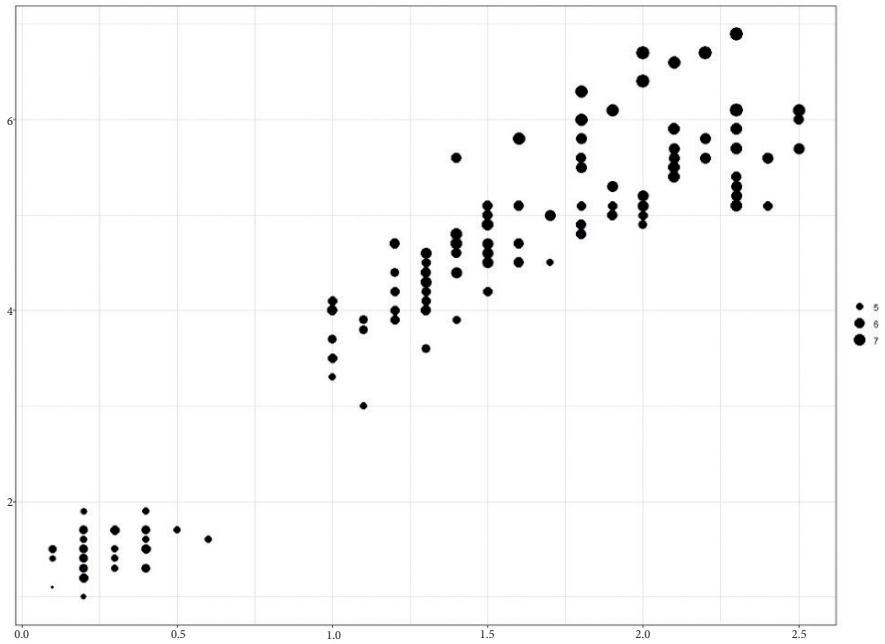
Bodové grafy sú využívané pri vizualizácii vzťahov medzi dvoma (alebo viacerými) atribútmi datasetu pomocou bodov. Štandardným spôsobom vizualizácie je porovnanie hodnôt dvoch atribútov v rovine:



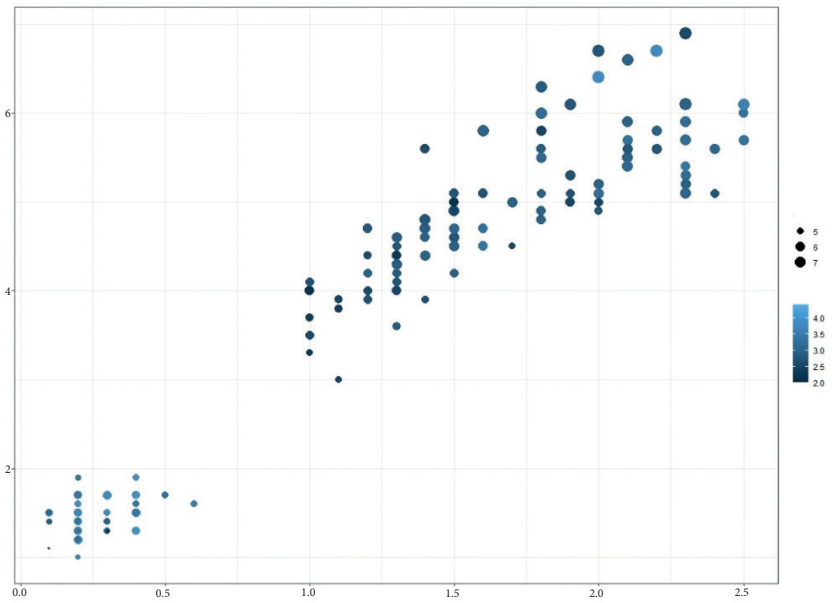
Pri takomto prístupe k vizualizácii dát musíme dbať na veľkosť bodu v grafe. Obrázok nižšie ukazuje presýtenie grafu spôsobené nevhodnou (príliš vysokou) veľkosťou bodu; podobný problém nastáva pri veľkom počte bodov, ktoré sú umiestnené blízko seba.



Veľkosť bodu však možno použiť v kontexte vizualizácie dát na sprostredkovanie dodatočných informácií. Ak nastavíme veľkosť bodu priamo úmernú veľkosti tretieho atribútu v datasete, môžeme vizualizovať vzťah medzi tromi atribútmi datasetu.

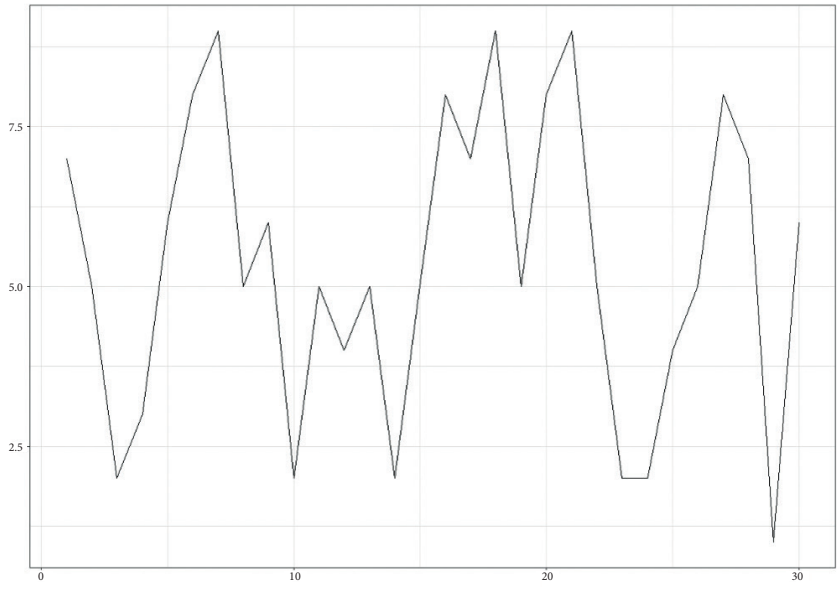


Tento koncept môžeme rozšíriť o ďalšie vlastnosti bodov, napríklad ich farbu (uvedené nižšie), a tak môžeme dosiahnuť vizualizáciu vzťahu medzi štyrmi atribútmi datasetu. Tento spôsob vizualizácie je však ťažko použiteľný pri veľkom počte bodov alebo atribútov. Vo všeobecnosti sa neodporúča vizualizovať viac ako tri alebo štyri atribúty v dvojdimenzionálnom priestore.

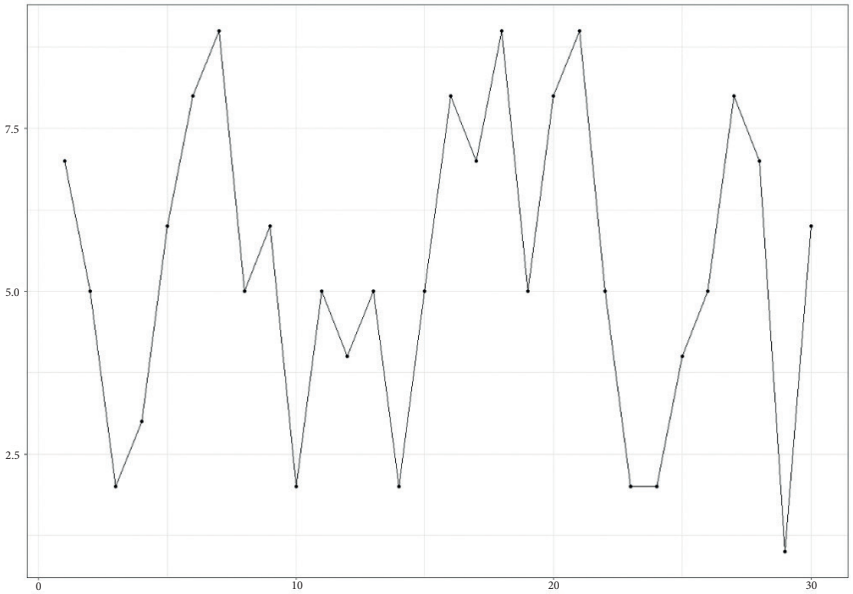


Čiarové grafy

Čiarové grafy slúžia na vizualizáciu priebehu hodnoty jedného atribútu v čase alebo na vizualizáciu fluktuácie hodnoty atribútu v závislosti od iného atribútu. Je dôležité poznamenať, že čiary v čiarových grafoch sú aproximáciou bodov – transformáciou diskrétnych dátových bodov na súvislé línie, a preto môžu byť na niektorých miestach nepresné. V porovnaní s bodovými grafmi sú čiarové grafy ťažko použiteľné pre kategorické (linguistické) dáta.



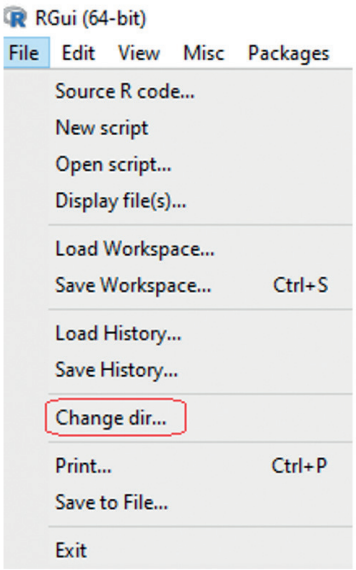
Keďže čiary sú aproximáciou dátových bodov, je odporúčané vizualizovať líniu a body, na ktorých je línia založená. Týmto spôsobom je znižovaná nejednoznačnosť správnosti hodnôt atribútov.



4.4 EXPLORATÍVNA DÁTOVÁ ANALÝZA V PRAXI

Táto časť učebnice je zameraná na praktickú aplikáciu metód exploratívnej analýzy dát v jazyku R. Nižšie uvedené príklady boli využívané vo verzii *R 4.3.0*, ale všetky prezentované koncepty a príkazy sú implementované vo všetkých verziách všetkých programovacích nástrojov vhodných na analýzu dát.

Ako prvé musíme zmeniť pracovný adresár pre našu prácu na adresár, kde sú uložené naše štruktúrované dáta – toto je možné vykonať pomocou hornej lišty programu: *File* → *Change dir* → *nastaviť pracovný adresár*. Na účely prezentácie metódy exploratívnej analýzy dát použijeme dataset Iris opísaný v *prílohe A*.



Po zmene pracovného adresára môžeme začať načítavať dataset do jazyka R. Túto operáciu je možné vykonať niekoľkými spôsobmi. Uvádzame (z nášho pohľadu) ten najjednoduchší - príkazy *read.table* a *read.csv*, ktoré fungujú rovnako pre rôzne typy vstupných súborov. Príkaz v tvare *read.csv* je používaný pre súbory *.csv*, ktoré sú najbežnejším štruktúrovaným vstupom pre nástroje na analýzu údajov. Formulár *read.table* je použiteľný v prípade vstupu vo formáte *.txt* alebo *.data*.

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

kde *názov* je názov súboru, v ktorom sú naše dáta uložené spolu s príponou súboru, klauzula *header* označuje, či má súbor so vstupnými dátami hlavičku (T pre true) alebo nie (F pre false) a časť príkazu *sep* očakáva znak, ktorým sú oddelené hodnoty atribútov vo vstupnom súbore.

Aby sme mohli načítaný dataset v programe ďalej používať, potrebujeme ho uložiť pod vybraným názvom, napríklad *názov_dát*:

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

Príklad takéhoto načítania dátového súboru uloženého pod názvom *nase_data.data* môže vyzerať nasledovne. V druhom prezentovanom príkaze uložíme náš dataset pod názvom *data*:

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

Exploratívna dátová analýza – Krok 1 – Oboznámenie sa s datasetom

V rámci oboznámenia sa s datasetom môžeme vykonať niekoľko veľmi jednoduchých operácií. Prvou z nich je vypísanie celého datasetu v konzole nástroja R pomocou *názov_dát*. To však nie je praktické pri veľkých datasetoch, ktoré obsahujú tisíce entít, a preto uvádzame druhú verziu príkazu na výpis entít datasetov – *head*, ktorý do konzoly vypíše definovaný počet entít od začiatku datasetu.

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

Príkladom tohto konceptu môže byť výpis celého datasetu uloženého pod názvom *data* alebo zoznam prvých piatich entít tohto súboru.

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

Výstupom tohto príkazu v jazyku R bude pseudotabuľka v nasledujúcom formáte:

```
> 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
```

Po základnom oboznámení sa s datasetom, jeho atribútmi a hodnotami v nich môžeme pristúpiť k výpočtu mier centrality a variability. Všetky tieto funkcie sú odvodené od anglickej verzie názvu jednotlivých funkcií (napr. *sd* pre štandardnú odchýlku) a ich vstup je len jedným z atribútov datasetu zapísaných v tvare

title_of _data$title_of _attribute.

Najuniverzálnejším z týchto príkazov je súhrnná funkcia *summary*, ktorá meria minimum, 1. kvartil, medián, priemer, 3. kvartil a maximum pre všetky atribúty zo vstupného datasetu.

```
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)
```

Príkladom takejto analýzy štatistických vlastností prítomných v dátach je použitie nasledujúcich príkazov:

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

Výstup týchto funkcií vykonávaných na datasete Iris pozostáva z nasledujúceho súboru hodnôt:

```
> 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
```

Exploratívna dátová analýza – Krok 2 – Korelačná analýza

Ako bolo uvedené v predchádzajúcich častiach tejto učebnice, jednou z najdôležitejších častí exploratívnej analýza dát je korelačná analýza. Základnou formou príkazu pre korelačnú analýzu je výpočet korelačného koeficientu medzi dvoma atribútmi datasetu pomocou funkcie *cor*. V nižšie uvedenej syntaxi príkazu vidíme, že typ korelačného koeficientu, ktorý chceme pre dáta vypočítať, je možné definovať pomocou parametra *method = typ_korelácie*. V prípade, že parameter *method* neuvedieme, funkcia automaticky použije Pearsonov korelačný koeficient.

```
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”)
```

V rámci analýzy dát však chceme preskúmať všetky vzťahy medzi všetkými atribútmi datasetu, a preto môžeme vytvoriť korelačnú maticu:

```
cor(title_of_data)

cor(title_of_data, method = “spearman”)
```

Korelačnú analýzu datasetu Iris je možné vykonať pomocou nasledujúcich jednoduchých príkazov:

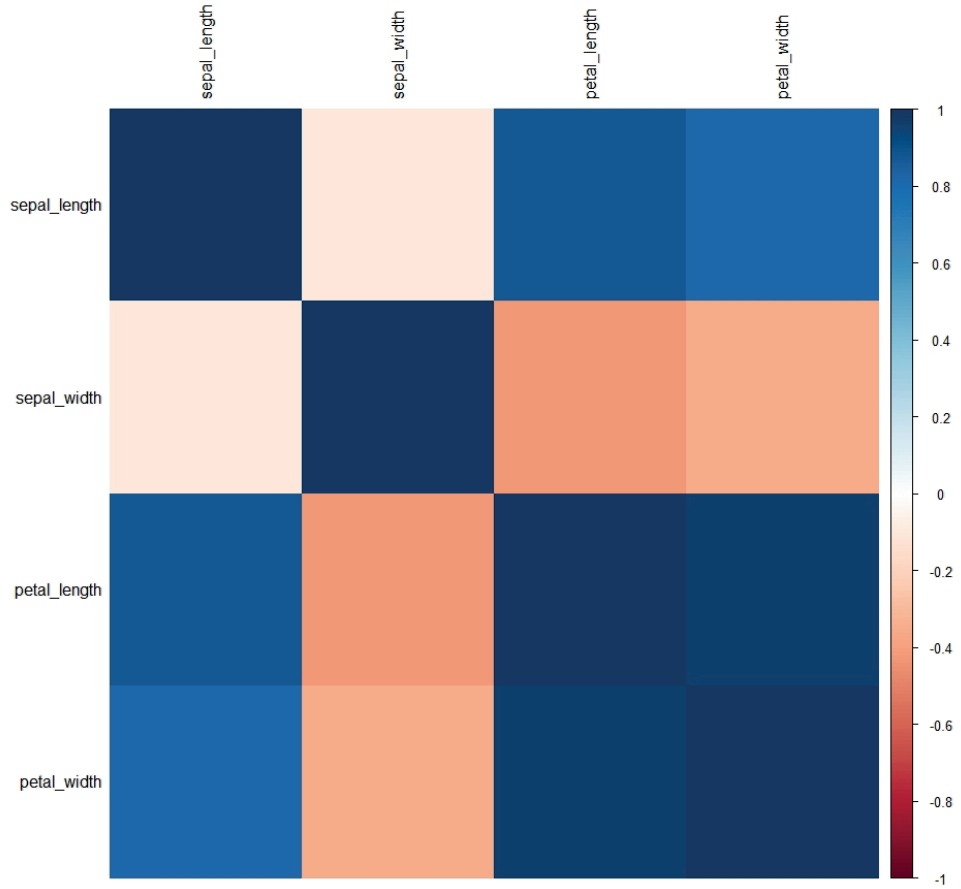
```
cor(data[, 1:4])

cor(data[, 1:4], method = “spearman”)
```

Poznámka: Keďže dataset Iris obsahuje jeden atribút, ktorého hodnoty sú lingvistické (trieda) a korelačná matica sa skladá iba z číselných hodnôt, funkcia cor musí mať ako vstup len prvé štyri (číselné) atribúty. Dosiahneme to výberom stĺpcov 1:4 z datasetu s názvom 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
```

Ako bolo uvedené v časti 4.2, pre veľké datasety je odporúčané používať korelačnú tepelnú mapu. Aby sme mohli použiť túto metódu vizualizácie, musíme v jazyku R nainštalovať balík funkcií, ktorý obsahuje funkciu vizualizácie korelačnej tepelnej mapy s názvom *corrplot*. Po nainštalovaní tohto balíka ho len načítame pomocou požadovanej funkcie a potom vytvoríme tepelnú mapu korelácie:

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



Korelačná matica a tepelná mapa pre dataset Iris hovoria o vzťahoch, ktoré možno použiť pri ďalšej analýze dát. Konkrétne nás zaujímajú všetky vzťahy medzi atribútmi, kde bol akýkoľvek typ korelačného koeficientu vyšší ako 0,8 alebo nižší ako -0,8:

- ρ(sepal_length, petal_length) ≈ 0.87
- ρ(sepal_length, petal_width) ≈ 0.82
- ρ(petal_length, petal_length) ≈ 0.94

Tieto vzťahy sú medzi hodnotami atribútov sú vhodné na vizualizovanie.

Exploratívna dátová analýza – Krok 3 – Vizualizácia dát

Po analýze korelácií sme pripravení vizualizovať páry atribútov, v ktorých sme zaznamenali silnú koreláciu alebo antikoreláciu. Pred samotnou vizualizáciou však musíme nainštalovať balík slúžiaci na účely vizualizácie:

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

Balík *ggplot2* je jeden z najpopulárnejších balíkov používaných vo vizualizácii dát, ktorý obsahuje funkcie na vykreslenie bodových, čiarových, stĺpcových a iných grafov. V tejto časti učebnice vyberieme niekoľko jednoduchých príkladov týchto funkcií.

Bodové grafy

Najzákladnejšou a zároveň najefektívnejšou vizualizáciou dát je bodový graf. V rámci jazyka R a balíka *ggplot2* používame funkciu *ggplot* na vytvorenie akéhokoľvek typu grafu, pričom funkcia očakáva niekoľko hodnôt ako vstup. Pre najjednoduchšie grafy sú týmito vstupmi:

- Názov datasetu, s ktorým pracujeme (v našom prípade je týmto názvom data).
- Sekcia *aes*, ktorej názov je odvodený z anglického *aestetics,* očakáva informáciu aspoň o jednej osi. Tieto informácie sú prezentované vo forme *názov_osi* (*x* alebo *y*) = *názov_atribútu* reprezentovaného osou.
- Typ grafu.

Vo všeobecnosti syntax príkazov pre bodové grafy obsahuje priradenie dvoch atribútov osiam grafu a sekcii príkazu *+geom_point()*. Zovšeobecnená syntax pre tento typ príkazu je uvedená nižšie:

```
ggplot(title_of_data, aes(x = title_of_attribute_1, y = title_of_
attribute_2)) + geom_point()
```

Farbu bodov môžeme zmeniť pomocou rozšírenia sekcie príkazu *+geom_point()* pridaním druhej sekcie *aes* platnej len pre samotné body - konkrétne *+geom_point(aes(color = "názov farby"))*. Namiesto názvu farby môžeme v sekcii *+geom_point()* zadať názov atribútu z datasetu, s ktorým pracujeme. Týmto spôsobom dosiahneme vizualizáciu v dvoch dimenziách (atribútoch) s dodatočnou dimenziou označenou farbou bodov, ktorá sa mení na základe hodnôt zvoleného atribútu. Aby sme dodržali princípy efektívnej vizualizácie dát prezentované v predchádzajúcej časti textu, na koniec príkazu pridávame klauzulu *+theme_bw()*, ktorá zaistí biele pozadie pod samotným grafom, čím sa maximalizuje pomer medzi dátami a farbou, ktorá je v grafe použitá.

```
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()
```

Jednoduchý príklad tohto prístupu je uvedený nižšie. Predstavujeme tiež dva ďalšie koncepty:

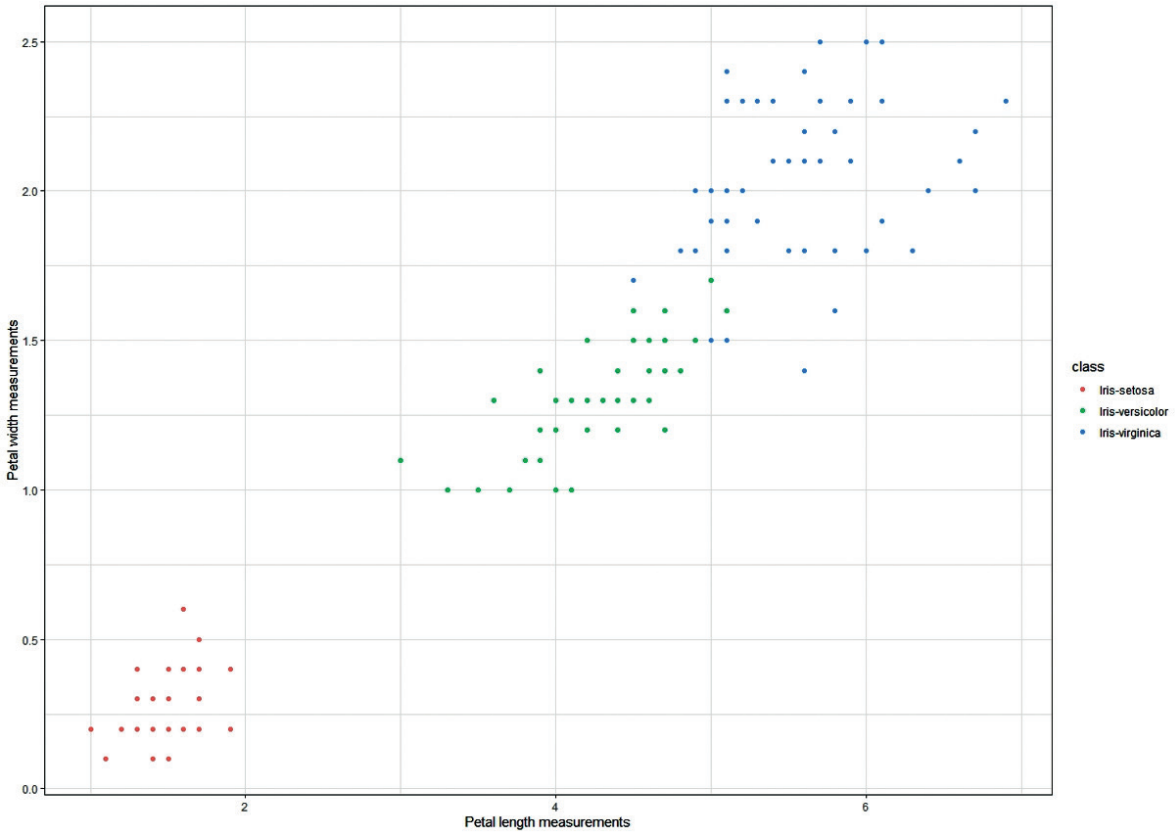
- Vytvorený graf je možné uložiť pod zvoleným názvom, napríklad *graf1*, ako je uvedené nižšie.
- Do takto uloženého grafu môžeme pridať ďalšie časti príkazu. V nižšie uvedenom príklade používame *+xlab()* a *+ylab()* na pridanie názovo pre osi *x* a *y*. Graf je následne vykreslený volaním jeho názvu v konzole.

```
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
```

Nasledujúci kód pre dataset 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
```

produkuje tento graf:



Čiarové grafy

Ďalším z často používaných typov grafov je čiarový graf, ktorý slúži na vizualizáciu priebehu hodnoty jedného atribútu v čase alebo na vizualizáciu kolísania hodnoty atribútu v závislosti od iného atribútu. Syntax príkazu čiarového grafu v balíku *ggplot2* sa výrazne nelíši od predchádzajúcich príkladov uvedených v tejto časti učebnice. Jediným rozdielom je typ geometrie použitej pri kreslení grafu – v prípade čiarových grafov ide o *+geom_line()*. Pomocou voľby typu línie v sekcii príkazu *geom_line()* môžeme zmeniť aj typ línie, ktorá sa používa pri vykresľovaní dát.

```
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”)
```

Podobne ako pri bodových grafoch môžeme jednoducho zmeniť farbu geometrie - v tomto prípade línie - pomocou klauzuly *color*, ktorú je možné kombinovať so všetkými typmi línií v grafe.

```
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”)
```

Keďže líniový graf je vždy aproximáciou dátových bodov, je vhodné vizualizovať samotné body spolu s líniovým grafom. Takto je možné grafy zostaviť kombináciou geometrie línií a bodov nasledovným spôsobom:

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

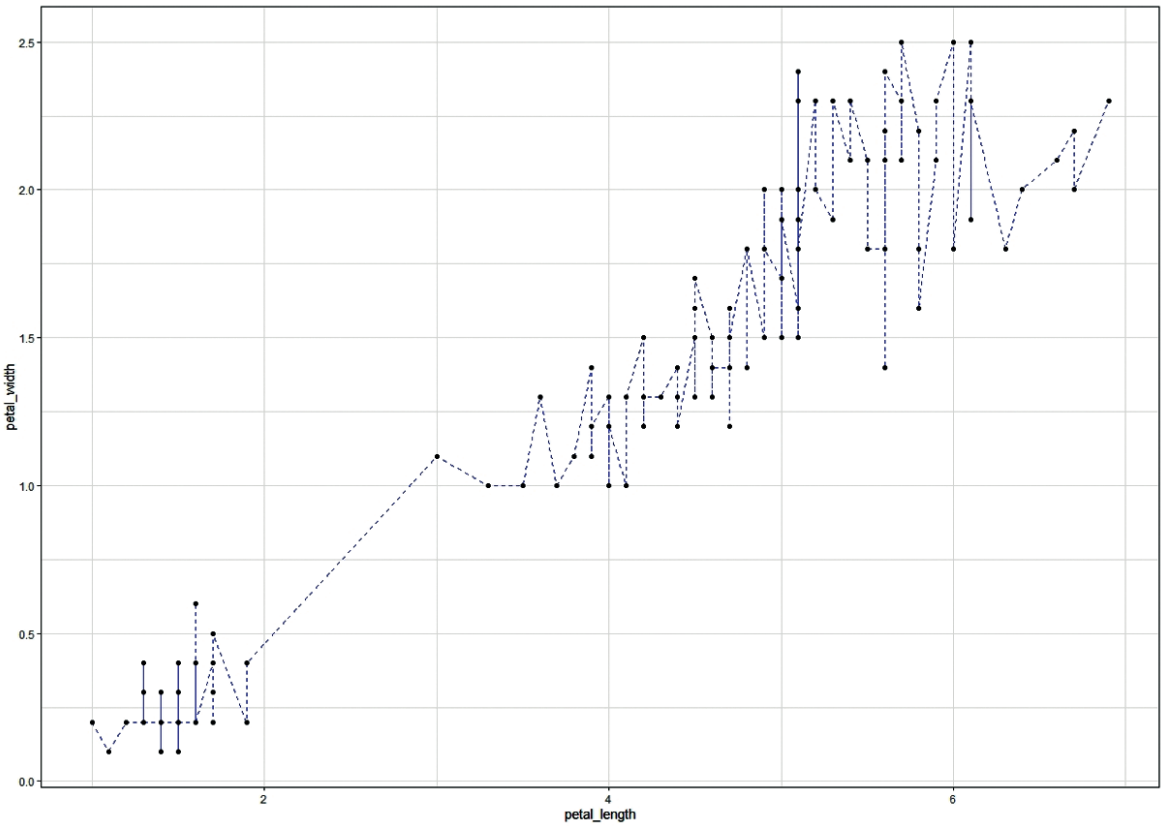
Je pochopiteľné, že kombinácia všetkých vyššie uvedených možností je možná:

```
ggplot(data, aes(x=attribute_title_1, y=attribute_title_2)) + geom_
line(linetype = “dashed”, color = “blue”) + geom_point()
```

Takýto kód pre dataset Iris:

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

generuje nasledovný graf:



Skutočná sila líniových grafov je v ich schopnosti vizualizovať porovnanie medzi hodnotami jedného atribútu a hodnotami množiny atribútov – inými slovami, možnosť vykresliť viac ako jednu líniu. V príklade syntaxe nižšie vidíme, že základná funkcia *ggplot* obsahuje iba jeden atribút (ten, ktorý je umiestnený na osi *x*), a na vizualizáciu hodnôt *atribútu2* a *atribútu3* používame os *y*. Táto vizualizácia je vykonávaná pomocou samostatných + *geom_line()* sekcií kódu. S týmto prístupom uvádzame aj množstvo kombinácií vyššie uvedených možností.

```
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”)
```

Majme dataset Iris, v ktorom sme namerali silné korelácie medzi tromi atribútmi – *sepal_length, petal_length* a *petal_width*. Kolísanie *petal_width* a *petal_length* v závislosti od sepal_length je možné vizualizovať pomocou dvoch línií, ktoré budú oddelené ich typom alebo farbou. V našom prípade:

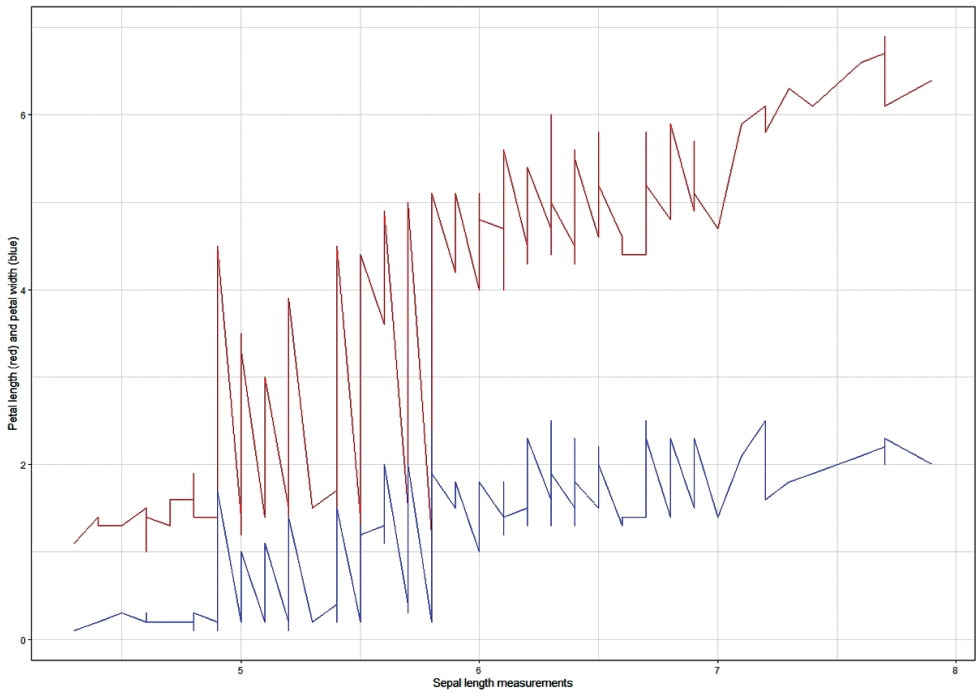
- kolísanie hodnôt atribútu *petal_length* na základe *sepal_length* je znázornené pomocou červenej línie,
- kolísanie hodnôt *petal_width* na základe *sepal_length* je vizualizované pomocou modrej línie.

```
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”)
So, the following code for the Iris dataset
```

