UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

APLIKÁCIA ARMA MODELOV A MACHINE LEARNINGU NA MODELOVANIE ČASOVÝCH RADOV

DIPLOMOVÁ PRÁCA

2017 Bc. Eva KUČMÍNOVÁ

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

APLIKÁCIA ARMA MODELOV A MACHINE LEARNINGU NA MODELOVANIE ČASOVÝCH RADOV

DIPLOMOVÁ PRÁCA

Študijný program:	Ekonomicko-finančná matematika a modelovanie
Študijný odbor:	9.1.9. Aplikovaná matematika
Školiace pracovisko:	Katedra aplikovanej matematiky a štatistiky
Vedúci práce:	doc. RNDr. Beáta Stehlíková, PhD.

Bratislava 2017 **Bc. Eva KUČMÍNOVÁ**

13515133

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Eva Kučmínová
Študijný program: ekonomicko-finančná matematika a modelovanie (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: 9.1.9. aplikovaná matematika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Aplikácia ARMA modelov a machine learningu na modelovanie časových radov.
Application of ARMA models and machine learning for time series modelling.

Cieľ: Pri ARIMA modelovaní sa kontroluje autokorelácia rezíduí. Tá je však schopná zachytiť len lineárnu závislosť. Môže sa teda stať, že v rezíduách nie je signifikantná autokorelácia, ale je medzi nimi prítomná nelineárna závislosť. Túto sa na praktických príkladoch pokúsime nájsť metódami machine learningu.

Vedúci: RNDr. Beáta Stehlíková, PhD.
Katedra: FMFI.KAMŠ - Katedra aplikovanej matematiky a štatistiky
Vedúci katedry: prof. RNDr. Daniel Ševčovič, CSc.

Dátum zadania: 21.01.2016

Dátum schválenia: 25.01.2016

prof. RNDr. Daniel Ševčovič, CSc.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie

Touto cestou by som sa chcela poďakovať svojej vedúcej diplomovej práce doc. RNDr. Beáte Stehlíkovej, PhD. za odborné rady, metodické usmernenia a poskytnutú odbornú literatúru, ktoré mi pomohli k vypracovaniu mojej diplomovej práce. Rovnako by som sa chcela poďakovať rodičom, blízkym a kolegom, špeciálne Mgr. Eve Karvajovej a Michalovi Páleníkovi, za ich trpezlivosť a snahu o vytvorenie vhodných podmienok na tvorbu mojej diplomovej práce.

Abstrakt v štátnom jazyku

KUČMÍNOVÁ, Eva: Aplikácia ARMA modelov a machine learningu na modelovanie časových radov [Diplomová práca], Univerzita Komenského v Bratislave, Fakulta matematiky, fyziky a informatiky, Katedra aplikovanej matematiky a štatistiky; školiteľ: doc. RNDr. Beáta Stehlíková, PhD., Bratislava, 2017, 75 s.

Práca sa zaoberá modelovaním rezíduí ARMA modelov použitím nelineárnych metód. Cieľom práce je zvýšiť kvalitu lineárnych modelov modelovaním ich rezíduí. Dôvodom modelovania rezíduí je vysoká miera využívania lineárnych modelov v praxi napriek ich nižšej kvalite. Ako metódy na modelovanie rezíduí využijeme neurónovú sieť a metódu oporných bodov. Dáta, v ktorých identifikujeme viacero režimov, budeme modelovať aj pomocou zhlukovej analýzy a SETAR modelov. V praktickej časti práce porovnáme kvalitu predikcií získaných týmito metódami na konkrétnych dátach.

Kľúčové slová: časové rady, ARMA model, rezíduá, neurónová sieť, metóda oporných bodov, zhluková analýza, SETAR model

Abstract

KUČMÍNOVÁ, Eva: Application of ARMA models and machine learning for time series modelling [Diploma Thesis], Comenius University in Bratislava, Faculty of Mathematics, Physics and Informatics, Department of Applied Mathematics and Statistics; Supervisor: doc. RNDr. Beáta Stehlíková, PhD., Bratislava, 2017, 75 p.

Thesis deals with the issue of modelling ARMA residuals using nonlinear methods. The aim of the work is increase of quality of ARMA models modelling their residuals. The reason of residuals modelling is the high rate of using linear models in practice despide their lower quality. Artificial neural network and support vector machines will be used for residuals modelling. Data possible to separate into the more regimes will be simulated also using cluster analysis and SETAR models. We will compare quality of predictions obtained using these methods on the real data.

Keywords: Time series, ARMA model, residuals, artificial neural network, support vector machine, cluster analysis, SETAR model

Obsah

Zoznam obrázkov

Zoznam tabuliek

Úvod

Modelovanie dát nelineárnymi modelmi je vo všeobecnosti presnejšie ako použitie lineárnych modelov, pretože dokáže identifikovať aj nelineárne závislosti v dátach. Napriek nižšej kvalite lineárnych modelov sa v praxi často uprednostňujú práve tieto modely. Dôvodom je ich nižšia komplikovanosť, ale hlavne lepšia interpretovateľnosť. Výsledky modelovania často interpretujeme aj ľuďom, ktorí nemajú hlbšie štatistické poznanie, no je nevyhnutné vedieť im zdôvodniť výsledky modelovania. Práve z tohto dôvodu sú lineárne modely vo všeobecnosti využívanejšie.

Cieľom našej diplomovej práce je zvýšiť kvalitu lineárnych ARMA modelov. O zvýšenie kvality modelovania sa pokúsime modelovaním rezíduí, ktoré vznikajú ako vedľajší výsledok lineárneho modelu. Takto vzniknuté rezíduá budeme modelovať metódami machine learningu. Chceme vytvoriť odhad predikovaných hodnôt, ktorý je presnejší ako lineárny model, no zároveň si zachováva vlastnosť interpretovateľnosti lineárneho modelu.

Práca je rozdelená do troch častí. V prvej časti stručne vysvetlíme lineárne ARMA modely, ktoré budeme využívať na modelovanie reálnych časových dát. Oboznámime so základnymi vlastnosťami, ktoré musí časový rad spĺňať na to, aby sa dal ARMA modelmi modelovať a popíšeme testy, ktoré budeme pri overovaní splnenia jednotlivých vlastností využívať.

V druhej časti práce opíšeme metódy machine learningu, ktoré budeme využívať na modelovanie rezíduí ARMA modelov. Vysvetlíme princípy, na ktorých sú tieto metódy založené, a uvedieme zjednodušené príklady na priblíženie postupu, na ktorom sú metódy založené.

Tretiu najrozsiahlejšiu časť práce venujeme aplikácii lineárnych a nelineárnych metód na reálne dáta. Zverejníme výsledky, ktoré dostaneme pri modelovaní jednotlivými metódami a porovnáme ich s výsledkami získanými inými metódami aplikovanými na dané dáta. V prípade, že bude metóda vhodná na modelovanie dát, zúžime okruh úloh, pre ktoré je daná metóda aplikovateľná. Ak bude metóda nevhodná na odhadovanie a predikovanie konkrétnych dát, pokúsime sa identifikovať, prečo sa dáta nepodarilo namodelovať. V prípade, že bude tento problém riešiteľný použitím inej metódy, pokúsime sa implementovať tento spôsob riešenia.

1 Časové rady a ARMA procesy

V tejto kapitole si najskôr zadefinujeme pojem časového radu a vysvetlíme pojmy s ním súvisiace. Následne predstavíme a vysvetlíme modely, ktoré slúžia na predikciu časových radov. Popíšeme postupy a testy slúžiace na určovanie rádu modelu. V celej tejto kapitole budeme vychádzať z publikácii [11] a [7].

1.1 Časový rad

Pod pojmom časový rad chápeme chronologicky usporiadaný súbor dát, ktorý zaznamenáva časový vývoj určitého javu. Väčšinou je jav zaznamenávaný v rovnakých časových intervaloch, vo všeobecnosti to tak však byť nemusí. V časovom vývoji dát môžeme pozorovať náhodnosť procesu, trend či periodicitu, v tomto prípade nazývanú sezónnosť. Týmto vlastnostiam sa budeme bližšie venovať v nasledujúcich častiach práce.
Časový rad možno chápať ako stochastický proces, ktorý pozorujeme v časoch *t, t-1, t-2, ...* a pozorované hodnoty označíme ako $x_{t-1}, x_{t-2}, x_{t-3}, ...$ Bolo by naivné predpokladať, že x_t vieme modelovať výhradne použitím dát z minulých časov. Model pre dáta musí obsahovať náhodnosť na postihnutie javov, ktorých vývoj nie je obsiahnutý v našom modeli, resp. javov, ktoré nemožno predvídať. Táto náhodnosť je reprezentovaná pomocou bieleho šumu μ. Pod pojmom biely šum budeme chápať proces spĺňajúci vlastnosti (1):

$$\begin{aligned} E(\mu_t) &= 0 \quad \forall t, \\ Var(\mu_t) &= \sigma^2 \quad \forall t, \\ cov(\mu_t, \mu_s) &= 0 \quad \forall t \neq s \end{aligned} \tag{1}$$

Lineárnu závislosť x_t od dát $x_{t-1}, x_{t-2}, x_{t-3}, ...$ a bieleho šumu $\mu_t, \mu_{t-1}, \mu_{t-2}, ...$ možno vo všeobecnosti vyjadriť rovnicou (2):

$$x_t = \delta + \sum_{j=1}^{J} \psi_j x_{t-j} + \sum_{k=0}^{K} \phi_k \mu_{t-k} \tag{2}$$

Takto definovaný proces nazývame *ARMA(J,K) procesom.* V rovnici (2) vyjadruje δ konštantu nezávislú v čase, členy $x_{t-1}, x_{t-2}, x_{t-3}, ...$ nazývame *autoregresné členy*, skrátene *AR členy* a $\mu_t, \mu_{t-1}, \mu_{t-2}, ...$ sú členy *moving average procesu*, resp. *MA členy.*

1.2 Autoregresné procesy

Pod pojmom autoregresný proces chápeme proces, v ktorom možno x_t zapísať len pomocou dát z predchádzajúcich časov $x_{t-1}, x_{t-2}, ...$ a bieleho šumu v čase t μ_t, prípadne môže byť proces ešte posunutý o konštantu nezávislú od času. Rád takto definovaného autoregresného procesu zodpovedá časovo najstarším dátam, ktoré sú v modeli použité:

$$x_t = \delta + \alpha_1 x_{t-1} + \alpha_2 x_{t-2} + ... + \alpha_k x_{t-k} + \beta\mu_t \tag{3}$$

Proces definovaný v rovnici (3) je autoregresný proces k-teho rádu, skrátene AR(k) proces.

1.3 Moving average procesy

Procesy s kĺzavým priemerom, lepšie známe pod pojmom *moving average procesy*, sú procesy definovateľné len pomocou bielych šumov z časov $t, t-1, t-2, ...$ Podobne ako v prípade autoregresných procesov definovaných v časti 1.2, proces definovaný predpisom

$$x_t = \delta + \mu_t + \beta_1\mu_{t-1} + \beta_2\mu_{t-2} + ... + \beta_k\mu_{t-k} \tag{4}$$

je moving average proces k-teho rádu, skrátene MA(k) proces.

1.4 Vlastnosti procesov

Pri definovaní ARMA procesov sa často stretneme s pojmom *lag*, označenie L. Týmto pojmom označujeme časový posun v dátach o obdobie dozadu, čo znamená, že $x_{t-1} = Lx_t$. Umocnenie lagu na n bude znamenať časový posun o n období dozadu. Takto vieme proces definovaný vzťahom (2) prepísať do tvaru (5):

$$x_t = \delta + \psi_1 Lx_t + \psi_2 L^2 x_t + \psi_3 L^3 x_t + ... + \phi_0\mu_t + \phi_1 L\mu_t + \phi_2 L^2\mu_t + \phi_3 L^3\mu_t + ... \tag{5}$$

Úpravami môžeme výraz (5) ekvivalentne zapísať v tvare (6):

$$(1 - \psi_1 L - \psi_2 L^2 - \psi_3 L^3 - ...)x_t = \delta + (\phi_0 + \phi_1 L + \phi_2 L^2 + \phi_3 L^3 + ...)\mu_t \tag{6}$$

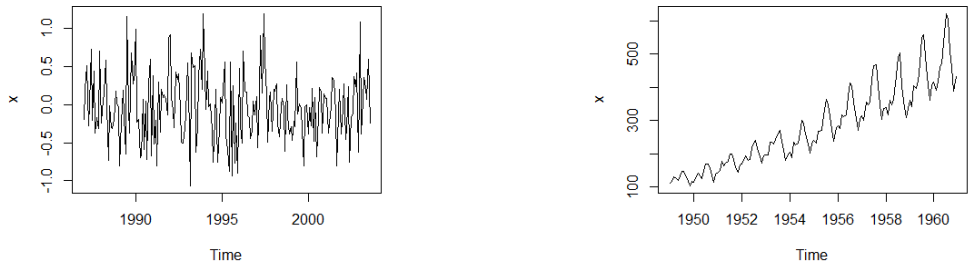
1.4.1 Stacionarita

Voľne povedané, v prípade stacionarity očakávame, že stacionárny proces sa nebude v časovom vývoji výraznejšie líšiť a jeho hodnoty budú oscilovať. Na Obr. 1 môžeme

pozorovať realizácie dvoch procesov. Na Obr. 1a) vidíme stacionárny proces oscilujúci okolo nuly. Matematicky tieto vlastnosti popíšeme dvoma vlastnosťami - proces je stacionárny, ak:

1. stredná hodnota je konštantná v čase
2. autokovariancia, tj. miera lineárnej závislosti medzi x_t a x_s pre $\forall t, s$ závisí len od vzdialenosti časov t a s

Ako dôsledok 2. vlastnosti zvolením $t = s$ dostaneme, že rozptyl procesu je konštatný. Na Obr. 1b) má časový rad rastúcu tendenciu, hovoríme, že má rastúci trend. Takýto časový rad nemá konštantú strednú hodnotu, okolo ktorej by osciloval.



(a) stacionárny proces[1] (b) nestacionárny proces[2]

Obr. 1: Stacionarita časového radu

Stacionarita bude nevyhnutným predpokladom pre modelovanie časového radu ARMA modelmi. Stacionárny proces môžeme dosiahnuť diferencovaním, viacnásobným diferencovaním, prípadne použitím inej funkcie, ktorá nám zabezpečí stacionaritu. V tomto prípade sa najčastejšie využíva logaritmizácia dát.

V nasledujúcej časti predstavíme metódu, pomocou ktorej môžeme jednoznačne určiť, či proces je, alebo nie je stacionárny. Z definície stacionarity a vlastností bieleho šumu definovanými rovnicami (1) v časti 1.1 jednoznačne vyplýva, že biely šum je stacionárny proces. Z vlastnosti linearity strednej hodnoty ($E(X + Y) = E(X) + E(Y)$) rovnako vyplýva, že ľubovoľný proces, ktorý možno zapísať ako súčet prírastov bieleho

[1] realizácia bieleho šumu s rozdelením N(0, 0,4) vygenerovaná programom R

[2] zdroj: Box, G. E. P., Jenkins, G. M. and Reinsel, G. C. (1976) Time Series Analysis, Forecasting and Control

šumu, bude stacionárny. Ekvivalenciu tohto tvrdenia dokázal v roku 1938 Herman Ole Andreas Wold. Dokázal, že každý stacionárny proces definovaný vzťahom (3) možno zapísať v tvare tzv. Woldovej reprezentácie:

$$x_t = \delta + \sum_{j=0}^{\infty} \psi_j \mu_{t-j} \tag{7}$$

Ekvivalentnou postačujúcou podmienkou stacionarity ARMA procesu (3) je podmienka vzťahujúca sa na polynóm lagov autoregresnej časti. Proces (3) je stacionárny práve vtedy, keď sú všetky korene polynómu $(1 - \psi_1 L - \psi_2 L^2 - \psi_3 L^3 - ...)$ v absolútnej hodnote väčšie ako 1. Ak proces obsahuje iba MA časť, tak je stacionárny.

1.4.2 Invertovateľnosť

O invertovateľnom procese hovoríme vtedy, ak proces vieme zapísať v tvare $AR(\infty)$ procesu. Našou úlohou je teda ľubovoľný ARMA proces definovaný vzťahom (2) zapísať v tvare:

$$x_t = \widehat{\delta} + \sum_{j=1}^{\infty} \psi_j x_{t-j} + \mu_t \tag{8}$$

Rovnako ako v prípade stacionarity sa dá odvodiť, že ARMA(J,K) proces definovaný vzťahom (2) je invertovateľný práve vtedy, keď sú korene polynómu $(\phi_0 + \phi_1 L + \phi_2 L^2 + \phi_3 L^3 + ...)$ z rovnice (6) v absolútnej hodnote väčšie ako 1. Z podmienky invertability rovnako vyplýva, že ľubovoľný AR(k) proces je vždy invertovateľný.

1.4.3 Autokorelačná funkcia

Autokorelačná funkcia, skrátene ACF, bude vyjadrovať závislosť x_t od dát $x_{t-1}, x_{t-2}, ..., x_{t-s}$. Túto závislosť možno vyjadriť pomocou korelácie x_t a x_{t-s}. Symbolicky môžeme závislosť zapísať nasledovnou rovnicou:

$$\rho(s) = cor(x_t, x_{t-s}) \quad \forall s = 1, 2, ..., \tag{9}$$

kde $\rho(s)$ vyjadruje autokorelačnú funkciu procesu. Ak $cov(x_t, x_{t-s})$ označíme ako $\gamma(s)$ platí pre $\rho(s)$ a $\gamma(s)$ nasledujúci vzťah:

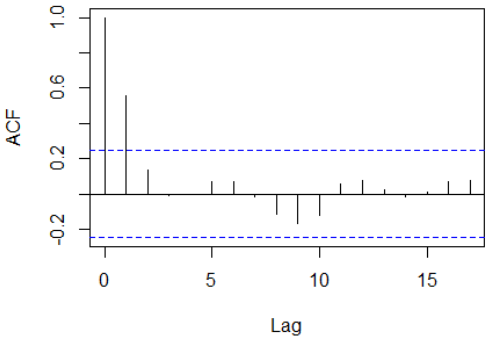
$$\rho(s) = \frac{\gamma(s)}{\gamma(0)} \tag{10}$$

V skutočnosti však nepočítame skutočnú autokorelačnú funkciu procesu, ale len jej asymptoticky nevychýlený odhad $\widehat{\rho(s)}$, ktorý nazývame *výberová autokorelačná funkcia*.

$$\widehat{\rho(s)} = \frac{\widehat{\gamma(s)}}{\widehat{\gamma(0)}}, \tag{11}$$

$$\widehat{\rho(s)} = \frac{\sqrt{\sum_{t=s+1}^{T}(x_t - \overline{x})(x_{t-s} - \overline{x})}}{\sqrt{\sum_{t=1}^{T}(x_t - \overline{x})^2}} \tag{12}$$

Na Obr. 2 môžeme vidieť výberovú autokorelačnú funkciu realizácii autoregresného procesu rádu 2, vygenerovanú v programe *R* funkciou *arima.sim* obsiahnutou v knižnici *graphicsQC*:



Obr. 2: Výberová autokorelačná funkcia

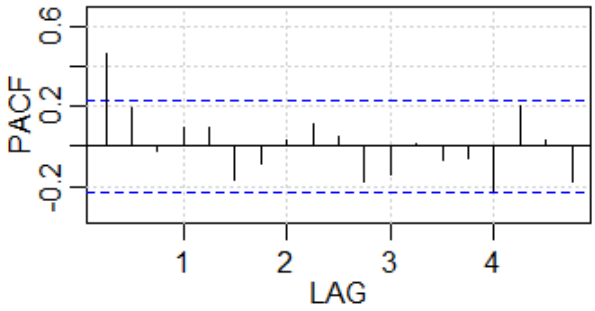
Z rovnice (10) jednoznačne vyplýva, že autokorelácia pre lag 0 je rovná 1. Z definície autokorelačnej funkcie vyplýva, že pre AR(k) proces budú hodnoty k+1, k+2, ... autokorelačnej funkcie nulové. Očakávame teda, že hodnoty výberovej autokorelačnej funkcie budú nesignifikantné.

1.4.4 Parciálna autokorelačná funkcia

Parciálna autokorelačná funkcia vyjadruje podmienenú koreláciu medzi dátami x_t a x_{t-j} za podmienky $x_{t-1}, x_{t-2}, ..., x_{t-j+1}$. Vychádzajúc z [20] formálne zapísaná parciálna autokorelácia pre proces definovaný rovnicou (2) má tvar:

$$\rho(j) = \frac{cov(x_t, x_{t-j}|x_{t-1}, x_{t-2}, ..., x_{t-j+1})}{\sqrt{Var(x_t|x_{t-1}, x_{t-2}, ..., x_{t-j+1})Var(x_{t-j}|x_{t-1}, x_{t-2}, ..., x_{t-j+1})}} \tag{13}$$

Rovnako ako v prípade autokorelačnej funkcie, ani v tomto prípade nepočítame skutočné hodnoty parciálnej autokorelačnej funkcie, ale len ich odhady.



Obr. 3: Výberová parciálna autokorelačná funkcia

Hodnoty k+1, k+2, ... parciálnej autokorelačnej funkcie MA(k) procesu budú nulové. Z tohto dôvodu očakávame nesignifikantné hodnoty výberovej parciálnej autokorelačnej funkcie.

Je dôležité poznamenať, že ACF a PACF slúžia len na samostatné určovanie rádu AR a MA procesov. Odhady získané z týchto funkcií nemožno kombinovať, a teda ich nemožno použiť na odhad rádu ARMA procesu. V prípade ARMA procesu môžu slúžiť len na približný odhad, ktorý však môže byť zlý. V tomto prípade je nutné testovať viacero typov ARMA procesov.

1.4.5 Jednotkový koreň

Podmienka stacionarity definovaná v časti 1.4.1 môže viesť aj k hraničnej situácii, kedy je niektorý z koreňov polynómu $(1-\psi_1 L-\psi_2 L^2-\psi_3 L^3-...)$ rovný 1. Predpokladajme, že násobnosť jednotkového koreňa je 1. To znamená, že polynóm $(1-\psi_1 L-\psi_2 L^2-\psi_3 L^3-...)$ možno zapísať v tvare:

$$(1-L)(1-\widetilde{\psi_1}L-\widetilde{\psi_2}L^2-...-\widetilde{\psi_{k-1}}L^{k-1}-...) \tag{14}$$

Ak teda uvažujeme model pre diferencie x_t, tj.$\Delta x_t = x_t - x_{t-1}$, AR časť procesu má tvar

$$(1-\widetilde{\psi_1}L-\widetilde{\psi_2}L^2-...-\widetilde{\psi_{k-1}}L^{k-1}-...)\Delta x_t \tag{15}$$

Keďže bola násobnosť jednotkového koreňa 1, takto definovaný proces už je stacionárny. Diferencovanie procesu teda odstraňuje existujúci jednotkový koreň. Vo všeobecnosti platí, že ak proces obsahuje k-násobný jednotkový koreň, tak k-násobné diferencovanie

procesu tento jednotkový koreň odstráni. Naviac, ak zvyšné korene procesu sú mimo jednotkového kruhu, tak proces je po k-násobnom diferencovaní už stacionárny.
Na testovanie jednotkového koreňa budeme využívať funkciu *ur.df* definovanú v knižnici *urca* [21].V prípade, že ARMA procesom modelujeme diferencie, proces nazývame ARIMA(p, d, q) procesom, kde p označuje stupeň AR členov, q označuje pôvodný stupeň MA členov a d označuje stupeň diferencovania procesu.

