VYSOKÁ ŠKOLA EKONOMICKÁ V PRAZE
Fakulta financí a účetnictví
Katedra bankovnictví a pojišťovnictví
Studijní obor: Finance

INTELIGENTNÍ SYSTÉMY A PREDIKCE FINANČNÍCH ČASOVÝCH ŘAD

Autor diplomové práce: Bc. Sabína Urbanová

Vedoucí diplomové práce: prof. RNDr. Jiří Witzany, Ph.D.

Rok obhajoby: 2019

Čestné prehlásenie:

Prehlasujem, že diplomovú prácu na tému „Inteligentní systémy a predikce finančních časových řad“ som vypracovala samostatne a všetku použitú literatúru a ďalšie pramene som riadne označila a uviedla v priloženom zozname.

V Prahe dňa 6.1.2019 .....

Podpis

Poďakovanie:

Týmto by som sa chcela poďakovať prof. RNDr. Jiřímu Witzanymu Ph.D.za vedenie mojej diplomovej práce a za cenné rady a pripomienky pri jej spracovaní. Ďalej by som chcela poďakovať mojej rodine a priateľovi za podporu v doterajšom štúdiu

Abstrakt:
Diplomová práca sa zaoberá popisom a zostrojením troch modelov, doprednej neurónovej siete učiacej sa na základe algoritmu spätného šírenia chyby, rekurentnej siete LSTM (long short-term memory) a autoregresný integrovaný kĺzavý priemer ARIMA. K analýze sú použité informácie o otváracích, uzatváracích cenách, najvyššej a najnižšej cene v danom dni a zobchodovaného objemu desiatich akcií obchodovaných na New York Stock Exchange. Dostupné dáta (od januáta 2010 do decembra 2016) sú rozdelené do dvoch množín, na prvej, trénovacej množine prebieha proces učenia, na druhej, testovacej, overujeme predikčnú schopnosť modelu, kde predikovaná hodnota je uzatváracia cena daného dňa. Pre posúdenie, či daný model má pre nás, ako investora, nejaký prínos si definujeme jednoduchú stratégiu -v prípade že predpovedaný denný výnos akcie je kladný (model predpovedá rast uzatváracej ceny v nasledujúcom dni), danú akciu nakupujeme. V prípade, že predpovedaný denný výnos akcie je záporný (predpovedaný pokles uzatváracej ceny), danú akciu predávame. Na konci prebehne zhodnotenie, ktorý model nám za použitia danej stratégie prinesie najlepšie výsledky.

Abstract:
In the diploma thesis we are describing and applying three models - Feedforward neural network, whose learning process is based on backpropagation principles, recurent LSTM (long short-term memory) network and ARIMA autoregressive integrated moving average. For analysis, information about opening, closing price, highest, lowest price of a day and volume of trade are used. Used data (since January 2010 until December 2016) are split into training and testing dataset. On the first one, training dataset, model is learning and on the second, testing dataset, we are making a prediction of closing price. For evaluation, whether a model is useful for investors, we develop a simple strategy – if predicted daily return is positive (model is forecasting a price growth), we buy certain stock. If predicted daily return is negative (model is forecasting a price loss), we sell certain stock. At the end, we evaluate, which model provides us the best results.

Obsah

Úvod 7
1. Strojové učenie 9
1.1 Neurónové siete 10
1.1.1 Neurofyziologický pôvod 10
1.1.2 Matematický model neurónu 12
1.2 Metódy učenia 14
1.2.1 Učenie s učiteľom 14
1.2.2 Problém preučenia siete 16
1.2.3 Učenie bez učiteľa 17
2 Členenie neurónových sietí 19
2.1 Dopredné (Feed-forward) siete 19
2.1.1 Perceptrónové siete 19
2.1.2 Topológia perceptrónu 20
2.1.3 Viacvrstvá perceptrónová sieť 21
2.1.4 Učenie viacvrstvej perceptrónovej siete 22
2.1.5 Algoritmus spätného šírenia chyby (Backpropagation) 22
2.2 Rekurentné neurónové siete 24
2.2.1 Long short-term memory (LSTM) 25
2.2.2 Architektúra LSTM siete 28
3 Box – Jenkinsová metodológia 31
3.1 Stochastický proces 31
3.1.1 Stacionarita 31
3.1.2 Autokorelačná funkcia 32
3.1.3 Parciálna autokorelačná funkcia 32
3.2 Lineárny stochastický proces 34
3.2.1 Autoregresné procesy AR(*p*) 34
3.2.2 Procesy kĺzavých priemerov MA (*q*) 35
3.2.3 Zmiešaný proces ARMA (*p,q*) 35
3.2.4 Integrované procesy ARIMA (*p,d,q*) 36
3.2.5 Predpoveď pomocou ARIMA modelu 37
4 Metriky vyhodnotenia modelu 38
5 Modelovanie 40
5.1 Popis dát 40
5.2 Príprava dát 41

5.2.1 Normalizácia dát ... 41
5.2.2 Transformácia pre učenie s učiteľom ... 42
5.2.3 Rozdelenie dát na trénovaciu a testovaciu množinu ... 44
5.3 Konšrukcia LSTM modelu ... 45
5.3.1 Výber vhodných parametrov LSTM siete ... 45
5.3.2 Architektúra LSTM siete ... 52
5.4 Konštrukcia Backpropagation modelu ... 55
5.4.1 Výber vhodných parametrov Backpropagation siete ... 55
5.5 Arima model ... 60
5.5.1 Test stacionarity časovej rady ... 60
5.5.2 Odstránenie vplyvu trendu ... 61
5.5.3 Určenie počtu parametrov ... 63
5.6 Obchodná stratégia ... 65
6 Výsledky predpovedí ... 66
6.1 Rekapitulácia výsledkov ... 67
6.2 Výsledky obchodnej stratégie ... 68
Záver ... 70
Použitá literatúra ... 71
Príloha ... I

Úvod

Pod slovným spojením *umelá inteligencia* či *strojové učenie* si ešte nie tak dávno väčšina ľudí predstavila scénu zo sci-fi filmu nezodpovedajúcu realite. V posledných rokoch sa však stalo takzvaným „*buzz word*“ - stretávame sa s ním na každom kroku. Strojové učenie, ako podoblasť umelej inteligencie sa zaoberá spôsobmi, ktoré umožňujú technickým systémom „učiť sa“. Z utopických filmov sa výdobytky strojového učenia postupne dostávajú do nášho každodenného života, či už ide o využívanie chatbotov pre komunikáciu s poskytovateľmi produktov a služieb, robotickú automatizáciu procesov, rozoznávanie obrazov a zvukov a mnoho iných.

Nejde však o žiadne vynálezy posledných rokov, niektoré algoritmy využívané pre strojové učenie siahajú do osemdesiatych rokov minulého storočia, kedy vyšlo značné množstvo prelomových vedeckých článkov a veľa z nich nadväzovalo na ešte staršie výskumy.

Cieľom tejto práce bude bližšie sa venovať základným princípom neurónových sieti – metóde strojového učenia, ktorá bola pôvodne inšpirovaná neurobiológiou mozgu a ich využitiu v oblasti predpovede finančných časových radov. Neurónové siete naučíme a použijeme na predikciu vývoja cien akcií desiatich firiem obchodovaných na New York Stock Exchange, konkrétne na jeden model použijeme dopredné neurónové siete a na druhý siete rekurentné. Na rovnakých dátach sa pokúsime o predpoveď pomocou tradičnejšieho, a povedzme o niečo konzervatívnejšieho, modelu ARIMA. Na záver porovnáme výsledky týchto troch modelov a pokúsime sa o zhodnotenie, či do problematiky predpovede na finančných trhoch majú neurónové siete čo priniesť.

S myšlienkou, že dynamická dopredná neurónová sieť by mohla byť použitá k predpovedi budúcej hodnoty premennej prišiel už Smolensky (1986)[1]. Aplikovať na časové rady sa ju pokúsilo niekoľko autorov, medzi nimi napríklad Werbos (1988)[2], ktorý k predpovedi použil proces *backpropagation*, či Lapedes a Farber (1987)[3], ktorí pomocou viacvrstvého perceptrónu predpovedali nelineárne dynamické procesy

[1] SMOLENSKY, Paul. *Information processing in dynamical systems: foundations of Harmony Theory*. 1986.
[2] WERBOS, Paul J. *Generalization of backpropagation with application to a recurrent gas market model*. 1988.
[3] LAPEDES, Alan a Robert FARBER. *How neural nets work*. 1987.

vykazujúce chaotické chovanie. Tang, Almeida, Fishwick (1991)[4], porovnávajú schopnosť predpovede klasického ARIMA modelu a neurónových sieti ANN. V tomto porovnaní vychádza schopnosť predpovede ANN horšie. Z novších prác zmieňme napríklad Zhanga (2011)[5], ktorý porovnáva schopnosť predpovede troch modelov, klasický ARIMA model, ANN model (neurónové siete) a tzv. Hybridný model, teda kombináciu ANN a ARIMA. Zisťuje, že kombinácia oboch modelov produkuje najlepšie výsledky.

Pri hodnotení prínosnosti umelej inteligencie v akejkoľvek oblasti však nesmieme zabúdať ani na potenciálne prekážky. Systémy strojového učenia sú často označované ako tzv. čierne skrinky (angl. *black boxes*), z toho dôvodu, že nie úplne vieme popísať čo sa v nich nachádza, či nie je možné to presne interpretovať.

[4] TANG, Zaiyong, Chrys de ALMEIDA a Paul A. FISHWICK. *Time series forecasting using neural networks vs. Box- Jenkins methodology*. 1991.
[5] ZHANG, G. Peter. *Time series forecasting using a hybrid ARIMA and neural network mode*. 2011.

1. Strojové učenie

Jednou z najdôležitejších vlastností živých organizmov, ktorá je často považovaná za definíciu inteligencie, je schopnosť prispôsobovať sa meniacim sa podmienkam (adaptovať sa), respektíve učiť sa na základe vlastných skúseností. Strojové učenie je výsledkom snahy vybaviť touto schopnosťou i technické systémy.

Rozlišujeme dva typy učenia, a to *učenie sa znalosťami (knowledge acquisition*), kedy hľadáme koncepty a obecné zákonitosti a *učenie sa schopnosťami (skill refinement),* kedy sa na základe precvičovania určitej činnosti snažíme zdokonaliť svoje schopnosti.

V závislosti na tom, akú informáciu dostáva systém o tom, či sa učí správne, rozlišujeme *učenie sa s učiteľom* a *učenie sa bez učiteľa*. V prvom prípade pracujeme s *učiteľom*, ktorý systému poskytuje informácie o požadovanom chovaní. V druhom prípade nemáme k dispozícii *učiteľa*, systém pracuje len s príkladmi a zaujímavé koncepty si vytvára sám. [6]

Medzi najvýznamnejšie metódy strojového učenia patria napríklad:

- rozhodovacie stromy
- rozhodovacie pravidlá
- neurónové siete
- bayesovské siete
- podporné vektory (Support vector machines)
- algoritmus K-najbližších susedov (k-NN)

V nasledujúcej práci sa budeme bližšie venovať práve neurónovým sieťam a rozoberieme si ich základné typy.

[6] BERKA, Petr. *Dobývání znalostí z databází*. Praha: Academia, 2003. 366 s. ISBN 80-200-1062-9.

1.1 Neurónové siete

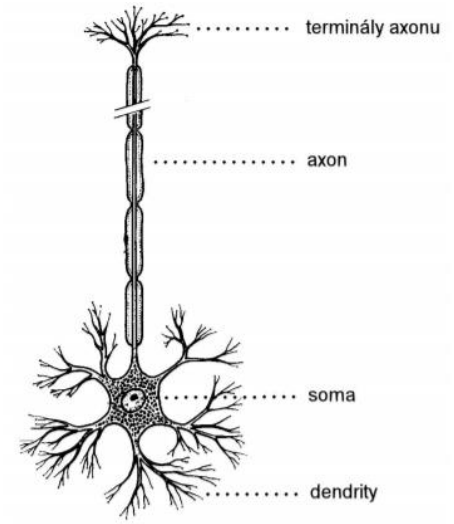
Neurónové siete radíme medzi nedeklaratívne systémy umelej inteligencie zložené z formálnych neurónov, ktoré sú vzájomne prepojené tak, že výstup neurónu je vstupom iných (väčšinou viacerých) neurónov. Nedeklaratívne sú preto, že nemusíme dopredu definovať pravidlá riešenia, tie sa stanovujú až behom učenia.

1.1.1 Neurofyziologický pôvod

Pôvodná snaha neurovied bola vytvoriť taký matematický model, ktorý by vystihoval, akým spôsobom funguje ľudský mozog. Neurofyziológia teda slúžila ako zdroj inšpirácie pri výskume neurónových sietí, avšak dnes známe modely sa nesnažia presne modelovať ľudský mozog, ale snažia sa napodobniť jeho základné funkcie.

Nervová sústava reaguje na vonkajšie (z okolitého prostredia) alebo vnútorne podnety (z vnútra tela), premieňa ich na vzruchy, ktoré následne vedie po celom tele vďaka nervovým bunkám (neurónom). Vzruchy sú následne analyzované v centrálnej nervovej sústave a premieňajú sa na adekvátne odpovede, ktoré potom cez synapsie neurónov, (základnou štruktúrnou a funkčnou jednotkou nervovej sústavy) putujú k výkonnému organu.

Obrázok 1: Biologický neurón



Zdroj: Šíma, Neruda, Teoretické otázky neurónových sítí, 1996., str. 22.

Pojem neurón zahŕňa telo nervovej bunky, zvané somat a jeho výbežky. Tieto výbežky môžu byt dvoch druhov. ***Dendrity,*** teda tie, ktoré prijímajú vstupné informácie, spravidla ich má jeden neurón viacero. ***Axonom*** označujeme výbežok vedúci vzruch smerom od tela neurónu. Z neho obvykle vybočuje množstvo malých vetiev, takzvaných terminálov, ktoré sú zakončené blanou, tá sa dotýka dendritov iných neurónov. K prenosu informácie slúži medzineurónové rozhranie, tzv. ***synapsia.***

Soma i axon sú obalené membránou, ktorá za určitých okolností generuje elektrické impulzy, ktoré sú z axonu prenášané na dendtity iných neurónov synaptickými bránami. Intenzita podráždenia závisí na miere synaptickej priepustnosti. Podľa nej môžeme neuróny rozdeliť na excitačné, umožňujú šírenie nervového vzruchu v nervovej sústave, a na inhibičné, ktoré spôsobujú útlm vzruchu. Synaptická priepustnosť sa po každom priechode signálu mení. [7]

[7] KITTNAR, Otomar. *Lékařská fyziologie*. 2011.

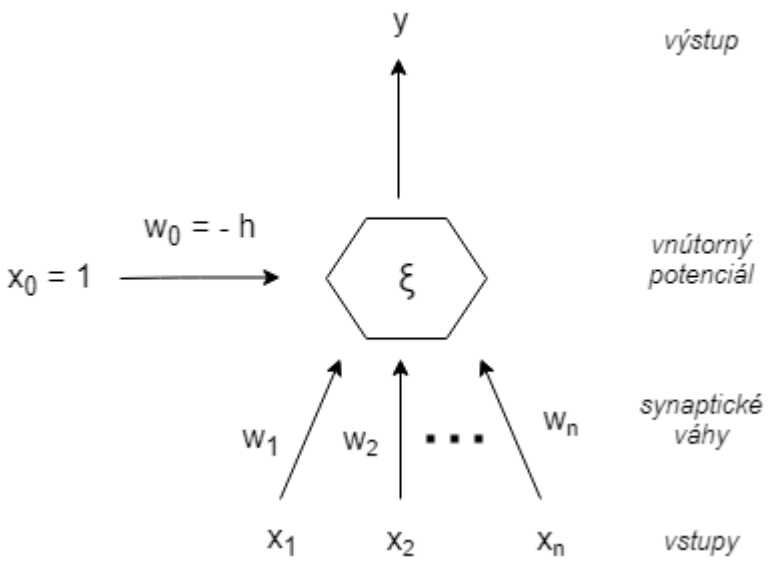
1.1.2 Matematický model neurónu

Neurónová sieť sa skladá z troch vrstiev neurónov: vstupných, skrytých a výstupných. Každý neurón transformuje vstupný signál na výstupný signál podľa svojej aktivačnej (prenosovej) funkcie. Túto funkciu môžeme zapísať ako transformáciu τ vstupného signálu $\mathbf{x} = (x_1, x_2, ..., x_n)$ na výstupný signál $\mathbf{y} = (y_1, y_2, ..., y_m)$:

$$\mathbf{y} = \tau\,(\mathbf{x})$$

Pri formulácii matematického modelu budeme vychádzať z tzv. formálneho neurónu (viz. nasledujúci obrázok).

Obrázok 2: Formálny neurón



Zdroj: vlastné spracovanie podľa - Šíma, Neruda, Teoretické otázky neuronových sítí, 1996., str. 24.

Formálny neurón má *n* vstupov $x_1, x_2, ..., x_n$, ktoré sú ohodnotené odpovedajúcimi synaptickými váhami $w_1, w_2, \ldots, w_n$ Tieto váhy vyjadrujú ich mieru synaptickej priepustnosti. Inhibičný charakter neurónu vyjadríme zápornou hodnotou synaptických váh.

Funkciu celkového vnútorného potenciálu neurónu môžeme zjednodušiť tým, že prah (*h*) nahradíme umelým vstupom x_0, ktorý má vždy hodnotu 1 a synaptickú váhu w_0, rovnú zápornej hodnote prahu. ***Celkový vnútorný potenciál neurónu*** (ξ) potom vyčíslime ako vážený súčet všetkých vstupných hodnôt.

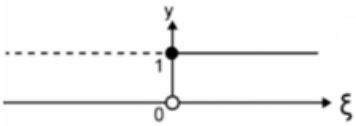
$$\xi = \sum_{i=}^{n} w_i x_i$$

Hodnota vnútorného potenciálu (ξ) je po dosiahnutí prahovej hodnoty (*h*) transformovaná aktivačnou (prenosovou) funkciou τ na výstup neurónu (*y*).

Aktivačná funkcia je obecne nelineárna funkcia, ktorá transformuje hodnotu vnútorného potenciálu neurónu. Nižšie sú uvedené najčastejšie využívané aktivačné funkcie:

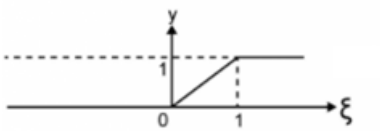
Obrázok 3: Aktivačné funkcie neurónu

- skoková funkcia



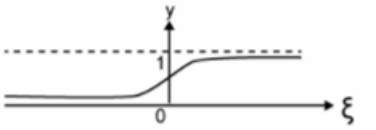
$$\sigma(\xi) = \begin{matrix} 1, \dots\ ak\ \xi \geq 0 \\ 0, \dots\ ak\ \xi < 0 \end{matrix}$$

- po častiach lineárna funkcia



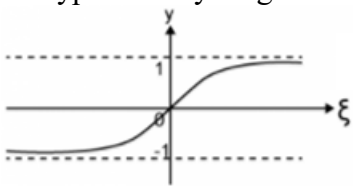
$$\sigma(\xi) = \begin{matrix} 0, \dots\ ak\ \xi > 1 \\ \xi, \dots\ ak\ 0 \leq \xi \leq 1 \\ 1, \dots\ ak\ \xi < 0 \end{matrix}$$

- logistická sigmoida



$$\sigma(\xi) = \frac{1}{1 + e^{\xi}}$$

- hyperbolický tangens



$$\sigma(\xi) = \tan\left(\frac{1}{2}\,\xi\right) = \frac{1 - e^{\xi}}{1 + e^{\xi}}$$

Zdroj: vlastné spracovanie

1.2 Metódy učenia

Učiť sieť riešiť nejaký problém, znamená hľadať hodnoty všetkých váh a prahov jednotlivých neurónov. Priamy výpočet týchto hodnôt často nie je možný, preto v procese učenia prispôsobujeme hodnoty váh tak, aby nám neurónová sieť poskytla čo najlepšie výsledky.

Rozlišujeme teda dva režimy, a to adaptívny (učiaci) a aktívny (vybavovací).

1.2.1 Učenie s učiteľom

Učenie s učiteľom je optimalizačná úloha založená na vyhodnocovaní výstupu. Obvykle disponujeme konečnou množinou M dvojíc x a y_d, získaných, napríklad, meraním a zaznamenaním vstupných a odpovedajúcich výstupných hodnôt . Táto množina dvojíc predstavuje príklad správnej transformácie zo vstupných vektorov na vektory výstupné. Všetky dostupné dáta môžeme rozdeliť na trénovaciu a testovaciu množinu (prípadne validačnú, viz. nasledujúci text). Na základe charakteru riešenej úlohy volíme ich vhodný pomer, ten nie je pevne daný, ale väčšinou približne 70% - 80% dát tvorí trénovaciu množinu. Tieto množiny by mali byť disjunktné.

Trénovaciu množinu s p prvkami definujeme ako:

$$T = \{[x_1, y_{d1}], [x_2, y_{d2}], \dots, [x_p, y_{dp}]\}$$

V adaptačnom režime je trénovacia množina použitá k učeniu siete a overeniu jej funkcie. Algoritmus prechádza iteratívne prvok po prvku a porovnáva jeho výstup y_{dp} (reakciu na predložený vstup) s požadovaným výstupom y_p. Po každej iterácii dochádza k vyhodnoteniu odchýlky a korekcii modelu. Táto odchýlka je pro p-tý vektor trénovacej množiny daná vzťahom:

$$\boldsymbol{e}_p = \boldsymbol{y}_{dp} - \boldsymbol{y}_p$$

Následne kvadratickú odchýlku po predložení p-tého vektoru vyčíslime ako:

$$\varepsilon_p = \sum_{j=1}^{N} \left({}_j\boldsymbol{e}_p \right)^2$$

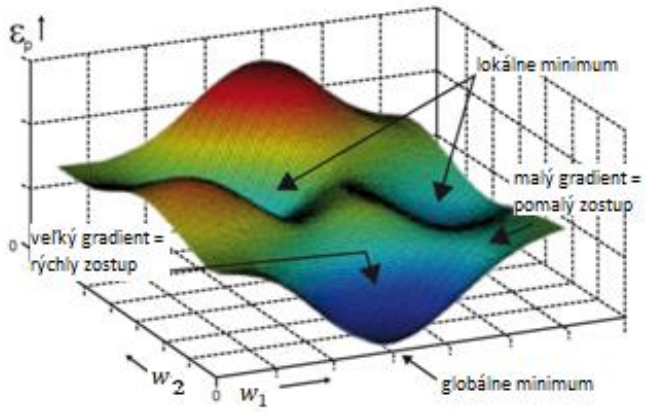
Ideálne by bolo, keby sme boli schopný vyčísliť odchýlku pre všetky možné vstupy siete, tie však nemáme k dispozícií. Môžeme ale definovať chybovú funkciu pre p

vstupov ktoré máme k dispozícii v trénovacej množine ako strednú hodnotu jednotlivých kvadratických odchýliek:

$$\varepsilon = E\{\varepsilon_p\} \text{ pre } \forall \varepsilon_p$$

Adaptačný algoritmus sa snaží o minimalizáciu chybovej funkcie adekvátnou úpravou váh ***w***, teda hľadá minimum na celkovej hyperploche. Každá váha siete predstavuje jednu dimenziu v *n*-dimenziálnom chybovom priestore. Nasledujúci obrázok zobrazuje hyperplochu pre dve váhy. w_2

Obrázok 4: Minimalizácia chybovej funkcie v procese adaptácie



Zdroj: portal.matematickabiologie.cz

Minimalizácia chybovej funkcie môže prebiehať napríklad gradientnou metódou, pričom gradient $\nabla\varepsilon$ vyjadríme ako parciálnu deriváciu chybovej funkcie podľa jednotlivých váh, kde *t* označuje krok učenia.

$$\nabla\varepsilon = \frac{\partial\varepsilon}{\partial w}(t)$$

Upravenú hodnotu váh potom vyčíslime ako:

$$\boldsymbol{w}(t+1) = \boldsymbol{w}(t) - \eta\nabla\varepsilon\,,$$

kde η je parameter, ktorý vyjadruje rýchlosť učenia, respektíve veľkosť kroku, ktorým sa algoritmus približuje ku globálnemu minimu. Môže nadobúdať hodnoty od 0 do 1 (vrátane). Väčšinou sa zo začiatku volí nižšia hodnota a počas procesu adaptácie rastie.

Pri vysokej hodnote parametru η hrozí, že pohyb po ploche bude príl1iš rýchly a preskočíme hľadané minimum (viz. problém preučenia siete). Naopak, ak je η príliš nízke, sieť je náchylnejšia na uviaznutie v lokálnom minime.

