Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Evidenčné číslo: FIIT-000-00000

Matúš Cuper

Optimalizácia konfiguračných parametrov predikčných metód

Bakalárska práca

Vedúci práce: Ing. Marek Lóderer

máj 2017

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Evidenčné číslo: FIIT-000-00000

Matúš Cuper

Optimalizácia konfiguračných parametrov predikčných metód

Bakalárska práca

Študijný program: Informatika
Študijný odbor: 9.2.1 Informatika
Miesto vypracovania: Ústav informatiky a softvérového inžinierstva, FIIT STU Bratislave
Vedúci práce: Ing. Marek Lóderer

máj 2017

Slovenská technická univerzita v Bratislave
FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ

Anotácia

Slovenská technická univerzita v Bratislave
FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ
Študijný program: Informatika
Autor: Matúš Cuper
Bakalárska práca: Optimalizácia konfiguračných parametrov predikčných metód
Vedúci práce: Ing. Marek Lóderer
máj 2017

V práci sme sa zamerali na problémy vznikajúce pri prekcii časových radov. V súčasnosti existuje veľké množstvo metód, ktoré nám zabezpečujú predpoveď sledovanej veličiny s prijateľne malou odchýlkou na krátke obdobie v blízkej budúcnosti. Cieľom bakalárskej práce bolo vytvoriť systém, ktorý používateľovi poskytne jednoduché rozhranie pre porovnanie jednotlivých predikčných algoritmov nad množinou dát, ktorú si sám zvolí. Hľadanie ich optimálneho nastavenia sa vykonáva pomocou optimalizačných algoritmov založených na správaní sa živočíchov v prírode.

V práci sme analyzovali a opísali množinu predikčných a optimalizačných algoritmov. Navrhli sme systém na hľadanie optimálnych parametrov predikčných metód, čím sme výrazne ovplyvnili ich presnosť. Systém bol implementovaný v programovacom jazyku R a na vytvorenie používateľského rozhrania bola použitá knižnica Shiny. Optimalizácie sme vykonávali nad dátovými množinami v doméne energetiky. Výsledný systém umožňuje používateľovi využívať silu predikčných algoritmov a nájsť ich optimálne parametre pre zabezpečenie čo najpresnejšej predikcie.

Slovak University of Technology Bratislava
FACULTY OF INFORMATICS AND INFORMATION TECHNOLOGIES

Annotation

Slovak University of Technology Bratislava
FACULTY OF INFORMATICS AND INFORMATION TECHNOLOGIES
Degree Course: Computer Science
Author: Matúš Cuper
Bachelor thesis: Optimizing configuration parameters of prediction methods
Supervisor: Ing. Marek Lóderer
May 2017

In the thesis we focused on problems, which appear in time series prediction. In present there are many methods, which predict observed value with acceptable small deviation for short time period in near future. The aim of bachelor thesis was create system, which provides simple user interaface to compare chosen prediction algorithms on dataset, which is chosen by user. Looking for their optimal setup is made by optimization algorithm based on nature-inspired behaviour.

In the thesis we analyzed and described set of prediction and optimization algorithms. We designed system for searching optimal parameters of prediction methods, which influence their accuracy significantly. System was implemented in programming language R and for creating user interaface was used library Shiny. Optimization was provided on dataset in energetics domain. The final system provides to user to use force of prediction algorithms and find out ther optimal parameters for the most accurate prediction.

ČESTNÉ PREHLÁSENIE

Čestne prehlasujem, že bakalársku prácu som vypracoval samostatne pod vedením vedúceho bakalárskej práce a s použitím odbornej literatúry, ktorá je uvedená v zozname použitej literatúry.

.....
Matúš Cuper

POĎAKOVANIE

Ďakujem vedúcemu bakalárskej práce Ing. Marekovi Lódererovi za odborné vedenie, cenné rady a pripomienky pri spracovaní bakalárskej práce.

Obsah

1 Úvod **1**

2 Analýza problému **2**
2.1 Časové rady . . . 2
2.1.1 Analýza časových radov . . . 2
2.1.2 Zložky časových radov . . . 3
2.2 Analýza predičkných algoritmov . . . 6
2.2.1 Lineárna regresia . . . 6
2.2.2 Stochastické modely . . . 7
2.2.3 Regresia založená na podporných vektoroch . . . 8
2.2.4 Rozhodovacie stromy . . . 9
2.2.5 Náhodné lesy . . . 9
2.2.6 Neurónové siete . . . 10
2.2.7 Učenie súborov klasifikátorov . . . 12
2.2.8 Exponencionálne vyrovnávanie . . . 12
2.3 Analýza optimalizačných algoritmov . . . 12
2.3.1 Genetický algoritmus . . . 13
2.3.2 Optimalizácia svorkou divých vlkov . . . 15
2.3.3 Umelá kolónia včiel . . . 18
2.3.4 Kolónia mravcov . . . 19
2.4 Meranie presnosti predpovedi . . . 19
2.4.1 Stredná chyba predpovede . . . 19
2.4.2 Stredná absolútna chyba . . . 20
2.4.3 Stredná percentuálna chyba . . . 20
2.4.4 Stredná aboslútna percentuálna chyba . . . 20
2.4.5 Stredná štvorcová chyba . . . 20
2.5 Zhodnotenie analýzy . . . 21

3 Špecifikácia požiadaviek **22**

4 Návrh riešenia **23**

5 Implementácia **24**

6 Zhodnotenie **25**

7 Záver **26**

8 Technická dokumentácia **27**

9 Plán do nasledujúceho semestra **28**

Literatúra **29**

Zoznam obrázkov

1 Príklad trendovej zložky časového radu. 3
2 Príklad sezónnej zložky časového radu. 4
3 Príklad reziduálnej zložky časového radu. 4
4 Príklad multiplikatívneho modelu 5
5 Príklad aditívneho modelu. 6
6 Príklad neurónu v neurónovej sieti. 10
7 Príklad viacvrstvovej spätne propagovanej neurónovej siete [12]. 11
8 Hierarchia vlkov vo svorke. 15
9 Príklad možného umiestnenia vlka v dvojrozmernom priestore na základe polohy koristi. 17

Zoznam tabuliek

1 Tabuľka rozdelenia vybraných biologicky inšpirovaných algoritmov [7]. . . 13
2 Proces vytvorenia novej generácie z rodičovských chromozónov. 14
3 Čo ešte určite musím stihnúť VYMAZAŤ (podľa dôležitosti) 28

1 Úvod

V súčasnosti sme obklopený množstvom zariadení, ktoré merajú a zhromažďujú informácie z iných zariadení. Príkladom môžu byť rôzne mobilné aplikácie, meteorologické stanice alebo chytré merače merajúce spotrebu elektriky, vody či plynu. Namerané dáta sa môžu meniť v závislosti od napr. hodiny, dňa alebo počasia. Samozrejme existujú aj merania, ktoré sú ovplyvnené ľudským správaním, ktoré sa môže líšiť v závislosti od vyššie uvedených faktorov, ale aj faktorov ako sú kultúrne tradície, zvyky či náboženstvo. Na základe nameraných dát vieme vytvoriť predpoveď, ktorá opäť môže ovplyvniť ostatné faktory a celé predpovedanie sa tak stáva opäť komplexnejším. Ak sú dáta merané v pravidelných intervaloch a konzistentne, nazývame ich časový rad.

V práci sme sa zamerali na predpovedanie veličín na základe ich historických meraní. Ostatné faktory pri tom neboli brané do úvahy. Tým sa stáva predpovedanie jednoduchšie a menej presné, čím vzniká priestor pre hlavný zámer práce, optimalizovanie predikčných metód na základe ich vstupných parametrov. Získame tým presnejšiu predpoveď ako keby sme nastavenie predikčných algoritmov nechali na náhode alebo vlastnom úsudku.

Optimalizačné algortimy, tak ako ich názov napovedá, slúžia na nájdenie optimálneho riešenia. Nájdenie najlepšieho riešenia požaduje preskúmanie všetkých možností. Stáva sa tak pomalým a výpočtovo náročným. Optimalizačné algoritmy rýchlo nájdu riešenie, ktoré je pre potreby našej práce postačujúce. Optimalizačné algoritmy môžeme rozdeliť do viacerých skupín, v práci sa však zameriavame prednostne na biologicky inšpirované algoritmy, ktoré sú jednou z najefektívnejších podskupín.

Výsledný systém poskytuje používateľovi webové grafické rozhranie, pomocou ktorého môže predpovedať hodnoty vložených časových radov. Má možnosť zvoliť si medzi viacerými predikčnými a optimalizačnými algoritmami. Rozhranie poskytuje používateľovi základné informácie o algoritmoch, ako aj vysvetlenie efektov jednotlivých vstupných parametrov metód. Je na zvážení používateľa, aké parametre zvolí pre optimalizačné algoritmy. Vstupné parametre predikčných metód budú zvolené optimálne na základe výstupných hodnôt optimalizačných algoritmov. Používateľ má k dispozícií vyhodnotenie predpovede, ktoré porovnáva predpovedanú a skutočnú hodnotu rôznymi metodikami.

Celá práca je rozdelená do niekoľkých kapitol. V kapitole 2 sme sa zamerali na analýzu problému. Definovali sme kľúčové pojmy, použité metódy, opísali sme časové rady, ich vlastnosti, rozdelili sme predikčné a optimalizačné algoritmy do skupín. Kapitola 3 popisuje funkcionalitu, ktorú systém poskytuje používateľom. V kapitole 4 sa zameriavame na návrh systému a v kapitole 5 už samotnou implemntáciou systému v jazyku R. Výsledky práce sú zhodnotené v kapitole 6.

2 Analýza problému

Predpovedanie spotreby elektriky je kľúčovou činnosťou pre plánovanie a prevádzkovanie rôznych elektronických zariadení. Hľadaný vzor pre časové rady spotreby elektriny je často komplexný a je zložité ho nájsť aj kvôli faktorom ako sú napr. zmeny cien elektriky na trhu. Preto sa stáva implementácia vhodného modelu zaručujúceho presnú predpoveď náročnou [16].

Práve preto vznikajú rôzne metódy na predpovedanie časových radov, ktoré sú bližšie popísané v nasledujúcich podkapitolách. Väčšina z nich poskytuje niekoľko vstupných parametrov ako rozhranie pre vnútorné nastavenie algortimov. Vďaka tomu máme možnosť ovplyvňovať mieru natrénovania modelu, veľkosť chyby predikcie alebo dĺžku obdobia, na ktorom budeme model trénovať. Nastavenie týchto parametrov sa môže pri rôznych datasetoch líšiť, a preto neexistuje univerzálne riešenie. Hľadať riešenie pomocou biologicky inšpirovaných algoritmov je efektívne a nájdené riešenie je optimálne. Ďalej sú v tejto kapitole opísané spôsoby merania chýb, ktoré slúžia na vyhodnotenie efektivity a správnosti nájdeného riešenia.