1.4.6 Ljung - Boxov test závislosti

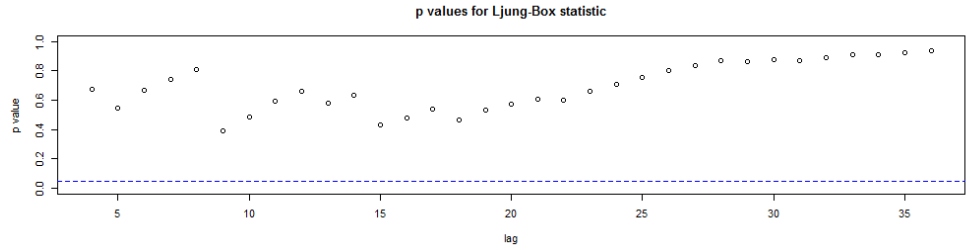
V roku 1978 štatistici George E. P. Box a Greta M. Ljung vytvorili test na testovanie kvality modelu fitujúceho časový rad. Podstatou tohto testu je testovanie závislostí jednotlivých lagov. Testovacia štatistika má tvar definovaný vzťahom (16) a má asymptotické χ^2_K rozdelenie. V prípade, že testujeme rezíduá ARMA(p, q) procesu, testovacia štatistika má opäť tvar (16) a asymptotické rozdelenie $\chi^2_{K-(p+q)}$. V prípade, že v rezíduách nie je identifikovaná signifikantná autokorelácia, model týmto testom prešiel. [2, str. 351]

$$Q = T(T+2)\sum_{k=1}^{K}\frac{\widehat{\rho(k)}^2}{(T-k)}, \tag{16}$$

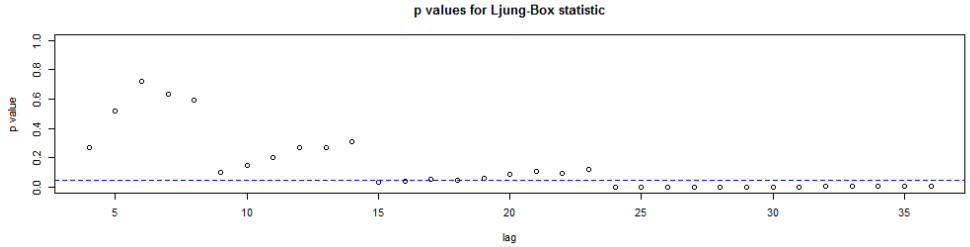
kde T je veľkosť testovanej vzorky, K je počet autokorelačných lagov a $\widehat{\rho(k)}$ je odhad korelácie definovaný vzťahom (17).

$$\widehat{\rho(k)} = \frac{\frac{1}{n-1}\sum_{i=1}^{n}(x_t - \widehat{E(x)})(x_{t+k} - \widehat{E(x)})}{\widehat{Var(x)}} \tag{17}$$

V rovnici (17) $\widehat{E(x)}$ a $\widehat{Var(x)}$ vyjadrujú odhad pre strednú hodnotu a disperziu časového radu.
V programe R budeme na testovanie autokorelácie modelov používať test zabudovaný vo funkcii *sarima*, ktorá je obsiahnutá v knižnici *astsa*. Túto funkciu využívame na odhad parametrov procesu. Ako ďalší výstup funkcia vypočíta testovaciu štatistiku Q a následne p-value pre všetky testované autokorelačné lagy L odhadnutého procesu. Na Obr.4 môžeme vidieť príklad dvoch modelov testovaných na autokoreláciu Ljung-Boxovým testom závislostí.



(a) model vyhovujúci Ljung-Boxovmu testu



(b) model nevyhovujúci Ljung-Boxovmu testu

Obr. 4: Ljung - Box test

Z Obr. 4 vieme vyhodnotiť kvalitu testovaných modelov. Prvý z modelov testom prešiel, druhý bol zamietnutý na základe autokorelácie identifikovanej pre väčšie lagy.

2 Machine learning metódy

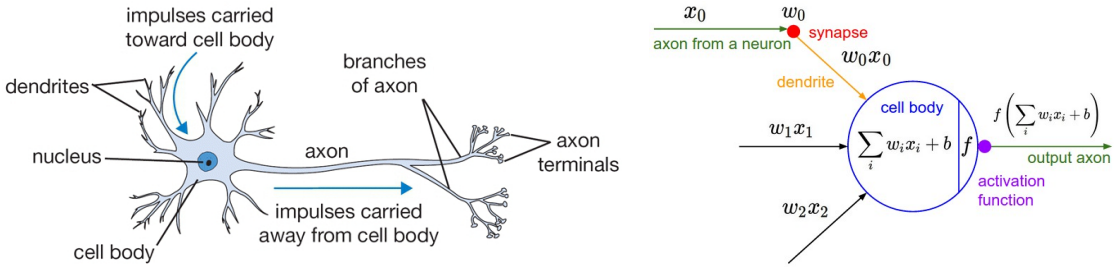
V tejto kapitole predstavíme Machine learning metódy, ktoré budeme využívať na modelovanie rezíduí, ktoré vzniknú použitím lineárnych ARMA modelov. Machine learning metódy použijeme na zvýšenie presnosti modelu. Presnosť predikcií ARMA modelu v závere porovnáme s presnosťou predikcií doplnených o namodelované rezíduá a vyhodnotíme najpresnejšiu metódu pre danú skupinu dát.

2.1 Neurónová sieť

Od čias skonštruovania prvého počítača po súčasnosť je neustála snaha o rozširovanie funkcií počítačov, ktoré by človeku uľahčili život. Snahu pretransformovať ľudské myslenie a konanie do reči programovacích kódov pozorujeme takmer vo všetkých oblastiach života. Stroje nahradili človeka v priemyselnej a strojárskej výrobe, dokážu poraziť človeka v hre ako je napríklad šach či karty, alebo dokonca dokážu riadiť autá a lietadlá.

Procesy, ktoré sú pre ľudský mozog prirodzené, ako je učenie sa zo skúsenosti, premýšľanie, vyhodnocovanie situácii či dokonca predvídanie budúcnosti, sa vo forme kódu snažíme naučiť aj stroj. Ľudský mozog je v tomto smere dokonalý a doposiaľ neprekonaný. V snahe zdokonaľovať stroje sa vedci snažia napodobniť ľudský mozog do najmenších detailov.

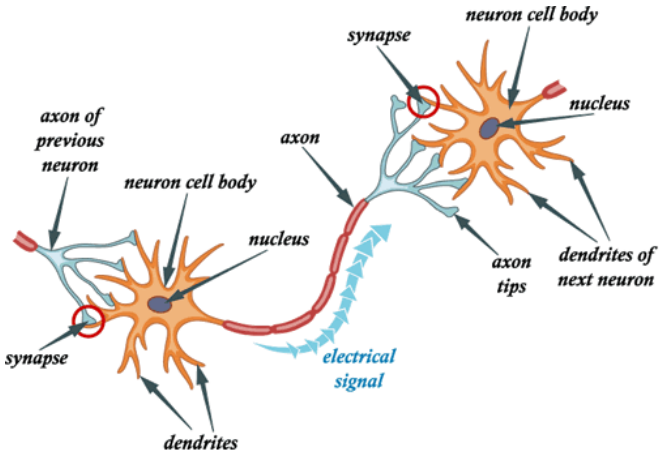
Základnou jednotkou ľudského mozgu je neurón. Miesto, kde sa neuróny spájajú, sa nazýva synapsia. Slúži na prevažne jednosmerný prenos vzruchov medzi bunkami. Tento proces prenosu informácie medzi neurónmi sa snaží zachytiť a replikovať umelá neurónová sieť. Na Obr. 5 môžeme vidieť paralely medzi biologickou bunkou a matematickým modelom zjednodušene popisujúcim tento proces. [4]



(a) Biologický neurón (b) Matematický model neurónu

Obr. 5: Neurónová sieť - biologická vs. umelá ([4])

Vstupným bodom neurónu sú dendrity, ktoré prijímajú vstupné signály. Tie sú následne bunkou spracované a odoslané von cez výstupný kanál - axón. Axón každej bunky je zakončený synaptickými vláknami, ktoré predávajú výstupnú informáciu dendritom inej bunky. Tento proces je zobrazený na Obr. 6.



Obr. 6: Biologický proces prenosu informácie [6]

Každá synapsa má svoju vlastnú synaptickú váhu, ktorá vyjadruje mieru prenosu medzi bunkami. Synaptické váhy tvoria základ pamäti ako takej a sú cieľom modelovania umelou neurónovou sieťou. Tieto synaptické váhy sú učenlivé a v priebehu svojej existencie si upravujú mieru pôsobenia jednej bunky na druhú vychádzajúc zo skúsenosti. V prípade kladnej synaptickej váhy pôsobí vzruch vzrušivo, záporná synaptická váha má utlmujúci efekt. [4]

Matematický model neurónu zobrazený na Obr. 5b) pracuje na princípe váženia vstupov synaptickými váhami a použitia aktivačnej funkcie. V prípade, že predpokladáme R vstupov do neurónu, model možno zapísať nasledujúcou rovnicou:

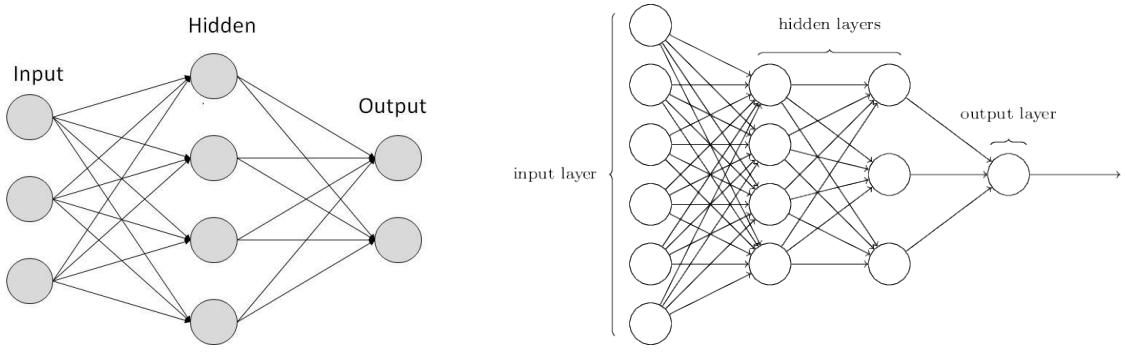
$$Y = f(\sum_{i=0}^{R} w_i X_i + b), \tag{18}$$

kde X a Y označujú vstupné a výstupné informácie neurónu,b je konštanta nazývaná aj *bias*, f je aktivačná funkcia a w_i sú synaptické váhy.

Vytváranie umelej neurónovej siete vedie ku konštrukcii modelu, ktorý si zo skúsenosti (tréningovej vzorky) sám určí optimálne váhy pri zadanej aktivačnej funkcii. V práci budeme pracovať s nasledujúcimi troma typmi aktivačných funkcií:

- lineárna funkcia: $f(u) = u$
- hyperbolický tangens $f(u) = tanh(u/2) = \frac{1-e^{-u}}{1+e^{-u}}$
- logistická funkcia $f(u) = \frac{1}{1+e^{-\beta u}}$

Umelá neurónová sieť predpokladá usporiadanie neurónov do viacerých vrstiev. Neurónová sieť vždy obsahuje vstupnú vrstvu, do ktorej vstupujú nami zadané vstupy, a výstupnú vrstvu, ktorej výsledkom sú výstupy generované neurónovou sieťou. Medzi týmito vrstvami sa môže ďalej nachádzať jedna alebo viac skrytých vrstiev.
Na Obr. 7a) môžeme vidieť neurónovú sieť s jednou skrytou vrstvou a dvoma výstupnými neurónmi. Umelá neurónová sieť s dvoma skrytými vrstvami a jedným výstupným neurónom je zobrazená na Obr.7b).



(a) Jedna skrytá vrstva [1] (b) Dve skryté vrstvy [30]

Obr. 7: Umelá neurónová sieť

Vo všeobecnosti môže mať umelá neurónová sieť niekoľko skrytých vrstiev. Neurónovú sieť s k výstupmi a s P počtom skrytých vstiev vieme zapísať nasledovným vzťa-

hom:

$$Y_k = g(\alpha_{0k} + \sum_{j=1}^{P} \alpha_{jk} f(w_{0j} + \sum_{i=1}^{R} w_{ij} X_i)), \tag{19}$$

kde:

- Y_k je výstup k-tého výstupného neurónu, pričom k=1, ..., S a S je počet výstupných neurónov
- X_i je i-ty vstupný neurón a i=1, ..., R a R je počet vstupných neurónov
- w_{ij} a α_{jk} sú synaptické váhy
- α_{0k} a w_{0j} sú konštanty nazývané *bias*
- f a g sú aktivačné funkcie.

Umelé neurónové siete definované vyššie patria medzi *dopredné umelé neurónové siete*. Ide o taký typ siete, ktorá má smer väzieb iba od vstupov smerom k výstupom. Okrem dopredných sietí poznáme aj *rekurentné neurónové siete*. Tento typ siete využíva aj spätné väzby, pričom tento typ siete je výpočtovo náročnejší. Graficky sme tieto typy sietí porovnali na Obr.8. [1]

(a) Dopredná neurónová sieť (b) Rekurentná neurónová sieť

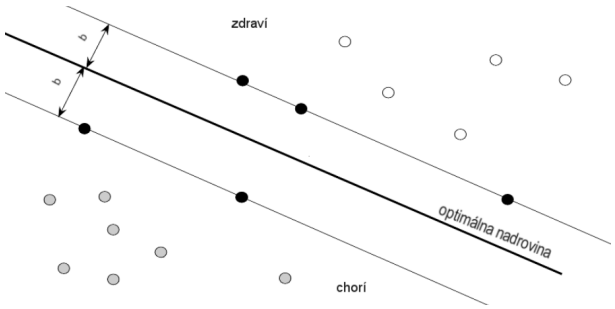
Obr. 8: Porovnanie doprednej a rekurentnej neurónovej siete [1]

2.2 Metóda oporných bodov

Táto metóda, lepšie známa pod anglickým ekvivalentom *Support vector machine*, sa v oblasti machine learningu využíva najmä na klasifikáciu a regresiu. Po prvý raz bola

definovaná v roku 1992 vedeckým tímom sústredeným okolo Vladimira Vapnika [31]. V tejto časti práce sme sa inšpirovali najmä publikáciami [3] a [15].

Vstupom pre metódu oporných bodov sú charakteristiky x_i pre $i = 1, ..., N$, ktoré sa transformujú do priestoru H vyššej dimenzie nelineárnou transformáciou $\phi(x_i)$, kde $\phi : X \to H$ a N je počet objektov. V tomto priestore sa potom hľadá oddeľovacia nadrovina $\langle w; \phi(x) \rangle + b$ a parametre w, b sú predmetom optimalizácie. Grafické zobrazenie tohto problému môžeme vidieť na Obr. 9.



Obr. 9: Metóda oporných bodov [3]

Na Obr. 9 vidíme aj tzv. *prahovú vzdialenosť b*. Tento parameter vyjadruje vzdialenosť najbližších objektov tréningovej vzorky od optimálnej nadroviny. Pre túto metódu bude mať zmysel definovať tzv. *rozhodovaciu funkciu*, ktorú odhadneme na tréningovej vzorke dát, ktoré nazývame *oporné vektory*.

Použitie metódy oporných bodov na klasifikáciu dát rieši optimalizačnú úlohu (20).

$$\begin{aligned} \min_{w,b,\zeta} \quad & \frac{1}{2} w^T w + C \sum_{i=1}^{n} \zeta_i \\ & \text{st.} \qquad y_i(w^T \phi(x_i) + b) \geq 1 - \zeta_i, \\ & \zeta_i \geq 0, \qquad i = 1, ..., n. \end{aligned} \tag{20}$$

S cieľom zjednodušiť výpočet možno namiesto úlohy (20) riešiť duálnu úlohu definovanú optimalizačnou úlohou (21).

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \alpha^T Q \alpha - e^T \alpha \\ & \text{st.} \qquad y^T \alpha = 0, \\ & 0 \geq \alpha_i \geq C, \qquad i = 1, ..., n, \end{aligned} \tag{21}$$

kde:

- e je jednotkový vektor,
- $y \in \{-1, 1\}^n$,
- $C>0$ je ohraničenie na chybu modelu,
- Q je pozitívne semidefinitná nxn matica, pričom jej zložky sú definované ako $G_{ij} = y_i y_j K(x_i, x_j)$ a $K(x_i, x_j) = \langle \phi(x_i); \phi(x_j) \rangle$ je funkcia nazývaná *kernel funkcia*, ktorú priblížime v ďalšej časti.

Rozhodovacia funkcia pre úlohy (20) a (21) má tvar (22).

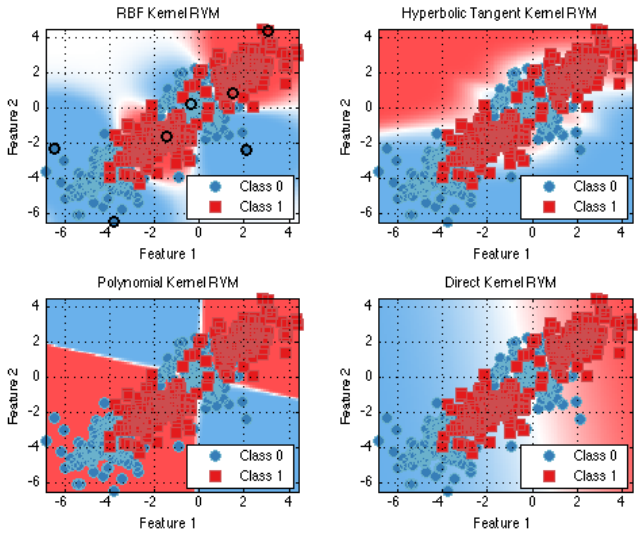
$$sgn(\sum_{i=}^{n} y_i \alpha_i K(x_i, x) + b) \tag{22}$$

Kernel funkcia je funkcia, ktorá určuje zložitosť klasifikačnej metódy. Pre použitie metód oporných bodov nám stačí poznať funkciu $K(x_i, x_j) = \langle \phi(x_i); \phi(x_j) \rangle$ a nie je nutné poznať priamo transformáciu $\phi(x_i)$. V našej práci budeme pracovať s kernel funkciami definovanými v Tabuľke 1. Volené parametre týchto funkcií majú prednastavené hodnoty pre funkciu *svm*, ktorá je súčasťou knižnice *e1071*. My budeme hodnotu týchto parametrov optimalizovať použitím funkcie *tune.svm*, pričom sa budeme snažiť minimalizovať chybu odhadu modelu.

Kernel funkcia	Predpis funkcie	Volené parametre
lineárna	$\langle x, x' \rangle$	žiadne
polynomiálna	$\gamma(\langle x, x' \rangle + c_0)^d$	γ, d, c_0
Gaussovská (RBF)	$\exp(-\gamma \|x - x'\|_2^2)$	γ
Hyperbolický tangens (Sigmoid)	$\tanh(\gamma \langle x, x' \rangle + c_0)$	γ, c_0

Tabuľka 1: Typy kernel funkcií [15]

Príklad klasifikácie použitím rôznych typov kernel funkcie zobrazujeme na Obr. 10. Môžeme vidieť kvalitu metódy pri použití rôznych typov kernel funkcie, pričom skutočná klasifikácia dát je vyznačená farbou zobrazenia samotného bodu a klasifikácia získaná metódou spolu s prahmi pre klasifikáciu sú farebne zobrazené na pozadí.



Obr. 10: SVM pre rôzne typy kernel funkcie [23]

Druhým spôsobom využitia metódy oporných bodov je regresia. V našej práci sa obmedzíme na $\epsilon - regresiu$, ktorá rieši optimalizačnú úlohu (23).

$$\begin{aligned} \min_{w,b,\zeta,\zeta^*} \quad & \frac{1}{2}w^T w + C\sum_{i=1}^{n}(\zeta_i^* + \zeta_i) \\ \text{st.} \quad & y_i - w^T\phi(x_i) - b \leq \epsilon + \zeta_i^*, \\ & w^T\phi(x_i) + b - y_i \leq \epsilon + \zeta_i, \\ & \zeta_i, \zeta_i^* \geq 0, \qquad i = 1, ..., n. \end{aligned} \tag{23}$$

Duálna úloha k primárnej úlohe (23) je definovaná predpisom (24). [28]

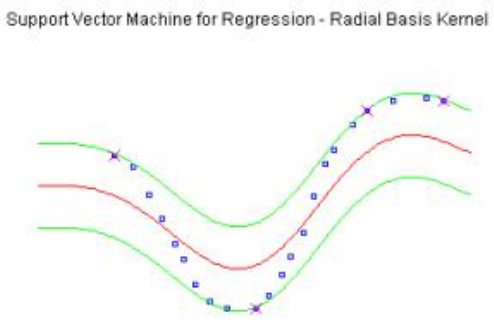
$$\begin{aligned} \min_{\alpha,\alpha^*} \quad & \frac{1}{2}(\alpha - \alpha^*)^T Q(\alpha - \alpha^*) + \epsilon e^T(\alpha + \alpha^*) - y^T(\alpha - \alpha^*) \\ \text{st.} \quad & e^T(\alpha - \alpha^*) = 0, \\ & 0 \geq \alpha_i, \alpha_i^* \geq C, \qquad i = 1, ..., n, \end{aligned} \tag{24}$$

pričom definícia premenných je totožná s premennými v úlohach (20) a (21). Rozhodovacia funkcia v prípade úlohy (23) a (24)má tvar(25).

$$\sum_{i=}^{n}(\alpha - \alpha^*)K(x_i, x) + b \tag{25}$$

V prípade regresie stanovíme aj hodnotu parametra ϵ, ktorý vyjadruje tolerovateľnú mieru chyby odhadu pre tréningovú vzorku dát. Väčšia hodnota parametra znamená

toleranciu väčšej chyby modelu, a zároveň použitie menšieho počtu oporných vektorov. Grafické zobrazenie tejto metódy môžeme vidieť na Obr. 11.