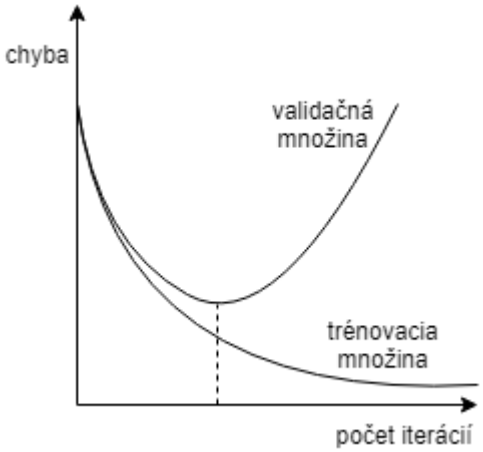
1.2.2 Problém preučenia siete

Po určitom čase sa môže stať, že algoritmus je perfektne adaptovaný na prvky z trénovacej množiny, pretože sieť sa naučila všetky (i tie nepodstatné) detaily tejto množiny, ale model nie je schopný zobecňovať a jeho predikčná schopnosť prudko klesá.

V priebehu učenia je vyhodnocovaná chyba na trénovacej i validačnej množine. Rozdiel medzi validačnou množinou a testovacou množinou je ten, že validačná množina vstupuje do procesu adaptácie ako parameter rozhodujúci o tom, kedy je vhodné učenie ukončiť.

Chybová funkcia na trénovacej množine bude klesať s rastúcim počtom iteračných krokov, avšak na validačnej množine začne od určitého počtu iteračných krokov rásť (predikčná schopnosť zhoršuje) .V tomto bode by bolo vhodné proces učenia zastaviť.

Obrázok 5: Chyba na trénovacej a validačnej množine



Zdroj: vlastné spracovanie

1.2.3 Učenie bez učiteľa

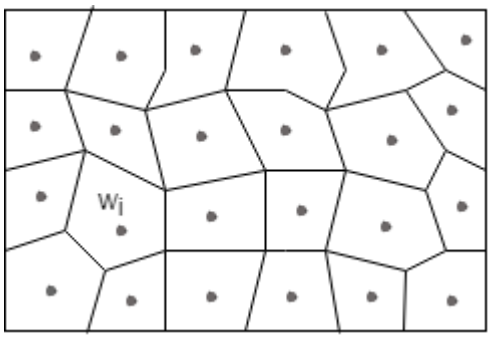
Pri učení bez učiteľa sa nevyhodnocuje správnosť výstupov, pred procesom trénovania nemá požadovaný výstup k dispozícii. Takéto učenie funguje na princípe zhlukovania podľa podobnosti a charakteristických znakov.

Proces učenia prebieha tak, že sieť vyberá charakteristické rysy vstupných vzorov a rozdelí ich do skupín. Počet skupín môže, ale nemusí byť dopredu stanovený. Po určitom čase trénovania sa naučí rozpoznávať vstupy a zaraďovať ich do skupín.

Podobne ako pri učení s učiteľom, aj v tomto prípade môže nastáť pretrénovanie, kedy charakteristické rysy vzorov už nie sú obecné, ale každý vstup sa môže stať samostatným výstupom – skupinou, ktorá sa od ostatných líši v nepodstatných detailoch.

Učenie bez učiteľa niekedy nazývame aj vektorová kvantizácia (VQ). Jej cieľom je aproximovať hustotu pravdepodobnosti reálnych vstupov *x* pomocou konečného počtu takzvaných reprezentantov (značených písmenom w_i) . Pomocou vektorovej kvantizácie rozdelíme priestor na disjunktné oblasti, ktoré tvoria Voroniho mozaiku.[8] Priestor je rozdelený na oblasti určujúce "sféry vplyvu" jednotlivých reprezentantov w_i (označených ako body) podľa minimálnej vzdialenosti.

Obrázok 6: Voroniho mozaika



Zdroj: vlastné spracovanie podľa - Kvasnička a kol., *Úvod do teórie neurónových sietí*, str. 156

Ku každému vstupu *x* priradí takého reprezentanta w_i, ktorý mu je najbližšie, teda minimalizujeme strednú kvadratickú odchýlku medzi vstupom *x* a jeho reprezentantom w_c (minimalizujeme chybu VQ). Hustotu pravdepodobnosti reálnych vstupných

[8] KVASNIČKA, Vladimír, Ľubica BEŇUŠKOVÁ, Jiří POSPÍCHAL, Igor FARKAŠ, Peter TIŇO a Andrej KRÁĽ. *Úvod do teórie neurónových sietí* [online]

vektorov x väčšinou nepoznáme, ale máme zadanú konečnú tréningovú množinu vzorov $T = \{x^{(t)}; t = 1, \dots, k\}$. Chybu VQ potom vypočítame ako:

$$E = \frac{1}{k} \sum_{t=1}^{k} \left\| x^{(t)} - w_c \right\|^2$$

Vzhľadom na to, že vzdialenosť c je závislá na vstupoch x a reprezentantoch w_i, nie je možné explicitne vyjadriť gradient chyby E, tak ako sme k tomu pristupovali doteraz. Preto boli navrhnuté iteratívne algoritmy, ktoré hľadajú približné riešenie, ako napríklad Loydov algoritmus.[7]

2 Členenie neurónových sietí

Neurónové siete môžeme rozdeliť na dve základné skupiny:

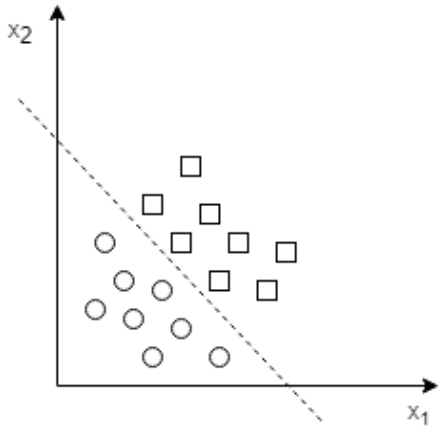
2.1 Dopredné (Feed-forward) siete

V ďalšom texte si bližšie rozoberieme základné typy dopredných sietí. Ich neuróny sú usporiadané do niekoľkých vrstiev tak, že všetky prepojenia v sieti vedú od neurónu v nižšej vrstve smerom k neurónu z vyšších vrstiev, nikdy však nie naopak. Neurón nemôže byť prepojený sám so sebou. Sieť prenáša vstupný signál postupne cez všetky sieťové vrstvy smerom k výstupu. Patria sem:

2.1.1 Perceptrónové siete

Pod týmto názvom rozumieme najjednoduchšie siete, pozostávajúce z výkonných prvkov (neurónov), ktoré majú nastaviteľné váhové koeficienty a nastaviteľný prah. Perceptrón bol navrhnutý v roku 1957 Rosenblattom[9]. Predpokladá lineárne separovateľné triedy v príkladovom priestore, nad ktorými následne dokáže vykonať klasifikáciu (teda rozdelenie do tried). [10]

Obrázok 7: Lineárna separácia tried v príkladovom priestore



Zdroj: vlastné spracovanie

[9] Rosenblatt F.: Principles of Neurodynamics Perceptron and the Theory of Brain Mechanism, Spartan Press, Washington, 1962.

[10] SINČÁK, Peter. *Neurónové siete - Inžiniersky prístup (1. diel): Čo je to perceptrón?* [online].

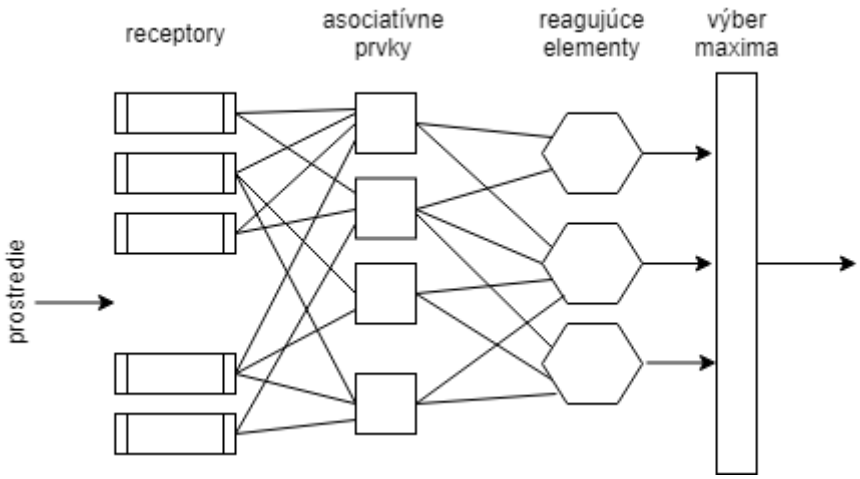
Lineárne separovateľné sú také triedy, kedy je možné oddeliť objekty v príkladovom priestore patriace do jednej triedy od objektov patriacich do inej triedy pomocou nadroviny. Na Obrázku 7 môžeme vidieť lineárnu separáciu v 2-rozmernom priestore priamkou. V 3-rozmernom priestore by boli dané objekty oddelené rovinou.

2.1.2 Topológia perceptrónu

Rosenblattov jednoduchý perceptrón mal tri vrstvy, boli nimi:

- senzorová vrstva
- asociatívna vrstva
- reagujúce elementy

Obrázok 8: Topológia Rosenblattovho perceptrónu



Zdroj: Kotek a kol.: Metody rozpoznávání a jejich aplikace, 1993

Senzorová vrstva (nazývaná aj sietnica), prijíma informácie z prostredia. Tvoria ju *receptory*, teda prvky, ktorých výstup nadobúda hodnoty 1 alebo 0, podľa toho, či sú excitované prostredím alebo nie. Výstupy receptorov sú pomocou náhodne vybraných, no pevne stanovených (teda na procese učenia sa nezúčastňujú) väzieb vedené na asociatívne elementy. Všetky váhy asociatívnych elementov majú pevné hodnoty +1 alebo -1. Takýchto elementov je v perceptróne desiatky tisíc. Keď sa asociatívny element aktivuje, teda súhrn jeho vstupov prekročí zadaný prah, vydá hodnotu 1. Tieto výstupy sú opäť náhodnými väzbami prepojené na reagujúce elementy. Váhy sú však

tento krát premenlivé, teda proces učenia prebieha u perceptrónov na úrovni reagujúcich elementov. Platí, že počet reagujúcich neurónov je rovnaký ako počet tried, do ktorých klasifikujeme.

V poslednom bloku – výber maxima sa vyberie ten reagujúci element, ktorý má najvyšší výstup vypočítaný ako vážený súčet:

$$\sum_{i=1}^{m} w_i x_i$$

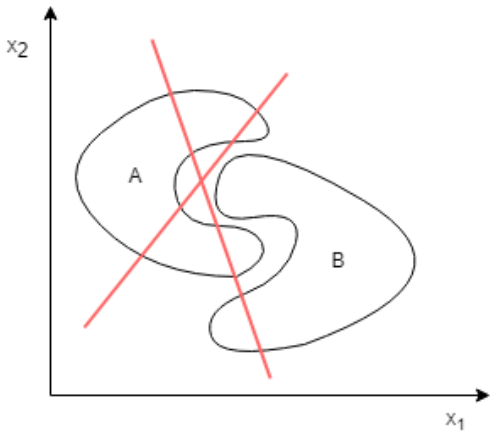
Takýto element najlepšie odpovedá triede, do ktorej vstup patrí.[11]

Model s jedným perceptrónom dokáže klasifikovať vstupy najviac do dvoch tried. Zvýšenie ich počtu v modely, prípadne pridanie ďalších vrstiev nám umožní klasifikáciu do viacerých tried.

2.1.3 Viacvrstvá perceptrónová sieť

Na rozdiel od jednovrstvého obsahuje viacvrstvá perceptrónová sieť minimálne jednu ďalšiu, skrytú vrstvu. Počet neurónov v skrytej vrstve môže byť rôzny, volí sa podľa typu riešenej úlohy. Zaradenie skrytej vrstvy nám umožňuje rozdeliť vstupný priestor na konvexné oblasti, tzn. dokáže aproximovať akúkoľvek spojitú funkciu.

Obrázok 9: Rozdelenie konvexných oblastí pomocou priamok

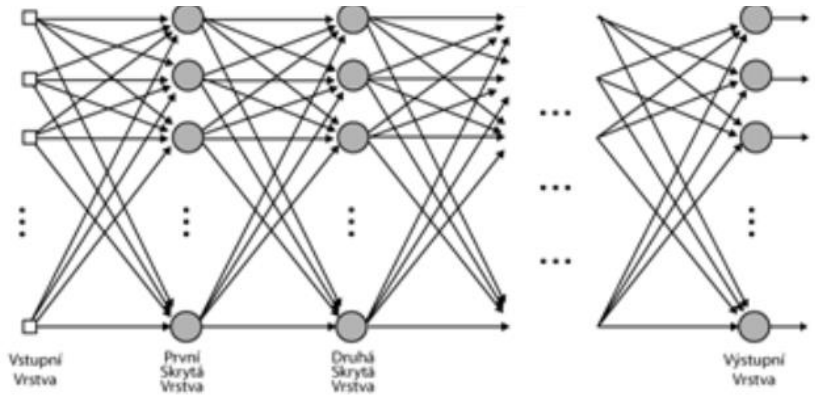


Zdroj: vlastné spracovanie podľa - portal.matematickabiologie.cz

[11] BERKA, Petr. *Dobývání znalostí z databází*. Praha: Academia, 2003.

Architektúru viacvrstvej perceptrónovej siete môžeme vidieť na nasledujúcom obrázku.

Obrázok 10: Organizácia viacvrstvej perceptrónovej siete



Zdroj: portal.matematickabiologie.cz

Neuróny jednej vrstvy nie sú navzájom prepojené, medzi neurónmi susedných vrstiev väčšinou existuje úplné prepojenie. Spoje medzi neurónmi predstavujú dráhu pre šírenie signálu a každý spoj je ohodnotený váhou w_i, tá upravuje intenzitu šíreného signálu.

Hovoríme o vrstvenej priamoväzbovej sieti – väzby medzi neurónmi sú len priame a jednosmerné.

2.1.4 Učenie viacvrstvej perceptrónovej siete

Perceptróny patria medzi modely, ktoré sú založené na učení s učiteľom. Proces učenia spočíva v predkladaní vstupov *x* sieti a na základe odozvy y_p dôjde k vyčísleniu odchýlky ε (viz. kapitola *učenie s učiteľom*).

2.1.5 Algoritmus spätného šírenia chyby (Backpropagation)

Táto metóda učenia s predlohou je založená na výpočte chyby na výstupoch jednotlivých neurónov. Ako prvé sa počítajú chyby pre neuróny na výstupnej vrstve. Táto chyba je využitá k adaptácii váh neurónov v skrytej vrstve a následne je spočítaná chyba na výstupoch týchto neurónov. Úprava váh je postupne šírená štruktúrou siete od výstupnej vrstvy smerom k vstupnej vrstve. Tento proces sa opakuje, pokiaľ nie je funkcia siete dostatočne presná.

Úpravu váh pre ľubovolný *j*-tí neurón *k*-tej vrstvy siete môžeme vyjadriť ako:

$$\boldsymbol{{}^{k}_{j}\mathbf{w}}(t+1) = {}^{k}_{j}\mathbf{w}(t) - \eta\left({}^{k}_{j}e_p\right)\tau'\left({}^{k}_{j}\xi_p\right)\boldsymbol{{}^{k-1}_{j}y_p}$$

kde od pôvodnej hodnoty váh odčítame súčin parametru rýchlosti učenia (η), chyby *j*-tého neurónu *k*-tej vrstvy($^{k}_{j}e_p$), derivácie aktivačnej výstupnej funkcie $\tau\left(^{k}_{j}\xi_p\right)$ a vektoru výstupov na *k-1*-tej vrstve ($\boldsymbol{^{k-1}_{j}y_p}$).

Chybu *j*-tého neurónu výstupnej vrstvy definujeme ako dvojnásobok rozdielu požadovaného výstupu *j*-tého neurónu ($_{j}y_{dp}$) a aktivačnej výstupnej funkcie $\tau\left(^{k}_{j}\xi_p\right)$:

$$^{k}_{j}e_p = -2\left({}_{j}y_{dp} - \tau\left(^{k}_{j}\xi_p\right)\right)$$

A chybu na výstupe neurónov *skrytých* vrstiev ako súčet chýb všetkých neurónov na výstupoch nasledujúcej vrstvy ($^{k+1}_{i}e_p$), ktoré sú násobené deriváciou príslušnej aktivačnej funkcie a hodnotou synaptických váh medzi neurónmi oboch vrstiev ($^{k+1}_{i}w_j$).

$$^{k}_{j}e_p = -2\sum_{i=1}^{M} \tau'\left(^{k+1}_{i}\xi_p\right)\, ^{k+1}_{i}e_p\, ^{k+1}_{i}w_j$$

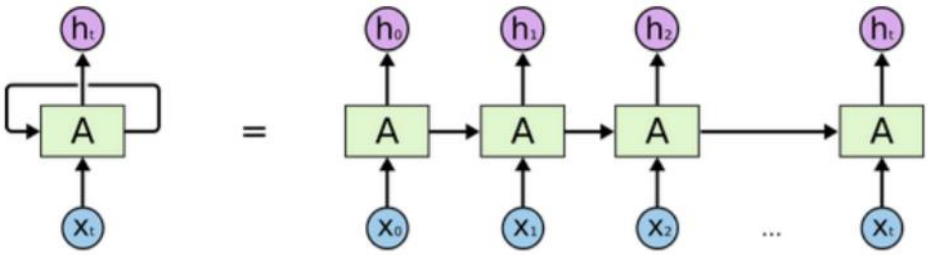
2.2 Rekurentné neurónové siete

Tradičné, teda i vyššie spomínané dopredné siete fungujú na princípe bez konceptu pamäte. Vezmú daný vstupný vektor a pri spracovávaní ďalšieho vektoru si nepamätajú informácie o vektore predchádzajúcom. Obecne prevláda názor, že nie sú príliš vhodné na riešenie úloh, ktoré si vyžadujú zapamätávanie, k akým patrí aj predikcia časových radov. Preto si ďalej predstavíme rekurentné neurónové siete (RNNs).

Svojou cyklickou štruktúrou vhodnejšie reprezentujú vstupné dáta a uchovávajú svoje predošlé stavy, čím umožňujú informácii aby pretrvávala. Uplatnenie nachádzajú hlavne pri práci so sekvenčnými dátami (text, video, hudba ale aj časové rady). Ide predovšetkým o úlohy, v ktorých sa snažíme predikovať nasledujúcu hodnotu rady hodnôt určitej veličiny v čase.

Topológia takejto siete je znázornená na Obrázku 11. Neurón nie je prepojený len s predchádzajúcou a nasledujúcou vrstvou, ale aj sám so sebou – tieto väzby neurónu samého na seba sú reprezentované skrytou maticou synaptických váh. V každom ďalšom kroku učenia nepredkladáme len vstup *x*, ale aj skrytý stav z predchádzajúceho kroku. Rekurentné siete tak môžu využiť staršiu informáciu pri riešení aktuálneho problému.

Obrázok 11: Porovnanie topológie rekurentnej a doprednej siete



Zdroj: Olah, CH. - Understanding LSTM Networks

Problémom, na ktorý môžeme naraziť u rekurentných sietí je *problém dlhodobých (vzdialených) závislostí*. Ide o také prípady, kedy je vzdialenosť medzi príkladom obsahujúcim relevantnú informáciu a predpovedaným príkladom veľká[12]. Napríklad, v úlohách, kde sa model snaží doplniť slovo do vety na základe daného textu, ak je

[12] Tento problém bol bližšie popísaný Hochreiterom (1991) a Bengiom (1994). V prípade predpovede vývoja 10-ročnej časovej rady po uplynutí desiateho roku, informácie z prvých rokov by neboli dostatočne využité.

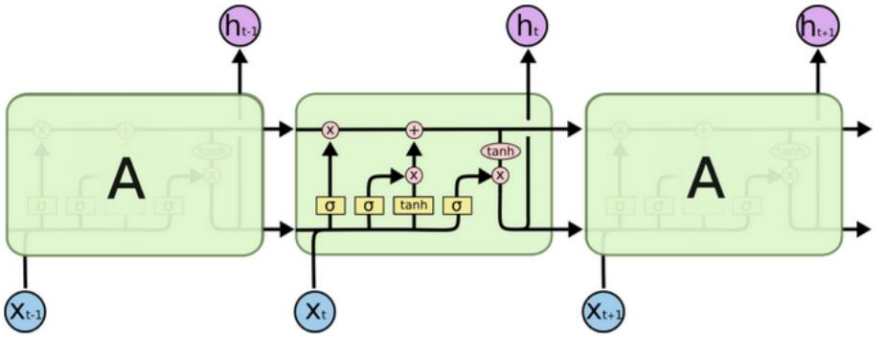
medzera medzi relevantným slovom a miestom, kam má byť slovo doplnené, veľmi veľká, algoritmus nedokáže tieto informácie spojiť.

2.2.1 Long short-term memory (LSTM)

Je špeciálnym druhom rekurentnej siete, ktorý má o niečo zložitejšiu štruktúru. Namiesto jednej skrytej vrstvy sa skladá zo štyroch, navzájom spolupracujúcich vrstiev, nazývaných aj ***brány*** (gates). Prvý krát bola predstavená v roku 1997 Hochreiterom a Schmidhuberom. Následne sa stala veľmi populárnou pre riešenie širokého spektra problémov. Jedná sa o veľmi užitočný nástroj na predpoveď časových radov, pretože ho môžeme použiť k predpovedi ľubovoľne dlhého časového úseku v budúcnosti. Na rozdiel od ostatných rekurentných sietí, LSTM siete dokážu pracovať s dlhodobými (vzdialenými) závislosťami, nemajú problém so zapamätaním si informácie po dlhé časové obdobie.

Štruktúra LSTM sieti je rovnako ako u ostatných rekurentných sietí *reťazovitá,* avšak nejde o prepojenie jednotlivých samostatných vrstiev, ale každá vrstva pozostáva zo štyroch navzájom ovplyvňujúcich sa elementov, medzi ktorými dochádza k nižšie popísaným matematickým operáciám.

Obrázok 12: Štruktúra LSTM siete



Zdroj: Olah, CH. - Understanding LSTM Networks

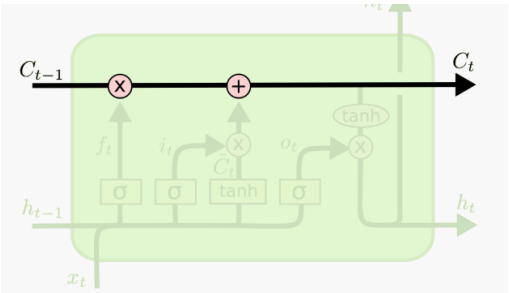
Sú nimi:

- **Memory cell (pamäťová bunka) –** reprezentuje vnútorný stav bunky, ktorý je tvorený krátkodobou a dlhodobou pamäťou.

- **Input Gate i_t** – určuje, aké množstvo informácií z aktuálneho vstupu ovplyvní stav bunky (memory cell). Stanovia sa hodnoty, ktoré budú v stave bunky upravené na základe sigmoidálnej funkcie.
- **Forget Gate -** rozhoduje o tom, aké informácie chceme ponechať a akých sa chceme zbaviť.
- **Output Gate** - ovplyvňuje výstup tým, že určuje aké množstvo informácii z aktuálneho stavu pamäťovej bunky sa dostane do skrytej vrstvy.

Hodnotu pamäťovej bunky (memory cell) ovplyvňujú informácie plynúce cez brány, ktoré majú schopnosť určiť ako, a do akej miery tieto informácie pamäťovú bunku ovplyvnia. Brány sú založené na sigmoidálnych funkciách a operáciách násobenia.

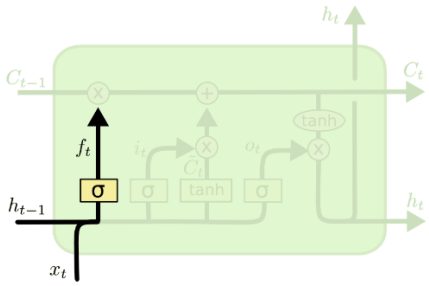
Obrázok 13: Hodnota pamäťovej bunky



Zdroj: vlastné spracovanie podľa Olah, CH. - Understanding LSTM Networks

Prvá brána cez ktorú plynú informácie pri vstupe je ***forget gate.*** Vstupmi sú ako nové informácie x_t, tak výstupná hodnota predchádzajúcej vrstvy h_{t-1}. Na každý vstup je aplikovaná sigmoidálna funkcia σ, ktorá podáva výsledok v intervale 0 až 1 (vrátane). Výstup 0 znamená, že chceme zabudnúť celú vstupnú informáciu, výstup 1 znamená, že ju celú ponecháme. Hodnota medzi 0 a 1 znamená, že informáciu zapíšeme do pamäti len čiastočne.