2.1 Časové rady

Časový rad je množina dátových bodov nameraná v čase postupne za sebou. Matematicky je definovaný ako množina vektorov $x(t)$, kde t reprezentuje uplynulý čas. Premenná $x(t)$ je považovaná za náhodnú premennú. Merania v časových radoch sú usporiadané v chronologickom poradí [1].

Časové rady delíme na spojité a diskrétne. Pozorovania pri spojitých časových radoch sú merané v každej jednotke času, zatiaľ čo diskrétne obsahujú iba pozorovania v diskrétnych časových bodoch. Hodnoty toku rieky, teploty či koncentrácie látok pri chemickom procese môžu byť zaznamenané ako spojitý časový rad. Naopak, populácia mesta, produkcia spoločnosti alebo kurzy mien reprezentujú diskrétny časový rad. Vtedy sú pozorovania oddelené rovnakými časovými intervalmi, napr. rokom, mesiacom či dňom [1]. V našom prípade sú namerané dáta dostupné každú celú štvrťhodinu.

2.1.1 Analýza časových radov

V praxi je vhodný model napasovaný do daného časového radu a zodpovedajúce parametre sú predpovedané na základe známych dát. Pri predopovedaní časových radov sú dáta z predchádzajúcich meraní zhromaždované a analyzované za účelom navrhnutia vhodného matematického modelu, ktorý zachytáva proces generovania dát pre časové rady. Pomocou tohto modelu sú predpovedané hodnoty budúcich meraní. Takýto prístup je užitočný, keď nemáme veľa poznatkov o vzore v meraniach idúcich za sebou alebo máme model, ktorý poskytuje nedostatočne uspokojivé výsledky [1].

Cieľom predikcií časových radov je predpovedať hodnotu premennej v budúcnosti na základe doteraz nameraných dátových vzoriek. Matematicky zapísané ako

$$\hat{x}(t + \Delta_t) = f(x(t - a), x(t - b), x(t - c), ...) \tag{1}$$

Hodnota $\hat{x}$ vo vzorci 1 je predpovedaná hodnota jednorozmerného diskrétneho časového radu x. Úlohou je nájsť takú funkciu $f(x)$, pre ktorú bude $\hat{x}$ predstavovať predpovedanú hodnotu časového radu. Táto funkcia by mala predpovedať hodnoty v budúcnosti konzistentne a objektvíne [19].

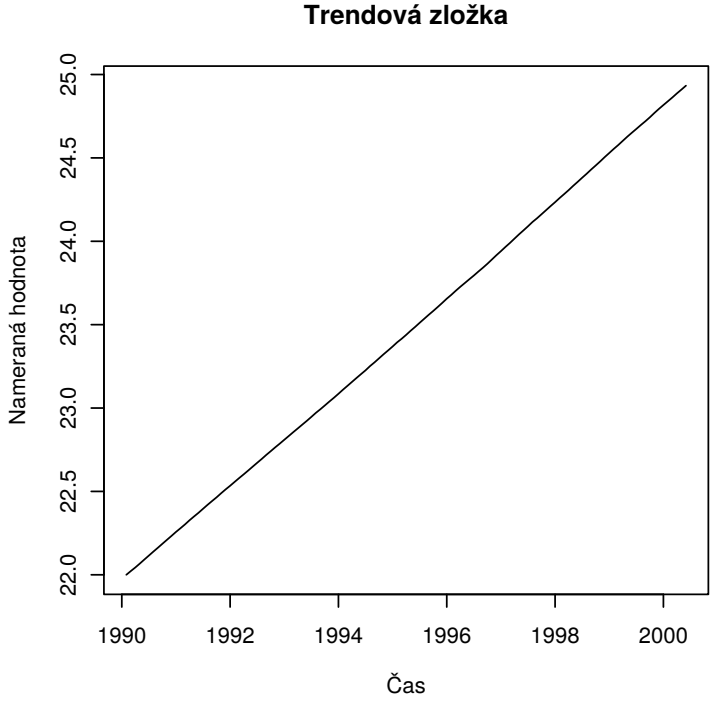
Časové rady sú najčastejšie vizualizované ako graf, kde pozorovania sú na osy y a plynúci čas na osy x. Pre lepšie vysvetlenie časových radov, budú nasledujúce odstavce obsahovať aj obrázky zobrazujúce vygenerovanú dátovú množinu.

2.1.2 Zložky časových radov

Pri predpovedaní časových radov ako napr. meraní odberu elektriky vznikajú 2 typy trendov. Prvým typom je trvalá alebo dočasná zmena spôsobená ekonomickými alebo ekologickými faktormi. Druhým typom je sezónna zmena, spôsobená zmenami ročných období a množstvom denného svetla. Môžeme ju pozorovať na úrovni dňov, týždňov alebo rokov. Veličina, ktorú sa snažíme predpovedať postupne mení svoje správanie a model sa tak stáva nepresným. Kvôli tomu je nutné v každom modely rozdeľovať tieto typy tendencií, aby sme vedeli model zmenám prispôsobiť [8].

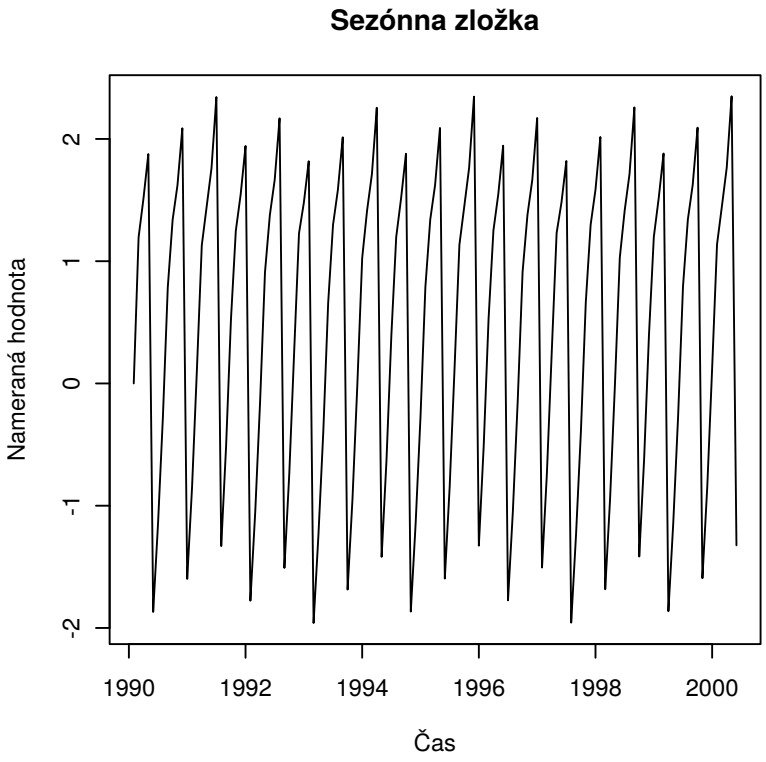
Vo všeobecnosti sú časové rady zložené zo 4 hlavných zložiek, ktoré môžeme odlíšiť od pozorovaných dát. Jedná sa o trendovú, cyklickú, sezónnu a reziduálnu zložku [1].

Trendová zložka predstavuje správanie časového radu v dlhodobom časovom horizonte. Z tohto pohľadu má časový rad tendenciu klesať, rásť alebo stagnovať. Príkladom môže byť nárast populácie či klesajúca úmrtnosť [1].

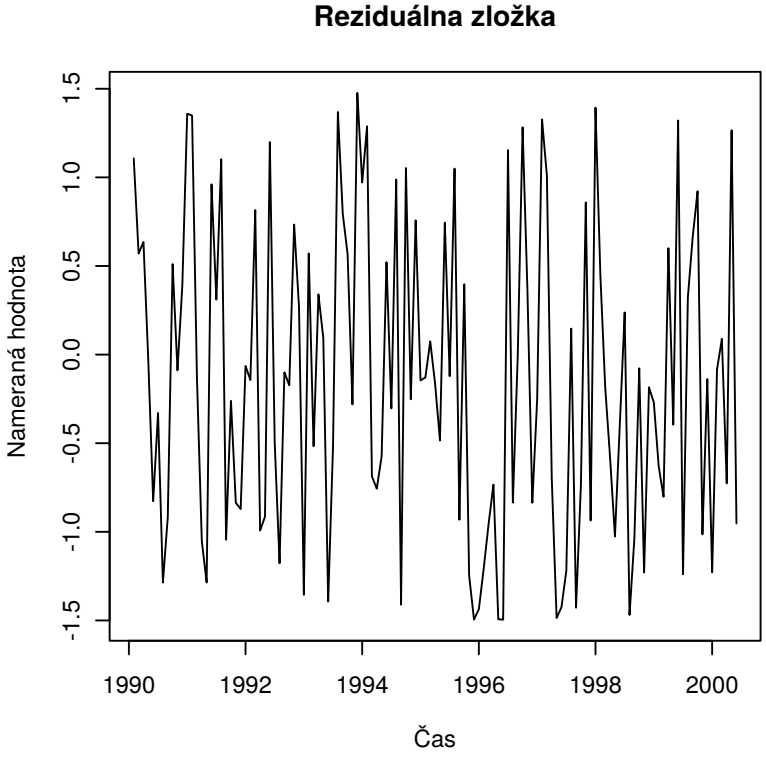


Obr. 1: Príklad trendovej zložky časového radu.

Cyklická zložka je spôsobená zmenami, ktoré sa cyklicky opakujú. Dĺžka periódy je 2 a viac rokov, čo zodpovedá strednodobému časovému horizontu. Táto zložka je zastúpená najmä pri ekonomických časových radoch napríklad podnikateľský cyklus pozostávajúci zo 4 fáz, ktoré sa stále opakujú [1].



Obr. 2: Príklad sezónnej zložky časového radu.



Obr. 3: Príklad reziduálnej zložky časového radu.