Obr. 11: Metóda oporných bodov pre ϵ regresiu [27]

Iné typy klasifikačných a regresných úloh sú definované vo vignette knižnice *e1071* [17].

2.3 Zhluková analýza

V tejto časti predstavíme niektoré z metód nehierarchického zhlukovania. Táto podkapitola je spracovaná podľa učebných textov [9] a [18]. Vstupom pre zhlukovú analýzu je matica M, ktorá obsahuje n počet čŕt m objektov:

$$M = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1j} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2j} & \dots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{i1} & x_{i2} & \dots & x_{ij} & \dots & x_{in} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \dots & x_{mj} & \dots & x_{mn} \end{bmatrix} \tag{26}$$

Každý riadok matice M tak obsahuje vektor čŕt i-tého objektu. Na základe podobnosti čŕt jednotlivých objektov sú zhlukovou analýzou objekty rozdelené do k zhlukov. Každá z metód zhlukovej analýzy pracuje na inom princípe delenia. V práci pracujeme s časovými radmi, kde dáta neobsahujú črty. V tomto prípade sme za črty dát označili jednotlivé lagy, čiže dáta v čase t sú charakterizované dátami z časov *t-1, t-2, ...* .

2.3.1 K-means metóda

V tejto časti opíšeme jednu z metód zhlukovania, pričom budeme vychádzať z [18]. Cieľom tejto metódy je nájsť centroidy pre zadaný počet zhlukov, ktoré minimalizujú vzdialenosť bodov v danom zhluku od jeho centroidu. Centroidy tak tvoria ťažiská zhlukov. V prípade, že dáta chceme rozdeliť do K zhlukov a centroidy označíme ako μ_i, kde $i \in \{1, ..., K\}$, úlohu, ktorú rieši táto metóda môžeme zapísať nasledovne:

$$\arg\min_{c} \sum_{i=1}^{K} \sum_{x \in c_i} d(x, \mu_i) = \arg\min_{c} \sum_{i=1}^{K} \sum_{x \in c_i} \parallel x - \mu_i \parallel_2^2 \tag{27}$$

kde c_i je množina bodov prislúchajúcich i-tému zhluku a $d(x, \mu_i)$ je euklidovská vzdialenosť týchto bodov od prislúchajúceho centroidu $d(x, \mu_i) = \parallel x - \mu_i \parallel_2^2$.
V ďalšej časti predstavíme si najčastejšie používaný algoritmus pre k-means zhlukovanie - Lloydov algoritmus a jeho spojitú modifikáciu - Forgyho algoritmus. Ďalšie používané algoritmy sú opísané v [18].

- **Lloydov algoritmus(1957)**

  1. Zvolíme počet zhlukov a počiatočnú polohu centroidov.
  2. Každé pozorovanie priradíme k centroidu, ktorý je k nemu najbližšie:

  $$c_i = \{j : d(x, \mu_i) \leq d(x, \mu_l) \qquad l \neq i, j = 1, ..., n\} \tag{28}$$

  3. Pre vytvorené zhluky vypočítame nové centroidy zodpovedajúce priemeru bodov v danom zhluku:

  $$\mu_i = \frac{1}{|c_i|} \sum_{j \in c_i} x_j, \qquad \forall i, \tag{29}$$

  kde $|c_i|$ je počet pozorovaní v danom zhluku.

  4. Pre novovytvorené centroidy opakujeme krok 2. Algoritmus opakuje kroky 2. a 3. až kým novovytvorené centroidy nezodpovedajú predchádzajúcim centroidom.

- **Forgyho algoritmus(1965)**

  Tento algoritmus je totožný s postupom v prípade Lloydovho algoritmu. Jediným rozdielom je, že Forgyho algoritmus uvažuje spojité rozdelenie dát, zatiaľ

čo Lloydov algoritmus pracuje s dátami z diskrétneho rozdelenia. Spojitá verzia minimalizačnej úlohy (27) má v tomto prípade tvar:

$$\arg\min_{c} \sum_{i=1}^{K} \int_{x \in c_i} \rho(x) d(x, \mu_i) dx, \tag{30}$$

kde $\rho(x)$ je funkcia hustoty pravdepodobnosti pre x.

2.3.2 K-medoids metóda

Táto metóda už nepoužíva ťažiská ako centroidy zhlukov, ale konkrétne body = dáta, ktoré minimalizujú vzdialenosti vo vnútri zhlukov. Dáta, ktoré takto zvolíme, nazývame *medoidy*. Hlavnou výhodou tejto metódy oproti metóde k-means je práve získaný medoid. Tento medoid a jeho črty sú akokeby reálne reprezentatívne hodnoty zhluku. Medzi základné algoritmy tejto metódy, s ktorým budeme pracovať pri analýze časových radov, patrí *Partitioning Around Medoids (PAM) algoritmus.*[9]

PAM algoritmus

1. Z počiatočnej množiny dát vyberieme k bodov, ktoré označíme ako medoidy: m_1, m_2, ..., m_k. Dáta, ktoré nie sú medoidmi, označme d_1, d_2, ..., d_{n-k}.

2. Všetky zvyšné dáta, ktoré nie sú medoidmi, priradíme do zhluku najbližšieho medoidu. Vypočítame vzdialenosti bodov od medoidov pre každý zhluk.

3. Následne, za medoid označíme niektoré z dát d_1, d_2, ..., d_{n-k}. Získali sme tak novú množinu medoidov.

4. Dáta opäť rozdelíme do zhlukov podľa ich vzdialenosti od novovybraných medoidov.

5. Cieľom je nájsť medoidy minimalizujúce vzdialenosť dát im prislúchajúcim.

2.3.3 Zhlukovanie založené na normálnom modeli

Poslednou použitou zhlukovacou metódou je zhlukovanie založené na normálnom modeli, pričom sa inšpirujeme [10]. Táto metóda predpokladá, že p-rozmerné vektory čŕt pozorovaní každého zhluku *(j= 1, ..., k)* majú viacrozmerné normálne rozdelenie

$N_p(\mu_j, \Sigma_j)$.

Pre vysvetlenie optimalizačného problému musíme najskôr definovať symboliku. Označme symbolom γ zatriedenie n objektov do k zhlukov, teda γ je vektor dĺžky n a obsahuje hodnoty *1*,...,k určujúce zhluk, do ktorého objekt patrí. x_i je vektor čŕt i-tého objektu, pričom $i = 1, ..., n$. Hustotu rozdelenia $N_p(\mu_j, \Sigma_j)$ označme $f(.; \mu_{\gamma_i}, \Sigma_{\gamma_i})$

Cieľom je nájsť optimálne zatriedenie objektov $\widehat{\gamma}$. Toto zatriedenie nájdeme maximalizáciou vierohodnostnej funkcie

$$L(\theta|x_1, x_2, ..., x_n) = \prod_{i=1}^{n} f(x_i; \mu_{\gamma_i}, \Sigma_{\gamma_i}) \tag{31}$$

Úlohu (31) vieme riešiť pomocou genetických algoritmov alebo *Expectation maximalization (EM) algoritmu.*

2.4 SETAR modely

V tejto časti definujeme režimové autoregresné modely známe pod názvom *TAR modely (Treshold Autoregressive models)*, ktoré prvý raz predstavil a definoval Tong v roku 1983. Vychádzať budeme najmä z [8].

TAR modely definujú viacero autoregresných modelov pre jeden časový rad pričom prahová hodnota *(treshold)* určuje zaradenie dát do príslušného autoregresného modelu. TAR modely sú po častiach lineárne. Prahové delenie časového radu do režimov delí dáta do k-režimov s lokálnymi lineárnymi modelmi. Toto prahové delenie robí model nelineárnymi v prípade použitia aspoň dvoch režimov.

Špeciálnou triedou TAR modelov sú SETAR modely *(Self Exciting Threshold Autoregressive models)*. SETAR procesy rádu p, ktoré budeme v našej práci analyzovať, možno definovať pomocou rovnice (32).

$$X_t = \begin{cases} \delta_{10} + \sum_{j=1}^{p} \psi_{1j} x_{t-j} + \mu_t & X_{t-d} \leq r \\ \delta_{20} + \sum_{j=1}^{p} \psi_{2j} x_{t-j} + \mu_t & X_{t-d} > r \end{cases} \tag{32}$$

pričom časový rad je definovaný po častiach AR(p) modelmi, r vyjadruje prahovú hodnotu SETAR modelu a d je časové oneskorenie.

3 Praktická časť

V tejto kapitole priblížime dáta, ktoré následne použijeme na modelovanie lineárnymi a nelineárnymi modelmi. Odprezentujeme výsledky získané aplikovaním týchto metód na časové rady. Praktické výpočty spolu s grafickými zobrazeniami sme získali použitím štatistického softvéru R. Niektoré časti zdrojového kódu sú obsahom príloh tejto práce. Pri modelovaní sme sa rozhodli pracovať s viacerými skupinami dát z rôznych oblastí, aby sme zvýraznili široké možnosti uplatnenia postupov prezentovaných v tejto práci. Rovnako chceme zdôrazniť, že žiadnu z použitých metód nemožno považovať za lepšiu či horšiu vo všeobecnosti, ale každá z metód môže pre určité dáta priniesť kvalitnejšie a presnejšie výsledky ako ostatné.

Na začiatku modelovania si dáta rozdelíme na tréningovú, validačnú a testovaciu časť. Na tréningovej časti dát odhadneme ARMA model použitím funkcií knižnice *fArma* a *astsa* [33, 25]. aj nelineárny model dát. Spravíme predikcie na validačnej časti dát pre viaceré typy metód a vyberieme metódu s najkvalitnejšími predikciami. Následne spravíme predikcie na validačných a testovacích dátach a vyhodnotíme kvalitu danej metódy. Pri procese deľby dát a ich následného využitia na predikovanie sme vychádzali z online kurzu *Practical machine learning* [22]. Zdrojový kód použitý na modelovanie rozdelením dát do troch skupín pri použití neurónovej siete na tvorbu predikcií je k dispozícii v Prílohe A. Zdrojový kód na použitie metódy oporných bodov na toto modelovanie sme obsiahli v Prílohe B. Kvalitu modelov budeme merať veľkosťou chyby predikcií, ktorú má použitý model. Riadiť sa budeme chybou MSE definovanou vzťahom (33):

$$\begin{aligned} RSS &= \sum_{i=1}^{n} (\hat{X}_i - X_i)^2 \\ MSE &= \frac{RSS}{n} \\ RMSE &= \sqrt{MSE}, \end{aligned} \tag{33}$$

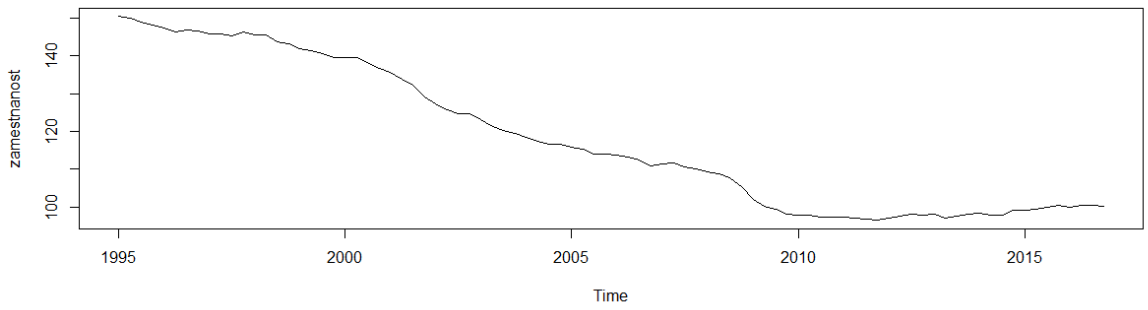
kde X_i sú reálne dáta a $\hat{X}_i$ je odhad získaný metódou SVM. Pri voľbe optimálnych vstupných parametrov pre metódu oporných bodov použijeme funkciu *tune.svm* obsiahnutú v knižnici *e1071* [17]. Túto metódu optimalizácie vstupných parametrov sme prevzali z [29].

3.1 Zamestnanosť v oblasti manufaktúrnej výroby spotrebných tovarov

Nasledujúce dáta zaznamenávajú počet ľudí v tisícoch zamestnaných v odvetví manufaktúrnej výroby spotrebných tovarov v USA. Dáta sme čerpali z knižnice *Quandl* softvéru *R* [16]. Od začiatku pozorovaní v roku 1995 až do hospodárskej krízy v rokoch 2008-2009 pozorujeme prudký prepad zamestnanosti v tomto odvetví. Následne pozorujeme ustálenie situácie.

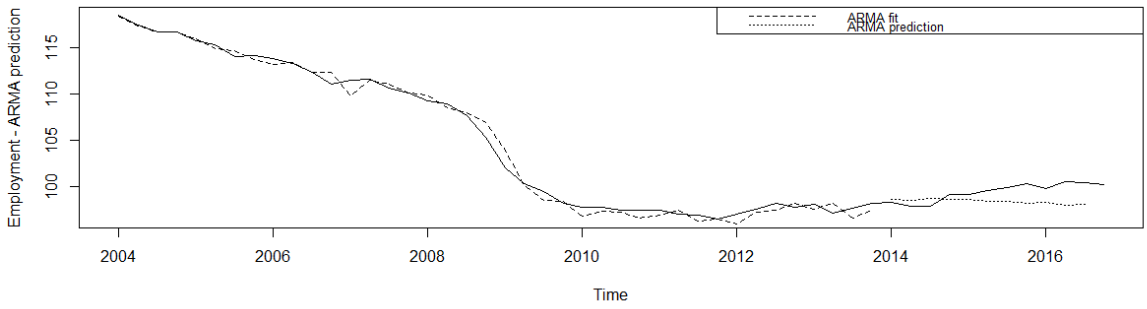
Dôvodom počiatočného prepadu zamestnanosti je uzatvorenie severoamerickej dohody NAFTA medzi USA, Kanadou a Mexikom o voľnom obchode medzi týmito štátmi. Táto dohoda mala za následok prepad zamestnanosti v pozorovovanom segmente. Následná kríza v rokoch 2008-2009 a pohotové opatrenia vtedajšej vlády na zmiernenie dopadu krízy v ekonomike mali za následok zastavenie poklesu zamestnanosti v tomto segmente.

Našim cieľom je modelovať vývoj zamestnanosti v tomto segmente v prípade platnosti tejto zmluvy aj naďalej. Z tohto dôvodu teda možno považovať vplyv dát spred ustálenia situácie v segmente za málo signifikantný a mohol by spôsobiť nepresnosť výsledkov. Preto sme sa rozhodli zanedbať dáta spred roku 2004 a tým minimalizovať vplyv uzatvorenia dohody NAFTA na aktuálny vývoj. Ešte lepšie predikcie by sme mohli dostať zohľadením iba pokrízovej zamestnanosti v danom segmente, čo by však v prípade kvartálnych dát bola príliš malá pozorovaná vzorka. Obr. 12 zobrazuje grafický vývoj zamestnanosti v pozorovanom segmente.

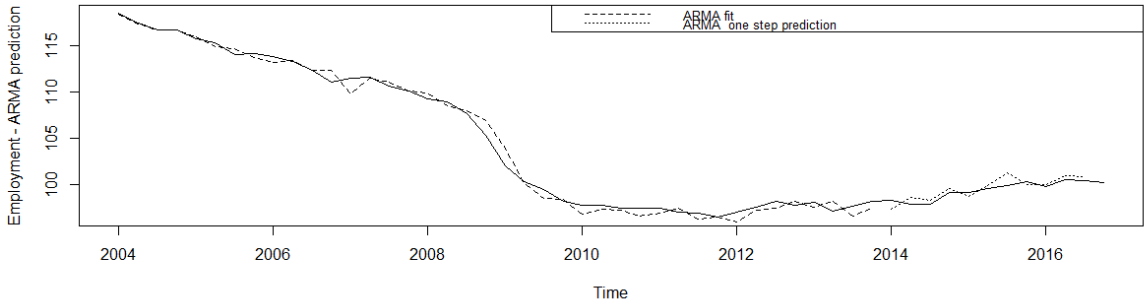


Obr. 12: Vývoj zamestnanosti v oblasti manufaktúrnej výroby spotrebných tovarov v rokoch 1995 - 2016

Dáta sme namodelovali sezónnym ARIMA modelom (1,1,0)x(0,1,1) a tieto dáta majú ročnú sezónnosť. Porovnanie skutočných dát s hodnotami získanými týmto modelom spolu s predikciami na nasledujúcich 10 kvartálov môžeme vidieť na Obr. 13a). Chyba takto získaných predikcií je 3.49957.



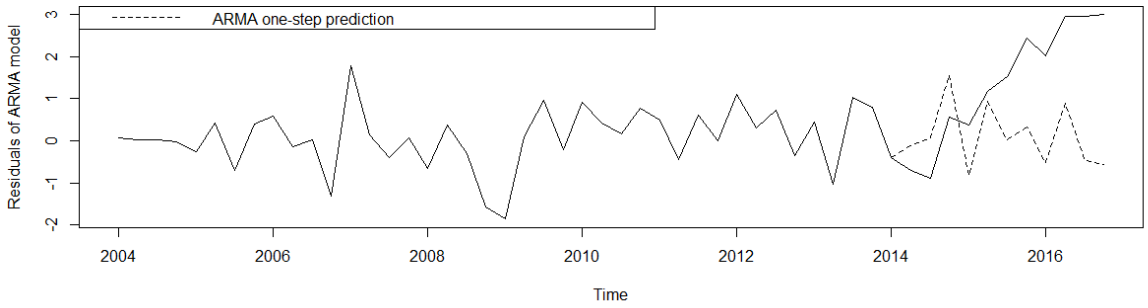
(a) Predikcie ARMA modelu



(b) Jednokrokové predikcie ARMA modelu

Obr. 13: Modelovanie a predikovanie ARMA modelom

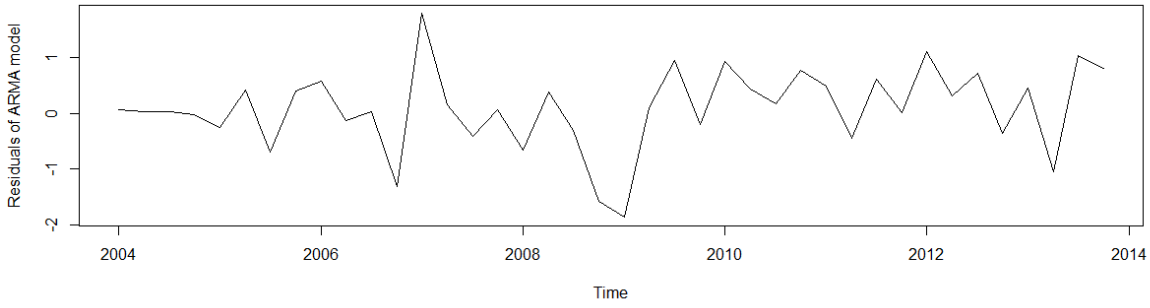
V nasledujúcom kroku sa pokúsime spresniť predikcie lineárneho modelu použitím jednokrokových predikcií. To znamená, že v čase t predikujeme iba hodnotu času *t+1*. Následne, v čase *t+1*, už poznáme hodnotu pre čas *t+1* a predikujeme len hodnotu času *t+2*. Tento postup aplikujeme na celé predikované obdobie. Takto získané predikcie sú zobrazené na Obr. 13b. Chyba jednokrokových predikcií je v tomto prípade 0.4914998. Na nasledujúcom Obr. 14 môžeme vidieť rozdiel rezíduí lineárneho modelu v prípade tvorby predikcií v jednom kroku v porovnaní s predikovaním opakovanými jednokrokovými predikciami.



Obr. 14: Porovnanie rezíduí ARMA modelu s rezíduami jednokrokových predikcií

3.1.1 Neurónová sieť

V tejto časti budeme modelovať časový vývoj rezíduí ARMA modelu neurónovou sieťou. Pokúsime sa odhadnúť model pre tréningovú vzorku dát a následne vytvoríme predikcie rezíduí. Tieto predikcie pripočítame k predikciám ARMA modelu. Cieľom bude predikovať dáta modelom s vyššou kvalitou predikcií. Na odhad modelu budeme používať funkciu *nnetar* a pri predikciách využijeme funkciu *forecast*, obe sú obsiahnuté v knižnici *forecast*[12]. Grafický vývoj rezíduí, ktoré budú predmetom modelovania, môžeme vidieť na Obr. 15.



Obr. 15: Rezíduá ARMA modelu dát zamestnanosti

V prvej časti sme predikovali celé predikčné obdobie naraz a bez využitia validácie. Pri modelovaní bez validácie sme určili optimálny počet prvkov skrytej vrstvy na tréningovej vzorke. Ak počet vstupov neurónovej siete označíme n, defaultná hodnota pre počet prvkov skrytej vrstvy je $\frac{n}{2} + 1$. V našej práci sme otestovali všetky hodnoty

z intervalu $(\frac{n}{4}, \frac{3n}{4})$ a vybrali tú, ktorá minimalizuje chybu modelu. Následne, keď sme získali optimálny počet prvkov skrytej vrstvy, sme testovacej vzorke spravili predikcie v jednom kroku.

V nasledujúcej Tabuľke 2 sú zhrnuté chyby jednotlivých predikcií a optimálny počet prvkov skrytej vrstvy v prípade tvorby predikcií v jednom kroku a bez využitia validácie. Na tréningovej vzorke sme teda odhadli kvalitu jednotlivých modelov a model s najmenšou chybou sme následne použili na predikovanie testovacej vzorky dát. Pre porovnanie uvádzame, že chyba predikcií ARMA modelu je 3.49957.

Aktivačná funkcia	Chyba modelu	S	Chyba predikcií
lineárna	9.101444e-09	14	3.569672
logistická	9.821831e-09	25	3.688969
Hyperbolický tangens(Sigmoid)	1.120362e-08	27	3.791701

Tabuľka 2: Chyby neurónovej siete pri predikovaní v jednom kroku bez využitia validačnej časti

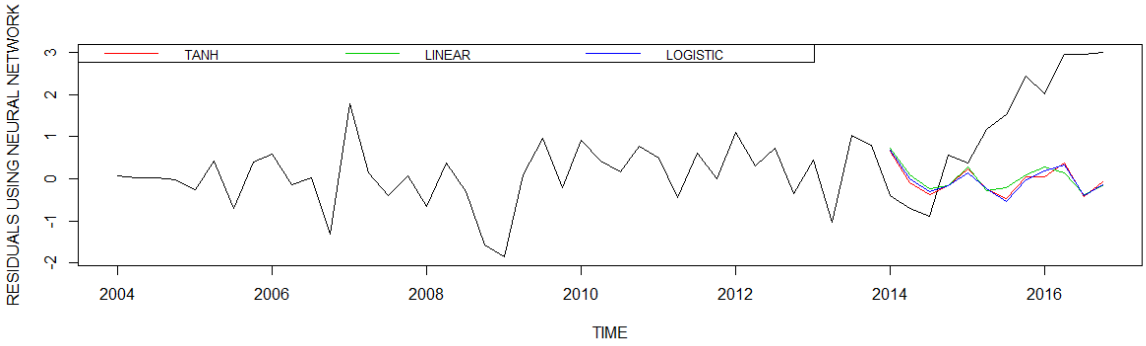
V ďalšej časti sme postup opakovali, pričom sme využili pri výpočte aj validáciu. Na validačnej vzorke dát sme určili počet prvkov skrytej vrstvy, pri ktorom je minimalizovaná chyba predikcií rezíduí. Za najlepší sme teda nepovažovali model s minimálnou chybou odhadu dát tréningovej vzorky, ale model, ktorý minimalizuje chybu predikcií na validačnej vzorke dát. Tabuľka 3 obsahuje chybu predikcií na obdobie validačnej vzorky pre optimálny počet prvkov skrytej vrstvy S a chybu výsledných predikcií.