Nasledovný kód pre dataset 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)")
```

produkuje graf:



Takáto vizualizácia dát môže byť použitá na hľadanie trendov a vzorov v dátach, ktoré následne môžu byť použité v komplexnejších prístupoch k analýze dát prezentovaných v ďalších častiach tejto učebnice. Exploratívna analýza dát má jeden nedostatok – je ťažké ju použiť na skutočne veľkých datasetoch, ktoré vyžadujú redukciu dimenzionality a niektoré ďalšie techniky na korektnej analýzy.

KAPITOLA 5

FUZZY MNOŽINY

Autorom tejto časti učebnice je Alžbeta Michalíková z Katedry informatiky, Fakulty prírodných vied, Univerzity Mateja Bela v Banskej Bystrici zo Slovenska.

V bežnom živote často používame vágne pojmy ako mladý človek, trochu soli, mierne doprava, silný vietor, vysoká teplota, nízka cena... Tieto výrazy nemajú presné hranice. Nie sú jasne definované. Môžeme ich nazvať **vágne** alebo **fuzzy**. Prvú myšlienku matematického modelovania pomocou fuzzy konceptov možno nájsť v článku Lotfiho A. Zadeha [1]:

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

Používanie fuzzy množín sa označuje ako **práca s prirodzeným jazykom** (počítanie nielen s číslami, ale aj s pojmami ľudskej reči). Zároveň sa predpokladá, že rôzni ľudia vnímajú rôzne pojmy rôzne. Hlavná časť nasledujúceho textu vychádza z vysokoškolskej učebnice [2] **Fuzzy množiny v informatike**, ktorá je určená študentom aplikovanej informatiky na Katedre informatiky Fakulty prírodných vied Univerzity Mateja Bela v Banskej Bystrici.

Prečo používať fuzzy logiku?

- Myšlienka fuzzy logiky je ľahko pochopiteľná,
- ide o flexibilný systém, ktorý je odolný voči nepresným údajom,
- dokáže pracovať so skúsenosťami odborníkov,
- dokáže modelovať nelineárny systém akejkoľvek zložitosti,
- možno ju použiť pri riadení štandardných technických zariadení.

Použitie fuzzy množín

- expertné systémy,
- rozpoznávanie a klasifikácia objektov,
- teória riadenia a regulácie,
- databázové systémy,
- matematické modelovanie,
- najaktuálnejšie – vysvetliteľné neurónové siete.

Oblasti použitia fuzzy množín

Fuzzy množiny je možné použiť kdekoľvek, kde je do výpočtu zahrnutá neistota. Často sa používajú v zariadeniach, ktoré z ekonomického hľadiska nepredstavujú drahé spotrebiče, napr. **elektronické domáce spotrebiče** (práčky, mikrovlnné rúry, vysávače, holiace strojčeky, tlakomery, ...). Nájdeme ich ale aj **v zložitých a ekonomicky aj výpočtovo náročných zariadeniach**, napríklad

- riadenie metra v Japonsku – (mesto Sendai – od roku 1988) [3],
- riadenie vysokej pece (regulácia teploty, ktorá môže byť riadená efektívnejšie ako s konvenčnými regulátormi),
- riadenie jadrových elektrární [4], ...

Príklad: Predstavme si, že je v inzeráte na zaujímavú pracovnú pozíciu uvedené, že sa vyžaduje, aby vek kandidátov bol v intervale 20-30 rokov. Opíšme túto úlohu!

Aký je definičný obor pre množinu vek človeka, t. j. v akom rozsahu budete uvažovať vek človeka?

Môžete opísať túto množinu pomocou jej charakteristickej funkcie?

Môže na tento inzerát odpovedať človek, ktorý bude mať zajtra 31. narodeniny?

Aký je definičný obor pre množinu vek človeka?

Definičný obor množiny predstavuje všetky prípustné hodnoty uvažovanej premennej. Táto množina sa zvyčajne označuje pomocou písmena $\mathbb{X}$. V našom príklade to môže byť množina napríklad $\mathbb{X} = \langle 0, \infty)$.

Môžete opísať túto množinu pomocou jej charakteristickej funkcie?

Charakteristická funkcia je funkcia, ktorá priraďuje číslo 1 tým prvkom, ktoré patria do uvažovanej množiny. Na druhej strane, tým prvkom, ktoré do uvažovanej množiny nepatria, priraďuje číslo 0. Táto funkcia sa zvyčajne označuje písmenom χ. V našom príklade má charakteristická funkcia nasledujúci zápis:

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

Môže na tento inzerát odpovedať človek, ktorý bude mať zajtra 31. narodeniny?

NIE! - pretože kým si niekto prečíta jeho odpoveď, už daný človek nebude spĺňať požadovanú podmienku.

Príklad: Majme si podobnú situáciu: V inzeráte na zaujímavú pracovnú ponuku je uvedená požiadavka, že firma hľadá mladých ľudí.

Zmenila sa situácia v porovnaní s predchádzajúcim príkladom?

Aký je definičný obor tejto množiny – množiny mladých ľudí?

Ako môžeme opísať množinu mladých ľudí?

Môže na tento inzerát odpovedať človek, ktorý bude mať zajtra 31. narodeniny?

Zmenila sa situácia v porovnaní s predchádzajúcim príkladom?

ÁNO! – množina mladých ľudí predstavuje **fuzzy množinu**. Neexistuje žiadna ostrá hranica pre hodnotu **vek človeka**, ktorá určuje, ktorí ľudia patria do tejto skupiny!

Aký je definičný obor tejto množiny – množiny mladých ľudí?

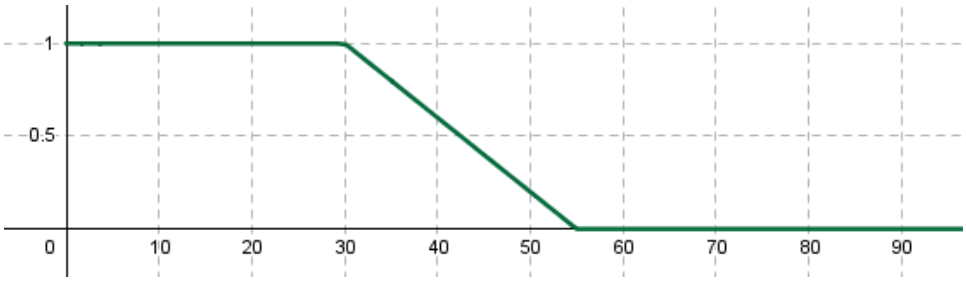
Definičným oborom fuzzy množiny môže byť ľubovoľná množina, ktorá obsahuje všetky prípustné hodnoty. V tomto prípade to môže to byť rovnaká množina ako pri klasickej množine, t.j. množina $\mathbb{X}=\langle 0, \infty)$.

Ako môžeme opísať množinu mladých ľudí?

Zamyslime sa najprv nad rôznymi hodnotami veku človeka. Napríklad 20-ročný človek je určite mladý. Preto mu pridelíme stupeň mladosti rovný 1. Podobne aj 30-ročného človeka môžeme považovať za mladého. Preto mu priradíme stupeň mladosti rovný 1. Ak uvažujeme, že 35-ročný človek už nie je porovnateľne mladý s 20-ročným človekom, môžeme mu priradiť stupeň mladosti 0.8, ... Na opísanie fuzzy množín používame takzvané funkcie príslušnosti. Označujú sa písmenom μ. Pomocou tejto funkcie priraďujeme každému prvku z definičného oboru nejakú hodnotu z jednotkového intervalu (t.j. z intervalu $\langle 0,1\rangle$). Najprv sa pozrime na hodnoty veku človeka, ktoré určite považujeme za vek mladého človeka. Nech sú tieto hodnoty z intervalu $\langle 0,30\rangle$. Funkcia príslušnosti priradí týmto vstupným hodnotám výstupnú hodnotu rovnajúcu sa 1. Teraz sa pozrime na hodnoty veku človeka, ktoré už určite nepovažujeme za vek mladého človeka. Príkladom takýchto hodnôt môžu byť hodnoty väčšie ako 55. K týmto vstupným hodnotám funkcia príslušnosti priradí výstupnú hodnotu rovnú 0. Zostáva určiť výstupné hodnoty príslušnosti pre vstupné hodnoty z intervalu $\langle 30,55\rangle$. Očakávame, že výstupná hodnota (príslušnosti do množiny mladých ľudí) bude postupne klesať z 1 na 0 (pozri Obrázok 1).

Fuzzy množinu mladých ľudí označme písmenom B. Potom opísanú funkciu zapisujeme nasledovne

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



Obrázok 1. Funkcia príslušnosti fuzzy množiny mladý ľudia

Poznámky:

Pojmy fuzzy množina a funkcia príslušnosti sa často považujú za ekvivalentné.

Výstupná hodnota, ktorá je priradená konkrétnej vstupnej hodnote, sa nazýva **stupeň príslušnosti**.

Nech fuzzy množina, určená vzorcom uvedeným vyššie, predstavuje zápis fuzzy množiny mladý ľudia. Určte, aký stupeň príslušnosti priradí táto funkcia ľuďom, ktorí majú 20, 35 a 45 rokov?

Použitím vyššie uvedeného vzorca dostaneme

μ_B (20) = 1, t. j. 20-ročný človek je určite mladý,

μ_B (35) = 0,8, t. j. 35-ročný človek je mladý so stupňom 0,8,

μ_B (40) = 0,6, t. j. 40-ročný človek je mladý so stupňom 0,6.

Môže na tento inzerát odpovedať človek, ktorý bude mať zajtra 31. narodeniny?

ÁNO! – pretože jeho stupeň príslušnosti k fuzzy množine μ_B je rovný 0,96 (pretože $\mu_B(31) = 0,96$). Táto hodnota predstavuje vysoký stupeň príslušnosti k fuzzy množine mladých ľudí.

Poznámka:

Existuje veľké množstvo rôznych typov funkcií príslušnosti fuzzy množín. Niektoré z nich si ukážeme v nasledujúcom príklade.

Príklad: Modelujme fuzzy množinu C reálnych čísel, ktorá predstavuje výraz „približne 7".
Aký je definičný obor množiny C?
Ako môžeme opísať vlastnosti tejto množiny?
Výraz „približne 7" používame v bežnom živote napríklad vo vetách

Vonku je približne 7 stupňov Celzia.

alebo

V obchode som minul približne 7€.

Aký je definičný obor množiny C?

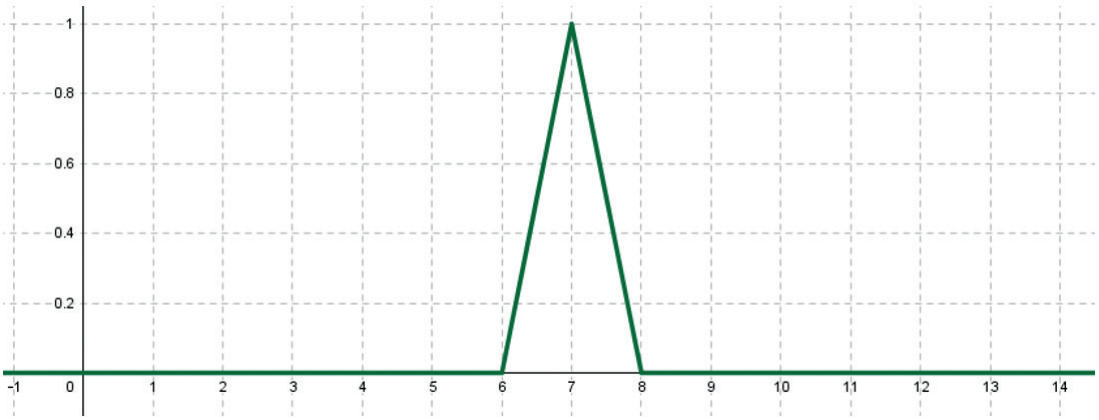
Zvyčajne za definičný obor volíme najväčšiu možnú množinu. V tomto príklade by to mohla byť celá množina reálnych čísel, t. j. X=R.

Ako môžeme opísať vlastnosti tejto množiny?

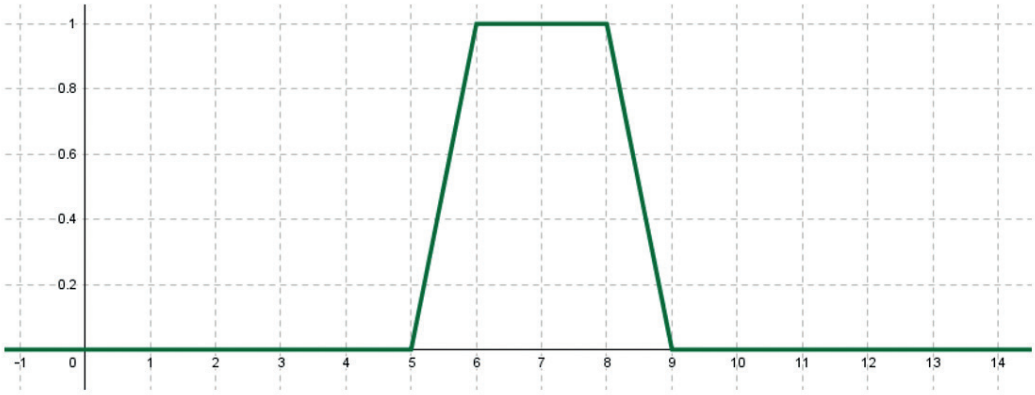
Uvažovanú množinu „približne 7" sme pomenovali ako množina C. Funkcia príslušnosti fuzzy množiny μ_C musí spĺňať dve podmienky:

1. μ_C (7) = 1,
2. s rastúcim rozdielom |x-7| by hodnoty funkcie μ_C mali klesať k nule.

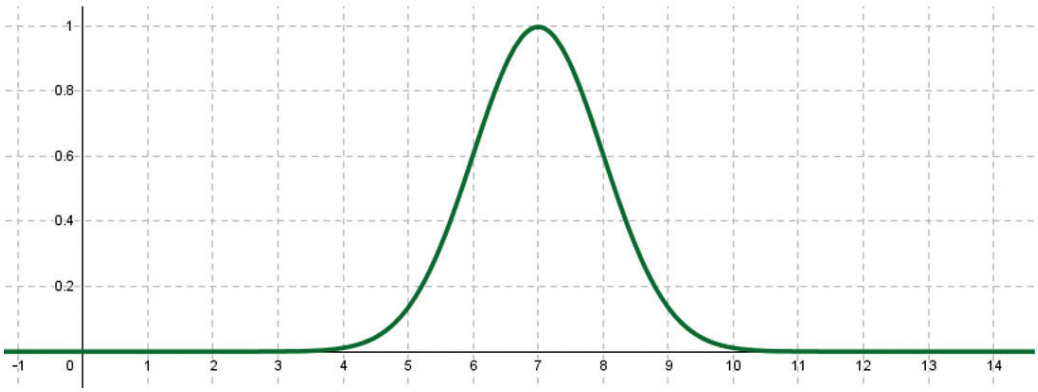
Príklady niekoľkých funkcií spĺňajúcich uvedené podmienky sú zobrazené na Obrázku 2, Obrázku 3 a Obrázku 4.



Obrázok 2. Trojuholníková funkcia príslušnosti predstavujúca hodnotu „približne 7“



Obrázok 3. Lichobežníková funkcia príslušnosti predstavujúca hodnotu „približne 7“



Obrázok 4. Ďalšia funkcia príslušnosti predstavujúca hodnotu „približne 7“

Typy funkcií príslušnosti

Na Obrázkoch 2-4 sme videli, že funkcia príslušnosti môže mať veľa rôznych foriem. Ukážeme si niektoré z nich, ktoré sú definované v softvéri MATLAB. Softvér MATLAB budeme neskôr používať na riešenie úloh s použitím rôznych fuzzy prístupov.

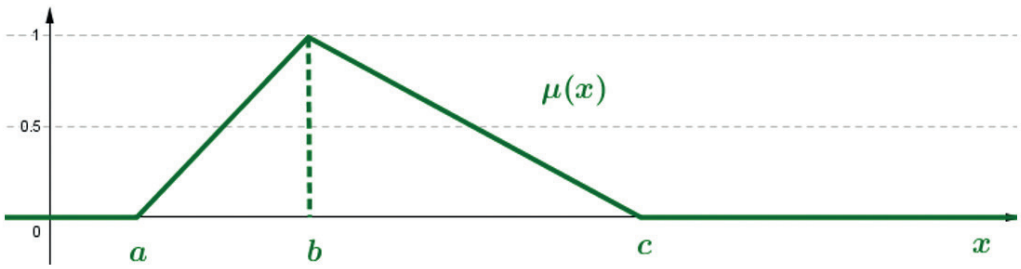
Lineárne funkcie príslušnosti

Lineárne funkcie príslušnosti predstavujú najjednoduchší typ funkcií príslušnosti. Sú vytvorené použitím častí priamok, konkrétne polpriamok a úsečiek. Delia sa na dve základné skupiny:

- trojuholníkové funkcie príslušnosti,
- lichobežníkové funkcie príslušnosti.

Trojuholníková funkcia príslušnosti

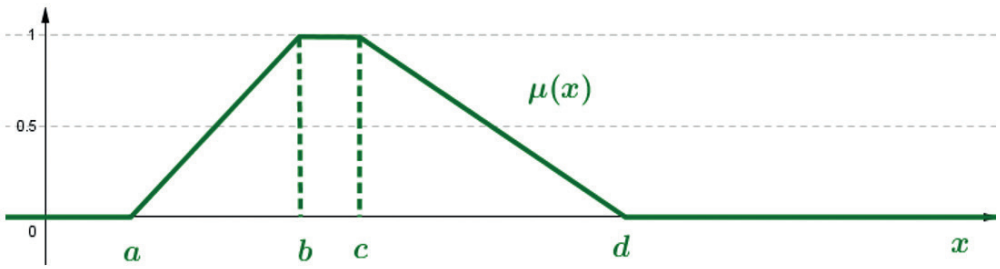
Trojuholníková funkcia príslušnosti pozostáva zo štyroch častí (pozri Obrázok 5). Prvá časť priraďuje vstupným hodnotám výstupnú hodnotu rovnajúcu sa nule (interval (-∞, a⟩ na Obrázku 5). Druhá časť rastie lineárne z hodnoty 0 na hodnotu 1 (interval (a, b⟩ na Obrázku 5). Tretia časť lineárne klesá z hodnoty 1 na hodnotu 0 (interval (b, c⟩ na Obrázku 5). Posledná časť opäť priraďuje výstupnú hodnotu rovnajúcu sa 0 (interval (c, ∞) na Obrázku 5). Vo všeobecnosti je táto funkcia príslušnosti opísaná tromi parametrami a, b, c. V softvéri MATLAB je označená ako **trimf** a pre jej parametre používame zápis [**a b c**]. Všimnite si, že trojuholníková funkcia príslušnosti dosiahne výstupnú hodnotu rovnajúcu sa 1 len pre jeden vstup (konkrétne pre vstupnú hodnotu b).



Obrázok 5. Všeobecná trojuholníková funkcia príslušnosti

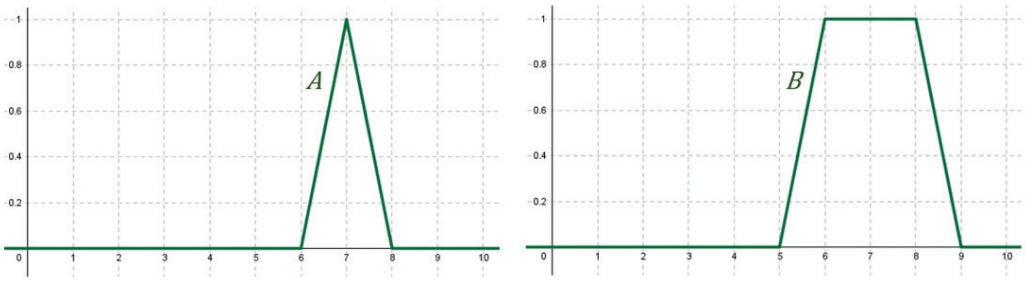
Lichobežníková funkcia príslušnosti

Lichobežníková funkcia príslušnosti pozostáva z piatich častí (pozri Obrázok 6). Na rozdiel od trojuholníkovej funkcie príslušnosti sa táto funkcia skladá z intervalu vstupných hodnôt, ktoré dosahujú výstupnú hodnotu rovnú 1. Vo všeobecnosti je táto funkcia príslušnosti popísaná štyrmi parametrami a, b, c, d. V softvéri MATLAB sa označuje ako **trapmf** a pre jej parametre používame zápis [**a b c d**].



Obrázok 6. Všeobecná lichobežníková funkcia príslušnosti

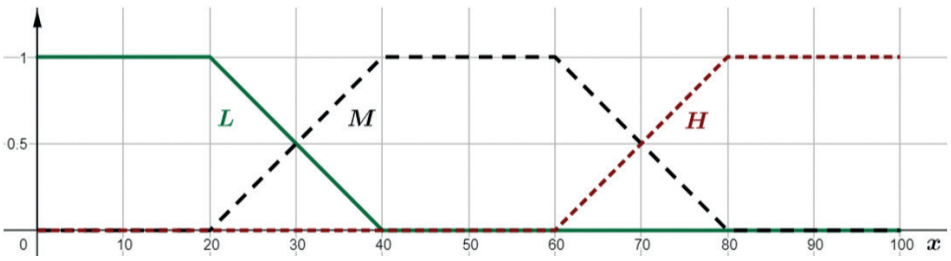
Príklad: Napíšte, ako v softvéri MATLAB zapíšete fuzzy množiny A, B, ktoré sú zobrazené na Obrázku 7.



Obrázok 7. Fuzzy množiny A a B z príkladu

Fuzzy množina A je reprezentovaná pomocou trojuholníkovej funkcie príslušnosti. Jej zápis v programe MATLAB je preto A [6 7 8]. Fuzzy množina B je reprezentovaná pomocou lichobežníkovej funkcie príslušnosti. Jej zápis v programe MATLAB je B [5 6 8 9].

Príklad: V reálnych aplikáciách pre nejaký pozorovaný jav často používame pojmy nízky, stredný a vysoký jav. Napríklad, ak uvažujeme teplotu vody, môžeme hovoriť o nízkej teplote, strednej teplote a vysokej teplote (pozri Obrázok 8). Zatiaľ čo pojem stredná teplota (funkcia S) sa dá popísať pomocou lichobežníkovej funkcie príslušnosti, ako bolo uvedené v predchádzajúcom texte, hodnoty nízka teplota (funkcia N) a vysoká teplota (funkcia V) sú špecifické v tom zmysle, že musia byť popísané s použitím asymetrických funkcií príslušnosti. Ako môžeme napísať predpis týchto funkcií v softvéri MATLAB?



Obrázok 8. Fuzzy množiny N, S a V z príkladu

Na popis nejakej fuzzy množiny v softvéri MATLAB môžeme použiť aj parametre, ktoré nepatria do definičného oboru skúmanej premennej. Preto v prvom kroku musíme určiť definičný obor pojmu „teplota vody“. Nech je to X=⟨0,100⟩. Potom môžeme písať

N [–20 – 10 20 40], *S* [20 40 60 80], *V* [60 80 110 120]

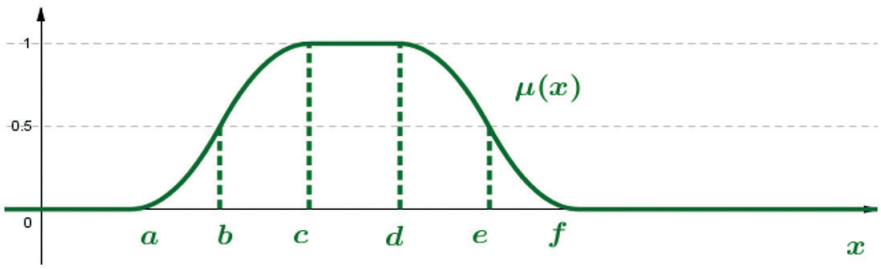
Funkcie príslušnosti založené na polynomických funkciách

Tento typ funkcie je vytvorený pomocou polynomických (konkrétne kvadratických) funkcií. Delia sa do troch základných skupín:

- funkcia príslušnosti typu Pi,
- funkcia príslušnosti typu S,
- funkcia príslušnosti typu Z.

Funkcia príslušnosti typu Pi

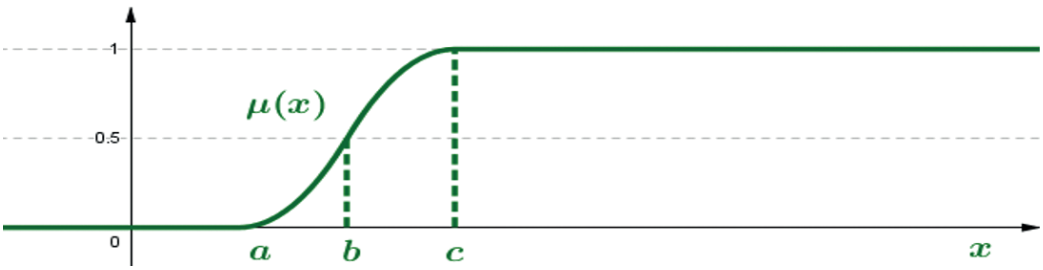
Funkcia príslušnosti typu Pi je definovaná šiestimi **parametrami a, b, c, d, e, f** (pozri Obrázok 9). Má štyri časti, ktoré sú definované kvadratickými funkciami (intervaly ⟨a, b⟩; ⟨b, c⟩; ⟨d, e⟩; ⟨e, f⟩), dve časti, kde je každej vstupnej hodnote priradená hodnota 0 (interval (- ∞, a⟩; (f, ∞)) a jedna časť, kde je každej vstupnej hodnote priradená hodnota 1 (interval ⟨c, d⟩). V softvéri MATLAB sa označuje ako **pimf**.



Obrázok 9. Funkcia príslušnosti typu Pi

Funkcia príslušnosti typu S

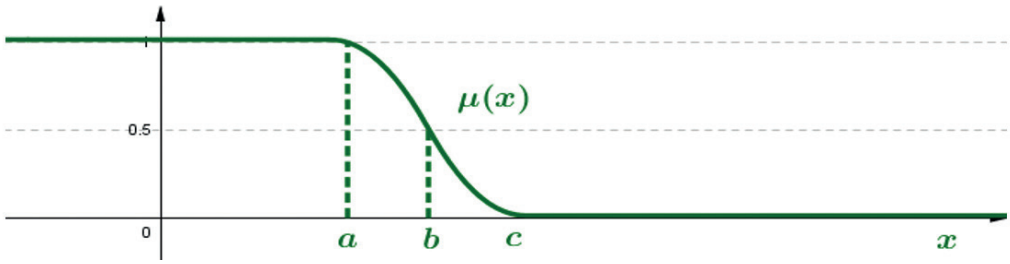
Funkcia príslušnosti typu S je definovaná tromi **parametrami a, b, c** (pozri Obrázok 10). Má dve časti, ktoré sú definované kvadratickými funkciami (intervaly ⟨a,b⟩; ⟨b,c⟩), jednu časť, v ktorej je každej vstupnej hodnote priradená hodnota 0 (intervaly (-∞, a⟩) a jednu časť, kde je každej vstupnej hodnote priradená hodnota 1 (interval ⟨c, ∞)). V softvéri MATLAB sa označuje ako **smf**.



Obrázok 10. Funkcia príslušnosti typu S

Funkcia príslušnosti typu Z

Funkcia príslušnosti typu Z je definovaná tromi **parametrami a, b, c** (pozri Obrázok 11). Má dve časti, ktoré sú definované kvadratickými funkciami (intervaly ⟨a,b⟩; ⟨b,c⟩), jednu časť, kde je každému vstupu priradená hodnota 1 (interval (-∞, a⟩) a jednu časť, kde je každej vstupnej hodnote priradená hodnota 0 (interval ⟨c, ∞)). V softvéri MATLAB sa označuje ako **zmf**.



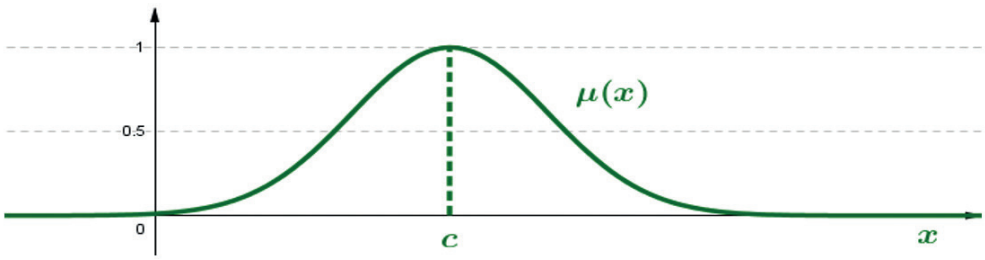
Obrázok 11. Funkcia príslušnosti typu Z

Poznámka:

Funkcie príslušnosti typu S a Z predstavujú asymetrické funkcie príslušnosti. V predchádzajúcom príklade by sme ich mohli použiť na modelovanie nízkych a vysokých hodnôt premennej teplota vody.

Funkcie príslušnosti odvodené zo štatistického prístupu

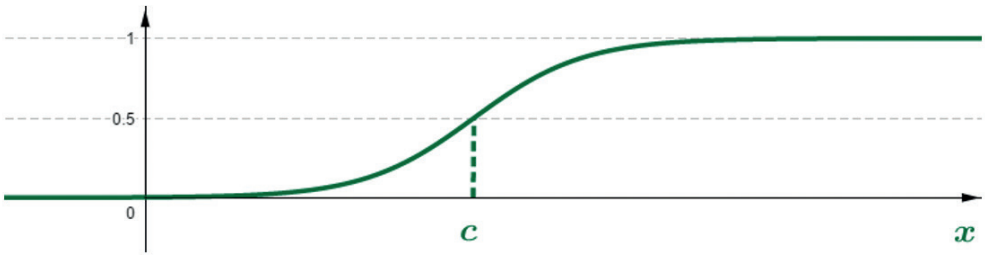
Ak máme veľký súbor údajov, môžeme ho spracovať pomocou štatistického prístupu. **Gaussove funkcie príslušnosti** (**gaussmf**) sú odvodené z klasickej Gaussovej distribučnej krivky, ktorá má dva **parametre c, σ** (pozri Obrázok 12), kde c predstavuje priemer a σ predstavuje štandardnú odchýlku údajov.



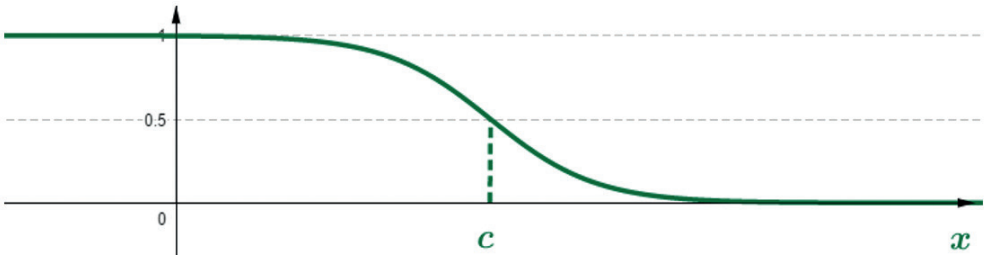
Obrázok 12. Gaussova funkcia príslušnosti

Sigmoidálna funkcia príslušnosti

Gaussove funkcie príslušnosti nedokážu popísať asymetrické funkcie príslušnosti. Z tohto dôvodu sa pri veľkých súboroch údajov používajú na popis asymetrických funkcií príslušnosti **sigmoidálne funkcie príslušnosti** (**sigmf**) s dvoma **parametrami a, c** (pozri Obrázok 13 a Obrázok 14). Parametre a, c sa opäť získajú pomocou štatistického prístupu.



Obrázok 13. Sigmoidálna funkcia príslušnosti pre a > 0



Obrázok 14. Sigmoidálna funkcia príslušnosti pre a < 0

Poznámky:

V ďalších častiach tejto učebnice budeme používať hlavne trojuholníkové a lichobežníkové funkcie príslušnosti. V reálnych aplikáciách sa často používajú aj gaussove a sigmoidálne funkcie príslušnosti a ich parametre sa volia s využitím štatistickej analýzy dát.

V reálnom živote zvyčajne **najprv požiadame odborníka**, aby popísal problém vhodnými funkciami. Potom, **v druhom kroku**, zvyčajne **špecifikujeme parametre** funkcií s **využitím (štatistického) spracovania veľkej skupiny dát**.

Aby sme mohli pracovať s fuzzy množinami, musíme definovať **základné operácie na fuzzy množinách – prienik, spojenie a doplnok**. Podobne, ako existuje veľké množstvo typov funkcií príslušnosti, je definovaných aj niekoľko typov operácií s fuzzy množinami. Spomenieme len takzvané **štandardné operácie** na fuzzy množinách, ktoré **navrhol profesor Zadeh**.

Definícia (štandardný prienik)

Nech X je definičný obor a A, B sú fuzzy množiny. Štandardný prienik dvoch fuzzy množín A, B je fuzzy množina A ∩ B s funkciou príslušnosti

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

Definícia (štandardné zjednotenie)

Nech X je definičný obor a A, B sú fuzzy množiny. Štandardné zjednotenie dvoch fuzzy množín A, B je fuzzy množina A ∪ B s funkciou príslušnosti

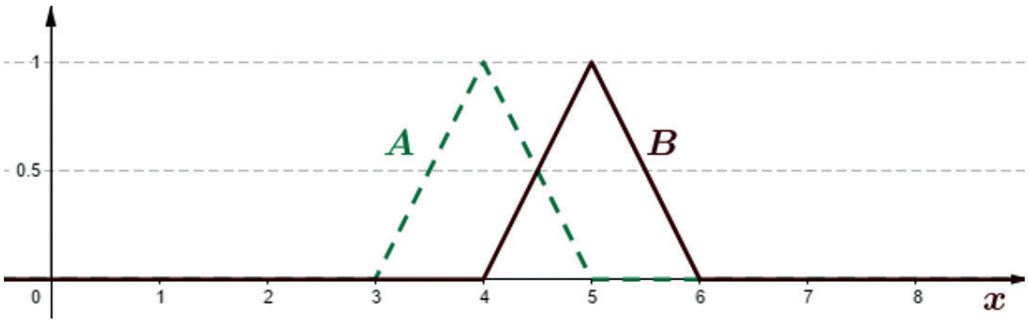
$$\mu_{A\cup B}(x) = \max\big(\mu_A(x), \mu_B(x)\big).$$

Definícia (štandardný doplnok)

Nech X je definičný obor a A je fuzzy množina. Štandardným doplnkom fuzzy množiny A *je fuzzy množina* $\overline{A}$ *s funkciou* príslušnosti

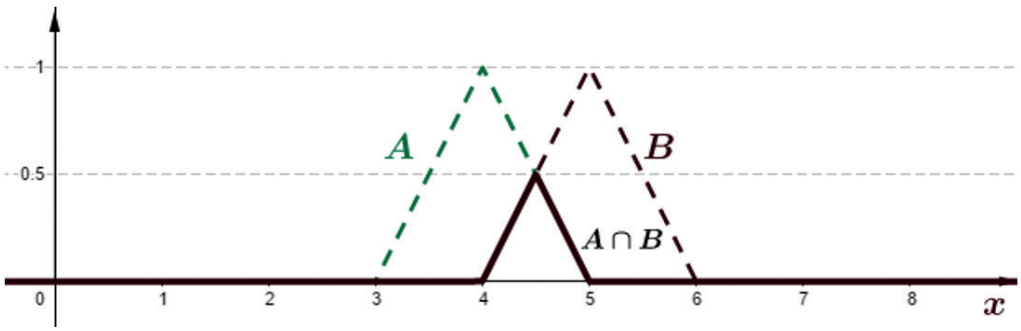
$$\mu_{\bar{A}}(x) = 1 - \mu_A(x).$$

Príklad: Na Obrázku 15 sú zobrazené dve fuzzy množiny A, B. Použitím predchádzajúcich definícií určte graficky prienik a zjednotenie fuzzy množín A, B a tiež doplnok k fuzzy množine A.



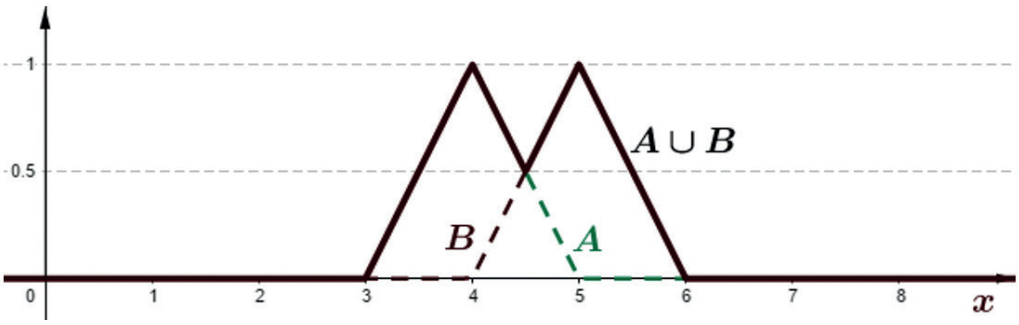
Obrázok 15. Fuzzy množiny A, B z príkladu

Štandardným prienikom dvoch fuzzy množín A, B je fuzzy množina A ∩ B s funkciou príslušnosti $\mu_{A\cap B}(x)=\min(\mu_A(x),\mu_B(x))$. Riešenie je zobrazené na Obrázku 16.



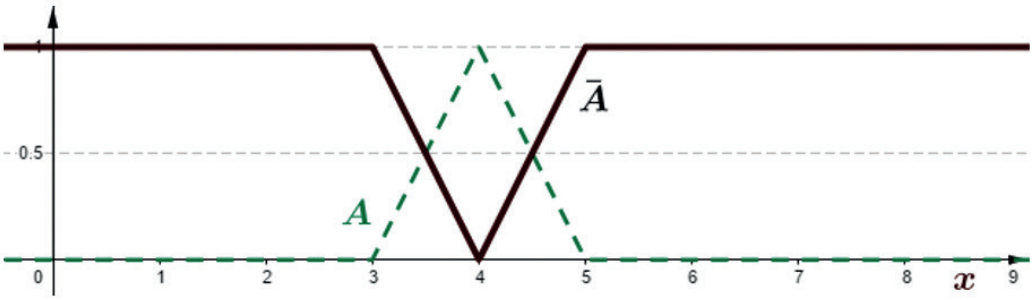
Obrázok 16. Štandardný prienik fuzzy množín A, B z príkladu

Štandardným zjednotením dvoch fuzzy množín A, B je fuzzy množina A ∪ B s funkciou príslušnosti $\mu_{A\cup B}(x)=\max(\mu_A(x),\mu_B(x))$. Riešenie je zobrazené na obrázku 17.



Obrázok 17. Štandardné zjednotenie fuzzy množín A, B z príkladu

Štandardným doplnkom fuzzy množiny $\overline{A}$ je fuzzy množina s funkciou príslušnosti $\mu_{\overline{A}}(x)=1-\mu_{\overline{A}}(x)$. Riešenie je zobrazené na obrázku 18.



Obrázok 18. Štandardný doplnok fuzzy množiny A z príkladu.

KAPITOLA 6

FUZZY ODVODZOVANIE

Autorom tejto časti učebnice je Alžbeta Michalíková z Katedry informatiky, Fakulty prírodných vied, Univerzity Mateja Bela v Banskej Bystrici zo Slovenska.

Fuzzy odvodzovanie je proces, v ktorom odvodzujeme dôsledky na základe informácií, ktoré pozostávajú z vágnych pojmov. Napríklad v reálnom živote často používame pravidlá ako

"Ak je vonku zima, oblečiem sa do teplého oblečenia."

Tieto pravidlá získavame pozorovaním, učením sa, uvažovaním atď. Vo fuzzy odvodzovaní používame takzvané AK - POTOM (IF-THEN) fuzzy pravidlá, ktoré majú nasledujúci tvar:

$$\underbrace{AK\ \langle \ldots \rangle}_{= predpoklad} \qquad \underbrace{POTOM \langle \ldots \rangle}_{= záver}$$

Pre naše potreby si pravidlo upravíme

"Ak je vonku zima, oblečiem sa do teplého oblečenia."

do tvaru:

"AK je Teplota nízka, POTOM Oblečenie je teplé."

Slová „**Teplota**" a „**Oblečenie**" sa nazývajú **jazykové** (**lingvistické) premenné**. Preto ich píšeme veľkými písmenami. Hodnoty „**nízka**" a „**teplé**" sa nazývajú **hodnoty jazykových premenných**. Jazykové premenné, ktoré sa nachádzajú v časti vety za časticou AK, t. j. v predpokladoch, sa nazývajú **vstupné jazykové premenné**. Jazykové premenné, ktoré sú umiestnené v dôsledku takéhoto tvrdenia, sa nazývajú **výstupné jazykové premenné**. Použitím operácií konjunkcie (AND), disjunkcie (OR) a negácie (NOT) môžeme vytvoriť zložitejšie pravidlá, napríklad:

"AK Teplota je nízka A Oblačnosť je vysoká, potom Oblečenie je teplé."

Ak definujeme všetky prípustné pravidlá, dostaneme množinu pravidiel, ktorá sa nazýva **báza pravidiel**. Existujú rôzne prístupy ako vytvoriť bázu pravidiel. Jednu z nich si podrobne vysvetlíme - **Sugenovu metódu** - a následne ukážeme jej použitie na príkladoch.

Sugenova metóda

Autormi tejto metódy sú **T. Takagi**, **M. Sugeno** a **G. Kang** [5]. Navrhli ju v roku 1985. Táto metóda bola navrhnutá na modelovanie systémov, v ktorých je možné opísať závislosť medzi vstupnými a výstupnými premennými funkciou, ktorá je nelineárna, ale niektoré jej časti sú lineárne.

Sugenova metóda bola prvýkrát použitá pri **modelovaní parkovania áut**. Dnes sa používa na aproximáciu údajov nelineárnymi funkciami **v klasifikácii**, **regulácii a riadení**, **fuzzy rozhodovaní**, **expertných systémoch**, ...

V Sugenovej metóde sú hodnoty **vstupných premenných** popísané pomocou **funkcií príslušnosti príslušných fuzzy množín**. Sú navrhnuté odborníkom. **Výstupné premenné** sú opísané funkciami, ktoré môžu byť buď **konštantné**, **lineárne** alebo **polynomické funkcie ľubovoľného stupňa**.

Sugenove pravidlá s konštantnými výstupnými funkciami – výstupná premenná každého pravidla je opísaná funkciou, ktorá je konštantná. Vo všeobecnosti má pravidlo tvar

$$P_j: AK\ X_1\ is\ A_{1j}\ A\ X_2\ is\ A_{2j}\ A\ \dots\ A\ X_n\ is\ A_{nj},\ POTOM\ Y\ is\ b_j.$$

Sugenove pravidlá s lineárnymi výstupnými funkciami - výstupná premenná každého pravidla je opísaná lineárnou funkciou. Vo všeobecnosti má pravidlo tvar

$$P_j: AK\ X_1\ is\ A_{1j}\ A\ \dots\ A\ X_n\ is\ A_{nj},\quad POTOM\ Y\ is\ a_{1j}\,x_1 + \dots + a_{nj}\,x_n + b_j,$$

kde $a_{1j},\dots,a_{nj}$, b_j s sú reálne čísla.

Sugenove pravidlá s polynomiálnymi výstupnými funkciami - výstupná premenná každého pravidla je opísaná polynomickou funkciou ľubovoľného stupňa.

$$P_j: AK\ X_1\ is\ A_{1j}\ A\ \dots\ A\ X_n\ is\ A_{nj},\quad POTOM\ Y\ is\ a_{1j}\,x_1^{m1} + \dots + a_{nj}\,x_n^{mn} + b_j,$$

kde $a_{1j},\dots,a_{nj}$, b_j sú reálne čísla $m_1,\dots,m_n$ a sú prirodzené čísla.

Príklad: Sugenovo pravidlo s funkciou konštantného výstupu
Chceme ohodnotiť študentov pomocou Sugenovej metódy. Potom jedno z pravidiel môže znieť

P: AK Hodnotenie prezentácie je vysoké A Hodnotenie testov je vysoké,

POTOM Celkové hodnotenie je rovné 1 (=A).

Príklad: Sogenove pravidlá s lineárnymi výstupnými funkciami

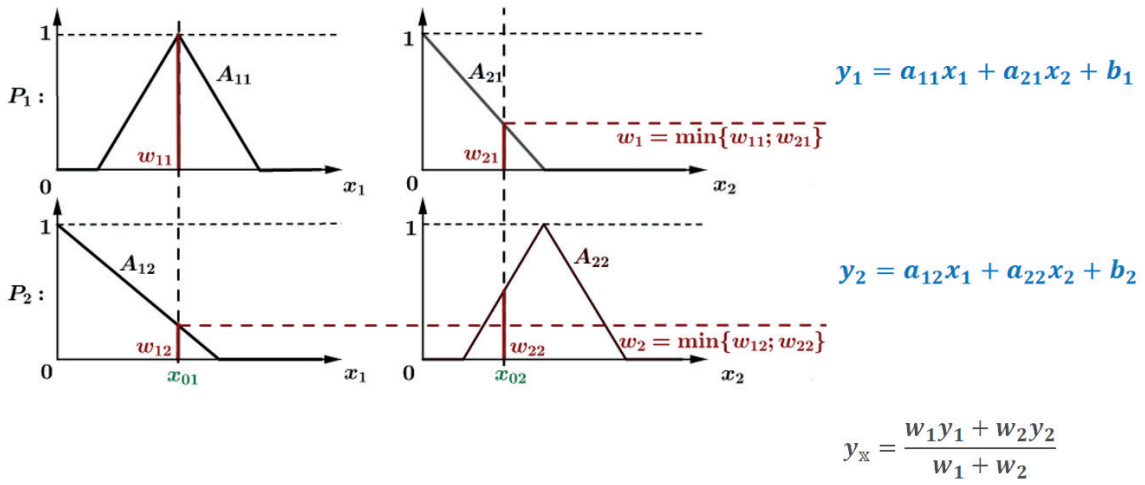
Vieme, že pre niektoré hodnoty polohy auta (napríklad pre malé hodnoty) pôjde auto po priamke, ktorej predpis vieme určiť. Potom pravidlo má nasledujúci tvar

P: AK Hodnota pozície je nízka,POTOM Predpis priamky je 3,25x+2,5.

Majme bázu pravidiel s k pravidlami. Nech modelovaný systém obsahuje n vstupov. ktoré môžeme zapísať ako vektor $\mathbb{X}=(x_1, x_2,\ldots,x_n)$. Nech každé pravidlo obsahuje výstupnú funkciu v tvare $y_j = a_{1j}{x_1}^{m_1} + a_{2j}{x_2}^{m_2} + \cdots + a_{nj}{x_n}^{m_n} + b_j$. Potom sa celkový výstup modelovaného systému $y_{\mathbb{X}}$ vypočítame podľa vzorca

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

kde w_j je váha j-tého pravidla (pozri Obrázok 19).



Obrázok 19. Celkový výstup systému pri použití Sugenovej metódy s dvoma vstupnými premennými a dvoma pravidlami