Obrázok 14: Prechod vstupnej informácie cez forget gate

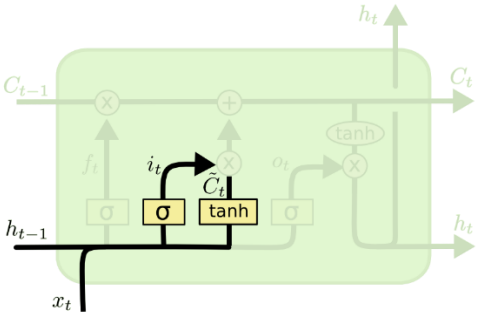


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

Zdroj: Olah, CH. - Understanding LSTM Networks

V ďalšom kroku sa rozhodujeme aké množstvo informácie budeme uchovávať v stave pamäťovej bunky. Tento proces pozostáva z dvoch krokov. Najskôr sigmoidálna funkcia vstupnej brány ***(input gate)*** rozhodne ktoré hodnoty a do akej miery sa na základe novej informácie upravia, následne hyperbolický tangens vstupnej brány vytvorí vektor „kandidátov“ na nové hodnoty $\widetilde{C_t}$, ktoré budú pridané do stavu pamäťovej bunky.

Obrázok 15: Prechod informácie input gate



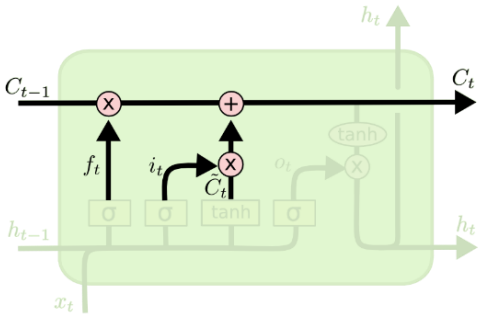
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

Zdroj: Olah, CH. - Understanding LSTM Networks

Na základe predchádzajúcich algoritmov upravíme pôvodný stav pamäťovej bunky C_{t-1} do nového stavu C_t. Stav C_{t-1} vynásobíme výstupom rovnice pre forget gate f_t (vymažeme z pamäti informácie, o ktorých sme rozhodli, že ich má model zabudnúť) a pripočítame vektor navrhovaných nových hodnôt $\widetilde{C_t}$, prenásobený výstupom rovnice

vstupnej brány (hodnoty sú upravené v akom rozsahu, ako sme sa rozhodli, že upravíme pôvodné hodnoty).

Obrázok 16: Úprava hodnoty pamäťovej bunky

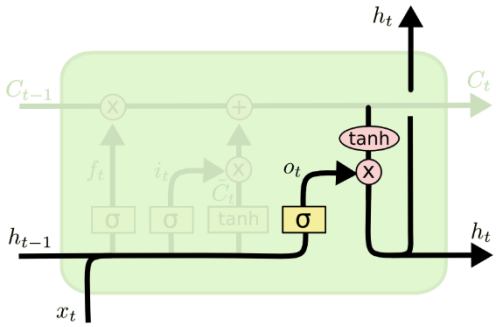


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Zdroj: Olah, CH. - Understanding LSTM Networks

Celkový výstup vrstvy bude vychádzať z hodnoty pamäťovej bunky, tá bude ale upravená ďalšou sigmoidálnou funkciou, ktorá určí, ktorá časť bunky nás zaujíma, teda má byť výstupom. Následne aplikujeme hyperbolický tangens (s hodnotami výstupu od -1 do 1), takže výstupom budú len časti, ktoré sme vybrali.

Obrázok 17: Výpočet výstupu vrstvy LSTM siete



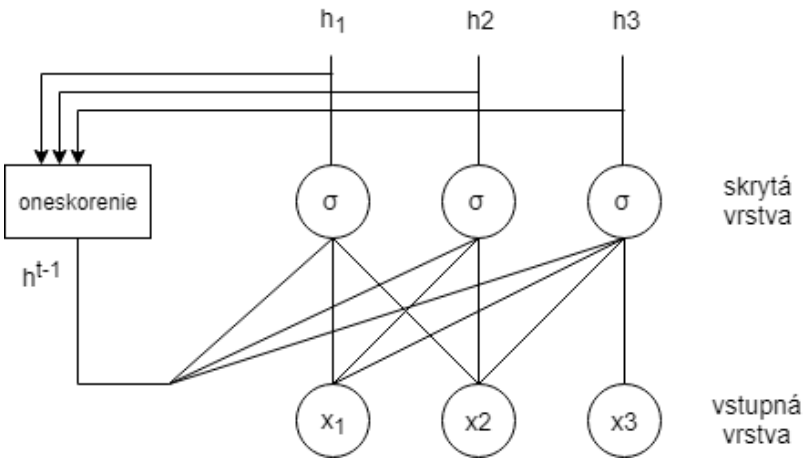
$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Zdroj: Olah, CH. - Understanding LSTM Networks

2.2.2 Architektúra LSTM siete

Modelovanie sekvenčných dát sieťou LSTM je zabezpečené jej špecifickou architektúrou, kde výstup skrytej vrstvy sa so zvoleným časovým oneskorením stáva opäť vstupom pre skrytú vrstvu, tak ako je to znázornené na Obrázku 18.

Obrázok 18: Architektúra LSTM siete



Zdroj: vlastné spracovanie podľa - Andy Thomas, An introduction to neural networks for beginners

Vstupná vrstva – neexistuje obecné pravidlo o tom, koľko neurónov by mala obsahovať vstupná vrstva. Ideálnym prístupom je empiricky testovať výsledky pri rôznom počte vstupných neurónov a následne vybrať variantu, ktorá dosiahla najnižšiu chybu na testovacej množine.

Výstupná vrstva - počet neurónov na výstupnej vrstve je determinovaný tým, aký typ problému riešime. Pokiaľ je naším cieľom priradiť položky dát do jednej z viacerých preddefinovaných tried, počet výstupných neurónov bude zodpovedať počtu tried, do ktorých dáta klasifikujeme. V prípade, že je naším cieľom predikovať hodnotu (či viacero hodnôt) časovej rady, ide teda o problém regresný, výstupná vrstva bude obsahovať len jeden neurón.

Počet skrytých vrstiev a ich neurónov - existuje empiricky vypozorované pravidlo, že pre väčšinu riešených úloh je dostačujúca jedna či dve skryté vrstvy a situácií, kedy pridanie tretej skrytej vrstvy zlepší presnosť modelu, je len málo.

Témou počtu skrytých vrstiev a neurónov v nich sa bližšie zaobetá *Jeff Heaton* vo svojej publikácii *Introduction to Neural Networks for Java*.[13]

Obecne, použitie príliš malého počtu neurónov v skrytej vrstve môže spôsobiť nedostatočné naučenie siete, pretože nedostatok neurónov nedokáže adekvátne posúdiť signály v rozsiahlom datasete. Naopak, príliš veľké množstvo neurónov v skrytej vrstve môže vyústiť do preučenia siete.

Podľa Heatona by sme sa mali pri voľbe počtu neurónov v skrytých vrstvách držať troch základných pravidiel:

1.) optimálny počet neurónov v skrytej vrstve je číslo medzi počtom neurónov vo vstupnej vrstve a počtom neurónov vo výstupnej vrstve
2.) počet neurónov v skrytej vrstve by mal byť práve 2/3 počtu neurónov vo vstupnej vrstve plus počet neurónov vo výstupnej vrstve
3.) počet neurónov v skrytej vrstve by mal byť menší než dvojnásobok počtu neurónov vo vstupnej vrstve

[13] HEATON, Jeff. *Introduction to Neural Networks for Java*. 2nd Edition. Heaton Research, Inc.; 2 edition, 2008.

3 Box – Jenkinsová metodológia

Ešte pred tým, ako sa budeme venovať samotným ARIMA modelom, si definujeme základnú terminológiu používanú v rámci Box-Jenkinsovej metodológie.

3.1 Stochastický proces

Táto metodológia chápe časové rady ako realizáciu stochastického procesu, ako vo svojej publikácii píše Arltová, Arlt (2002, str. 85)[14] - *„Stochastický proces je v čase usporiadaná rada náhodných veličín.“*

Stochastické procesy sú také procesy, ktoré sa riadia zákonmi pravdepodobnostného charakteru, sú teda zaťažené neistotou. Na rozdiel od deterministických procesov, ktoré je možno vyjadriť matematickou funkciou, pri vyjadrení stochastického procesu musíme do modelu zahrnúť náhodnú zložku ε. Časovú radu potom chápeme ako realizáciu stochastického procesu.

3.1.1 Stacionarita

Základnou požiadavkou Box-Jenkinsovej metodológie na časovú radu je jej stacionarita. Rozlišujeme striktne stacionárny a slabo stacionárny proces. Náhodný proces je striktne stacionárny, ak všetky jeho pravdepodobnostné charakteristiky[15] sú invariantné voči posunom v čase. U slabo stacionárneho procesu sú korelačná a kovariančná funkcia závislé na časovej vzdialenosti hodnôt, pre rozptyl a strednú hodnotu platí, že sú v čase nemenné.

Striktne stacionárne rady nie sú veľmi časté, preto pod pojmom stacionarita budeme rozumieť slabú stacionaritu.

[14] ARLT, Josef, ARLTOVÁ, Markéta, RUBLÍKOVÁ, Eva: Analýza ekonomických časových řad s příklady. Praha: Vysoká škola ekonomická, 2002. 147 s. ISBN 80-245-0307-7.

[15] pod pravdepodobnostnými charakteristikami rozumieme strednú hodnotu, rozptyl, korelačnú a kovariančnú funkciu

3.1.2 Autokorelačná funkcia

Autokorelačná funkcia (ACF) popisuje silu lineárnej závislosti (korelácie) medzi dvoma pozorovaniami vrátane náhodných vplyvov medzi týmito pozorovaniami. Vyčíslením jej hodnoty pre pozorovania y_t a y_{t-k}, zahŕňame aj vplyv ich korelácie s veličinami $y_{t-1}, y_{t-2}, \ldots, y_{t-k+1}$.

Autokorelačnú funkciu časovej rady medzi veličinami y_t a y_{t-k} vyjadríme ako:

$$\rho_k = \frac{C(y_t, y_{t-k})}{\sqrt{D(y_t)}\sqrt{D(y_{t-k})}} = \frac{\Upsilon_k}{\Upsilon_0},$$

kde v čitateli dosadíme autokovarianciu[16] medzi pozorovaniami y_t a y_{t-k}, v menovateli dosadíme súčin smerodajných odchýliek týchto pozorovaní. Pre stacionárnu časovú radu platí, že rozptyl postupnosti je konštantný v čase, teda $\sqrt{D(y_t)} = \sqrt{D(y_{t-k})}$.

Pri analýze časovej rady máme však k dispozícii len určité úseky spracovávaných veličín, preto musíme priebeh autokorelačnej funkcie odhadnúť. Jej odhad pre stacionárnu časovú radu pre $k = 1, 2, \ldots, T-1$ dostaneme ako:

$$\widehat{\rho_k} = \frac{\sum_{t=k+1}^{T}(y_t - \bar{y})(y_{t-k} - \bar{y})}{\sum_{t=1}^{T}(y_t - \bar{y})^2}$$

3.1.3 Parciálna autokorelačná funkcia

Korelácia medzi dvoma pozorovaniami je často spôsobená tým, že dané pozorovania sú korelované s treťou veličinou. Na rozdiel od autokorelačnej funkcie, parciálna autokorelačná funkcia (PACF) popisuje koreláciu medzi pozorovaniami yt a y_{t-k} , očistenú o vplyv pozorovaní ležiacich medzi nimi.

Pri odvodení vzťahu pre PACF vychádzame z toho, že parciálna autokorelácia ρ_{kk} predstavuje regresný koeficient v autoregresii k-tého radu, kde e_i je veličina nekorelovaná s veličinami y_{t-j}, pričom $1 \leq j < k$:

$$y_t = \rho_{kk}\, y_{t-1} + \rho_{k2}\, y_{t-2} + \cdots + \rho_{kk}\, y_{t-k} + \varepsilon_t,$$

[16] autokovarianciu medzi dvoma pozorovaniami časovej rady vyjadríme ako strednú hodnotu $E[(x_t - \mu)(x_{t-k} - \mu)]$

Po prenásobení tejto rovnice Y_{t-j}, dostaneme rovnicu pre autokovarianciu v tvare:

$$\gamma_j = \rho_{k1}\,\gamma_{j-1} + \rho_{k1}\,\gamma_{j-2} + \cdots \rho_{kk}\,\gamma_{j-k}\,,\ \ j = 1,2,\ldots,k$$

Po vydelení tejto rovnice veličinou γ_0 dostaneme rovnicu autokorelácie:

$$\rho_j = \rho_{k1}\,\rho_{j-1} + \rho_{k1}\,\rho_{j-2} + \cdots \rho_{kk}\,\rho_{j-k}\,,\ \ j = 1,2,\ldots,k$$

Pre j = 1, 2, ..., k teda platí:

$$\begin{aligned}
\rho_1 &= \rho_{k1}\rho_0 + \rho_{k2}\rho_0 + \cdots + \rho_{kk}\rho_{k-1},\\
\rho_2 &= \rho_{k1}\rho_0 + \rho_{k2}\rho_0 + \cdots + \rho_{kk}\rho_{k-1},\\
\vdots\ \ & \quad\ \vdots \qquad\quad \vdots \qquad\qquad\quad \vdots\\
\rho_k &= \rho_{k1}\rho_{k-1} + \rho_{k2}\rho_{k-2} + \cdots + \rho_{kk}\rho_0
\end{aligned}$$

Výslednú rovnicu parciálnej autokorelácie dostaneme z predchádzajúceho vzťahu využitím Cramerovho pravidla ako podiel determinantov matíc *k*-ého radu P* a P.

$$\rho_{kk} = \frac{|\boldsymbol{P}_k{}^*|}{|\boldsymbol{P}_k|}$$

Pričom matice $\boldsymbol{P}_k{}^*$ a $\boldsymbol{P}_k$ môžeme vyjadriť ako:

$$\mathbf{P}_k = \begin{pmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_{k-1} \\ \rho_1 & 1 & \rho_1 & \cdots & \rho_{k-2} \\ \rho_2 & \rho_1 & 1 & \cdots & \rho_{k-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \cdots & 1 \end{pmatrix} \qquad \mathbf{P}_k^* = \begin{pmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_1 \\ \rho_1 & 1 & \rho_1 & \cdots & \rho_2 \\ \rho_2 & \rho_1 & 1 & \cdots & \rho_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \cdots & \rho_k \end{pmatrix}.$$

Tieto dve matice sa odlišujú len posledným stĺpcom. Dosadením do vzorca pre parciálnu autokoreláciu dostaneme vzťah medzi hodnotami parciálnych autokorelácii ρ_{kk} a hodnotami autokorelácie ρ_k.

$$\rho_{11} = \frac{|\rho_1|}{|1|} = \rho_1,\ \ \rho_{22} = \frac{\begin{vmatrix} 1 & \rho_1 \\ \rho_1 & \rho_2 \end{vmatrix}}{\begin{vmatrix} 1 & \rho_1 \\ \rho_1 & 1 \end{vmatrix}} = \frac{\rho_2 - {\rho_1}^2}{1 - {\rho_1}^2}, atď$$

Odhad parciálnej autokorelačnej funkcie r_{kk} spočítame za predpokladu, že $r_{11} = r_1$ ako:

$$r_{kk} = \frac{r_k - \sum_{j=1}^{k-1} r_{k-1,j} r_{k-j}}{1 - \sum_{j=1}^{k-1} r_{k-1,j} r_j} \quad pre\ všetky\ k > 1,$$

kde $r_{kj} = r_{k-1,j} - r_{kk} r_{k-1,j}\ pre\ j = 1, 2, \dots, k-1.$

3.2 Lineárny stochastický proces

Lineárny proces je definovaný ako nekonečná rada:

$$Y_t = \varepsilon_t + \psi_1 \varepsilon_{t-1} + \psi_2 \varepsilon_{t-2} + \cdots,$$

kde ψ sú ľubovoľné parametre a ε predstavujú tzv. biely šum s nulovou strednou hodnotou, konštantným rozptylom a autokorelačnou funkciou nezávislou na posunoch v čase. Stacionárny lineárny proces je vyjadrený ako lineárna kombinácia rady nekorelovaných rovnako rozdelených náhodných veličín.[17]

V Box – Jenkinsovej metodológii uvažujeme také lineárne procesy, ktoré majú konečný (čo najmenší) počet nenulových parametrov.

Tromi základnými typmi lineárnych procesov sú:

- **Autoregresné procesy** – AR (*p*)
- **Procesy kĺzavých priemerov** - MA (*q*)
- **Zmiešané procesy** – ARMA (*p*, *q*)

Pričom premenné *p* a *q* udávajú počet parametrov – ten sa volí tak, aby výsledné modely spĺňali podmienky stacionarity a invertibility[18].

3.2.1 Autoregresné procesy AR(*p*)

Autoregresné procesy vychádzajú z predpokladu, že každé pozorovanie je závislé na určitom počte predchádzajúcich pozorovaní tejto časovej rady. Autoregresný proces *p*-tého radu môžeme zapísať ako:

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} \dots + \phi_p y_{t-p} + \varepsilon_t,$$

[17] KŘIVÝ, Ivan. *Analýza časových řad*, 2012.

[18] invertibilné procesy, sú také procesy, pri ktorých ak lineárny proces zapíšeme pomocou operátora spozdenia ako π(B)Y_t=ε_t, kde π(B)=1-$\pi_1 B - \pi_2 B^2$, tak platí, že π (B) konverguje pro B≤1

pričom ε_t označuje biely šum s nulovou strednou hodnotou, konštantným rozptylom a nulovou ACF a PACF, ϕ je autoregresný parameter, ktorý tiež môžeme nazvať "pamäť" procesu. Čím viac sa v absolútnej hodnote bližší jednej, tým je pamäť dlhšia a naopak, čím je bližší nule, tým je pamäť kratšia. V prípade, že sa rovná nule, jedná sa o proces bieleho šumu. V týchto modeloch sú všetky autokorelácie nenulové a v náhodných postupnostiach sa navzájom ovplyvňujú pozorovania, ktoré sú od seba ľubovoľne ďaleko v čase. V prípade stacionárnych časových radov s rastúcim rozdielom v čase závislosť medzi pozorovaniami slabne.

Autoregresný proces tiež môžeme zapísať pomocou operátora spätného posunutia B ako:

$$\phi_p(B)y_t = a_t,$$

kde platí $\phi_p(B) = \left(1 - \phi_1 B - \cdots - \phi_p B^p\right)$.

3.2.2 Procesy kĺzavých priemerov MA (q)

Tieto procesy sú založené na myšlienke, že rezíduá časovej rady sa dajú vyjadriť ako lineárna kombinácia minulých rezíduí. Namiesto použitia predchádzajúcich pozorovaní časovej rady pre predikciu využíva minulé hodnoty bieleho šumu ε_t. Proces kĺzavých priemerov q-tého radu môžeme zapísať ako:

$$y_t = c_0 + \varepsilon_t - \Theta_1 y_{t-1} - \Theta_2 y_{t-2} - \cdots - \Theta_q y_{t-q}$$

Zápis pomocou operátora spätného posunutia B vyzerá nasledovne:

$$y_t = \Theta_q(B)a_t,$$

pričom platí $\Theta_p(B) = \left(1 - \Theta_1 B - \cdots - \Theta_p B^p\right)$.

3.2.3 Zmiešaný proces ARMA (p,q)

Vyššie spomínané procesy kĺzavých priemerov a autoregresných postupností môžeme kombinovať do obecnejšieho modelu, tzv. ARMA modelu. Ten potom môžeme zapísať ako:

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t + \Theta_1 y_{t-1} + \Theta_2 y_{t-2} + \cdots + \Theta_q y_{t-q}$$

A opäť za pomoci operátora spätného posunutia B ako:

$$\phi(B) y_t = \Theta(B) \varepsilon_t$$

3.2.4 Integrované procesy ARIMA (*p,d,q*)

Tieto procesy nám umožňujú modelovať i nestacionárne časové rady a to zavedením tzv. *diferenčného operátora* . Pôvodné časové rady však musia byť na stacionárne rady prevoditeľné.

Proces ARIMA môžeme zapísať ako:

$$\phi(B) W_t = \Theta(B) \varepsilon_t,$$

kde W predstavuje diferencovanú verziu pôvodnej rady.

$$W_t = \Delta^d Y_t,$$

pričom d predstavuje rad diferencovania a Δ je diferenčný operátor, teda $(1 - B)^d Y_t$.

Pomocou operátora spätného posunutia zapíšeme integrovaný zmiešaný proces ARIMA ako:

$$\phi(B)(1 - B)^d Y_t = \Theta(\mathrm{B}) \varepsilon_t$$

Parametre procesu ARIMA sú určené nasledovne:

- **p:** počet oneskorených pozorovaní, ktoré budú použité k predpovedi
- **d:** počet diferenciácií pôvodných pozorovaní časovej rady
- **q:** počet oneskorených rezíduí, ktoré budú použité k predpovedi

V prípade, že nejaký z parametrov má hodnotu 0, znamená to, že daný element nie je použitý v modely.

Pri zostavovaní ARIMA modelu postupujeme podľa týchto krokov :

1. Preskúmaním časovej rady zistíme či je normálne rozdelená. V prípade zápornej odpovede použijeme príslušné operácie, akými sú napríklad logaritmická transformácia, či min-max transformácia.

2. Následne si prehliadneme ACF a PACF pôvodnej časovej rady. V prípade, že hodnoty výberovej ACF a PACF v prvom oneskorení sa približujú jednej a hodnoty výberovej ACF postupne klesajú, je vhodné časovú radu stacionarizovať.
3. Po stacionarizácii časovej rady si opäť prehliadneme ACF a PACF za účelom stanovenia hodnôt atribútov p a q. Pre konkrétne hodnoty parametrov sa rozhodujeme podľa podobnosti výberových funkcií s nasledujúcou tabuľkou:

Tabuľka 1: Princípy pre určovanie parametrov p a q

	AR(p)	MA(q)	ARMA(p,q)
ACF	exponenciálny pokles	po oneskorení q je nulová	Od oneskorenia ($q - p$) exponenciálny pokles
PACF	po oneskorení p je nulová	exponenciálny pokles	Od oneskorenia ($q - p$) exponenciálny pokles

Zdroj: vlastné spracovanie podľa

3.2.5 Predpoveď pomocou ARIMA modelu

Proces ARIMA, ktorý označíme ako y_t, diferencovaním prevedieme na stacionárny proces ARMA, označený ako y_t^d. Tento krok môžeme zapísať nasledovne:

$$(1 - B)^d y_t = y_t^d, \qquad pre\ t = 1, \dots, n$$

Po vyjadrení člena y_t z predchádzajúcej rovnice v závislosti na $y^d{}_t$ platí:

$$y_t = \ y_t^d - \sum_{j=1}^{d} \binom{d}{j} (-1)^j\, y_{t-j}\,, \qquad pre\ t = 1, \dots, n$$

Pokiaľ máme k dispozícii pozorovania časovej rady do času t, najlepšiu lineárnu predpoveď $P_t y_{t+h}$ vypočítáme ako[19]:

$$P_t y_{t+h} = P_t y_t^d - \sum_{j=1}^{d} \binom{d}{j} (-1)^j\, P_t y_{t+h-j}, \qquad t = 1, \dots, n \quad h > 0$$

Vzorec pre $P_t y_{t+h}^d$ následne vyjádříme pomocou regresnej komponenty:

[19] Vo väčšine prípadov je hodnota d=1, vzorec sa preto dá upraviť na tvar $P_t y_{t+h} = P_t y_t^d + P_t y_{t+h-j}$

$$P_t y_{t+h}^d = d_{t+h} + \sum_{i=1}^{p} \phi_i P_t y_{t+h-i}^d + \sum_{j=1}^{q} \Theta_j P_t \varepsilon_{t+h-j}^d$$

Podobne budeme pokračovať pre pozorovania v čase t+h-1, … , t+1.