Aktivačná funkcia	Chyba validačnej vzorky	S	Chyba predikcií
lineárna	1.569056	29	3.790353
logistická	1.386177	28	3.850053
Hyperbolický tangens(Sigmoid)	1.58681	29	3.774969

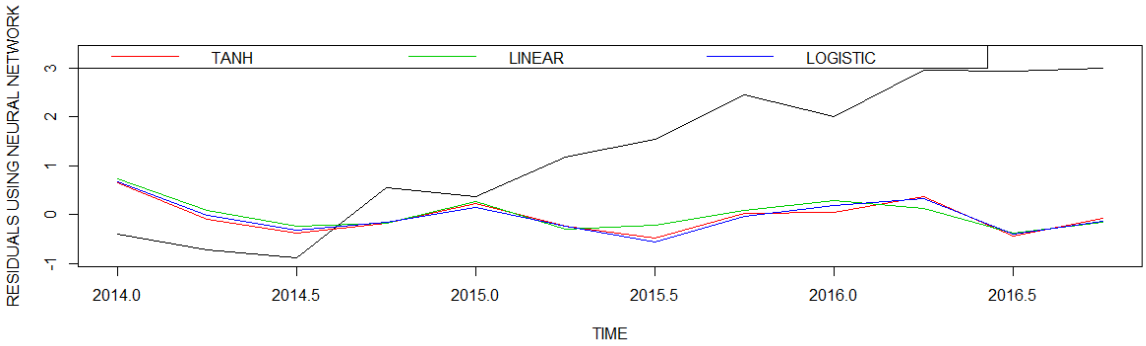
Tabuľka 3: Chyby neurónovej siete pri predikovaní v jednom kroku s využitím validačnej časti

Grafické zobrazenie predikcií pre rôzne typy aktivačných funkcií s využitím validácie

v porovnaní so skutočnými rezíduami predikcií ARMA modelu je obsahom Obr. 16.



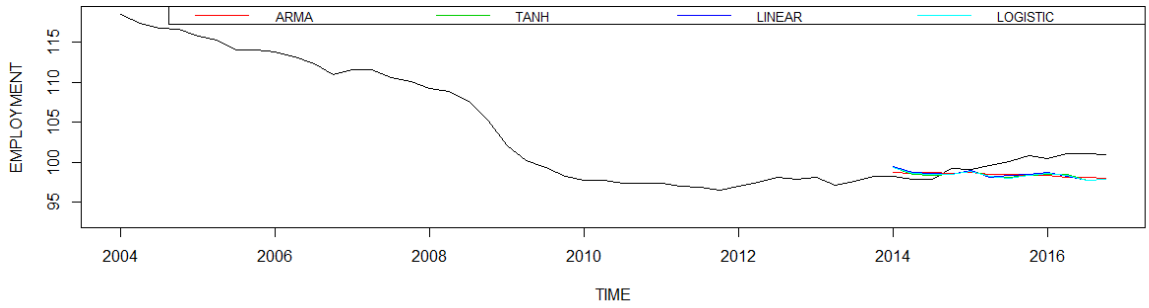
(a) Priebeh časového radu



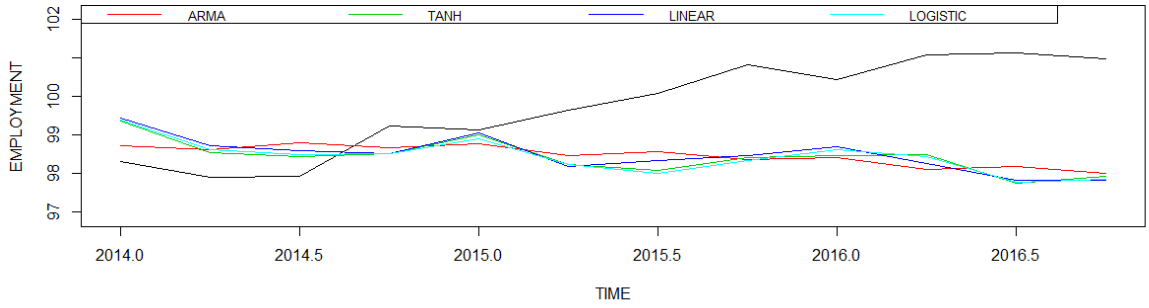
(b) Detailné zobrazenie predikcií

Obr. 16: Predikcie rezíduí modelované neurónovou sieťou v porovnaní so skutočnými rezíduami predikcií ARMA modelu

Porovnaním Tabuľky 2 a 3 vidíme, že použitie validácie neprinieslo zlepšenie kvality predikcií pre naše dáta. Hoci predikcie na validačnej vzorke nasvedčovali možné zlepšenie kvality predikcií, výsledné predikcie sa nám vylepšiť nepodarilo. Dôvodom môže byť fakt, že vývoj časového radu v predikovanom období sa výrazne líši od vývoja v čase odhadu modelu, čo môžeme vidieť na Obr.16. To znamená, že hodnoty rezíduí sú v čase predikcií výrazne vyššie ako v období trénovania modelu. Nakoľko naša neurónová sieť nazerá na rezíduá ako na časový rad, pričom nemá žiadnu informáciu o prepojení tohto časového radu na ARMA model, je očakávateľné, že nemôže zareagovať na prudkú zmenu vo vývoji tohto časového radu.
Na Obr.17 môžeme vidieť výsledné predikcie v porovnaní so skutočnými dátami zamestnanosti pre rôzne typy aktivačných funkcií pri využití validácie.



(a) Priebeh časového radu



(b) Detailné zobrazenie predikcií

Obr. 17: Predikcie zamestnanosti neurónovou sieťou pre rôzne typy aktivačnej funkcie

V druhej časti opakujeme postup modelovania rezíduí, pričom použijeme jednokrokové predikcie. Opäť sme najskôr modelovali bez využitia validácie. V Tabuľke 4 vidíme prehľad minimálnych chýb odhadov na tréningovej vzorke dát. Výsledná chyba je rovnako obsahom nasledujúcej tabuľky s označením *Chyba predikcií*. Pripomenieme, že chyba jednokrokových ARMA predikcií je 0.4914998.

Aktivačná funkcia	Chyba modelu	S	Chyba predikcií
lineárna	9.101444e-09	14	0.572511
logistická	9.821831e-09	25	0.7821626
Hyperbolický tangens(Sigmoid)	1.120362e-08	27	0.8401861

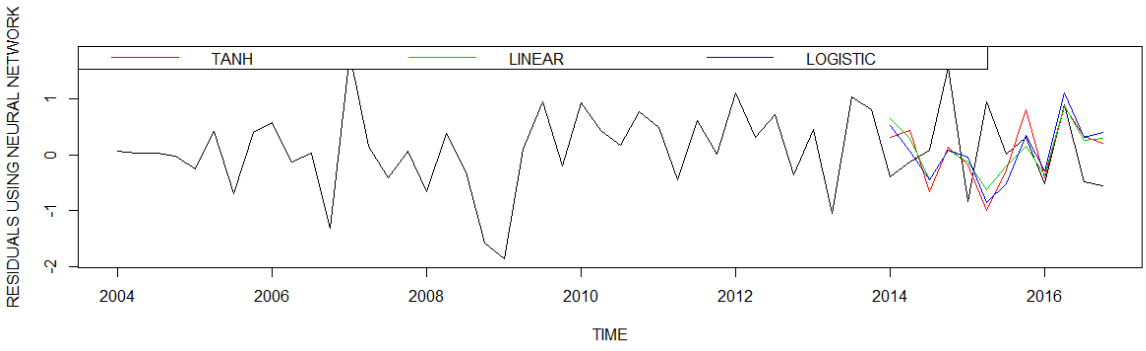
Tabuľka 4: Chyby predikcií neurónovej siete použitím jednokrokových predikcií bez využitia validácie

V ďalšej časti sme postup jednokrokových predikcií zopakovali, pričom sme validačné obdobie použili na určenie počtu prvkov skrytej vrstvy. Zdrojový kód použitý v tejto časti je k dispozícii v Prílohe A. Prehľad chýb získaných predikcií je v Tabuľke 5.

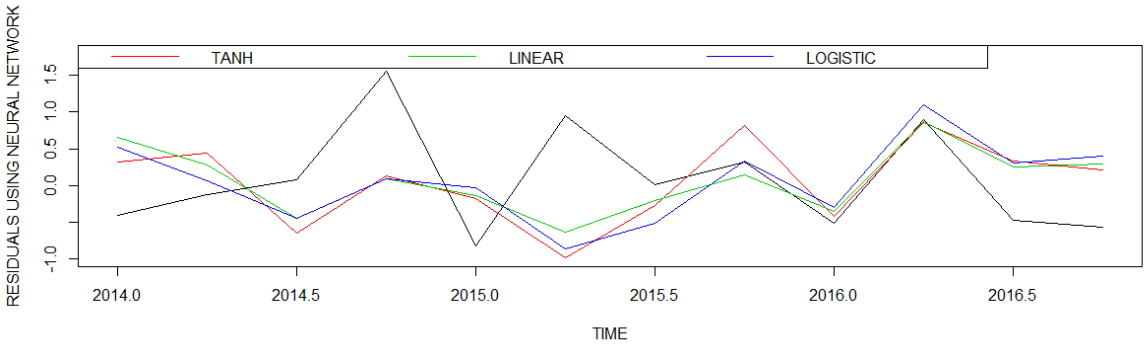
Aktivačná funkcia	Chyba validačnej vzorky	S	Chyba predikcií
lineárna	0.5752537	12	0.7958304
logistická	0.712122	11	0.7196305
Hyperbolický tangens(Sigmoid)	0.6098272	13	0.8355308

Tabuľka 5: Chyby predikcií metódou neurónovej siete použitím validácie a jednokrokových predikcií

Grafické zobrazenie jednokrokových predikcií neurónovej siete v porovnaní s jednokrokovými predikciami ARMA modelu môžeme vidieť na Obr. 18.



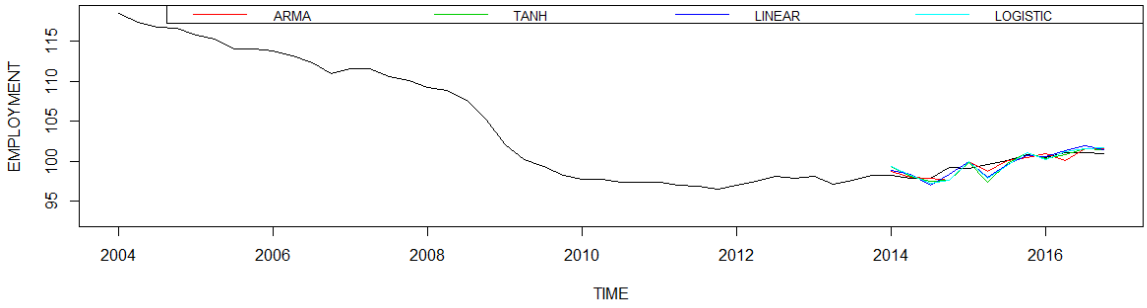
(a) Priebeh časového radu



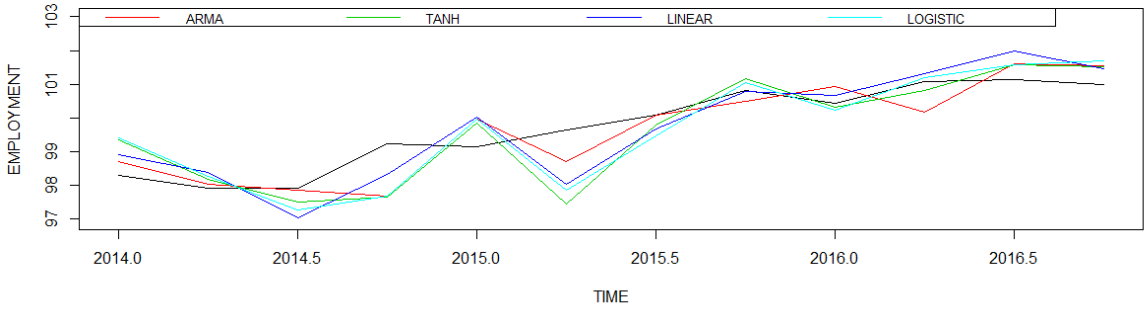
(b) Detailné zobrazenie predikcií

Obr. 18: Jednokrokové predikcie rezíduí neurónovou sieťou (bez validácie)

Na Obr. 19 vidíme výsledné jednokrokové predikcie zamestnanosti v USA pri použití validácie a rôznych typov aktivačných funkcií.



(a) Priebeh časového radu



(b) Detailné zobrazenie predikcií

Obr. 19: Jednokrokové predikcie dát zamestnanosti pre rôzny typy aktivačnej funkcie neurónovej siete s využitím validácie

Z Tabuliek (2) a (4) sme videli, že chyba predikcií, ktoré sú súčtom ARMA predikcií a predikcií rezíduí neurónovou sieťou, je väčšia ako chyba lineárneho modelu. Dôvodom je celkovo nízka kvalita modelu neurónovej siete, pretože už samotný odhad modelu na tréningovej časti dát mal väčšiu chybu ako celkové predikcie ARMA modelu.

3.1.2 Metóda oporných bodov

V tejto časti použijeme na predikovanie rezíduí metódu oporných bodov. Vytvoríme si informačnú maticu, ktorá bude obsahovať informácie o každom pozorovaní. V tomto prípade budeme za informácie považovať lagy pozorovania a budeme pracovať v dvojročnou históriou. To znamená, že pozorovaniu z času t prislúcha vektor informácií ob-

sahujúci pozorovanie z času *t-1, t-2, ..., t-8*. Takto definovaná úloha však neumožňuje vytváranie viacerých predikcií v jednom kroku, použijeme teda priamo jednokrokovú metódu.

Pracovať budeme so všetkými troma typmi jadrovej funkcie. Vyhodnotíme, ktorá sa javila byť najlepšia z tréningovej a validačnej vzorky, a následne tento predpoklad overíme na testovacej vzorke. V nasledujúcej Tabuľke 6 môžeme vidieť optimálne hodnoty parametrov funkcií získané funkciou *tune.svm*.

Optimálne hodnoty parametrov				
Typ použitej kernel funkcie	γ	c_0	ϵ	Penalizačná konštanta
lineárna	/	/	0,6	1
polynomiálna	0,1	0,3	0,1	10
Gaussovská	1	/	0,1	10
Hyperbolický tangens(Sigmoid)	0,1	0	1	10

Tabuľka 6: Vstupné hodnoty pri použití rôznych typov kernel funkcie

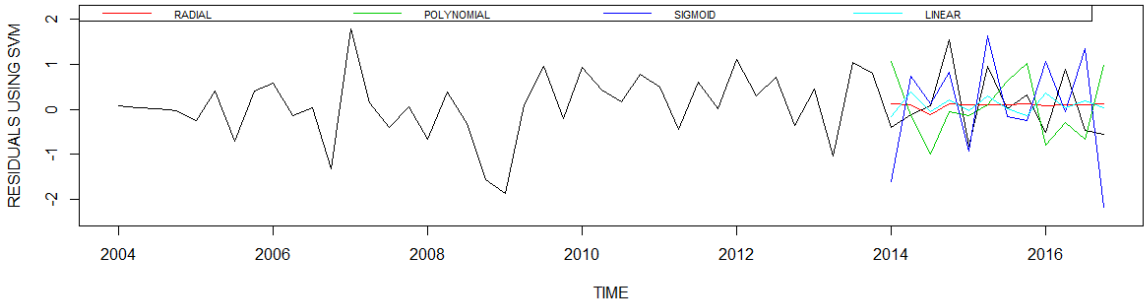
Tabuľka 7 sumarizuje výsledné chyby tohto modelovania. Ako *chybu modelu* označujeme MSE chybu definovanú vzťahom (33) vypočítanú na tréningovej vzorke dát. Pojmom *chyba predikcií* označujeme výslednú chybu validačnej a testovacej vzorky. Chyba predikcií tak označuje MSE chybu vypočítanú na validačnej a testovacej vzorke.

Typ použitej kernel funkcie	Chyba modelu	Chyba predikcií
lineárna	0.6543108	0.4764402
polynomiálna	0.186044	0.9812367
Gaussovská	0.0709068	0.486204
Hyperbolický tangens(Sigmoid)	2.998852	1.075022

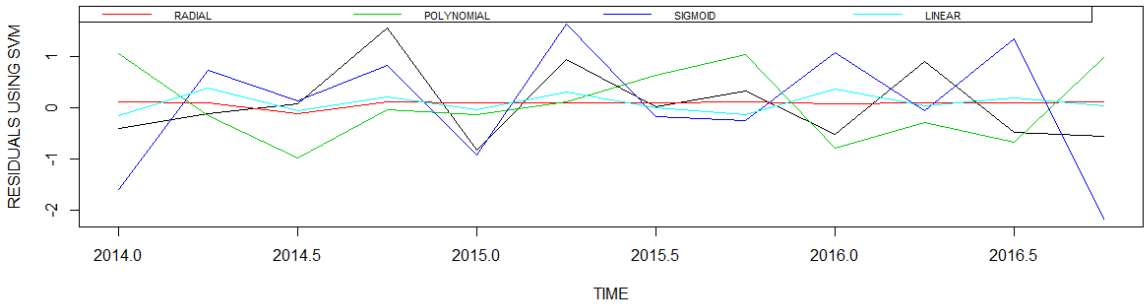
Tabuľka 7: Chyby modelu a predikcií pri použití rôznych typov kernel funkcie

Pre porovnanie pripomenieme, že chyba jednokrokových predikcií ARMA modelu je 0.4914998. Vidíme, že použitie lineárnej a Gaussovskej funkcie v tomto prípade prinieslo

zlepšenie predikcií. Použitím sigmoidu a polynomiálnej funkcie sme však predikcie nevylepšili. Použitie funkcie sigmoid však naznačovalo veľké odchýlky už na tréningovej vzorke dát. Grafické zobrazenie predikcií metódou oporných bodov je na Obr.20.



(a) Priebeh časového radu



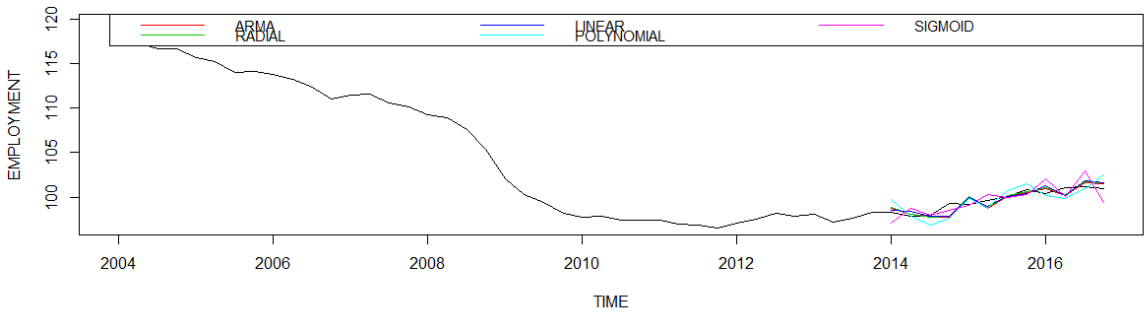
(b) Detailné zobrazenie predikcií

Obr. 20: Predikcie rezíduí metódou oporných bodov pre rôzne typy kernel funkcie v porovnaní so skutočnými rezíduami predikcií ARMA modelu

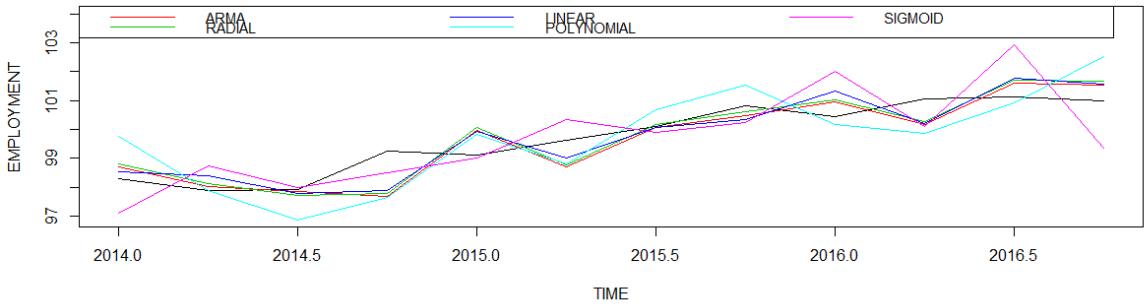
Na Obr. 20 vidíme, že sigmoidová funkcia má veľký problém s presnosťou prvej a poslednej predikcie. Tieto nepresnosti zapríčili, že metóda oporných bodov so sigmoidovou funkciou je výrazne horšia v porovnaní s ostatnými kernel funkciami. Ani prípadné neuvažovanie týchto dvoch hodnôt však nenaznačuje vyššiu kvalitu tejto metódy na zvyšných predikciách. Z Obr. 20 rovnako vidno, že najpresnejšie metódy, metódy s lineárnym a Gaussovským jadrom, sú v predikciách veľmi konzervatívne a majú malú disperziu.

Metóda oporných bodov teda priniesla zlepšenie predikcií lineárneho ARMA modelu pre niektoré typy kernel funkcií. Grafické zobrazenie výsledných predikcií, získaných

ako súčet ARMA predikcií a predikcií metódy oporných bodov, môžeme vidieť na nasledujúcom Obr. 21.