Poznámka k spôsobu určovania váh:

Majme pravidlo P_j s n vstupnými premennými ($x_1, x_2,\ldots, x_n$). Najprv vypočítame váhy w_{ij} ako priesečník medzi hodnotou nameraného vstupu x_i a príslušnou funkciou príslušnosti A_{1j}. Potom vypočítame celkovú váhu w_j pravidla P_j pomocou nasledujúceho vzorca

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

Ako vhodne navrhnúť fuzzy pravidlá?

- Môžeme požiadať odborníka, aby opísal svoje znalosti pomocou vhodných funkcií.
- Hodnoty parametrov funkcií vieme určiť spracovaním veľkého množstva známych údajov.

Táto metóda má niekoľko názvov, napríklad **Sugenova metóda**, **Takagiho-Sugenov fuzzy inferenčný systém**, **Takagiho-Sugenov regulátor**, ... Tieto názvy predstavujú rovnakú metódu. Použitý názov súvisí s oblasťou, v ktorej sa metóda používa.

Využitie metódy Sugeno predvedieme v dvoch rôznych oblastiach

- v oblasti klasifikácie údajov,
- v oblasti aproximácie údajov.

V oboch prípadoch budeme odborníkmi, ktorí navrhnú pravidlá daného systému [2], [6], [7]. **Na vytvorenie hodnôt vstupných lingvistických premenných použijeme** najjednoduchšie typy funkcií príslušnosti – **lineárne funkcie príslušnosti**. **Na vytvorenie hodnôt výstupných lingvistických premenných použijeme konštantné funkcie na klasifikáciu a lineárne funkcie na aproximáciu**.

KAPITOLA 7

VYUŽITIE SUGENOVEJ METÓDY NA KLASIFIKÁCIU DÁT

Autorom tejto časti učebnice je Alžbeta Michalíková z Katedry informatiky, Fakulty prírodných vied, Univerzity Mateja Bela v Banskej Bystrici zo Slovenska.

V tejto časti učebnice budeme pracovať s datasetom **Iris** (pozri prílohu A). Pripomeňme, že súbor údajov Iris pozostáva zo 150 vzoriek kvetov Iris. Pre každý kvet máme **štyri základné atribúty** - dĺžku a šírku kališných lístkov a dĺžku a šírku okvetných lístkov v centimetroch (alebo milimetroch). Na základe uvedených atribútov chceme zaradiť kvety do troch tried zodpovedajúcich trom druhom kosatcov (**Iris Setosa**, **Iris Virginica** a **Iris Versicolor**).

Na spracovanie údajov uvedených v tejto časti učebnice použijeme softvér Excel a softvér MATLAB.

Príklad: Klasifikujte údaje zo súboru údajov Iris do vhodného počtu tried pomocou Sugenovej metódy. (Riešenie tohto príkladu nájdete v prílohe B.)

Najprv skúsme odpovedať na nasledujúce otázky:

1. *Koľko **vstupných premenných** je v datasete Iris?*
2. *Čo použijeme **na popis vstupných premenných**?*
3. *Aký **typ fuzzy funkcií príslušnosti** použijeme?*
4. *Aký bude **výstup**?*
5. *Čo použijeme **na popis výstupných premenných**?*
6. *Aký **typ pravidiel** použijeme?*
7. *Napíšte **príklad jedného pravidla**!*

Teraz si stiahnite súbor údajov Iris z nejakej webovej stránky a skopírujte ho do Excelovského súboru. **Označte** prvých 50 entít **červenou farbou**, ďalších 50 entít **modrou farbou** a zvyšok **zelenou farbou**. V programe Excel vytvorte štyri nezávislé hárky a skopírujte farebnú tabuľku do každého z nich. V prvom hárku zoraďte hodnoty (od najmenšej po najväčšiu) podľa prvého stĺpca. Podobne zoraďte hodnoty v každom hárku podľa jedného zo stĺpcov. Na modelovanie vstupných premenných použijeme

lichobežníkové funkcie. Z týchto údajov určte hodnoty parametrov vstupných premenných a vyplňte ich do nasledujúcich tabuliek (Tabuľka 1).

Tabuľka 1. Parametre vstupných premenných

Vstup1:

Názov	Parametre
Definičný obor	
Červená	
Modrá	
Zelená	

Vstup2:

Názov	Parametre
Definičný obor	
Červená	
Modrá	
Zelená	

Vstup3:

Názov	Parametre
Definičný obor	
Červená	
Modrá	
Zelená	

Vstup4:

Názov	Parametre
Definičný obor	
Červená	
Modrá	
Zelená	

V ďalšom kroku určte hodnoty výstupných parametrov. Vyplňte Tabuľku 2 správnymi hodnotami, ak uvažujete, že pre výstupnú jazykovú premennú použijete **konštantné funkcie**.

Tabuľka 2. Parametre výstupnej premennej

Výstup:

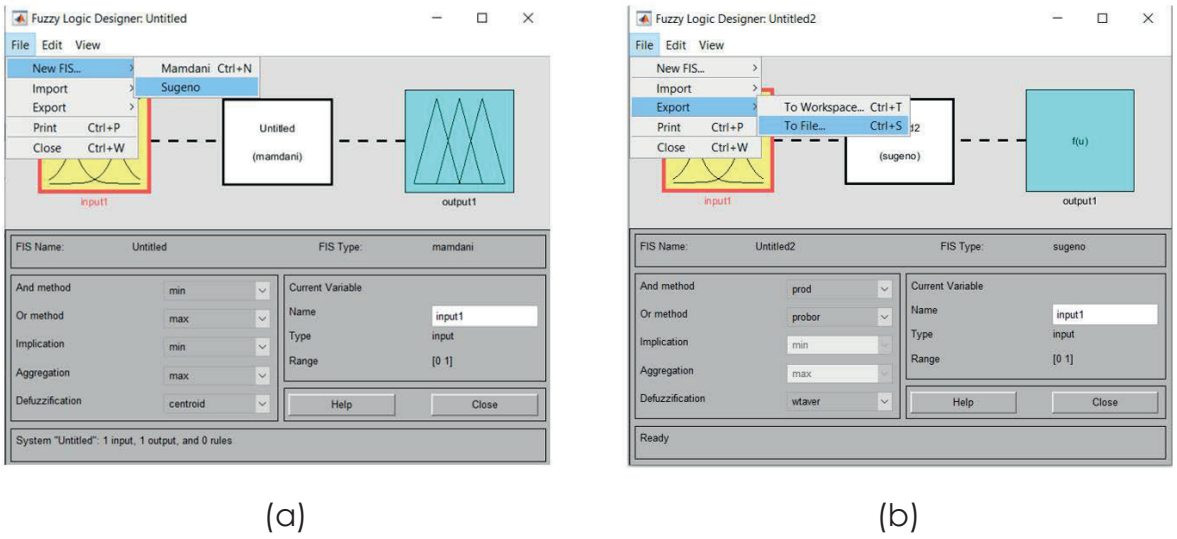
Názov	Parametre
Definičný obor	
Červená	
Modrá	
Zelená	

Navrhnite počet pravidiel, ktoré použijete a napíšte ich v správnom tvare.

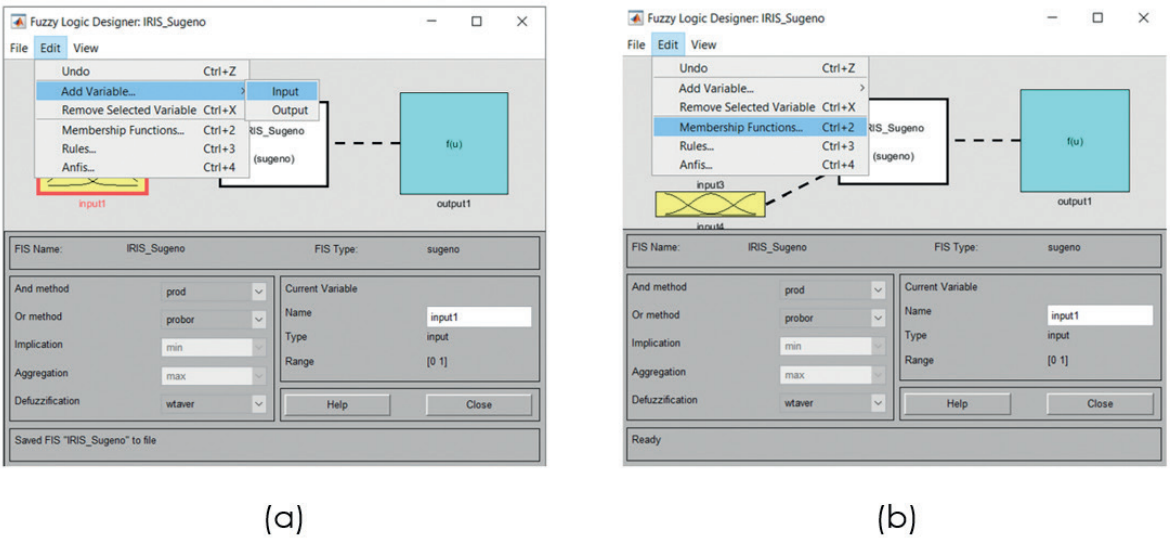
Pravidlá:

Teraz získané hodnoty spracujeme v softvéri MATLAB. Otvorte softvér MATLAB a do príkazového okna napíšte príkaz **fuzzy**. Tento príkaz otvorí grafické prostredie pre prácu s fuzzy množinami. Použijeme metódu Sugeno (Sugeno fuzzy inference system = Sugeno FIS), preto musíme otvoriť tento typ FIS (pozri Obrázok 20a). Tento FIS môžeme premenovať a uložiť napríklad ako súbor **IRIS_Sugeno** (pozri Obrázok 20b). Potrebujeme pridať štyri vstupné jazykové premenné – jednu už máme, preto pridáme tri nové **vstupné premenné** (pozri Obrázok 21a) a následne upravíme parametre ich funkcií prísluš-

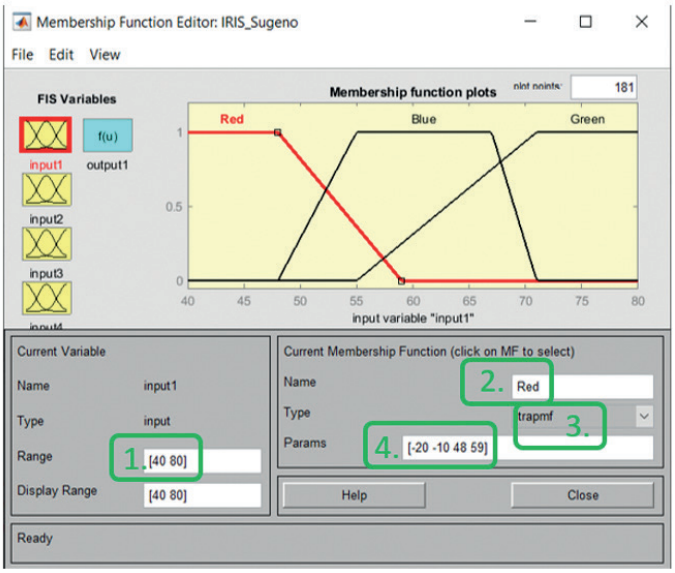
nosti (pozri Obrázok 21b). Teraz pre každú vstupnú premennú postupne zmeníme rozsah premennej, pridáme názvy funkcií príslušnosti, zmeníme typ funkcií príslušnosti a pridáme parametre ku každej funkcii príslušnosti (použijeme Tabuľku 1). Tieto kroky sú znázornené na Obrázku 22.



Obrázok 20. Otvorenie nového Sugeno FIS (a) a premenovanie/uloženie FIS (b) v softvéri MATLAB

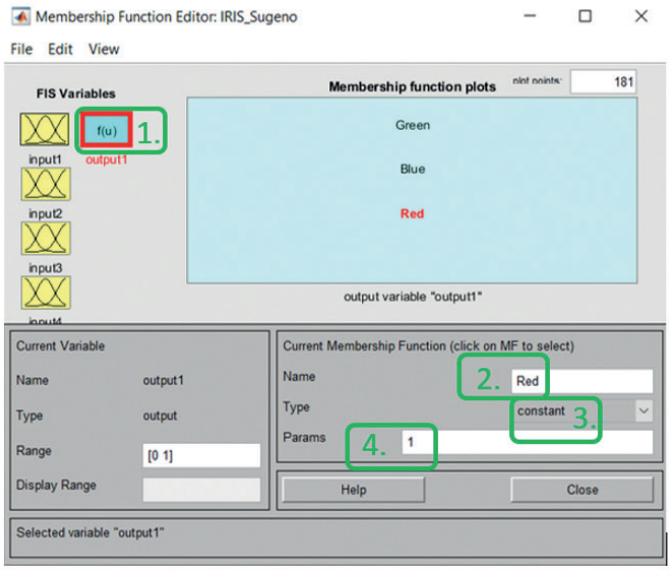


Obrázok 21. Pridanie nových premenných (a) a úprava funkcií príslušnosti (b) v softvéri MATLAB



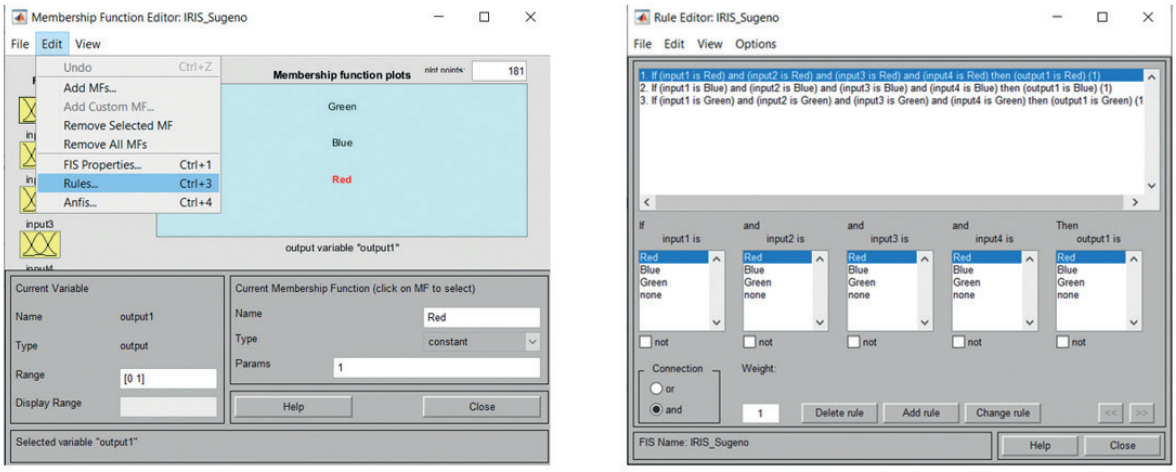
Obrázok 22. Zmena parametrov vstupnej funkcie príslušnosti v softvéri MATLAB

Teraz zmeníme hodnoty **výstupnej premennej**. Na úpravu výstupnej premennej použijeme dvojité kliknutie na modrý obdĺžnik s názvom output1. Získame nové menu pre výstupnú premennú, ako je znázornené na Obrázku 23. Do tohto menu naplníme hodnoty z Tabuľky 2.



Obrázok 23. Zmena parametrov výstupných hodnôt v softvéri MATLAB

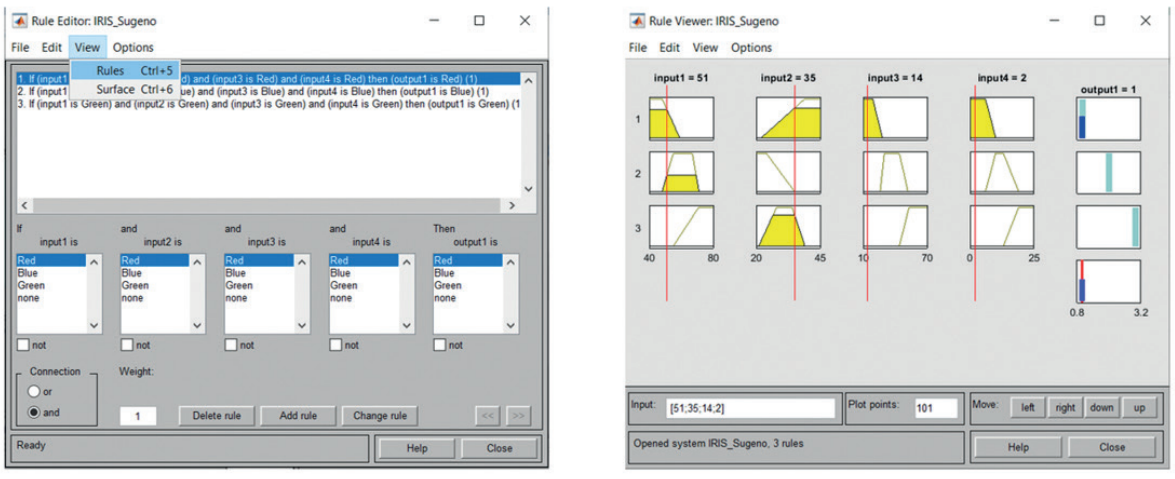
Naším posledným krokom je **vytvorenie pravidiel** nášho systému. Otvoríme menu pravidiel (pozri Obrázok 24a) a použijeme tri jednoduché pravidlá (pozri Obrázok 24b).



(a) (b)

Obrázok 24. Otvorenie menu na tvorbu pravidiel (a) a pridanie pravidiel (b) v softvéri MATLAB

Náš **systém je pripravený**. Teraz **môžeme vyhodnotiť výsledky**, ktoré systém dáva pre známe vstupy. Môžeme otvoriť prehliadač pravidiel (pozri Obrázok 25a) a pridať špecifickú hodnotu ku každej vstupnej premennej (pozri Obrázok 25b). Tieto hodnoty môžeme pridať posunutím červených čiar v hornej časti ponuky alebo zmenou hodnôt parametrov usporiadanej štvorice v spodnej časti ponuky.



(a) (b)

Obrázok 25. Otvorenie prehliadača pravidiel (a) a pridanie konkrétnych hodnôt do vstupných parametrov (b) v softvéri MATLAB

Poznámky:

Vstupné hodnoty zobrazené na obrázku 25b prislúchajú prvému riadku tabuľky datasetu IRIS. Ako vidíme, systém zaradil objekt s týmito vstupnými atribútmi do triedy 1 (výstup1 = 1). Nakoľko trieda 1 prislúcha druhu Iris Setosa, je to hodnota, ktorú sme vo výsledku očakávali!

Ukázali sme, ako môžeme klasifikovať jeden objekt z dátového súboru IRIS. Tento prístup môžeme použiť pre každý riadok tabuľky. Všetky riadky tabuľky môžeme samozrejme klasifikovať aj v jednom kroku pomocou postupnosti príkazov z príkazového riadku. Vieme tiež vypočítať **úspešnosť klasifikácie** pri použití daného FIS. Použitím takých parametrov pre vstupné a výstupné funkcie, ako sú uvedené v prílohe B, sme dosiahli úspešnosť 94,6667 %, t. j. 142 kvetov zo všetkých 150 kvetov bolo klasifikovaných správne.

Úspešnosť klasifikácie je možné zlepšiť použitím niekoľkých rôznych prístupov. Napríklad môžeme použiť viac hodnôt vstupných jazykových premenných a následne vytvoriť viac pravidiel. V prezentovanom príklade sme použili tri hodnoty pre každú zo vstupných premenných (**červená – modrá – zelená**). Mohli by sme použiť aj päť hodnôt každej vstupnej premennej, ktoré predstavujú hodnoty **veľmi_malá_hodnota – malá_hodnota – stredná_hodnota – vysoká_hodnota – veľmi_vysoká_hodnota**. Potom môžeme kombináciou hodnôt týchto vstupných premenných vytvoriť ďalšie pravidlá. Na druhej strane môžeme použiť iné metódy, ktoré boli určené na optimalizáciu parametrov hodnôt vstupných a tiež výstupných premenných. Jedným z nich je takzvaný **Adaptívny Neuro-Fuzzy Inferenčný Systém** = ANFIS (angl. Adaptive Neuro-Fuzzy Inference System), ktorý optimalizuje parametre vytvoreného FIS s využitím neurónovej siete. Základné poznatky o neurónových sieťach sú uvedené v ďalšej časti tejto učebnice.

KAPITOLA 8

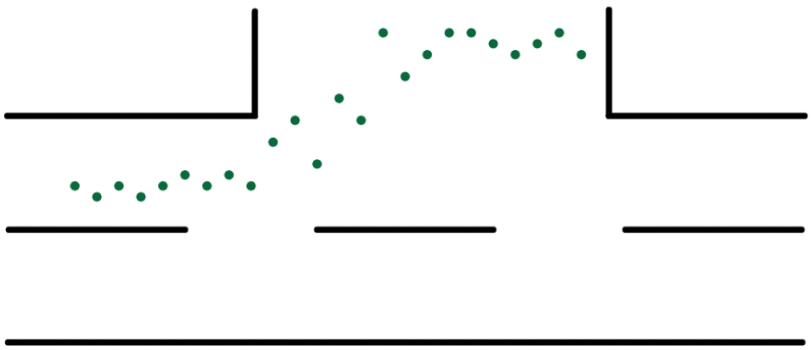
VYUŽITIE SUGENOVEJ METÓDY NA APROXIMÁCIU DÁT

Autorom tejto časti učebnice je Alžbeta Michalíková z Katedry informatiky, Fakulty prírodných vied, Univerzity Mateja Bela v Banskej Bystrici zo Slovenska.

Pojem aproximácia znamená nahradenie presnej hodnoty jej približnou hodnotou. V mnohých prípadoch máme veľké množstvo údajov, ktoré potrebujeme spracovať. Často je užitočné **aproximovať** takéto údaje nejakou jednoduchšou funkciou, ktorá poskytuje aproximáciu skutočného výstupu pre presné vstupné hodnoty. **Sugenova metóda** bola navrhnutá na aproximáciu takých údajov, ktoré sa na niektorých častiach definičného oboru dajú popísať pomocou lineárnej funkcie (v 2D prípade sa dajú aproximovať časťou priamky) a na zvyšnej časti definičného oboru ich treba aproximovať nejakou vhodnou nelineárnou funkciou. V tejto časti textu si priblížime spôsob spracovania údajov, ktoré predstavujú dráhu pohybu automobilu.

Na spracovanie údajov uvedených v tejto časti učebnice použijeme softvér Excel a softvér MATLAB.

Príklad: Predstavme si, že vyvíjame autonómne vozidlo. Jedným z problémov, ktoré musíme vyriešiť, je nájsť funkciu, ktorá bude popisovať zaparkovanie automobilu na nejaké parkovacie miesto. Pri riešení tohto problému môžeme požiadať profesionálneho vodiča, aby niekoľkokrát zaparkoval na určitom konkrétnom mieste, a môžeme zachytiť dráhu automobilu v určitých časových okamihoch pomocou senzorov (pozri Obrázok 26).



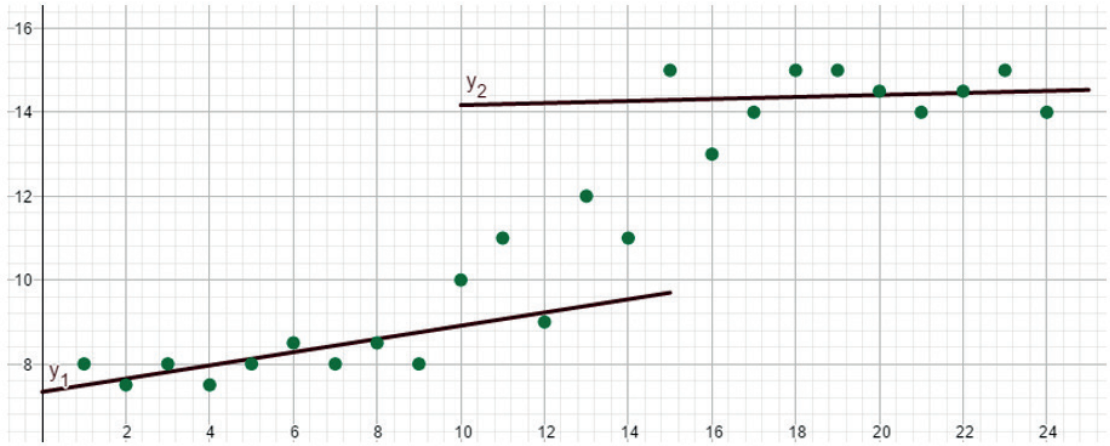
Obrázok 26. Poloha automobilu počas procesu parkovania

Riešenie: Uvedený pohyb automobilu možno opísať v dvoch častiach pomocou priamok. Prvá časť – pohyb po rovnej ceste pred parkovaním. Druhá časť - pohyb na parkovisku. Pohyb medzi týmito dvoma časťami budeme aproximovať pomocou Sugenovej metódy.

V prvom kroku umiestnime získané údaje do karteziánskeho súradnicového systému (pozri Tabuľku 3 a Obrázok 27). Môžeme tiež nakresliť priamky, ktoré predstavujú priamočiary pohyb a pomenovať ich a . Ako vidíme, existuje niekoľko bodov, ktoré prispejú k popisu práve jednej priamky (body s ich hodnotou x z intervalov ⟨1,10⟩ a ⟨15,24⟩) a tiež údaje, ktoré prispejú k popisu dvoch priamok (body s ich hodnotou x z intervalu ⟨10,15⟩). Táto informácia je dôležitá, keď navrhujeme funkcie príslušnosti použitej fuzzy množiny.

Tabelul 3. Súradnice aproximovaných údajov

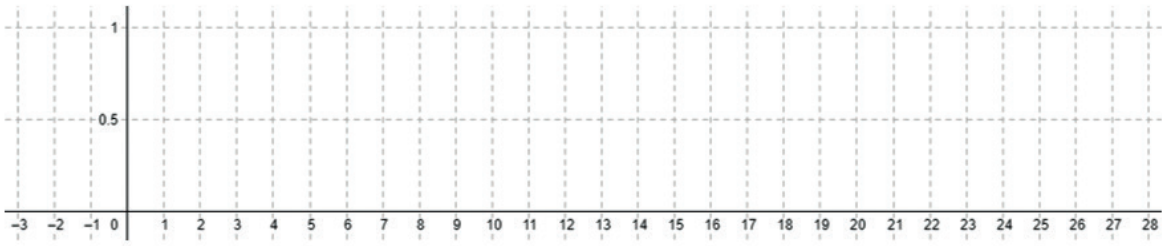
x	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*
y	*8*	*7,5*	*8*	*7,5*	*8*	*8,5*	*8*	*8,5*	*8*	*10*	*11*	*9*
x	*13*	*14*	*15*	*16*	*17*	*18*	*19*	*20*	*21*	*22*	*23*	*24*
y	*12*	*11*	*15*	*13*	*14*	*15*	*15*	*15*	*14*	*15*	*15*	*14*



Obrázok 27. Umiestnenie nameraných údajov do karteziánskeho súradnicového systému

Odpovedzme na nasledujúce otázky:

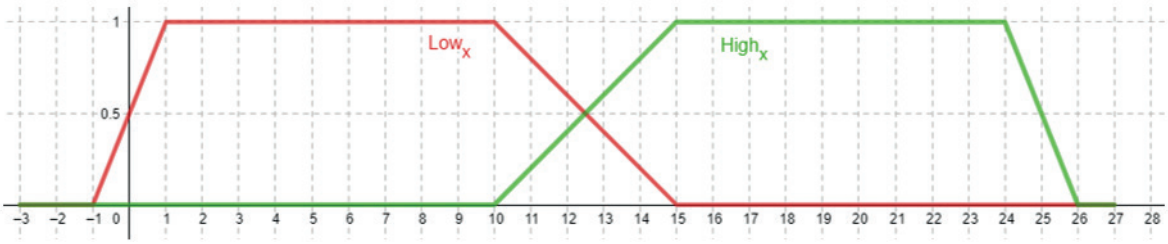
1. **Koľko máme vstupných premenných**? Pomenujte tieto premenné!
2. **Koľko hodnôt vstupných premenných použijeme**? Pomenujte tieto hodnoty premenných!
3. **Čo použijeme na popis vstupných premenných**?
4. **Aký typ fuzzy funkcií príslušnosti** použijeme?
5. Viete nakresliť tieto funkcie príslušnosti? Použite nasledujúcu mriežku:



6. Viete napísať **definičný obor** a **parametre týchto funkcií**? Môžete napísať predpis týchto funkcií v softvéri MATLAB?
7. **Čo bude výstupom systému**?
8. Čo použijeme na **popis výstupných premenných**?
9. **Koľko pravidiel použijeme**?
10. **Napíšte príklad jedného pravidla**!

Odpovede:

Existuje **len jedna vstupná premenná** – môžeme ju nazvať **Poloha** (auta) **na osi x**. Táto vstupná premenná má dve hodnoty – **Nízka hodnota súradnice x** a **Vysoká hodnota súradnice x**. Popíšeme ich pomocou fuzzy množín. Použijeme lichobežníkového funkcie príslušnosti. Môžu byť nakreslené napríklad tak, ako je to zobrazené na Obrázku 28.



Obrázok 28. Lichobežníkové funkcie príslušnosti na aproximáciu údajov z príkladu

Definičným oborom týchto fuzzy funkcií je $\mathbb{X}$=[1,24]. P Pre hodnotu **Nízka hodnota súradnice x** máme parametre *Nízke x*=[-1,1,10,15]. Pre hodnotu **Vysoká hodnota súradnice x** máme parametre *Vysoké x*=[10,15,24,26].

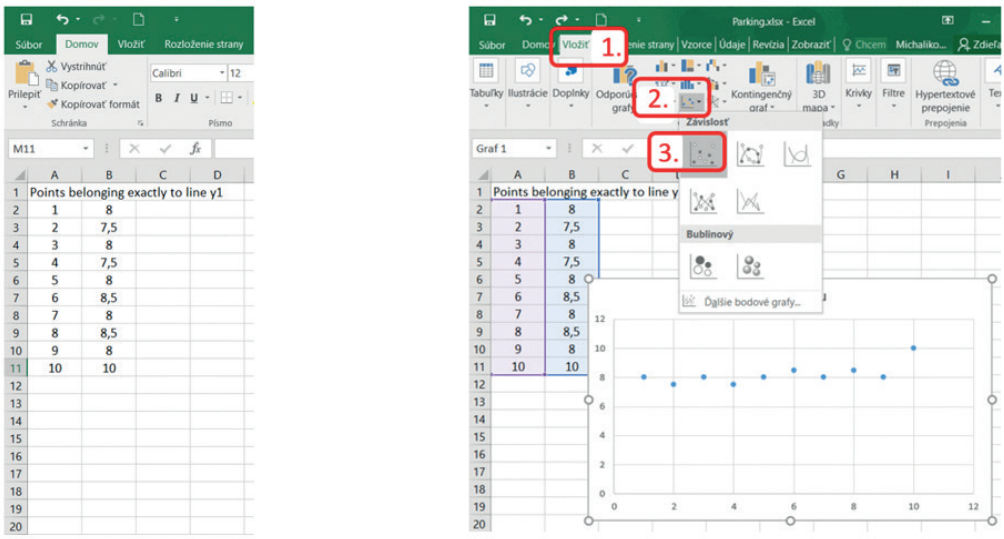
Výstupná premenná predstavuje **Polohu auta na osi y**. Ako výstup použijeme lineárnu funkciu (priamku). Použijeme dva riadky y_1 a y_2. Na popis parametrov týchto priamok použijeme softvér Excel (pozri nižšie). Potom budeme mať dve pravidlá typu AK-POTOM, ktoré zapíšeme takto:

R1: AK Poloha auta na osi x je *Nízke x*, POTOM Poloha auta na osi y je y_1.

R2: AK Poloha auta na osi x je *Vysoké x*, POTOM Poloha auta na osi y je y_2.

Potrebujeme vypočítať parametre lineárnych funkcií, ktoré predstavujú výstup jednotlivých pravidiel. Tieto parametre budú vypočítané z tých hodnôt dát, ktoré prispievajú k popisu práve jednej priamky, t. j. k popisu priamky y_1 prispieva prvých 10 dátových bodov. Skopírujeme si ich do Excelovského

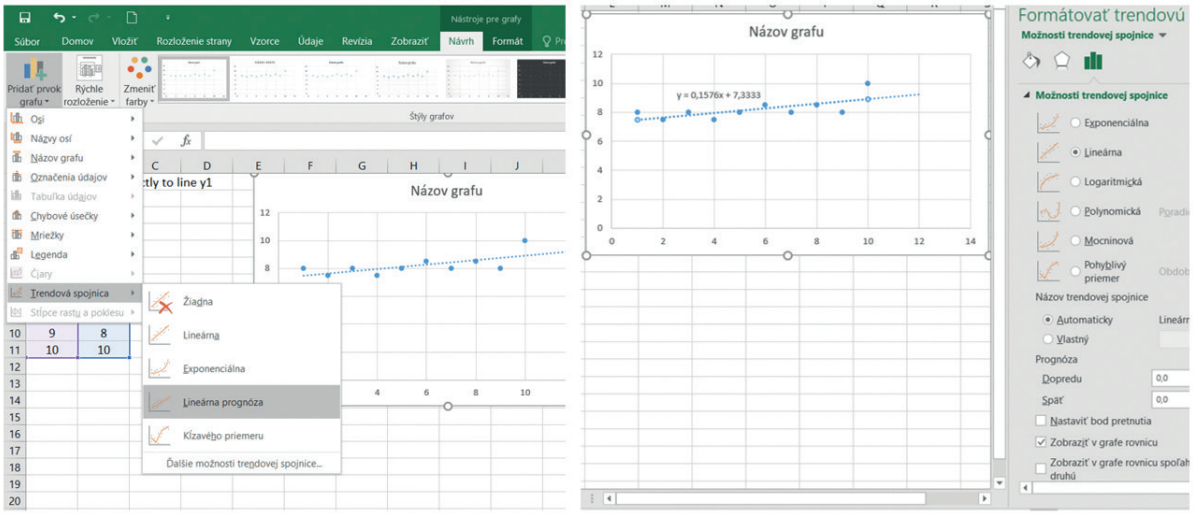
súboru – každý bod do jedného riadku (pozri Obrázok 29a). Potom označíme tieto body a použijeme nasledujúcu postupnosť krokov **Vložiť** → **Graf** → **Bodový** (Obrázok 29b).



(a) (b)

Obrázok 29. Spracovanie vstupných údajov v softvéri Excel

Potom pridáme prvok grafu podľa Obrázka 30a a zobrazíme rovnicu priamky (pozri Obrázok 30b).



(a) (b)

Obrázok 30. Zobrazenie rovnice priamky v programe Excel

Získali sme parametre pre priamku y_1. Rovnakým postupom získame parametre pre priamku y_2. Potom y_1=0,1576x + 7,3333 a y_2= 0,0242x + 13,927. V programe MATLAB zapíšeme tieto parametre ako usporiadané dvojice nasledovného tvaru y_1 [0,1576 7,3333] a y_2 [0,0242 13,927].

Súhrn všetkých hodnôt vstupných a výstupných parametrov pre aproximáciu údajov týkajúcich sa parkovania automobilu je uvedený v Tabuľke 4.

Tabuľka 4. Súhrn vstupných a výstupných parametrov pre aproximáciu údajov

Input:

Názov	Parametre
Definičný obor	[1, 24]
Nízke_x	[−1, 1, 10, 15]
Vysoké_x	[10, 15, 24, 26]

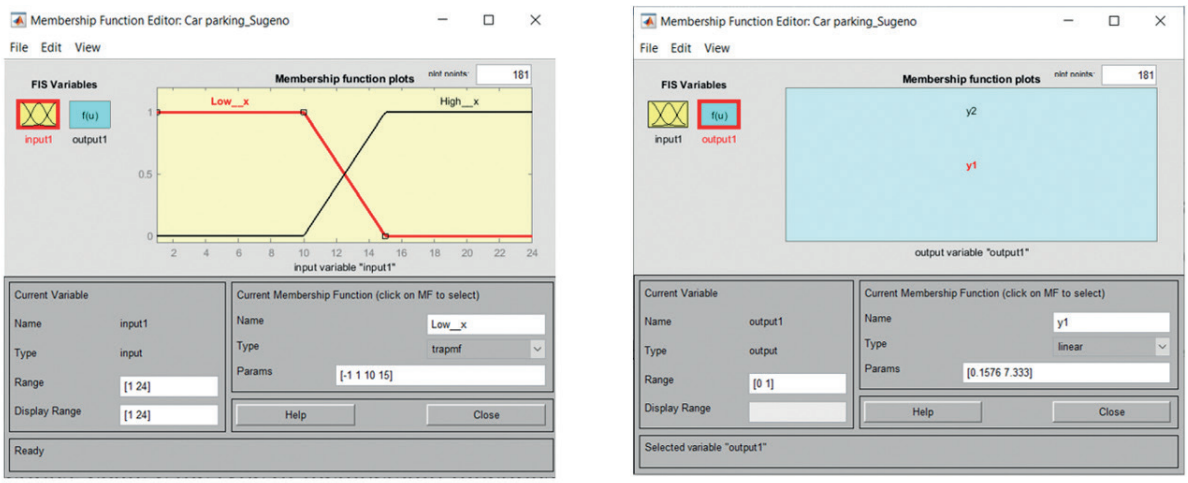
Output:

Názov	Parametre
Definičný obor	---
y_1	[0,1576 7,3333]
y_2	[0,0242 13,927]

Už sme určili všetky parametre, preto môžeme vytvoriť FIS typu Sugeno v softvéri MATLAB. Prvé kroky sú podobné tým, ktoré boli uvedené v kapitole 7 (klasifikácia údajov). Otvorte softvér MATLAB a do príkazového riadku napíšte príkaz **fuzzy**. Tento príkaz otvorí grafické prostredie pre prácu s fuzzy množinami. Použijeme metódu Sugeno (Sugeno fuzzy inference system = Sugeno FIS). Preto otvoríme tento typ FIS (pozri Obrázok 20a) a zároveň ho premenujeme a uložíme (pozri Obrázok 20b), napríklad ako súbor **Parking_Sugeno**.

Máme len jednu **vstupnú jazykovú premennú**. Pre túto premennú máme dve funkcie príslušnosti. Program MATLAB nám ale ponúka tri funkcie príslušnosti, preto ak chceme jednu z nich odstrániť, stačí kliknúť na jednu z funkcií v grafe a použiť tlačidlo Delete. Potom upravíme parametre funkcií príslušnosti (použijeme hodnoty z Tabuľky 4). Konečná konfigurácia pre vstupné funkcie príslušnosti je zobrazená na Obrázku 31a.

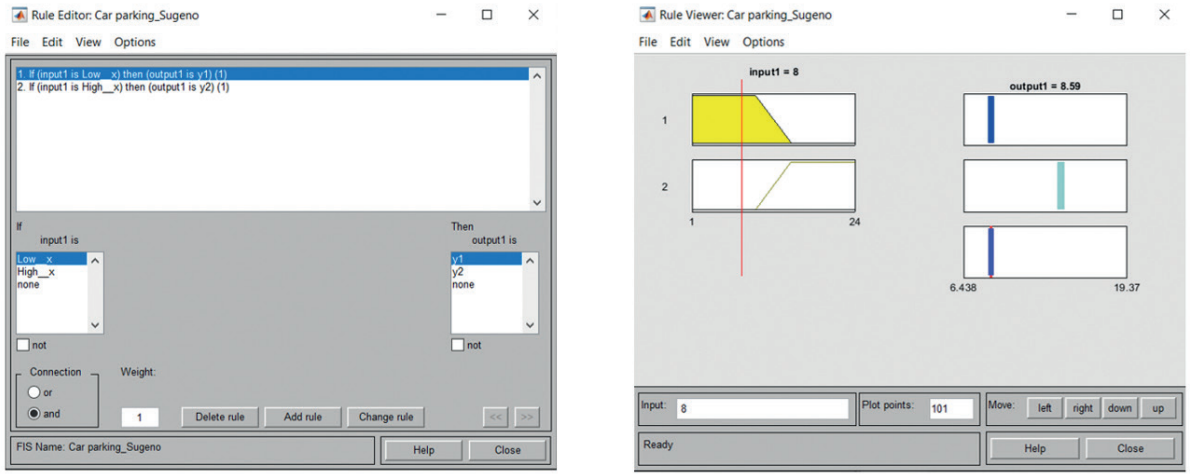
Teraz musíme zmeniť **hodnoty výstupnej premennej**. Na úpravu výstupnej premennej opäť použijeme dvojklik na modrý obdĺžnik s názvom output1. Dostaneme menu pre výstupnú premennú. Podobne, ako to bolo pri tvorbe predchádzajúceho FIS, vyplníme všetky hodnoty (z Tabuľky 4). Nezabudnime, že v tomto FIS je typ výstupnej funkcie lineárny (pozri Obrázok 31b).



(a) (b)

Obrázok 31. Konfigurácia vstupných a výstupných premenných v softvéri MATLAB

Naším posledným krokom je vloženie pravidiel pre vytvorený systému. Použijeme dve jednoduché pravidlá (pozri Obrázok 32a). Náš systém je pripravený. Teraz môžeme vyhodnotiť výsledky, ktoré systém dáva pre známe vstupy. Môžeme otvoriť prehliadač pravidiel a pridať do vstupnej premennej jej špecifickú hodnotu (pozri Obrázok 32b).



Obrázok 32. Nastavenie pravidiel a vyhodnotenie výsledkov v softvéri MATLAB

Existuje tiež možnosť vykresliť funkciu, ktorú sme vytvorili pomocou tohto FIS (pozri Obrázok 33).

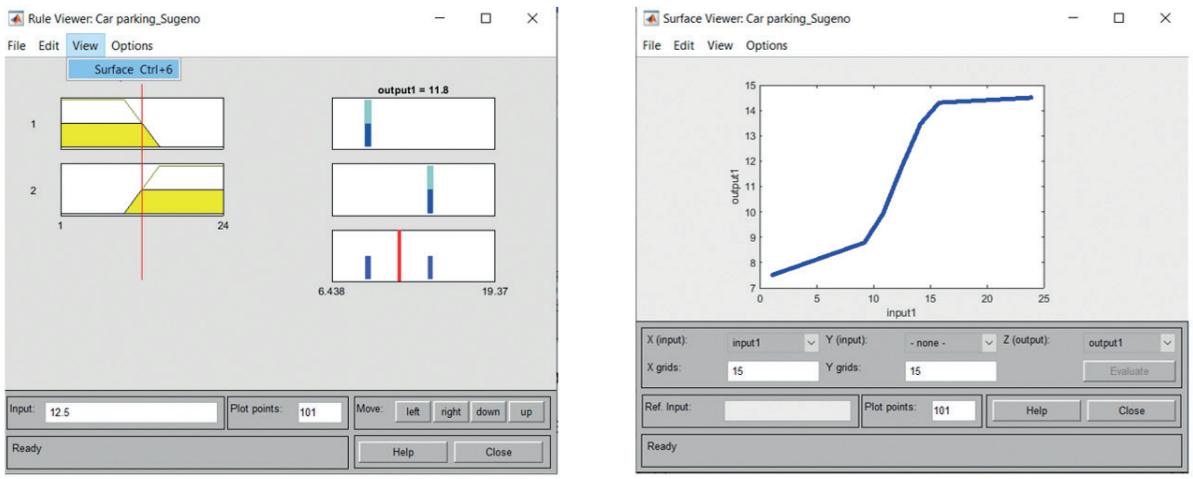
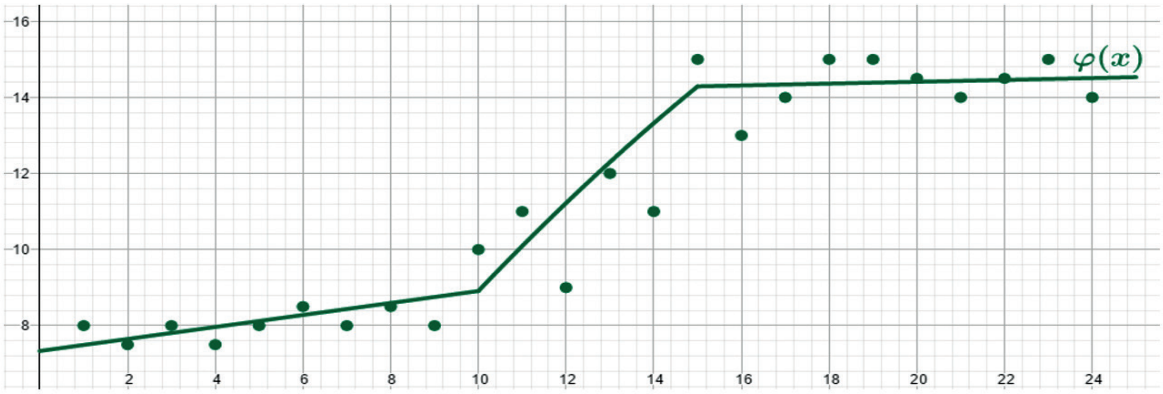


Figura 33. Otvorenie prehliadača výsledných funkcií a výsledná funkcia v softvéri MATLAB

Poznámky:

Vstupná hodnota zobrazená na Obrázku 32b sa rovná 8. Ako môžeme vidieť, systém dal tejto vstupnej hodnote výstup 8,59. Skutočná hodnota (pozri Tabuľku 3) bola 8,5. Získaná hodnota preto predstavuje dobrú aproximáciu tohto bodu.

Pôvodné (reálne) dáta môžeme porovnať so získanou funkciou. Existujú dva základné **prístupy k porovnávaniu** (hodnoteniu) **výsledkov**. **Prvý prístup je grafický**, pričom do jedného obrázka vykreslíme pôvodne zadané údaje aj získanú funkciu. **Druhý prístup predstavuje výpočet chyby vytvoreného systému**. Na Obrázku 34 je grafické porovnanie reálnych údajov a získanej funkcie.



Obrázok 34. Grafické porovnanie reálnych údajov a získanej funkcie

Ukázali sme, ako môžeme získať približnú hodnotu výstupu k jednej konkrétnej vstupnej hodnote (Obrázok 32b). Samozrejme, takýmto spôsobom môžeme aproximovať všetky vstupy, ktoré sú uvedené v Tabuľke 3. Môžeme to spraviť aj v jednom kroku pomocou postupnosti príkazov z príkazového riadku. Následne môžeme vypočítať celkovú chybu systému. Existuje viac typov chýb, ktoré sa používajú na porovnanie získaných výsledkov. Najpoužívanejšou chybou je takzvaná **stredná kvadratická odchýlka** - **MSE** (angl. **Mean Square Error**). Táto chyba sa vypočíta podľa vzorca

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

kde n predstavuje počet vstupných údajov, hodnoty $f(x_i)$ predstavujú skutočné výstupy a $\varphi(x_i)$ predstavujú výstupy vypočítané pomocou FIS.

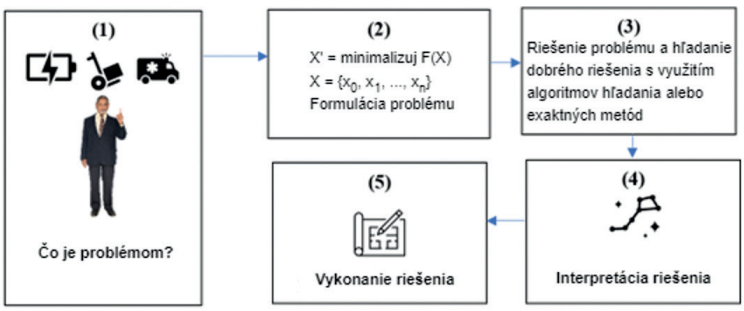
Kvalitu aproximácie možno zlepšiť použitím niekoľkých rôznych prístupov. Napríklad môžeme použiť viac funkcií príslušnosti a potom vytvoriť viac pravidiel. V prezentovanom príklade sme použili dve funkcie príslušnosti (**Nízke x** – **Vysoké x**). Môžeme použiť tri hodnoty vstupnej premennej, ktoré predstavujú hodnoty **Nízke x** – **Stredné x** – **Vysoké x**. Potom nájdeme predpis 3 priamok a vytvoríme tri pravidlá kombináciou vstupných a výstupných hodnôt. Keď máme veľkú množinu vstupov, môžeme použiť aj iné typy funkcií príslušnosti (boli spomenuté v Kapitole 5). Určenie parametrov funkcií potom môže byť dané štatistickým rozložením údajov. Na druhej strane môžeme opäť použiť inú metódu, ktorá bola navrhnutá na optimalizáciu parametrov hodnôt vstupných, ale aj výstupných premenných.

KAPITOLA 9

ÚVOD DO OPTIMALIZÁCIE

Autorom tejto časti učebnice je Fatih Kilic z Adanskej univerzity vied a technológií v Adane z Turecka.

Problém optimalizácie je riešený v mnohých oblastiach štúdia s cieľom nájsť optimálne riešenie v stavovom priestore (priestore obsahujúcom všetky možné riešenia) pomocou matematických a heuristických metód. Existujú rôzne optimalizačné problémy, ako sú inžinierske, finančné, medicínske a výrobné problémy. Obrázok 1 prezentuje hlavné kroky riešenia optimalizačných problémov. V prvom kroku je potrebné vyriešiť optimalizačný problém s cieľom zlepšiť súčasné systémy alebo navrhnúť nové systémy. Chceme napríklad umiestniť nemocnicu do najlepšej pozície vzhľadom na dopyt a potenciálny počet pacientov. Po druhé, tento problém musí byť matematicky formulovaný ako štruktúra riešenia, cieľ a obmedzenia. Štruktúra riešenia pozostáva z rozhodovacích premenných. V príklade uvedenom vyššie by rozhodovacie premenné mohli byť možné pozície kandidátskych nemocníc pre tento problém. Objektívna funkcia meria kvalitu riešenia na vyhodnotenie medzi kandidátskymi riešeniami – pri prípade nemocníc môže byť takouto funkciou súčet vzdialeností medzi nemocnicami a potenciálnymi pacientmi. Všetky riešenia môžu byť uskutočniteľné alebo neuskutočniteľné z dôvodu vopred definovaných obmedzení. Tieto obmedzenia definuje odborník. Pre nemocničný problém môže byť najmenej jedna nemocnica v podoblasti, ktorú požadujú zainteresované strany. V troch krokoch sa implementujú známe metódy na nájdenie dobrých riešení. Tieto metódy vytvárajú optimálne riešenie alebo dobré riešenia blízke optimálnemu riešeniu. Zainteresované strany interpretujú riešenia a v prípade potreby vykonajú menšie revízie riešenia. Nakoniec sa riešenie realizuje.



Obrázok 1. Hlavné kroky riešenia optimalizačných problémov

Z pohľadu matematickej definície môže byť akýkoľvek optimalizačný problém definovaný ako:

$$\max_{x \in F \subseteq \mathbf{S}} / \min f(x), \qquad \text{Eq. (1)}$$

kde x predstavuje množinu rozhodovacích premenných, F obsahuje uskutočniteľné riešenia, S reprezentuje stavov priestor a $f(x)$ je objektívna funkcia. *max/min* majú za cieľ nájdenie maximálnej alebo minimálnej hodnoty $f(x)$.

Teraz môžeme sformulovať obmedzenia a rozsah údajov a premenných. Príklad môže byť definovaný nasledovne:

$$\sum_{j}^{n} x_j < \mathrm{b} \qquad \text{Eq. (2)}$$

$$x_j \in \{0,1\} for\ j = 1 \ldots . n \qquad \text{Eq. (3)}$$

kde x_j môže byť *0* alebo *1* pre všetky *j* a suma všetkých *x* je menej ako *b*.

Problémy, ktoré hľadajú spojité premenné sú klasifikované ako problémy spojitej optimalizácie. Tabuľka 1 ukazuje dobre známe problémy spojitej optimalizácie. Riešenie (X) pozostáva z D-rozmerných reálnych hodnôt. Každý rozmer je medzi vopred definovaným minimálnym a maximálnym číslom. Dimenzia je množstvo rozhodovacích premenných, ktoré sú používané na demonštráciu ich výkonu pri zavádzaní optimalizačných algoritmov.

Tabuľka 1 – Unimodálne funkcie

Dimension	Range	Equation
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]	$F_7(x) = \sum_{i=1}^{n} i x_i^4 + \mathrm{random}\,[0, 1]$

9.1 ALGORİTMY LOKÁLNEHO HĽADANİA

Algoritmy lokálneho hľadania sa v informatike a príbuzných výpočtových vedách používajú na riešenie optimalizačných problémov. Tieto algoritmy sú definované ako zovšeobecnené heuristické vyhľadávacie algoritmy a môžu byť implementované pre rôzne optimalizačné problémy po ich sformulovaní.

Algoritmy lokálneho hľadania sa zvyčajne zaoberajú jediným riešením, s cieľom kedykoľvek vytvoriť lepšie riešenie. Simulované žíhanie, hľadanie s Tabu listom, Horolezecká metóda a iné sú dobre známe algoritmy lokálneho hľadania.

Algoritmus 1 zobrazuje hlavné kroky Horolezeckého algoritmu.

Algoritmus 1: Horolezecký algoritmus