4 Metriky vyhodnotenia modelu

Bodová predpoveď, ktorú budeme zostrojovať predstavuje bodový odhad hodnoty časovej rady v určitom budúcom okamihu. Okrem bodovej predpovede rozlišujeme intervalovú predpoveď (predpovedný interval), tá je analógiou intervalu spoľahlivostí.[20]

Chyba predpovede e_t v čase *t* je definovaná vzťahom:

$$e_t = Y_t - \widehat{Y}_t(t-1),$$

kde Y_t označuje skutočnú hodnotu v čase a $\widehat{Y}_t(t-1)$ predstavuje predpoveď tejto hodnoty zostrojenú v bezprostredne predchádzajúcom časovom okamihu.
Mieru kvality predpovede meriame veličinami, ktorými sú:

- **priemerná štvorcová chyba MSE** (Mean Squared Error)

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

Určuje ako sa v priemere líši odhad hodnoty časovej rady od danej odhadovanej hodnoty, je počítaná pomocou druhej mocniny odchýlky odhadu od správnej hodnoty.

- **priemerná absolútna odchýlka MAE** (Mean Absolute Error)

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

Vyjadruje aritmetický priemer absolútnych odchýliek od odhadovanej hodnoty.
Z princípu výpočtu vidíme, že MSE posudzuje prísnejšie veľké chyby predpovede než tie menšie.

- **Koeficient determinácie**

[20] KŘIVÝ, Ivan. *Analýza časových řad, str.9*

Vyjadruje akú časť variability endogénnej premennej sa nám podarilo vysvetliť pomocou modelu. Najčastejšie sa uvádza v percentách (%) a nadobúda hodnoty od 0 do 1, vrátane týchto hraničných hodnôt:

Celkovú variabilitu endogénnej premennej (celkový súčet štvorcov) vypočítame ako súčet štvorcov odchýliek od priemernej hodnoty:

$$\boldsymbol{S_T = \sum_{i=1}^{n} (Y_i - \overline{Y})^2}$$

Reziduálny súčet štvorcov udáva nevysvetlenú variabilitu endogénnej premennej a vypočítame ho ako súčet štvorcov odchýliek napozorovaných hodnôt od hodnôt odhadnutých pomocou modelu.

$$\boldsymbol{S_e = \sum_{i=1}^{n} (Y_i - \widehat{Y}_i)^2}$$

Z predchádzajúcich dvoch veličín následne vyjadríme samotný koeficient determinácie:

$$\boldsymbol{R^2 = 1 - \frac{S_e}{S_T}}$$

5 Modelovanie

Cieľom praktickej časti práce bude vytvoriť tri modely použiteľné k predikcii pohybu kurzu vybraných akcií, konkrétne rekurentnú neurónovú sieť LSTM, doprednú perceptrónovú sieť využívajúcu učenie *backpropagation* a integrovaný proces ARIMA . K napísaniu kódu bol použitý programovací jazyk Python 3.6.4., spolu s jeho užitočnými knižnicami: *numpy, pandas, matplotlib, scikit-learn, keras, tensorflow* a *statsmodels.*

Tensorflow je open-source knižnica zameraná na popis algoritmov strojového učenia a ich implementáciu. Bola vyvinutá spoločnosťou Google ako rýchlejšia alternatíva vtedy najvyužívanejšej knižnice pre neurónové siete *Theano*.

Keras sa využíva ako nadstavba ľubovoľnej z knižníc pre neurónové siete a používateľovi umožňuje jednoduchší a intuitívnejší návrh siete. Navyše nám umožňuje napísať kód, ktorý je nezávislý na použitej knižnici. V práci bude využitý *Keras* ako nadstavba k *Tensorflow*.

Knižnicu *Statsmodels* použijeme k zostaveniu modelu ARIMA a k základným štatistickým testom.

5.1 Popis dát

Pre analýzu boli použité informácie o cenách desiatich akcií, a to konkrétne ich otváracích, uzatváracích cenách, najvyššej a najnižšej cene v danom dni a zobchodovaného objemu akcií v období od začiatku januára roku 2010 po koniec decembra roku 2016. Spoločnosti, ktorých ceny akcií boli analyzované, boli vybraté tak, aby pokrývali čo najširšie pole odvetví. Všetky akcie vybraných spoločností sú obchodované na *New York Stock Exchange.* Ide konkrétne o štyri technologické firmy – *Google, Yahoo, Amazon* a *Cisco,* ďalej *Cooper Companies*, spoločnosť pôsobiaca v zdravotníckom odvetví, *General Electric* zameriavajúca sa na dopravu (s presahom do energetiky, zdravotníctva, finančníctva a ďalších oborov), *Yum! Brands,* ktorá je vlastníkom množstva známych fastfoodových reťazcov, *UPS* - globálna kuriérska spoločnosť špecializujúca sa na prepravné služby a logistiku a *Bank of America,* medzinárodná banka poskytujúca investičné a ďalšie finančné služby.

S dátami pracujeme vo formáte *dataframe*. Ide o základný dátový typ, ktorý ponúka balíček *pandas*. Jednotlivé záznamy sú tu uvedené v riadkoch a časti záznamov sú zrovnané v stĺpcoch. Majú podobu 2d tabuliek, ktoré majú 5 stĺpcov *open* (otváracia cena) , *low* (najnižšia cena daného dňa), *high* (najvyššia cena daného dňa), *volume* (objem transakcií v danom dni), *adj_close* (uzatváracia cena, presunutá na koniec, aby sa nám s *dataframe*-om lepšie pracovalo) a načítavame ich z datasetu *prices-split-adjusted.csv* dostupného z platformy *Kaggle*[21].

Obrázok 19: Prvých desať riadkov dataframe-u s cenami akcií spoločnosti Amazon

date	open	low	high	volume	adj_close
2010-01-04	136.250000	133.139999	136.610001	7599900.0	133.899994
2010-01-05	133.429993	131.809998	135.479996	8851900.0	134.690002
2010-01-06	134.600006	131.649994	134.729996	7178800.0	132.250000
2010-01-07	132.009995	128.800003	132.320007	11030200.0	130.000000
2010-01-08	130.559998	129.029999	133.679993	9830500.0	133.520004
2010-01-11	132.619995	129.210007	132.800003	8779400.0	130.309998
2010-01-12	128.990005	126.550003	129.820007	9096300.0	127.349998
2010-01-13	127.900002	125.750000	129.710007	10723200.0	129.110001
2010-01-14	129.139999	126.400002	130.380005	9774900.0	127.349998
2010-01-15	129.179993	127.059998	129.649994	15376500.0	127.139999

Zdroj: vlastné spracovanie

5.2 Príprava dát

5.2.1 Normalizácia dát

Pri pohľade na graf vývoja ceny akcií spoločnosti Amazon pozorujeme, že volatilita cien sa v čase menila. Taktiež vidíme, že ceny akcie sa v čase pohybujú v rámci rôznych rozpätí. Hodnoty v druhej polovici pozorovaného časového obdobia sú podstatne vyššie

[21] *https://www.kaggle.com/dgawlik/nyse#prices-split-adjusted.csv*

a taktiež viac fluktuujú. Normalizáciou dát dosiahneme, že pozorované hodnoty sa budú pohybovať v rovnakom, nami určenom rozpätí.

Graf 1: Vývoj cien akcií spoločnosti Amazon pred procesom normalizácie

Zdroj: vlastné spracovanie dát

K normalizácii je možné použiť niekoľko typov transformácií. Často sa používa napríklad lineárna transformácia (hraničnými hodnotami sú najväčšia a najmenšia hodnota pozorovaného súboru), Z-score normalizácia (normalizácia podľa odchýlky od priemerných hodnôt), dekadická normalizácia (desatinné čiarky hodnôt sa posunú tak, aby po úprave patrili do daného intervalu) alebo prípadne nelineárna transformácia logistickou funkciou. V práci použijeme výpočet pomerov z maximálnej hodnoty, ktorú stanovíme ako 1.

Pomocou funkcie MinMaxScaler() sme ceny akcií transformovali do intervalu od 0 do 1.

5.2.2 Transformácia pre učenie s učiteľom

Keďže ide o typ problému, ktorý je riešený pomocou učenia s učiteľom, predkladané vstupy musia mať vyhovujúcu štruktúru. Sieti v trénovacom procese predkladáme na vstupe vektor známych hodnôt časových radov a známych výstupov tak, aby sa mohla naučiť predpovedať výstupy na nových, testovacích dátach.

Obrázok 20: Požadovaná štruktúra dát pre učenie s učiteľom

$$
\begin{array}{ccccc}
y(t_0) & y(t_1) & y(t_2) & y(t_3) & y(t_4) \\
y(t_1) & y(t_2) & y(t_3) & y(t_4) & y(t_5) \\
y(t_2) & y(t_3) & y(t_4) & y(t_5) & y(t_6) \\
 & \cdots & & &
\end{array}
$$

Zdroj: BERKA, Petr. Dobývání znalostí z databází, 2003.

K úprave do správnej štruktúry učenia s učiteľom použijeme funkciu *shift()* balíčku *pandas*, ktorá nám zduplikuje stĺpec s hodnotami časovej rady s tým, že každé pozorovanie posunie o jeden časový okamih dopredu (Žiadna hodnota, teda „*NaN*“ je pridaná na začiatok rady).

V prípade, že vo funkcii *shift* použijeme argument so záporným číslom, teda *shift(-1),* duplikovaná rada sa skráti o prvú hodnotu v poradí a hodnota *NaN* bude pridaná na koniec rady. Keď tento proces zopakujeme viac krát, dostaneme dostatočne dlhé vstupné sekvencie (X), pričom hodnoty v poslednom stĺpci použijeme ako výstup pre trénovanie. Na Obrázku 21 môžeme pozorovať použitie funkcie *shift()* s parametrom -4 pre akcie spoločnosti Cisco. Následne je nutné zmazať riadky obsahujúce NaN hodnoty.

Obrázok 21: Štruktúra trénovacích dát pre predpoveď cien akcií spoločnosti Cisco

	Vstupy X			Výstup y
	var1(t-3)	var1(t-2)	var1(t-1)	var1(t)
0	NaN	NaN	NaN	24.690001
1	NaN	NaN	24.690001	24.580000
2	NaN	24.690001	24.580000	24.420000
3	24.690001	24.580000	24.420000	24.530001
4	24.580000	24.420000	24.530001	24.660000
5	24.420000	24.530001	24.660000	24.590000
6	24.530001	24.660000	24.590000	24.200001
7	24.660000	24.590000	24.200001	24.639999
8	24.590000	24.200001	24.639999	24.950001
9	24.200001	24.639999	24.950001	24.400000

Zdroj: vlastné spracovanie

Vzhľadom na to, že ku konštrukcii predpovede využívame štyri rôzne premenné, vyššie popísané kroky zopakujeme na pre každú z nich

5.2.3 Rozdelenie dát na trénovaciu a testovaciu množinu

Časové rady sme rozdelili v pomere 80:20 na trénovaciu a testovaciu množinu. Prvá množina, obsahujúca data o cenách akcií od januára roku 2010 do júla 2015 bude použitá k „naučeniu“ modelu tým, že bude mať k dispozícii jak vstupy, tak k ním odpovedajúce výstupy. Druhú množinu, popisujúcu ceny akcií od augusta roku 2015 do decembra roku 2016 použijeme k tomu, aby sieť na základe vstupov predikovala výstupy. Tieto predikované výstupy následne porovnáme so skutočnými výstupmi testovacej množiny a vyhodnotíme predikčnú schopnosť modelu.

5.3 Konšrukcia LSTM modelu

LSTM siete sú špecifické tým, že do nich vstupujú dáta v maticovom formáte s tromi dimenziami, ktorými sú:

- **počet vzorkov (samples)** – ide o celkový počet sekvencií, ktoré sieťou prejdú v jednom okamihu. V prípade, že by sme sieťou prehnali celú časovú radu naraz, počet vzorkov by sa rovnal jednej. Takýto prístup sa však z praktických dôvodov nepoužíva a pozorovania v čase sa rozdelia do tzv. ***dielov (batches),*** viz. nasledujúca podkapitola. Preto hodnota tejto dimenzie bude korešpondovať so zvoleným parametrom ***batch size***.
- **počet časových krokov (time steps**)– označuje počet príkladov v jednom vzorku, teda v našom prípade počet príkladov v jednom ***diely.***
- **počet atribútov (features)** – je daný celkovým počtom rôznych pozorovaní, teda obvykle počtom stĺpcov.

Transformáciu do trojrozmernej dimenzie zabezpečíme pomocou funkcie $reshape()$.

Obrázok 22: Príklad maticovej štruktúry po dátovej úprave pre akcie spoločnosti Cisco

```
[[0.56555561 0.5804878  0.60123384 0.09365799 0.60418966]
 [0.59277778 0.60054195 0.59506446 0.06696856 0.59812566]
 [0.5894445  0.59837398 0.59562532 0.04991974 0.58930537]
 [0.57611106 0.58915989 0.58609083 0.04233739 0.59536937]
 [0.5805555  0.59349593 0.59338196 0.05608093 0.6025358 ]
 [0.59499994 0.60216802 0.60403805 0.05016834 0.59867692]
 [0.58388894 0.58428184 0.58216486 0.05003697 0.57717753]
 [0.57500006 0.59078596 0.59338196 0.06435818 0.60143321]
 [0.59166667 0.60813008 0.6146943  0.07997443 0.61852262]
 [0.615      0.59620591 0.61581601 0.07878938 0.58820283]
 [0.5894445  0.60704607 0.6034772  0.05334769 0.61300989]
 [0.59388894 0.58644992 0.59506446 0.06153417 0.5887541 ]
 [0.58777778 0.57235778 0.59057764 0.08444175 0.56504958]
 [0.55055556 0.51869924 0.55019627 0.112812   0.50937147]
 [0.50944444 0.51165312 0.50813233 0.07813832 0.51047406]
 [0.49277772 0.50623301 0.50925409 0.08542984 0.50551265]
 [0.49777772 0.51273713 0.51261915 0.07906987 0.51929435]
 [0.51388889 0.49051491 0.51037569 0.10048256 0.48456447]
 [0.50111117 0.49539301 0.50532801 0.1182618  0.48180808]
 [0.48444444 0.49214087 0.49467193 0.08375065 0.4961411 ]
 [0.49333328 0.49918699 0.50420642 0.08324837 0.51212787]
 [0.51277778 0.51761518 0.5120583  0.13022758 0.51488421]] 0.5198456163260409
```

Zdroj: vlastné spracovanie dát

5.3.1 Výber vhodných parametrov LSTM siete

Pred skonštruovaním samotného modelu potrebujeme zistiť ako nastaviť nižšie popísané parametre tak, aby sieť dosahovala čo najlepších výsledkov pri predpovedi.

Optimálne hodnoty zistíme empirickým testovaním, pričom presnosť modelu budeme určovať pomocou priemernej štvorcovej chyby (MSE) na trénovanej množine.

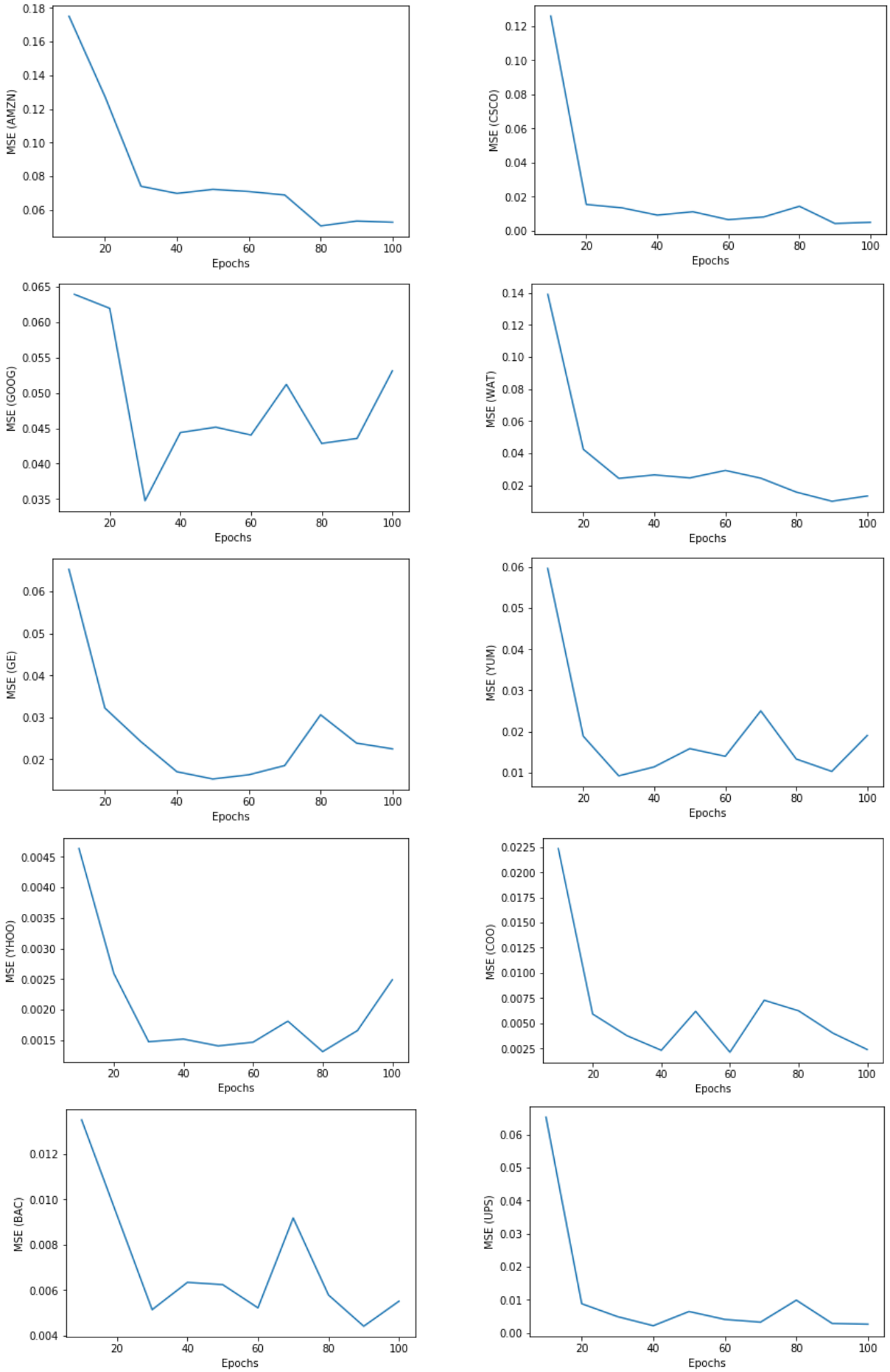
- **batch_size -** vzhľadom na to, že v tréningovom procese sieťou nemôže prejsť naraz celý dataset, sme nútený rozdeliť ho do viacerých častí (dielov), tzv. *„batches*“. Počet príkladov datasetu, ktoré predložíme sieti pred tým, ako dôjde k úprave váh sa označuje ako parameter ***batch size***. O optimálnej hodnote tohto parametru rozhodneme v podkapitole venujúcej sa architektúre siete, pretože jeho hodnota rozhoduje o počte neurónov vo vstupnej vrstve.
- **počet epoch -** udáva koľko krát prejdú dáta modelom počas procesu trénovania. Jedna epocha znamená jeden prechod všetkých dát sieťou, avšak väčšinou je jeden prechod nedostatočný a dáta musia sieťou prejsť viac krát. Optimálny počet epoch zistíme testovaním. S rastúcim počtom prechodov rastie počet úprav váh a po určitom čase môže dôjsť k preučeniu. Na grafoch uvedených na nasledujúcej strane môžeme vidieť, že naozaj, vo väčšine prípadov, po prekročení počtu epoch s najnižšou MSE, sa chyba začne zvyšovať

Tabuľka 2: Výsledky testovania optimálnych hodnôt parametru počet epoch

AMZN	80
CSCO	90
GOOG	30
WAT	90
GE	50
YUM	90
YHOO	80
COO	60
BAC	90
UPS	40

Zdroj: vlastné spracovanie

Obrázok 23: Úspešnosť LSTM modelu pri rôznom počte epoch



Zdroj: vlastné spracovanie

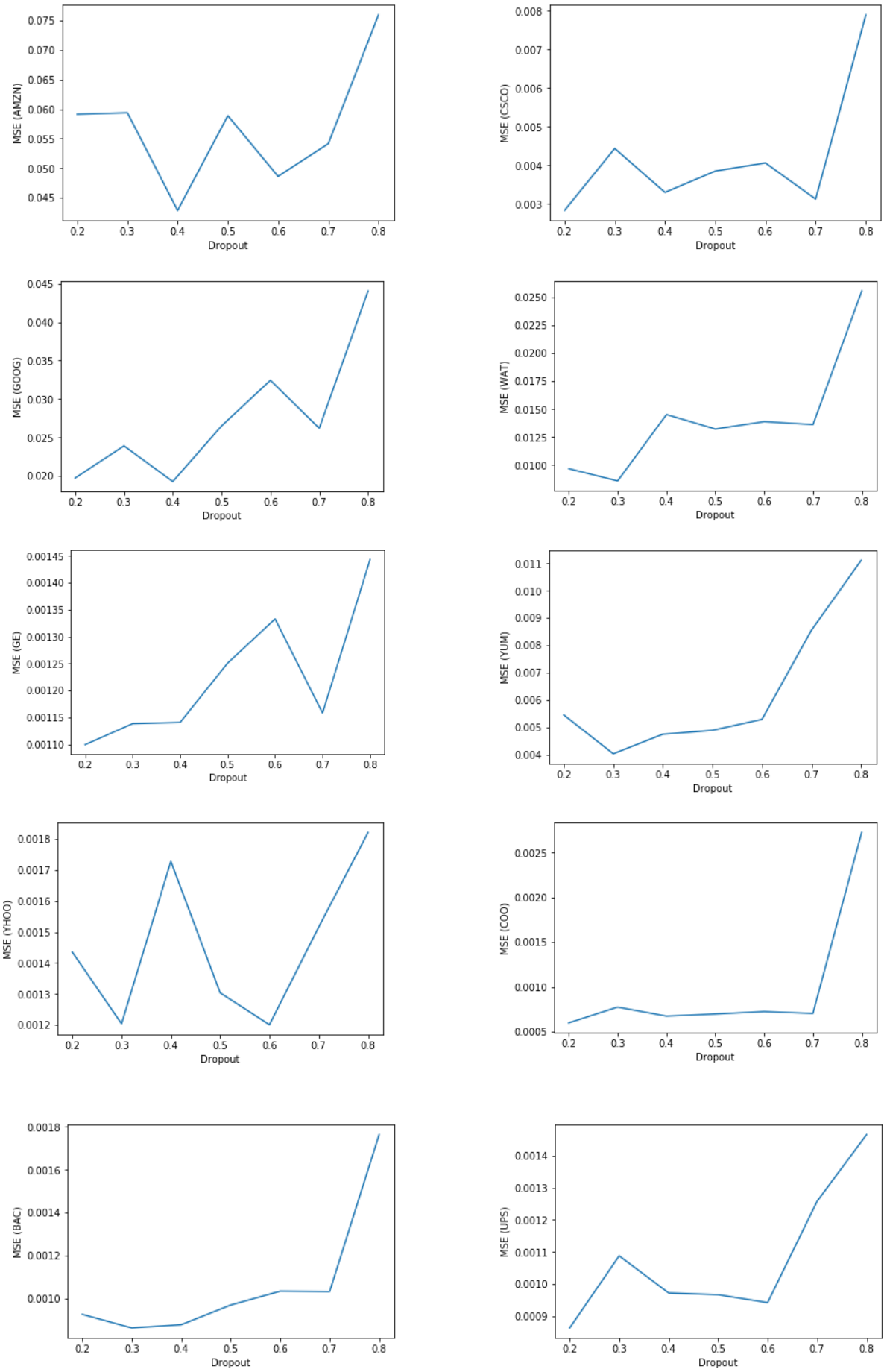
- **dropout –** je ďalším voliteľným parametrom pri zostavovaní siete. Je to metóda, podstatou ktorej je, že niektoré vstupné a spätné väzby vynecháme z procesu aktivácie a úpravy váh. Dochádza k vynechaniu určitého počtu náhodných neurónov v skrytej vrstve. Hodnota tohto parametru sa nachádza medzi 0 (dropout nebude použitý) a 1 (žiadne spojenie). Ide vlastne o udanie pravdepodobnosti, s ktorou budú dané väzby vynechané pri vstupe do ďalšieho LSTM bloku.