(a) Priebeh časového radu

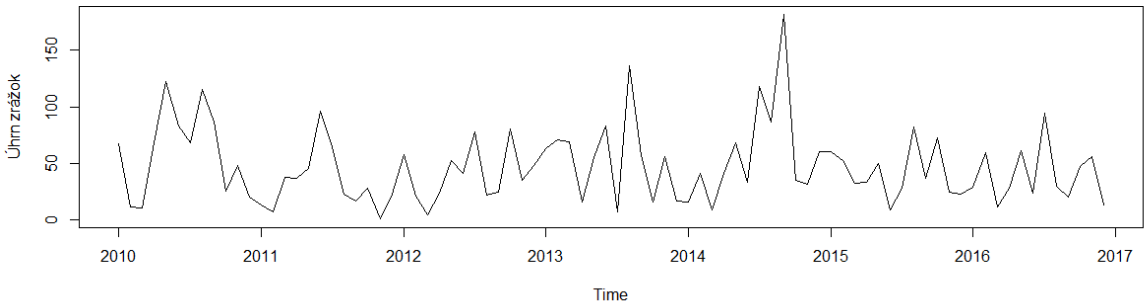


(b) Detailné zobrazenie predikcií

Obr. 21: Porovnanie ARMA predikcií dát zamestnanosti s predikciami metódy oporných bodov pre rôzne typy kernel funkcie

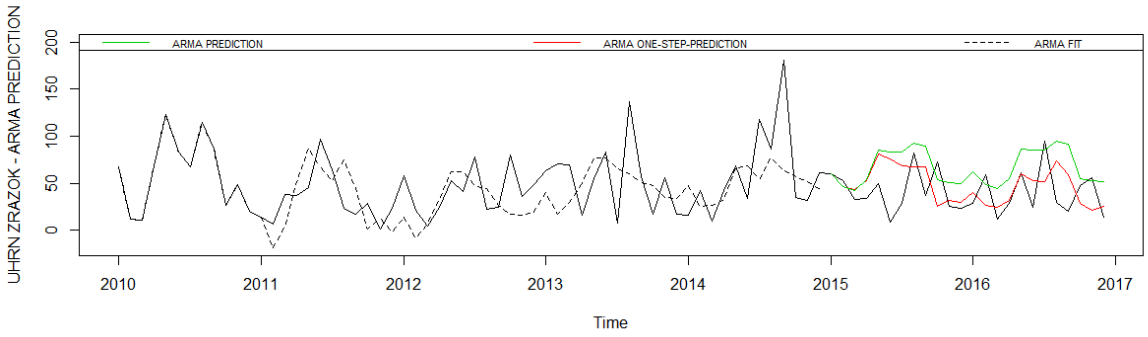
3.2 Zrážková činnosť v obci Most pri Bratislave

Tieto dáta zaznamenávajú celkový mesačný úhrn zrážok v milimetroch. Pochádzajú z obce Most pri Bratislave a zaznamenávajú vývoj v rokoch 2010 - 2016. Dáta zozbierala meteorologická stanica v tejto obci a sú zverejnené na ich stránke [26]. Priebeh týchto dát môžeme vidieť na Obr. 22.



Obr. 22: Mesačný úhrn zrážok v obci Most pri Bratislave v rokoch 2010 - 2016

Posledné dva roky použijeme ako validačnú a testovaciu vzorku na overovanie kvality predikcií modelov. Na modelovanie lineárnej časti tréningových dát sme použili model ARIMA(0,1,1)x(0,1,1), pričom dáta majú 12-mesačnú sezónnosť. Odhady hodnôt a predikcie odhadnuté lineárnym modelom sú zobrazené na nasledujúcom Obr. 23.

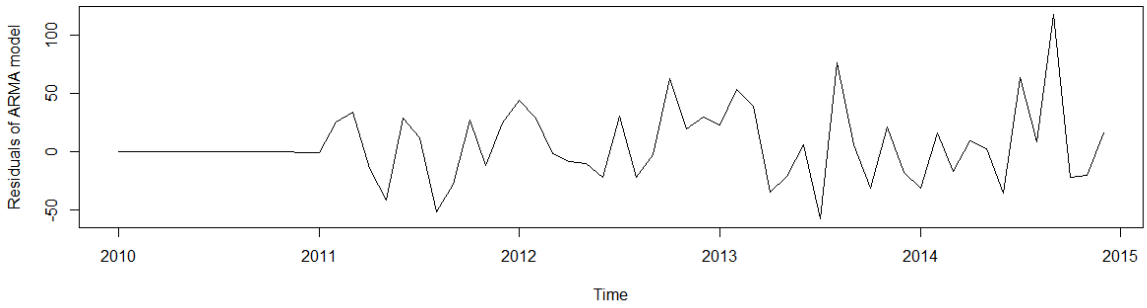


Obr. 23: Modelovanie a predikovanie zrážkovej činnosti použitím ARMA modelu

Chyba predikcií ARMA modelu v prípade predikovania celých dát v jednom kroku je 1380,567. Použitie jednokrokových predikcií viedlo k chybe 842,0515.

3.2.1 Neurónová sieť

Metódou neurónovej siete sa pokúsime modelovať rezíduá lineárneho ARMA modelu, ktorých priebeh vidíme na nasledujúcom Obr. 24. Následne budeme odhadovať predikcie dvoma rôznymi spôsobmi prístupu, predikovaním v jednom alebo viacerých krokoch. Pri procese modelovania budeme využívať validáciu.



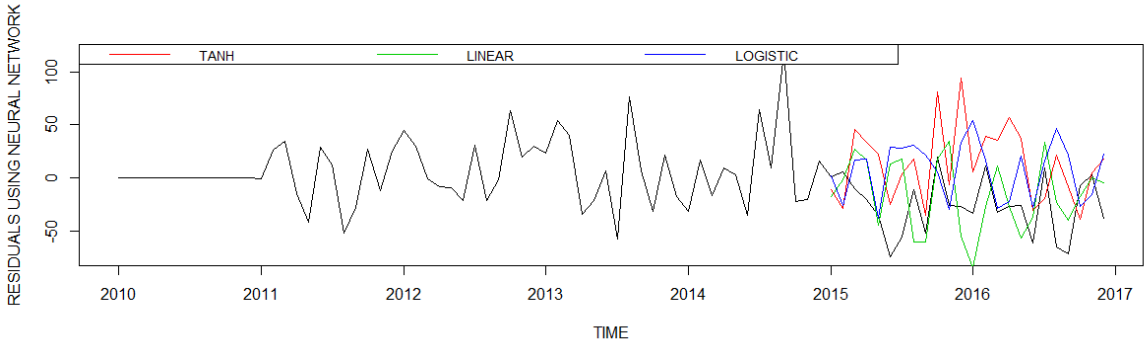
Obr. 24: Rezíduá ARMA modelu dát zrážkovej činnosti

V prvej časti použijeme metódu neurónovej siete na modelovanie všetkých predikcií v jednom kroku. Použitím tréningovej a validačnej vzorky dát určíme optimálny počet prvkov skrytej vrstvy. Následne vytvoríme model s týmto počtom prvkov skrytej vrstvy, spravíme predikcie a tie porovnáme so skutočnými hodnotami rezíduí validačnej a testovacej vzorky. Hodnoty získané týmto postupom môžeme vidieť v nasledujúcej Tabuľke 8.

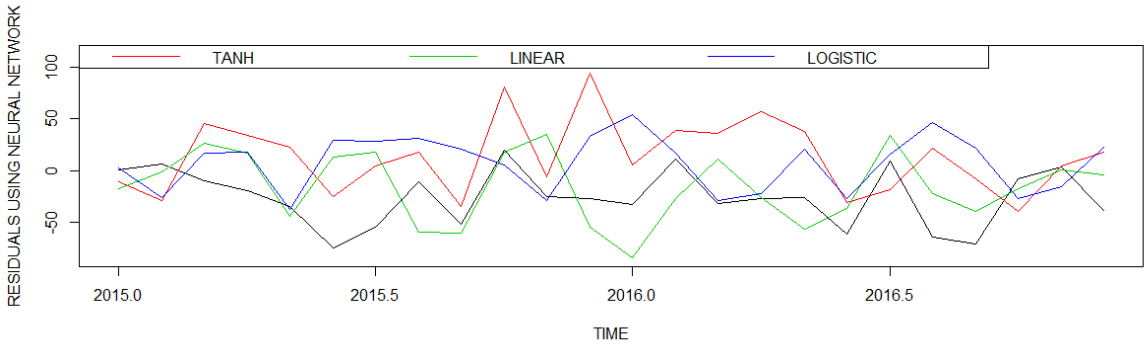
Aktivačná funkcia	Chyba valid. vzorky	Prvky skrytej vrstvy	Chyba predikcií
lineárna	1126.715	44	1466.114
logistická	976.3499	24	2843.005
Hyperbolický tangens(Sigmoid)	1116.398	43	2984.091

Tabuľka 8: Chyby neurónovej siete pri predikovaní v jednom kroku

Vidíme, že už chyba predikcií validačnej vzorky má chybu porovnateľnú s chybou ARMA modelu, ktorá je v tomto prípade 1380,567. Výsledné predikcie, ktoré sú očakávane horšie ako chyba predikcií validačnej vzorky, sú tak v prípade sigmoidu a logistickej aktivačnej funkcie viac než dvojnásobne horšie. Lineárne predikcie, ktoré dávajú najkonzervatívnejšie hodnoty, majú najmenšiu chybu spomedzi aktivačných funkcií neurónovej siete.
Grafické zobrazenie predikcií pre rôzne typy aktivačných funkcií v porovnaní so skutočnými rezíduami predikcií ARMA modelu je obsahom Obr. 25.



(a) Priebeh časového radu



(b) Detailné zobrazenie predikcií

Obr. 25: Predikcie rezíduí neurónovou sieťou pre rôzne typy aktivačnej funkcie v porovnaní so skutočnými rezíduami predikcií ARMA modelu

Na Obr. 25 vidno, že nelineárna metóda očakáva väčšie turbulencie v predikovanom období v porovnaní so skutočnosťou.

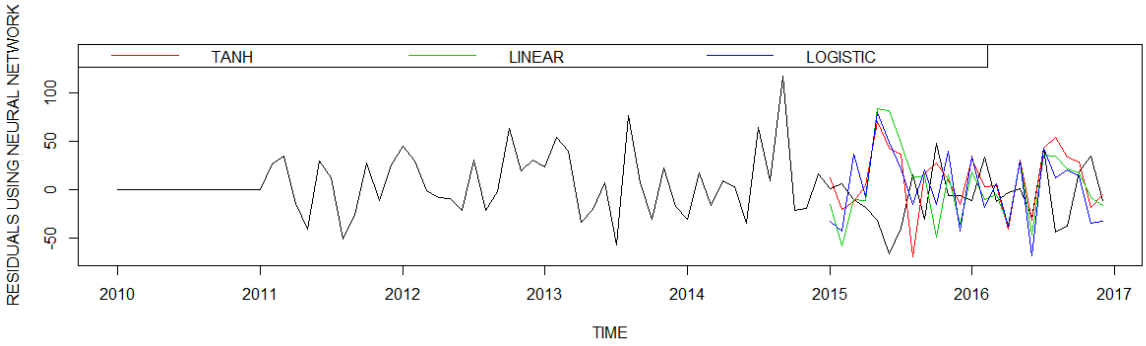
V druhej časti opakujeme postup s validačnou časťou modelovania na jednokrokové predikcie. Výsledky tohto modelovania sme zhrnuli v Tabuľke 9.

Aktivačná funkcia	Chyba valid. vzorky	Prvky skrytej vrstvy	Chyba predikcií
lineárna	2700.992	29	3193.798
logistická	2462.394	16	2784.72
Hyperbolický tangens(Sigmoid)	2473.212	35	2624.483

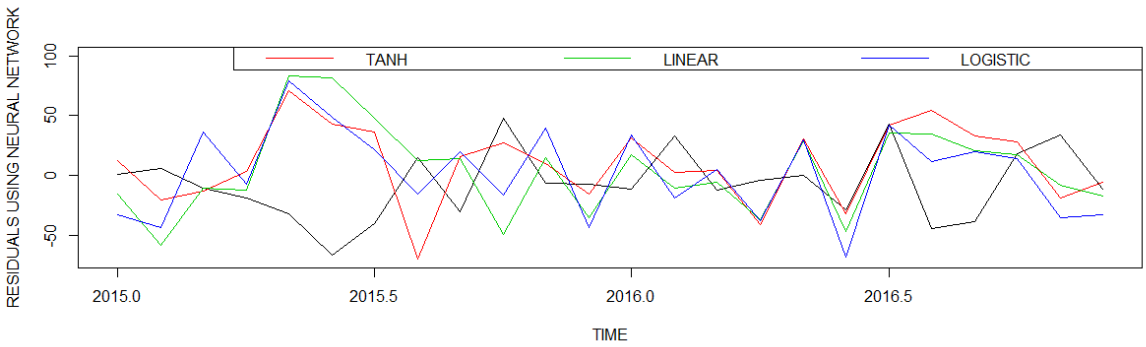
Tabuľka 9: Chyby predikcií neurónovej siete s využitím jednokrokových predikcií

V prípade jednokrokových predikcií aplikovaných na tieto dáta došlo k zhoršeniu

kvality výsledných predikcií. Dôvodom je veľká chyba už predikcií na validačnej vzorke, ktorá je zapríčinená veľkými výkyvmi v očakávaných hodnotách. Grafické zobrazenie jednokrokových predikcií neurónovej siete v porovnaní s jednokrokovými predikciami ARMA modelu môžeme vidieť na Obr. 26.



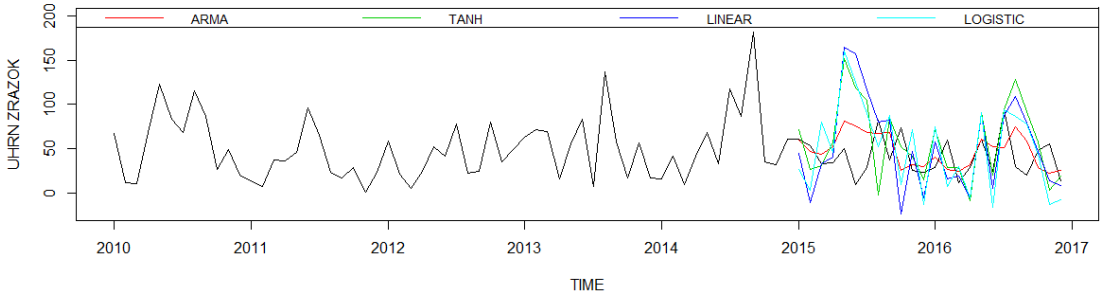
(a) Priebeh časového radu



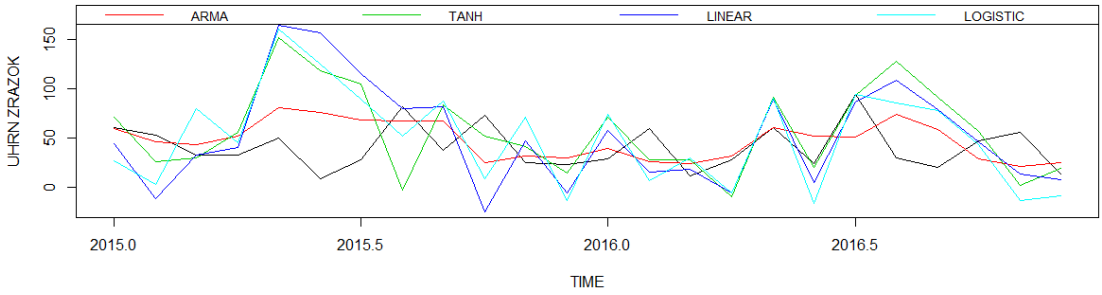
(b) Detailné zobrazenie predikcií

Obr. 26: Jednokrokové predikcie rezíduí neurónovej siete v porovnaní so skutočnými rezíduami

Ani v tomto prípade sa nám nepodarilo zvýšiť kvalitu lineárnych predikcií ARMA modelu. Dôvodom je opäť fakt, že nesprávne predikovaná hodnota rezíduí neurónovou sieťou vo viacerých prípadoch umocnila chybu ARMA modelu, čo viedlo k väčšej chybe výsledných predikcií. Tento jav môžeme vidieť na nasledujúcom Obr. 27, kde porovnávame výsledné predikcie získané ARMA modelom a neurónovou sieťou pre rôzne typy aktivačných funkcií.



(a) Priebeh časového radu



(b) Detailné zobrazenie predikcií

Obr. 27: Jednokrokové predikcie úhrnu zrážok metódou neurónovej siete

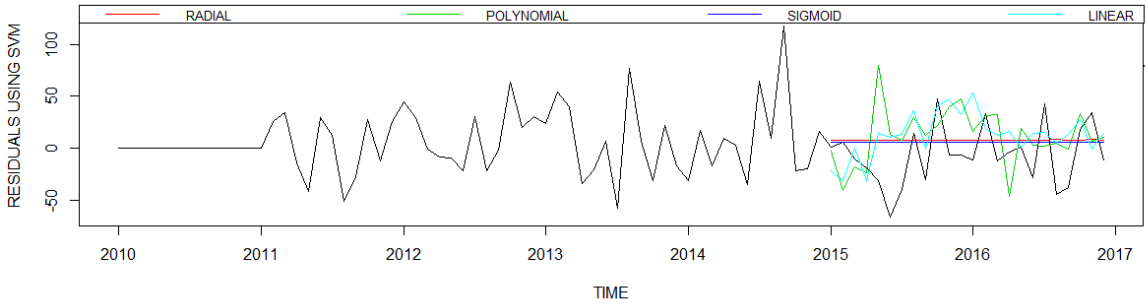
3.2.2 Metóda oporných bodov

Na použitie metódy vytvoríme ročnú informačnú maticu, teda matica bude obsahovať 12 mesačnú históriu. Zdrojový kód použitý na získanie výsledkov v tejto časti je obsiahnutý v Prílohe B. Optimálne vstupné hodnoty sme zhrnuli v Tabuľke 10.

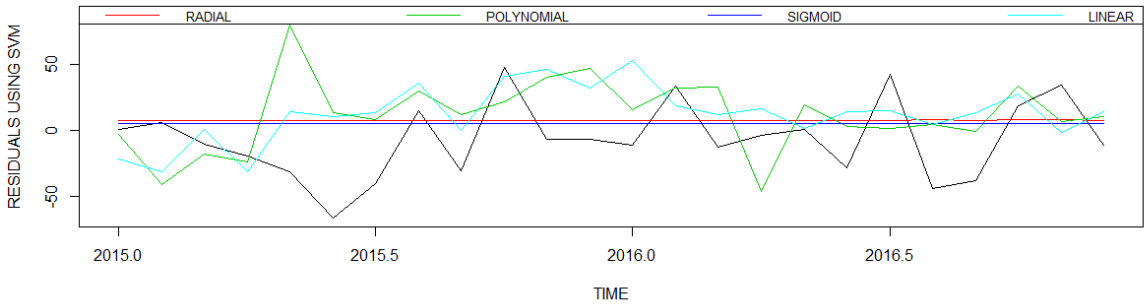
Optimálne hodnoty parametrov				
Typ použitej kernel funkcie	γ	c_0	ϵ	Penalizačná konštanta
lineárna	/	/	0,5	1
polynomiálna	0,2	0,1	0,2	10
Gaussovská	0,7	/	0,1	100
Hyperbolický tangens(Sigmoid)	0	0,5	0,2	100000

Tabuľka 10: Vstupné hodnoty pri použití rôznych typov kernel funkcie

Vidíme, že vstupné hodnoty sa v závislosti od použitej funkcie líšia. Výsledné predikcie rezíduí funkcií s týmito parametrami môžeme vidieť na Obr. 28.



(a) Priebeh časového radu



(b) Detailné zobrazenie predikcií

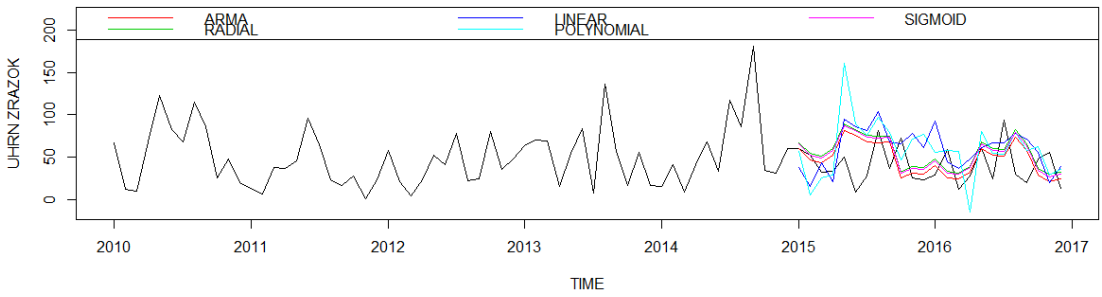
Obr. 28: Predikcie rezíduí metódou oporných bodov v porovnaní s ARMA rezíduami

Z Obr. 28 vidíme, že sigmoid a gaussovské kernel funkcie nám pre tieto dáta dávajú veľmi konzervatívne predikcie. Polynomiálna a lineárna funkcia naopak výrazne mení hodnoty v čase, no ich úspešnosť kopírovania reálnych rezíduí sa javí ako nízka. Tento predpoklad overíme vypočítaním chyby jednotlivých metód.
Chyby modelov a predikcií získané použitím parametrov uvedených v Tabuľke 10 môžeme pozorovať v nasledujúcej Tabuľke 11. Pre porovnanie chyba ARMA modelu je v tomto prípade 859.8063.

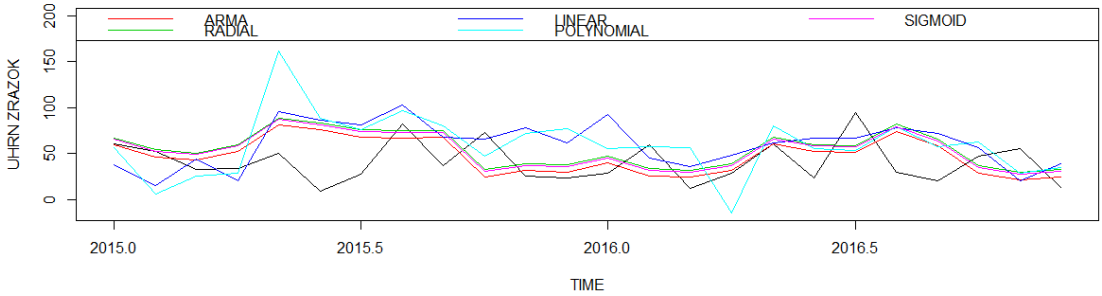
Typ použitej kernel funkcie	Chyba modelu	Chyba predikcií
lineárna	1155.317	1835.263
polynomiálna	405.0046	1835.263
Gaussovská	143.8173	999.6072
Hyperbolický tangens(Sigmoid)	1196.701	949.3624

Tabuľka 11: Chyby modelu a predikcií pri použití rôznych typov kernel funkcie

Tabuľka 11 potvrdzuje náš predpoklad o výhodnosti konzervatívnych predikcií. Vidíme, že sigmoid a gaussovská kernel funkcia, ktorých predikcie sa v čase výrazne nemenili, majú výrazne nižšiu chybu ako lineárna a polynomiálna kernel funkcia. Napriek tomu sa zhoršila kvalita predikcií v porovnaní s ARMA modelom. Výsledné predikcie úhrnu zrážok pre rôzne typy kernel funkcií sú graficky znázornené na Obr. 29.