```
1  aktuálneRiešenie = generuj prvé riešenie
2  vyhodnoť aktuálneRiešenie
3  iterácia =0
4  pokiaľ (!ukončovaciePodmienky)
5      susednéRiešenie = posun(aktuálneRiešenie)
6      AK susednéRiešenie lepšie ako aktuálneRiešenie
7          aktuálneRiešenie = susednéRiešenie
8      UKONČI AK
9      iterácia = iterácia + 1
```

V prvom kroku je náhodne generované počiatočné riešenie problému, ktoré je priradené k aktuálnemu riešeniu. Napríklad *X= [60.15, -50.07, 10.08, -80.01, 17.59]* pre funkciu *F1* v tabuľke 1. Každá položka tohto vektora je medzi *-100* a *+100* a je vybraná náhodne. Po druhé, susedné riešenie je generované pomocou aktuálneho riešenia a funkcie parciálnych úprav v každej iterácii. Sú vybrané náhodné čísla indexu a vybratá položka sa zmení na vektor *X*. Ak je susedské riešenie lepšie ako súčasné riešenie, aktuálne riešenie je aktualizované susedným riešením. Iterácie sú vykonávajú, kým nie je splnená ukončovacia podmienka alebo dosiahnutý maximálny počet iterácií algoritmu.

9.2 EVOLUČNÉ ALGORİTMY

Evolučné algoritmy (EA) sú populárnym optimalizačným a populačným algoritmom, ktorý napodobňuje biologickú evolúciu, ako je reprodukcia, kríženie, mutácia, selekcia a prežitie jedincov. Rôzne varianty EA sa zavádzajú pomocou procesov biologickej evolúcie. Genetický algoritmus bol navrhnutý Johnom Hollandom v roku 1962, zatiaľ čo evolučné stratégie navrhol Ingo Rechenberg o tri roky neskôr.

Kroky typického EA sú dané v Algoritme 2.

Algoritmul 2: Calcul evolutiv	
1	populácia = generuj náhodné počiatočné riešenie
2	iterácia =0
3	pokiaľ (!ukončovaniePodmienky)
4	hodnoty tzv. fitness funkcie sú počítané pre každého jedinca z populácie
5	na základe fitness funkcie sú jedinci vyberaní ako rodičia v danej populácii
6	na generovanie potomkov sú použité kríženie a mutácia
7	aktualizuj populáciu na základe nových potomkov a ich fitness hodnôt
8	iterácia = iterácia + 1

V prvých krokoch algoritmu sa populácia generuje náhodne na vopred definovanú veľkosť. Populácia pozostáva z riešení problému, pričom každý jedinec predstavuje jedno riešenie. Druhým krokom je súbor iteračných procesov. V druhom kroku je pomocou objektívnej funkcie počítaná tzv. fitness hodnota pre každého jedinca. Rodičia sú vyberaní na základe ich kondície alebo rôznych iných techník. Kríženie a mutácia sú reprodukčné procesy na generovanie nových riešení. Pre ďalšiu generáciu sa výberový proces vykonáva na základe fitness hodnôt jednotlivcov. Tieto kroky sa opakujú, kým nie sú splnené podmienky zastavenia.

Operátor kríženia

Operátor kríženia si vymieňa informácie medzi dvoma vybranými chromozómami rodičov tak, aby vygeneroval dvoch nových potomkov. Operátor kríženia je dôležitým exploratívnym operátorom v evolučných algoritmoch. Existujú rôzne všeobecné techniky kríženia, ako napríklad jednobodové, viacbodové, uniformné kríženia a techniky kríženia špecifické pre daný problém (pre problémy kombinatorickej optimalizácie), ako napríklad rekombinácia hrán, čiastočne mapované kríženie viacerých rodičov a kríženie založené na poradí. Tento operátor je vykonávaný podľa pravdepodobnosti kríženia.

Tabuľka 2 predstavuje príklad operátora jednobodového kríženia. Rodičia 1 a 2 sú vybraní jednotlivci, pričom v prvom a druhom riadku sú uvedené dve riešenia kurzívou a podčiarknuté. Bod rezu je vybraný náhodne a rodičia sú rozdelení na dve časti pre každého jednotlivca. Deti 1 a 2 sú generované tak, aby si vymenili druhé časti rodičov a vzali rovnaké prvé časti rodičov.

Tabuľka 2. Príklad jednobodového operátora kríženia

	X_1	X_2	X_3	X_4	X_5
Părinte 1	60.15	-50.07	10.08	-80.01	17.59
Părinte 2	40.22	30.08	20.09	-20.05	60.85
Copil 1	60.15	-50.07	20.09	-20.05	60.85
Copil 2	40.22	30.08	10.08	-80.01	17.59

Operátor mutácie

Na zabezpečenie diverzity v populácii je používaný operátor mutácie. Tento operátor modifikuje rodiča tak, aby produkoval potomstvo – je vybraná náhodná poloha riešenia a zodpovedajúci gén (alebo bit) sa zmení – je vykonaná operácia mutácie. Existujú rôzne operátory mutácií. Jedným z operátorov mutácie je globálna mutácia, ktorá súčasne aktualizuje multipozíciu jednotlivcov.

Vzorka mutácie je uvedená v tabuľke 3. X_3 sa vyberie náhodne, použije sa tzv. flip-flop metóda, čo znamená, že nová hodnota X_3 je 0.

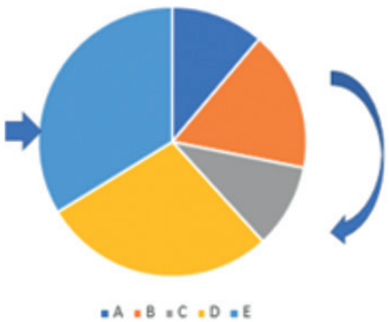
Tabuľka 3. Príklad operátora mutácie

	X_1	X_2	X_3	X_4	X_5
Individ	*1*	*0*	***1***	*0*	*1*
Individ nou	*1*	*0*	***0***	*0*	*1*

Stratégie výberu sa používajú na zvýšenie pravdepodobnosti prežitia jedincov a potomkov s vyššou fitness hodnotou v ďalšej generácii a na výber rodičov. Výber ruletou a výber turnajom patria medzi najobľúbenejšie stratégie výberu.

Výber ruletou: Kruhové koleso pozostáva z *n* koláčov, kde *n* je počet riešení v populácii. Každé riešenie dostane časť koláča na základe jeho fitness hodnoty. Je vybraný bod na obvode kolesa a koleso sa roztočí.

Riešenie	Fitness hodnota
A	10
B	15
C	9
D	28
E	30



Obrázok 2. Vzorka populácie

Výber turnajom: V tomto prístupe, "*turnaj, k*" je náhodne vybraných *k*-jednotlivcov z populácie. Následne je vybraný jeden z nich (s najlepšou fitness hodnotou) pomocou turnaja.

9.3 RIEŠENIE PROBLÉMU BATOHU

Problém batohu spočíva v tom, že do definovaného batohu chceme vložiť jednu z množiny definovaných položiek s hmotnosťou a hodnotami s cieľom maximalizácie celkovej hodnoty uloženej v batohu. Tabuľka 4 ukazuje dataset testovacích dát pre problém batohu.

Tabuľka 4. Testovací dataset pre problém batohu

	Položka 1	Položka 2	Položka 3	Položka 4	Položka 5	Položka 6	Položka 7
Hmotnosť	30	20	10	45	15	33	25
Hodnota	10	5	30	16	50	13	13
Príklad riešenia	1	0	1	1	0	1	1

V rámci problému batohu používame nasledovné značenie, parametre a rozhodovacie premenné.

Značenie:

j: index pre položky, *j*

Parametre:

v_j: hodnota položky *j*

w_j: hmotnosť položky *j*

W: maximálna kapacita batohu

Rozhodovacie premenné:

$$x_j = \begin{cases} 1, \text{v prípade, že položka } j \text{ bola vybraná do batohu} \\ 0, \text{v inom prípade} \end{cases}$$

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

$$\text{za podmienok} = \sum_{j=1}^{J} w_j x_j < W$$

Kód fitness funkcie:

```
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
```

Kód genetického algoritmu:

```
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);
```

KAPITOLA 10

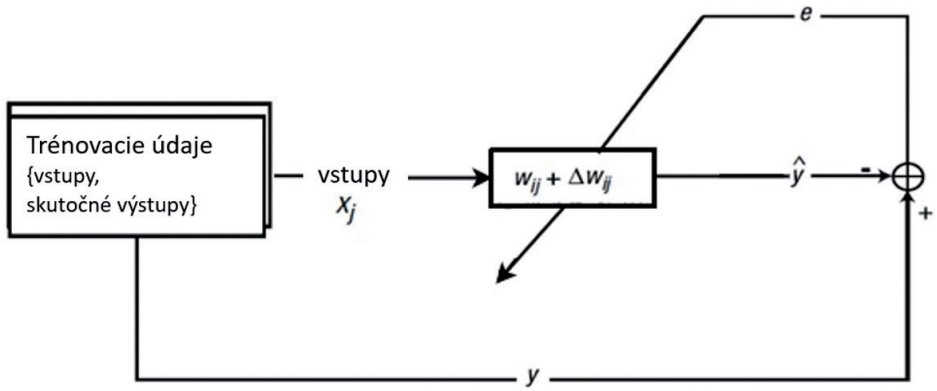
JEDNOVRSTVOVÉ NEURÓNOVÉ SIETE

Autorom tejto časti učebnice je Onder Tutsoy z Adanskej univerzity vied a technológií v Adane z Turecka.

Neurónové siete (angl. Neural Networks - NN) uchovávajú informácie vo forme váh získaných buď z učenia pod dohľadom – tzv. **učenia s učiteľom**, ktoré slúži na rozpoznávanie vzorov, alebo z **učenia bez učiteľa**, napríklad pri aproximácii funkcií. NN sú v podstate neparametrické modelovacie prístupy používané na približnú reprezentáciu reálnych systémov. Preto je jeho analytická (fundovaná a presná matematická) analýza náročná. Pri trénovaní NN, sa váhy aktualizujú na základe informácií, ktoré sú poskytnuté prostredníctvom vstupov. Systematický prístup používaný na aktualizáciu váh sa nazýva **pravidlo učenia**, ktoré využíva poskytnuté vstupné informácie. V podstate mapuje vstupné informácie na výstupné informácie. Keďže učenie je pre NN jediným spôsobom, ako systematicky ukladať a zapamätať si informácie, pravidlo učenia je dôležitou súčasťou procesu učenia, o ktorom budeme hovoriť ďalej.

10.1 DELTA PRAVIDLO

Pravidlo delta je reprezentatívnym pravidlom učenia sa jednovrstvových NN. Tréningový proces jednovrstvového NN je možné znázorniť tak, ako je to uvedené na Obrázku 1.



Obrázok 1. Bloková schéma tréningového procesu jednovrstvovej NN

Dôležité je poznamenať, že jednovrstvovú NN môže tvoriť jeden vstup s jedným výstupom (angl. single input single output - SISO), jeden vstup s viacerými výstupmi (angl. single input multiple output SIMO), s viacerými vstupmi s jedným výstupom (angl. multiple input single output MISO) alebo s viacerými vstupmi a viacerými výstupmi (angl. multiple input multiple output MIMO). Počet vstupov a výstupov sa mení v závislosti od charakteru riešeného problému. Poznamenajme tiež, že dynamické procesy je možné modelovať iba NN, ktoré majú viac ako jeden vstup alebo výstup. Ak však problém učenia nie je spojený, ale problém učenia NN je konštruovaný ako spojený, potom sa účinnosť NN zníži. Preto je potrebné najskôr analyzovať charakter vstupných údajov a na základe získaných poznatkov o vstupných údajoch zostaviť NN.

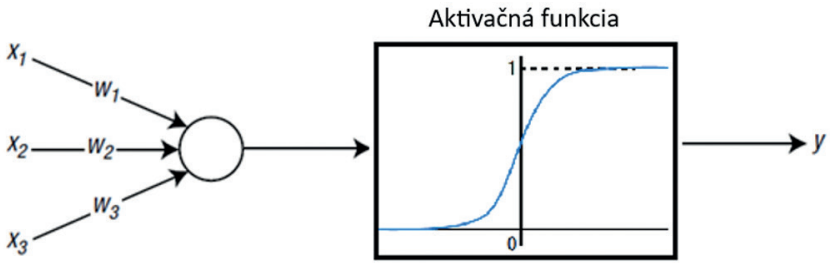
Pseudokód pre delta pravidlo učenia pre *m* vstupov a *n* výstupov v neurónovej sieti:

1. **Vstup**: Tréningový vstup je označený $x \in \mathbb{R}^{mxl}$, kde *l* je dĺžka každého čísla *m* vstupných údajov, $y \in \mathbb{R}^{lxn}$, predstavuje výstup, kde *n* je číslo výstupu, $w \in \mathbb{R}^{mxn}$, sú náhodne inicializované neznáme parametre matice/vektora, $\hat{y}_o \in \mathbb{R}^{nxl}$ je očakávaný výstup, $\hat{y} \in \mathbb{R}^{nxl}$ je odhadovaný výstup, $e \in \mathbb{R}^{lxn}$ je chyba trénovania, $0<\eta\leq 2/x(:,1)^T\, x(:,1)$ predstavuje rýchlosť učenia parametrov. Počet viacerých simulácií je označený ako *simMultiple*, matica, do ktorej sa ukladajú váhy w_s, vektor, do ktorého sa ukladá chyba e_s, prah zastavenia chyby je onačený ako e_t, vektor, do ktorého sa ukladá odhadovaný výstup je označený ako $\hat{y}_o$.
2. **Výstup**: Výsledná hodnota trénovaných parametrov *w;* posledná uložená chyba učenia e_s a posledný uložený výstup y_s.
3. pre *i* po *simMultiple*
4. pre *j* po *l*
5. 1. Vypočítajte aktuálny odhadovaný výstup $\hat{y}_o$
6. $\hat{y}_o(:,j) = w^T x(:,j)$
7. 2. Použite prahovú hodnotu σ na výstup (ak je to potrebné)
8. $\hat{y}(:,j) = \sigma(\hat{y}_o(:,j))$
9. 3. Určte chybu
10. $e(:,j) = y - \hat{y}(:,j)$
11. 4. Aktualizujte a uložte neznámy parameter
12. $w \rightarrow w + \eta e(:,j)\, x(:,j)^T$
13. $w_s = [w_s; reshape(w_s, 1, [])]$
14. koniec j
15. Uložte chybu a aktuálny odhadovaný výstup
16. $e_s = [e_s; reshape(e, 1, [])]$
17. $\hat{y}_s = [\hat{y}_s; reshape(\hat{y}, 1, [])]$
18. Ak $e(:,j) < e_t$ potom
19. výstup z vetvenia ak
20. koniec ak
21. koniec *i*

Delta pravidlo aktualizuje neznáme parametre iteračne (postupne v cykle), namiesto toho, aby ich riešilo naraz. Ide o typ numerickej iteračnej metódy využívajúcej gradientovú metódu najstrmšieho spádu (gradient descent). Gradientová metóda začína od počiatočnej hodnoty a postupne pokračuje smerom k riešeniu. Jej názov pochádza z jej správania, pri ktorom hľadá riešenie, ako keby sa guľa kotúľala z kopca po najstrmšej ceste. V tejto analógii je poloha gule aktuálnym výstupom z modelu a dno gule je hľadaným riešením. Je zaujímavé, že táto iteratívna metóda nemôže „hodiť guľu" na jej dno iba jedným hodom. Celý proces sa niekoľko krát opakuje, preto opätovné trénovanie modelu s rovnakými údajmi môže model vylepšiť.

Príklad: Delta pravidlo

Uvažujme NN, ktorá pozostáva z troch vstupných uzlov a jedného výstupného uzla, ako je znázornené na Obrázku 2.



Obrázok 2. – NN pozostávajúca z troch vstupných uzlov a jedného výstupného uzla

Ako je možné vidieť na obrázku 2, sigmoidálna funkcia sa používa na aktivačnú funkciu výstupného uzla. Majme štyri tréningové body, ako je uvedené v nasledujúcej tabuľke.

Tabuľka 1 - Tréningové body funkcie ALEBO s označením výstupu

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

V tomto príklade používame tzv. učenie s učiteľom, preto každý údajový bod pozostáva zo vstupno-výstupného páru. Posledné tučné číslo každého súboru údajov predstavuje správny výstup. Toto je problém funkcie ALEBO, pričom posledná hodnota vstupu (posledná tenká hodnota) predstavuje šum a je nastavená na hodnotu 1.

Keďže používame jednovrstvovú NN, ktorá obsahuje jednoduché tréningové dáta, kód výpočtu nie je zložitý. Keď si pozorne prečítate kód (za znakom % je zakaždým uvedený slovný komentár k uvedenému kódu), pochopíte správanie sa NN pri učení.

Príslušný kód prebieha takto:

Na začiatku sú tréningové parametre definované ako vo funkcii **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
```

Po zadefinovaní tréningových parametrov sa na proces učenia použije nasledujúca funkcia.

```
% Tento m-súbor trénuje jednu vrstvu NN pre problém ALEBO

% Nahrajme tréningové parametre

trainPar = trainParameters();

% Nahrajme pridelenú chybu

e = trainPar.e;

% Nahrajme pridelený odhadovaný výstup

yo_hat = trainPar.yo_hat;

% Nahrajme pridelený výstup s prahom
```

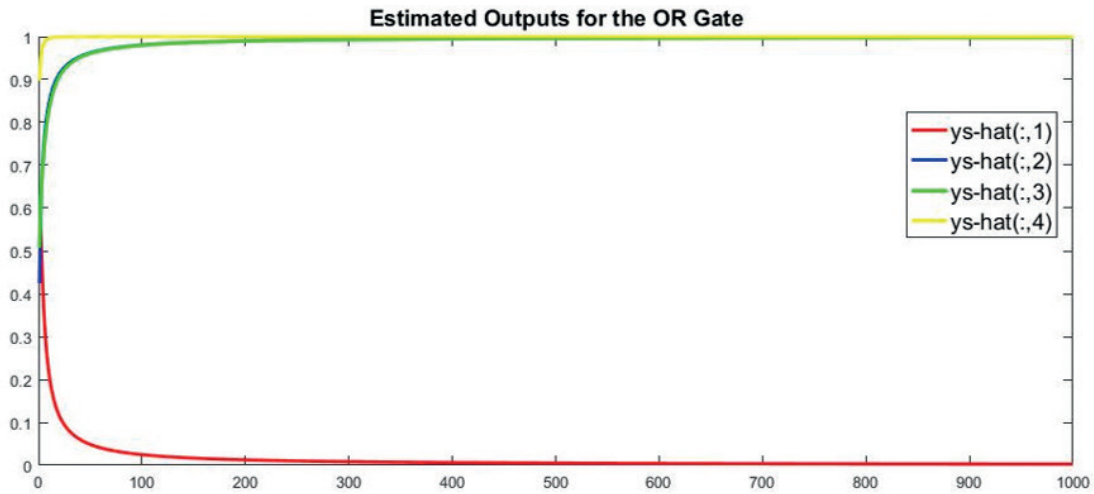
```
y_hat = trainPar.y_hat;
% Nahrajme pridelený neznámy parameter
w = trainPar.w;
% Nahrajme pridelenú rýchlosť učenia
mu = trainPar.mu;
% Zaveďme maticu pre uloženie neznámych parametrov
ws = [];
% Zaveďme maticu pre uloženie chyby
es = [];
% Zaveďme maticu pre uloženie odhadovaného výstupu
ys_hat = [];
for i=1:trainPar.simMultiple
  for j=1:size(trainPar.x,2)
     % Vypočítajme odhadovaný výstup
     yo_hat(j,:) = w'*trainPar.x(:,j);
     % Použijeme prahovú hodnotu pre odhadovaný výstup
     y_hat(j,:) = satOutput(yo_hat(j,:),trainPar);
     % Určíme aktuálnu chybu
     e(j,:) = trainPar.y(j,:) - y_hat(j,:);
     % Aktualizujeme rýchlosť učenia
     mu(i,j) =trainPar.mur/(trainPar.x(:,j)'*trainPar.x(:,j));
     % Aktualizujeme neznámy parameter vektor/maticu
     w = w + mu(i,j)*e(j,:)*trainPar.x(:,j);
     % Uložíme neznámy parameter vektor/maticu
     ws = [ws;reshape(w,1,[])];
  end
  % Uložíme históriu chýb
  es = [es;reshape(e,1,[])];
  % Uložíme odhadovaný výstup
  ys_hat = [ys_hat;reshape(y_hat,1,[])];
end
```

```
Nech funkcia satOutput slúži na vytvorenie sigmoidálnej aktivačnej funkcie,
potom ďalej postupujeme nasledovne:

function y_sat = satOutput(y_unsat, trainPar)

y_sat = trainPar.satUppper / (1 + exp(-y_unsat));

end
```

Potom môžeme odhadované výstupy **ys_hat** pre každú sadu vstupov vykresliť pomocou nasledujúceho kódu:

```
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('Odhadované výstupy pre funkciu ALEBO')
```

Spustením uvedeného kódu získame nasledujúci obrázok:



Obrázok 3. Výsledky odhadovaných výstupov pre funkciu ALEBO

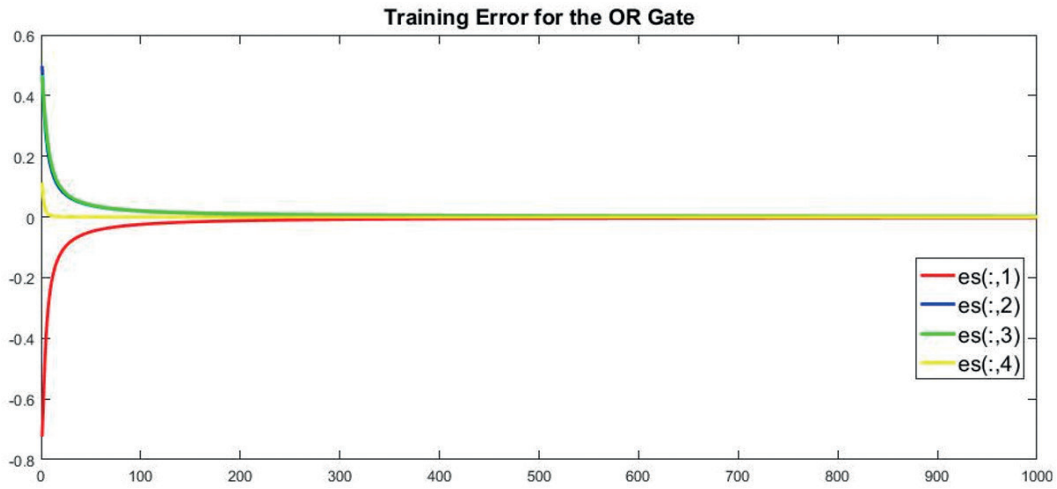
Po spustení uvedeného kódu získame nasledujúce výstupné hodnoty.

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

Tieto výstupné hodnoty sú veľmi blízke správnym výstupom v cieľovej hodnote y. Preto môžeme dospieť k záveru, že NN bola správne natrénovaná, aby dokázala určiť výstupy funkcie ALEBO.

Ak chceme vykresliť trénovaciu chybu, použijeme nasledovné príkazy:

```
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('Trénovacia chyba pre funkciu ALEBO')
```



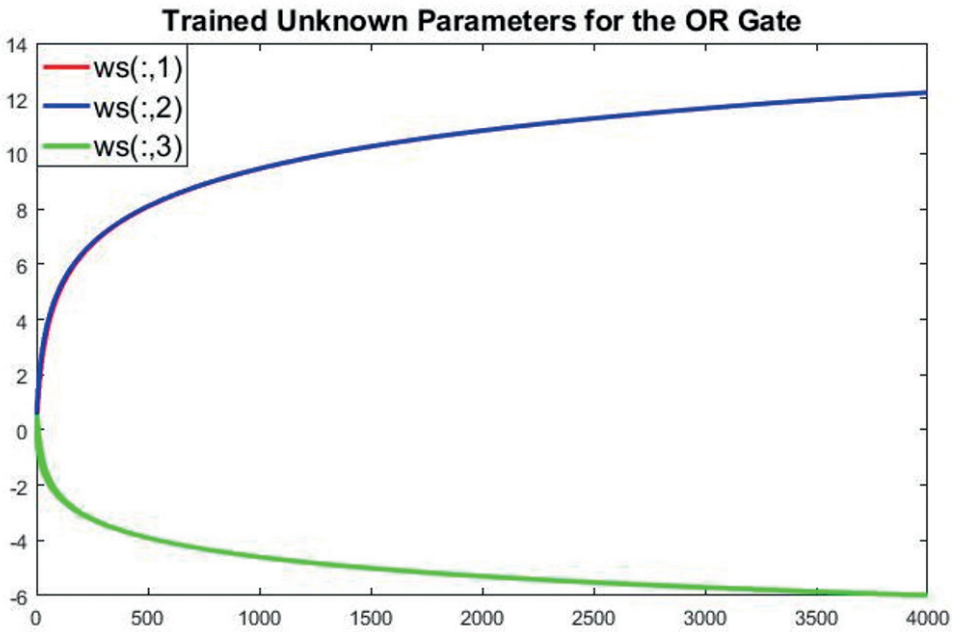
Obrázok 4. Výsledky trénovania chýb pre funkciu ALEBO

Ako je možné vidieť na Obrázku 4, chyba konverguje k nule pre príslušné dátové body funkcie ALEBO.

Nakoniec môžeme natrénované neznáme parametre vykresliť a zobraziť pomocou nasledujúcich príkazov:

```
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('Naučené neznáme parametre pre funkciu ALEBO')
```



Obrázok 5. Natrénované neznáme parametre pre funkciu ALEBO

Na Obrázku 5 je vidieť, že sú vynesené iba tri natrénované neznáme parametre. K tomu došlo v dôsledku násobenia matíc v nasledujúcom bloku kódu navzájom.

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

kde rozmer matice **size(trainPar.x)** je 3x4 a rozmer matice **size(trainPar.y)** je 4x1. Násobením týchto matíc dostaneme vektor s veľkosťou 3x1. V skutočnosti je celkom jednoduché nakresliť všetky natrénované neznáme parametre. Po prečítaní všetkých uvedených informáciách vykonajte samostatne požadovanú aktualizáciu v kóde!

10.2 OBMEDZENIA JEDNOVRSTVOVÝCH NN

Táto časť textu predstavuje kritický moment, prečo sa jednovrstvové NN museli rozvinúť na viacvrstvové NN. Pokúsime sa to ukázať na konkrétnom prípade. Uvažujme rovnakú NN, o ktorej sa hovorilo v predchádzajúcej časti. Pozostáva z troch vstupných uzlov a jedného výstupného uzla. Aktivačnou funkciou výstupného uzla je sigmoidálna funkcia. Predpokladajme, že máme štyri tréningové dátové body, ako je uvedené nižšie.

Tabuľka 2. Tréningové body funkcie XOR s označením výstupu

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

Ako je uvedené v Tabuľke 2, ide o problém funkcie XOR, ktorý má poslednú hodnotu vstupu uvedenú odchýlku rovnú 1. Od časti „Delta pravidlo" sa odlišuje v tom, že štvrtý výstup je rovný nule, zatiaľ čo vstupy zostávajú rovnaké. No rozdiel je na prvý pohľad sotva badateľný.

Keďže uvažujeme o tej istej NN, môžeme ju trénovať pomocou funkcie trainPar zo sekcie „Príklad: Delta pravidlo" s tým rozdielom, že má iné hodnoty pre výstup *y*, ako to už bolo spomenuté. Pred vykonaním kódu sa označený blok výstupných dátových bodov vo funkcii trainPar aktualizuje nasledovne:

```
% Pridelené výstupné hodnoty
trainPar.y = [0 1 1 0]';
```

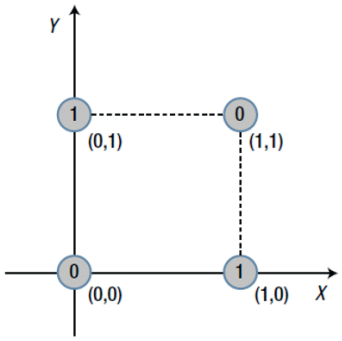
Po vykonaní takto upraveného kódu sa objavia nasledujúce hodnoty, ktoré pozostávajú z výstupu natrénovanej NN zodpovedajúcej trénovacím údajom. Môžeme ich porovnať so správnymi výstupmi danými „*y*" ako:

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

Ako je možné vidieť, dostali sme dve úplne odlišné množiny. Ak by sme NN trénovali dlhšie, dosiahli by sme podobné výsledky, t. j. dlhšie obdobie trénovania NN v tomto prípade nerozhoduje o získaní správneho výsledku.

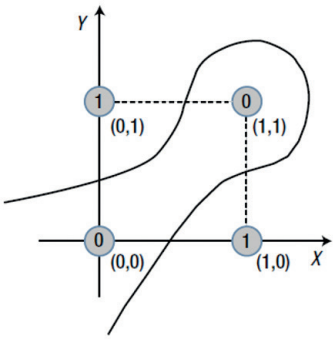
Čo sa vlastne stalo?

Ilustrácia tréningových údajov môže pomôcť objasniť tento problém. Interpretujme tri hodnoty vstupných údajov ako súradnice X, Y a Z. Keďže tretia hodnota (súradnica Z) je pevne stanovená ako 1, tréningové dáta je možné vizualizovať v rovine, ako je znázornené na nasledujúcom obrázku.



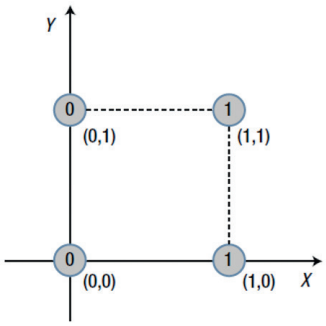
Obrázok 6. Interpretácia troch vstupných hodnôt údajov ako súradníc X, Y a Z

Hodnoty 0 a 1 v krúžkoch sú správne - očakávané výstupy priradené ku každej vstupnej trojici bodov. Jedna vec, ktorú si treba všimnúť na tomto obrázku je, že nemôžeme rozdeliť oblasti obsahujúce 0 a 1 priamkou. Môžeme ich však rozdeliť komplikovanou krivkou, ako je znázornené na Obrázku 7. Tento typ problémov sa nazýva lineárne neoddeliteľný.



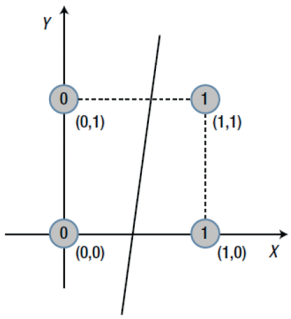
Obrázok 7. Oddelenie hodnôt 0 a 1 komplikovanou krivkou (lineárne neoddeliteľné body)

Ak sa pozrieme na údaje, ktoré boli použité v predchádzajúcej časti „Príklad: Delta pravidlo", tak sa body v rovine zobrazia, ako je to uvedené na Obrázku 8:



Obrázok 8. Tréningové dáta delta pravidla

V tomto prípade možno ľahko nájsť priamku, ktorá rozdeľuje oblasti s hodnotou 0 od oblasti s hodnotou 1. Ide o lineárne oddeliteľný problém, ako je znázornené na Obrázku 9:



Obrázok 9. Lineárne oddeliteľný problém

Zjednodušene povedané, jednovrstvová NN dokáže vyriešiť iba lineárne oddeliteľné problémy. Je to preto, že jednovrstvová NN je model, ktorý lineárne rozdeľuje priestor vstupných údajov. Aby sme prekonali toto obmedzenie jednovrstvovej NN, potrebujeme viac vrstiev v sieti. Táto potreba viedla k objaveniu sa viacvrstvových NN, ktoré môže dosiahnuť to, čo jednovrstvová NN nedokáže. Majte na pamäti, že jednovrstvová NN je použiteľná pre špecifické typy problémov. Viacvrstvová NN nemá žiadne takéto obmedzenia. Ďalšie podrobnosti nájdete v nižšie uvedených referenciách.

KAPITOLA 11

TVORBA NEURÓNOVÝCH SIETÍ V PROSTREDÍ MATLAB

Autorom tejto časti učebnice je Jarmila Škrinárová z Katedry informatiky, Fakulty prírodných vied, Univerzity Mateja Bela v Banskej Bystrici zo Slovenska.

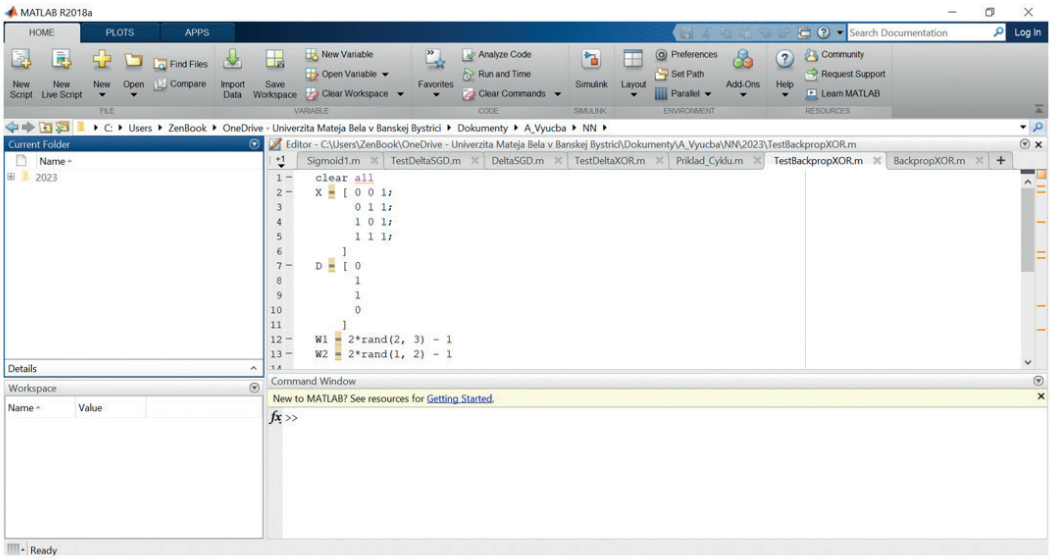
Matlab je vysokoúrovňový jazyk a interaktívne prostredie pre numerické výpočty, vizualizáciu a programovanie pre:

- analýzu údajov,
- vývoj algoritmov,
- tvorbu modelov a aplikácií.

Cieľom tejto kapitoly je naučiť sa pracovať s Matlabom a naučiť sa tvoriť jednoduché neurónové siete. Ukážeme metodický postup tvorby neurónových sietí v Matlabe. Predstavíme 3 riešené príklady tvorby neurónových sietí pomocou aplikácií a grafickým prostredím.

11.1 ZÁKLADY PRÁCE V PROSTREDÍ MATLAB - MATRIX LABORATORY

Najskôr sa zoznámime s prostredím Matlabu. V hornej časti okna sa nachádza pás s nástrojmi. Pod pásom s nástrojmi je plocha rozdelená na 4 okná, ktoré sú určené na navigáciu (pohyb po štruktúre adresárov), editovanie vykonávateľných skriptov, zobrazovanie pracovného priestoru a príkazové okno (pozri obrázok 1).



Obrázok 1. Editor, okno príkazov, pracovná plocha a navigátor

Najskôr sa naučíme pracovať v príkazovom okne, kde za znak >> píšeme príkazy (pozri obrázok 2).

Obrázok 2. Okno príkazov

Príkazy jednoduchých výpočtov, práca s premennými vektormi a maticami

Príklady 1 až 4 postupne precvičujeme priamo v príkazovom okne.

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

Vektor je jednorozmerné pole prvkov. Jednotlivé prvky vektorov píšeme do hranatých zátvoriek a oddeľujeme ich čiarkou alebo medzerou. Všimnime si, že poznámku píšeme za znak %. V ďalších častiach tejto kapitoly budeme pracovať s neurónovými sieťami a pre učenie neurónovej siete potrebujeme vstupné a výstupné údaje. Ak má sieť jeden vstup, vstupné dáta sú vo forme jednorozmerného poľa prvkov (vektora). Ak má sieť 1 výstup, výstupné údaje sú rovnako vo forme jednorozmerného poľa prvkov [3].

Príklad 5:	Príklad 6:	Príklad 7:	Príklad 8:
>> **u1=[1 2 3 4]** %riadkový vektor u1 = 1 2 3 4	>> **u2=[1 2 1 2]** % riadkový vektor u2 = 1 2 1 2	>> **u1.*u2** % skalárny súčin 2 vektorov ans = 1 4 3 8	>> **v=[-1; -7; -3]** % stĺpcový vektor v = -1 -7 -3

Príklad 9:	Príklad 10:	Príklad 11:	Príklad 12:
>> **w=[1 7 -2]'** % transponovaný vektor w = 1 7 -2	>> **6:2:12** % Prvky pravidelného vektora môžeme vygenerovať, ak poznáme prvý a posledný prvok vektora a krok. ans = 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

Príklad 13:	Príklad 14:
>> **W=2*rand(1,3)-1** W = 0.9298 -0.6848 0.9412	>> **x2=linspace(-1, 4, 8)** % interval je od -1 do 4 a 8 je počet prvkov x2 = -1.0000 -0.2857 0.4286 1.1429 1.8571 2.5714 3.2857 4.0000

Ak používame v neurónových sieťach viac ako jeden vstup, alebo výstup, potrebujeme pripraviť dáta vo forme dvojrozmerných polí. V Matlabe sú dvojrozmerné polia reprezentované maticami. Preto si precvičíme prácu s maticami.