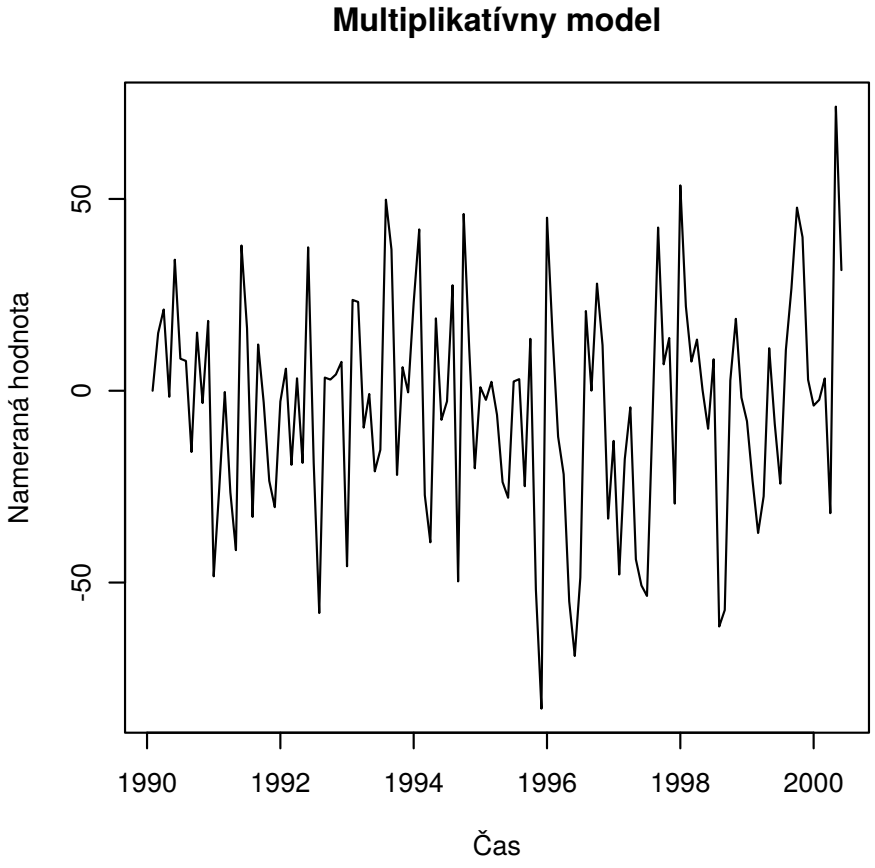
Sezónna zložka predstavuje kolísanie časových radov počas ročných období. Dôležitými faktormi pri tom sú napr. klimatické podmienky, tradície alebo počasie. Napríklad predaj zmrzliny sa v lete zvyšuje, ale počet predaných lyžiarskych súprav klesá [1].

Reziduálna zložka predstavuje veličinu, ktorá nemá žiadny opakovateľný vzor a ani dlhodobý trend. V časových radoch má nepredvídateľný vplyv na pozorovanú veličinu. V štatistike zatiaľ nie je definovaná metóda na jej meranie. Označuje sa aj ako náhodná zložka alebo biely šum. Je spôspobená nepredvídateľnými a nepravideľnými udalosťami [1].

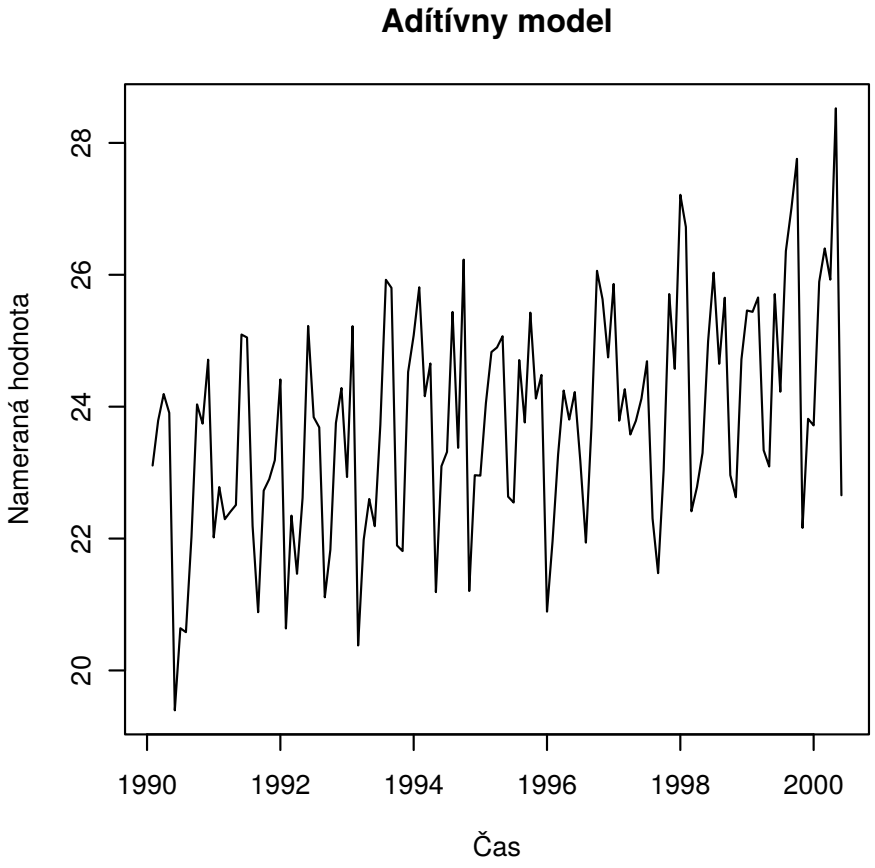
Vo všeobecnosti sa pre tieto 4 zložky používajú 2 rôzne modely. Je to multiplikatívny model a aditívny model.

$$\begin{aligned} Y(t) &= T(t) \times S(t) \times C(t) \times I(t) \\ Y(t) &= T(t) + S(t) + C(t) + I(t) \end{aligned} \tag{2}$$

Vo vzorci 2 predstavuje $Y(t)$ meranie v čase t. Premenné $T(t)$, $S(t)$, $C(t)$ a $I(t)$ sú zložkami trendu, sezónnosti, cyklu a náhodnosti. Multiplikatívny model, zobrazený na obrázku 4 je založený na predpoklade, že časové rady môžu byť na sebe závislé a môžu byť ovplyvňované medzi sebou, zatiaľ čo aditívny model, zobrazený na obrázku 5 predpokladá nezávislosť zložiek [1].



Obr. 4: Príklad multiplikatívneho modelu



Obr. 5: Príklad aditívneho modelu.

2.2 Analýza predičkných algoritmov

Na základe množstva výskumov, ktoré analyzovali predikčné algoritmy, sme si zvolili reprezentatívnu vzorku algoritmov. Našou snahou bude rôzne predikčné metódy optimalizovať pomocou optimalizačných algoritmov. Pred tým je potrebné analyzovať a pochopiť použité predikčné metódy. Taktiež je potrebné identifikovať ich vstupné parametre, keďže ich neskôr budeme optimalizovať.

2.2.1 Lineárna regresia

Najpoužívanejšia štatistická metóda, ktorá modeluje vzťah závislej premennej a vysvetľujúcej premmennej. Závislú premmenu predstavuje veličina, ktorú sa snažíme predpovedať, čo je v našom prípade spotreba elektriky. Vysvetľujúca premenná v sebe zahŕňa rôzne faktory, ktoré ovplyvňujú závislú premennú. Môžeme si pod tým predstaviť deň v týždni, počasie, tradície alebo rôzne udalosti, ktoré majú vplyv na predpoveď. [13, 16].

Predpokladajme typický regresný problém. Dáta pozostávajúce z množiny *n* meraní majú formát $\{(x_1, f(x_1)), ..., (x_n, f(x_n))\}$. Úlohou regresie je odvodiť funkciu $\hat{f}$ z dát, kde

$$\hat{f} : X \to \mathbb{R}, \text{ kde, } \hat{f}(x) = f(x), \forall x \in X, \tag{3}$$

Funkcia f vo vzorci 3 reprezentuje reálnu neznámu funkciu. Algoritmus použitý na odvodenie funkcie $\hat{f}$ sa nazýva indukčný algoritmus. Funkcia $\hat{f}$ sa nazýva model alebo prediktor.

Obvykle je úlohou regresie minimalizovať odchýlku funkcie pre štvorcovú chybu, konkrétne strednú štvorcovú chybu MSE [17].

Keďže časový rad pozostáva z viacerých zložiek, môžeme ho zapísať ako funkciu $L(t)$ definovanú ako

$$L(t) = Ln(t) + \sum a_i x_i(t) + e(t) \tag{4}$$

Vo vzorci 4 funkcia $Ln(t)$ predstavuje odber elektriky v čase t. Hodnota a_i je odhadovaný pomaly meniaci sa koeficient. Faktory $x_i(t)$ nezávisle vplývajú na spotrebu elektriky. Môže sa jednať napríklad o počasie alebo zvyky ľudí. Komponent $e(t)$ je biely šum, ktorý má nulovú strednú hodnotu a pevnú varianciu. Číslo n je počet meraní, obvykle 24 alebo 168, v našom prípade 96 meraní počas jedného dňa [13].

Lineárna regresná analýza je štatistická metóda používaná na modelovanie vzťahov, ktoré môžu existovať medzi veličinami. Nachádza súvislosti medzi závislou premennou a potenciálnymi vysvetľujúcimi premennými. Používame pri tom vysvetľujúce premenné, ktoré môžu byť namerané súčasne so závislými premennými alebo aj premenné z úplne iných zdrojov. Regresná analýza môže byť tiež použitá na zlúčenie trendu a sezónných zložiek do modelu. Keď je raz model vytvorený, môže byť použitý na zásah do spomínaných vzťahov alebo, v prípade dostupnosti vysvetľujúcich premenných, na vytvorenie predikcie [15].

Viacnásobná lineárna regresia sa snaží modelovať vzťah medzi dvoma alebo viacerými vysvetľujúcimi premennými a závislou premennou vhodnou lineárnou rovnicou pre pozorované dáta. Výsledný model je vyjadrený ako funkcia viacerých vysvetľujúcich premenných [8].

Túto funkciu môžeme zapísať ako

$$Y(t) = V_t a_t + e_t \tag{5}$$

Vo vzorci 5 t označuje čas, kedy bolo meranie uskutočnené. $Y(t)$ predstavuje celkový nameraný odber elektriky. Vektor V_t reprezentuje hodnoty vysvetľujúcich premenných v čase merania. Vysvetľujúce premenné môžu predstavovať meteorologické vplyvy, ekonomický nárast, ceny elektriky či kruzy mien. Chybu modelu v čase t zapíšeme ako e_t [13, 16].

2.2.2 Stochastické modely

Tieto metódy časových radov sú založené na predpoklade, že dáta majú vnútornú štruktúru, ako napr. autokoreláciu, trend či sezónnu variáciu. Najprv sa precízne zostaví vzor zodpovedajúci dostupným dátam a potom sa na jeho základe predpovie budúca hodnota veličiny [13].

Autoregresný model predpovedá budúcu hodnotu premmennej ako súčet lineárnej kombinácie p predchdzajúcich meraní, náhodnej chyby a konštanty. V literatúre sa označuje ako AR (autoregressive model). Matematicky môžeme autoregresný model zapisať ako

$$y_t = c + \sum_{i=1}^{p} \varphi_i y_{t-i} + \varepsilon_t \tag{6}$$

Vo vzorci 6 hodnota y_t predstavuje predpovedanú hodnotu v čase t. Náhodnú chybu v čase t zapíšeme ako ε_t. Hodnoty φ_i sú parametre modelu a c je konštanta. Konštantou p označujeme rad modelu [1].