Tabuľka 3: Výsledky testovania optimálnych hodnôt parametru dropout pre LSTM sieť

AMZN	0.5
CSCO	0.2
GOOG	0.4
WAT	0.3
GE	0.4
YUM	0.3
YHOO	0,6
COO	0.6
BAC	0.3
UPS	0.6

Zdroj: vlastné spracovanie

Obrázok 24: Úspešnosť LSTM modelu pri rôznych hodnotách parametru dropout



Zdroj: vlastné spracovanie

- **window_size** – predpoveď je konštruovaná tak, že pevne daný počet predchádzajúcich hodnôt je použitý k predikcii nasledujúcej hodnoty. Parameter *„window_size“* udáva dĺžku sekvencie, ktorá sa použije k predikcii. Tieto sekvencie pevnej dĺžky sa postupné posúvajú o jednu hodnotu v čase, teda pripojí sa nasledujúca hodnota (hodnota predtým predpovedaná) a vynechá sa prvá hodnota v predchádzajúcej sekvencii.

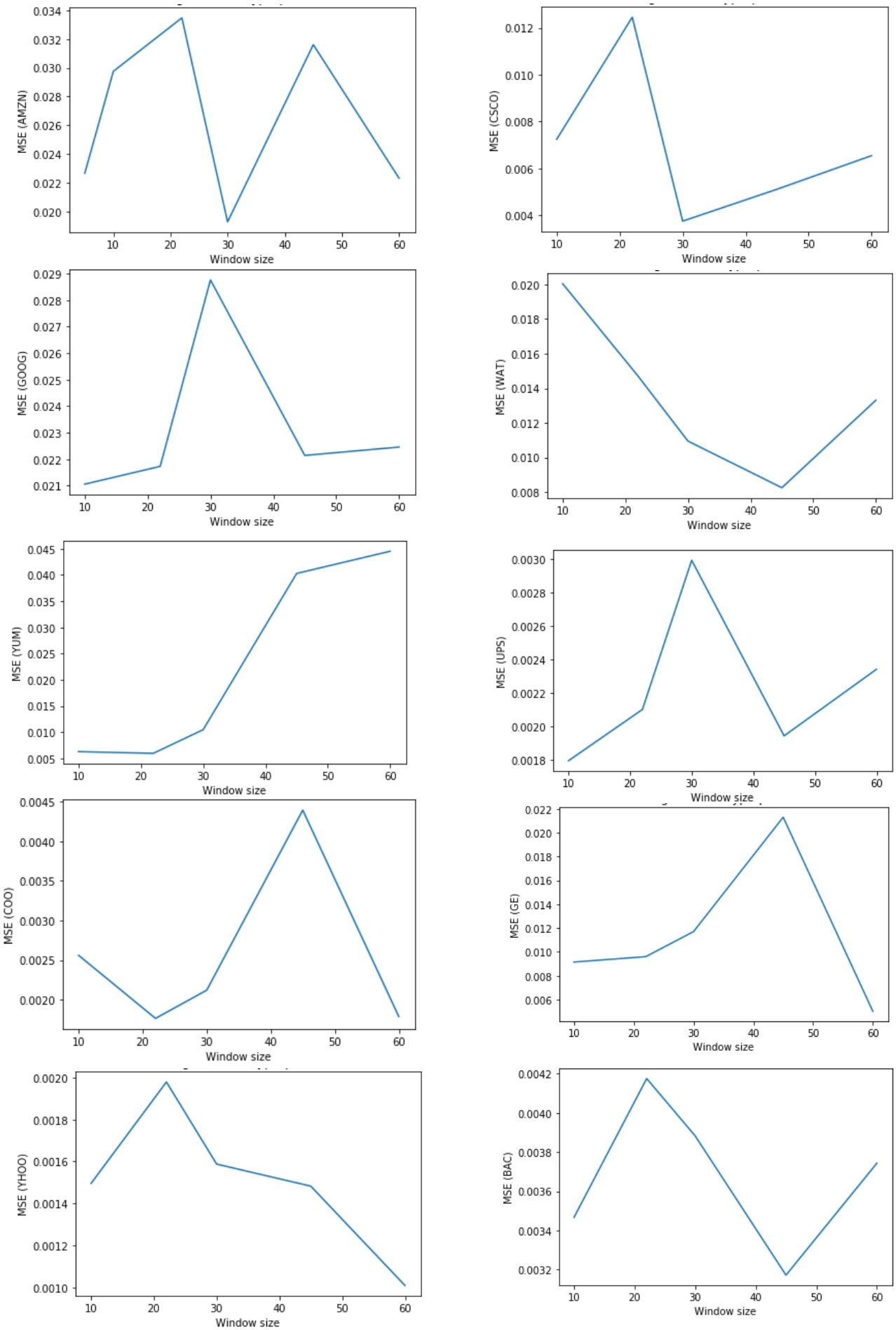
Tabuľka 4: Výsledky testovania optimálnych hodnôt parametru window size pre LSTM sieť

AMZN	30
CSCO	30
GOOG	10
WAT	45
GE	22
YUM	22
YHOO	60
COO	60
BAC	45
UPS	10

Zdroj: vlastné spracovanie

Nejedná sa síce o obecne platné pravidlo, no môžeme vypozorovať, že ideálna hodnota window_size sa u niektorých časových radov rovná 30 či 22, čo odpovedá jednému mesiacu pre 30 dní, prípadne mesiacu s 22 obchodnými dňami.

Obrázok 25: Úspešnosť LSTM modelu pri rôznych hodnotách parametru parametru



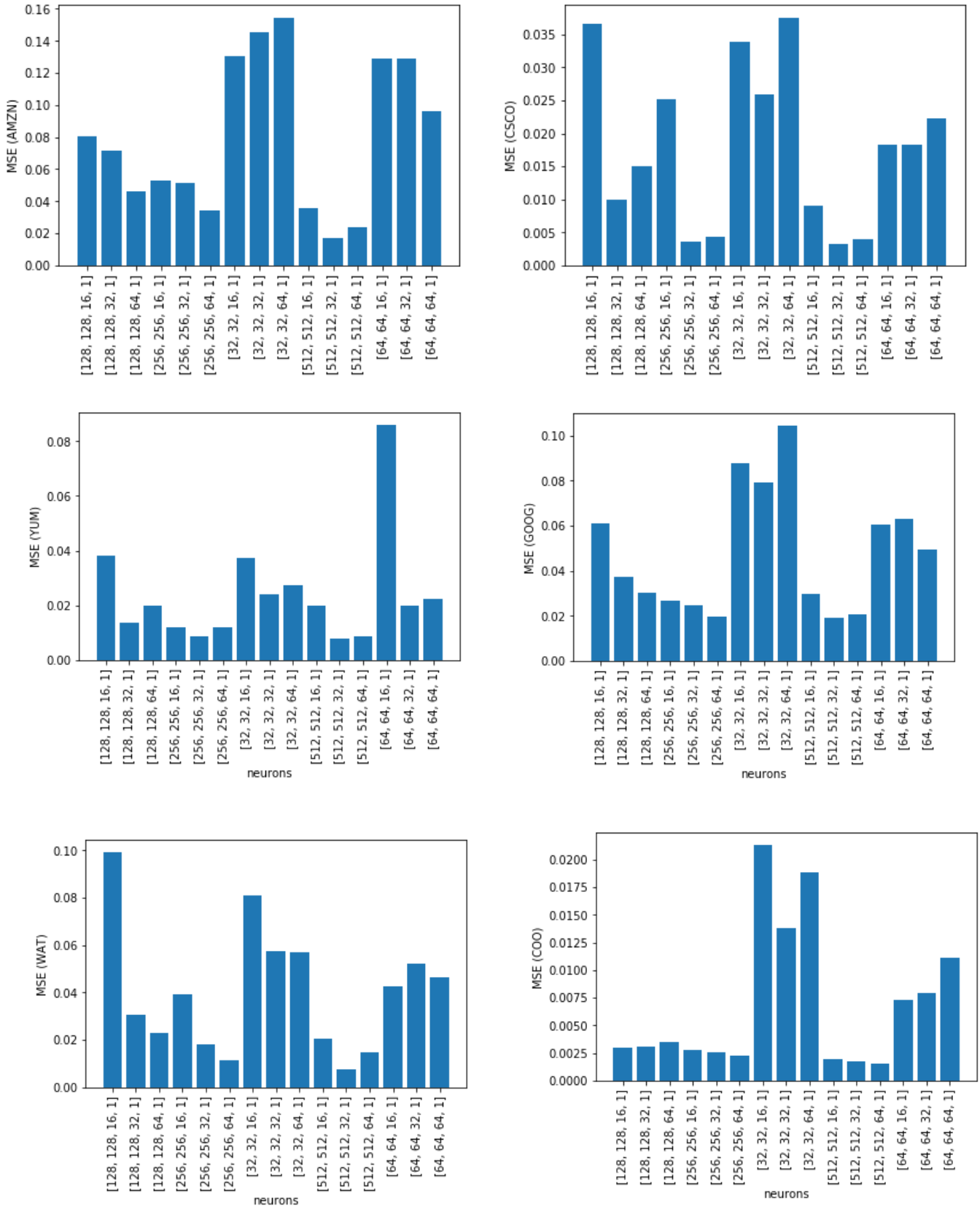
Zdroj: vlastné spracovanie

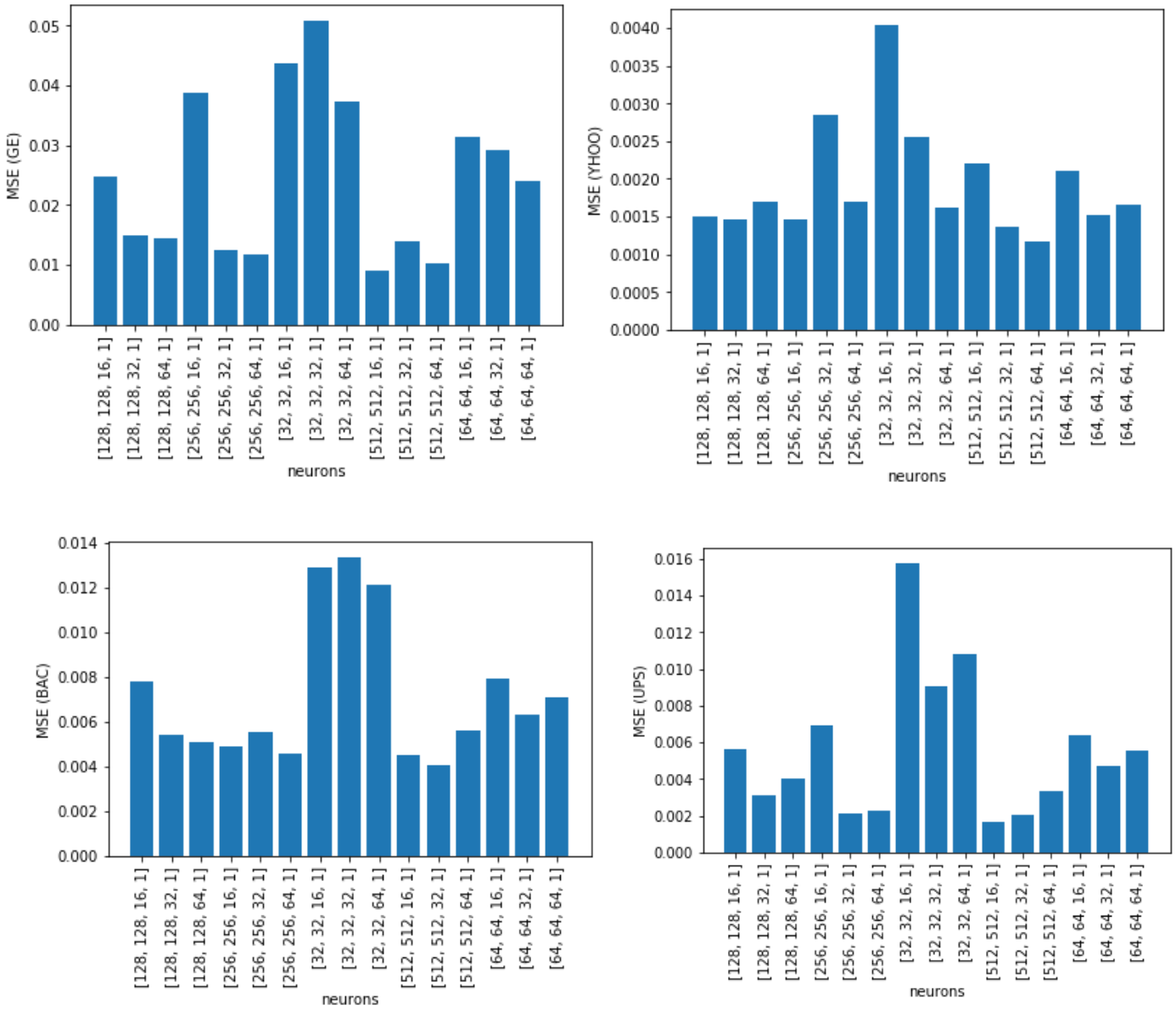
5.3.2 Architektúra LSTM siete

Aby sme zistili optimálny počet neurónov v jednotlivých vrstvách, či samotný počet skrytých vrstiev, opäť spočítame priemernú štvorcovú chybu (MSE) pre rôzne kombinácie počtu neurónov a vrstiev. Pre predikovanie uzatváracích cien jednotlivých akcii použijeme vždy tú štruktúru siete, kde vyšla MSE najnižšia. Na nasledujúcich grafoch môžeme vidieť, aké hodnoty MSE vyšli pre rôzne kombinácie počtu neurónov a skrytých vrstiev pre dané akcie.

V predchádzajúcej kapitole sme uviedli, že hodnoty parametru ***batch size*** zistíme pri testovaní optimálnej štruktúry, hodnota tohto parametru nám totižto udáva počet neurónov vo vstupnej vrstve. Pre predikciu cien akcií väčšiny spoločností, menovite ***Amazon, Cisco, Google, Waters, Yum, Bank of America*** použijeme štruktúru `[512, 512, 32, 1]`. Batch size bude teda 512, sieť bude mať 2 skryté vrstvy, prvá z nich bude mať 512, druhá 32 neurónov. Výstupná vrstva bude mať vo všetkých prípadoch len jeden neurón. V prípade spoločností ***Yahoo,*** a ***The Cooper Companies*** použijeme štruktúru `[512, 512, 64, 1]`, takže v druhej skrytej vrstve bude mať 64 neurónov na rozdiel od predchádzajúcich 32. Pre spoločnosti ***General Electric*** a ***UPS*** použijeme štruktúru `[512, 512, 16, 1]`, teda len 512 neurónov vo vstupnej vrstve, 512 neurónov v prvej skrytej, 16 neurónov v druhej skrytej vrstve a 1 neurón vo výstupnej vrstve.

Obrázok 26: Úspešnosť LSTM modelu pri rôznom počte





Zdroj: vlastné spracovanie

5.4 Konštrukcia Backpropagation modelu

Dáta upravíme na vstupy (*X*) a výstupy (*y*) rozdelením na dva *dataframe*-y. Prvý, *dataframe X* obsahuje prvé 4 stĺpce, teda atribúty - otváracia cena, najnižšia cena daného dňa, najvyššia cena daného dňa a objem transakcií v danom dni. *Dataframe y* pozostáva len z jedného atribútu – uzatváracej ceny, hodnoty, ktorú budeme predikovať.

Následne, podobne ako u LSTM modelu, vytvoríme trénovaciu a testovaciu množinu v pomere 80:20. Na rozdiel od LSTM ale nepoužívame parameter *window size* a dáta neupravujeme do maticového formátu s tromi dimenziami.

5.4.1 Výber vhodných parametrov Backpropagation siete

Výber optimálnych parametrov sme opäť testovali na trénovacích dátach a použité budú tie, ktoré dosahujú najnižšiu MSE na trénovacej množine.

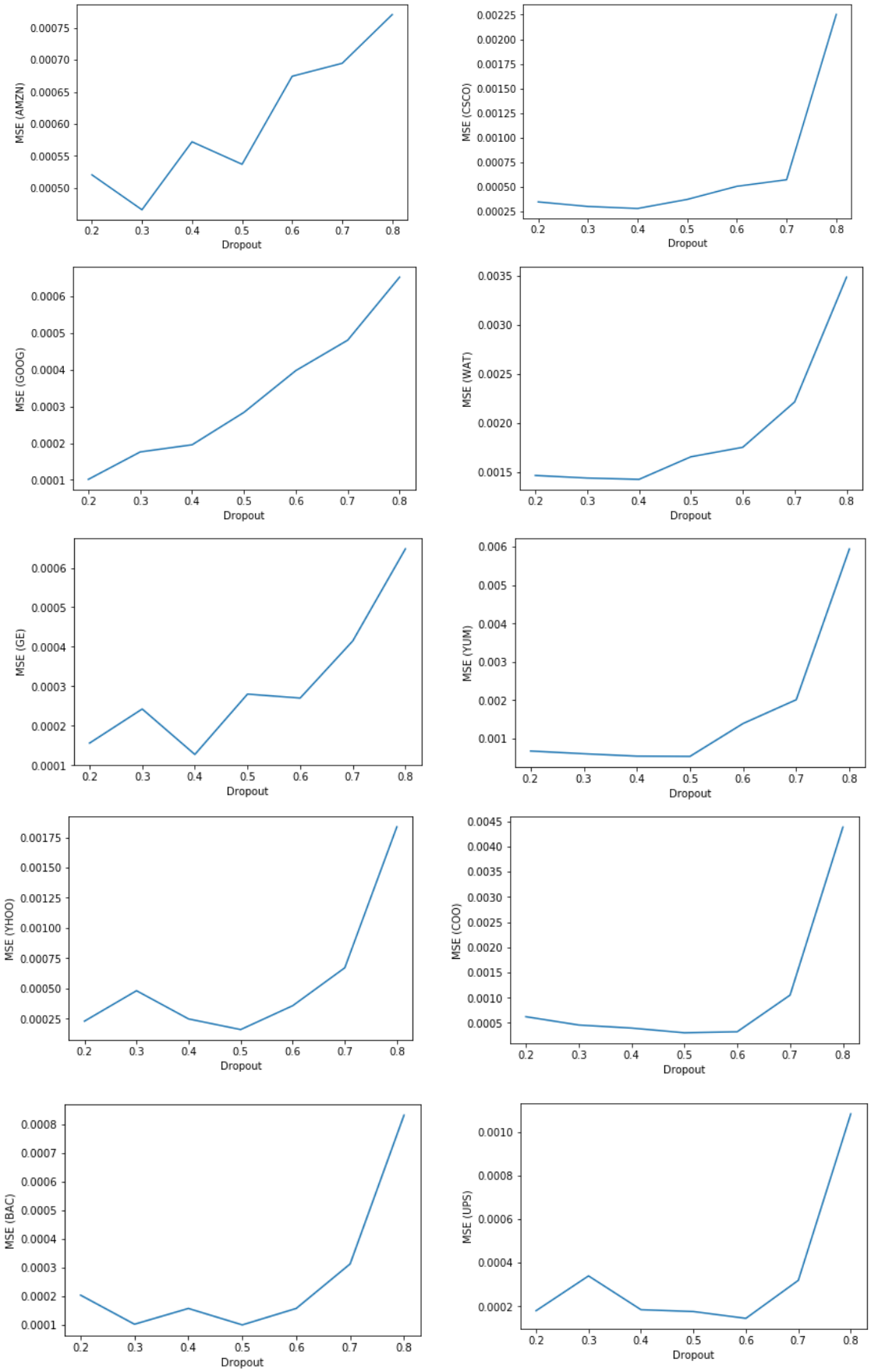
Tabuľka 5: Výsledky testovania optimálnych hodnôt parametrov dropout a počet epoch pre BKP

	dropout	*počet epoch*
AMZN	0.2	100
CSCO	0.4	100
GOOG	0.2	80
WAT	0.4	90
GE	0.4	100
YUM	0.5	100
YHOO	0,5	60
COO	0.5	90
BAC	0.5	90
UPS	0.6	80

Zdroj: vlastné spracovanie

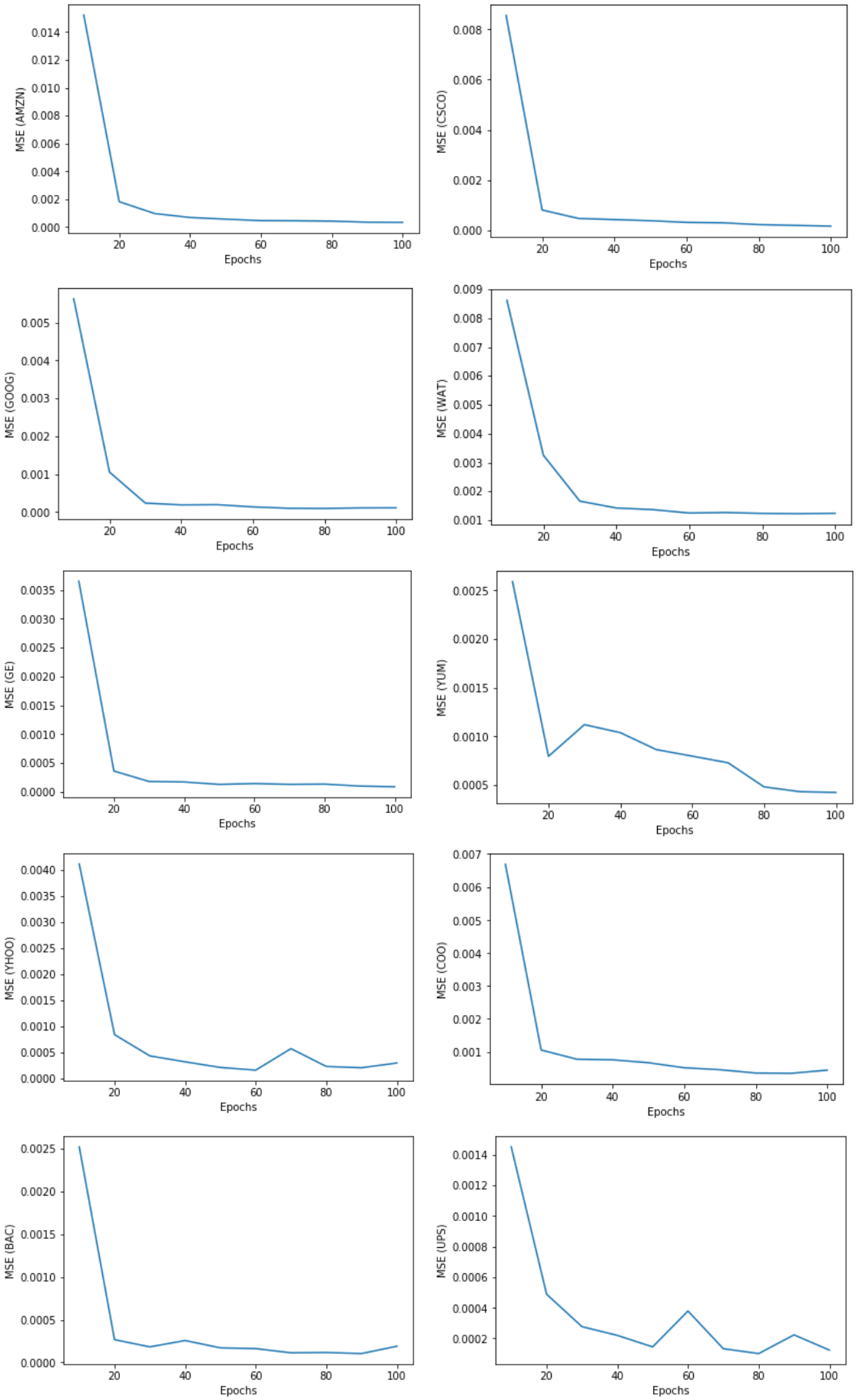
Výsledky výberu parametrov si môžeme prezrieť na grafoch na nasledujúcich dvoch stranách.