(a) Priebeh časového radu

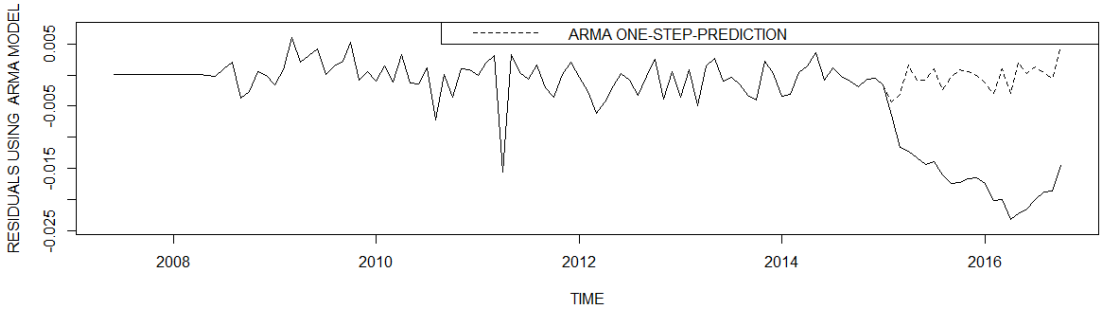


(b) Detailné zobrazenie predikcií

Obr. 29: Predikcie úhrnu zrážok metódou oporných bodov pre rôzne typy kernel funkcie v porovnaní s predikciami ARMA modelu

3.3 Miera zlyhania bankových klientov

V záverečnej časti práce budeme modelovať skutočné bankové dáta. Práve z tohto dôvodu nebudeme môcť zverejniť skutočný priebeh dát, graficky budeme prezentovať len výsledky modelovania rezíduí ARMA modelu. Mesačné dáta odzrkadľujú schopnosť klientov plniť svoje záväzky voči banke v období rokov 2007 - 2016. Toto obdobie bolo ovplyvnené krízou na finančných trhoch, ktorá ovplyvnila aj ekonomickú situáciu klientov banky. Kríza tak spôsobila výrazný nárast miery zlyhania klientov, ktorý sa v pokrízovom období len pomaly dostával na pôvodnú úroveň. Očakávame, že táto prudká zmena vo vývoji zhorší schopnosť modelu vytvoriť kvalitný model a rovnako zníži mieru kvality predikcií.
Najskôr sme dáta odhadli lineárnym modelom a spravili sme predikcie lineárneho modelu. Nakoľko tieto predikcie vykazovali nízku kvalitu, spravili sme následne aj jednokrokové predikcie. Rezíduá ARMA modelu spolu s porovnaním oboch typov predikcií môžeme vidieť na nasledujúcom Obr. 30.



Obr. 30: Predikcie rezíduí miery zlyhania klientov použitím ARMA modelu

Chyba predikcií v prípade predikovania celej predikovanej vzorky v jednom kroku bola 0.0002823432, použitím jednokrokových predikcií sme chybu zredukovali na 3.983691e-06. V ďalšej časti sa pokúsime chybu ešte vylepšiť použitím nelineárnych metód.

3.3.1 Neurónová sieť

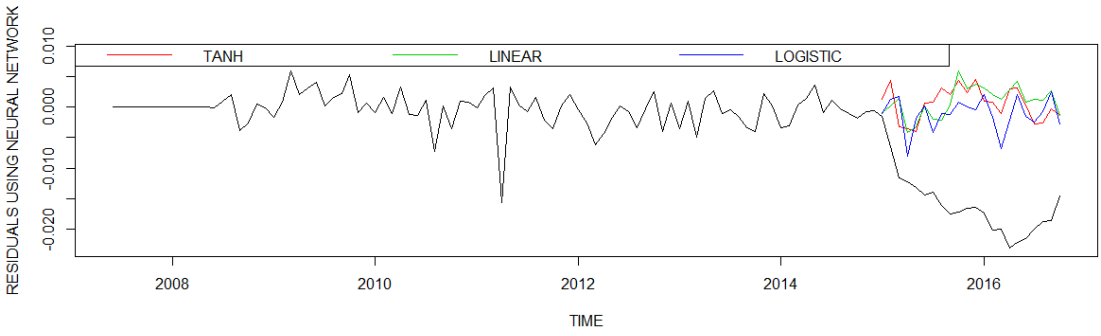
Rezíduá ARMA modelu sme sa podobne ako v predchádzajúcich prípadoch rozhodli namodelovať neurónovou sieťou pre rôzne typy aktivačnej funkcie. Na validačnej vzoke dát sme odhadli optimálny počet prvkov skrytej vrstvy a najlepšiu aktivačnú funkciu.

Následne sme tento optimálny počet použili na vytvorenie predikcií na celé predikované obdobie pre všetky typy aktivačnej funkcie a overili sme tak predpoklad najlepšej aktivačnej funkcie. Získané výsledky môžeme vidieť v Tabuľke 12.

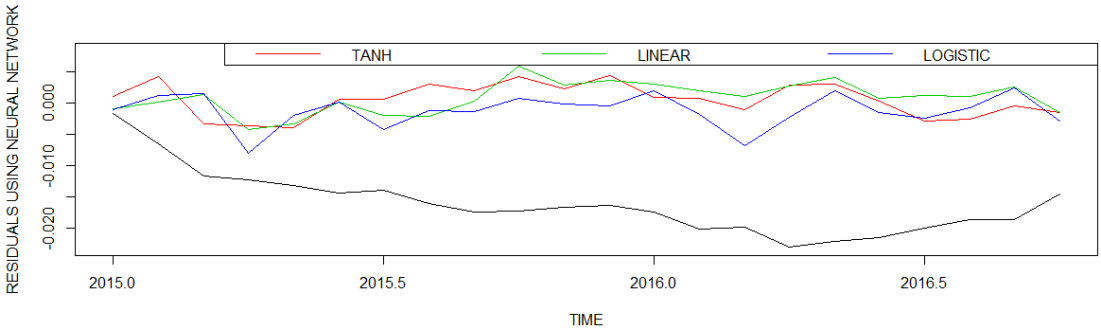
Aktivačná funkcia	Chyba valid. vzorky	Prvky skrytej vrstvy	Chyba predikcií
lineárna	0.0001843404	24	0.0003296322
logistická	0.000186195	29	0.0002525381
Hyperbolický tangens(Sigmoid)	0.0001789359	23	0.0003091012

Tabuľka 12: Chyby neurónovej siete pri predikovaní v jednom kroku

Vidíme, že chyba predikcií na validačnej vzorke je pre všetky typy aktivačnej funkcie veľmi podobná. Bolo by teda veľmi náročné na základe týchto malých rozdielov zamietnuť jeden z typov aktivačnej funkcie. Rovnako vidíme, že najlepšie výsledné predikcie z hľadiska minimalizácie chyby generuje logistická funkcia, ktorá však mala najväčšiu chybu predikcií na validačnej vzorke. Iba použitím tejto aktivačnej funkcie sa nám podarilo mierne vylepšiť ARMA predikcie, ostatné typy aktivačnej funkcie viedli k zhoršeniu kvality ARMA predikcií.



(a) Priebeh časového radu



(b) Detailné zobrazenie predikcií

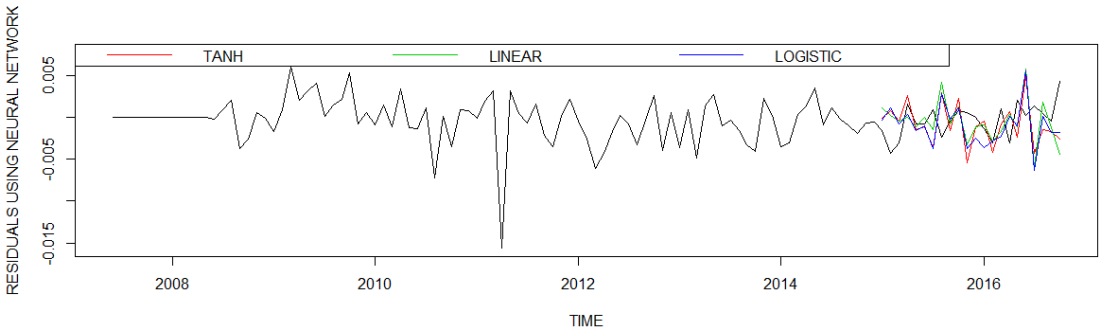
Obr. 31: Predikcie rezíduí ARMA modelu neurónovou sieťou pre rôzne typy aktivačnej funkcie v porovnaní so skutočnými rezíduami

Na Obr. 31 môžeme vidieť predikcie rezíduí ARMA modelu neurónovou sieťou pre všetky typy aktivačnej funkcie spolu so skutočnými rezíduami ARMA modelu.
V snahe ešte viac skvalitniť výsledné predikcie sme použili jednokrokový spôsob predikovania miery zlyhania klientov. Ten sme porovnali s kvalitou výsledných jednokrokových predikcií ARMA modelu. Výsledky získané použitím tohto spôsobu výpočtu môžeme vidieť v Tabuľke 13. Chyba jednokrokových ARMA predikcií miery zlyhania bankových klientov dosahovala hodnotu 3.983691e-06.

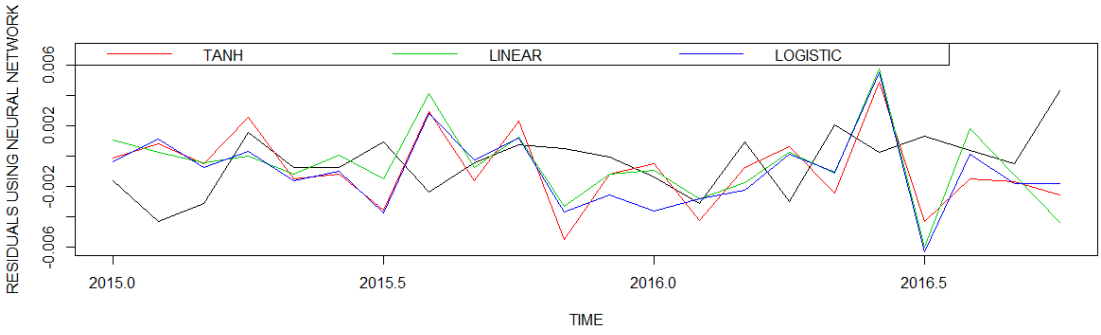
Aktivačná funkcia	Chyba valid. vzorky	Prvky skrytej vrstvy	Chyba predikcií
lineárna	5.896812e-06	31	1.325566e-05
logistická	5.71303e-06	32	1.229431e-05
Hyperbolický tangens(Sigmoid)	6.613297e-06	25	1.219168e-05

Tabuľka 13: Chyby neurónovej siete pri predikovaní rezíduí ARMA modelu využitím jednokrokových predikcií

Chyba predikcií validačnej vzorky v Tabuľke 13 je väčšia ako chyba výsledných ARMA predikcií. Nemožno teda očakávať, že výsledné jednokrokové predikcie použitím neurónovej siete prinesú zlepšenie kvality jednokrokových ARMA predikcií. Jednokrokové predikcie rezíduí pre všetky typy aktivačnej funkcie v porovnaní so skutočnými rezíduami ARMA modelu sú zobrazené na nasledujúcom Obr. 32.



(a) Priebeh časového radu



(b) Detailné zobrazenie predikcií

Obr. 32: Jednokrokové predikcie rezíduí ARMA modelu neurónovou sieťou pre rôzne typy aktivačnej funkcie v porovnaní so skutočnými rezíduami

Výsledné predikcie, ktoré vznikli ako súčet jednokrokových ARMA predikcií a jednokrokových predikcií rezíduí použitím neurónovej siete, majú tak rádovo vyššiu chybu ako samotné jednokrokové ARMA predikcie. Použitie tejto nelineárnej metódy sa tak v tomto prípade nejaví ako dobrý spôsob zvyšovania kvality predikcií. V ďalšej časti práce použijeme na modelovanie rezíduí metódu oporných bodov.

3.3.2 Metóda oporných bodov

Mesačné dáta používané v tejto časti, ktoré majú aj 12-mesačnú sezónnosť, budeme modelovať pomocou 12-mesačnej informačnej matice. Vstupné parametre funkcie sme s využitím validačnej vzorky dát odhadli pomocou funkcie *tune.svm*. Optimálne hodnoty vstupných parametrov spolu s chybou odhadu tento metódy vidíme v Tabuľke 14.

Optimálne hodnoty parametrov				
Typ použitej kernel funkcie	γ	c_0	ϵ	Penalizačná konštanta
lineárna	/	/	0,8	1
polynomiálna	0,1	0,2	0,3	10
Gaussovská	1	/	0,1	10
Hyperbolický tangens(Sigmoid)	0	0,5	1	100000

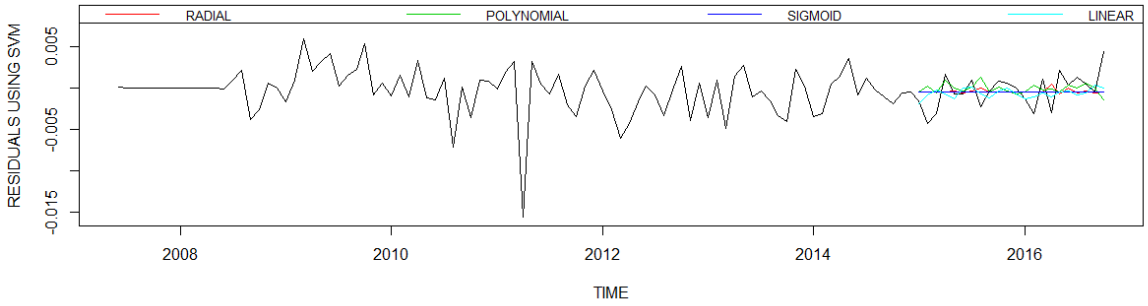
Tabuľka 14: Vstupné hodnoty pri použití rôznych typov kernel funkcie

Hodnoty v Tabuľke 14 sme použili na vytvorenie odhadu predikcií rezíduí na celé predikované obdobie. Chybu predikcií validačnej vzorky spolu s chybou výsledných predikcií pre všetky typy kernel funkcie sme zhrnuli v Tabuľke 15.

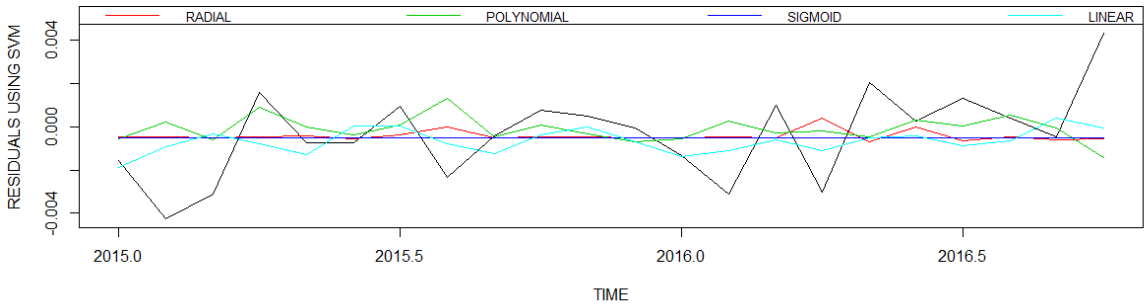
Typ použitej kernel funkcie	Chyba modelu	Chyba predikcií
lineárna	8.816444e-06	3.360044e-06
polynomiálna	5.101815e-06	4.860574e-06
Gaussovská	1.011922e-06	4.266375e-06
Hyperbolický tangens(Sigmoid)	9.163958e-06	3.86635e-06

Tabuľka 15: Chyby modelu a predikcií pri použití rôznych typov kernel funkcie

V tomto prípade síce chyby modelu nenasvedčovali možnosť vylepšenia výsledných predikcií, no i tak lineárna a sigmoidová kernel funkcia dokázali vylepšiť ARMA predikcie dát. Dôvodom môže byť práve použitie validačnej vzorky, ktoré bráni prefitovaniu modelu. Vybrať najoptimálnejší typ kernel funkcie na základe chyby modelu by sme však ani v tomto prípade nevedeli spraviť správne. Časový priebeh predikcií rezíduí metódou oporných bodov v porovnaní so skutočnými hodnotami rezíduí ARMA modelu sme graficky zobrazili na Obr. 33.



(a) Priebeh časového radu



(b) Detailné zobrazenie predikcií

Obr. 33: Jednokrokové predikcie rezíduí ARMA modelu metódoou oporných bodov pre rôzne typy kernel funkcie v porovnaní so skutočnými rezíduami

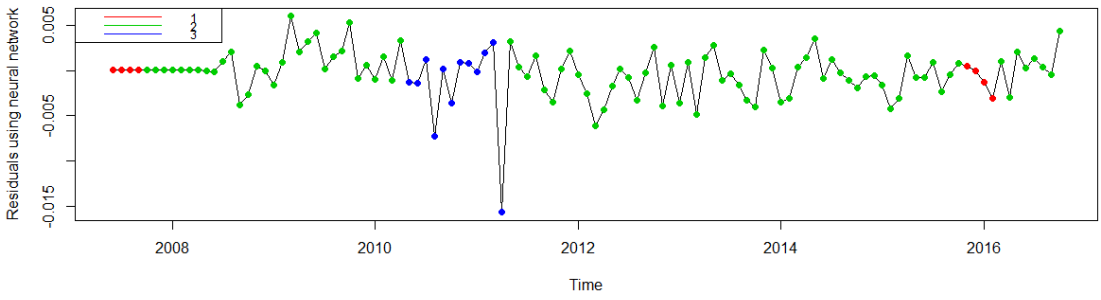
Nakoľko ani neurónová sieť, ani metóda oporných bodov nepriniesli výrazné zlepšenie výsledných predikcií, nehodnotíme tieto metódy ako vhodné pre modelovanie rezíduí uvedených dát. Pokúsime sa tieto metódy aplikovať opäť, nebudeme však pri tom modelovať celé dáta, ale dáta rozdelené do zhlukov. Cieľom bude odfiltrovať vplyv krízy, ktorá výrazne ovplyvnila schopnosť klientov plniť si svoje záväzky voči banke.

3.3.3 Zhluková analýza

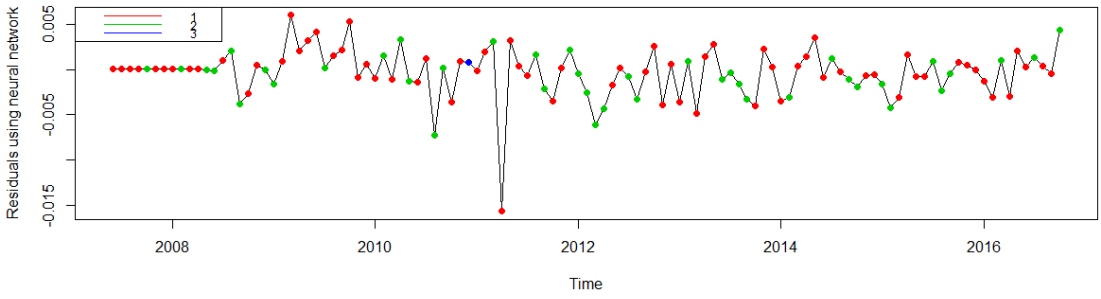
Zhluková analýza definovaná v časti 2.3 neumožňuje vytváranie predikcií použitím výhradne tejto metódy. Informačnú maticu využijeme na rozčlenenie rezíduí do zhlukov na základe podobnosti ich histórie. Následne odhadneme model pre každý zhluk samostatne použitím neurónovej siete a metódy oporných bodov. Pri výstavbe zhlukovej analýzy pre metódy *k-means* a *k-medoids* sme použili funkciu *kcca*. Funkcia spolu s názornými príkladmi, ktorými sme sa inšpirovali, sú obsiahnuté v knižnici *flexclust* [14]. Zhlukovanie založené na normálnom modeli sme vytvorili pomocou funkcie *Mclust*, ktorá sa nachádza v knižnici *mclust* [6].

V práci budeme pracovať iba s jednokrokovými predikciami, čo nám umožní vytvoriť informačnú maticu aj pre dáta z predikovaného obdobia. Následne, keď budeme vedieť, do ktorého zhluku patrí hodnota, ktorú chceme predikovať, spredikujeme ju modelom vytvoreným na tomto zhluku.

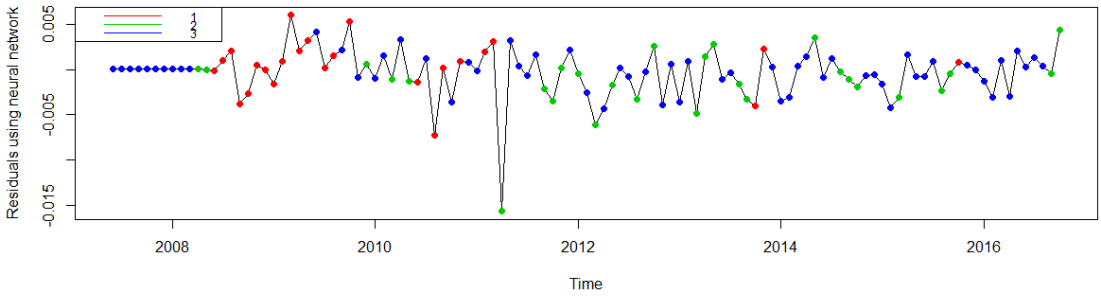
Ako metódy zhlukovania použijeme metódy definované v časti 2.3 a porovnáme ich kvalitu. Rozdelenie rezíduí do zhlukov pri použití týchto metód zhlukovania sme graficky znázornili na Obr. 34. Na obrázku budeme vidieť, že zhlukovanie normálnym modelom sa najviac priblížilo odfiltrovaniu krízového obdobia rozdelením do zhlukov.