Model kĺzavého priemeru na rozdiel od autoregresného modelu používa ako vysvetľujúce premenné chyby predchádzajúcich meraní a nie priamo hodnoty. V literatúre sa označuje ako MA (moving average). Matematicky môžeme tento vzťah zapísať ako

$$y_t = \mu + \sum_{j=1}^{q} \Theta_j \varepsilon_{t-j} + \varepsilon_t \tag{7}$$

Vo vzorci 7 hodnota y_t predstavuje strednú hodnotu postupnosti meraní v čase t. Hodnoty Θ_j sú parametre modelu a konštantou q označujeme rad modelu. Vychádzame z predpokladu, že náhodná zložka ε_t je biely šum, čo je rovnomerne distribuovaná náhodná premenná, ktorá má nulovú strednú hodnotu a konštantnú varianciu σ^2 [1].

Autoregresívny model kĺzavého priemeru reprezentuje súčasnú hodnotu časového rádu linárne na základe jeho hodnôt a hodnôt bieleho šumu v predchádzajúcich periódach. V literatúre sa označuje ako ARMA (autoregressive moving average) [13].

Ide o kombináciu autoregresie (AR) a kĺzavého priemeru (MA), vhodnú pre modelovanie jednorozmerných časových radov. Matematicky môžeme reprezentovať tento model ako súčet predchádzajúcich modelov

$$y_t = c + \varepsilon_t + \sum_{i=1}^{p} \varphi_i y_{t-i} \sum_{j=1}^{q} \Theta_j \varepsilon_{t-j} \tag{8}$$

Rad modelu určuje p a q [1].

Autoregresívny integrovaný model kĺzavého priemeru je generalizáciou modelu ARMA. V literatúre sa označuje ako ARIMA (autoregressive integrated moving average). Modely typu ARMA môžu byť použité iba na statické časové rady. Mnoho časových radov v praxi vykazuje nestatické správanie a napr. tie, ktoré obsahujú komponenty trendu a sezónnosti. Kvôli tomu bol navrhnutý model ARIMA, ktorý je zahŕňa v sebe aj prípady nestatických časových radov. Z nestatických časových radov sa vytvárajú statické pomocou konečného počtu derivovaní dátových bodov. Vzniká tak matematický model, ktorý môžeme zapísať ako

$$\left(1 - \sum_{i=1}^{p} \varphi_i L^i\right)(1 - L)^d y_t = \left(1 + \sum_{j=1}^{q} \Theta_j L^j\right) + \varepsilon_t \tag{9}$$

Vzorec 9 môžeme zapísať aj jednoduchšie a to

$$\varphi(L)(1 - L)^d y_t = \Theta(L)\varepsilon_t \tag{10}$$

Vo vzorci 9 predstavujú premenné p, d a q rad autoregresného modelu, modelu kĺzavého priemeru a integrovaného modelu. Hodnota d zodpovedá stupňu derivovania, zvyčajne je rovná 1. V prípade, že $d = 0$ dostaneme klasický ARMA model. Rovnakým spôsobom vieme dostať modely AR a MA [1].

2.2.3 Regresia založená na podporných vektoroch

Regresia založená na podporných vektoroch a metóda podporných vektorov je založená na štatistickej teórií učenia, nazývanej aj VC teória, podľa svojich autorov, Vapnik a Chervonenkisa [19].

Metóda podporných vektorov je použitá na množstvo úloh strojového učenia ako je rozoznávanie vzorov, klasifikácia objektov a v prípade predikcií časových radov to je regresná analýza. Regresia založená na podporných vektoroch je postup, ktorého funkcia je predpovedaná pomocou nameraných dát, ktorými je metóda podporných vektorov natrénovaná. Toto je odklon od tradičných predpovedí časových radov, v zmysle že metodá podporných vektorov nepoužíva žiadny model, ale predikciu riadia samotné dáta [19].

Táto predikčná metóda poskytuje malý počet voľných parametrov. Garantuje konvergenciu k ideálnemu riešeniu a môže byť výpočtovo efektívna [19].

2.2.4 Rozhodovacie stromy

Rozhodovacie stromy sú jednou z najrozšírenejších učiacich metód. Používajú sa najmä na klasifikáciu, ale v súčasnosti sa využívajú aj na regresiu. Rozhodovací strom je reprezentovaný ako množina uzlov a im prislúchajúcich hrán. Uzly reprezentujú atribúty a výstupné hrany sú vždy označené konkrétnou hodnotou pre atribút, z ktorého vychádzajú. Rozhodovanie začína v koreni stromu a končí po dosiahnutí listového uzla. Pre riešenie jedného problému je možné vytvoriť stromy s rôznym počtom a usporiadaním uzlov. Najlepším riešením je strom s najmenším počtom rozhodovacích uzlov [18].

Hlavnou výhodou rozhodovacích stromov oproti ostatným modelom je, že strom produkuje model, ktorý je reprezentovateľný ako pravidlá alebo logické výroky. Taktiež klasifikácia sa môže vykonávať bez komplikovaných výpočtov. Táto technika môže byť použítá ako na kategorické premenné tak aj na spojité. Rozhodovacie stromy neposkytujú také dobré výsledky pre nelineárne dáta ako neurónové siete. Vo všeobecnosti je táto metóda vhodnejšia pre predpovedanie kategorických dát alebo na dáta, ktoré v sebe obsahujú viditeľný trend [23].

Regresný rozhodovací strom

2.2.5 Náhodné lesy

Náhodné lesy sú kombináciou predpovedí stromov. Každý strom závisí od hodnoty náhodného vektora s rovnakým rozdelením. Chyba lesu závisí od sily jednotlivých strom a koreláciou medzi nimi. Náhodný les môžeme definovať ako klasifikátor pozostávajúci z množiny stromov $\{h(x, \Theta_k), k = 1, 2, ...\}$, kde $\{\Theta_k\}$ sú nezávislé rovnomerne distribuované náhodné vektory a každý strom sa podiela na hlasom na voľbe triedy vstupu x. S nárastom počtu stromov hodnota $\{\Theta_k\}$ konverguje k určitému bodu. Tým je zabezpečené, že náhodné lesy sa s pridávajúcim počtom stromov nepretrénujú ale veľkosť chyby sa postupne ústáli. Pri výbere náhodného vektora sa snažíme pri zachovaní jeho sily minimalizovať koreláciu, čím zvyšujeme presnosť celého výpočtu [3].

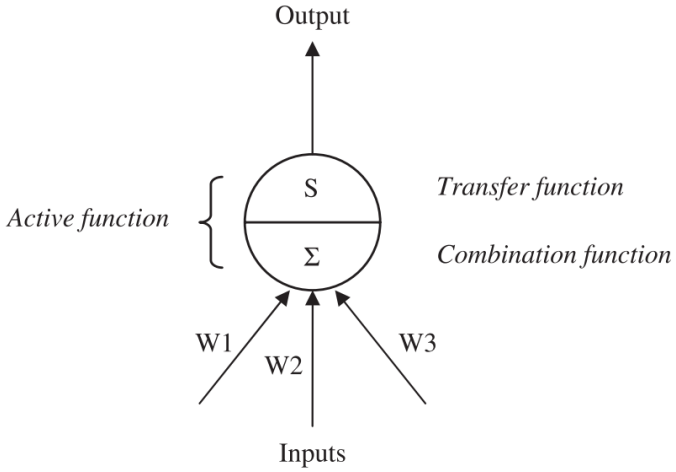
Väčšinou sú náhodné lesy používané na klasifikačné problémy, avšak je možné ich aplikovať aj na regresiu. Regresné náhodné lesy sú tvorené rastom stromov závyslých na náhodnom vektore $\{\Theta\}$. Prediktor stromu $h(x, \Theta)$ nadobúda číselné hodnoty narozdiel od štíkov tried ako je to pri klasifikačných problémoch. Predpokladáme trénovaciu množinu, ktorá je nezávislou distribúciou náhodneho vektora Y, X. Potom môžeme strednú štvorcovú generalizačnú chybu pre číselný prediktor h matematicky zapísať ako

$$E_{X,Y}(Y - h(X))^2 \tag{11}$$

Prediktor náhodného lesu je tvorený priemerom k stromov, čo zapíšeme ako $h(x, \Theta_k)$ [3].

2.2.6 Neurónové siete

Návrh neurónových sietí je inšpirovanný neurofyziológiou ľudského mozgu. Model je analytická technika modelujúca procesy učenia v kognitívnych systémoch a neurologických funkciách mozgu. Má schopnosť predpovedať budúcu hodnotu merania konkrétnej premennej na základe hodnôt z predchádzajúcich meraní. Tento proces sa inak nazýva aj učenie z existujúcich dát. Tok v neurónovej sieti preteká cez jednotlivé neuróny. Na obrázku 6 môžeme vidieť príklad takéhoto neurónu [23].



Obr. 6: Príklad neurónu v neurónovej sieti.

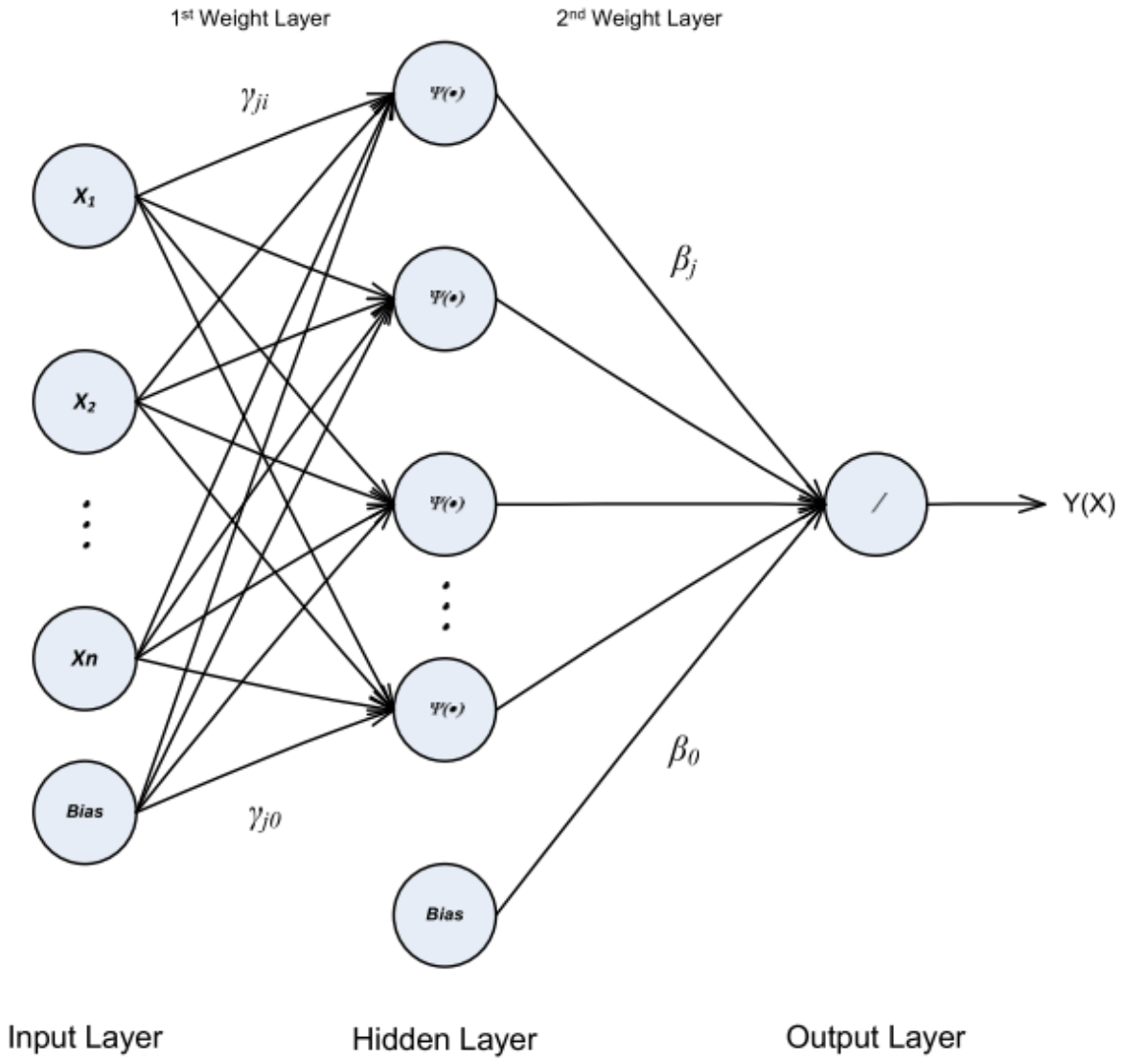
Neurónová sieť predstavuje orientovaný graf uzlov. Uzol neurónovej siete sa nazývame neurón. Každý uzol počíta svoj výstup na základe vstupov od susedných uzlov. Výpočet prebieha aplikovaním funkcie, ktorá sa nazýva sigmoid, na vážený súčet vstupov [9].