Obrázok 27: Úspešnosť BKP modelu pri rôznych hodnotách parametru dropout



Zdroj: vlastné spracovanie

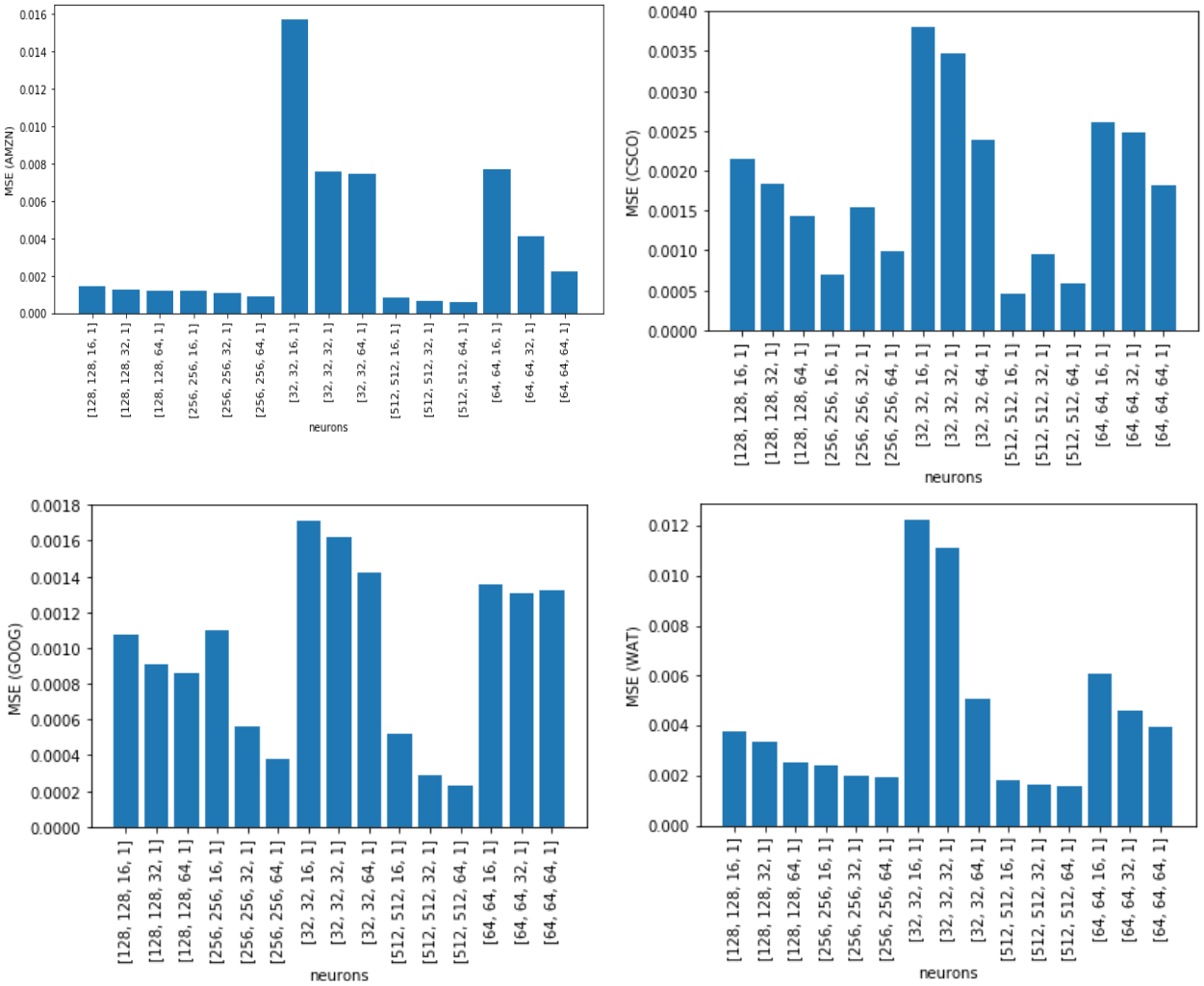
Obrázok 28: Úspešnosť BKP modelu pri rôznom počte epoch

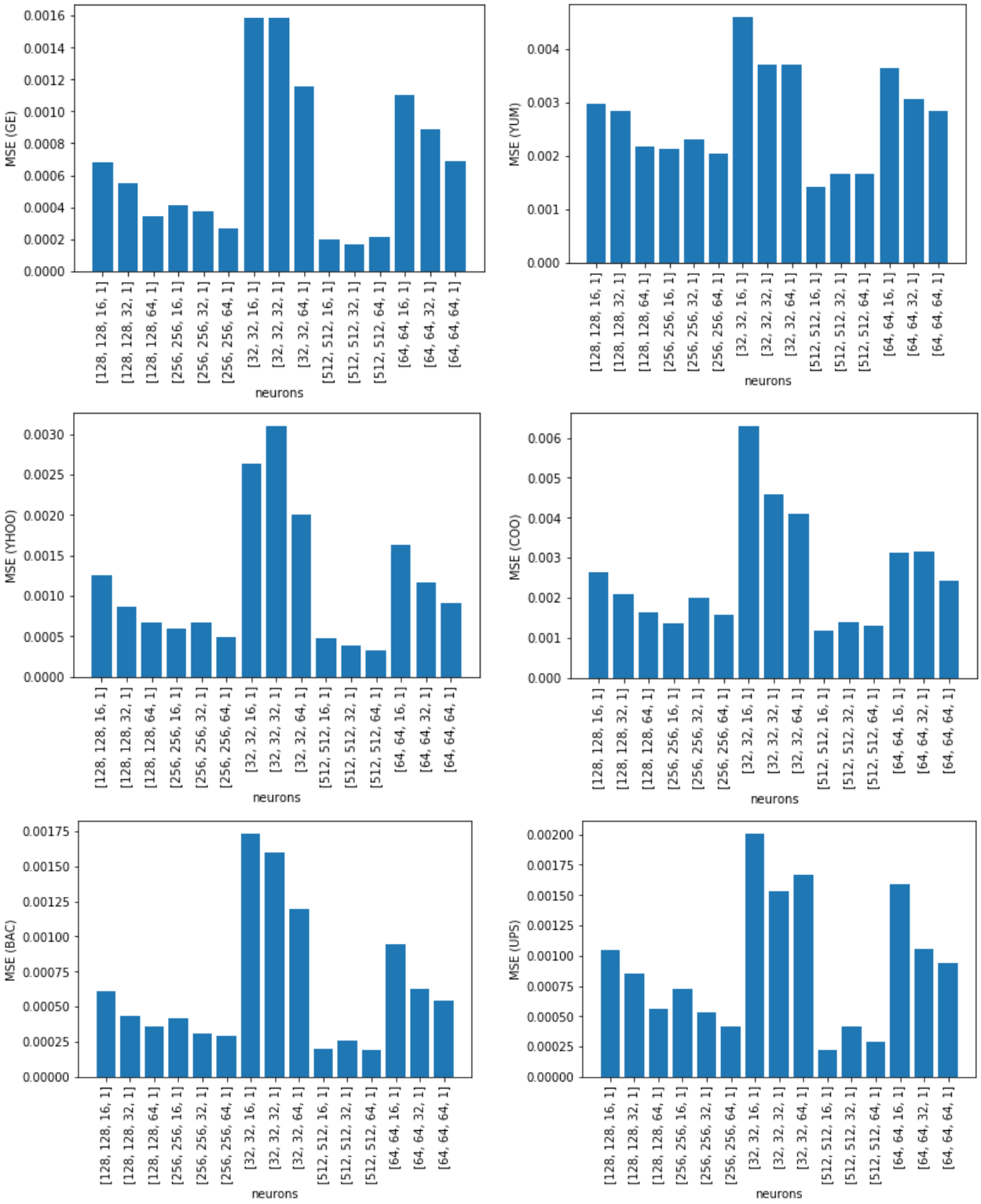


Zdroj: vlastné spracovanie

Pre prevažnú väčšinu modelov – ***Amazon***, ***Yahoo***, ***Cisco***, ***Waters*** a ***Bank of America,*** bude použitá architektúra `[512, 512, 64, 1]`, teda 512 neurónov vo vstupnej vrstve, 512 neurónov v prvej skrytej vrstve, 64 v druhej skrytej vrstve a 1 neurón vo výstupnej vrstve. Pre predpoveď cien akcií spoločností ***Google***, ***The Cooper Companies, UPS*** a ***Yum!*** použijeme architektúru `[512, 512, 16, 1]`, kde v druhej skrytej vrstve použijeme namiesto 64 neurónov len 16 neurónov. Pre spoločnosť ***General Electric*** použijeme v druhej skrytej vrstve 32 neurónov – štruktúra siete bude `[512, 512, 32, 1]`.

Obrázok 29: Úspešnosť BKP modelu pri rôznom počte neurónov





Zdroj: vlastné spracovanie

5.5 Arima model

Pre pohodlnejšiu analýzu ARIMA modelu prevedieme hodnoty stĺpca „*adj_closed*“ na objekt *time series*.

5.5.1 Test stacionarity časovej rady

Keďže Arima modely vyžadujú stacionaritu časovej rady, dáta testujeme na prítomnosť trendu. To je možné spraviť viacerými spôsobmi.

1. **Zobrazenie štatistík v grafe** – prvý spôsob, ktorý je síce rýchlejší, ale nepresnejší je vizuálny test. Na grafe si zobrazíme kĺzavý priemer a kĺzavý rozptyl a pozorujeme ako sa mení v čase.

Graf 2: Vývoj priemeru a rozptylu v čase pre pôvodnú časovú radu

Zdroj: vlastné spracovanie

Napriek tomu, že rozptyl a teda aj smerodajná odchýlka sú pomerne malé, môžeme vidieť, že v čase postupne narastajú. A keďže, ako sme uviedli v kapitole venovanej Box-Jenkinsovej metodológii, stacionárna rada má konštantnú strednú hodnotu, konštantný rozptyl a autokoreláciu, ktorá nie je závislá v čase. Preto môžeme predpokladať, že daná časová rada nebude stacionárna.

2. **Dickey-Fullerov Test** - druhým, presnejším spôsobom je výpočet testovej štatistiky Dickey-Fullerovho testu, ktorého nulovou hypotézou H_0 je, že časová rada je nestacionárna. Test má určené kritické hodnoty pre rôzne úrovne pravdepodobnosti.

Obrázok 30: Výstup Dickey-Fullerovho testu pre pôvodnú časovú radu

```
Results of Dickey-Fuller Test:
Test Statistic                   0.495121
p-value                          0.984724
#Lags Used                       4.000000
Number of Observations Used   1757.000000
Critical Value (10%)             -2.567646
Critical Value (1%)              -3.434077
Critical Value (5%)              -2.863186
dtype: float64
```

Zdroj: vlastné spracovanie

V prípade, že vypočítaná testová štatistika je menšia než kritická hodnota, zamietame nulovú hypotézu H_0 a môžeme časovú radu prehlásiť za stacionárnu. V našom prípade, je väčšia ako kritická hodnota pre 90%-ný interval spoľahlivosti. Časová rada nie je stacionárna.

5.5.2 Odstránenie vplyvu trendu

Všetky použité časové rady sú zo svojej podstaty nestacionárne – vykazujú trend, je vhodné ich previesť do stacionárnej podoby. To zabezpečíme transformáciou na logaritmické prírastky (*logaritmic returns*) podľa vzorca:

$$r_t = \ln(1 + R_t) = \ln\left(\frac{P_t}{P_{t-1}}\right) = \ln(P_t) - \ln(P_{t-1})$$

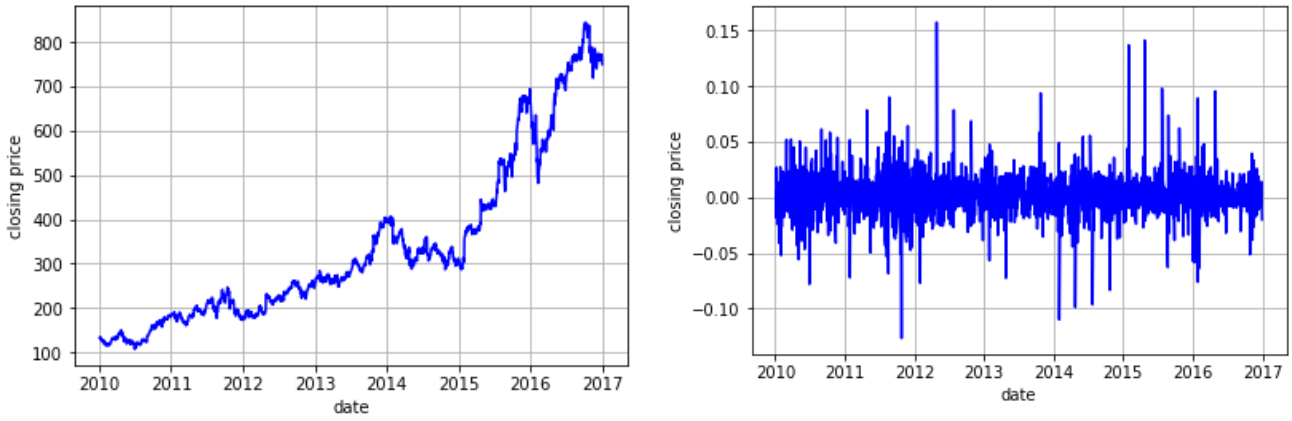
Oproti použitiu klasických relatívnych prírastkov prinášajú logaritmické prírastky výhodu normálneho rozdelenia výslednej časovej rady.

Ďalšou výhodou je časová aditivita. Logaritmické prírastky je možné jednoducho sčítať.

$$r_t(k) = \ln\big(1 + R_t(k)\big) = \ln((1 + R_t)(1 + R_{t-1}) \dots (1 + R_{t-k+1}))$$

Na obrázku môžeme vidieť pôvodnú – nestacionárnu časovú radu a časovú radu po prevode na logaritmické prírastky, teda takú, ktorá je už stacionárna.

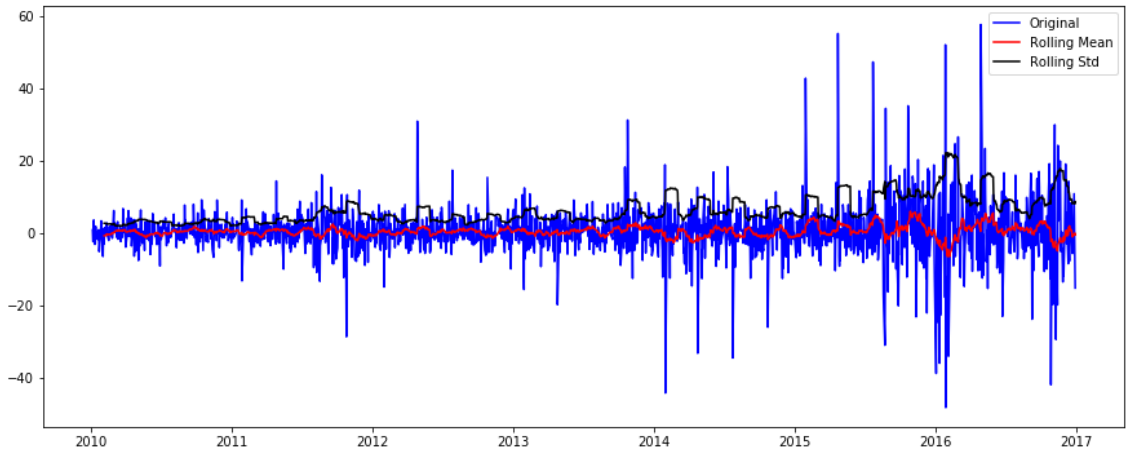
Obrázok 31: Pôvodná a diferencovaná časová rada



Zdroj: vlastné spracovanie

Stacionaritu ešte skontrolujeme vizuálne, pozorujeme vývoj kĺzavého priemeru a kĺzavého rozptylu v čase. Vidíme, že aj napriek tomu, že kĺzavý rozptyl rastie, tieto výchylky dosahujú oveľa nižších hodnôt ako pred stacionarizáciou.

Graf 3: Vývoj priemeru a rozptylu v čase pre upravenú časovú radu



Zdroj: vlastné spracovanie

Z výsledkov Dickey-Fullerovho testu vidíme, že testová štatistika má oveľa nižšiu hodnotu ako kritická hodnota pre 99%-ný interval spoľahlivosti. Môžeme preto zamietnuť nulovou hypotézou H_0 o nestacionarite časovej rady. Časová rada je stacionárna.

Obrázok 32: Výstup Dickey-Fullerovho testu pre novú časovú radu

```
Results of Dickey-Fuller Test:
Test Statistic                  -22.588013
p-value                           0.000000
#Lags Used                        3.000000
Number of Observations Used    1757.000000
Critical Value (10%)             -2.567646
Critical Value (1%)              -3.434077
Critical Value (5%)              -2.863186
dtype: float64
```

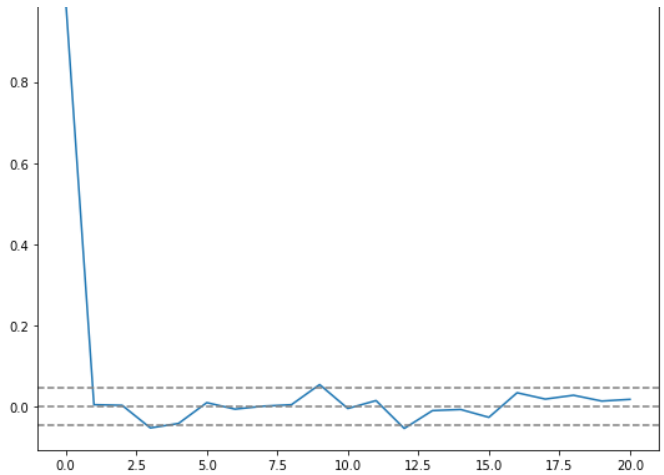
Zdroj: vlastné spracovanie

5.5.3 Určenie počtu parametrov

Výsledná predikcia bude závisieť na zvolenom počte parametrov p, d, q. Pre zvolenie ich správnej hodnoty si premietneme autokorelačnú funkciu a parciálnu autokorelačnú funkciu.

1. **Autokorelačná funkcia (ACF) -** zvolené *q* bude odpovedať hodnote, pre ktorú autokorelačná funkcia prvý krát pretne hornú hranicu intervalu spoľahlivosti. V našom prípade to bude hodnota 1.

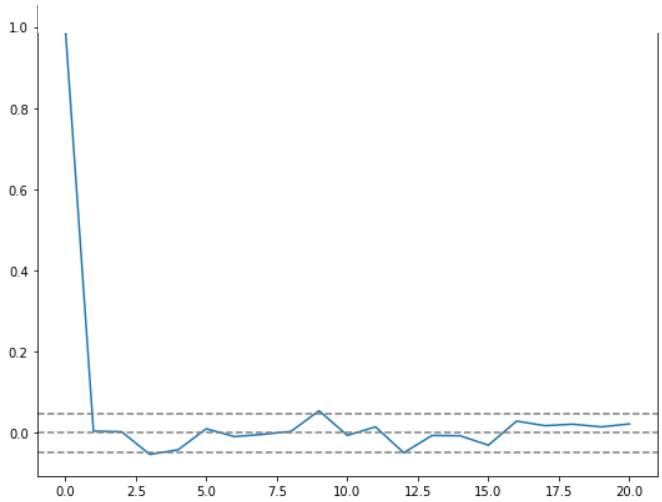
   Graf 4: ACF pre ceny akcií spoločnosti Amazon



   Zdroj: vlastné spracovanie

2. **Parciálna autokorelačná funkcia (PACF)** –zvolené p bude odpovedať hodnote, pre ktorú parciálna autokorelačná funkcia prvý krát pretne hornú hranicu intervalu spoľahlivosti. V tomto prípade pôjde opäť o hodnotu 1.

Graf 5: ACF pre ceny akcií spoločnosti Amazon



Zdroj: vlastné spracovanie

5.6 Obchodná stratégia

Pre celkové porovnanie využiteľnosti výsledkov modelov budeme simulovať obchodovanie na trhu počas obdobia, ktoré odpovedá našej testovacej množine.

Uvažujeme investora, ktorý špekuluje na uzatváraciu cenu nasledujúceho dňa s počiatočným kapitálom v hodnote 1 000 EUR. Pre rozhodovanie o nákupe, či predaji využíva výsledky predpovedí z našich troch modelov. Rozhodnutie v čase t_0 bude závisieť na predpovedanej hodnote pre obdobie t_1. V prípade, že predpovedaný denný logaritmický výnos akcie X je kladný (model predpovedá rast uzatváracej ceny v nasledujúcom dni), investor sa rozhodne pre nákup danej akcie v hodnote 1000 EUR . V prípade, že denný logaritmický výnos akcie X predpovedaný modelom je záporný (teda predpovedá pokles uzatváracej ceny v nasledujúcom dni) investor predá akcie v hodnote 1000 EUR.

Tabuľka 6: Rozhodovacia stratégia investora

Očakávam rast	$P_t y_{t+1} > 0$	Nákup akcií X v hodnote 1000 EUR
Očakávam pokles	$P_t y_{t+1} < 0$	Predaj akcií X v hodnote 1000 EUR
Očakávam konštantný vývoj	$P_t y_{t+1} = 0$	Držím pozíciu (nenakupujem/nepredávam akcie)

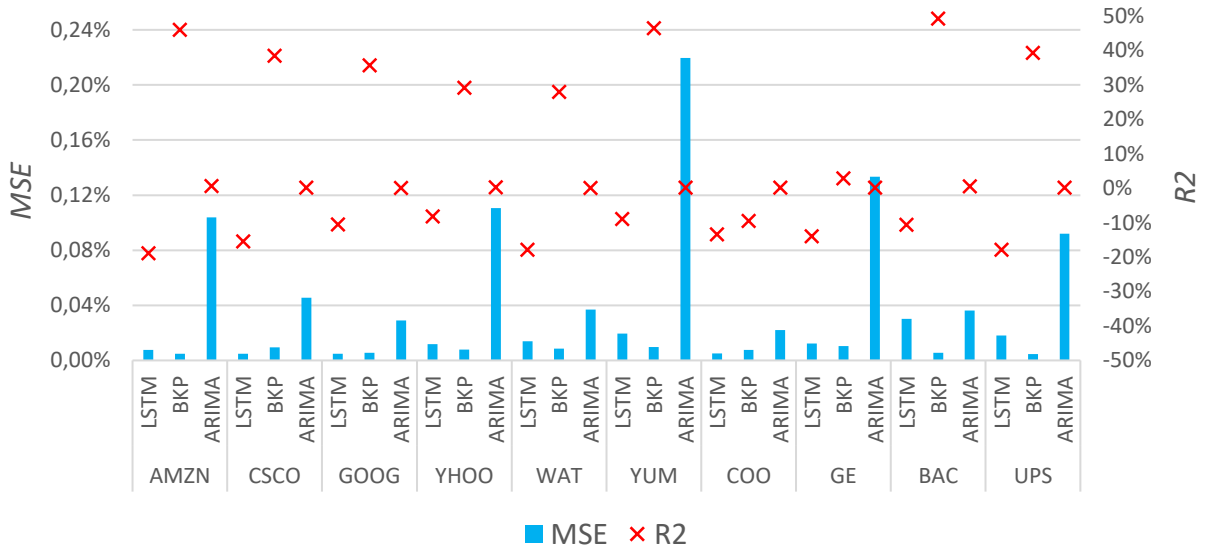
Zdroj: vlastné spracovanie

Po uplynutí doby, ktorá odpovedá našej testovacej množine uzatvára pozíciu a realizuje zisk alebo stratu. Model, ktorého predpovede nám umožnia realizovať najvyšší zisk pri obchodovaní so všetkými desiatimi titulmi budeme považovať za najlepší.

6 Výsledky predpovedí

Výsledky predpovedí modelov LSTM, Backpropagation a ARIMA môžeme pozorovať na Grafe 6 a v Tabuľke 6. Ako kritéria presnosti boli použité metriky, ktoré boli bližšie popísané v kapitole 6 - *Metriky vyhodnotenia modelu*. Požadujeme čo najmenšiu hodnotu *MSE (ľavá os grafu) a čo* najvyššiu hodnotu R^2 *(pravá os grafu).*

Graf 6: Výsledky predpovedí



Zdroj: vlastné spracovanie

Z Grafu 6 môžeme vypozorovať, že najvyššie hodnoty R^2 dosahujú modely Backpropagation a najnižšie hodnoty MSE predovšetkým modely Backpropagation. či LSTM. Podrobnejšie výsledky však zistíme z nasledujúcej tabuľky.

Hodnoty R^2 boli počítané z denných logaritmických výnosov a pod skutočným a predpovedaným výnosom rozumieme výnos za celé testovacie obdobie, teda obdobie od januára roku 2010 do júla 2015.