Príklad 15:	Príklad 16:
>> **A=[1 -1 2 -3; 3 0 4 5; 3.2, 5 -6 12]** % matica 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=[]** % prázdna matica O = []

Príklad 17:	Príklad 18:
>> **B=[A; u1]** % rozšírenie matice o 1 riadok (vektor 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]** % rozšírenie matice o 1 stĺpec (vektor 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

Príklad 19:	Príklad 20:
>> **Z=zeros(2,5)** % vytvorenie nulovej matice s rozmerom 2 riadky a 5 stĺpcov. Z = 0 0 0 0 0 0 0 0 0 0	>> **O1=ones(3,4)** % vytvorenie jednotkovej matice s rozmerom 3 riadky a 4stĺpce. O1 = 1 1 1 1 1 1 1 1 1 1 1 1

Príklad 21:	Príklad 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, :)** % výpis 2. riadku matice A ans = 3 0 4 5 >> **A(:, 3)** % Výpis 3. stĺpca matice A ans = 2 4 -6	>> **I=eye(5,8)** % vytvorenie diagonálnej matice s rozmerom 5 riadkov a 8 stĺpcov 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
Príklad 23:	Príklad 24:
>> **R1=rand(3,5)** % vytvorenie náhodnej matice s rozmerom 3 riadkov a 5 stĺpcov, s hodnotami v rozsahu 0 až 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	>> **R2=randn(4)** % vytvorenie náhodnej matice s rozmerom 4x4, s hodnotami v štandardnej distribúcii 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

Príklad 25:	Príklad 26:
>> **A=[1 5 0; -1 2 3; 1 2 1]** A = 1 5 0 -1 2 3 1 2 1 >> **c=2** c = 2 % násobenie matíc >> **D=A*c** D = 2 10 0 -2 4 6 2 4 2	>> **B=[1 2 3; 1 2 3; 1 2 3]** B = 1 2 3 1 2 3 1 2 3 >> **E=B*D** % násobenie matíc E = 4 30 18 4 30 18 4 30 18

Príklad 27:	Príklad 28:
>> **A =[2 3; 0 10]** **B =[1 0; -3 5]** % násobenie matíc A = 2 3 0 10 B = 1 0 -3 5 >> **C=A*B** C = -7 15 -30 50	% skalárny súčin matíc, matice A a B sú z predchádzajúceho príkladu >> **C=A.*B** C = 2 0 0 50

Často používame údaje, ktoré obsahujú množstvo prvkov. Preto je praktické, ak tieto dáta zapíšeme do súboru, aby sme ich neskôr mohli použiť. Všetky dáta, s ktorými sme doteraz pracovali v Matlabe sa ukladajú do pracovného priestoru a vidíme ich v ľavom spodnom okne. Informácie o obsahu pracovného priestoru môžeme zobraziť pomocou príkladov 29 a 30.

Príklad 29:	Príklad 30:
>> **who** % výpis názvov a rozmerov premenných v pracovnom priestore. Your variables are: A B C O ans u v w x z	>> **whos** % pracovný priestor Name Size Bytes Class Attributes 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

Do súboru môžeme uložiť všetky premenné, vektory a matice z pracovného priestoru (pozri príklad 31), alebo len vybrané premenné, vektory a matice (pozri príklad 32).

Príklad 31:	Príklad 32:
>> **save data** % uloží všetky údaje do súboru data.dat	>> **save data1 u1 u2 v** % uloží premenné u1, u2 do súboru data1.mat

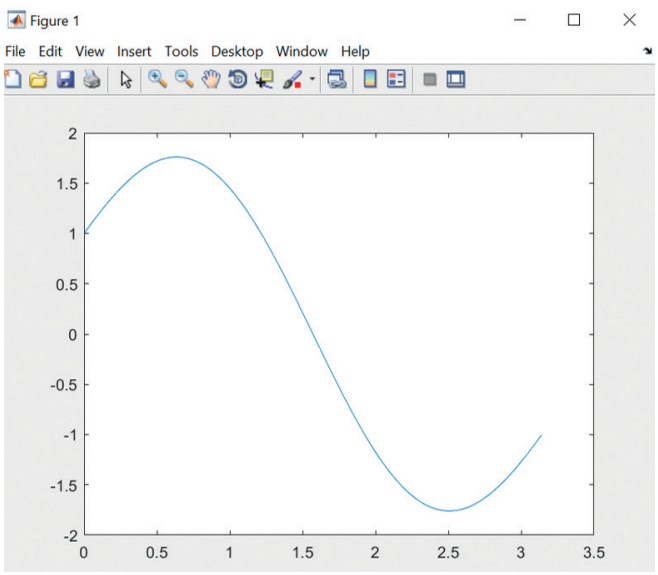
Údaje, uložené v súbore môžeme kedykoľvek načítať do pracovného priestoru Matlabu a znova s nimi pracovať. Pozri príklady 33 a 34.

Príklad 33:	Príklad 34:
>> load data1 % načíta všetky premenné zo súboru data1.mat	**>> load data.dat -MAT** % načíta všetky premenné zo súboru data.dat

Pri vykresľovaní priebehu určitej funkcie je potrebné, aby bol počet prvkov na osi x rovnaký ako na osi y. Nakreslenú funkciu vidíme na obrázku v príklade 35. Funkcia sa vykreslí v novom okne, ktoré je možné editovať – vkladať názvy osí, nadpis, atď.

Príklad 35:

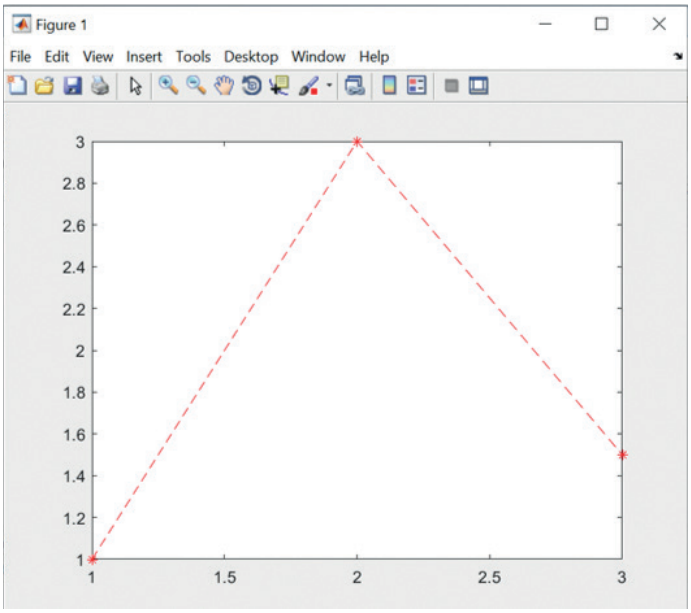
```
>> x=linspace(0, pi);
%plot the function
>> y=sin(2*x)+cos(x) ;
>> plot(x,y)
```



V príkaze plot je možné zmeniť farbu aj typ čiary. Pozri príklad 36.

Príklad 36:

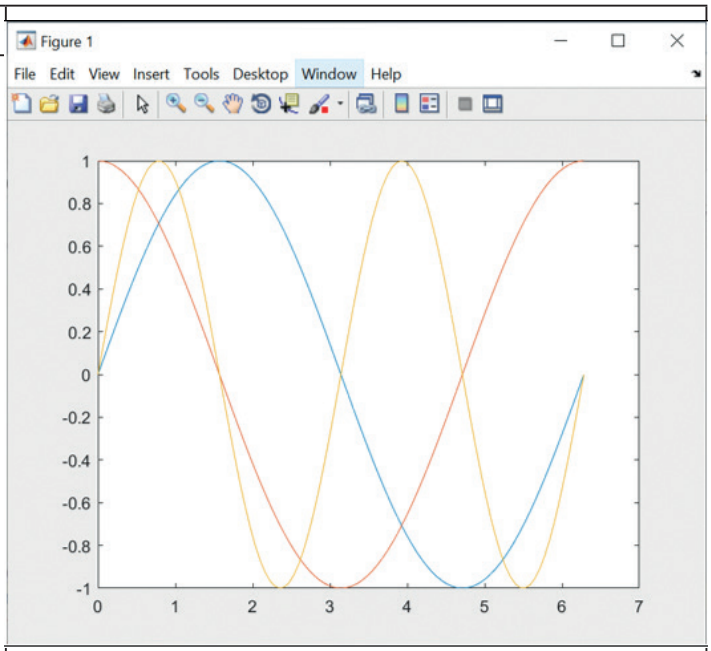
```
>> u=[1 2 3] ;
>> v=[1 3 1.5] ;
>> plot(u,v, 'r--*')
```



Rovnako je možné vykresliť viac funkcií do jedného obrázka. Pozri príklad 37.

Príklad 37:

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



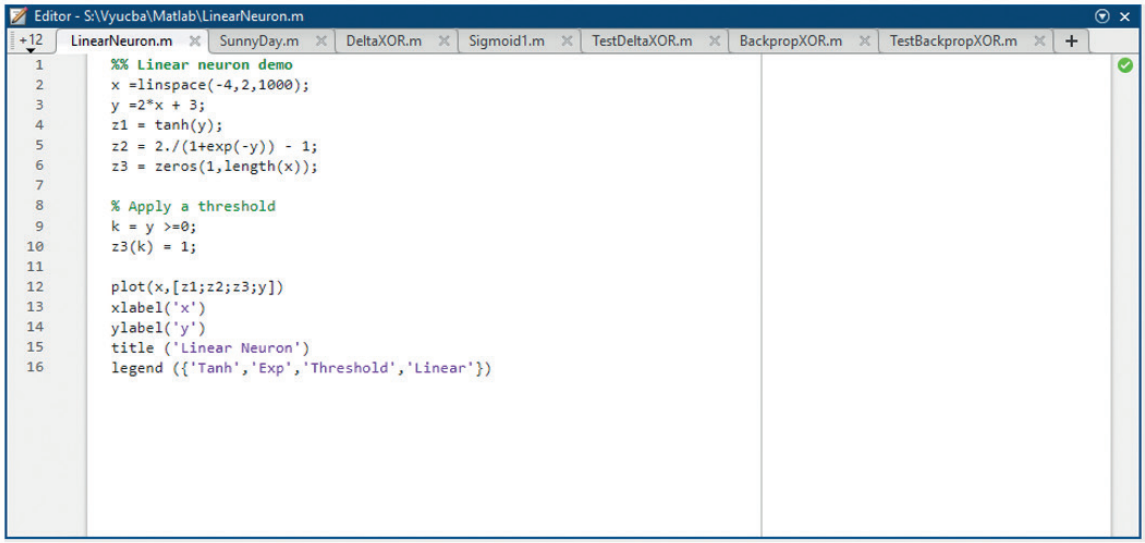
Pre ilustráciu ukážeme jednoduchý príklad programu (príklad 38), kde sa v cykle postupne vypisujú jednotlivé riadky matice X. Výstup programu sa nachádza v pravom stĺpci.

Príklad 38:

```
>> clear all
X = [ 0 0 1;
      0 1 1;
      1 0 1;
      1 1 1;
    ] ;
  N = 4;          % vypíš vždy 1 riadok matice 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
```

V predchádzajúcich častiach sme ukázali ako píšeme príkazy do príkazového riadku. To nie je praktické ak potrebujeme napísať veľa príkazov. Takéto za sebou idúce príkazy napíšeme v textovom editore a uložíme do súboru s príponou .m. Takto vytvoríme skript, ktorý môžeme otvoriť a spustiť v prostredí Matlabu. Príkazy, ktoré zapisujeme do skriptu je vhodné najskôr vyskúšať v príkazovom riadku Matlabu, aby neobsahovali chyby.



Obrázok 3. Príklad skriptu (M - file) in Matlab editor window.

11.2 METODIKA A PRÍKLADY TVORBY NEURÓNOVÝCH SIETÍ V PROSTREDÍ MATLAB

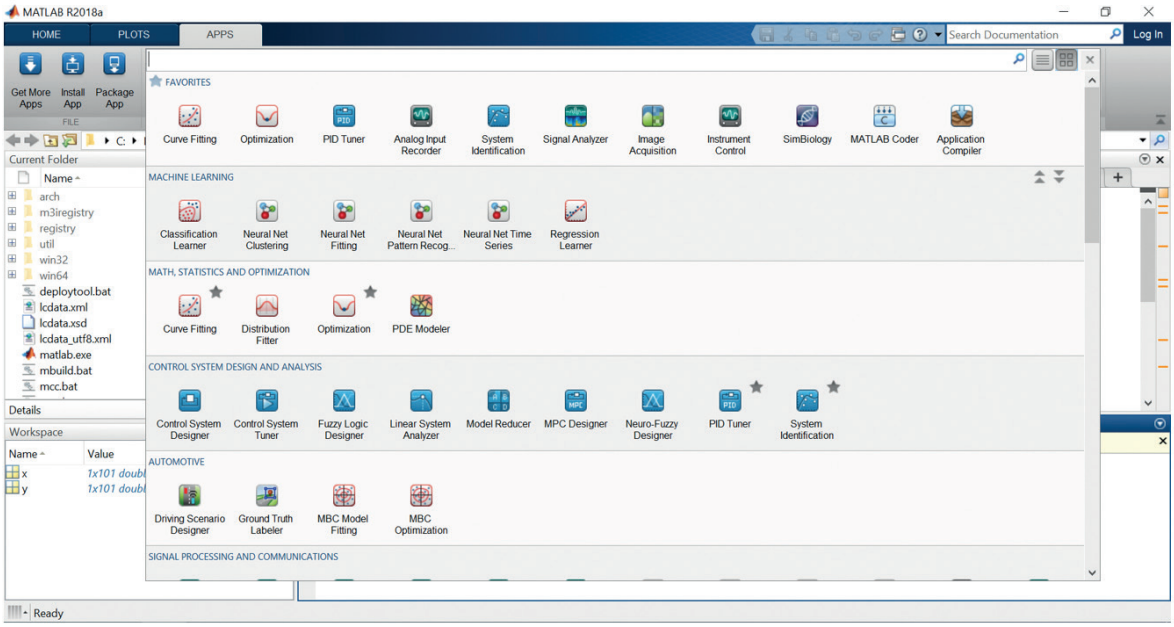
V reálnom svete sa často stáva, že dokážeme urobiť rôzne merania, ale nedokážeme opísať jednoduchým matematickým modelom správanie sa určitého systému. To znamená, že máme namerané hodnoty vstupov do systému a im zodpovedajúce výstupy, ale na základe vstupov nevieme vypočítať výstupy. Na tento účel používame neurónové siete, ktoré sa dokážu naučiť vzťah medzi vstupmi (z určitého intervalu hodnôt) a výstupmi, alebo klasifikovať vstupy do určitých skupín. Dobre naučená neurónová sieť dokáže pre rôzne vstupné hodnoty (z rovnakého intervalu) vyprodukovať správne výstupy.

Cieľom tejto podkapitoly je čitateľne špecifikovať metodiku tvorby neurónových sietí v prostredí Matlab. Následne vyriešime jednoduché príklady, ktoré nám pomôžu porozumieť postupu tvorby neurónových sietí.

Metodika tvorby neurónových sietí v grafickom prostredí Matlab

Metodiku opíšeme v jednotlivých krokoch:

1. krok: Príprava údajov. Zvyčajne potrebujeme 2 sady údajov (sadu vstupov a sadu im zodpovedajúcich očakávaných výstupov). Ak máme jednu vzorku údajov, kde 1 vstupu, zodpovedá 1 očakávaný výstup (target), potom oba datasety majú rovnaký rozmer, t.j. 1 riadok x počet stĺpcov (vzoriek).
2. krok: V prostredí Matlab si zo záložky APPS zvolíme vhodnú aplikáciu, napríklad z kategórie Machine Learning, si zvolíme aplikáciu Neutral Net Fitting (pozri obrázok 4).



Obrázok 4. Aplikácie, ktoré sú súčasťou prostredia Matlab.

3. krok: Načítame vstupné dáta z datasetu a načítame očakávané výstupné dáta z datasetu (tie, kroré sme si pripravili v kroku 1).
4. krok: Zadáme pomer, v akom majú byť dáta rozdelené do 3 množín na trénovanie, validáciu a testovanie, napríklad 70 %, 15 % a 15 %.
5. krok: Navrhneme architektúru siete. Počet vstupov a výstupov siete sa automaticky nastaví podľa vstupných a očakávaných výstupných údajov. Potrebné je nastaviť počet neurónov v skrytých vrstvách. Napríklad, ak navrhujeme viacvrstvovú perceptrónovú sieť, ktorá má 2 skryté vrstvy, potrebujeme zadať počet neurónov pre prvú a pre druhú skrytú vrstvu.
6. krok: Výber učiaceho algoritmu. Zvolíme jeden z pripravených učiacich algoritmov napríklad Levenberg – Marquardt, Bayesian Regularization alebo škálovaný konjugovaný gradient.
7. krok: Spustíme proces učenia sa siete. Je treba zobrať na vedomie, že v aplikácii sú prednastavené určité hodnoty. Napríklad počet epoch učenia môže byť nastavený na 1000 a presnosť učenia je vyjadrená pomocou strednej kvadratickej chyby a regresie. Stredná kvadratická chyba (angl . Mean Square Error, MSE) je priemerný rozdiel štvorcov (druhých mocnín) medzi výstupmi siete po a očakávanými výstupmi siete pred tréningovým procesom. Naším cieľom je, po učení siete, získať čo najmenšie hodnoty chýb. Nulová hodnota znamená žiadnu chybu. Regresia (R) vyjadruje mieru korelácie medzi výstupmi a očakávanými výstupmi siete. Hodnota R 1 znamená blízku koreláciu a 0 znamená žiadnu koreláciu alebo inými slovami, existuje náhodný vzťah.
8. Krok: Ak je pre nás dosiahnutá presnosť učenia dostatočná, proces učenia sa siete končí. V opačnom prípade treba zmeniť architektúru siete (počty skrytých vrstiev a počty neurónov v nich), resp. zmeniť učiaci algoritmus, alebo ak je to možné zmeniť počet epoch učenia sa siete. To znamená, že opakujeme postup od kroku 5. Treba brať do úvahy, že vysoký počet epoch učenia môže viesť k tzv. preučeniu siete.

Príklad tvorby jednoduchých neurónových sietí – aplikácia Neural Net Fitting Function

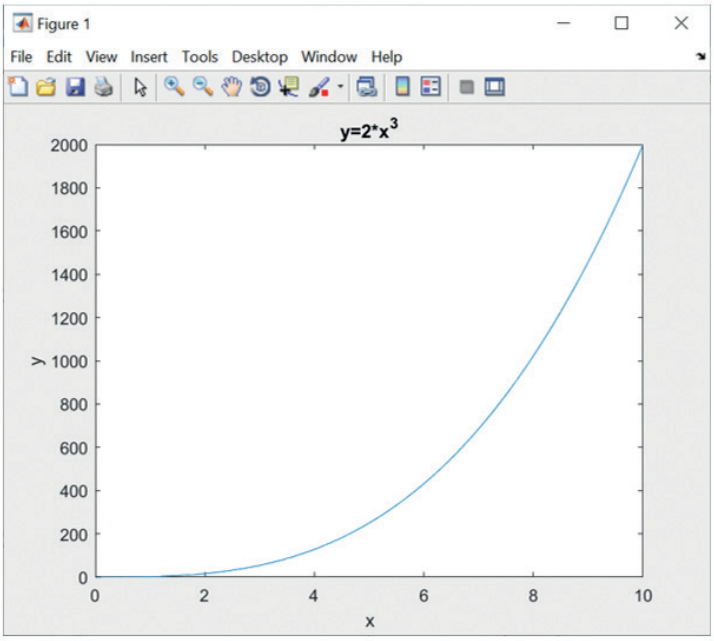
V tomto príklade ukážeme, ako sa neurónová sieť naučí hodnoty funkcie. Postupujeme podľa metodiky, ktorá je uvedená v úvode časti 11.2, v tejto kapitole.

Krok 1: Pre jednoduchosť nebudeme používať namerané údaje, ale vytvoríme si vstupné a výstupné data v Matlabe. Pomocou príkazov v Matlabe vytvoríme 2 sady údajov (pozri Výpis 1). Prvý dataset obsahuje vstupné s názvom data1.mat obsahuje hodnoty vstupov a druhý dataset s názvom data2.mat obsahuje hodnoty targets t.j. očakávané výstupy po naučení siete. Prvky vstupov a targets sú usporiadané v takom poradí, že jednotlivým vstupným prvkom, zodpovedajú príslušné prvky očakávaných výstupov [2].

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

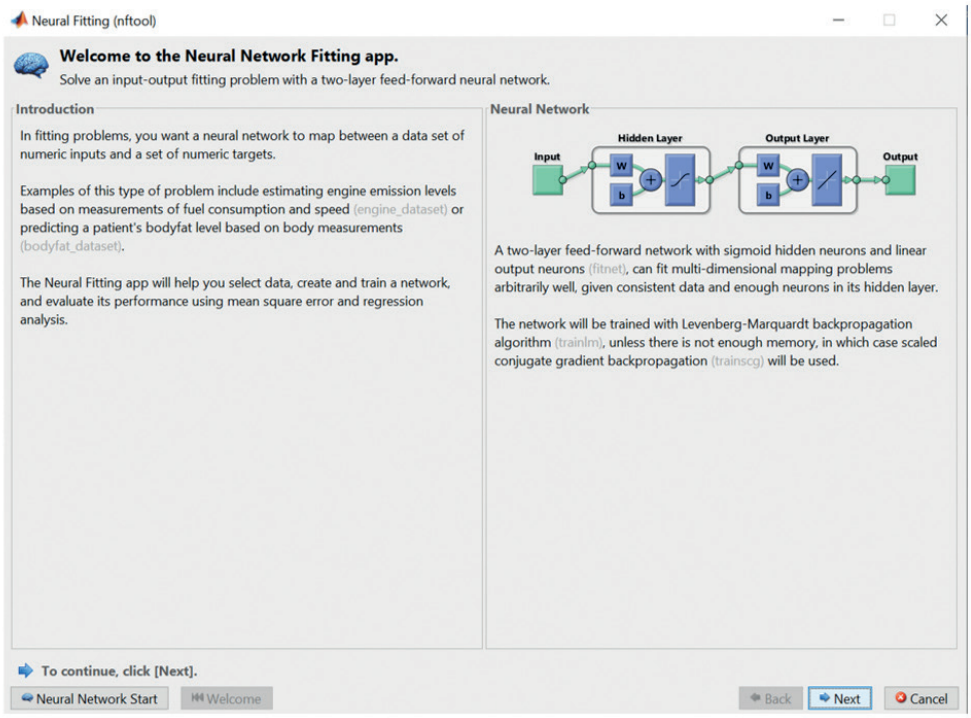
Výpis 1: Tvorba údajov vstupov a očakávaných výstupov

Po spustení príkazov z Výpisu 1 sa nám vypíšu hodnoty vektorov x a y, vykreslí sa graf funkcie (pozri obrázok 5) a do súborov sa uložia datasety.



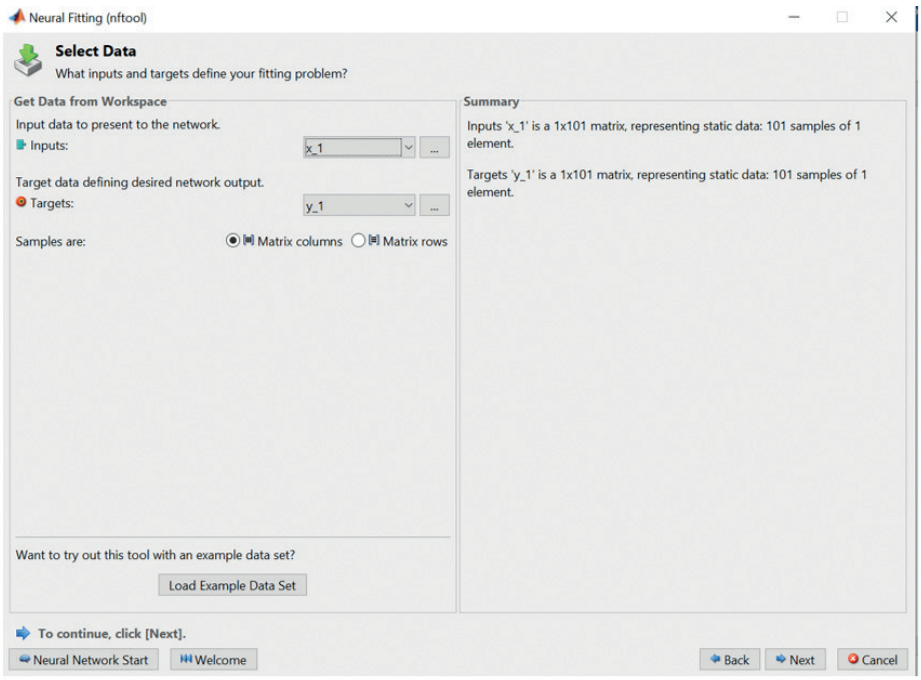
Obrázok 5. Priebeh funkcie $y = 2x^3$

Krok 2: V prostredí Matlab si zo záložky APPS zvolíme vhodnú aplikáciu, z kategórie Machine Learning, si zvolíme aplikáciu Neutral Net Fitting. Spustíme aplikáciu Neutral Net Fitting (pozri obrázok 6). V aplikácii sa pohybujeme pomocou tlačidla next.



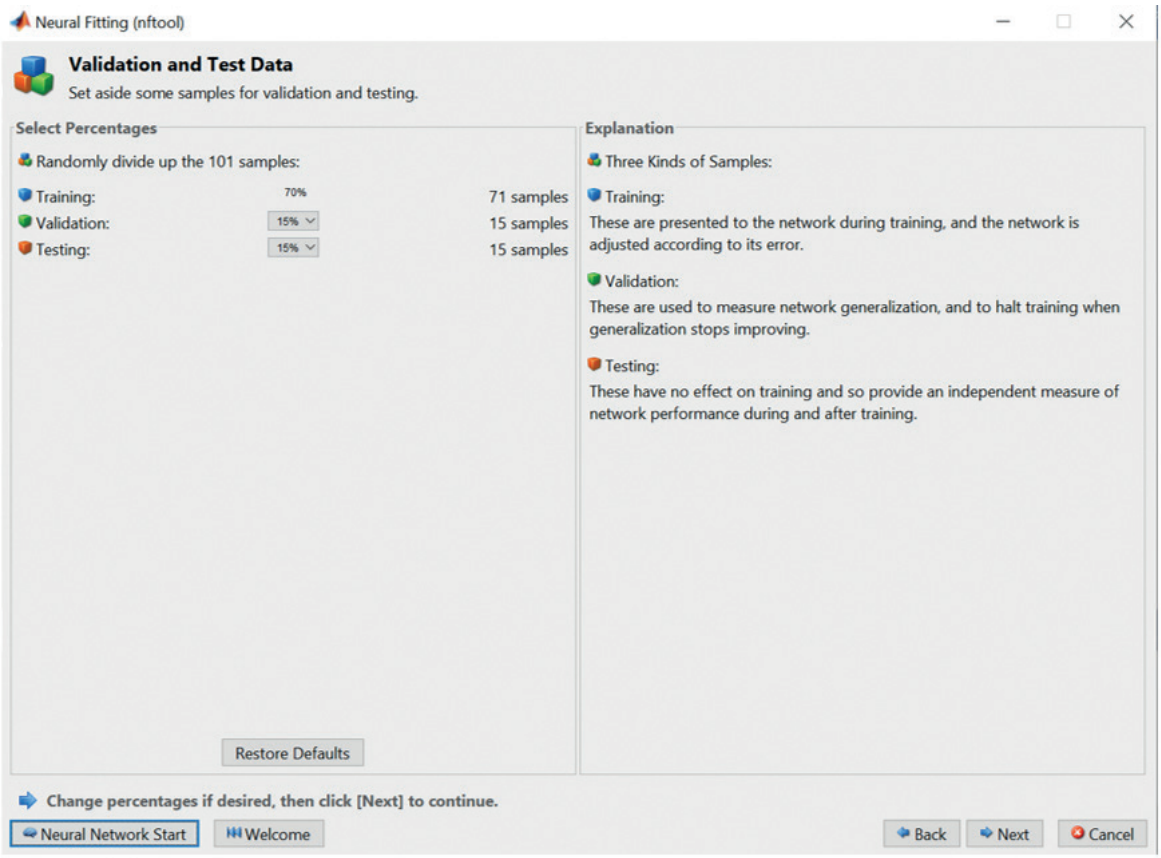
Obrázok 6. Vzhľad aplikácie Neural Net Fitting v Matlabe.

Krok 3: Načítame datasety vstupných a očakávaných výstupných údajov z pripravených súborov (pozri obrázok 7, vľavo). Keďže máme rovnaký počet vstupov aj očakávaných výstupov, dbáme na to, aby mali súbory rovnaký rozmer (pozri obrázok 7, vpravo).



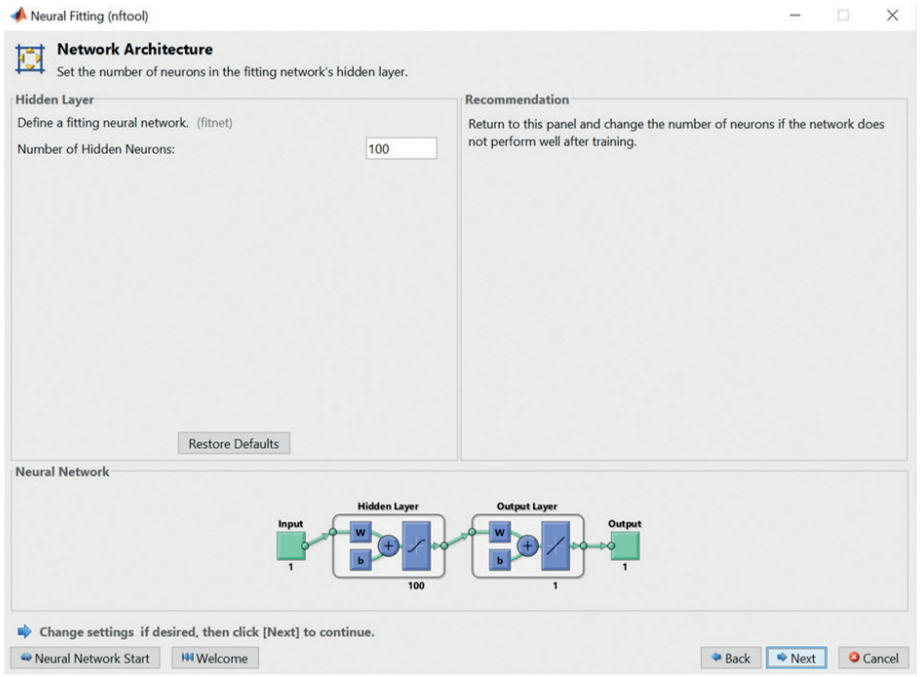
Obrázok 7. Spôsob načítania vstupných a očakávaných výstupných hodnôt.

Krok 4: Zadáme pomer, v akom majú byť dáta rozdelené do 3 množín na trénovanie, validáciu a testovanie. V našom prípade to je 70 % údajov na trénovanie siete, 15 % na validáciu v rámci procesu učenia sa siete a 15 % na testovanie (pozri obrázok 8).



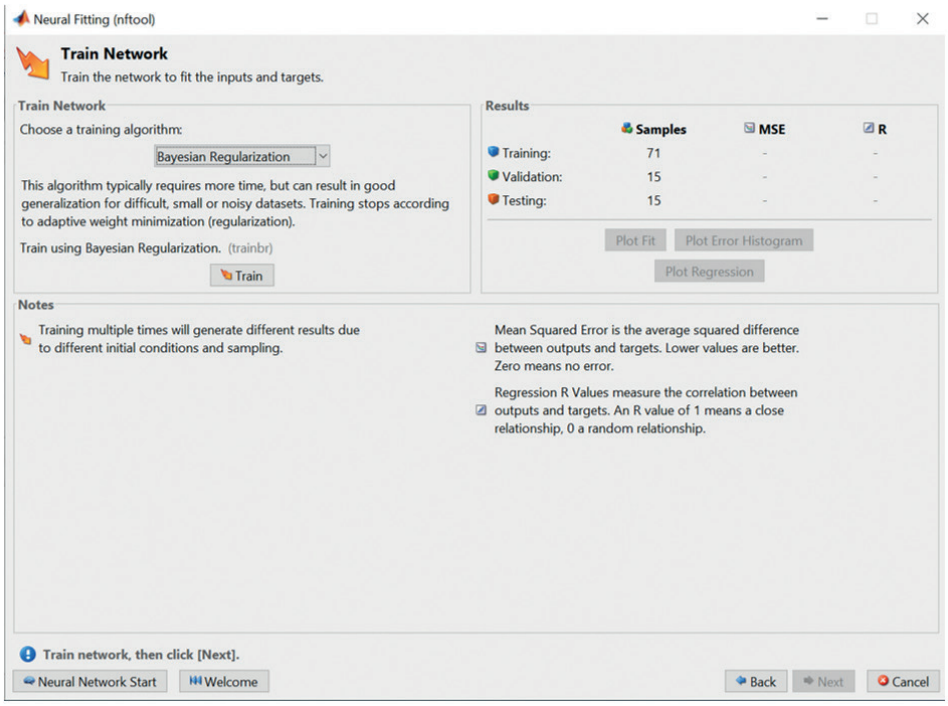
Obrázok 8. Spôsob rozdelenie vzoriek údajov v riešenej úlohe.

Krok 5: Navrhneme architektúru siete. Počet vstupov a výstupov siete sa automaticky nastavil podľa vstupných a výstupných údajov tak, že naša sieť má 1 vstup a 1 očakávaný výstup (pozri obrázok 9). Výstupná vrstva má len 1 neurón, pretože máme 1 výstup zo siete. Počet neurónov vo výstupnej vrstve sa tiež nastavuje automaticky. V našom prípade máme len 1 skrytú vrstvu, lebo používame aplikáciu Neutral Net Fitting. Počet neurónov v skrytej vrstve nastavíme na hodnotu 100. V prípade, že sa sieť nenaučí dostatočne presne vzťah medzi vstupmi a očakávanými výstupmi, môžeme sa k nastavovaniu architektúry siete vrátiť a zmeniť počet skrytých neurónov.



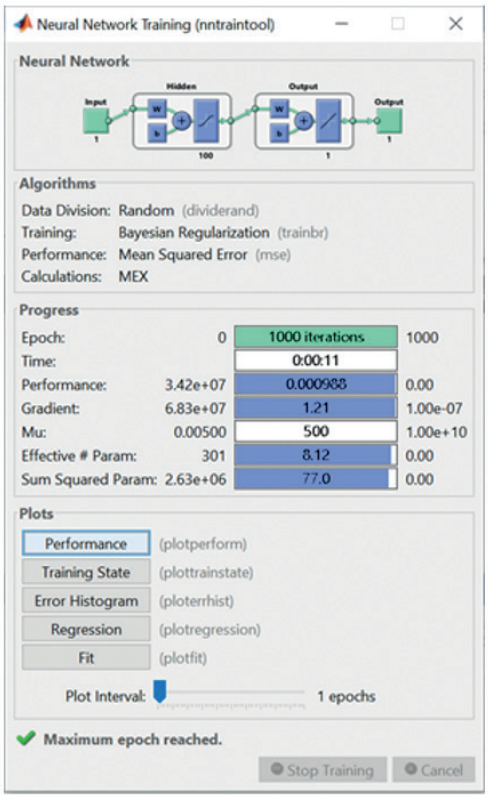
Obrázok 6. Návrh architektúry siete pre našu úlohu.

Krok 6: Výber učiaceho algoritmu. Môžeme si zvoliť jeden z troch učiacich algoritmov: Levenberg – Marquardt, Bayesian Regularization alebo Scaled conjugate gradient. Zvolíme si algoritmus Bayesian Regularization (pozri obrázok 10).



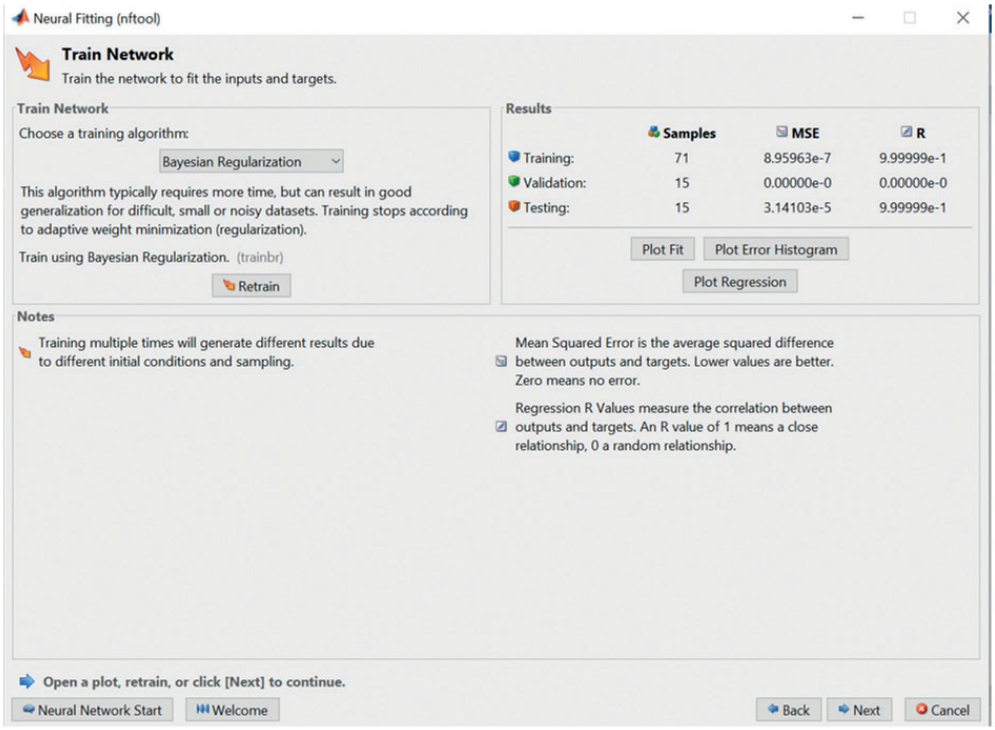
Obrázok 10. Výber učiaceho algoritmu.

Krok 7: Proces učenia sa siete spustíme stačením tlačidla Train. Počet epoch učenia je nastavený na 1000 a proces učenia sa môžeme sledovať tak, ako to vidíme na obrázku 11.



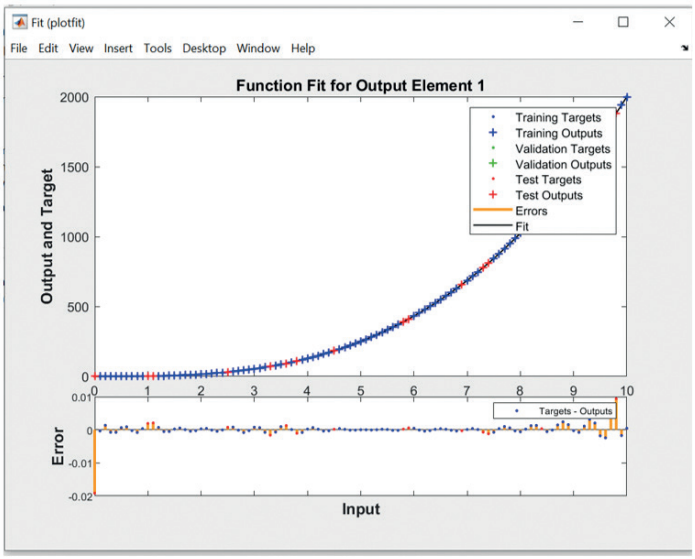
Obrázok 11. Priebeh učenia sa siete.

Presnosť učenia je vyjadrená pomocou MSE a R (pozri obrázok 12). Stredná kvadratická chyba (angl . Mean Square Error, MSE) je priemerný rozdiel štvorcov (druhých mocnín) medzi výstupmi siete po a očakávanými výstupmi siete pred tréningovým procesom. Naším cieľom je, po učení siete, získať čo najmenšie hodnoty chýb. Nulová hodnota znamená žiadnu chybu. Vidíme, že sieť sa naučila našu funkciu s chybou MSE $8{,}95*10^{-7}$, čo predstavuje bezvýznamnú chybu. Testovanie siete potvrdilo, že sa sieť naučila správne s nízkou chybou MSE v hodnote $3{,}14*10^{-5}$. Regresia (R)vyjadruje mieru korelácie medzi výstupmi a očakávanými výstupmi. Hodnota R 1 znamená blízku koreláciu a 0 znamená žiadnu koreláciu alebo inými slovami, existuje náhodný vzťah. Výpočet korelácií po trénovaní siete a rovnako aj po testovaní siete dosiahol hodnotu 1, čím sa potvrdilo, že sa sieť veľmi dobre naučila vzťah medzi vstupmi a výstupmi siete.



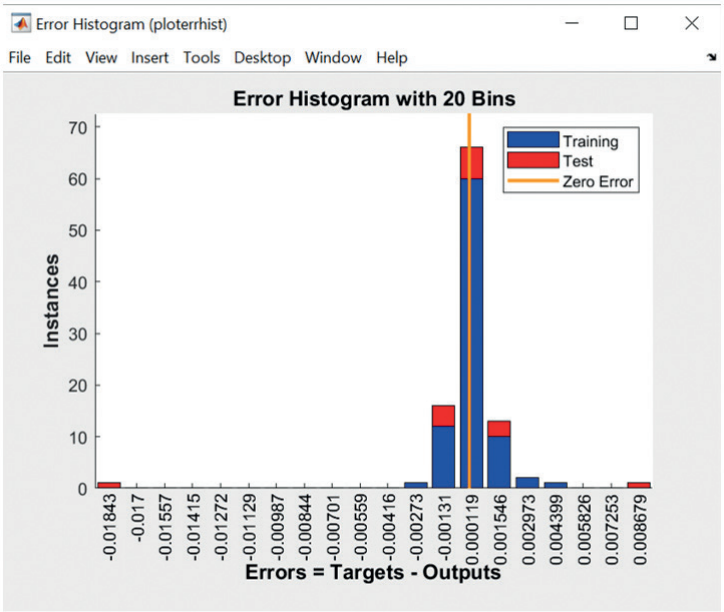
Obrázok 12. Chyby a korelácie po procese učenia a testovania siete.