(a) Zhlukovanie rezíduí modelom založenom na normálnom rozdelení



(b) Zhlukovanie rezíduí metódou k-means



(c) Zhlukovanie rezíduí metódou k-medoids

Obr. 34: Rozdelenie rezíduí do zhlukov pri použití rôznych metód zhlukovania

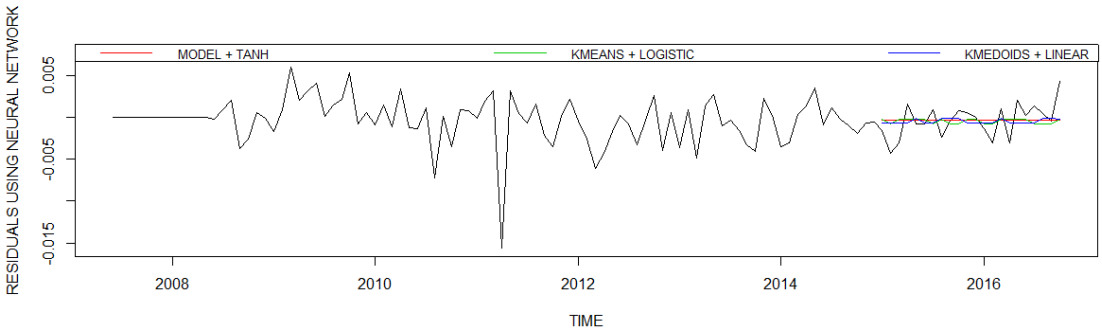
V prvej časti budeme dáta v zhlukoch modelovať neurónovou sieťou. Narozdiel od iných častí práce, nebudeme využívať funkciu *nnetar* zabudovanú v knižnici *forecast* [12], ale využijeme funkciu *nnet* z knižnice *nnet* [32]. Prvá z menovaných funkcií je priamo určená na modelovanie časových radov. Rozdelenie dát do zhlukov, ktorému sa venujeme v tejto časti však neumožňuje dáta v každom zhluku modelovať ako časový rad, pretože nepracujeme s celým radom, ale len s jeho vybranými hodnotami. Rovnako

vytvárame v tejto časti pre každý zhluk model (použitím funkcie *train* z knižnice *caret* [13]) pre každý typ aktivačnej funkcie, čím dostávame veľké množstvo výsledných modelov. Nebudeme teda robiť analýzu každého z nich, zhrnieme iba výsledné chyby pre každý typ aktivačnej funkcie. Rovnako v tejto časti nerobíme validáciu vytvoreného modelu. Zdrojový kód pre jednu z metód zhlukovania, konkrétne *metódu k-medoids*, sme zhrnuli v Prílohe C. V Tabuľke 16 vidíme výsledné chyby predikcií jednotlivých modelov.

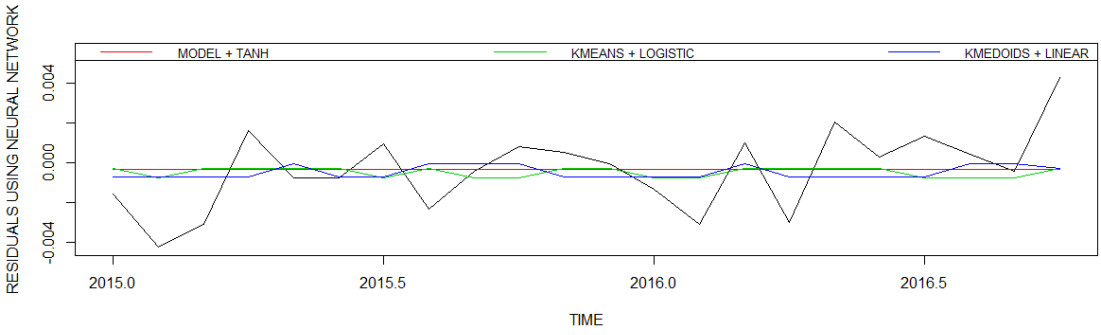
Typ použitej aktivačnej funkcie	K-MEANS	K-MEDOIDS	MODEL
lineárna	3.790733e-06	3.725369e-06	3.848965e-06
logistická	3.790516e-06	3.725574e-06	3.848966e-06
Hyperbolický tangens(Sigmoid)	3.725603e-06	3.790562e-06	3.848965e-06

Tabuľka 16: Predikcie miery zlyhania klientov neurónovou sieťou pri rozdelení rezíduí do zhlukov

Tabuľka 16 ukazuje zhlukovanie metódou k-medoids ako najvhodnejšiu pre modelovanie neurónovou sieťou, naopak modelové zhlukovanie sa javí ako najmenej kvalitné z pohľadu predikcií. V porovnaní s chybou jednokrokových predikcií ARMA modelu, ktoré mali chybu 3.983691e-06, sa podarilo tieto predikcie vylepšiť pre všetky typy zhlukovania a všetky typy aktivačnej funkcie neurónovej siete.
Na Obr.35 porovnávame jednotlivé spôsoby zhlukovania, pričom každý typ zhlukovania zobrazíme použitím aktivačnej funkcie minimalizujúcej výslednú chybu.



(a) Priebeh časového radu



(b) Detailné zobrazenie predikcií

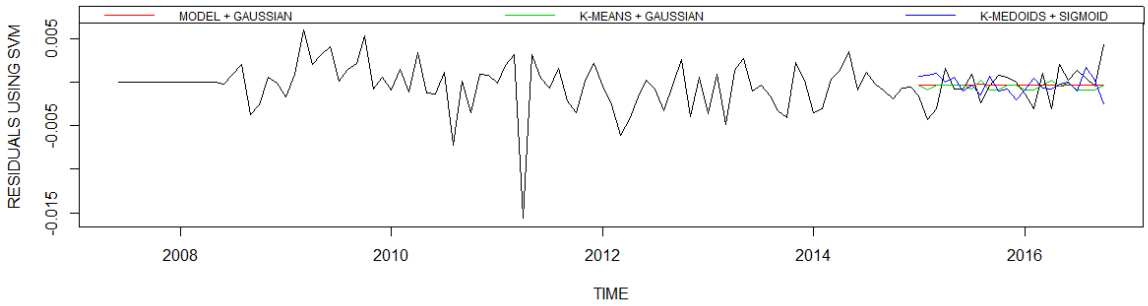
Obr. 35: Jednokrokové predikcie rezíduí ARMA modelu neurónovou sieťou pre všetky typy aktivačnej funkcie v porovnaní so skutočnými rezíduami

V druhej časti využijeme rovnaké zhlukovanie a zmeníme metódu zhlukovania rezíduí. Použijeme metódu oporných bodov. Porovnávať budeme kvalitu predikcií pre všetky metódy zhlukovania a pre všetky typy kernel funkcií. Výsledné chyby predikcií pri použití týchto metód môžeme vidieť v nasledujúcem Tabuľke 17.

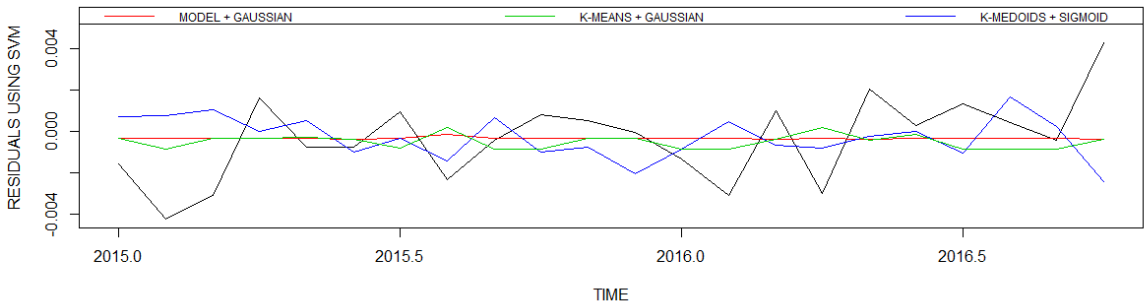
Typ použitej kernel funkcie	KMEANS	KMEDIANS	MODEL
lineárna	6.266115e-06	7.059509e-06	4.102368e-06
polynomiálna	5.395663e-06	1.201647e-05	4.900005e-06
Gaussovská	4.093187e-06	7.875607e-06	3.888793e-06
Hyperbolický tangens(Sigmoid)	6.529481e-06	6.061669e-06	4.845045e-06

Tabuľka 17: Predikcie miery zlyhania klientov metódou oporných bodov pri rozdelení rezíduí do zhlukov

Veľkosť chyby predikcií, ktorá je obsahom Tabuľky 17, nasvedčuje, že najkvalitnejšie predikcie sme získali použitím normálneho modelu ako metódy zhlukovania. Chyba predikcií tejto zhlukovacej metódy je najmenšia pre všetky typy kernel funkcie. Metóda oporných bodov spolu so zhlukovaním rezíduí sa nejaví ako optimálna pre naše dáta, pretože výsledná chyba predikcií je väčšia ako chyba jednokrokových ARMA predikcií, ktorá je 3.983691e-06. Chybu ARMA predikcií sa nám podarilo zmenšiť iba použitím gaussovskej kernel funkcie a normálneho modelu ako metódy zhlukovania. Na Obr. 36 porovnáme jednotlivé spôsoby zhlukovania, pričom každý typ zhlukovania zobrazíme použitím kernel funkcie, ktorá minimalizuje výslednú chybu predikcií.



(a) Priebeh časového radu

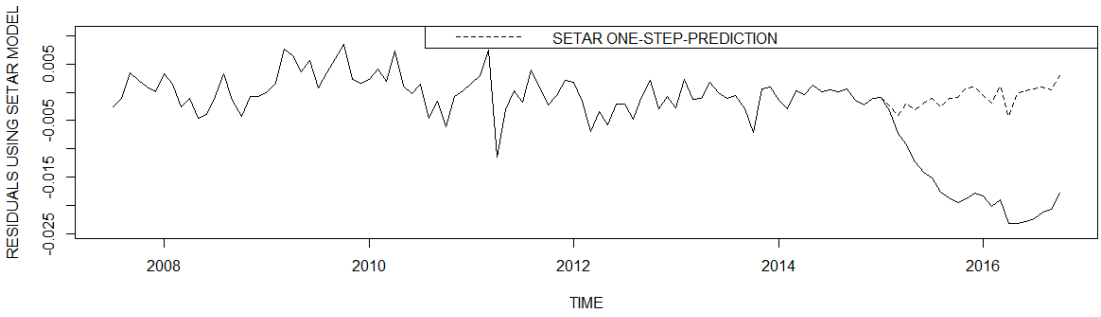


(b) Detailné zobrazenie predikcií

Obr. 36: Jednokrokové predikcie rezíduí ARMA modelu neurónovou sieťou pre všetky typy aktivačnej funkcie v porovnaní so skutočnými rezíduami

3.3.4 SETAR modely

V tejto časti budeme modelovať dáta opäť rozdelením do zhlukov. Narozdiel od časti 3.3.3, nebudeme zhlukovať rezíduá, ale pôvodné dáta. Na toto zhlukovanie však využijeme SETAR modely, priamo určené na modelovanie časových radov rozdelených do režimov. Pri spôsobe výberu optimálnych vstupných parametrov i samotného použitia modelov sme sa inšpirovali publikáciami [5] a [24]. Predikcie SETAR modelu sme spravili naraz i jednokrokovými predikciami. Rezíduá týchto predikcií, ktoré budú predmetom modelovania nelineárnymi metódami, sú zobrazená na Obr. 37.



Obr. 37: Predikcie rezíduí miery zlyhania klientov použitím SETAR modelu

Celková chyba predikcií SETAR modelu je 0.0003106091 a pri použití jednokrokových predikcií chyba dosiahla hodnotu 3.800578e-06. Pre porovnanie chyba ARMA predikcií bola 0.0002823432 a chyba jednokrokových ARMA predikcií bola 3.983691e-06. Výhodou SETAR rezíduí je fakt, že v týchto rezíduách by sa už nemal nachádzať vplyv krízy. Ten sme odfiltrovali režimovým modelovaním SETAR modelmi.

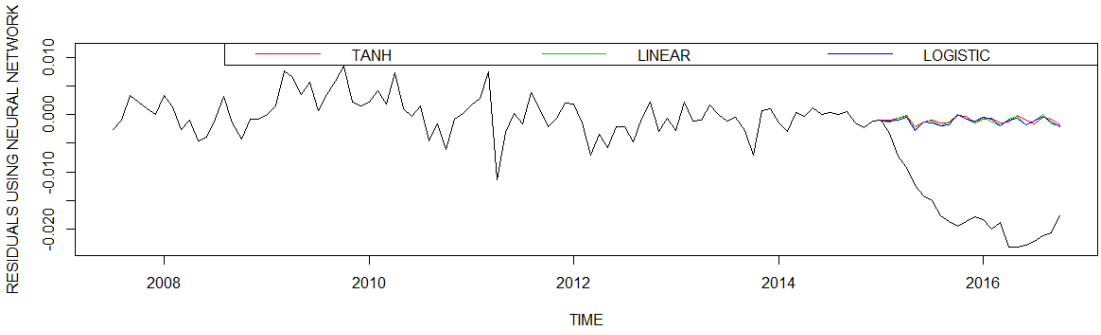
3.3.4.1 Neurónová sieť

Rezíduá definované v časti 3.3.4 budeme modelovať najskôr pomocou neurónovej siete v jednom kroku. Optimálny počet prvom skrytej vrstvy sme odhadli na validačných dátach. Tento počet, chyba predikcií validačnej vzorky, ako aj chyba výsledných predikcií sú obsiahnuté v Tabuľke 18.

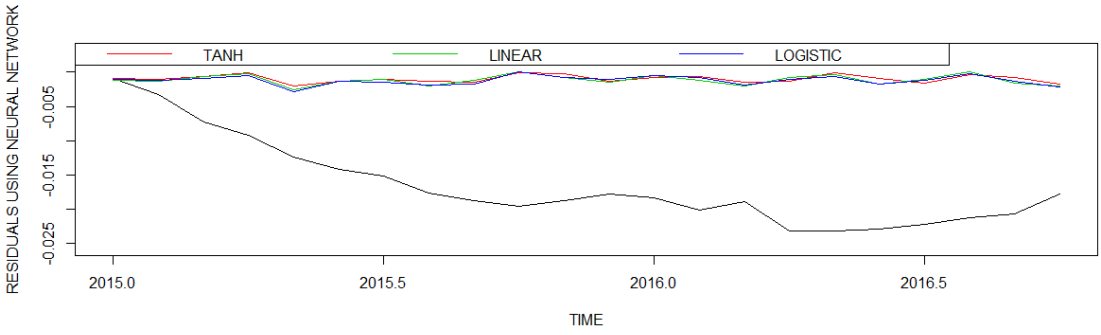
Aktivačná funkcia	Chyba valid. vzorky	Prvky skrytej vrstvy	Chyba predikcií
lineárna	0.0001710944	58	0.0001002849
logistická	0.0001695494	67	9.549112e-05
Hyperbolický tangens(Sigmoid)	0.0001682714	32	9.721157e-05

Tabuľka 18: Chyby neurónovej siete pri predikovaní v jednom kroku

V Tabuľke 18 vidíme, že chybu SETAR modelu sa podarilo vylepšiť pre všetky typy aktivačnej funkcie. Kvalita kombinovaných predikcií SETAR modelu a rezíduí neurónovej siete je lepšia aj v porovnaní s kombinovanými predikciami ARMA modelov. Predikcie SETAR rezíduí neurónovou sieťou sme graficky znázornili na Obr. 38.



(a) Priebeh časového radu



(b) Detailné zobrazenie predikcií

Obr. 38: Predikcie rezíduí SETAR modelu neurónovou sieťou pre rôzne typy aktivačnej funkcie v porovnaní so skutočnými rezíduami

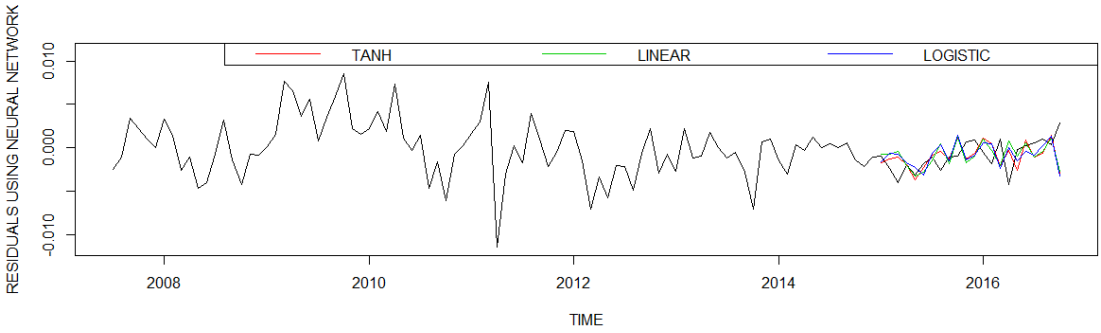
Nakoľko jednokrokové predikcie SETAR modelu majú rádovo nižšiu chybu ako predikcie v jednom kroku, pokúsime sa vylepšiť aj jednokrokové SETAR predikcie. Chyby

predikcií spolu s optimálnym počtom prvom v skrytej vrstve sú zhrnuté v Tabuľke 19.

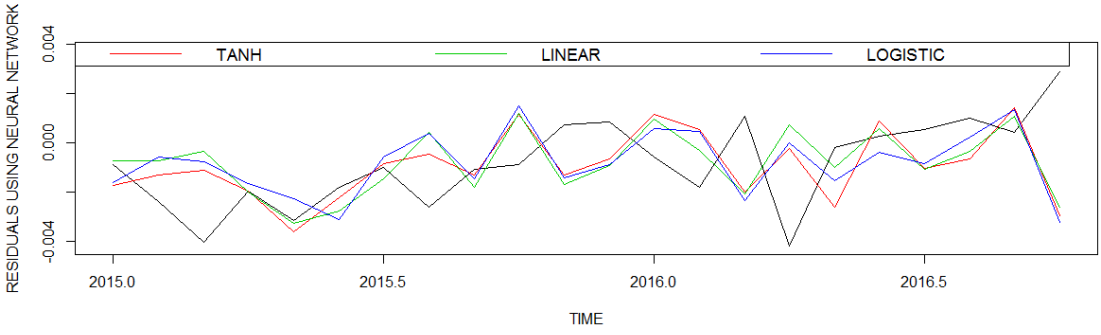
Aktivačná funkcia	Chyba valid. vzorky	Prvky skrytej vrstvy	Chyba predikcií
lineárna	2.376943e-06	38	5.281916e-06
logistická	2.333189e-06	56	5.388433e-06
Hyperbolický tangens(Sigmoid)	2.200256e-06	38	4.84272e-06

Tabuľka 19: Chyby neurónovej siete pri predikovaní využitím jednokrokových predikcií

Z Tabuľky 19 vidíme, že SETAR predikcie sa nám nepodarilo vylepšiť ani pre jednu z aktivačných funkcií. Chyba SETAR predikcií je však rádovo nižšia ako v prípade modelovania rezíduí ARMA modelu. Jednokrokové predikcie SETAR rezíduí modelované neurónovou sieťou sme graficky znázornili na Obr. 39.



(a) Priebeh časového radu



(b) Detailné zobrazenie predikcií

Obr. 39: Jednokrokové predikcie rezíduí SETAR modelu neurónovou sieťou pre rôzne typy aktivačnej funkcie v porovnaní so skutočnými rezíduami

3.3.4.2 Metóda oporných bodov

V tejto časti budeme opäť využívať 12-mesačnú informačnú maticu. Budeme tak robiť jednokrokové predikcie. Pomocou *tune.svm* sme parametre funkcie *svm* pre všetky typy kernel funkcie odhadli na hodnoty uvedené v Tabuľke 20.

Optimálne hodnoty parametrov				
Typ použitej kernel funkcie	γ	c_0	ϵ	Penalizačná konštanta
lineárna	/	/	0,6	1
polynomiálna	0,1	0,1	0,1	10
Gaussovská	1	/	0,1	10
Hyperbolický tangens(Sigmoid)	0	0,7	0,6	100000

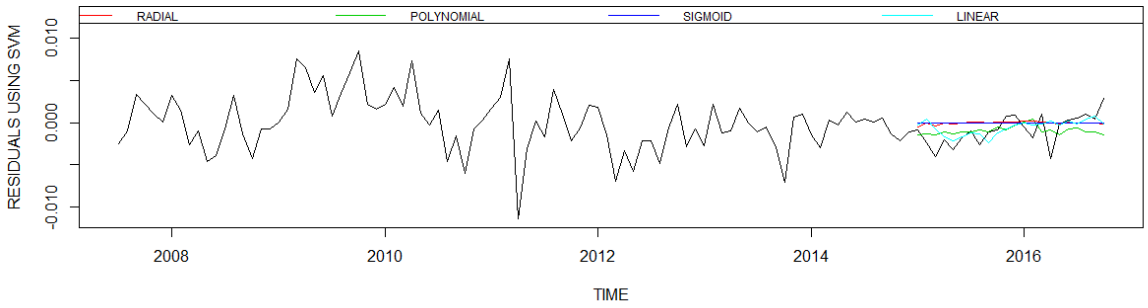
Tabuľka 20: Vstupné hodnoty pri použití rôznych typov kernel funkcie

Použitím hodnôt uvedených v Tabuľke 20 sme spravili predikcie rezíduí SETAR modelu. Výsledné chyby týchto predikcií dosiahli hodnoty uvedené v Tabuľke 21.

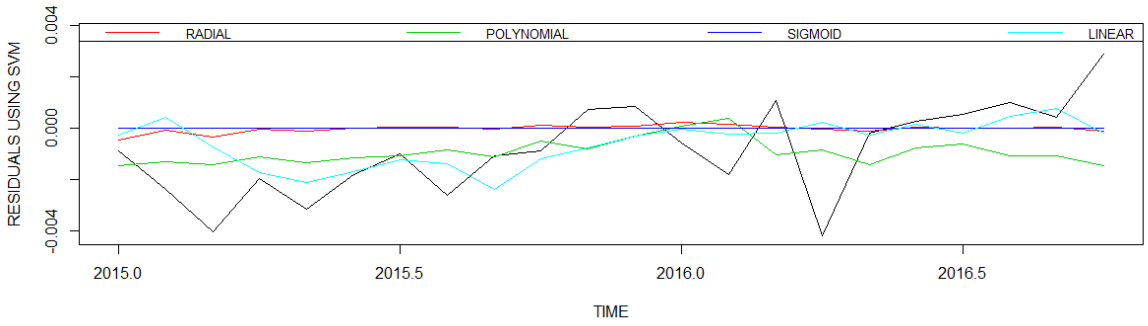
Typ použitej kernel funkcie	Chyba modelu	Chyba predikcií
lineárna	9.761009e-06	2.824859e-06
polynomiálna	2.622898e-06	3.198703e-06
Gaussovská	1.31595e-06	3.648808e-06
Hyperbolický tangens(Sigmoid)	1.197614e-05	3.782704e-06

Tabuľka 21: Chyby modelu a predikcií pri použití rôznych typov kernel funkcie

Predikcie SETAR rezíduí získané modelovaním neurónovou sieťou sme graficky znázornili na Obr. 40.