Trénovanie sieti je proces nastavovania, čo najlepších váh na vstupy jednotlivých neurónov. Chyba neurónovej siete sa najčastejšie počíta pomocou spätnej propagácie (backpropagation) čím dostaneme rast chyby pre danú neurónovú sieť [23].

Je veľa typov neurónových sietí napríklad viacvrstvové perceptónové siete, samoriadiace siete, siete s viacerými skrytými vrstvami, alebo aj viacvrstvové spätne propagované neurónové siete, pričom príklad takejto siete môžeme vidieť aj na obrázku 7. V každej skrytej vrstve je množstvo neurónov. Hlavnou výhodou je, že väčšina sietí nepotrebuje. Na druhej strane, trénovanie obvykle zaberá veľa času. Výstupom siete je lineárna rovnica váh prepojených so vstupom [13].

Viacvrstvový perceptón je jednou z najpoužívanejších neurónových sietí. Pozostáva z uzlov a im prislúchajúcim hranám. Uzly sú zoskupované do rôznych vrstiev. Prvá vrstva je vstupná vrstva, kde počet d označuje počet vstupných parametrov vstupujúcich do siete. Táto vrstva je následne prepojená hranami so skrytou vrstvou pozostávajúcou z h uzlov. Tá je potom prepojená s výstupnou vrstvou s c uzlami. Kvôli tomu sa tieto siete zvyknú označovať aj ako dopredné siete [18].

Elementy skrytých a výstupných vrstiev sú umelé neuróny pozostávajúce z uzlov, viacerých vstupujúcich a jednou výstupnou hranou. Funkciou neurónu je transformovať lineárnu kombináciu vstupov pomocou nelineárnej aktivačnej funkcie, čiže každú vstupnú hranu prenásobiť jej váhou a výsledok týchto súčinou sčítať. Tak dostaneme pre neurón j vzorec 12



Obr. 7: Príklad viacvrstvovej spätne propagovanej neurónovej siete [12].

opisujúci lineárnu kombináciu vstupov a_j

$$a_j = \sum_{i=1}^{d} w_{ji} x_i \tag{12}$$

Príčom váha w_{ji} označuje váhu medzi neurónom i na vstupnej vrstve a neurónom j na skrytej vrstve [18].

Aktivovanie neurónu j závisí od jeho aktivačnej funkcie $g(a_j)$. Jednu z najpoužívanejších aktivačných funkcií, logistickú sigmoidnú funkciu, môžeme matematicky zapísať ako

$$g(a) \equiv \frac{1}{1 + exp(-a)} \tag{13}$$

Je zrejmé, že funkcia zo vzoraca 13 vracia hodnoty v rozmedzí $(0, 1)$ [18].

2.2.7 Učenie súborov klasifikátorov

Používa sa na jednodňovú predikciu. Ak *h* je počet meraní, ktoré sú denne dostupné, v deň *t* sa vykoná *h* predikcií podľa váženého priemeru *m* modelmi. Nasledujúci deň sa vypočíta chyba predpovede, na základe ktorej sa znova prepočítajú váhy a každý model sa aktualizuje[8].

Učenie súborov klasifikátorov môžeme rozdeliť na homogénne a heterogénne učenie.

Homogénne učenie súborov klasifikátorov Pozostáva z modelov rovnakého typu, ktoré sa učia na rôznych podmnožinách datasetu.

Heterogénne učenie súborov klasifikátorov Aplikuje rôzne typy modelov nad rovnakými dátovými množinami[8].

2.2.8 Exponencionálne vyrovnávanie

Táto metóda, tak ako vyššie popísané metódy, najprv na základe dát z predchádzajúcich meraní vytvorí model. Ten sa v ďalej použíje na redpovedanie budúcich dát. Tento vzťah môžeme matematicky zapísať ako

$$y(t) = \beta(t)^T f(t) + \varepsilon(t) \tag{14}$$

Vo vzorci 14 sa opäť nachádza biely šum $\varepsilon(t)$. Hodnota $\beta(t)$ predstavuje **coefficient vector** a $f(t)$ **fitting function** [16].

Jednou z predikčných metód časových radov, ktorá používa exponencionálne vyrovnávanie je aj Wintersova metóda. Pri sezónnych dátach sú použité 3 vyrovnávacie parametre a to trend, sezónosť a stacionárnosť. Analýza a predpoveď priamym aplikovaním Wintersovej metódy môže byť náročné, keďže dátové množiny väčšinou obsahujú **riedke pozorovania (sparse values)**. Tento problém môžeme vyriešiť kombináciou ostatných metód, ako je napr. autoregresívny model či spektrálna analýza, s exponencionálnym vyrovnávaním [16].

2.3 Analýza optimalizačných algoritmov

Optimalizačné algoritmy, ktoré majú potenciál nájsť globálne alebo lokálne riešenie problému. Lokálne optimalizácie, nazývané aj vyhľadávacie algortimy, sa pokúšajú nájsť lokálne minimum v okolí štartovacieho riešenia. Väčšina týchto algoritmov je deterministických. Pri hľadaní minima používajú vyhodnocovaciu funkciu, na základe ktorej, aktualizujú doterajšie riešene. Z nových možných riešení je najlepšie to s najnižšiou hodnotou, čiže to ktoré je najlacnejšie. Kvôli tejto vlastnosti sa zvyknú tieto algortimy označovať aj ako nenásytné algoritmy (greedy algorithms). Spravidla tieto algoritmy nenájdu globálne minimum v prípade, že sa nachádza ďalej od štartovacieho riešenia ako nejaké lokálne minimum [21].

Na druhej strane, optimalizačné algoritmy s pontenciálom nájdenia globálneho minima nachádzajú riešenie, ktoré postupne konverguje k optimálnemu riešeniu. To ale nie je algoritmami úplne garantované. Algortimy s globálnym potencionálom majú väčší prehľad o svojom okolí, a preto uviaznutie v lokálnom minime je zriedkavé [21].

Nasledujúca tabuľka 1 zobrazuje rozdelenie algoritmov s pontenciálom nájdenia globálneho minima. Algoritmy sú biologicky inšpirované a ich ďalšie delenie vyplýva zo spoločných znakov, na ktorých sú založené.

Tabuľka 1: Tabuľka rozdelenia vybraných biologicky inšpirovaných algoritmov [7].

Model ľudskej mysle	**Umelé imunitné systémy**	**Rojová inteligencia**	**Ostatné napr. prírodné úkazy**
Teória fuzzy množin	Evolučný algoritmus	Optimalizácia kolóniu mravcov	Tektonické dosky
Rough Set theory	Umelé neurónové siete	Optimalizácia rojom častíc	Veľký kolaps
Granular COmputing	Celulárný automat	Foraging bacteria	Oceánske prúdy
Perception-Based Computing	Membrane computing	Včelí zhlukovací algoritmus	Prílivové vlny
Wisdom Technology	**DNA computing**	Vyhľadávanie kukučkou	Sopečné erupcie
Anticipatory Computing		Inteligentná kvapka vody	Zemetrasenia

2.3.1 Genetický algoritmus

Genetický algoritmus patrí medzi bilogicky inšpirované algoritmy patriace do triedy evolučných algoritmov. Genetický algoritmus je stochastický optimalizačný algoritmus, ktorého úlohou je nájdenie globálneho riešenia pre zadaný problém, čiže sa nestane, že riešenie spadne do lokálne minima a nenájde sa tak optimálne riešenie. Od tradičných algoritmov sa líšia hlavne v počte riešení, ktoré sú kandidátmi na najlepšie riešenie. Tradičné vyhľadávacie algoritmy prehľadávajú dôkladne iba jedno riešenie, zatiaľ čo genetické algoritmy hlbšie spracujú viacero kandidátov naraz. Každý kandidát na optimálne riešenie problému je reprezentovaný dátovou štruktúrou, ktorú označujeme pojmom jedinec. Súbor jedincov tvorí populáciu. Začiatok procesu začína náhodnými riešeniami populácie, ktorý sa postupne vylepšuje [5].

Vytvorenie novej generácie sa vykonáva pomocou genetických operátorov: a to selekciou, krížením a mutáciou. Proces selekcie vyberie kvalitnejšie chromozóny, ktoré prežijú a vyskytnú sa tak aj v ďalšej generácii [22].

Pri genetických algoritmoch sa zavádzajú pojmy ako chromozón, fitness funkcia, kríženie, elitárstvo, operátor reprodukcie či mutácie [5].

Chromozón je pomenovanie pre jedinca. V literatúre sa tiež zvykne používať označenie kandidát [2].

Inicializácia je proces, ktorý po vygenerovaní generácie priradí kandidátom náhodné hodnoty. Pri prehľadávaní dvojrozmerného priestoru to budú náhodné hodnoty oboch súradníc [14].

Fitness funkcia je funkcia určujúca efektívnosť chromozónu. Každý jedinec v množine je ohodnotený pomocou tejto funkcie, ktorá vracia číselnú reprezentáciu riešenia, čiže ako dobre daný kandidát vyriešil zadaný problém. Pri hľadaní optimálneho riešenia sa porovnáva hodnota fitness funkcie aktuálneho riešenie s hodnotou funkcie cieľového riešenia, ale vo všeobecnosti, čím je hodnota väčšia, tým je kandidátovo riešenie lepšie [5, 14, 22].