Krok 8: Môžeme konštatovať, že dosiahnutá presnosť učenia a testovania siete je dostatočná a proces učenia sa siete končí. Preto sa môžeme detailnejšie pozrieť hodnoty a priebeh naučenej funkcie ak postupne stlačíme tlačidlá Plot Fit, Plot Error Histogram a Plot Regression (pozri obrázok 12). Po stlačení Plot Fit vidíme priebeh hodnôt očakávaných výstupov a výstupov siete v závislosti od vstupov, po procese učenia a testovania siete (pozri obrázok 13).



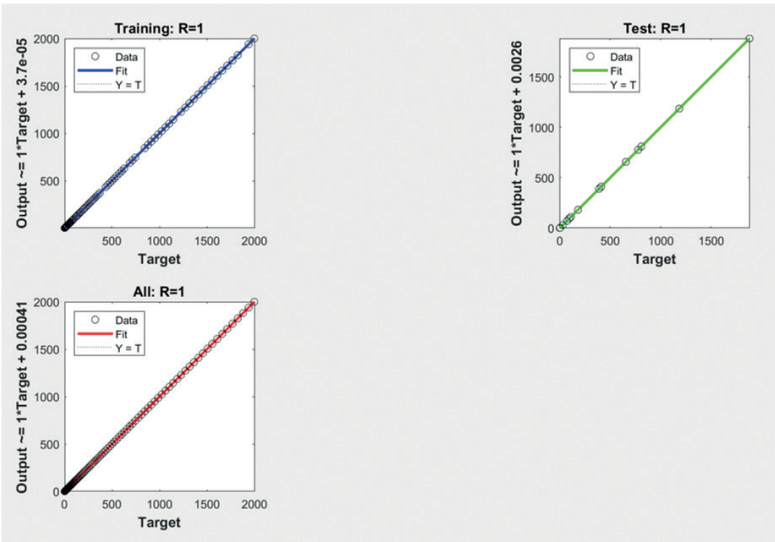
Obrázok 13. Priebeh hodnôt očakávaných výstupov a výstupov siete v závislosti od vstupov, po procese učenia a testovania siete.

Po stlačení Plot Error Histogram vidíme hodnoty chýb a ich početnosť (pozri obrázok 14). Pričom v tomto prípade je absolútna chyba vyjadrená ako rozdiel medzi hodnotou očakávaným výstupom a výstupom siete vo vzťahu k určitému vstupu siete. Vidíme, že najčastejšie sa vyskytovala chyba 0.000119.



Obrázok 14. Hodnoty absolútnych chýb a ich početnosť.

Po stlačení Plot Regression vidíme hodnoty korelácie medzi hodnotami očakávaných výstupov a výstupnými hodnotami v procese trénovania, procese testovania a z oboch procesov (pozri obrázok 15). Výpočet korelácií po trénovaní siete a rovnako aj po testovaní siete dosiahol hodnotu 1, čím sa potvrdilo, že sa sieť veľmi dobre naučila vzťah medzi vstupmi a výstupmi siete.



Obrázok 15. Korelácie medzi hodnotami očakávaných výstupov a výstupnými hodnotami v procese trénovania, procese testovania a z oboch procesov.

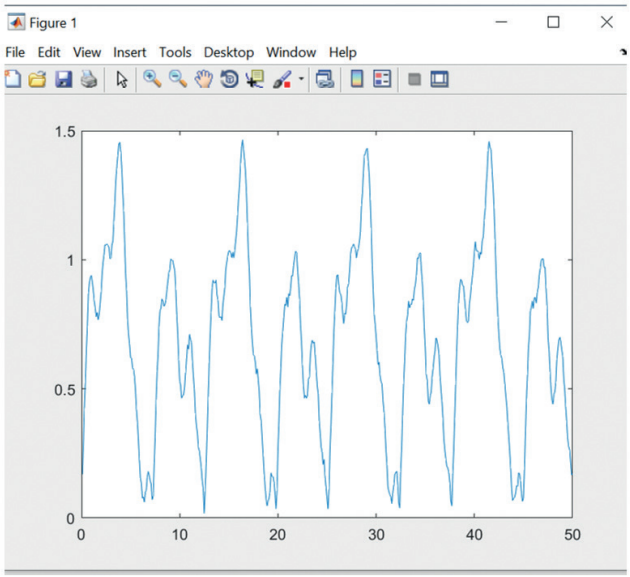
Príklad tvorby neurónovej siete s cieľom naučiť sa namerané hodnoty

Cieľom tohto príkladu je naučiť neurónovú sieť hodnoty, ktoré sme získali meraním. Postupujeme podľa metodiky, ktorá je uvedená v časti 11.2, v tejto kapitole.

Krok 1. Príprava údajov. Máme 500 nameraných hodnôt údajov, ktoré sú uložené v súbore data4 (pozri obrázok 16). Kvôli prehľadnosti x – ovú os vykresľujeme od hodnoty 0.1 po hodnotu 50 s krokom 0.1. Pozri Listing 2.

```
x=0.1:0.1:50
save data3 x
load data4 y
plot(x,y)
```

Listing 2. Sada vstupov a nameraných výstupov

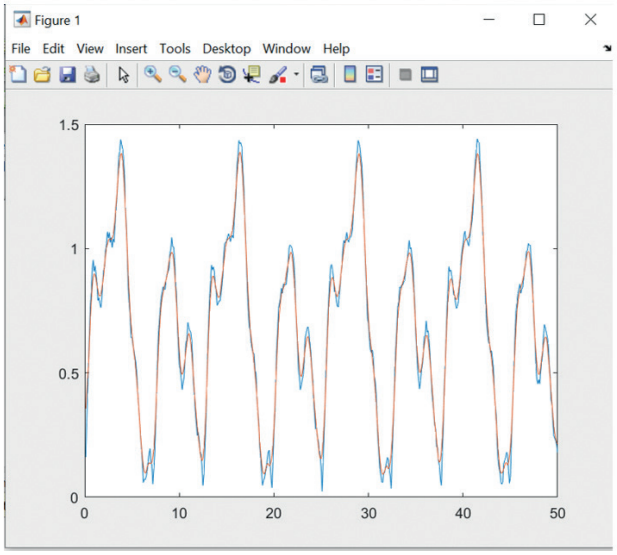


Obrázok 16. Namerané hodnoty

Niekedy sú namerané údaje zašumené a preto je potrebné ich upraviť [1]. Môžeme použiť plávajúci priemer, ktorý vyhladí údaje tak, aby sa ich mohla naučiť neurónová sieť. Plávajúci priemer postupne prechádza všetky vyhladzované hodnoty a aktuálnu hodnotu nahradí priemernou hodnotou tzv. okna. Okno tvorí aktuálna hodnota a určitý počet hodnôt pred a po aktuálnej hodnote. Teraz ukážeme spôsob vyhladenia našich nameraných údajov, ktoré sú uložené v súbore data4.mat. Náš listing 2 rozšírime o príkazy tak, že namerané údaje vyhladíme pomocou plávajúceho priemeru s oknom, ktoré má šírku 9 hodnôt. Upravené údaje, ktoré sú vo vektore m uložíme do súboru data5.mat a pri učení siete ich použijeme ako očakávané výstupy (pozri Výpis 3.) a obrázok 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
```

Výpis 3. Sada vstupov a sada očakávaných výstupov s vyhladenými nameranými hodnotami



Obrázok 17. Namerané údaje sú modrej farby a vyhladené údaje sú červené

Krok 2: V prostredí Matlab zvolíme aplikáciu Neutral Net Fitting.

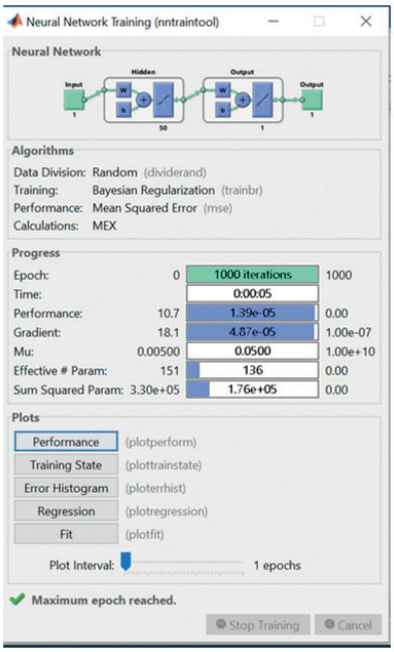
Krok 3. Načítame dáta do aplikácie. V súbore data3.mat sú vstupné data a v súbore data5.mat sú očakávané výstupy.

Krok 4. Pomer necháme rovnaký, ako v predchádzajúcom príklade.

Krok 5: Navrhneme architektúru siete. Zvolíme 50 neurónov v skrytej vrstve.

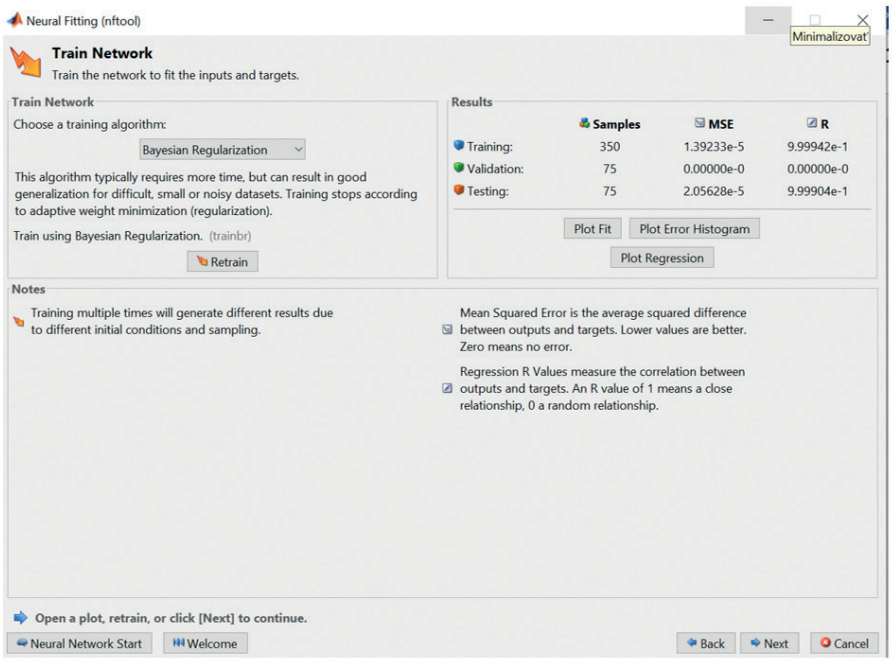
Krok 6. Zvolíme algoritmus učenia Bayesian Regularization.

Krok 7. Spustíme proces učenia (pozri obrázok 18).



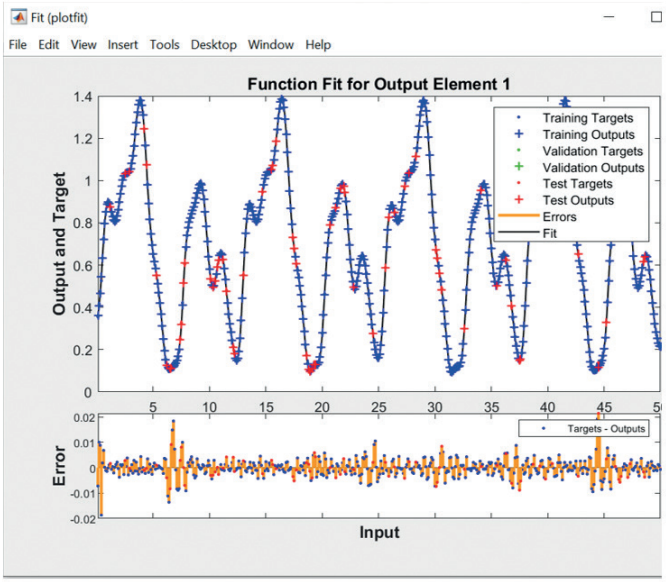
Obrázok 18. Priebeh učenia sa siete.

Krok 8: Pozrieme sa detailnejšie pozrieť na výsledky učenia sa siete (obrázok 19). Na základe chyby učenia s hodnotou $1.39x10^{-5}$ a chyby testovania siete s hodnotou $1.39x10^{-5}$ konštatujeme, že sa sieť naučila správne hodnoty.



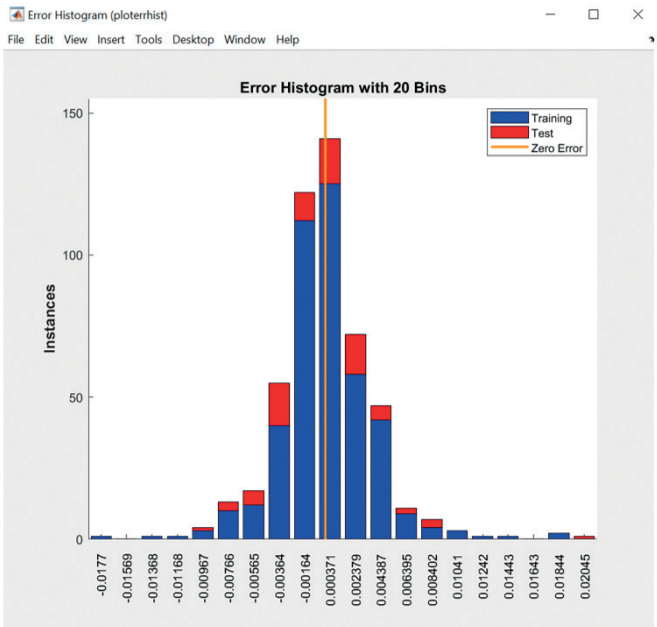
Obrázok 19. Zobrazenie chýb učenia.

Na obrázku 20 vidíme priebeh hodnôt očakávaných výstupov a výstupov siete v závislosti od vstupov, po procese učenia sa a testovania siete. Naučené hodnoty sa prekrývajú s očakávanými výstupnými hodnotami. V dolnej časti obrázka vidíme zobrazené chyby, ktoré sú minimálne.



Obrázok 20. Výsledky učenia siete - priebeh očakávaných výstupov a naučených hodnôt.

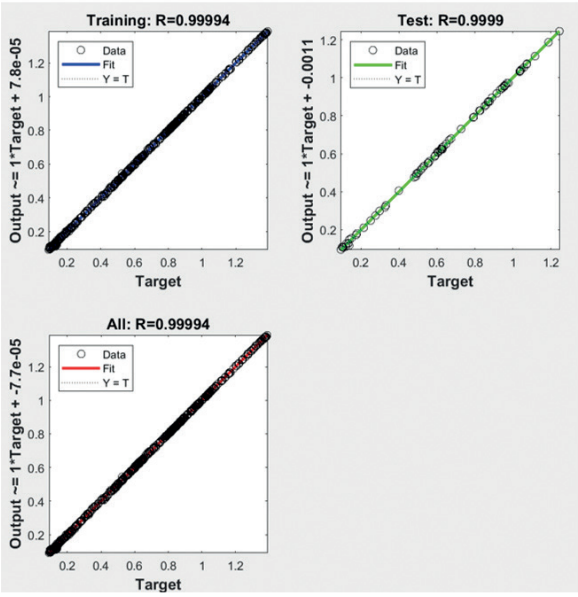
Na histograme, na obrázku 21, môžeme vidieť veľkosť a početnosť chýb, čo znova dokazuje, že chyby sú minimálne.



Obrázok 21. Výsledky učenia siete – histogram – veľkosť a početnosť chýb.

Zobrazené regresie (pozri obrázok 22), ktoré dosiahli hodnotu 1, z pohľadu trénovaných, testovaných a všetkých vstupov rovnako ukazujú, že máme veľmi dobre naučenú sieť.

Celkovo môžeme konštatovať, že dosiahnutá presnosť učenia a testovania siete je dostatočná a proces učenia sa siete končí.



Obrázok 22. Výsledky učenia siete – výpočet regresií.

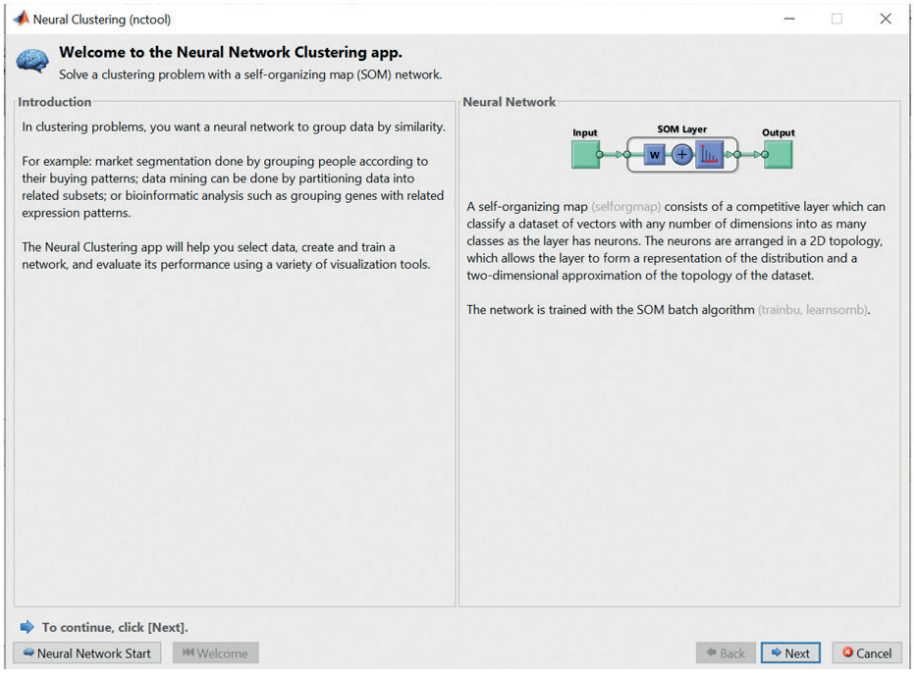
Príklad klasifikácie pomocou neurónovej siete samoorganizujúcej sa mapy

V tomto príklade riešime známu úlohu klasifikácie kvetov Iris. Použijeme IRIS dataset a opíšeme riešený príklad, ktorý je súčasťou prostredia Matlab. Kvety iris môžeme opísať pomocou 4 parametrov, pričom hodnoty (sepal length, sepat width, petal length and petal width) v datasete sú uvedené v centimetroch. Preto každý kvet je charakterizovaný 4 prvkami. Našou úlohou je vytvoriť takú neurónovú sieť samoorganizujúcej sa mapy (angl. self-organizing map neural network, SOM), ktorá klasifikuje typy kvetov iris do tried tak, že podobné typy sa nachádzajú v skupine blízko seba. Mapa sa vytvorí na základe podobnosti vzoriek a naučená neurónová sieť dokáže klasifikovať aj neznáme vzorky [4].

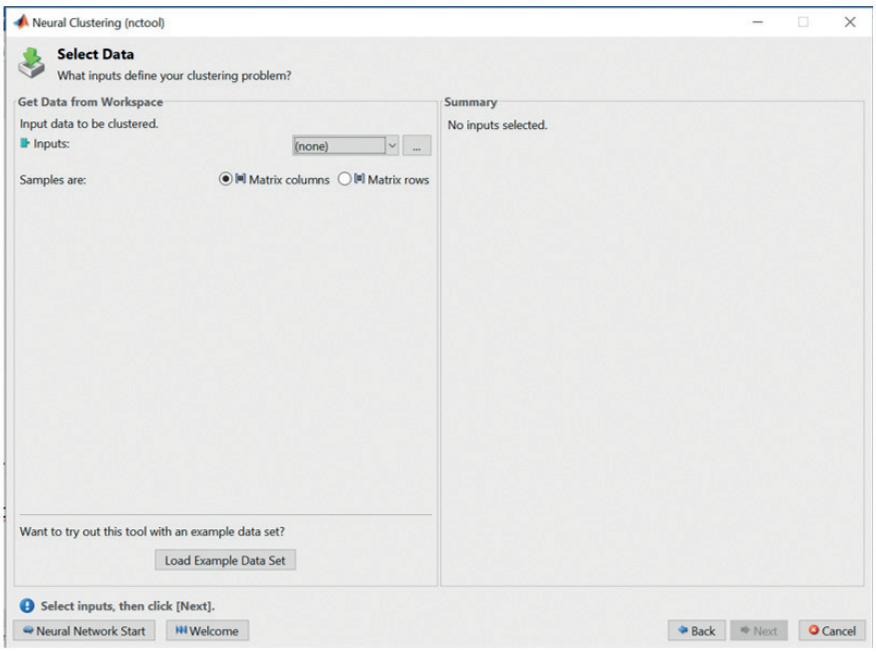
Postupujeme podľa metodiky z podkapitoly 2.1.

Krok1: Krok vynecháme. Použijeme pripravený dataset, ktorý je dostupný v Matlabe a tento dataset podrobne opíšeme v kroku 3.

Krok 2: V prostredí Matlab si zo záložky APPS zvolíme vhodnú aplikáciu, z kategórie Machine Learning, si zvolíme aplikáciu Neutral Net Clustering a spustíme aplikáciu. Táto aplikácia nám pomôže vytvoriť neurónovú sieť samoorganizujúcej sa mapy. Pozri obrázok 23.



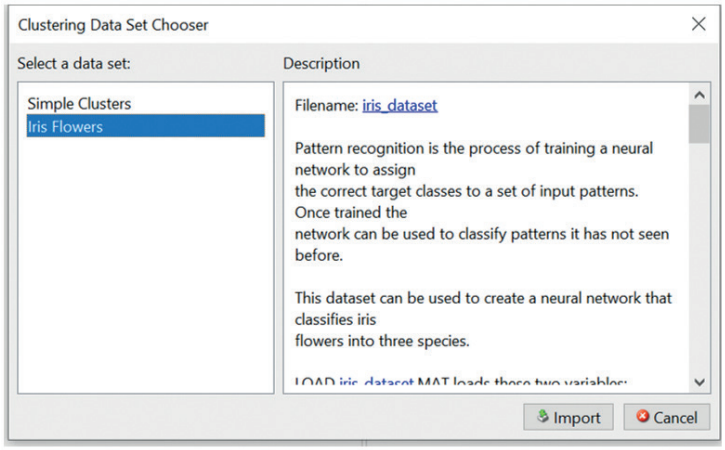
Obrázok 23. Aplikácia Neural Network Clustering.



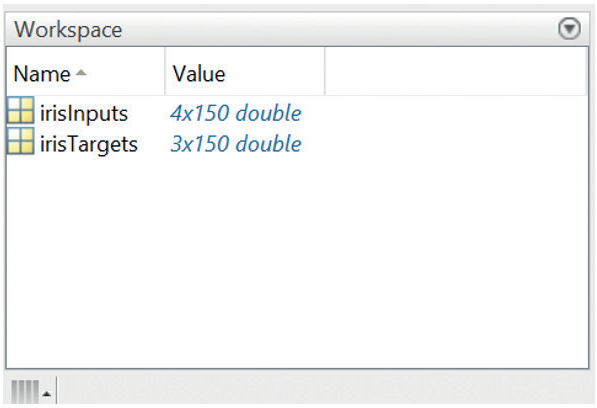
Obrázok 24. Načítanie pripraveného datasetu

Krok 3: Načítame datasety. Pozri obrázok 24.

Pretože chceme použiť pripravený IRIS dataset, stlačíme Load Example Data Set, zvolíme Iris Flowers a importujeme dáta. Pozri obrázok 25.



Obrázok 25. Importovanie datasetu Iris flowers.



Obrázok 26. Pracovný priestor po načítaní datasetu Iris flowers.

Po načítaní datasetu sa v workspace okne Matlabu zobrazia matice irisInputs a irisTargets (očakávané výstupy), pozri obrázok 26. Vidíme, že matica irisInputs má rozmer 4 riadky x 150 stĺpcov. Jeden 1 kvet je opísaný 4 parametrami, preto jeden stĺpec reprezentuje jednu vzorku kvetu. Vstupný dataset obsahuje 150 vzoriek kvetov. Matica irisTargets špecifikuje zaradenie každej vstupnej vzorky do jednej z 3 tried. Aby sme lepšie porozumeli údajom, obe matice postupne vypíšeme v príkazovom okne. Ukážky údajov môžeme vidieť vo výpisoch 4 a 5.

```
>> 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
```

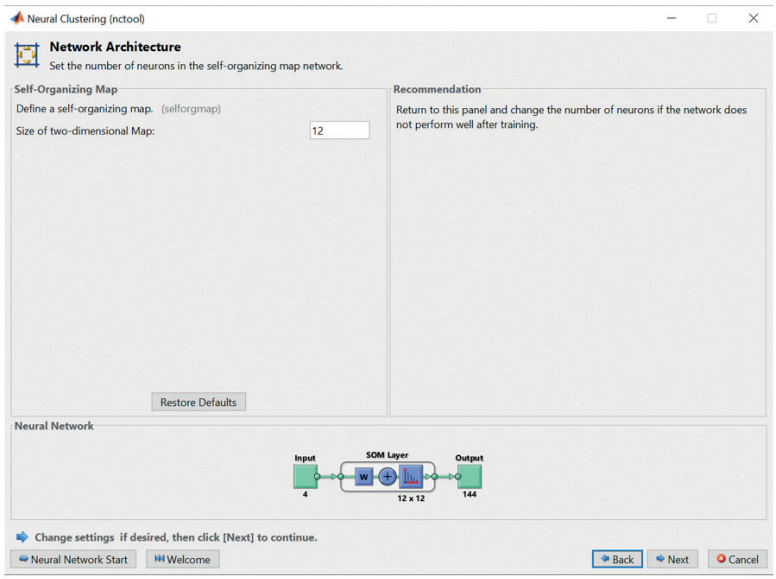
Výpis 4: Ukážka datasetu irisInputs

```
>> 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
```

Výpis 5: Ukážka datasetu irisTargets (očakávaných výstupov)

Krok 4: Rozdelenie vzoriek na trénovacie a testovacie je prednastavené a preto to nie je potrebné zadávať v tejto aplikácii.

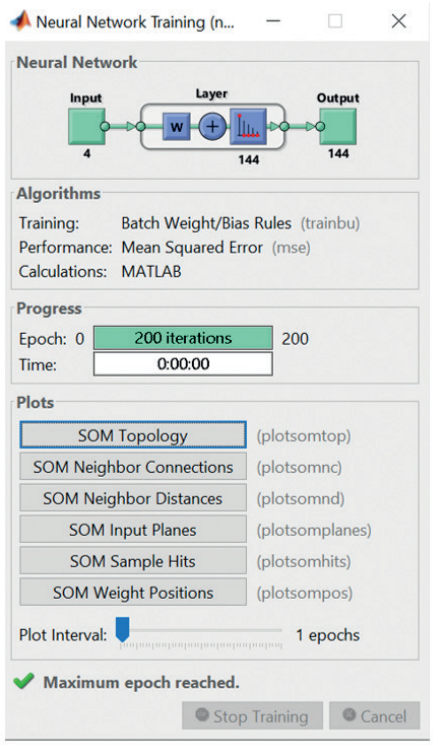
Krok 5: Navrhneme architektúru siete. V tomto prípade je potrebné zadať počet neurónov vrstvy SOM. Vrstva predstavuje dvojrozmerné štvorcové pole. Preto, keď zadáme veľkosť poľa 12, vytvorí sa dvojrozmerné pole 12x12 prvkov. Pozri obrázok 27. Potom mapa na výstupe siete bude mať rozmer 12x12, t.j. 144 prvkov. Preddefinovaná topológia výstupnej mapy je hexagonálna.



Obrázok 27. Návrh architektúry siete.

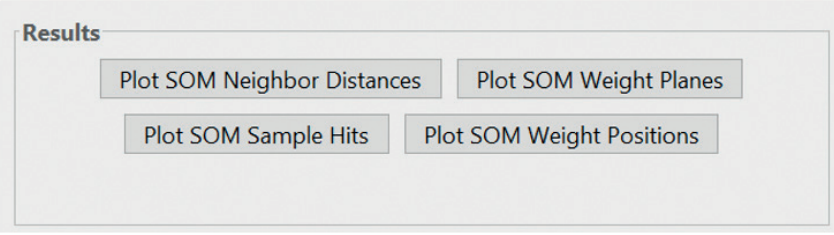
Krok 6: Výber učiaceho algoritmu. V aplikácii je preddefinovaný dávkový SOM algoritmus, preto tento krok vynecháme.

Krok 7: Proces učenia sa siete spustíme stačením tlačidla Train. Počet epoch učenia je nastavený na 200 a proces učenia sa môžeme sledovať tak, ako to vidíme na obrázku 28.

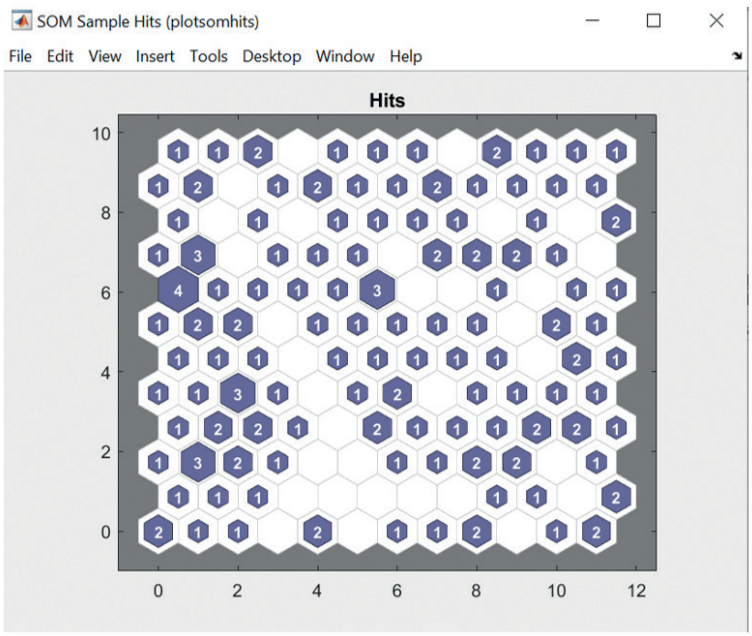


Obrázok 28. Proces učenia sa siete.

Krok 8: Výsledky učenia siete zobrazíme pomocou 4 tlačidiel (pozri obrázok 29). Prvý výsledok je klasifikácia tried pre každý kvet a Plot SOM sample hits (pozri obrázok 30) ukazuje počet kvetov v každej triede. Oblasti neurónov s vyššími hodnotami reprezentujú triedy podobných často zastúpených kvetov. Naopak, oblasti s malými hodnotami znamenajú kvety s menej početným výskytom.

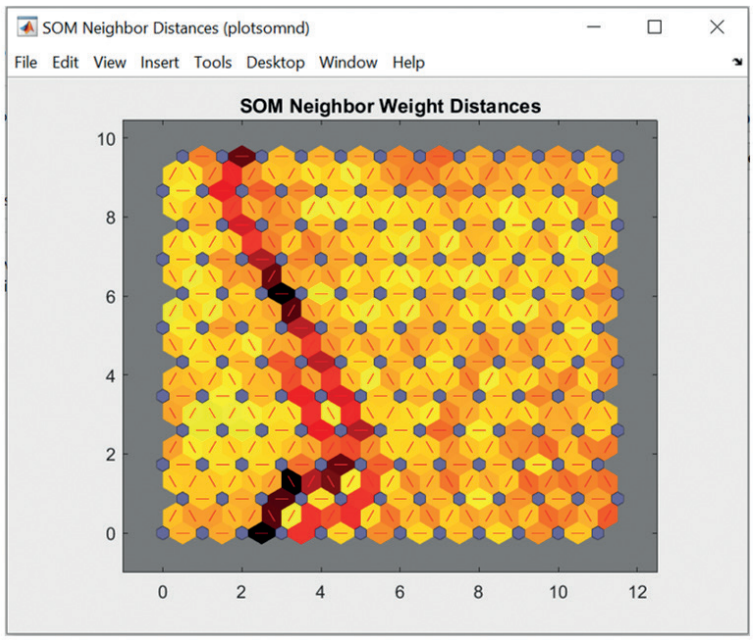


Obrázok 29. Výsledky učenia siete.



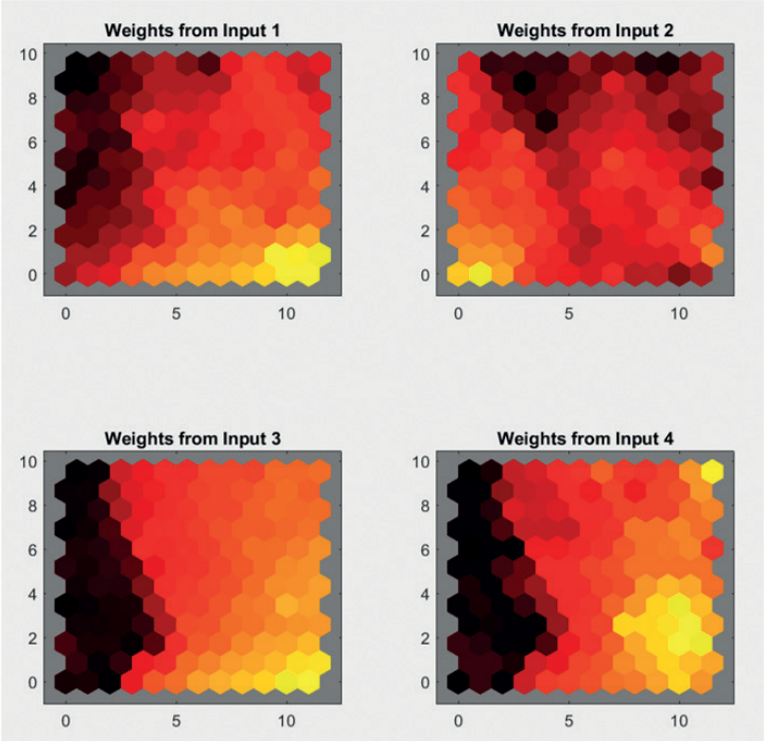
Obrázok 30. Výsledky učenia siete - počet kvetov v každej triede.

Výsledok, ktorý zobrazíme pomocou Plot SOM Neighbor Distances vyjadruje euklidovskú vzdialenosť triedy neurónov od jej susedov. Skupiny neurónov, ktoré tvoria svetlé spojenia znamenajú veľkú podobnosť kvetov vo vstupnom súbore. Naopak, tmavé svetlé spojenia reprezentujú vzdialené oblasti s nízkym počtom kvetov, alebo oblasti bez kvetov. Pozri obrázok 31. Tmavé hranice (spojenia) oddeľujú veľké oblasti vstupného priestoru a indikujú, že kvety v oddeleným oblastiach majú odlišné črty.



Obrázok 31. Výsledky učenia siete - početnosť a triedy kvetov vo vstupnom priestore.

Výsledné váhy siete z pohľadu 4 vstupných čŕt kvetov zobrazíme pomocou Plot SOM weight planes. Pozri obrázok 32.



Obrázok 32. Výsledky učenia siete – mapy váh pre jednotlivé vstupy siete.

Váhy prepájajú každý vstup s každým zo 144 výstupných neurónov siete. Tmavé farby reprezentujú väčšie váhy. Vstupy, ktoré majú rovnakú farbu na mape spolu silne korelujú.

Záver

V tejto kapitole sme sa naučili základy práce s Matlabom. Naučili sme sa pripravovať a upravovať namerané údaje, ktoré sme neskôr použili na učenie neurónových sietí. Predstavili sme metodický postup tvorby neurónových sietí v Matlabe pomocou aplikácií a grafickým prostredím. Naučili sme sa tvoriť jednoduché neurónové siete pomocou 3 riešených príkladov tvorby neurónových sietí.

KAPITOLA 12

PRÍLOHY

Táto časť obsahuje prílohy k práci uvedenej v hlavnej časti prezentovanej učebnice. Nasledovných päť príloh obsahuje údaje a informácie o cvičeniach súvisiace s učebnicou a kurzom, v ktorom možno učebnicu použiť, konkrétne:

- **Príloha A** opisuje dataset Iris používaný v príkladoch sekcií 3 – 8.
- **Príloha B** obsahuje príklady riešení problémov prezentovaných v sekcii 7.
- **Príloha C** je zameraný na prezentáciu vybraných datasetov venujúcich sa znečisteniu ovzdušia a klimatickým zmenám, ktoré je možné použiť ako zdroje dát pre metódy dátovej analýzy.
- **Príloha D** opisuje dopad znečistenia ovzdušia na ľudské zdravie.
- **Príloha E** obsahuje návrh kurikula pre kurz, v ktorom môžeme byť táto učebnica použitá.

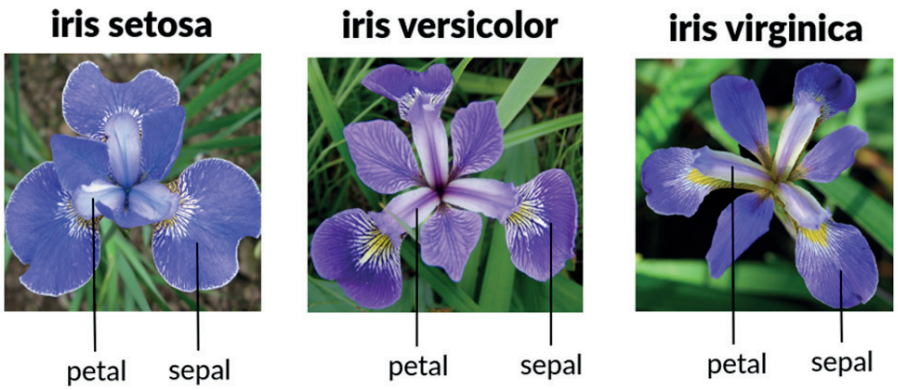
PRÍLOHA A STRUČNÝ OPIS DATASETU IRIS

Autorom tejto časti učebnice sú Alžbeta Michalíková a Adam Dudáš z Katedry informatiky, Fakulty prírodných vied, Univerzity Mateja Bela v Banskej Bystrici zo Slovenska.

Dataset Iris je jedným z najčastejšie používaných datasetov pri analýze dát, práci s predikčnými modelmi a pri ladení algoritmov na spracovanie dát.

Tento dataset bol vytvorený Edgerom Andersenom a prvýkrát bol kontexte v analýzy dát prezentovaný v publikácii *Fisher, R.A. "The use of multiple measurements in taxonomic problems" Annual Eugenics, 1936.* Tento dataset obsahuje **päť atribútov** meraných na **150 jedincoch** kvetu kosatca (dataset má veľkosť 150 × 5). Kvet pozostáva zo šiestich lístkov štruktúrovaných v dvoch cykloch, kde:

- lístky sepal tvoria vnútorný cyklus kvetu,
- lístky petal tvoria vonkajší cyklus kvetu.



Dataset Iris bol zostavený meraním dvoch hodnôt pre každý typ lístku (meraním jeho šírky a dĺžky), čo tvorí **štyri číselné atribúty**:

- dĺžka a šírka sepal lístkov meraná v centimetroch alebo milimetroch,
- dĺžka a šírka petal lístkov meraná v centimetroch alebo milimetroch.

Piatym atribútom datasetu je kategorická hodnota **trieda** niekedy označovaná ako **druh**, pomocou ktorého sú entity datasetu rozdelené do troch tried:

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

Každá z týchto tried je v datasete zastúpená rovnomerne – **50 entitami**. Uvádzame príklad jednej vzorovej entity z každej triedy datasetu Iris:

Entita	Sepal dĺžka	Sepal šírka	Petal dĺžka	Petal šírka	Trieda
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

Práca s datasetom Iris

Dataset Iris je tak štandardizovaný, že väčšina nástrojov na spracovanie a analýzu dát má interný príkaz, ktorý možno použiť na načítanie tohto datasetu.

Napríklad v jazyku R namiesto názvu dátového súboru používame len príkaz *iris*.

Príklad: Zadaním názvu načítaného datasetu v jazyku R získame konzolový výstup, ktorý obsahuje všetky atribúty a entity datasetu. V prípade datasetu Iris môžeme zadať iris (bez potreby načítať dataset).