6.1 Rekapitulácia výsledkov

Tabuľka 7: Rekapitulácia výsledkov

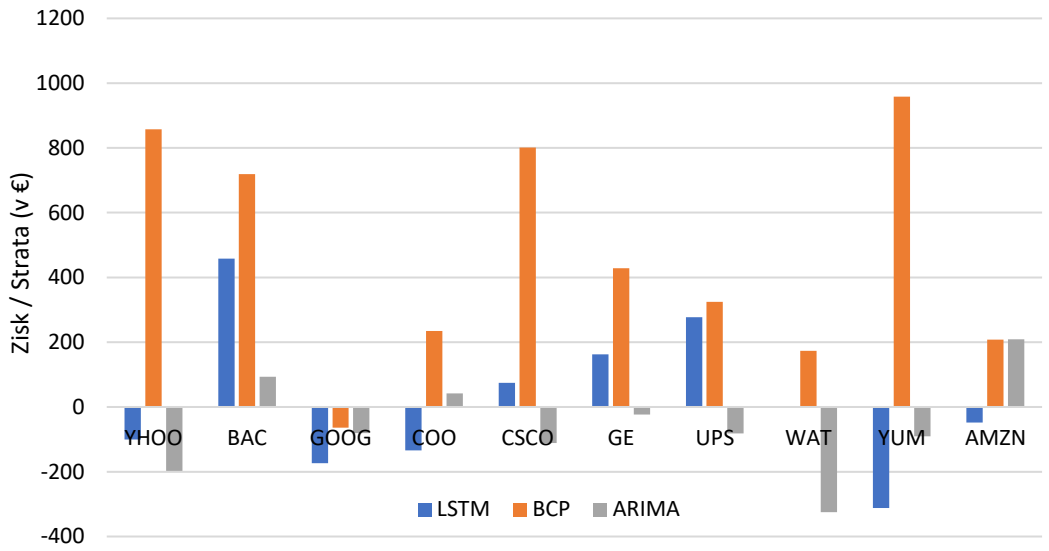
	Algoritmus	R2	MSE	MAE	Skutočný výnos	Skutočný priemerný výnos	Predpovedaný výnos	Predpovedaný priemerný výnos
AMZN	LSTM	-18.8419%	0.0077%	0.6382%	18.0498%	0.052%	16.5777%	0.0478%
	BKP	45.9827%	0.0048%	0.5779%			16.7705%	0.0476%
	ARIMA	-0.6548%	0.1038%	2.2778%			19.0241%	0.0541%
CSCO	LSTM	-15.392%	0.0049%	0.5018%	4.7873%	0.0138%	6.1298%	0.0177%
	BKP	38.5022%	0.0094%	0.6943%			3.1393%	0.0089%
	ARIMA	-0.1883%	0.0455%	1.5347%			5.5212%	0.0157%
GOOG	LSTM	-10.5045%	0.0049%	0.4975%	11.09%	0.0312%	11.1252%	0.0321%
	BKP	35.6975%	0.0056%	0.5468%			9.8157%	0.0279%
	ARIMA	-0.1249%	0.0291%	1.216%			13.08%	0.0392%
YHOO	LSTM	-8.2031%	0.0118%	0.8222%	3.7601%	0.0108%	4.0174%	0.0116%
	BKP	29.2072%	0.0078%	0.6674%			3.473%	0.0099%
	ARIMA	-0.3108%	0.1107%	2.5144%			2.1768%	0.0062%
WAT	LSTM	-17.8938%	0.0138%	0.7909%	2.901%	0.0084%	0.0238%	0.0068%
	BKP	27.9766%	0.0085%	0.6347%			0.0014%	0.0049%
	ARIMA	-0.1356%	0.037%	1.2861%			0.0035%	0.0099%
YUM	LSTM	-8.8834%	0.0195%	0.8567%	2.9942%	0.0086%	0.0239%	0.007%
	BKP	46.4881%	0.0097%	0.7372%			3.1205%	0.0088%
	ARIMA	0.2316%	0.2196%	2.9351%			0.3363%	0.001%
COO	LSTM	-13.4218%	0.005%	0.5262%	0.5052%	0.0015%	-0.2996%	-0.0086%
	BKP	-9.4764%	0.0077%	0.5958%			0.1668%	0.0005%
	ARIMA	0.1835%	0.0221%	1.0964%			-0.4363%	-0.0012%
GE	LSTM	-13.9542%	0.0123%	0.7888%	16.1499%	0.0465%	15.2416%	0.0439%
	BKP	2.8653%	0.0104%	0.6238%			14.4981%	0.0412%
	ARIMA	-0.2247%	0.1333%	2.6356%			15.4657%	0.0439%
BAC	LSTM	-10.5664%	0.0301%	0.1312%	12.8105%	0.0369%	12.5609%	0.0362%
	BKP	49.2974%	0.0056%	0.5779%			9.6868%	0.0275%
	ARIMA	-0.5114%	0.0362%	0.1435%			10.7272%	0.0305%
UPS	LSTM	-17.8938%	0.018%	0.1066%	10.3517%	0.0298%	11.7879%	0.034%
	BKP	39.3612%	0.0047%	0.5114%			6.4439%	0.0183%
	ARIMA	-0.1963%	0.0921%	2.2891%			8.9719%	0.055%

Zdroj: vlastné spracovanie

6.2 Výsledky obchodnej stratégie

V prípade, že by sme postupovali podľa investičnej stratégie popísanej v podkapitole 7.6. dosiahli by sme nasledujúcich výsledkov. Najlepší pre investičné rozhodovanie sa javí model Backpropagation. Modely LSTM a ARIMA vychádzajú, čo sa týka úspešnosti investičnej stratégie, podobne. U niektorých titulov vychádza lepšie model LSTM, u iných zas ARIMA. Graf 7 zobrazuje konečné stavy účtov po uplynutí testovacej doby, ktorá pre nás znamená dobu, počas ktorej sme nakupovali či predávali akcie na dennej báze. V prípade, že konečný stav účtu presahuje vstupný kapitál 1000 €, realizujeme zisk, ktorý je zobrazený na grafe.

Graf 7: Výsledky obchodnej stratégie



Zdroj: vlastné spracovanie

Presnejšie môžeme výsledky porovnať na základe Tabuľky 7, z ktorej vidíme, že pri rozhodovaní podľa algoritmu Backpropagation by sme realizovali zisky u deviatich titulov z desiatich. Jedine v prípade akcií spoločnosti Google by sme po skončení testovacieho obdobia realizovali stratu. V prípade algoritmu LSTM by sme dosiahli zisk u piatich titulov a pri rozhodovaní na základe modelu ARIMA len v troch prípadoch z desiatich.

Tabuľka 7: Konečné zostatky účtov po aplikovaní investičnej stratégie

	LSTM	BKP	ARIMA
YHOO	899,5232	1857,551	802,8277
BAC	1458,388	1718,633	1093,058
GOOG	826,5927	936,0063	919,0322
COO	866,0666	1235,17	1041,531
CSCO	1074,093	1801,189	888,3364
GE	1162,605	1428,139	976,9151
UPS	1277,578	1324,81	918,2221
WAT	1001,866	1173,196	675,4396
YUM	688,283	1958,064	909,122
AMZN	951,6923	1207,619	1209,164

Zdroj: vlastné spracovanie

Záver

V práci sme sa venovali zostrojeniu troch modelov – rekurentnej neurónovej siete LSTM, doprednej siete využívajúcej algoritmus učenia Backpropagation a modelu ARIMA. Na trénovacích dátach sme modely naučili a následne na testovacích sme vyskúšali ich predikčnú schopnosť.

K analýze sme použili informácie o cenách desiatich akcií od začiatku januára roku 2010 po koniec decembra roku 2016, a to konkrétne ich otváracích, uzatváracích cenách, najvyššej a najnižšej cene v danom dni a zobchodovaného objem u akcií desiatich firiem obchodovaných na New York Stock Exchange - *Google, Yahoo, Amazon* a *Cisco*, ďalej *Cooper Companies, General Electric, Yum! Brands* a *Bank of America*. Predikovaná hodnota bola uzatváracia cena daného dňa.

Napriek tomu, že v literatúre je často k predpovedi časových radov preferovaný model LSTM, v uskutočnenej analýze sa javí k použitej investičnej stratégii vhodnejší model, ktorý využíva algoritmus Backpropagation. Ide teda o pomerne prekvapiví záver, ktorý je možno vysvetliť komplexnosťou a netrivialnosťou modelu LSTM, kedy by pri obšírnejšej analýze a lepšom nastavení parametrov bolo možné získať lepšie výsledky.

Na záver zostáva skonštatovať, že neurónové siete môžu predstavovať zaujímavú alternatívu ku tradičným prístupom k modelovaniu časových radov, avšak netreba zabúdať že vo všetkých spomínaných prípadoch vychádzame z historických údajov o minulých cenách / zobchodovanom objeme na troch, ale pre dôslednejšiu analýzu by bolo vhodné rozšíriť model o dáta vypovedajúce o hospodárení danej firmy (informácie o zisku, kapitály, vyplatených dividendách..) či o dáta vypovedajúce situácii na trhu (úrokové miery, technické indikátory, tržný sentiment).

Použitá literatúra

ARLT, Josef, ARLTOVÁ, Markéta, RUBLÍKOVÁ, Eva: Analýza ekonomických časových řad s příklady. Praha: Vysoká škola ekonomická, 2002. 147 s. ISBN 80-245-0307-7.

BERKA, Petr. *Dobývání znalostí z databází*. Praha: Academia, 2003. 366 s. ISBN 80-200-1062-9.

FÁBRY, Jan. Matematické modelování. Vyd. 1. Praha: Oeconomica, 2009, 146 s. ISBN 978-80-245-1266-2.

FORBELSKÁ, Marie. *Stochastické modely časových řad* [online]. Brno: Ústav matematiky a statistiky, Přírodovědecká fakulta Masarykovy univerzity, 2013 [cit. 2018-12-05]. Dostupné z: https://www.math.muni.cz/~vondra/uvm/vystupy/KA1/M5201/M5201.pdf

HEATON Jeff. *Introduction to Neural Networks for Java*. 2nd Edition. Heaton Research, Inc.; 2 edition, 2008.

HOLČÍK, Jiří, KOMENDA, Martin (eds.) a kol. *Matematická biologie: e-learningová učebnice* [online]. 1. vydání. Brno: Masarykova univerzita, 2015. ISBN 978-80-210-8095-9.

HUŠEK, Roman. *Ekonometrická analýza*. Vyd. 1. Praha: Oeconomica, 2007. 367 s. ISBN 978-80-245-1300-3.

KITTNAR, Otomar. *Lékařská fyziologie*. Praha: Grada, 2011. ISBN 978-80-247-3068-4.

KŘIVÝ, Ivan. *Analýza časových řad* [online]. Ostrava: Ostravská univerzita v Ostravě, 2012, 122 s. [cit. 2018-12-01]. Dostupné z: http://www1.osu.cz/~bujok/files/ancas.pdf

KVASNIČKA, Vladimír, Ľubica BEŇUŠKOVÁ, Jiří POSPÍCHAL, Igor FARKAŠ, Peter TIŇO a Andrej KRÁĽ. *Úvod do teórie neurónových sietí* [online]. [cit. 2018-04-18]. Dostupné z: https://encyklopediapoznania.sk/data/eknihy/informatika/uvod_do_teorie_neuronovych_sieti.pdf

LAPEDES, Alan a Robert FARBER. *How neural nets work*. 1987.

MARČEK, D. – MARČEK, M. 2006. Neurónové siete a ich aplikácie. Žilina: EDIS – vydavateľstvo ŽU, 2006

NOVÁK, Mirko a kol. *Umělé neuronové sítě: teorie a aplikace*. Praha: Beck, 1998. xix, 382 s. Učebnice informatiky. ISBN 80-7179-132-6.

PRÁŠKOVÁ, Zuzana. *Základy náhodných procesů II*. 1. vyd. Praha: Karolinum, 2007. 151 s. Učební texty Univerzity Karlovy v Praze. ISBN 978-80-246-0971-3.

SINČÁK, Peter. *Neurónové siete - Inžiniersky prístup (1. diel): Čo je to perceptrón?* [online]. [cit. 2018-04-01]

SMOLENSKY, Paul. *Information processing in dynamical systems: foundations of Harmony Theory*. 1986.

ŠÍMA, Jiří a NERUDA, Roman. *Teoretické otázky neuronových sítí*. Vyd. 1. Praha: Matfyzpress, 1996. 390 s. ISBN 80-85863-18-9.

TANG, Zaiyong, Chrys de ALMEIDA a Paul A. FISHWICK. *Time series forecasting using neural networks vs. Box- Jenkins methodology*. 1991.

THOMAS, Andy. *An introduction to neural networks for beginners: Adventures in Machine Learning* [online]. [cit. 2019-01-13].

WERBOS, Paul J. *Generalization of backpropagation with application to a recurrent gas market model*. 1988.

ZHANG, G. Peter. *Time series forecasting using a hybrid ARIMA and neural network mode*. 2011.

Internetové zdroje

https://www.altumintelligence.com/articles/a/Time-Series-Prediction-Using-LSTM-Deep-Neural-Networks

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/

https://www.quantinsti.com/blog/artificial-neural-network-python-using-keras-predicting-stock-price-movement#disqus_thread

https://www.kaggle.com/dgawlik/nyse#prices-split-adjusted.csv

Zoznam Obrázkov

Obrázok 1: Biologický neurón 10
Obrázok 2: Formálny neurón 12
Obrázok 3: Aktivačné funkcie neurónu 13
Obrázok 4: Minimalizácia chybovej funkcie v procese adaptácie 15
Obrázok 5: Chyba na trénovacej a validačnej množine 16
Obrázok 6: Voroniho mozaika 17
Obrázok 7: Lineárna separácia tried v príkladovom priestore 19
Obrázok 8: Topológia Rosenblattovho perceptrónu 20
Obrázok 9: Rozdelenie konvexných oblastí pomocou priamok 21
Obrázok 10: Organizácia viacvrstvej perceptrónovej siete 22
Obrázok 11: Porovnanie topológie rekurentnej a doprednej siete 24
Obrázok 12: Štruktúra LSTM siete 25
Obrázok 13: Hodnota pamäťovej bunky 26
Obrázok 14: Prechod vstupnej informácie cez forget gate 27
Obrázok 15: Prechod informácie input gate 27
Obrázok 16: Úprava hodnoty pamäťovej bunky 28
Obrázok 17: Výpočet výstupu vrstvy LSTM siete 28
Obrázok 18: Architektúra LSTM siete 29
Obrázok 19: Prvých desať riadkov dataframe-u s cenami akcií spoločnosti Amazon 41
Obrázok 20: Požadovaná štruktúra trénovacích dát pre učenie s učiteľom 43
Obrázok 21: Štruktúra trénovacích dát pre predpoveď cien akcií spoločnosti Cisco 43
Obrázok 22: Príklad maticovej štruktúry po dátovej úprave pre akcie spoločnosti Cisco 45
Obrázok 23: Úspešnosť LSTM modelu pri rôznom počte epoch 47
Obrázok 24: Úspešnosť LSTM modelu pri rôznych hodnotách parametru dropout 49
Obrázok 25: Úspešnosť LSTM modelu pri rôznych hodnotách parametru parametru window size 51
Obrázok 26: Úspešnosť LSTM modelu pri rôznom počte neurónov 53
Obrázok 27: Úspešnosť BKP modelu pri rôznych hodnotách parametru dropout 56
Obrázok 28: Úspešnosť BKP modelu pri rôznom počte epoch 57
Obrázok 29: Úspešnosť BKP modelu pri rôznom počte neurónov 58
Obrázok 30: Výstup Dickey-Fullerovho testu pre pôvodnú časovú radu 61
Obrázok 31: Pôvodná a diferencovaná časová rada 62
Obrázok 32: Výstup Dickey-Fullerovho testu pre novú časovú radu 63

Zoznam tabuliek

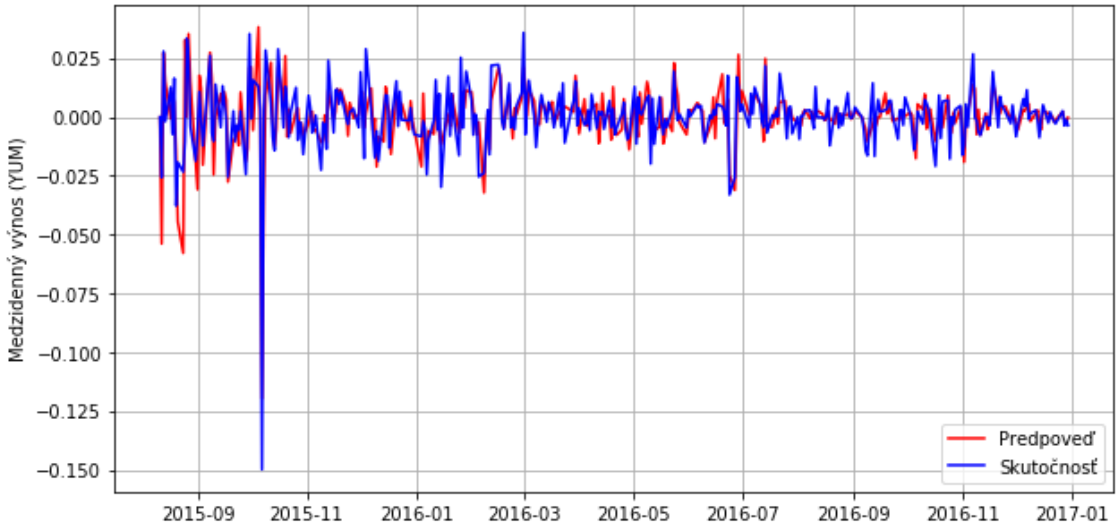
Tabuľka 1: Princípy pre určovanie parametrov p a q 37
Tabuľka 2: Výsledky testovania optimálnych hodnôt parametru počet epoch 46
Tabuľka 3: Výsledky testovania optimálnych hodnôt parametru dropout pre LSTM sieť 48
Tabuľka 4: Výsledky testovania optimálnych hodnôt parametru window size pre LSTM sieť 50
Tabuľka 3: Výsledky testovania optimálnych hodnôt parametrov dropout a počet epoch pre BKP 55
Tabuľka 5: Rozhodovacia stratégia investora 65
Tabuľka 6: Rekapitulácia výsledkov 67

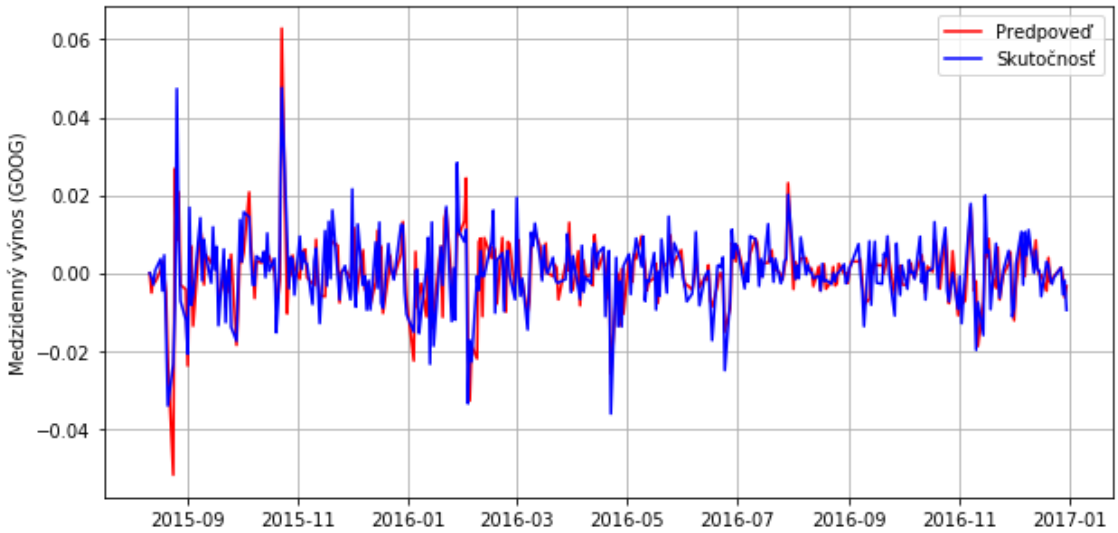
Zoznam grafov

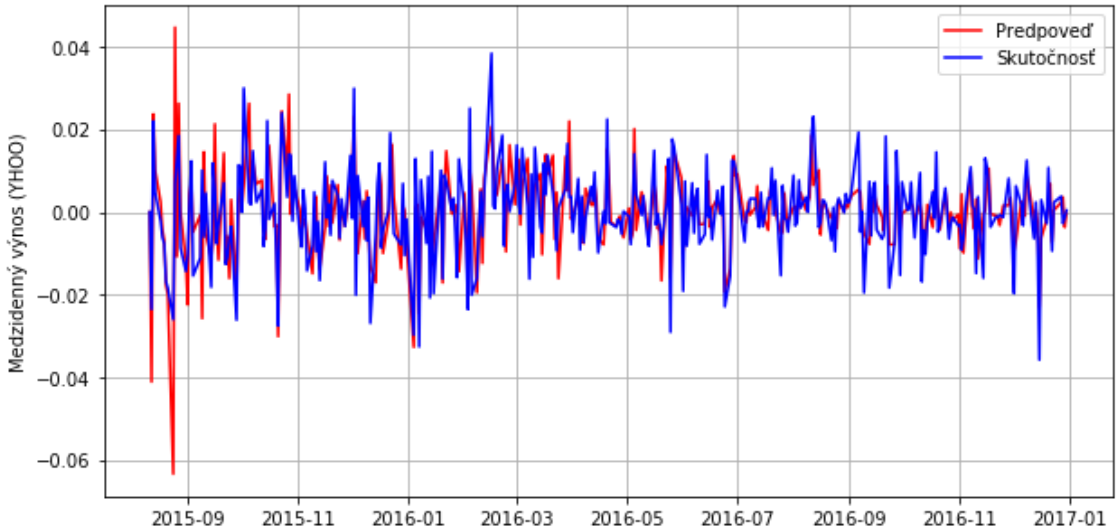
Graf 1: Princípy pre určovanie parametrov p a q 42
Graf 2: Výsledky testovania optimálnych hodnôt parametru počet epoch 60
Graf 3: Výsledky testovania optimálnych hodnôt parametru dropout pre LSTM sieť 62
Graf 4: Výsledky testovania optimálnych hodnôt parametru window size pre LSTM sieť..... 63
Graf 5: Výsledky testovania optimálnych hodnôt parametrov dropout a počet epoch pre BKP 64
Graf 6: Výsledky predpovedí 66
Graf 7: Výsledky obchodnej stratégie..... 68