Tabuľka 2: Proces vytvorenia novej generácie z rodičovských chromozónov.

rodič č. 1	0110	0101 0010	0011
rodič č. 2	1100	1101 1110	1111
potomok č. 1	0110	1101 1110	0011
potomok č. 2	1100	0101 0010	1111

Operátor reprodukcie je obvykle prvý operátor, ktorý sa uplatní na populáciu. Operátor náhodne vyberie reťazce z dvoch chromozónov na párenie [5].

Kríženie je operátor kombinácie. Kríženie vykonáva výmenu blokov chromozónov. Z druhého chromozónu je vybraný reťazec náhodnej veľkosti, ktorý sa vymení s rovnako dlhým reťazcom z prvého chromozónu. V tabuľke 2 je tento proces znázornený graficky [5].

Vzniká problém ako, čo najvhodnejšie, určiť bod kríženia a vybrať tak najkvalitnejšie bloky chromozónu. Kvôli tomu je potrebné definovať nový element nazývaný väzba, označovaný ako d_i

$$d_i = \frac{a_i + a_{i+1}}{2} \tag{15}$$

Za predpokladu, že chromozón reprezentujeme ako maticu veľkosti $1 \times N$, potom väzbu definovanú vzorcom 15 môžeme rezprezntovať ako maticu $1 \times (N-1)$. Vypočítaním priemeru susedných hodnôt aj pre cieľovú maticu a nájdením priemeru matice, určíme body kríženia. Chromozón rozdelíme na miestach, kde je väzba, čo najmenšia [22].

Elitárstvo je to proces pridávania chromozónov s najlepšou hodnotou funkcie fitness priamo do ďalšej populácie. Zaisťuje to, že najlepšie riešenie budúcej generácie bude vždy lepšie, alebo pri najhoršom rovnaké, ako najlepšie riešenie predchádzajúcej generácie [6].

Operátor mutácie sa vykoná po vykonaní operátora reprodukcie. Mutácia chromozónu väčšinou predstavuje operáciu, ktorá s nízkou pravdepodobnosťou invertuje jeden bit v chromozóne [5].

Princíp genetického algoritmu môžeme znázorniť v nasledujúcom pseudokóde [5]

Algorithm 1 Pseudokód genetického algoritmu

1: Náhodne inicializovanie jedincov
2: Vyhodnotenie fitness funkcie pre každého jedinca
3: Výber jedincov pre ďalšiu populáciu na základe fitnesss funkcie
4: Kríženie jedincov
5: Mutovanie jedincov
6: Ak bolo nájdené žiadané optimálne riešenie pokračuj, inak návrat na krok 2
7: Vráť optimálne riešenie

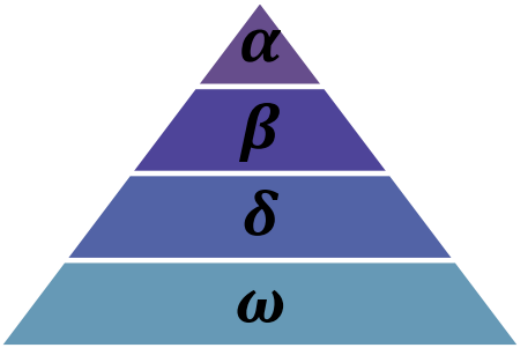
Veľkou výhodou genetického algoritmu je, že mutácia predchádza skĺznutiu do lokálnych miním a kombinácia chromozónov vedie k rýchlemu približovaniu sa k optimálnemu riešeniu. Napriek týmto výhodám, majú genetické algoritmy aj niekoľko nevýhod [6].

- Reprezentovanie kandidátov je príliš obmedzujúce

- Mutácia a kríženie sú v súčasnosti aplikovateľné iba na chromozóny reprezentované bitovým reťazcvom alebo číslami
- Definovanie fitness funkcie je často netriviálnou záležitosťou a jej generalizácia je náročná

2.3.2 Optimalizácia svorkou divých vlkov

Algoritmus je založený na správaní vlka sivého, ktorý je na vrchole potravinového reťazca a preferuje život vo svorke. Lov pozostáva zo stopovania, prenasledovania, približovania sa, obkľúčenia koristi a útokom na korisť. Vlkov vo svorke možno rozdeliť do niekoľkých skupín. V obrázku 8 je znázornená hierarchia týchto skupín [20].



Obr. 8: Hierarchia vlkov vo svorke.

Alfa vlky sú vodcami svorky. Ich úlohou je robiť rozhodnutia ohľadom lovu, miesta na spanie či času zobudenia. Tieto príkazy diktujú svorke, avšak bolo pozorované aj demokratické správanie, kedy alfa vlky nasledovali ostatné vlky zo svorky. Všetky vlky uznávajú postavenie alfa vlkov vo svorke. Zaujímavosťou je, že vodcom svorky nemusí byť najsilnejší jedinec, ale môže to byť aj jedinec najlepší v organizovaní svorky [20].

Beta vlky sú druhým stupňom v hierarchii vlčej svorky, podriadené alfa vlkom, ktorým pomáhajú v rozhodovaní a organizácií svorky. Sú najlepšími kandidátmi na alfa vlkov v prípade úmrtia alebo zostarnutia alfa vlkov. Mali by rešpektovať alfa vlky a organizovať nižšie postavené vlky. Hrajú rolu radcu a ďalej distribujú príkazy a vracajú sa s odozvou na ne [20].

Delta vlky inak nazývané aj podriadené vlky zastupujú vo svorke úlohy prieskumníkov, strážcov, starejších, lovcov či opatrovateľov. Prieskumníci hliadkujú hranice a varujú svorku pred nebezpečím. Strážcovia zabezpečujú bežpečie svorky. Starejší sú skúsenými vlkmi, ktorí boli alfa alebo beta vlkmi. Lovci pomáhajú alfa a beta vlkom pri love. Opatrovatelia sa starajú o slabé, choré alebo zranené vlky [20].

Omega vlky majú najnižšiu hodnosť vo svorke. Hrajú rolu obetných baránkov, ktoré sa podrobia ostatným vlkom. K potrave sa dostanú ako úplne posledné. Aj keď sa možno javí ich postavenie vo svorke zbytočné, boli pozorované prípady, kedy ich strata spôsobila vo svorke nedorozumenia [20].

Spoločenská hierarchia reprezentovaná matematickým modelom, označuje najvhodnejšie riešenie ako alfa, druhé a tretie najvhodnejšie ako beta resp. delta. Riešenia ostatných kandidátov označujeme ako omega. Algoritmus optimalizácie (v prírode lovu) je vedený alfa, beta a delta kandidátmi, ktorí sú nasledovaní kandidátmi omega [20].

Obkľúčenie koristi môžeme matematicky reprezentovať ako

$$\begin{gathered}\vec{D} = |\vec{C} \cdot \vec{X_p}(t) - \vec{X}(t)| \\ \vec{X}(t+1) = \vec{X_p}(t) - \vec{A} \cdot \vec{D} \\ \Downarrow \\ \vec{D_\alpha} = |\vec{C_1} \cdot \vec{X_\alpha} - \vec{X}|, \vec{D_\beta} = |\vec{C_2} \cdot \vec{X_\beta} - \vec{X}|, \vec{D_\delta} = |\vec{C_3} \cdot \vec{X_\delta} - \vec{X}| \\ \vec{X_1} = \vec{X_\alpha} - \vec{A_1} \cdot \vec{D_\alpha}, \vec{X_2} = \vec{X_\beta} - \vec{A_2} \cdot \vec{D_\beta}, \vec{X_3} = \vec{X_\delta} - \vec{A_3} \cdot \vec{D_\delta}\end{gathered} \tag{16}$$

Aktuálnu iteráciu vo vzorci 16 zapíšeme ako t, vektor $\vec{X_p}$ predstavuje vektor polohy koristi a $\vec{X}$ je vektor polohy vlka. Z tohto predpokladu môžeme odvodiť spodnú dvojicu vzorcov, ktoré opisujú obkľúčenie koristi. Vektory $\vec{A}$ a $\vec{C}$ sú koeficienty, ktoré zapíšeme ako

$$\begin{aligned}\vec{A} &= 2\vec{a} \cdot \vec{r_1} - \vec{a} \\ \vec{C} &= 2 \cdot \vec{r_2}\end{aligned} \tag{17}$$

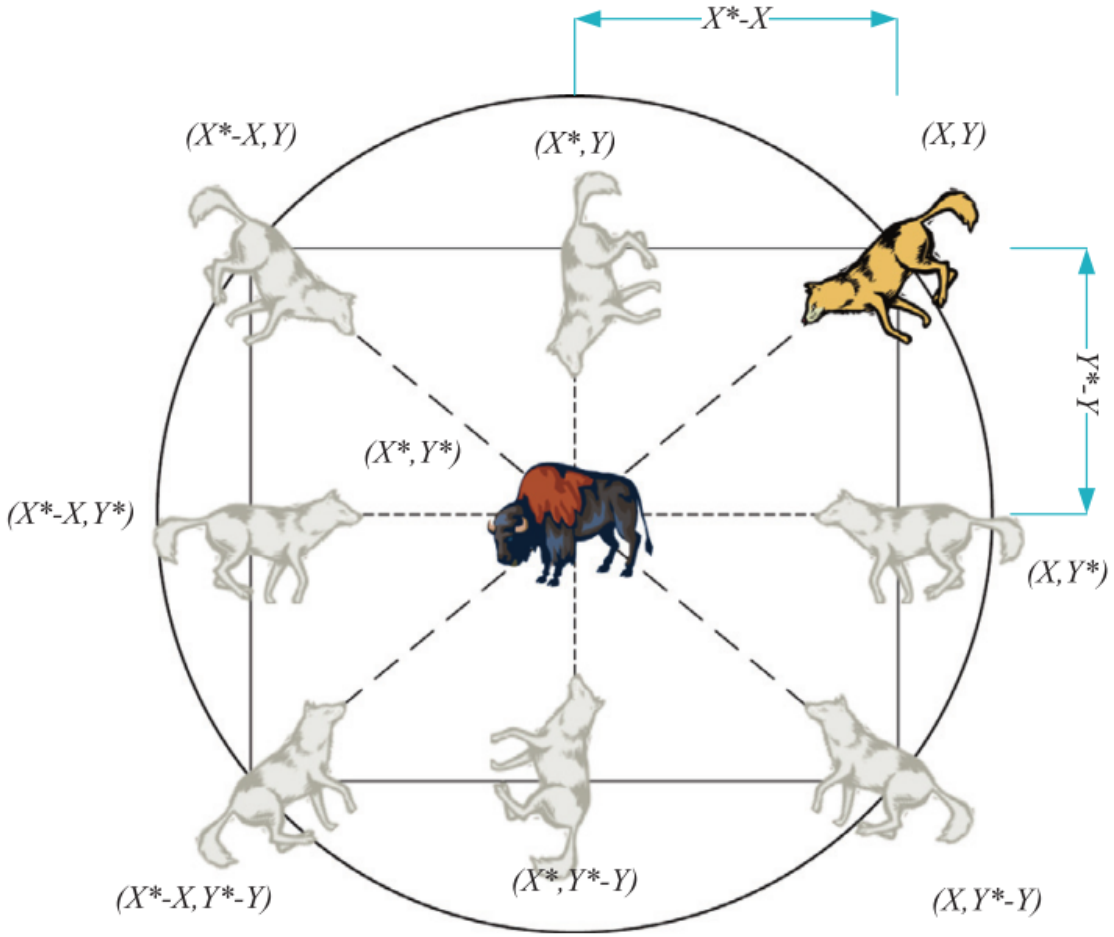
Vo vzorci 17 premenná $\vec{a}$ sa počas výpočtu lineárne znižuje od 2 po 0. Vektory $\vec{r_1}$ a $\vec{r_2}$ sú náhodnými vektormi v rozsahu [0, 1]. Vlk môže svoju pozíciu (X, Y) aktualizovať v závislosti od pozície koristi (X^*, Y^*). Obrázok 9 ilustruje možné aktualizované pozície, ktoré môže najlepší agent dosiahnuť. Tieto pozície získame aplikovaním vzorcov 16 [20].