```
> 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
```

V prípade práce s nástrojom, ktorý nemá dataset Iris takto dostupný, je možné dataset voľne stiahnuť napríklad na:

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

PRÍLOHA B RIEŠENIA OTÁZOK ZAMERANÝCH NA FUZZY KLASIFIKÁCIU

Autorom tejto časti učebnice je Alžbeta Michalíková z Katedry informatiky, Fakulty prírodných vied, Univerzity Mateja Bela v Banskej Bystrici zo Slovenska.

Zadanie:

Pomocou Sugenovej metódy klasifikujte údaje zo súboru údajov Iris do vhodného počtu tried.

Riešenie:

Odpovedajte na nasledujúce otázky:

1. *Koľko **vstupných premenných** je v datasete Iris?*
   V datasete Iris máme štyri vstupné jazykové premenné.

2. *Čo použijeme **na popis vstupných premenných**?*
   Na popis vstupných jazykových premenných použije fuzzy množiny. Budú zadané pomocou funkcií príslušnosti.

3. *Aký **typ fuzzy funkcií príslušnosti** použijeme?*
   Použijeme lichobežníkové funkcie príslušnosti.

4. *Aký bude **výstup**?*
   výstupom systému bude konkrétna trieda, do ktorej patria jednotlivé kvety Iris (riadky tabuľky/objektov).

5. *Čo použijeme **na popis výstupných premenných**?*
   Na popis výstupných premenných použijeme konštantné funkcie (konštanty).

6. *Aký **typ pravidiel** použijeme?*
   Budeme používať Sugenove pravidlá typu AK-POTOM.

7. *Napíšte **príklad jedného pravidla**!*
   AK Vstup1 je malý a Vstup2 je malý a Vstup3 je stredný a Vstup4 je vysoký, POTOM Výstup je trieda1 (alebo Iris_Setosa).

Určte hodnoty parametrov vstupných jazykových premenných a vyplňte ich do nasledujúcich tabuliek (Tabuľka B.1).

Poznámka:

Pri fuzzy množinách nemusíte dostať totožné riešenie s riešením, ktoré je uvedené v Tabuľkách. Riešenie je vždy závislé od pozorovateľa, teda od Vás.

Tabuľka B.1: Parametre vstupných premenných

Vstup1:

Názov	Parametre
Definičný obor	**[40 80]**
Červená	[-20 -10 48 59]
Modrá	[48 55 67 71]
Zelená	[55 71 81 90]

Vstup2:

Názov	Parametre
Definičný obor	**[20 45]**
Červená	[22 39 46 50]
Modrá	[0 10 24 35]
Zelená	[21 28 34 39]

Vstup3:

Názov	Parametre
Definičný obor	**[10 70]**
Červená	[0 5 19 28]
Modrá	[26 30 44 52]
Zelená	[44 53 75 80]

Vstup4:

Názov	Parametre
Definičný obor	**[0 25]**
Červená	[-10 -5 6 10]
Modrá	[6 10 13 19]
Zelená	[13 19 30 35]

Určte hodnoty výstupných parametrov. Vyplňte Tabuľku B.2 správnymi hodnotami, ak uvažujete, že pre výstupnú jazykovú premennú použijete **konštantné funkcie**.

Tabuľka B.2: Parametre výstupnej premennej

Výstup:

Názov	Parametre
Definičný obor	**[1 3]**
Červená	1
Modrá	2
Zelená	3

Navrhnite počet pravidiel, ktoré použijete a napíšte ich v správnom tvare.

Pravidlá:

1. AK vstup1 je červený a vstup2 je červený a vstup3 je červený a vstup4 je červený, POTOM výstup je červený.
2. Ak je vstup1 je modrý a vstup2 je modrý a vstup3 je modrý a vstup4 je modrý, POTOM výstup je modrý.
3. Ak je vstup1 je zelený a vstup2 je zelený a vstup3 je zelený a vstup4 je zelený, POTOM výstup je zelený.

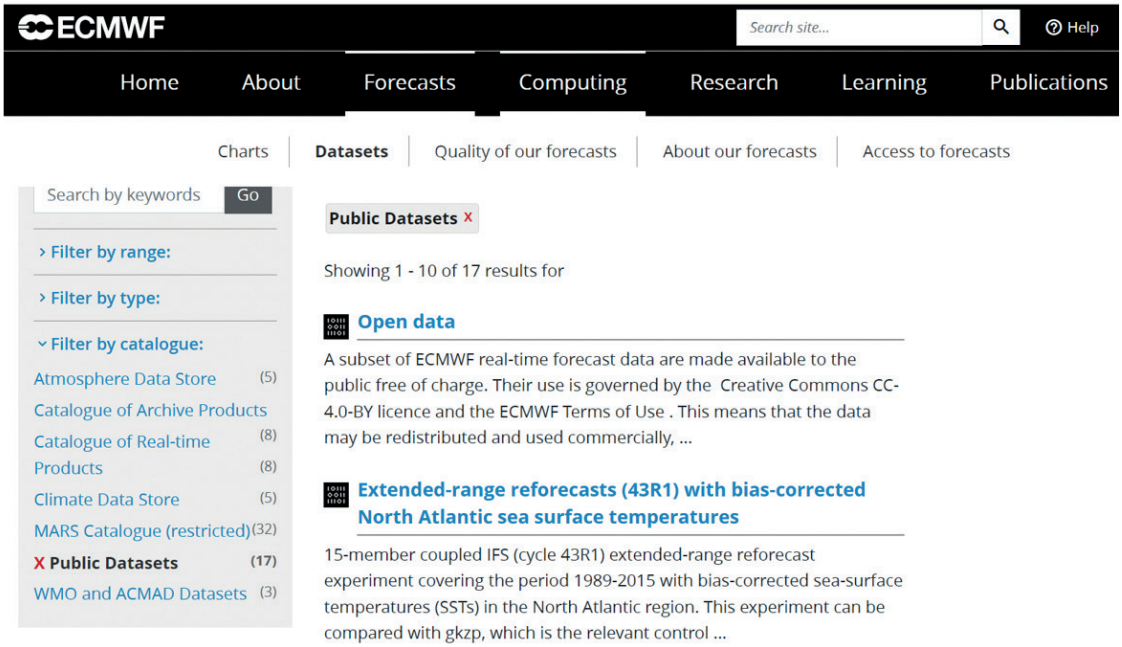
C STRUČNÝ OPIS DATASETOV ZAMERANÝCH NA KLIMATICKÉ ZMENY

Túto časť učebnice napísali Mihaela Tinca Udristioiu z Katedry fyziky Prírodovedeckej fakulty a Silvia Puiu z Katedry manažmentu, marketingu a podnikovej správy Fakulty ekonomiky a podnikovej správy Univerzity v Craiove, Rumunsko.

Výskum môže napredovať ľahšie a rýchlejšie, ak majú výskumníci a používatelia vo všeobecnosti otvorený prístup k informáciám. Keďže máme internet na dosah ruky, musíme vedieť, kde hľadať presné, aktuálne a spoľahlivé zdroje informácií. Všetky tieto dôvody zdôrazňujú, prečo je úloha databáz taká dôležitá. Údaje sú štruktúrované a to zvyčajne tak, aby sa dali ľahko transformovať a spracovať podľa potrieb výskumníka alebo používateľa.

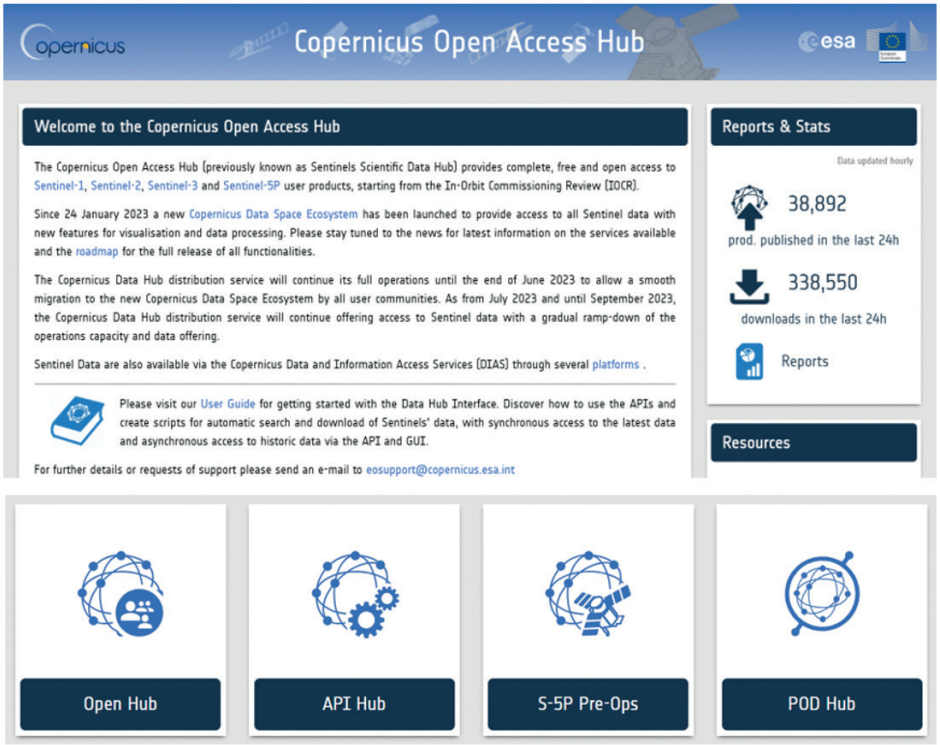
Európska komisia vytvorila niekoľko otvorených zdrojov údajov o zmene klímy. Je ľahké tieto súbory údajov stiahnuť, ťažšie je analyzovať ich prostredníctvom exploratívnej a prediktívnej analýzy a používať rôzne algoritmy na vytváranie matematických modelov. Copernicus, European Climate Assessment and Dataset, Climate Explorer a Indecis sú len niektoré z nich. Potrebujeme údaje na monitorovanie klimatických zmien a predpovedanie počasia, pozorovanie citlivosti klímy na rôzne parametre, vytváranie rôznych scenárov a sledovanie vývoja niektorých procesov v krátkodobom a dlhodobom horizonte. V nasledujúcom texte autori stručne predstavia niektoré databázy.

Európske centrum pre strednodobú predpoveď počasia (ECMWF) spracúva údaje z približne 90 satelitných prístrojov v rámci každodenných operatívnych činností asimilácie a monitorovania údajov. V integrovanom predpovednom systéme je denne k dispozícii približne 60 miliónov pozorovaní s kontrolovanou kvalitou, z ktorých väčšinu tvoria satelitné merania. ECMWF využíva aj všetky pozorovania z iných ako satelitných zdrojov vrátane hlásení z povrchu a z lietadiel.



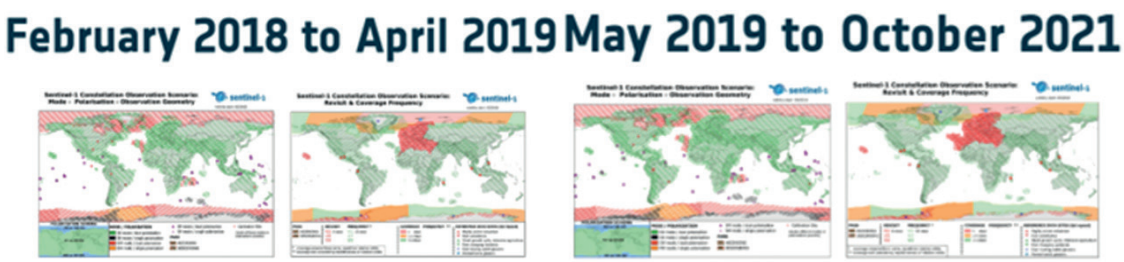
Obrázok C.1. Snímka obrazovky časti Súbor verejných údajov z ECMWF (zdroj: https://www.ecmwf.int/en/forecasts/datasets/search)

Program Copernicus je súčasťou vesmírneho programu EÚ na pozorovanie Zeme. Riadi ho Európska komisia (EK). EK realizuje program Copernicus v spolupráci s členskými štátmi EÚ, Európskou vesmírnou agentúrou (ESA), Európskou organizáciou pre využívanie meteorologických družíc (EUMETSAT), Európskym strediskom pre strednodobé predpovede počasia (ECMWF), Spoločným výskumným centrom (JRC), Európskou environmentálnou agentúrou (EEA), Európskou námornou bezpečnostnou agentúrou (EMSA), agentúrou Frontex, organizáciou SatCen a Mercator Océan. Program Copernicus obsahuje súbory klimatických údajov z rôznych zdrojov (reanalýzy, satelitné produkty, klimatické prognózy). Databáza Copernicus je jedným z najčastejšie používaných súborov údajov o zmene klímy, pracuje s predpovednými modelmi a ladí algoritmy na spracovanie údajov. Má satelity (SENTINELS 1-6) s významnými misiami.



Obrázok C.2: Obrázok rozhrania Copernicus Open Access Hub (zdroj: https://scihub.copernicus.eu/)

SENTINEL-1 má dva satelity na polárnej obežnej dráhe a pracuje 24 hodín z 24 a 7 dní zo 7, bez prázdnin. Využíva radarové snímanie na zber snímok bez ohľadu na počasie.



Obrázok C.3 : Obraz poskytnutý SENTINELOM 1 pre dva časové intervaly
(zdroj: https://sentinels.copernicus.eu/web/sentinel/missions/sentinel-1/observation-scenario/archive)

SENTINEL-2 pozostáva z dvoch satelitov na polárnej obežnej dráhe na rovnakej slnečnej synchrónnej dráhe, vzdialených od seba 180°. Sledujú zmeny podmienok na zemskom povrchu. Veľká šírka záberu (290 km) a dlhý čas návratu (10 dní na rovníku s jednou družicou a päť dní s dvoma družicami za bezoblačných podmienok, čo predstavuje 2 - 3 dni v stredných zemepisných šírkach) pomáhajú pri monitorovaní zmien zemského povrchu. Používatelia majú k dispozícii produkty SENTINEL-2. Niektoré produkty sú určené len pre odborníkov (žiarenia z vrcholu atmosféry v geometrii senzorov) a iné sú určené pre všetkých používateľov (odrazivosti z vrcholu atmosféry v kartografickej geometrii a atmosféricky korigované odrazivosti povrchu v tej istej geometrii). Pilotné produkty sa generujú len na požiadanie. Existujú dve kategórie pilotných produktov: harmonizované povrchové odrazivosti SENTINEL-2 a Landsat-8/9 v kartografickej geometrii.

SENTINEL-3 vykonáva merania topografie morského povrchu, teploty morského a pevninského povrchu a farby oceánu a pevniny. Cieľom je podporiť systémy predpovedania oceánov a monitorovanie životného prostredia a klímy.

SENTINEL-4 monitoruje kľúčové stopové plyny a aerosóly v kvalite ovzdušia nad Európou, čím podporuje službu Copernicus Atmosphere Monitoring Service (CAMS) s rýchlym časom revízie. Spektrálne a rádiometricky kalibrovaná a geolokalizovaná radiácia Zeme a spektrálne a rádiometricky kalibrovaná slnečná radiácia sú k dispozícii ako parametre pre všetkých používateľov, ale parametre spracovania údajov, kalibrácia a diagnostické údaje prístroja sú k dispozícii len pre odborných používateľov.

SENTINEL-5 je spektrometrický systém s vysokým rozlíšením pracujúci v ultrafialovej až krátkovlnnej infračervenej oblasti so siedmimi rôznymi spektrálnymi pásmi: 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) a SWIR-3 (2305-2385 nm). Sentinel-5 poskytuje informácie o kvalite ovzdušia a interakcii medzi zložením a klímou (O_3, NO_2, SO_2, HCHO, CHOCHO a aerosóly). Sentinel-5 poskytuje kvalitatívne parametre pre CO, CH_4 a stratosférický O_3 s denným globálnym pokrytím pre aplikácie v oblasti klímy, kvality ovzdušia a ozónu/povrchového UV žiarenia.

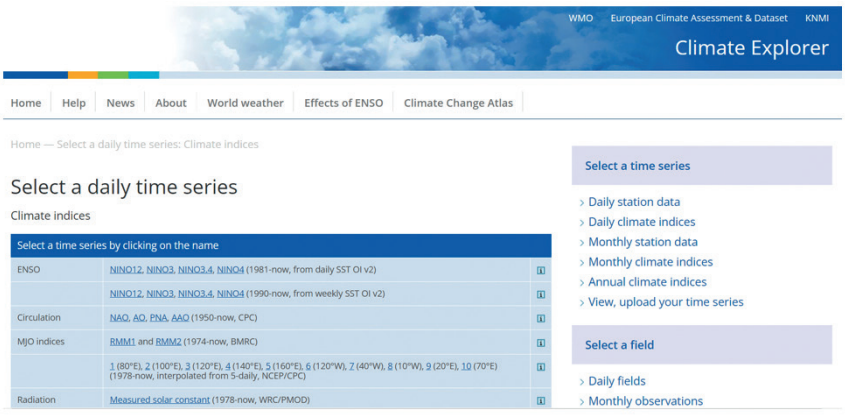
SENTINEL-5P vykonáva atmosférické merania s vysokým časopriestorovým rozlíšením na monitorovanie kvality ovzdušia, ozónu a UV žiarenia a na monitorovanie a predpovedanie klímy.

Copernicus SENTINEL-6 Michael Freilich sa zameriava na zvyšovanie hladiny morí v dôsledku klimatických zmien a je ďalšou referenčnou misiou radarovej altimetrie, ktorá má predĺžiť merania výšky morskej hladiny minimálne do roku 2030.

Ďalšou dôležitou databázou je **ECA&D**, ktorá obsahuje pozorovania z meteorologických staníc a súbory údajov z nich odvodených na európskej úrovni; výskumníci považujú tieto súbory údajov za referenčné údaje. Táto stránka obsahuje informácie týkajúce sa zmien extrémov počasia a klímy a denné súbory údajov potrebné na monitorovanie a analýzu týchto extrémov.

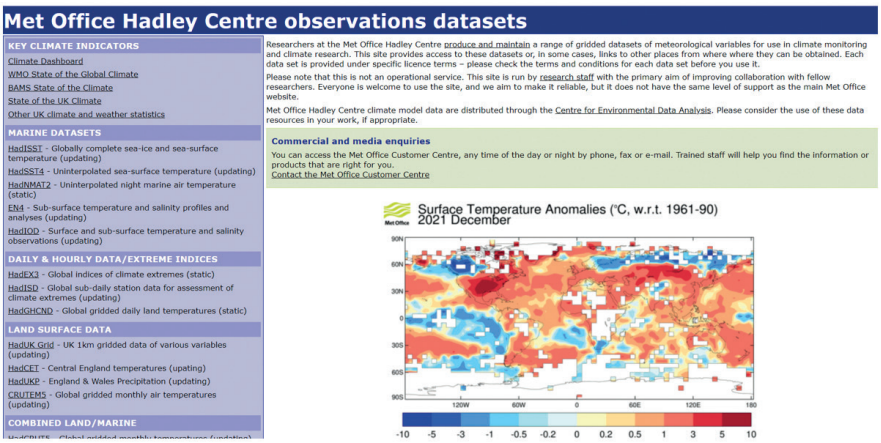
Obrázok C.4: Rozhranie ECA&D a WMO (zdroj: https://www.ecad.eu)

KNMI Climate Explorer je ďalšia databáza, ktorá obsahuje triedu klimatických údajov (časových radov alebo polí) z reanalýz a klimatických modelov vrátane klimatických prognóz; jej výhodou je priateľskejšie rozhranie (vrátane grafických zobrazení). Z tohto dôvodu predstavuje dobrý vzdelávací nástroj. Používatelia si môžu stiahnuť časové rady o denných a mesačných údajoch zo staníc a klimatických indexoch. Na ročnej úrovni sú k dispozícii len ročné klimatické indexy. Výskumníci si môžu tieto informácie stiahnuť výberom poľa, ako sú denné polia, mesačné pozorovania, mesačné polia reanalýzy, mesačné a sezónne historické rekonštrukcie, mesačné sezónne hindcasty, mesačné priebehy scenára CMIP3+, mesačné priebehy scenára CMIP5, ročné extrémy CMIP5, mesačné priebehy scenára CMIP6, mesačné priebehy scenára CORDEX, atribučné priebehy.



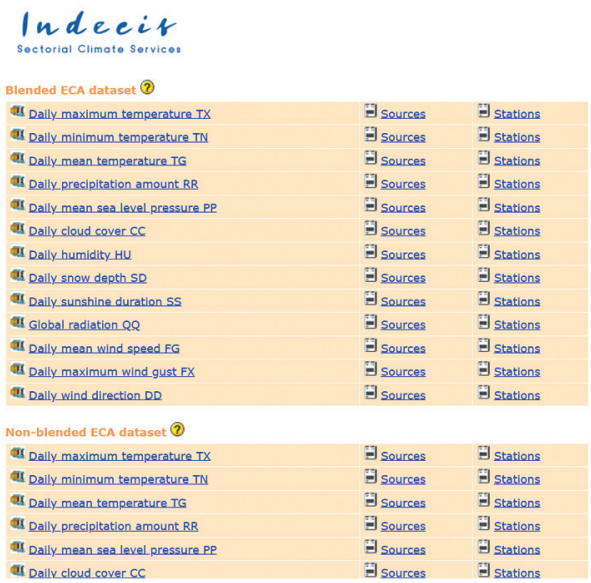
Obrázok C.5: Snímka obrazovky z KNMI Climate Explorera
(zdroj: https://climexp.knmi.nl/selectdailyindex.cgi?id=someone@somewhere)

Met Office Hadley Centre poskytuje súbory údajov o meteorologických premenných. Vedci tieto informácie využívajú pri monitorovaní klímy a výskume klímy. Ide o tieto triedy: kľúčové klimatické ukazovatele, súbory údajov o moriach, denné a hodinové údaje/extrémne indexy, údaje o zemskom povrchu, kombinované údaje o zemskom/morskom tlaku, údaje o horných vrstvách ovzdušia, jednorazové údaje sprevádzajúce články v časopisoch a staršie súbory údajov.



Obrázok C.6: Rozhranie Met Office Hadley Centre
(zdroj: https://www.metoffice.gov.uk/hadobs/index.html)

Indecis obsahuje klimatické údaje o poľnohospodárstve, znižovaní rizika katastrof, energetike, zdraví, vode a cestovnom ruchu (http://indecis.eu/indices.php). Tu sú len klimatické indexy - mnohé, s rôznymi aplikáciami; platforma má definície klimatických indexov s grafickým znázornením vo forme mapy a série údajov v jednom bode, plus možnosť stiahnutia. Táto databáza je tiež dobrou vzdelávacou pomôckou. Obsahuje denné údaje staníc, údaje staníc kontrolujúcich kvalitu, homogenizované údaje staníc, obnovené údaje staníc a mriežkové verzie indexov.

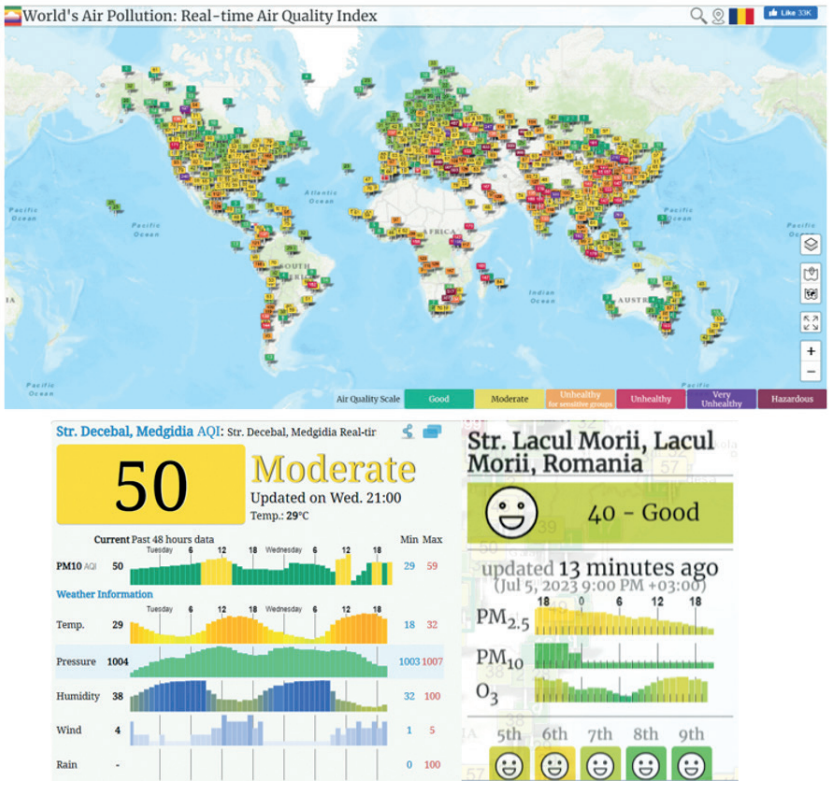


Obrázok C.7 : Triedy údajov, ktoré možno prevziať zo systému Indecis
(zdroj: https://www.ecad.eu/dailydata/predefinedseries.php)

Európska environmentálna agentúra poskytuje údaje o kvalite ovzdušia s otvoreným zdrojovým kódom.

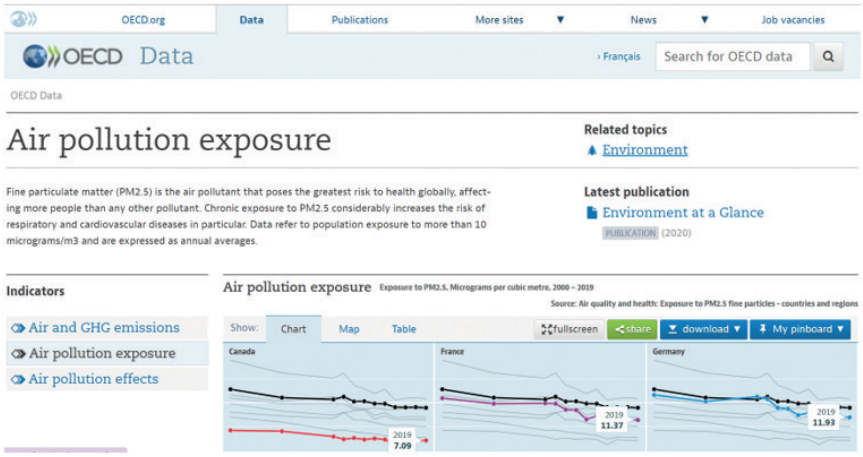
Obrázok C.8: Súbory údajov poskytnuté Európskou environmentálnou agentúrou (zdroj: https://www.eea.europa.eu/themes/air/explore-air-pollution-data)

Svetové znečistenie ovzdušia obsahuje senzory národných environmentálnych agentúr a poskytuje informácie o indexe kvality ovzdušia v reálnom čase.



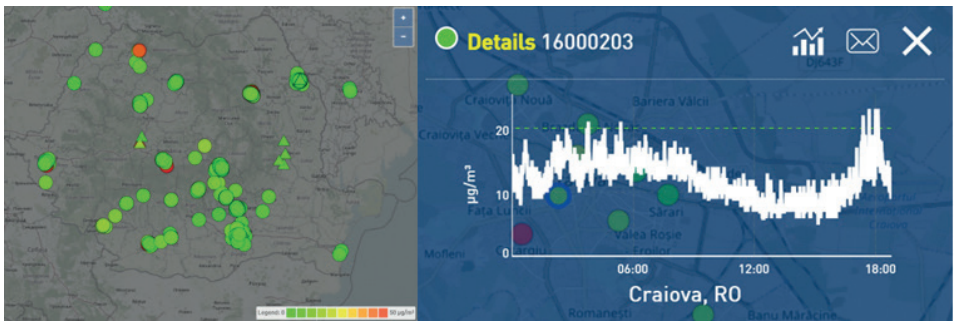
Obrázok C.9 : Mapa senzorov kvality ovzdušia na celom svete a ďalšie informácie po kliknutí na senzor (zdroj: https://waqi.info/)

OECD obsahuje informácie o ukazovateľoch, ako sú emisie do ovzdušia a GHC, vystavenie znečisteniu ovzdušia a účinky znečistenia ovzdušia, vo forme grafov, máp alebo tabuliek, aby si každý mohol jednoducho vizualizovať vývoj údajov v čase. Jednoduchým kliknutím sa otvoria údaje usporiadané do grafov, máp a tabuliek.



Obrázok C.10 : Rozhranie OECD (zdroj: https://data.oecd.org/air/air-pollution-exposure.htm)

Existujú aj občianske vedecké iniciatívy a komunitou riadené siete senzorov. Tieto siete obsahujú nízko nákladové senzory, ktoré monitorujú kvalitu ovzdušia občanmi v ich komunitách a majú vynikajúce pokrytie veľkej oblasti Európy. Niektoré z týchto sietí vybudovali dobrovoľníci v rámci niektorých projektov na vzdelávacie účely. uRADMonitor® je takýmto príkladom v Rumunsku. Sieť poskytuje otvorený prístup k údajom v reálnom čase. Správcovia môžu na požiadanie poskytnúť historické údaje. Iniciatívy občianskej vedy podporujú transparentnosť a zodpovednosť pri monitorovaní životného prostredia. Ďalšie príklady sú nasledovné: Sieť komunitných senzorov ovzdušia (CAIRSENSE), sieť Smart Citizen®, Verejné laboratórium pre otvorenú technológiu a vedu alebo sieť Public Lab, iniciatíva Eye on Earth, Globálne vzdelávanie a pozorovanie v prospech životného prostredia (GLOBE), HabitatMap©, projekt komunitného monitorovania ovzdušia v oblasti Imperial County a Program občianskeho pozorovania počasia (CWOP).



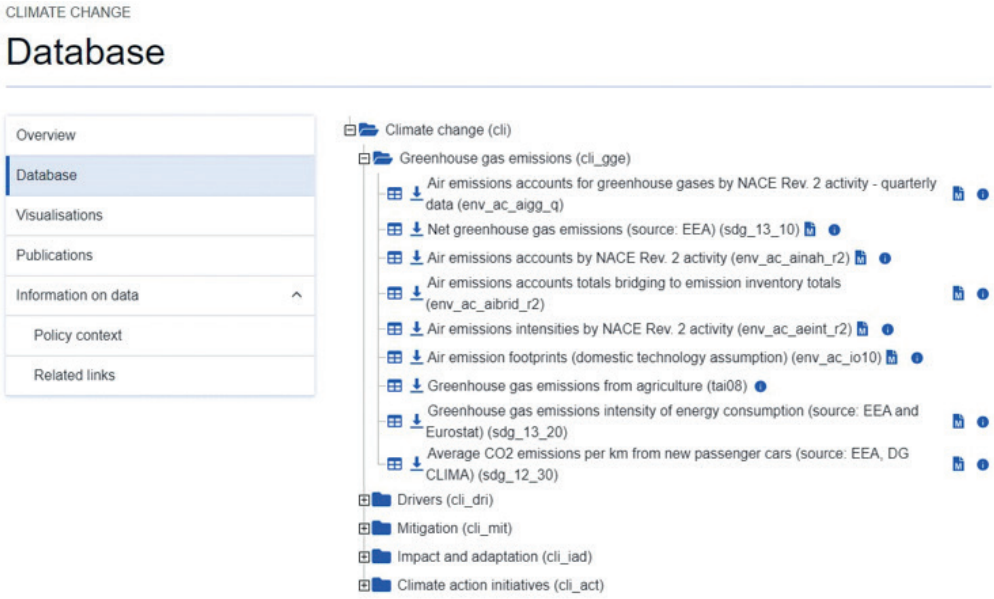
Obrázok C.11: Snímka siete uRADMonitor® (zdroj: https://www.uradmonitor.com/)

Štatistické databázy pomáhajú výskumu napredovať, pretože poskytujú dôležité údaje o viacerých premenných a dlhších obdobiach. Tieto informácie majú zásadný význam pri vyvodzovaní záverov,

vytváraní, predpovedaní alebo zmierňovaní scenárov. Ak napríklad potrebujeme údaje o zmene klímy, napríklad o úrovni znečistenia, máme k dispozícii viacero databáz s otvoreným zdrojovým kódom, ku ktorým môžeme pristupovať a využívať ich pre naše ciele. Rozhodnutia sú založené na vstupných údajoch; naše rozhodnutia sú dobré len vtedy, ak máme správne informácie. Preto je dôležité vybrať si spoľahlivé zdroje informácií, napríklad od známych národných a medzinárodných organizácií.

Jedným z týchto zdrojov je databáza Eurostatu, ktorá poskytuje štatistické údaje o mnohých zaujímavých aspektoch európskych krajín. Jedným z dôvodov, prečo si môžeme byť touto databázou istí, je jej dlhá história (70 rokov) a skutočnosť, že ju zastrešuje Európska únia.

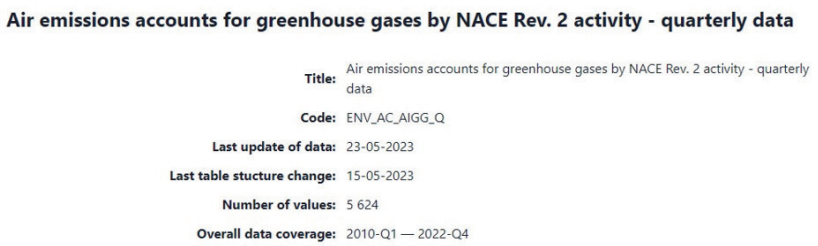
Uveďme si príklad, ako môžeme získať prístup k údajom o zmene klímy pomocou databázy Eurostatu. Môžeme vstúpiť priamo na webovú stránku a vyhľadať zmenu klímy alebo to urobiť pomocou vyhľadávača. Ak prejdeme na stránku https://ec.europa.eu/eurostat/web/climate-change/database, môžeme nájsť viacero údajov pre náš cieľ, ako je znázornené na obrázku C.12.



Obrázok C.12: Snímka obrazovky z webovej stránky Eurostatu týkajúci sa informácií o zmene klímy (zdroj: https://ec.europa.eu/eurostat/web/climate-change/database)

Databáza o zmene klímy má mnoho zložiek: emisie skleníkových plynov, faktory zmeny klímy, zmierňovanie zmeny klímy, vplyv a adaptácia a iniciatívy v oblasti klímy. Každá z nich obsahuje údaje, ktoré si používateľ môže stiahnuť. Informácie sú bezplatné a prístupné pre každého vo viacerých formátoch.

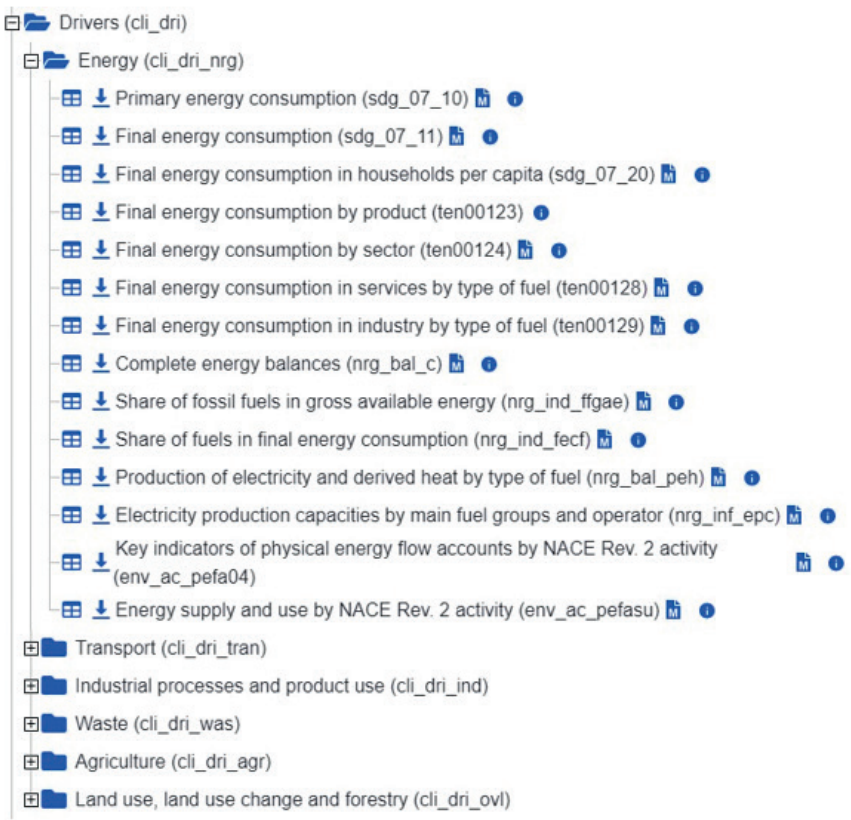
V priečinku s emisiami skleníkových plynov si môžeme všimnúť niekoľko údajov. Ak prejdeme na prvý z nich - Množstvo emisií skleníkových plynov v ovzduší (štvrťročné údaje), po kliknutí na pravé tlačidlo nájdeme ďalšie informácie. Zobrazí sa okno, ktoré je znázornené na obrázku C.13. Vidíme teda, že údaje sú k dispozícii za 13 rokov, od roku 2010 do roku 2022. Databáza bola aktualizovaná v máji 2023.



Obrázok C.13: Snímka obrazovky po kliknutí na informačné tlačidlo (zdroj: https://ec.europa.eu/eurostat/web/climate-change/database)

Ak máme záujem zistiť viac o faktoroch zmeny klímy, vyberieme druhú zložku a na obrázku C.14 si všimneme, že sú tu údaje o všetkých dôležitých faktoroch, ako sú energia, doprava, priemyselné procesy, odpady, poľnohospodárstvo a využívanie pôdy, zmeny vo využívaní pôdy a lesníctvo. V prípade zložky Energia si môžeme stiahnuť údaje týkajúce sa konečnej spotreby energie, konečnej spotreby energie na obyvateľa, konečnej spotreby energie podľa odvetví atď.

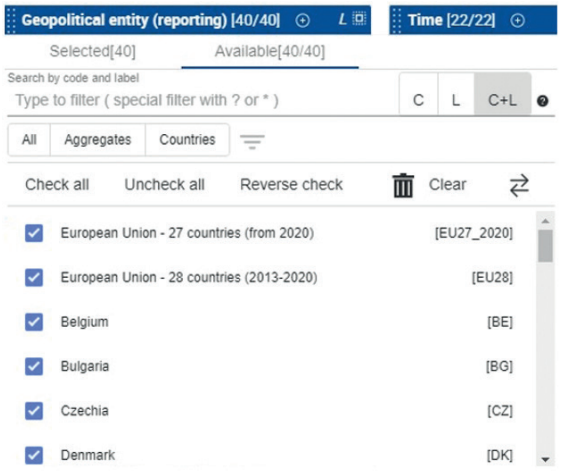
Dôležitý je výber údajov, ktoré používateľ potrebuje na dosiahnutie svojich cieľov. Údaje sú surové a nespracované, takže používateľ môže použiť niekoľko nástrojov na spracovanie údajov a všimnúť si trend, predpovedať niektoré scenáre a poskytnúť výsledky, ku ktorým dospel. Podniky, jednotlivci, vlády a iné zodpovedné osoby využijú tieto výsledky na prevenciu alebo zlepšenie niektorých aspektov.



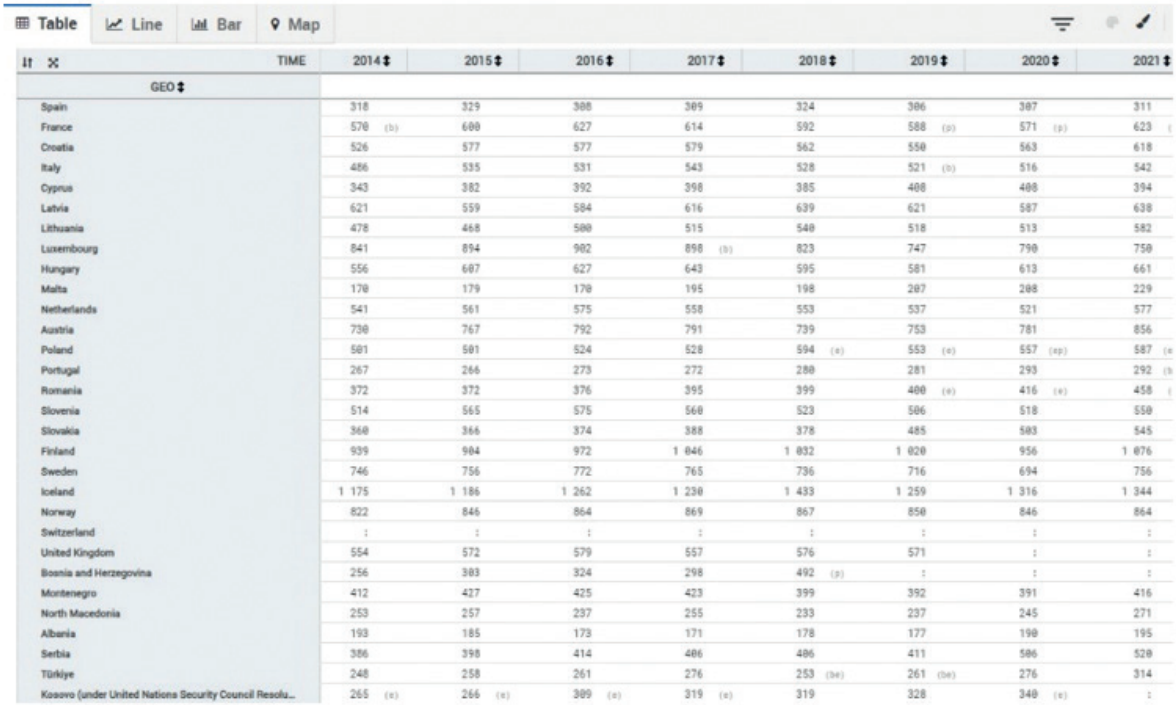
Obrázok C.14: Snímka obrazovky s informáciami z Eurostatu o faktoroch zmeny klímy (zdroj: https://ec.europa.eu/eurostat/web/climate-change/database)

Teraz sa pozrime, ako vyzerajú informácie, ak chceme skontrolovať konečnú spotrebu energie v domácnostiach na obyvateľa. Na obr. C.14 je pri tomto ukazovateli uvedený kód v zátvorkách: SDG 7. Táto informácia je v skutočnosti odkazom na siedmy cieľ udržateľného rozvoja z Agendy 2030 Organizácie Spojených národov. Ten sa týka dostupnej a čistej energie.

Ak klikneme na prvú ikonu, ktorá vyzerá ako tabuľka, môžeme si prečítať vysvetlenie týkajúce sa ukazovateľa, ale tiež si môžeme vybrať formát údajov (tabuľka, riadok, stĺpec, mapa) a premenné, ktoré potrebujeme (krajiny a roky) - obr. C.15 a C.16.

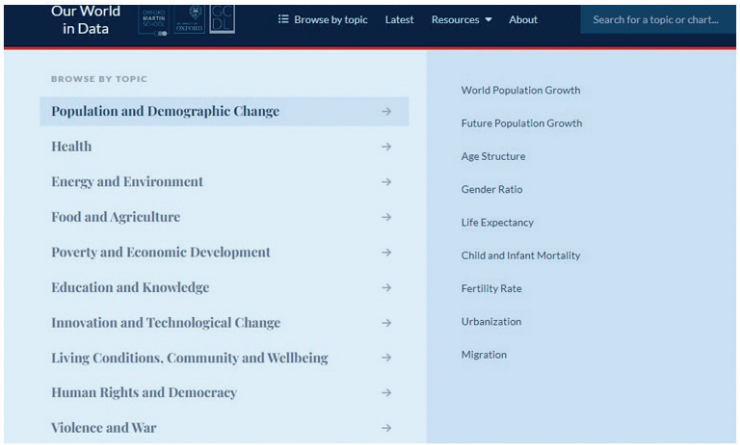


Obrázok C.15: Snímka obrazovky filtrov, ktoré možno použiť pre údaje z Eurostatu (zdroj: https://ec.europa.eu/eurostat/web/climate-change/database)



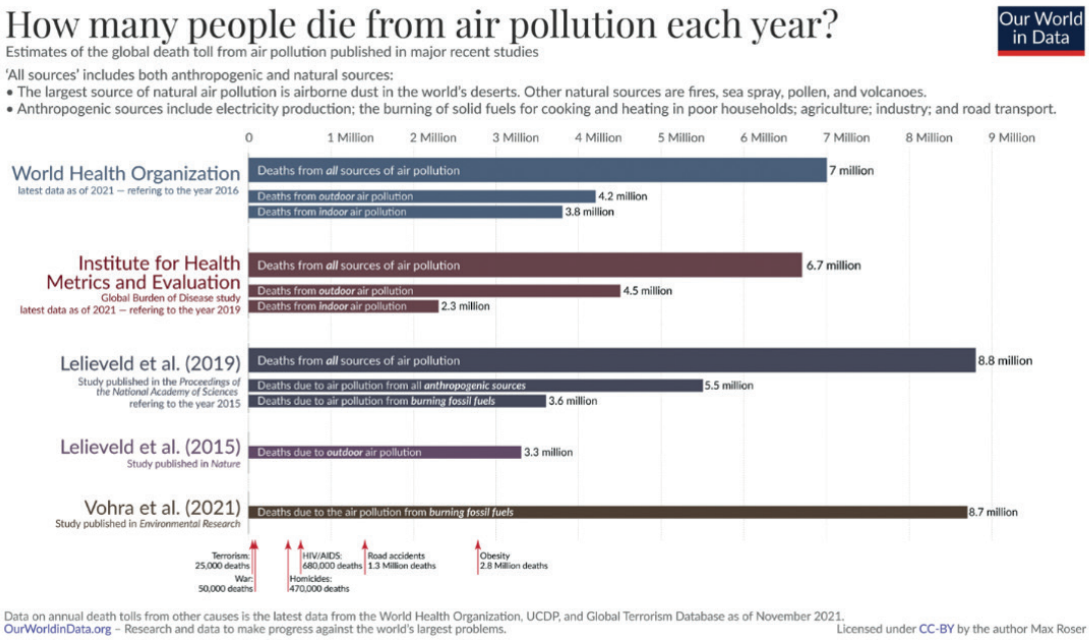
Obrázok C.16 : Snímka obrazovky zobrazených informácií, ak zvolíme formát tabuľky (zdroj: https://ec.europa.eu/eurostat/web/climate-change/database)

Okrem Eurostatu môžu výskumníci využívať aj iné databázy s otvoreným zdrojovým kódom. Môžeme spomenúť databázu Our World in Data, ktorá má na svojej webovej stránke (https://ourworldindata.org/) ako hlavný cieľ uvedené: uverejňujú "výskum a údaje na dosiahnutie pokroku v boji proti najväčším problémom sveta, ktoré sa týkajú populačnej dynamiky, energie a životného prostredia, zdravia, potravín, chudoby, vzdelávania, životných podmienok, ľudských práv, technologických zmien a násilia a vojny. Zastrešuje ju nezisková organizácia, ale je veľmi citovaná v prehľade literatúry a v médiách.

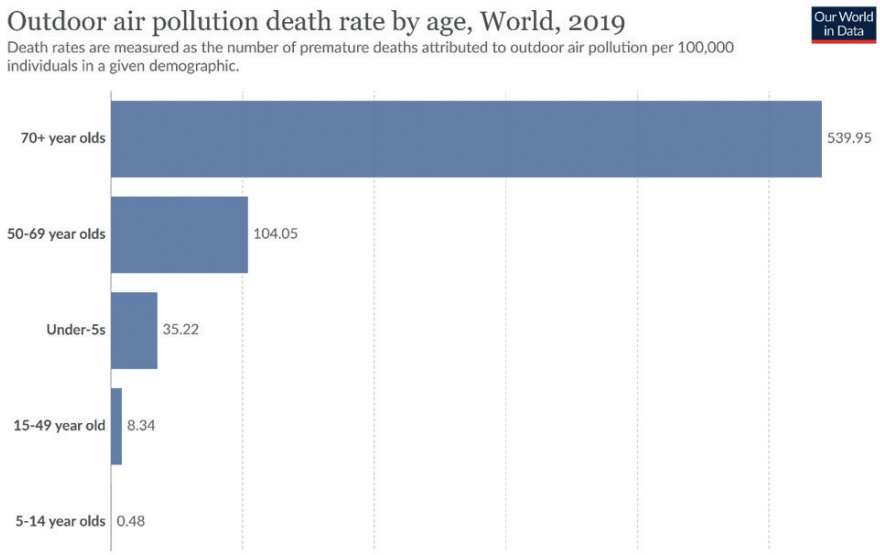


Obrázok C.17: Snímka obrazovky z Our World in Data týkajúci sa tém, ktorými sa zaoberá táto publikácia (zdroj: https://ourworldindata.org/)

Ak nás zaujíma znečistenie ovzdušia, vyberieme položku Energia a životné prostredie a môžeme si vybrať znečistenie vonkajšieho alebo vnútorného ovzdušia. Táto webová stránka ponúka články a prvotné štatistické údaje pre váš výskum alebo činnosť. Môžete tak zistiť, koľko úmrtí na celom svete možno pripísať znečisteniu ovzdušia (obr. C.18). Môžete tiež zistiť mieru úmrtí v dôsledku ovzdušia podľa veku a stiahnuť si údaje vo forme tabuľky alebo grafu (obr. C.19).



Obrázok C.18: Celosvetové úmrtia v dôsledku znečistenia ovzdušia
(zdroj: https://ourworldindata.org/data-review-air-pollution-deaths)



Obrázok C.19: Úmrtnosť na znečistenie ovzdušia podľa veku
(zdroj: https://ourworldindata.org/outdoor-air-pollution#outdoor-air-pollution-deaths-by-age)

PRÍLOHA D VPLYV ZNEČISTENIA OVZDUŠIA NA ĽUDSKÉ ZDRAVIE

Túto časť učebnice napísala Slaveya Petrova z Katedry ekológie a ochrany životného prostredia Biologickej fakulty Univerzity Paisij Chilendarskieho v Plovdive, Bulharsko

Znečistenie ovzdušia je kontaminácia vnútorného alebo vonkajšieho prostredia akýmkoľvek chemickým, fyzikálnym alebo biologickým činidlom, ktoré modifikuje prirodzené vlastnosti atmosféry.

Častým zdrojom znečisťovania ovzdušia sú domáce spaľovacie zariadenia, motorové vozidlá a priemyselné zariadenia. Znečisťujúce látky, ktoré sú hlavným problémom verejného zdravia, zahŕňajú tuhé častice (PM), oxid uhoľnatý (CO), ozón (O_3), oxid dusičitý (NO_2) a oxid siričitý (SO_2).

Podľa Európskej environmentálnej agentúry (EEA) by každá z látok znečisťujúcich ovzdušie mohla súvisieť s iným zdrojom/zdrojmi:

Hlavným zdrojom tuhých častíc v roku 2020 bola rezidenčná, obchodná a inštitucionálna spotreba energie. Významnými zdrojmi PM10 bol aj výrobný a ťažobný priemysel a poľnohospodárstvo. V rokoch 2005 až 2020 bol zaznamenaný klesajúci trend emisií tuhých častíc (PM10 a PM2,5) – poklesli o 30 % a 32 %.

- Poľnohospodárstvo bolo hlavným zdrojom amoniaku (94 % celkových emisií) a metánu (56 %) v roku 2020. Emisie amoniaku klesli od roku 2005 do roku 2020 len o 8 %. Išlo o najnižšie percentuálne zníženie všetkých znečisťujúcich látok.
- Hlavným zdrojom oxidov dusíka v roku 2020 bola cestná doprava, ktorá odhalila 37 % emisií. V rokoch 2005 až 2020 bol zistený výrazný pokles emisií oxidov dusíka až o 48 %.
- Hlavným zdrojom oxidu siričitého bol sektor zásobovania energiou, ktorý bol zodpovedný za 41 % emisií v roku 2020. Emisie oxidu siričitého klesli medzi rokmi 2005 a 2020 o 79 %.
- Výrobný a ťažobný priemysel a sektor zásobovania energiou boli hlavnými zdrojmi emisií ťažkých kovov v roku 2020. V rokoch 2005 až 2020 sa najväčšie zníženie emisií zistilo v prípade niklu (64 %) a arzénu (62 %).

D.1 DRUHY ŠKODLIVÍN A ZDRAVOTNÉ RIZIKÁ

Častice

Častica (angl. Particulate Matter, PM) je bežným zástupným indikátorom znečistenia ovzdušia. Hlavnými zložkami frakcií PM sú sírany, dusičnany, amoniak, chlorid sodný, čierne uhlie, minerálny prach a voda.

Zdravotné riziká spojené s časticami s priemerom menším ako 10 a 2,5 mikrometrov (PM10 resp. PM2,5) sú obzvlášť dobre zdokumentované. PM môže preniknúť hlboko do pľúc a vstúpiť do krvného obehu, čo spôsobuje kardiovaskulárne (ischemická choroba srdca), cerebrovaskulárne (mŕtvica) a dýchacie účinky. Dlhodobá aj krátkodobá expozícia PM je spojená s chorobnosťou a úmrtnosťou na kardiovaskulárne a respiračné ochorenia. Dlhodobá expozícia je spojená s nepriaznivými perinatálnymi následkami a rakovinou pľúc.

Oxid uhoľnatý (CO)

Oxid uhoľnatý je bezfarebný toxický plyn bez zápachu a chuti, ktorý vzniká nedokonalým spaľovaním uhlíkatých palív, ako je drevo, benzín, drevené uhlie, zemný plyn a petrolej. Oxid uhoľnatý difunduje cez pľúcne tkanivá a do krvného obehu, čo sťažuje bunkám tela viazať sa na kyslík. Tento nedostatok kyslíka poškodzuje tkanivá a bunky. Vystavenie oxidu uhoľnatému môže spôsobiť ťažkosti s dýchaním, vyčerpanie, závraty a iné príznaky podobné chrípke. Vystavenie vysokým hladinám oxidu uhoľnatého môže byť smrteľné.

Ozón (O_3)

Ozón na úrovni zeme je jednou z hlavných zložiek fotochemického smogu. Prejavuje sa reakciou s plynmi v prítomnosti slnečného žiarenia. Stojí za zmienku, že ozón môžu vytvárať domáce zariadenia, ako sú napríklad prenosné čističe vzduchu. Vystavenie nadmernému ozónu môže spôsobiť problémy s dýchaním, spustiť astmu, znížiť funkciu pľúc a viesť k ochoreniu pľúc.

Oxid dusičitý (NO_2)

NO_2 je plyn, ktorý sa bežne uvoľňuje pri spaľovaní paliva v dopravnom a priemyselnom sektore. Domáce zdroje oxidov dusíka (NOx) zahŕňajú zariadenia, ktoré spaľujú palivá, ako sú pece, krby, plynové kachle a pod. Vystavenie oxidu dusičitému môže dráždiť dýchacie cesty a zhoršiť ochorenia dýchacích ciest.

Oxid siričitý (SO_2)

SO_2 je bezfarebný plyn s ostrým zápachom, ktorý sa vyrába spaľovaním fosílnych palív (uhlie a ropa) a tavením minerálnych rúd obsahujúcich síru. Expozícia SO_2 je spojená s prijatím do nemocnice s astmou a návštevami na pohotovosti.

Polycyklické aromatické uhľovodíky (PAH)

Polycyklické aromatické uhľovodíky (PAH) sú prítomné v atmosfére vo forme častíc. Ide o skupinu chemikálií, ktoré vznikajú predovšetkým nedokonalým spaľovaním organických látok (napr. varenie mäsa) a fosílnych palív v koksovacích peciach, dieselových motoroch a kachliach na drevo. Tiež môžu byť prítomné v tabakovom dyme. Krátkodobá expozícia môže dráždiť oči a dýchacie cesty. Dlhodobá expozícia PAH je spojená s rakovinou pľúc.

D.2 GLOBÁLNE SMERNICE WHO O KVALITE OVZDUŠIA

Od roku 1987 WHO pravidelne vydáva usmernenia o kvalite ovzdušia založené na zdraví, aby pomohla vládam a občianskej spoločnosti znížiť vystavenie ľudí znečisteniu ovzdušia a jeho nepriaznivým účinkom. Hlavným cieľom je ponúknuť kvantitatívne zdravotné odporúčania pre manažment kvality ovzdušia, vyjadrené ako dlhodobé alebo krátkodobé koncentrácie niekoľkých kľúčových látok znečisťujúcich ovzdušie. Prekročenie úrovní smernice o kvalite ovzdušia (AQG) je spojené s významnými rizikami pre verejné zdravie. Tieto usmernenia nie sú právne záväznými normami a nemajú záväzný charakter. Poskytujú však členským štátom WHO nástroj založený na dôkazoch, ktorý môžu implementovať do národných programov na zníženie úrovní látok znečisťujúcich ovzdušie, aby sa znížila obrovská zdravotná záťaž v dôsledku vystavenia znečisteniu ovzdušia na celom svete.

Tabuľka D.1: Odporúčané pokyny pre kvalitu ovzdušia pre každú znečisťujúcu látku [5]

Znečisťujúca látka	Smerná hodnota	Priemerný čas	Odkaz na smernicu
$PM_{2.5}$	5 µg/m^3	rok	WHO 2021
	15 µg/m^3	24-hodín	
PM_{10}	15 µg/m^3	rok	WHO 2021
	45 µg/m^3	24 hodín	
Oxid uhoľnatý (CO)	4 µg/m^3	24 hodín	WHO 2021
Oxid dusičitý (NO_2)	10 µg/m^3	rok	WHO 2021
	25 µg/m^3	24-hodín	
Oxid siričitý (SO_2)	40 µg/m^3	24-hodín	WHO 2021
Formaldehyd	0.1 µg/m^3	30 minút	WHO 2010
Polycyklické aromatické uhľovodíky	8.7×10^{-5} per ng/m^3		WHO 2010
Radón	100 Bq/m^3		WHO 2010
Olovo	0.5 µg/m^3	rok	WHO Regional Office for Europe, 2000

D.3 ŠTÚDIA VPLYVU ZNEČISTENIA OKOLITÉHO OVZDUŠIA NA ZDRAVIE

Za progresívnu zmenu zloženia atmosféry, ktorá negatívne ovplyvňuje kvalitu života, je v minulom storočí zodpovedné zvýšené spaľovanie fosílnych palív a neustále zintenzívňovanie dopravy.

Jedným z hlavných aspektov je vplyv výfukových plynov vozidiel na zdravie, na ktoré sú obzvlášť citlivé deti. Výfukové plyny obsahujú viac ako 200 druhov znečisťujúcich látok, z ktorých niektoré sú: CO_2, NO_x, CO, SO_x, nízkomolekulárne uhľovodíky, aldehydy (formaldehyd, acetaldehyd, akroleín), benzén, 1,3 butadién, polycyklické uhľovodíky, častice oxidačných zložiek (elementárny uhlík, adsorbované aromatické uhľovodíky, malé množstvá síranov, dusičnanov, kovov a iných prvkov) atď.

Hoci znížená ekonomická aktivita počas recesie viedla k zníženiu emisií do ovzdušia, vo všeobecnosti sa má za to, že automobilová doprava v Európe je zodpovedná za škodlivé úrovne látok znečisťujúcich ovzdušie a štvrtinu emisií skleníkových plynov v Európskej únii. Normy "Euro" pre vozidlá dosiahli určitý úspech, ale výrazne neznížili NO_2.

Látky znečisťujúce ovzdušie, ako je oxid uhoľnatý (CO), oxid siričitý (SO_2), oxidy dusíka (NO_x), prchavé organické zlúčeniny (VOC), ozón (O_3), ťažké kovy a tuhé častice (PM2,5 a PM10) sa líšia v ich chemickom zložení, reakčných vlastnostiach, čase rozpadu a schopnosti difúzie na veľké alebo krátke vzdialenosti. Znečistenie vonkajšieho ovzdušia je hlavným environmentálnym zdravotným problémom, ktorý postihuje každého v krajinách s nízkymi, strednými a vysokými príjmami, pretože môže spôsobiť respiračné a iné ochorenia a je dôležitým zdrojom chorobnosti a úmrtnosti [9]. O týchto účinkoch látok znečisťujúcich ovzdušie na ľudské zdravie a ich mechanizme pôsobenia budeme stručne hovoriť ďalej.

Znečistenie ovzdušia má akútne a chronické účinky na ľudské zdravie a ovplyvňuje niekoľko rôznych systémov a orgánov. Siaha od menšieho podráždenia horných dýchacích ciest až po chronické respiračné a srdcové choroby, rakovinu pľúc, akútne respiračné infekcie u detí a chronickú bronchitídu

u dospelých, zhoršujúce už existujúce ochorenie srdca a pľúc alebo astmatické záchvaty. Krátkodobé a dlhodobé expozície sú navyše spojené s predčasnou úmrtnosťou a zníženou strednou dĺžkou života.

WHO odhaduje, že v roku 2019 bolo približne 37 % predčasných úmrtí súvisiacich so znečistením vonkajšieho ovzdušia spôsobených ischemickou chorobou srdca a mozgovou príhodou, 18 % a 23 % úmrtí bolo spôsobených chronickou obštrukčnou chorobou pľúc a akútnymi infekciami dolných dýchacích ciest, v uvedenom poradí, a 11 % úmrtí bolo spôsobených rakovinou v dýchacom trakte. Odhaduje sa, že znečistenie okolitého (vonkajšieho) ovzdušia v mestách aj na vidieku zapríčiňuje od roku 2019 celosvetovo 4,2 milióna predčasných úmrtí ročne; táto úmrtnosť je spôsobená vystavením jemným časticiam, ktoré spôsobujú kardiovaskulárne a respiračné ochorenia a rakovinu.

V celej EÚ je bežné, že úrovne znečistenia ovzdušia sú vyššie ako najnovšie odporúčania WHO. Stále existujú známky zlepšenia, ale niektoré skutočnosti sú uvedené nižšie:

- V roku 2021 bolo 97 % mestského obyvateľstva vystavených koncentráciám jemných častíc, ktoré prekračujú zdravotnú úroveň stanovenú Svetovou zdravotníckou organizáciou.
- Každý rok sa odhaduje, že viac ako 1 200 úmrtí u ľudí mladších ako 18 rokov je spôsobených znečistením ovzdušia v členských a spolupracujúcich krajinách EEA [10].
- Údaje z roku 2021 ukazujú, že stredovýchodná Európa a Taliansko hlásili najvyššie koncentrácie tuhých znečisťujúcich látok, predovšetkým v dôsledku spaľovania tuhých palív na vykurovanie domácností a ich využitia v priemysle.
- Všetky krajiny EÚ hlásili úrovne ozónu a oxidu dusičitého nad úrovňou smerníc pre zdravie, ktoré stanovila Svetová zdravotnícka organizácia.
- Približne 275 000 predčasných úmrtí spôsobia každý rok jemné častice a 64 000 úmrtí oxid dusičitý (NO_2).
- Celkovo bolo 97 % mestského obyvateľstva EÚ vystavených úrovniam jemných častíc, ktoré presahujú najnovšie usmernenia stanovené WHO v roku 2021.

Nepriaznivé účinky vystavenia znečisteniu ovzdušia sú globálnym problémom verejného zdravia v rozvojových aj rozvinutých krajinách, pretože deti a mladí ľudia sú obzvlášť zraniteľní voči účinkom znečistenia ovzdušia.

Epidemiologické štúdie najviac vypovedajú o hodnotení zdravotných účinkov znečistenia ovzdušia. Jednými z najzraniteľnejších, sú deti v predškolskom a mladšom školskom veku, pretože trávia viac času vonku, majú vyššiu intenzitu metabolických procesov a nasávajú relatívne väčší objem vzduchu ako dospelí. Zároveň si ešte neosvojili zlozvyky (fajčenie, konzumácia alkoholu a pod.) a nie sú vystavené priemyselným rizikám. V rozsiahlom komplexe negatívnych zdravotných účinkov výfukových emisií najzreteľnejšie vyčnievajú poruchy dýchacej funkcie, kardiovaskulárneho a imunitného systému, hematopoézy a iné.

Rozsiahla štúdia zahŕňajúca deti v predškolskom a ranom školskom veku v 6 mestách severnej Číny ukázala silnú pozitívnu koreláciu medzi respiračnými príznakmi (kašeľ, ťažkosti s dýchaním, sipot a hlien) a celkovým rozptýleným prachom, oxidom siričitým a hladinami dusíka.

Zvlášť významný je vzťah medzi oxidom dusičitým a ozónom a príčinou alebo exacerbáciou respiračných ochorení s obštrukčným syndrómom, predovšetkým astmou. Typickým príkladom sú prípady, keď aj dočasne znížená intenzita premávky vozidiel znižuje respiračné symptómy horných dýchacích ciest.

Znečistenie ovzdušia ovplyvňuje fyzický a duševný vývoj v detstve a zhoršuje respiračné stavy, ako je astma a sezónna alergická rinitída (SAR), častejšie označovaná ako senná nádcha. Senná nádcha je

najčastejším chronickým ochorením u detí a najčastejšie sa vyskytuje medzi žiakmi škôl. Pribúdajú dôkazy o tom, že látky znečisťujúce ovzdušie, ako je ozón (O_3), môžu zvýšiť alergénnosť peľu, čo môže následne ovplyvniť kognitívny vývoj.

D.4 VPLYV ZNEČISTENIA OVZDUŠIA NA ZDRAVIE A ŽIVOTNÉ PROSTREDIE

Kvalita ovzdušia je pre Európanov hlavným problémom a je oblasťou, v ktorej je EÚ mimoriadne aktívna už viac ako 30 rokov. Hlavným cieľom EÚ v oblasti kvality ovzdušia je „dosiahnuť úroveň kvality ovzdušia, ktorá nebude mať za následok neprijateľné vplyvy a riziká pre ľudské zdravie a životné prostredie". Otázky v ročnom prieskume Flash Eurobarometer sú navrhnuté tak, aby podporili túto prácu tým, že poskytnú lepší prehľad o názoroch európskej verejnosti na kvalitu ovzdušia a znečistenie ovzdušia.

Prieskum Eurobarometer je navrhnutý tak, aby preskúmal:

- úroveň vedomostí o problémoch kvality ovzdušia;
- vnímanú závažnosť problémov s kvalitou ovzdušia a vnímané zmeny v kvalite ovzdušia za posledných desať rokov;
- vnímaný vplyv rôznych sektorov a činností na kvalitu ovzdušia;
- hlavné hrozby pre kvalitu ovzdušia;
- ekologické možnosti energetiky a dopravy;
- individuálne a iné opatrenia na zníženie problémov s kvalitou ovzdušia;
- a veľa ďalších.

Výsledky prieskumu z roku 2022 ukazujú, že kvalita ovzdušia je pre európskych občanov stále vážnym problémom. Všetky nespracované údaje z prieskumu sú voľne dostupné a sú dostupné online:

- Zatiaľ čo väčšina Európanov sa necíti dobre informovaná (60 %), takmer polovica respondentov sa domnieva, že kvalita ovzdušia sa za posledných desať rokov zhoršila (47 %).
- Väčšina Európanov si myslí, že zdravotné problémy, ako sú choroby dýchacích ciest (89 %), astma (88 %) a kardiovaskulárne choroby, sú v ich krajinách vážnymi problémami, spôsobenými znečistením ovzdušia. Eurobarometer odhaľuje, že občanom chýbajú informácie o problémoch s kvalitou ovzdušia v ich krajine.
- Väčšina Európanov je stále nedostatočne informovaná o existujúcich normách EÚ pre kvalitu ovzdušia, keďže o nich počula len menšina respondentov (27 %).
- Napriek tomu veľká väčšina respondentov (67 %), ktorí sú si vedomí noriem EÚ v oblasti kvality ovzdušia, tvrdí, že by sa mali posilniť.

Skríningový dotazník na hodnotenie vnímania znečistenia ovzdušia a rizika vystavenia vonkajšiemu a vnútornému znečisteniu ovzdušia

Na vypracovanie dotazníka sa, okrem mechanizmov z podobných štúdií o ochrane pred znečistením ovzdušia, použil aj súbor položiek založený na mnohých štandardizovaných odporúčaniach prieskumu. Položky boli starostlivo vytvorené tak, aby sa minimalizovali nejednoznačnosti a zvýšila sa zrozumiteľnosť. Celkovo pozostával z 25 položiek. Dotazník je vhodným nástrojom na hodnotenie postojov a vnímania obyvateľstva k znečisteniu ovzdušia a riziku vystavenia vonkajšiemu a vnútornému znečisteniu. Tento dotazník by mohli použiť vedci, výskumníci, úrady a plánovači podpory zdravia na rozvoj a implementáciu programov na podporu ochrany pred znečistením ovzdušia.

Dotazník A – hlavné položky

Prosím, prečítajte si všetky otázky a odpovedzte zaškrtnutím políčka alebo poskytnutím krátkeho vysvetlenia tam, kde je to vhodné.

Prieskum je anonymný a uisťujeme vás, že dôvernosť vašich individuálnych odpovedí bude zachovaná.

1. Pohlavie

☐ muž ☐ žena

2. Vek

☐ menej ako 3 roky ☐ 3-7 rokov ☐ 8-14 rokov

☐ 15-20 rokov ☐ 21-30 rokov ☐ 31-40 rokov

☐ 41-50 rokov ☐ 51-60 rokov ☐ viac ako 60 rokov

3. V akej oblasti žijete?

Krajina..... Sídlo .....

4. Povedali by ste, že bývate …

☐ na vidieku ☐ na dedine ☐ v malom meste

☐ v stredne veľkom meste ☐ vo veľkom meste/city

5. Zamestnanie

☐ žiak ☐ študent ☐ živnostník ☐ zamestnanec ☐ robotník ☐ bez profesionálnej aktivity ☐ dôchodca ☐ iné

Uveďte prosím.

6. Koľko ľudí 15- a viacročných žije vo vašej domácnosti, vrátane vás?

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ iné .... ☐ *Uveďte prosím.*

7. Aký je mesačný príjem vašej rodiny (na člena)?

☐ do 300 EUR ☐ 300-600 EUR ☐ 600-1000 EUR ☐ 1000-1500 EUR

☐ viac ako 1500 EUR ☐ iné ..... ☐ *Prosím špecifikujte.*

8. Aký máte typ kúrenia v domácnosti?

☐ elektrický ohrievač ☐ plynový ohrievač ☐ klimatizácia ☐ pec

☐ kúrenie drevom/peletami ☐ solárne kúrenie ☐ iné .....

Uveďte prosím.

9. Ako informovaný/á o problémoch kvality ovzdušia sa cítite vo vašej krajine?

☐ veľmi dobre informovaný/á ☐ dobre informovaný/á ☐ informovaný/á

☐ vôbec nie informovaný/á ☐ iné ..... *Prosím špecifikujte.*

10. Myslíte si, že za ostatných 10 rokov kvalita ovzdušia vo vašej krajine sa …?

☐ zlepšila ☐ zostáva rovnaká ☐ zhoršila

☐ iné ..... *Uveďte prosím.*

11. Aký veľký vplyv má, podľa vás, každá z nasledujúcich možností na kvalitu ovzdušia vo vašej krajine? Má veľký, mierny, malý alebo žiadny vplyv?

	Veľký vplyv	Mierny vplyv	Malý vplyv	Žiadny vplyv
Spotreba energie na bývanie (napr. uhlie a drevo na vykurovanie jednotlivých domácností)				
Poľnohospodárske emisie z fariem, hnojivá a spaľovanie poľnohospodárskeho odpadu				
Emisie z osobných a nákladných automobilov				
Emisie z medzinárodnej dopravy (napr. lode a lietadlá)				
Emisie z priemyselnej výroby (oceľ, cement, celulóza, papier atď.) a z elektrární na fosílne palivá				
Krajina				
Rieky / jazerá				
Čistý vzduch				
Iné				

12. Ktoré tri z nasledujúcich znečisťovateľov sú podľa vás hlavnými hrozbami pre kvalitu ovzdušia vo vašej krajine?

☐ cezhraničné emisie z iných krajín/regiónov
☐ prepravné činnosti
☐ výroba elektriny a tepla
☐ prírodné znečisťujúce látky (morská soľ, púštny piesok, sopečný popol)
☐ priemyselné činnosti
☐ emisie z jednotlivých domácností
☐ emisie z fariem
☐ iné *Uveďte prosím.*

13. Ktoré dva z nasledujúcich palivových systémov automobilov považujete z hľadiska kvality ovzdušia za najekologickejšie?

☐ benzín ☐ nafta ☐ biopalivo
☐ hybridné elektrické/benzínové autá ☐ hybridné elektrické/dieselové autá
☐ elektrické autá ☐ iné ..... *Uveďte prosím.*

14. Ktoré dva z nasledujúcich energetických systémov na vykurovanie domácností považujete z hľadiska kvality ovzdušia za najekologickejšie?

☐ ropa ☐ plyn ☐ uhlie ☐ biomasa (drevo)
☐ biomasa (pelety) ☐ elektrina ☐ diaľkové vykurovanie
☐ iné..... *Uveďte prosím..*

15. Existujú rôzne spôsoby, ako znížiť škodlivé emisie do ovzdušia. Urobili ste za posledné dva roky niečo z nasledujúceho, aby ste znížili tieto problémy? Vyberte všetky použiteľné.

☐ Zmenili ste svoj systém vykurovania bytov z vyšších emisií (napr. uhlie, olej alebo drevo) na nízkoemisné (napr. zemný plyn, pelety, elektrina)

☐ Vymenili ste staršie zariadenia využívajúce energiu (teplovodný bojler, rúru, umývačku riadu atď.) za novšie zariadenia s lepšou energetickou účinnosťou (napr. A+++ pre energetickú účinnosť)

☐ Často ste namiesto auta používali verejnú dopravu, jazdu na bicykli alebo chôdzu

☐ Kúpili ste si auto s nízkymi emisiami

☐ Kúpili ste si produkty s nízkymi emisiami, aby ste podporili svoj otvorený oheň alebo grilovanie (t. j. brikety namiesto uhlia)

☐ iné..... *Uveďte prosím.*

16. Povedali by ste, že nasledovné je veľmi vážny problém, dosť vážny problém, nie veľmi vážny problém, alebo nie vážny problém vo vašej krajine?

	Veľmi vážny problém	Pomerne vážny problém	Nie veľmi vážny problém	Nejde o vážny problém
Ochorenia dýchacích ciest (napríklad ochorenia pľúc)				
Kardiovaskulárne ochorenia (ochorenia srdca)				
Astma a alergia				
Acidifikácia (kyslé dažde, ovplyvňujúce lesy atď.)				
Eutrofizácia (nárast organickej hmoty v ekosystéme, ako je nadmerný rast rias spôsobujúci úhyn rýb v riekach alebo jazerách)				

17. Robí podľa vás každý z nasledujúcich priveľa, robí správne množstvo alebo nerobí dosť na podporu dobrej kvality ovzdušia vo vašej krajine?

	Robí príliš veľa	Robí správne množstvo	Nerobí dosť	Neviem
Domácnosti				
Farmári				
Výrobcovia energie				
Výrobcovia áut				
Orgány verejnej moci				

21. Do akej miery vás ovplyvňuje znečistenie ovzdušia?

☐ Dýchavičnosť/väčšie ťažkosti s dýchaním
☐ Menej aktivít vonku
☐ Robiť viac pre starostlivosť o moju pleť
☐ Robíte viac pre to, aby ste zostali zdraví
☐ Pocit depresie
☐ Podráždenie očí/nosa/hrdla
☐ Kožné problémy
☐ Chcete sa presťahovať na iné menej znečistené miesta
☐ Výskyt astmy
☐ Zlá viditeľnosť
☐ Obavy o životné prostredie
☐ iné..... *Uveďte prosím.*

22. Váš dom sa nachádza…?

☐ v tichej oblasti, nízka automobilová premávka
☐ v hlučnej oblasti, hustá automobilová doprava
☐ v hlučnej oblasti z dôvodu odlišného od zdroja dopravy
☐ iné..... *Uveďte prosím.*

23. Cítite u vás doma nejaké výfukové plyny z automobilovej dopravy?

☐ áno, každý deň ☐ áno, často ☐ zriedkavé ☐ iné..... *Uveďte prosím.*

24. Do akej miery pociťujete zriedkavo z vozidiel (hluk, výfukové plyny atď.) vo vašej domácnosti?

☐ veľmi vysoká ☐ stredná ☐ nízka ☐ iná..... *Uveďte prosím.*

25. Má vaša rodina problémy so spánkom v noci (prebúdzanie sa kvôli hluku z dopravy)?

☐ áno, veľmi často ☐ často ☐ zriedkavo ☐ iné..... *Uveďte prosím.*

Časť B – špeciálna časť o zdraví detí

Prosím, prečítajte si všetky otázky a odpovedzte zaškrtnutím políčka alebo poskytnutím krátkeho vysvetlenia tam, kde je to vhodné.

Prieskum je anonymný a uisťujeme vás, že dôvernosť vašich individuálnych odpovedí bude zachovaná.

1. Pohlavie

☐ muž ☐ žena

2. Vek

☐ menej ako 3 roky ☐ 3-7 rokov ☐ 8-15 rokov ☐ viac ako 15 rokov

3. Váha dieťaťa

☐ pri narodení …… …….. ☐ v súčasnosti …….

4. Vek matky pri narodení dieťaťa

☐ do 20 rokov ☐ 21-30 rokov ☐ 31-40 rokov
☐ 41-50 rokov ☐ viac ako 50 rokov

5. Ako dlho bolo dieťa dojčené (v mesiacoch)?

☐ do1 mesiaca ☐ 1-3 mesiacov ☐ 3-6 mesiacov

☐ 6-9 mesiacov ☐ 9-12 mesiacov ☐ iné ..... *Uveďte prosím.*

6. Sú vo vašej rodine fajčiari cigariet? Koľko?

☐ 1 ☐ 2 ☐ 3 ☐ iné ..... *Uveďte prosím či aj mama je fajčiarka.*

7. Máte doma domáce zvieratá? Koľko?

☐ áno ☐ nie ☐ iné..... *Uveďte prosím.*

8. Majú niektorí rodičia alebo bratia/sestry alergické ochorenie?

☐ áno ☐ nie ☐ iné..... *Uveďte prosím.*

9. Má vaše dieťa (respondent) alergické ochorenie?

☐ áno ☐ nie ☐ iné..... *Uveďte prosím.*

11. Trpí vaše dieťa ochoreniami dýchacích ciest (nádcha, bronchitída, zápal pľúc) častejšie ako štyrikrát do roka?

☐ áno ☐ nie ☐ iné..... *Uveďte prosím.*

12. Zaregistrovali ste u svojho dieťaťa (respondenta) počas posledných šiestich mesiacov niektoré z nasledujúcich príznakov?

	áno	nie	neviem
pretrvávajúci kašeľ			
Pískanie, sipenie			
suchý kašeľ v noci			
senná nádcha			
záchvaty ťažkostí s dýchaním (astma)			
chrípka alebo iné ochorenie postihujúce dýchací systém			

PRÍLOHA E KURIKULUM

Táto časť učebnice predstavuje kurikulum pre kurz „Pokročilé technológie spracovania veľkých dát". Tento učebný plán poskytol Fakulty prírodných vied Univerzity Mateja Bela v Banskej Bystrici. Tento partner realizoval kurz a všetci partneri ho budú realizovať počas projektu.

Univerzita: Univerzita Mateja Bela v Banskej Bystrici, Slovensko
Fakulta: Fakulta prírodných vied
Kód: DEK FPV/2d-fpv-401
Názov kurzu: Pokročilé technológie spracovania veľkých dát
Typ, záťaž a metóda výučby: Typ kurzu: výberový Odporúčaná záťaž: 2 hodiny seminárov/týždeň Metóda štúdia: kombinovaná Forma štúdia: denná Počet kreditov: 3 Odporúčaný semester: Druhý semester magisterského štúdia
Stupeň štúdia: druhý (magister)
Prerekvizity predmetu: žiadne
Podmienky absolvovania kurzu: a) priebežné hodnotenie: aktívna účasť na cvičeniach, plnenie úloh počas semestra 100 % b) záverečné hodnotenie: 0 % Hodnotenie predmetu je v súlade s klasifikačnou stupnicou určenou študijným poriadkom UMB.
Vzdelávacie výstupy: 1. Študenti nadobudnú zručnosti, vedomosti a skúsenosti v oblastiach: 2. Úvod do spracovania a analýzy dát 3. Úvod do základných úloh dátovej analýzy – regresia a klasifikácia 4. Úvod do práce s Veľkými dátamai – metódy vzorkovania 5. Štatistické metódy analýzy dát 6. Úvod do Exploratívnej analýzy dát – teória a prax 7. Úvod do fuzzy množín 8. Fuzzy množiny a regresná úloha 9. Fuzzy množiny a klasifikačná úloha 10. Úvod do neurónových sietí 11. V rámci kurzu študent nadobudne skúsenosti v používaní softvéru: 12. MATLAB 13. R

ZOZNAM BIBLIOGRFICKÝCH ODKAZOV

References for sections 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 Hal-l/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.

References for sections 5 – 8:

- ZADEH, L. A. Fuzzy Sets. In: Information and Control, 8, 1965, 338-353.
- MICHALÍKOVÁ, A.: Fuzzy množiny v informatike. rec. Mirko Navara, Martin Kalina, Martin Klimo. Belianum. Matej Bel University in Banská Bystrica, 1, 2020, 206p. ISBN 978-80-557-1707-4
- Sendai Subway. Japan Visitor [cit. 2023-02-02]. Online: https://www.japanvisitor.com/japan-transport/sendai-subway
- RUAN D.: Fuzzy Logic Applications in Nuclear Industry. Fuzzy Logic Foundations and Industrial Applications. 1996, 8, ISBN 978-1-4612-8627-1.
- 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.
- ROSS, T. J. Fuzzy Logic with Engineering Applications. John Wiley & Sons, 2005, 585s., ISBN 9780470743768.
- ZADEH, L. A., The Concept of a Linguistic Variable and its Application to Approximate Reasoning - 1, In: Information Sciences, 8, 1975, 199–249.

References for sections 9

- Ahmed, Z. H. (2010). Genetic algorithm for the traveling salesman problem using sequential constructive crossover operator. International Journal of Biometrics & Bioinformatics (IJBB), 3(6), 96.
- Aktaş, M., Yetgin, Z., Kılıç, F., & Sünbül, Ö. (2022). Automated test design using swarm and evolutionary intelligence algorithms. Expert Systems, 39(4), e12918.
- Bartz-Beielstein, T., Branke, J., Mehnen, J., & Mersmann, O. (2014). Evolutionary algorithms. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 4(3), 178-195.
- Bertsimas, D., & Tsitsiklis, J. (1993). Simulated annealing. Statistical science, 8(1), 10-15.
- Blickle, T. (2000). Tournament selection. Evolutionary computation, 1, 181-186.
- Cui, Y., Geng, Z., Zhu, Q., & Han, Y. (2017). Multi-objective optimization methods and application in energy saving. Energy, 125, 681-704.
- De La Iglesia, B. (2013). Evolutionary computation for feature selection in classification problems. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 3(6), 381-407.
- Gaivoronski, A. A., Lisser, A., Lopez, R., & Xu, H. (2011). Knapsack problem with probability constraints. Journal of Global Optimization, 49, 397-413.
- Glover, F., & Laguna, M. (1998). Tabu search (pp. 2093-2229). Springer US.
- Hansen P, Mladenovi'c N (1999) An introduction to variable neighborhood search. In: Voß S, Martello S, Osman IH, Roucairol C (eds) Metaheuristics: advances and trends in local search paradigms for optimization, chapter 30. Kluwer Academic Publishers, Dordrecht, pp 433–458
- Hayyolalam, V., & Kazem, A. A. P. (2020). Black widow optimization algorithm: a novel meta-heuristic approach for solving engineering optimization problems. Engineering Applications of Artificial Intelligence, 87, 103249.

- Hinson, J. M., & Staddon, J. E. R. (1983). Matching, maximizing, and hill-climbing. Journal of the experimental analysis of behavior, 40(3), 321-331.
- Holland JH. Outline for a logical theory of adaptive systems. J ACM. 1962;9(3):297–314
- Holland, J. H. (1973). Genetic algorithms and the optimal allocation of trials. SIAM journal on computing, 2(2), 88-105.
- Hoos, H. H., & Stützle, T. (2004). Stochastic local search: Foundations and applications. Elsevier.
- I. Rechenberg, Cybernetic solution path of an experimental problem. Royal Air-craft Establishment, Library Translation 1122, Farnborough, Reprint in: D.B. Fogel (Ed.), Evolutionary Computation, The Fossil Record, IEEE Press, Piscataway, NJ, 1965, pp. 301–309
- I. Rechenberg, Evolutionsstrategie—Optimisierung technischer Systeme nach Prinzipien der biologischen Evolution, Frommann-Holzboog, Stuttgart, 1973
- Kılıç, F., Yılmaz, İ. H., & Kaya, Ö. (2021). Adaptive co-optimization of artificial neural networks using evolutionary algorithm for global radiation forecasting. Renewable Energy, 171, 176-190.
- Kiliç, F., & Gök, M. (2013). A public transit network route generation algorithm. IFAC Proceedings Volumes, 46(25), 162-166.
- Li, X., Tang, K., Omidvar, M. N., Yang, Z., Qin, K., & China, H. (2013). Benchmark functions for the CEC 2013 special session and competition on large-scale global optimization. gene, 7(33), 8.
- Mirjalili, S. (2016). SCA: a sine cosine algorithm for solving optimization problems. Knowledge-based systems, 96, 120-133.
- Rossi, F., Van Beek, P., & Walsh, T. (Eds.). (2006). Handbook of constraint programming. Elsevier.
- Salkin, H. M., & De Kluyver, C. A. (1975). The knapsack problem: a survey. Naval Research Logistics Quarterly, 22(1), 127-144.
- Sharifi, A. A., & Aghdam, M. H. (2019). A novel hybrid genetic algorithm to reduce the peak-to-average power ratio of OFDM signals. Computers & Electrical Engineering, 80, 106498.
- Wang, L., Cao, Q., Zhang, Z., Mirjalili, S., & Zhao, W. (2022). Artificial rabbits optimization: A new bio-inspired meta-heuristic algorithm for solving engineering optimization problems. Engineering Applications of Artificial Intelligence, 114, 105082.
- Yang, J., & Soh, C. K. (1997). Structural optimization by genetic algorithms with tournament selection. Journal of computing in civil engineering, 11(3), 195-200.

References for section 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/

References for section 11:

- 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.
- 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.
- Get Started with Matlab. https://www.mathworks.com/help/matlab/getting-started-with-matlab.html
- Iris Clustering. https://www.mathworks.com/help/deeplearning/ug/iris-clustering.html

References for Appendices:

- 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 Recogni-

tion 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