(a) Priebeh časového radu



(b) Detailné zobrazenie predikcií

Obr. 40: Predikcie rezíduí SETAR modelu metódou oporných bodov pre všetky typy kernel funkcií v porovnaní so skutočnými rezíduami

Z Tabuľky 21 vidíme, že metóda oporných bodov umožnila vylepšiť predikcie SETAR modelu. Naviac, predikcie sme vylepšili pre všetky typy kernel funkcií. Hoci v prípade použitia funkcie SETAR pozorujeme väčšiu chybu modelu ako pre funkciu ARMA, chyba výsledných predikcií je menšia pri použití SETAR modelovania. Táto metóda zhlukovania sa teda javí ako lepšia v porovnaní so zhlukovou analýzou využitou v časti 3.3.3.

Záver

V tejto diplomovej práci sme sa zaoberali skvalitnením predikcií lineárnych ARMA modelov. Tieto modely sa v praxi bežne využívajú na modelovanie časových radov. Ich hlavnou výhodou je jednoduchosť interpretovateľnosti výsledkov na úkor kvality predikcií. V práci sme sa pokúsili tieto predikcie zlepšiť modelovaním rezíduí ARMA modelov.

Na modelovanie rezíduí sme využili neurónovú sieť a metódu oporných bodov. Dáta sme rozdelili do troch skupín, tréningovú, validačnú a testovaciu skupinu dát. Pri vytváraní modelov sme za kľúčovú považovali voľbu vstupných parametrov. Model sme najskôr vytvorili na tréningovej vzorke dát pre viaceré vstupné parametre. Ako optimálne vstupné parametre sme volili tie, ktoré minimalizovali chybu predikcií validačnej vzorky dát. Následne sme pre vybraný model vytvorili predikcie na validačnej a testovacej vzorke. Chybu týchto predikcií sme v závere porovnali s kvalitou predikcií ARMA modelov.

Kľúčovým sa pre nás stal vývoj rezíduí lineárneho modelu. Na konkrétnych príkladoch sme pozorovali, že rezíduá ARMA modelu z časti odhadu modelu sa výrazne líšili od rezíduí predikcií ARMA modelu. To znamená, že rezíduá ako časový rad, sa v predikovanom období vyvíjali odlišne od očakávaní vyplývajúcich z tréningovej vzorky. Časový rad s takýmto vývojom je vo všeobecnosti ťažko predikovateľný, o čom sme sa presvedčili aj v našej práci. Z dôvodu tohto vychýlenia rezíduí v predikovanom období sa nám ich nepodarilo namodelovať tak, aby sme vylepšili výsledné predikcie.

Zmenu časového vývoja rezíduí v období predikcií sme sa pokúsili vyriešiť jednokrokovými predikciami. V tomto prípade už jednokrokové predikcie ARMA modelu dosahovali rádovo nižšiu chybu predikcií. Túto chybu sa nám následne podarilo ešte vylepšiť nelineárnymi metódami pre niektoré špecifické prípady. Vo všeobecnosti sme však ani v tomto prípade nevedeli zaručiť skvalitnenie ARMA predikcií modelovaním rezíduí ARMA modelu neurónovou sieťou alebo metódou oporných bodov.

O niečo uspokojujúcejšie výsledky sme dosiahli v prípade modelovania časového radu, v ktorom sme pozorovali horizontálny posun v časovom vývoji. Predikcie pre časový rad tohto typu sa nám podarilo výrazne vylepšiť v prípade nahradenia lineárneho ARMA modelu režimovým SETAR modelom. Následné modelovanie rezíduí SETAR modelu

pôvodným spôsobom dokázalo tieto predikcie ešte dodatočne čiastočne vylepšiť.

Prínosom práce bola aplikácia známych metód neurónovej siete a metódy oporných bodov na rezíduá ARMA modelu. Výsledky našej analýzy nasvedčujú, že použitím nelineárnych metód sa nepodarilo rezíduá efektívne predikovať. Cieľ našej práce sa nám tak nepodarilo splniť. Ako možné riešenie pre skvalitnenie ARMA predikcií sme demonštrovali efektivitu využitia jednokrokových predikcií a režimových modelov v prípade, že majú opodstatnenie.

Zoznam použitej literatúry

[1] *Artificial Intelligence - Neural Networks*, dostupné na internete (22.11.2016): https://www.tutorialspoint.com/artificial_intelligence/artificial_intelligence_neural_networks.htm

[2] Cantaluppi, G.: *Computational Laboratory for Economics*, Notes for the students, EDUCatt, Milano, 2013

[3] Cimermanová, K.: *Štatistické metódy a algoritmy na výskum molekulovo orientovanej diagnostiky plúcnych chorôb*, Písomná práca k dizertačnej skúške, Slovenská akadémia vied, Bratislava, 2007, dostupné na internete: http://www.um.sav.sk/sk/images/stories/dep03/doc/minimovka

[4] *Convolutional Neural Networks for Visual Recognition*, dostupné na internete (22.11.2016): http://cs231n.github.io/neural-networks-1/

[5] Di Narzo,A.F.:*Nonlinear autoregressive time series models in R*, vignette, 2006, dostupné na internete(23.4.2017):https://cran.r-project.org/web/packages/tsDyn/

[6] Fraley,Ch., Raftery,A.E.: *Model-based Clustering, Discriminant Analysis and Density Estimation*,Journal of the American Statistical Association(2002), 611-631

[7] Fuller, W. A.: *Introduction to Statistical Time Series*, second edition, John Wiley & Sons, Iowa State University, Canada, 1996

[8] Gibson, D., Nur, D.: *Threshold Autoregressive Models in Finance: A Comparative Approach*, Proceedings of the Fourth Annual ASEARC Conference (2011), University of Western Sydney, Australia, dostupné na internete (24.4.2017): http://ro.uow.edu.au/cgi/viewcontent.cgi?article=1025&context=asearc

[9] Han, J., Pei, J., Kamber, M.: *Data Mining*, Southeast Asia Edition, University of Illinois at Urbana-Champaign, Morgan Kaufmann, 2006

[10] Harman, R.: *Searching for information hidden in multivariate data*, učebné texty, FMFI UK, Bratislava, 2011, dostupné na internete (25.4.2017): http://www.iam.fmph.uniba.sk/ospm/Harman/HABT05.pdf

[11] Harvey, A.C.: *Forecasting, Structural Time Series Models and the Kalman Filter*,Cambridge University Press, Cambridge, United Kingdom, 1989

[12] Hyndman, RJ.:*forecast: Forecasting functions for time series and linear models*, R package version 7.2, 2016, dostupné na internete(12.1.2017): http://github.com/robjhyndman/forecast

[13] Kuhn,M. et al.:*caret: Classification and Regression Training*, R package version 6.0-73, 2016, dostupné na internete(6.5.2017): https://CRAN.R-project.org/package=caret

[14] Leisch, F.: *A Toolbox for K-Centroids Cluster Analysis*, Computational Statistics and Data Analysis, 51 (2), 526-544, 2006

[15] Liu, D., Fei, S., Hou, Z.G. a spol.: *Advances in Neural Networks*, Springer Science & Business Media, Nanjing, China, 2007

[16] McTaggart,R., Daroczi, G., Leung, C.: *Quandl: API Wrapper for Quandl.com*, R package version 2.7.0., 2015, dostupné na internete (1.2.2017): https://CRAN.R-project.org/package=Quandl

[17] Meyer, D. et al.: *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071)*, TU Wien, R package version 1.6-7, 2015, dostupné na internete (1.5.2017): https://CRAN.R-project.org/package=e1071

[18] Morissette, L., Chartier, S.: *The k-means clustering technique: General considerations and implementation in Mathematica*, Tutorials in Quantitative Methods for Psychology, 2013, dostupné na internete (5.3.2017): http://www.tqmp.org/RegularArticles/vol09-1/p015/p015.pdf

[19] *Nerves, Neurons, Axons and Dendrites by Example*, dostupné na internete (6.1.2017): http://biology.stackexchange.com/questions/25967/nerves-neurons-axons-and-dendrites-by-example

[20] *Partial Autocorrelation Function (PACF)*, dostupné na internete (20.11.2016): https://onlinecourses.science.psu.edu/stat510/node/62

[21] Pfaff, B.: *Analysis of Integrated and Cointegrated Time Series with R*, Second Edition, Springer, New York, 2008, ISBN 0-387-27960-1

[22] *Practical machine learning*, online kurz, dostupné na internete (10.10.2016): https://www.coursera.org/learn/practical-machine-learning

[23] *PrtKernel*, dostupné na internete (3.1.2017): http://covartech.github.io/blog/2013/04/22/prtkernel/

[24] Stigler, M.:*Threshold cointegration: overview and implementation in R*, vignette, dostupné na internete(2.4.2017):http://cran.r-project.org/web/packages/tsDyn/

[25] Stoffer, D: *astsa: Applied Statistical Time Series Analysis*, R package version 1.4, 2016, dostupné na internete (1.5.2017): https://CRAN.R-project.org/package=astsa

[26] *Sumár úhrnov zrážok (mm)*, dostupné na internete (12.3.2017): http://pocasiemost.vija.sk/wxrainsummary.php

[27] *Support Vector Machine Regression*, dostupné na internete (8.1.2017): http://kernelsvm.tripod.com/

[28] *Support Vector Machines*, dostupné na internete (23.4.2017): http://scikit-learn.org/stable/modules/svm.html

[29] *Support Vector Regression with R*, dostupné na internete (23.4.2017): https://www.svm-tutorial.com/2014/10/support-vector-regression-r/

[30] *Using neural nets to recognize handwritten digits*, dostupné na internete (22.11.2016): http://neuralnetworksanddeeplearning.com/chap1.html

[31] Vapnik, V.: *The Nature of Statistical Learning Theory*, Springer, New York, 1995

[32] Venables, W. N., & Ripley, B. D.: *Modern Applied Statistics with S*, Fourth Edition, Springer, New York,2002, ISBN 0-387-95457-0

[33] Wuertz,D. et al: *fArma: ARMA Time Series Modelling*, R package version 3010.79, 2013, dostupné na internete(5.1.2017): https://CRAN.R-project.org/package=fArma

Príloha A

```
#neuronova siet + data rozdelene do troch skupin
chyba.nnetar1.valid<-matrix(rep(NA,21*4), nrow=4)
i<-1
for (s in (length(train1)/4):(length(train1)*3/4)){
    nnetar.fit1__1<-nnetar(arma.res1,p=length(arma.res1)/frekvencia1,P=,size=s,act.fct="linear",repeats=50)
    nnetar.fit1__2<-nnetar(arma.res1,p=length(arma.res1)/frekvencia1,P=,size=s,act.fct="tanh",repeats=50)
    nnetar.fit1__3<-nnetar(arma.res1,p=length(arma.res1)/frekvencia1,P=,size=s,act.fct="logistic",repeats=50)

    nnetar.forecast1.valid.lin<-forecast(nnetar.fit1__1,h=length(valid1))
    nnetar.forecast1.valid.tan<-forecast(nnetar.fit1__2,h=length(valid1))
    nnetar.forecast1.valid.log<-forecast(nnetar.fit1__3,h=length(valid1))

    nnetar.valid1.lin<-arma.forecast1$pred[1:length(valid1)]+nnetar.forecast1.valid.lin$mean
    nnetar.valid1.tan<-arma.forecast1$pred[1:length(valid1)]+nnetar.forecast1.valid.tan$mean
    nnetar.valid1.log<-arma.forecast1$pred[1:length(valid1)]+nnetar.forecast1.valid.log$mean

    chyba.nnetar1.valid[1,i]<-s
    chyba.nnetar1.valid[2,i]<-crossprod(nnetar.valid1.lin-valid1)/length(valid1)
    chyba.nnetar1.valid[3,i]<-crossprod(nnetar.valid1.tan-valid1)/length(valid1)
    chyba.nnetar1.valid[4,i]<-crossprod(nnetar.valid1.log-valid1)/length(valid1)
    i<-i+1
}
nnetar.best1.lin.valid<-which(chyba.nnetar1.valid==min(chyba.nnetar1.valid[2,]), arr.ind = TRUE) #s=29
nnetar.best1.tan.valid<-which(chyba.nnetar1.valid==min(chyba.nnetar1.valid[3,]), arr.ind = TRUE) #s=28
nnetar.best1.log.valid<-which(chyba.nnetar1.valid==min(chyba.nnetar1.valid[4,]), arr.ind = TRUE) #s=29

#mame najlepsie parametre,predikovanie celej treningovej vzorky naraz
##vytvorime model a jeho predikcie s pouzitim hodnot minimalizujucich chybu
nnetar.fit1.lin.valid<-nnetar(arma.res1,p=length(arma.res1)/frekvencia1,P =,size=29,act.fct="linear",repeats=50)
nnetar.fit1.tan.valid<-nnetar(arma.res1,p=length(arma.res1)/frekvencia1,P =,size=28,act.fct="tanh",repeats=50)
nnetar.fit1.log.valid<-nnetar(arma.res1,p=length(arma.res1)/frekvencia1,P=,size=29,act.fct="logistic",repeats=50)

nnetar.forecast1.valid.lin<-forecast(nnetar.fit1.lin.valid,h=length(valid.test1))
nnetar.forecast1.valid.tan<-forecast(nnetar.fit1.tan.valid,h=length(valid.test1))
nnetar.forecast1.valid.log<-forecast(nnetar.fit1.log.valid,h=length(valid.test1))

nnetar.valid1.lin<-arma.forecast1$pred+nnetar.forecast1.valid.lin$mean
nnetar.valid1.tan<-arma.forecast1$pred+nnetar.forecast1.valid.tan$mean
nnetar.valid1.log<-arma.forecast1$pred+nnetar.forecast1.valid.log$mean

(chybaarma<-crossprod(arma.forecast1$pred-valid.test1)/length(valid.test1))
(chybafit.nnetar.lin.valid<-crossprod(nnetar.valid1.lin-valid.test1)/length(valid.test1))
(chybafit.nnetar.tan.valid<-crossprod(nnetar.valid1.tan-valid.test1)/length(valid.test1))
(chybafit.nnetar.log.valid<-crossprod(nnetar.valid1.log-valid.test1)/length(valid.test1))
```

Príloha B

```
# modelovanie rezidui pomocou metody opornych bodov
# odhad vstupnych parametrov
info<-12
(tuneResult3.pol<-tune(svm, train.historia3.df$y ~.,
data = train.historia3.df, validation.x = valid.historia3.df, validation.y = valid.historia3.df.y, kernel='polynomial',
    ranges = list(epsilon=seq(0.1,1,0.1), cost=10^(1:5), gamma = seq(0,1,0.1), coef0 = seq(0,1,0.1)),type = "eps-
    regression"))
(tuneResult3.sig<-tune(svm, train.historia3.df$y ~.,
data = train.historia3.df, validation.x = valid.historia3.df, validation.y = valid.historia3.df.y, kernel='sigmoid',
    ranges = list(epsilon=seq(0.1,1,0.1), cost=10^(1:5), gamma = seq(0,1,0.1), coef0 = seq(0,1,0.1)),type = "eps-
    regression"))
(tuneResult3.rad<-tune(svm, train.historia3.df$y ~.,
data = train.historia3.df, validation.x = valid.historia3.df, validation.y = valid.historia3.df.y, kernel='radial',
    ranges = list(epsilon=seq(0.1,1,0.1), cost=10^(1:5), gamma = seq(0,1,0.1)),type = "eps-regression"))
(tuneResult3.lin<-tune(svm, train.historia3.df$y ~.,
data = train.historia3.df, validation.x = valid.historia3.df, validation.y = valid.historia3.df.y, kernel='linear',
    ranges = list(epsilon=seq(0.5,1,0.1), cost=1^(1:10)),type = "eps-regression"))

#tvorba modelov
svm.model3.osp.pol<-svm(train.historia3.df$y ~ ., data=train.historia3.df, kernel='polynomial', epsilon = 0.2, cost
    = 10, gamma=0.1, coef0=0.2, type = "eps-regression")
svm.model3.osp.sig<-svm(train.historia3.df$y ~ ., data=train.historia3.df, kernel='sigmoid', epsilon = 0.2, cost =
    10000, gamma=0,coef0=0.5, type = "eps-regression")
svm.model3.osp.rad<-svm(train.historia3.df$y ~ ., data=train.historia3.df, kernel='radial', epsilon = 0.1, cost =
    100, gamma=0.6, type = "eps-regression")
svm.model3.osp.lin<-svm(train.historia3.df$y ~ ., data=train.historia3.df, kernel='linear', epsilon = 0.5, cost = 1,
    type = "eps-regression")

#predikcie
svm.osp.forecast3.pol<-predict(svm.model3.osp.pol, valid.test.historia3.df)
svm.arma.osp.forecast3.pol<-arma.forecast3.osp+svm.osp.forecast3.pol
svm.osp.forecast3.sig<-predict(svm.model3.osp.sig, valid.test.historia3.df)
svm.arma.osp.forecast3.sig<-arma.forecast3.osp+svm.osp.forecast3.sig
svm.osp.forecast3.rad<-predict(svm.model3.osp.rad, valid.test.historia3.df)
svm.arma.osp.forecast3.rad<-arma.forecast3.osp+svm.osp.forecast3.rad
svm.osp.forecast3.lin<-predict(svm.model3.osp.lin, valid.test.historia3.df)
svm.arma.osp.forecast3.lin<-arma.forecast3.osp+svm.osp.forecast3.lin

#vypocet chyby
(chybaarma<-crossprod(arma.forecast3.osp-valid.test3)/length(valid.test3))
(chybafit.svm.osp.rad<-crossprod(svm.arma.osp.forecast3.rad-valid.test3)/length(valid.test3))
(chybafit.svm.osp.pol<-crossprod(svm.arma.osp.forecast3.pol-valid.test3)/length(valid.test3))
(chybafit.svm.osp.sig<-crossprod(svm.arma.osp.forecast3.sig-valid.test3)/length(valid.test3))
(chybafit.svm.osp.lin<-crossprod(svm.arma.osp.forecast3.lin-valid.test3)/length(valid.test3))
```

Príloha C

```
#k-medoids
cluster.mclust.train_2<-kcca(train.historia6[,2:(info+1)], k=3, kccaFamily("kmedians"))
cluster.mclust.train_2.1<-predict(cluster.mclust.train_2)
predikcia6_2<-rep(NA, length(valid.test6))
prediction.data6_2<-data.frame(x = valid.test.clust[,2:(info+1)])
(predikcia6_2<-predict(cluster.mclust.train_2,newdata = valid.test.historia6[,2:(info+1)]))

clusters_2<-c(cluster.mclust.train_2.1,predikcia6_2)
prefered. classification _2<-clusters_2
#zhlukova analyza-KMEDIANS + nnet
my.grid <- expand.grid(.decay = c(0.5, 0.1), .size = c(5, 6, 7))
fit6.nnet.clust1.log_2<-train(y ~ ., data=DF1_2, method = "nnet", act.fct= "logistic", tuneGrid = my.grid,
    maxit = 1000, trace = F, linout = 1, rep = 10)
fit6.nnet.clust1.lin_2<-train(y ~ ., data=DF1_2, method = "nnet", act.fct= "linear", maxit = 1000, trace = F,
    linout = 1)
fit6.nnet.clust1.tan_2<-train(y ~ ., data=DF1_2, method = "nnet", act.fct= "tanh", maxit = 1000, trace = F,
    linout = 1)

fit6.nnet.clust2.log_2<-train(y ~ ., data=DF2_2, method = "nnet", act.fct= "logistic", tuneGrid = my.grid,
    maxit = 1000, trace = F, linout = 1, rep = 10)
fit6.nnet.clust2.lin_2<-train(y ~ ., data=DF2_2, method = "nnet", act.fct= "linear", maxit = 1000, trace = F,
    linout = 1)
fit6.nnet.clust2.tan_2<-train(y ~ ., data=DF2_2, method = "nnet", act.fct= "tanh", maxit = 1000, trace = F,
    linout = 1)

fit6.nnet.clust3.log_2<-train(y ~ ., data=DF3_2, method = "nnet", act.fct= "logistic", tuneGrid = my.grid,
    maxit = 1000, trace = F, linout = 1, rep = 10)
fit6.nnet.clust3.lin_2<-train(y ~ ., data=DF3_2, method = "nnet", act.fct= "linear", maxit = 1000, trace = F,
    linout = 1)
fit6.nnet.clust3.tan_2<-train(y ~ ., data=DF3_2, method = "nnet", act.fct= "tanh", maxit = 1000, trace = F,
    linout = 1)

prediction.cluster.nnet.log_2<-prediction.cluster.nnet.tan_2<-prediction.cluster.nnet.lin_2<-rep(NA, length(
    valid.test6))
for (p in 1:length(valid.test6)){
if (predikcia6_2[p]=="1"){
prediction.cluster.nnet.log_2[p]<-predict(fit6.nnet.clust1.log_2,valid.test.clust.df.x_2[p,])
prediction.cluster.nnet.lin_2[p]<-predict(fit6.nnet.clust1.lin_2,valid.test.clust.df.x_2[p,])
prediction.cluster.nnet.tan_2[p]<-predict(fit6.nnet.clust1.tan_2,valid.test.clust.df.x_2[p,])
}
if (predikcia6_2[p]=="2"){
prediction.cluster.nnet.log_2[p]<-predict(fit6.nnet.clust2.log_2,valid.test.clust.df.x_2[p,])
prediction.cluster.nnet.lin_2[p]<-predict(fit6.nnet.clust2.lin_2,valid.test.clust.df.x_2[p,])
prediction.cluster.nnet.tan_2[p]<-predict(fit6.nnet.clust2.tan_2,valid.test.clust.df.x_2[p,])
}
if (predikcia6_2[p]=="3"){
```

```
prediction . cluster . nnet.log__2[p]<-predict(fit6.nnet.clust3.log__2,valid.test.clust.df.x__2[p,])
prediction . cluster . nnet. lin__2[p]<-predict(fit6.nnet.clust3.lin__2,valid.test . clust . df.x__2[p,])
prediction . cluster . nnet.tan__2[p]<-predict(fit6.nnet.clust3.tan__2,valid.test.clust.df.x__2[p,])
}}

cluster . nnet.pred6.log__2<-arma.forecast6.osp+prediction.cluster.nnet.log__2
cluster . nnet.pred6. lin__2<-arma.forecast6.osp+prediction.cluster.nnet.lin__2
cluster . nnet.pred6.tan__2<-arma.forecast6.osp+prediction.cluster.nnet.tan__2

(chybaarma.osp<-crossprod(arma.forecast6.osp-valid.test6)/length(valid.test6))
(chybafit . cluster . nnet.osp.log__2<-crossprod(cluster.nnet.pred6.log__2-valid.test6)/length(valid.test6))
(chybafit . cluster . nnet.osp. lin__2<-crossprod(cluster.nnet.pred6.lin__2-valid.test6)/length(valid.test6))
(chybafit . cluster . nnet.osp.tan__2<-crossprod(cluster.nnet.pred6.tan__2-valid.test6)/length(valid.test6))
```