Hľadanie koristi je zabezpečené vektorom $\vec{A}$, ktorý mimo intervalu [-1, 1] núti vlky divergovať od seba, čím je zdôraznená potreba prehľadávať okolie a neskĺznuť do lokálneho minima. Náhodný vektor $\vec{C}$ similuje prekážky v prírode, s ktorými sa vlk stretne pri hľadaní koristi. V závislosti od vygenerovanej hodnoty, môže simulovať aj opačný prípad, ktorý je pre vlka priaznivejší [20].

Lov je vedený alfa vlkmi, beta a delta vlky sa tiež môžu na ňom príležitostne podielať. V prírode vlky disponujú schopnosťou rozpoznať umiestnenie koristi a obklúčiť ju. Avšak pri simulácií tohto správania, nemáme vedomosť o presnej polohe koristi. Preto na základe prvých troch najlepších riešení vypočítame predpokladanú polohu koristi, čím vznikne vzorec 18. Kandidáti, vrátane omeage vlkov, potom na základe rovnice 16 aktualizujú svoju polohu [20].

$$\vec{X}(t+1) = \frac{\vec{X_1} + \vec{X_2} + \vec{X_3}}{3} \tag{18}$$

Útok na korisť v matematickom modely dosiahne približovaním sa ku koristi, čiže znížením hodnoty $\vec{a}$. Tak dosiaheme aj zníženie hodnoty $\vec{A}$ až bude jeho hodnota v intervale [-1, 1]. Potom ďalšia vypočítana hodnota pozície vlka sa bude nachádzať medzi pôvodnou polohou vlka a polohou koristi. V prípade, že hodnota $\vec{A}$ sa nachádza mimo spomínaného intervalu, vlk diverguje od koristi, čo nastáva vo fáze hľadania koristi. Tým je zabezpečené prehľadávanie priestoru agentami [20].



Obr. 9: Príklad možného umiestnenia vlka v dvojrozmernom priestore na základe polohy koristi.

Princíp optimalizácie svorkou divých vlkov môžeme znázorniť v nasledujúcom pseudokóde [20]

Algorithm 2 Pseudokód optimalizácie svorkou divých vlkov

1: Náhodná inicializácia vlkov
2: Inicializácia a, A a C
3: Vyhodnotenie fitness funkcie pre každého jedinca
4: Priradenie do 3 najlepších riešení do X_α, X_β resp. D_δ
5: Opakuj kroky 5 až 10 pokiaľ nebude prekročený maximálny počet iterácií
6: Aktualizovanie polohy na základe rovnice 16 pre každého jedinca
7: Aktualizovanie a, A a C
8: Vyhodnotenie fitness funkcie pre každého jedinca
9: Aktualizovanie X_α, X_β a D_δ
10: Aktualizovanie počtu iterácií
11: Vráť optimálne riešenie X_α

2.3.3 Umelá kolónia včiel

V literatúre označovaný ako ABC algoritmus (Artifical bee colony) je pomerne nový medzi rojovými algoritmami. Princíp je založený na biologickom procese, správaní medonosných včiel pri hľadaní potravy. Každá včela na pracujúca v roji sa spolupodiela na tvorbe celého systému na globálnej úrovni. Správanie systému je určené lokálnym správaním, kde spolupráca a zladenie jedincov vedie k štruktúrovanému kolaboračnému systému [5].

V algoritme, kolónia umelých včiel pozostáva z 3 skupín včiel. Sú to včely robotnice, diváčky a prieskúmníčky. Včely si vymenieňajú informácie o zdrojoch potravy tancovaním (waggle dance) na tzv. tanečnej ploche. Včela diváčka čakajúca na tanečnej ploche, sa rozhoduje, ktorý zdroj potravy si vyberie. Včela robotnica najskôr tieto zdroje navštívi. Prieskumníčka uskutočňuje náhodné prehľadávanie priestoru. Pri aplikovaní ABC algoritmu, polovica včiel predstavuje robotnice, druhá polovica diváčky. Počet robotníc zodpovedá počtu zdrojov potravy v okolí úľu. Robotnica, ktorej zdroj je vyčerpaný inými včelami sa stáva prieskumníčkou [11].

Zväčšovaním množstva zdrojov potravy sa zväčšuje aj pravdepodobnosť vybratia zdroja diváčkou. Vďaka tomu je zabezpečené, že tanec robotníc, ktoré navštívili zdroje s najväčším množstvom nektáru, presvedčí najviac diváčok. Tie, na základe informácie o polohe zdroja, nájdu v jeho okolí nový zdroj, z ktorého budú zberať nektár. Po vyčerpaní nektáru, je tento zdroj opustený a včela sa presunie na zdroj, ktorý našli prieskumníčky [11].

Pozícia zdrojov jedla je reprezentácia možného riešenia daného problému. Množstvo nektáru je úmerné kvalite riešenia, ktoré je určené fitness funkciou. Počet robotníc a diváčiek je rovný počtu riešení v danej populácií. Každé riešenie je reprezentovateľné ako D-dimenzionálny vektor. Hodnota D môže predstavovať počet optimalizačných parametrov [11].

Včely majú riešenie uložené vo vlastnej pamäti, kde si vytvárajú aj modifikáciu riešení, čím nachádzajú nové riešenia, ktoré následné vyskúšajú. Ak nové riešenie je lepšie ako predchádzajúce, včely si ho zapamätajú a predchádzajúcé zabudnú. Inak si pamätajú staré riešenie. Keď všetky robotnice skončia proces hľadania, zdieľajú informácie o zdrojoch s diváčkami. Tie vyhodnotia zhromaždené informácie a vyberú zdroj s najlepším riešením. Opäť vytvoria nové riešenie jeho modifikovaním a skontrolujú ho [11].

Diváčky vyberajú zdroj v závislosti od pravdepodobnosti zdroja p_i, ktorú vypočítame ako

$$p_i = \frac{fit_i}{\sum_{j=1}^{n} fit_j} \tag{19}$$

Vo vzorci 19 predstavuje fit_i fitness funkciu, ktorá ohodnocuje riešenie i. Týmto spôsobom si vymieňajú medzi sebou informácie robotnice a diváčky [11].

Algoritmus počíta nové riešenie na základe starého, čo matematicky zapíšeme ako

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \tag{20}$$

Ak si počet včiel definujeme ako B potom na základe vzorca 20 platí $k \in 1, 2, ..., B$ a $j \in 1, 2, ..., D$. Indexy k a j sú vybrané náhodne, ale k musí byť rôzne od i. Hodnota ϕ_{ij} je z intervalu [-1, 1] vyberaná náhodne. Je zrejmé, že čím bude menší rozdiel medzi (x_{ij} a x_{kj}), tým bude menší rozdiel aj medzi starým a novým riešením [11].

Algoritmus má 3 konfigurovateľné parametre a to: počet robotníc alebo diváčok, ktorý je rovný počtu zdrojov jedla, limit, ktorý ohraničuje prehľadávaný priestor a maximálny počet cyklov, ktorý sa vykoná ak sa skôr nenájde optimálne riešenie [11].

Princíp umelej kolónii včiel znázorňuje nasledujúci pseudokód [11]

Algorithm 3 Pseudokód umelej kolónie včiel

1: Inicializácia včiel na náhodné zdroje potravy
2: Opakuj kroky 2 až 7 pokiaľ nebude dosiahnutý cieľ
3: Umiestnenie robotníc na zdroje potravy, ktoré boli nájdené
4: Vyhodnotenie zdroja na základe množstva nektáru
5: Umiestnenie diváčok na zdroje potravy na základe získaných informácií
6: Vyhodnotenie, ktoré včely sa stanú prieskumníčkami
7: Vyslanie prieskumníčok do prehľadávanie priestoru za účelom objavenia nových zdrojov potravy
8: Vráť optimálne riešenie

2.3.4 Kolónia mravcov

Pri tomto algoritme, mravce tiež opúšťajú mravenisko, kvôli hľadaniu zdrojov potravy náhodne. Potom vyhodnotia kvalitu zdroja potravy a donesú ho naspäť do mraveniska. Zanechávajú pri tom na zemi chemické stopy. Sila týchto stôp závisí od kvality nájdeného zdroja potravy. Mnoho výskumov využíva tento algoritmus na riešenie NP problémov, ako napríklad problém obchodných cestujúcich, vyfarbovanie grafov, smerovnaie áut alebo plánovacie problémy. Používa sa aj pri **cloud computing** na nájdenie optimálneho riešenia pri plánovaní úloh pre virtuálne servery [4].

Keď mravce hľadajú potravu prvý krát, hľadajú náhodne až kým nenájdu zdroj potravy. Zanechávajú pri tom za sebou chemickú stopu nazývanú feromón, ktorá tak vedie k zdroju. Tá následne priťahuje ostatné mravce k tomuto zdroju potravy. Tento proces pokračuje pokiaľ mravce nenájdu najkratšiu cestu vedúcu ku konkrétnemu zdroju potravy. Najkratšia cesta je určená naakumulovaným množstvom feromónov na ceste k zdroju potravy [4].

2.4 Meranie presnosti predpovedi

Pre vyhodnotenie efektívnosti a presnosti modelov je potrebné merať ich vlastnosti tak, aby sme ich vedeli medzi sebou porovnáva. V nasledujúcich spôsoboch merania sú použité pojmy ako aktuálna hodnota y_t, predpovedaná hodnota f_t alebo chyba predpovede e_t definovaná ako $e_t = y_t - f_t$. Veľkosť testovacej množiny budeme označovať ako n [1].

2.4.1 Stredná chyba predpovede

V literatúre označovaná ako MFE (mean forecast error). Matematickú funkciu zapísať ako

$$MFE = \frac{1}{n}\sum_{t=1}^{n} e_t \tag{21}$$

Týmto spôsobom meriame priemernú odchýlku predpovedanej hodnoty od aktuálnej. Zistíme tak smer chyby. Nevýhodou je, že kladné a záporné chyby sa vynulujú a potom nie je možné zistiť presnú hodnotu chyby. Pri namerani extrémnych chýb, nie sú nijak špeciálne penalizované. Taktiež hodnota chyby závisí od škály meraní a môže byť ovplyvnená aj transformáciami dát. Dobré predpovede majú hodnotu blízku 0 [1].

2.4.2 Stredná absolútna chyba

V literatúre označovaná ako MAE (mean absolute error). Patrí k jedným z najpoužívanejších. Funkciu môžeme zapísať ako

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|e_t| \tag{22}$$

Týmto spôsobom meriame priemernú absolútnu odchýlku predpovedanej hodnoty od aktuálnej. Zistíme tak celkový rozsah chyby, ktorá nastala počas predpovede. Narozdiel od merania chyby pomocou vzorca 21 sa kladné a záporné chyby nevynulujú, no ani napriek tomu nevieme určiť celkový smer chyby. Na druhej strane tiež nenastáva žiadna penalizácia pri extrémnych chybách, hodnota chyby závisí od škály meraní, môže byť ovplynená transformáciami dát a dobré predpovede majú hodnotu čo najbližšiu 0 [1, 10].

2.4.3 Stredná percentuálna chyba

V označovaná ako MPE (mean percentage error). Matematciky môžeme túto funkciu zapísať ako

$$MPE = \frac{1}{n}\sum_{t=1}^{n}\frac{e_t}{y_t} \times 100 \tag{23}$$

Vlastnosti sú veľmi podobné ako pri MAPE v časti 2.4.4. Chyba nám poskytuje prehľad o priemernej chybe, ktorá sa vyskytla počas predpovede. Naviac, oproti MAPE, získame prehľad o smere chyby, čo má však za následok, že opačné znamienka sa vynulujú. O modely, ktorého chyba MPE sa blíži k 0 nemôžeme s určitosťou tvrdiť, že funguje správne [1].

2.4.4 Stredná aboslútna percentuálna chyba

V literatúre označovaná ako MAPE (mean absolute percentage error). Vzorec, ktorým ju zapíšeme bude veľmi podobný vzorcu 23

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{e_t}{y_t}\right| \times 100 \tag{24}$$

Pomocou tohto merania chyby získavame percentuálny prehľad o priemernej absolútnej chybe, ktorá sa vyskytla počas predpovedi. Veľkosť chyby nezávisí od škály merania, ale je závislá od transformácií dát. Tiež nie je možné zistiť smer chyby a ani nenastáva žiadna penalizácia pri extrémnych chybách [1].

2.4.5 Stredná štvorcová chyba

V literatúre označovaná ako MSE (mean squarred error). Vzorcom ju zapíšeme ako

$$MSE = \frac{1}{n}\sum_{t=1}^{n}e_t^2 \tag{25}$$

Chyba meria priemernú štvorcovú odchýlku predpovedanej hodnoty. Opačné znamienka sa neovplyvňujú. Neposkytuje nám pohľad na smer chyby. Zabezpečuje penalizáciu extrémnych chýb. Zdôrazňuje fakt, že celková chyba je viac ovplyvnená jednotlivými veľkými chybami ako viacerými malými. Nevýhodou je, že chyba je veľmi citlivá na zmenu škály alebo transformáciu dát [1].

2.5 Zhodnotenie analýzy

3 Špecifikácia požiadaviek

4 Návrh riešenia

5 Implementácia

6 Zhodnotenie

7 Záver

8 Technická dokumentácia

9 Plán do nasledujúceho semestra

Poradie týždňa v letnom semestri	Popis plánovanej činnosti
1. týždeň	úprava vstupných dát na požadovaný formát, aplikovanie predikčných algoritmov
2. týždeň	aplikovanie predikčných a optimalizačných algoritmov
3. týždeň	implementácia optimalizačných algoritmov, ktorým chýba podpora knižníc v jazyku R
4. týždeň	implementácia optimalizačných algoritmov, tvorba kostry grafického rozhrania aplikácie
5. týždeň	tvorba jednotného rozhrania medzi aplikáciou a grafickým rozhraním
6. týždeň	implementácia grafického rozhrania a následne prepojenie s aplikáciou
7. týždeň	ošetrenie neplatných akcií vykonané používateľom
8. týždeň	testovanie aplikácie na rôznych vstupných dátach používateľmi
9. týždeň	tvorba technickej dokumentácie aplikácie
10. týždeň	testovanie algoritmov, dokumentácia a porovnanie algoritmov
11. týždeň	tvorba prezentácie a príprava na obhajobu projektu

Tabuľka 3: Čo ešte určite musím stihnúť VYMAZAŤ (podľa dôležitosti)

pridať ACO od Marca Doriga
pridať regresné stromy
nájsť a pridať vzorec pre SVR
pridať zhodnotenie analýzy
pridať špecifikáciu požiadaviek (opis use case)
pridať exponencionálne vyrovnávanie
pridať učenie súborov klasifikátorov

Literatúra

[1] Adhikari, R.: *An Introductory Study on Time Series Modeling and Forecasting*. Saarbrucken: LAP LAMBERT Academic Publishing, 2013, ISBN 9783659335082.

[2] Arun, K.; Rejimoan, R.: A survey on network path identification using bio inspired algorithms. In *2016 2nd International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB)*, Feb 2016, s. 387–389, doi:10.1109/AEEICB.2016.7538314.

[3] Breiman, L.: Random Forests. *Machine Learning*, ročník 45, č. 1, 2001: s. 5–32, ISSN 1573-0565, doi:10.1023/A:1010933404324.
URL http://dx.doi.org/10.1023/A:1010933404324

[4] Buhussain, A. A.; Grande, R. E. D.; Boukerche, A.: Performance Analysis of Bio-Inspired Scheduling Algorithms for Cloud Environments. In *2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, May 2016, s. 776–785, doi:10.1109/IPDPSW.2016.186.

[5] Chavan, S. D.; Kulkarni, A. V.; Khot, T.; aj.: Bio inspired algorithm for disaster management. In *2015 International Conference on Energy Systems and Applications*, Oct 2015, s. 776–781, doi:10.1109/ICESA.2015.7503455.

[6] Deolekar, R. V.: Defining parameters for examining effectiveness of genetic algorithm for optimization problems. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, March 2016, s. 1061–1064.

[7] Goel, L.; Gupta, D.; Panchal, V. K.; aj.: Taxonomy of nature inspired computational intelligence: A remote sensing perspective. In *2012 Fourth World Congress on Nature and Biologically Inspired Computing (NaBIC)*, Nov 2012, s. 200–206, doi:10.1109/NaBIC.2012.6402262.

[8] Grmanová, G.; Laurinec, P.; Rozinajová, V.; aj.: Incremental Ensemble Learning for Electricity Load Forecasting. *Acta Polytechnica Hungarica*, ročník 13, č. 2, 2016.

[9] Gruau, F.; lyon I, L. C. B.; Doctorat, O. A. D. D.; aj.: Neural Network Synthesis Using Cellular Encoding And The Genetic Algorithm. 1994.

[10] Gutiérrez, P. A.; Pérez-Ortiz, M.; Sánchez-Monedero, J.; aj.: Ordinal Regression Methods: Survey and Experimental Study. *IEEE Transactions on Knowledge and Data Engineering*, ročník 28, č. 1, Jan 2016: s. 127–146, ISSN 1041-4347, doi:10.1109/TKDE.2015.2457911.

[11] Karaboga, D.; Basturk, B.: A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of Global Optimization*, ročník 39, č. 3, 2007: s. 459–471, ISSN 1573-2916, doi:10.1007/s10898-007-9149-x.
URL http://dx.doi.org/10.1007/s10898-007-9149-x

[12] Kennedy, J.; Eberhart, R.: Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, ročník 4, Nov 1995, s. 1942–1948 vol.4, doi:10.1109/ICNN.1995.488968.

[13] Kumar Singh, A.; Khatoon, S.; Muazzam, M.; aj.: An Overview of Electricity Demand Forecasting Techniques. *Network and Complex Systems*, ročník 3, č. 3, 2013, ISSN 2225-0603.
URL www.iiste.org

[14] Lazinica, A.: *Particle swarm optimization*. Rijek, Crotia: InTech, 2009, ISBN 978-953-7619-48-0.

[15] Liu, L.; Hudak, G.: *Forecasting and Time Series Analysis Using the SCA Statistical System*, ročník zv. 1. Scientific computing Associates Corporation, 1992.
URL https://books.google.sk/books?id=8-HrAAAACAAJ

[16] Mahalakshmi, G.; Sridevi, S.; Rajaram, S.: A survey on forecasting of time series data. In *2016 International Conference on Computing Technologies and Intelligent Data Engineering (ICCTIDE'16)*, Jan 2016, s. 1–8, doi:10.1109/ICCTIDE.2016.7725358.

[17] Mendes-Moreira, J. a.; Soares, C.; Jorge, A. M.; aj.: Ensemble Approaches for Regression: A Survey. *ACM Comput. Surv.*, ročník 45, č. 1, December 2012: s. 10:1–10:40, ISSN 0360-0300, doi:10.1145/2379776.2379786.
URL http://doi.acm.org/10.1145/2379776.2379786

[18] Merz, C. J.: *Classification and Regression by Combining Models*. Dizertačná práca, University of California, 1998, aAI9821450.

[19] Sapankevych, N. I.; Sankar, R.: Time Series Prediction Using Support Vector Machines: A Survey. *IEEE Computational Intelligence Magazine*, ročník 4, č. 2, May 2009: s. 24–38, ISSN 1556-603X, doi:10.1109/MCI.2009.932254.

[20] Seeley, T. D.; Camazine, S.; Sneyd, J.: Collective decision-making in honey bees: how colonies choose among nectar sources. *Behavioral Ecology and Sociobiology*, ročník 28, č. 4, 1991: s. 277–290, ISSN 1432-0762, doi:10.1007/BF00175101.
URL http://dx.doi.org/10.1007/BF00175101

[21] Sen, M.; Stoffa, P.: *Global Optimization Methods in Geophysical Inversion*. Advances in Exploration Geophysics, Elsevier Science, 1995, ISBN 9780080532561.
URL https://books.google.sk/books?id=PT5MqSIvREMC

[22] Simonova, S.; Panus, J.: Genetic algorithms for optimization of thematic regional clusters. In *EUROCON 2007 - The International Conference on Computer as a Tool*, Sept 2007, s. 2061–2066, doi:10.1109/EURCON.2007.4400359.

[23] Tso, G. K.; Yau, K. K.: Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural networks. *Energy*, ročník 32, č. 9, 2007: s. 1761 – 1768, ISSN 0360-5442, doi:http://dx.doi.org/10.1016/j.energy.2006.11.010.
URL http://www.sciencedirect.com/science/article/pii/S0360544206003288