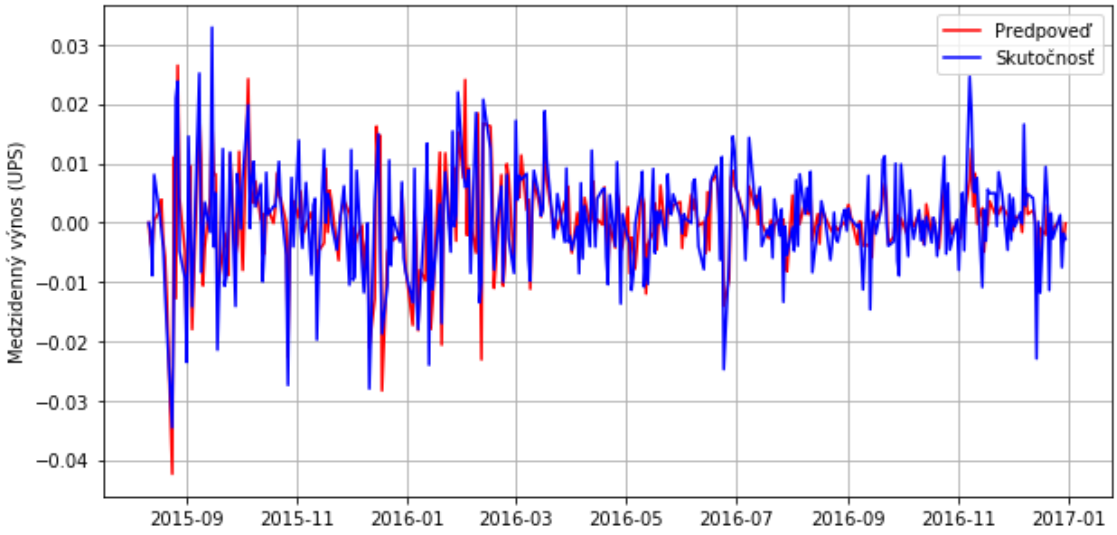
Príloha

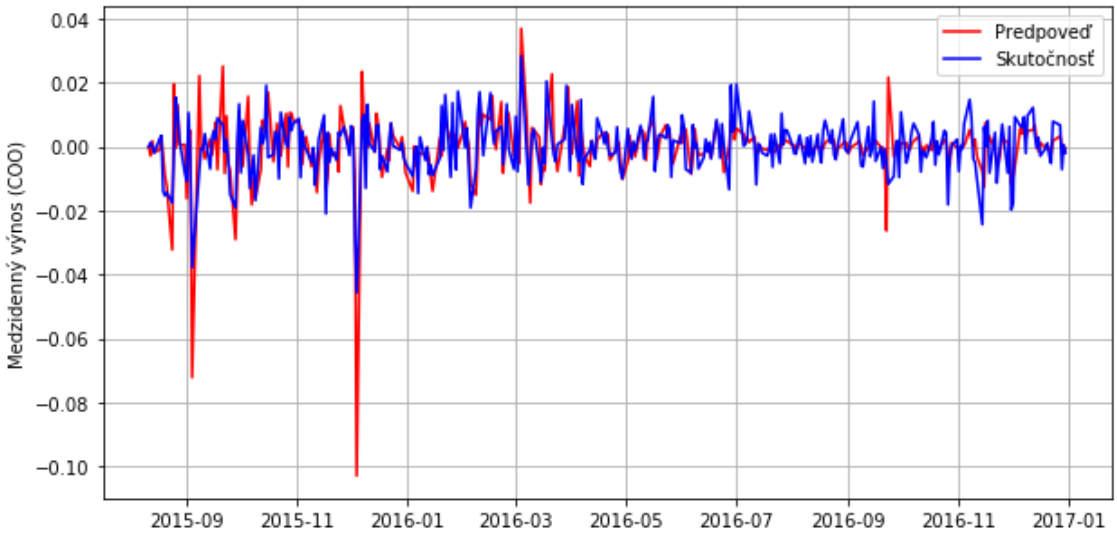
Výsledky Backpropagation modelu

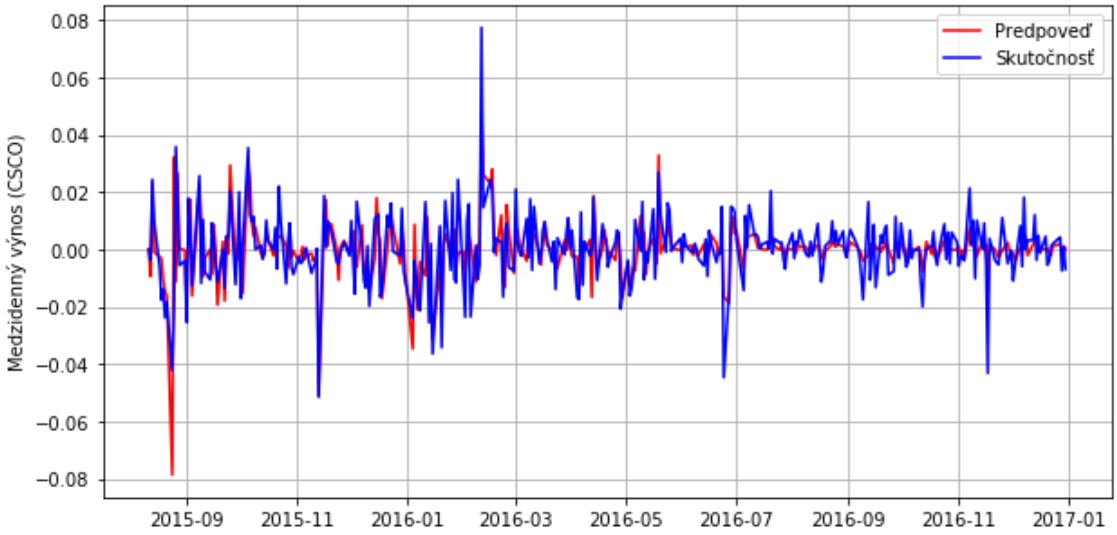


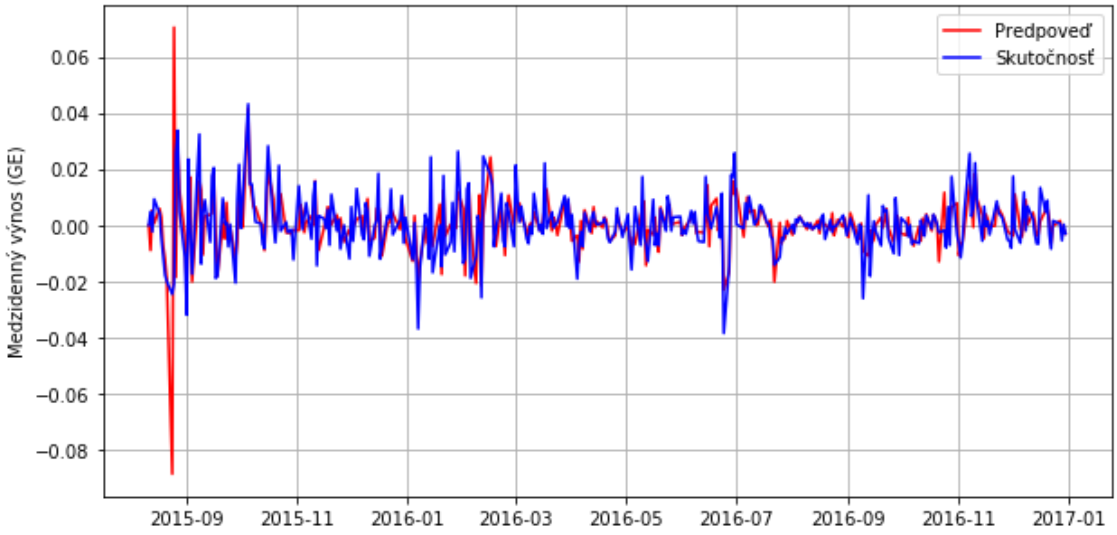


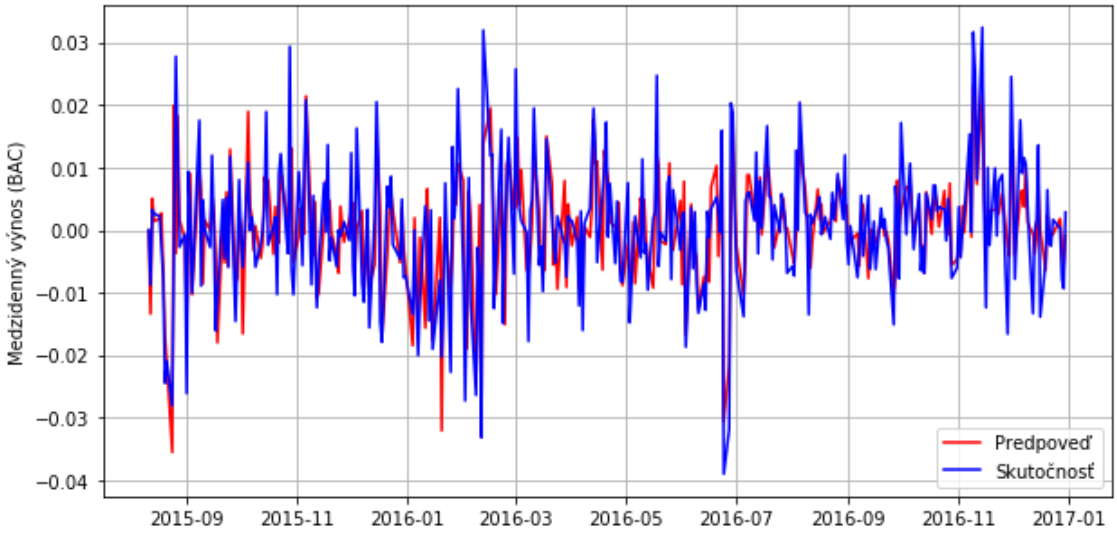


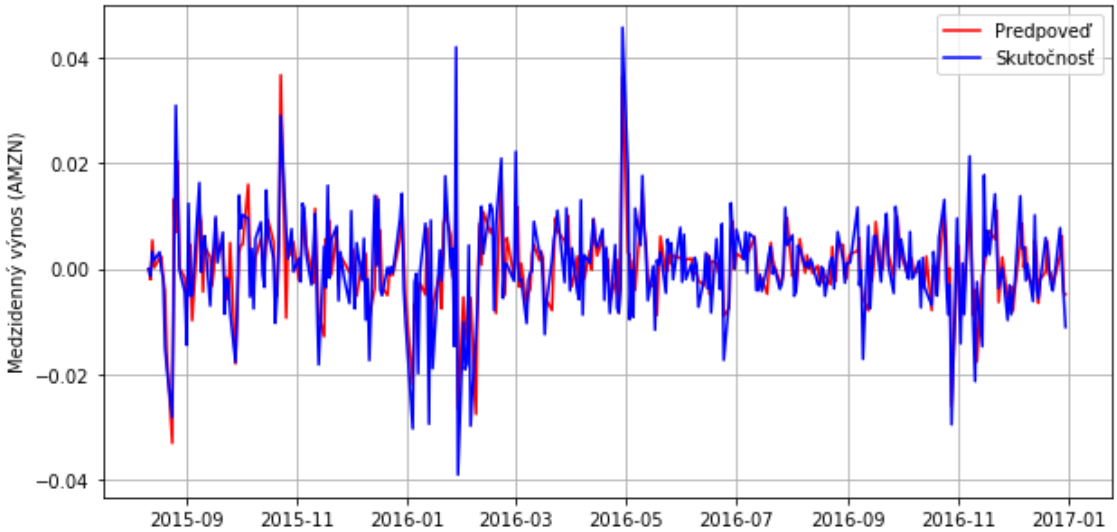

Predpoveď
Skutočnosť
Medzidenný výnos (UPS)
0.03
0.02
0.01
0.00
−0.01
−0.02
−0.03
−0.04
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01

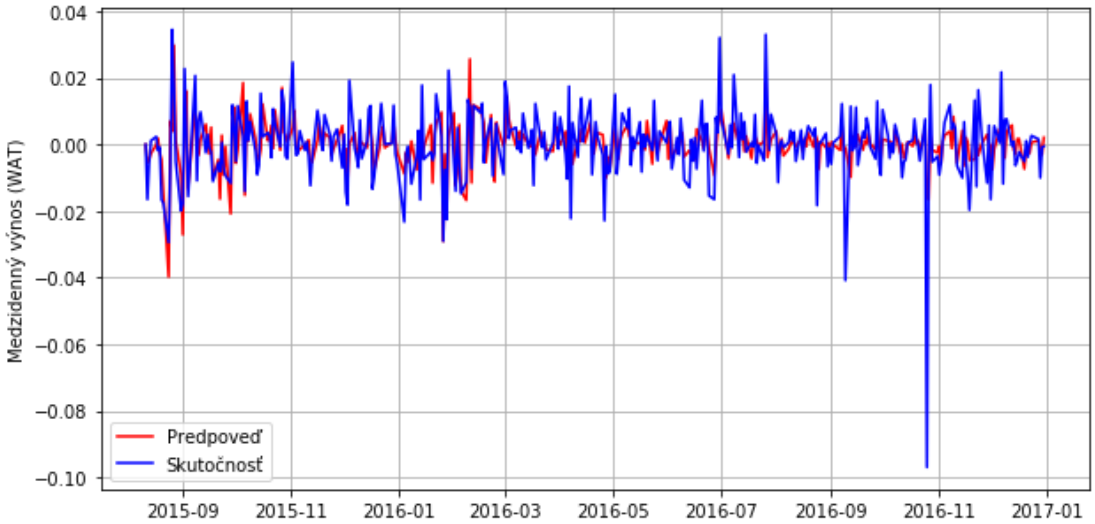

Predpoveď
Skutočnosť
Medzidenný výnos (COO)
0.04
0.02
0.00
−0.02
−0.04
−0.06
−0.08
−0.10
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01


Predpoveď
Skutočnosť
Medzidenný výnos (CSCO)
0.08
0.06
0.04
0.02
0.00
−0.02
−0.04
−0.06
−0.08
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01

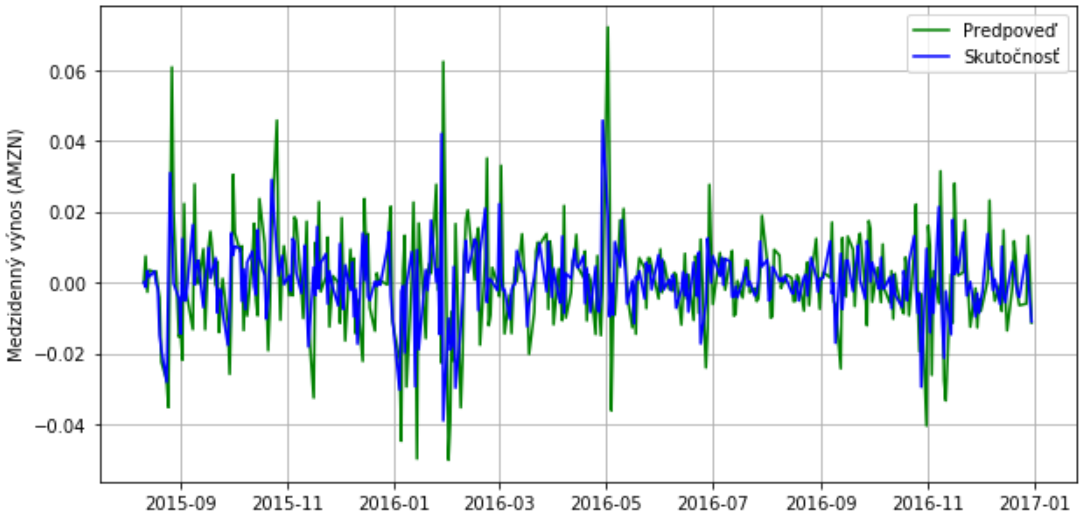

Predpoveď
Skutočnosť
Medzidenný výnos (GE)
0.06
0.04
0.02
0.00
−0.02
−0.04
−0.06
−0.08
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01

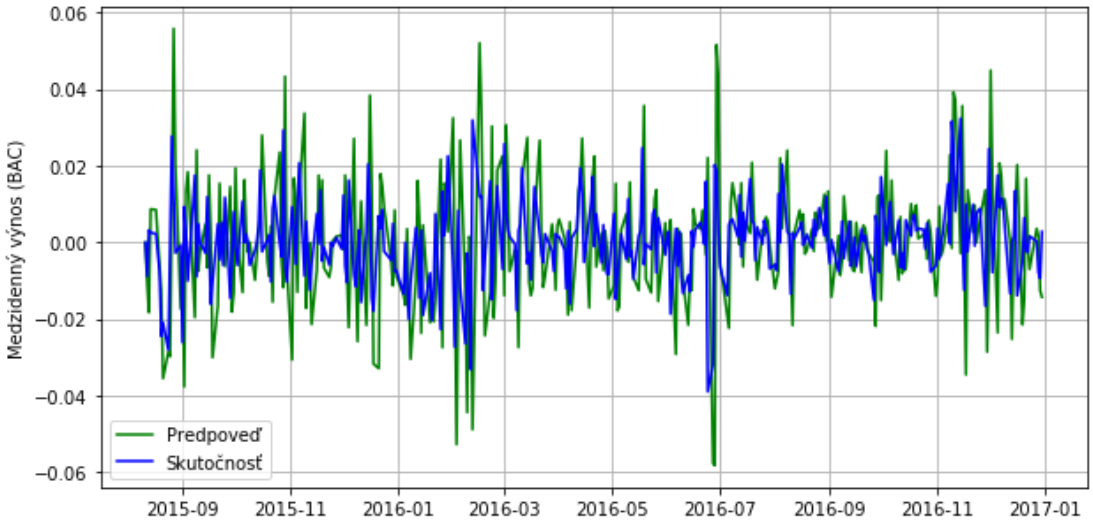

Medzidenný výnos (BAC)
0.03
0.02
0.01
0.00
−0.01
−0.02
−0.03
−0.04
Predpoveď
Skutočnosť
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01

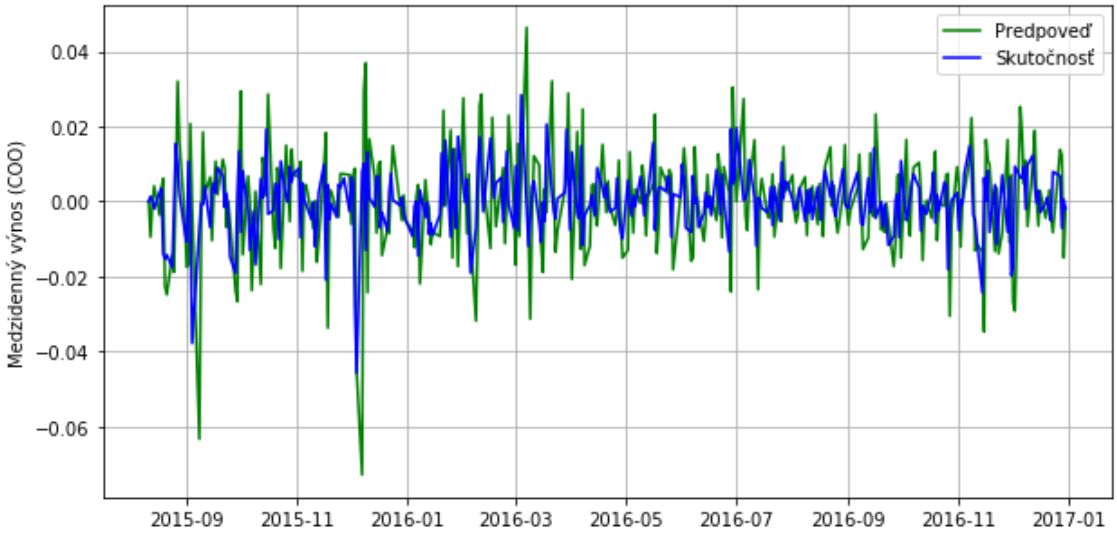

Predpoveď
Skutočnosť
Medzidenný výnos (AMZN)
0.04
0.02
0.00
−0.02
−0.04
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01

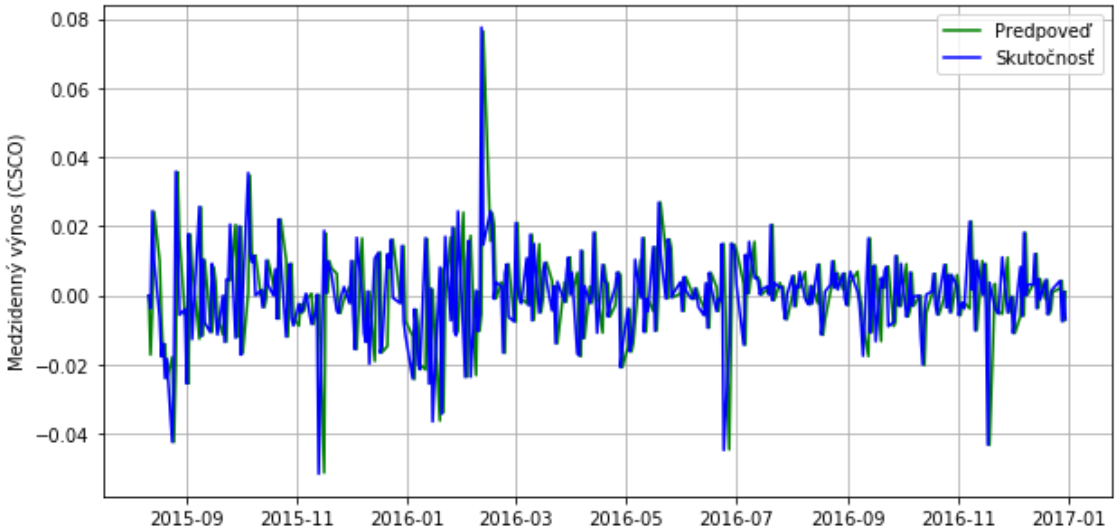


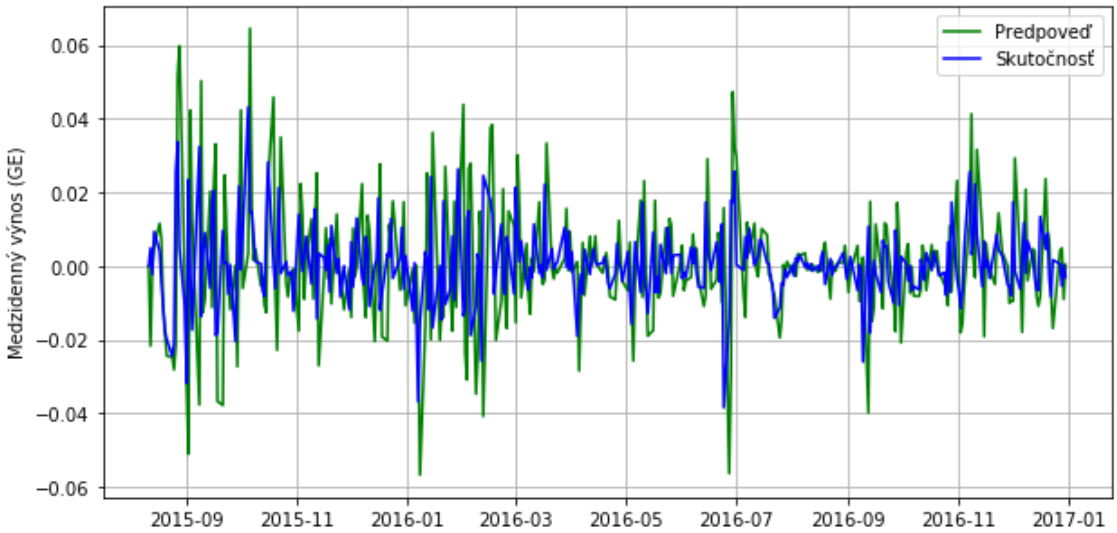
Výsledky ARIMA modelu

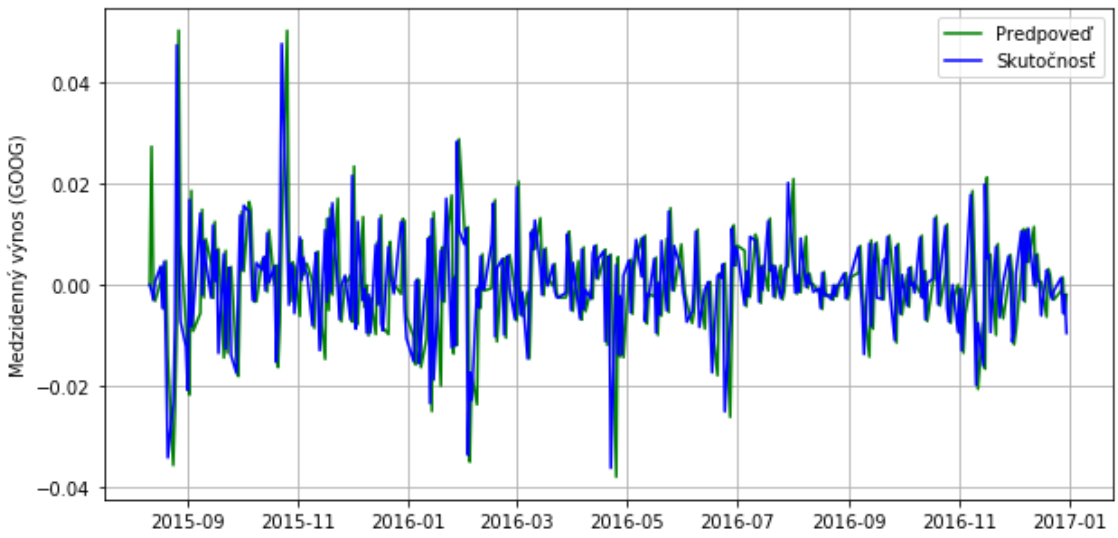


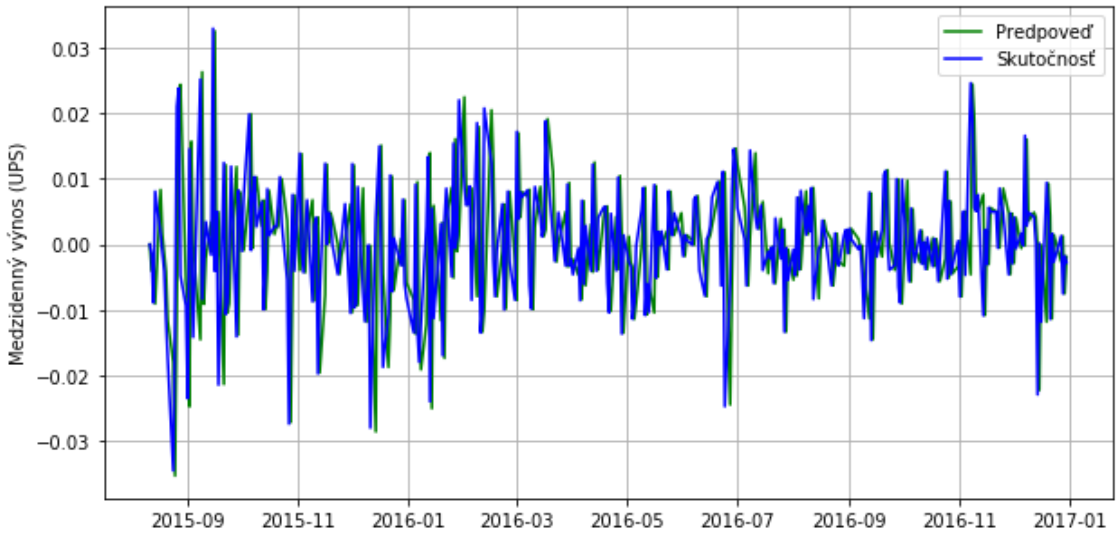


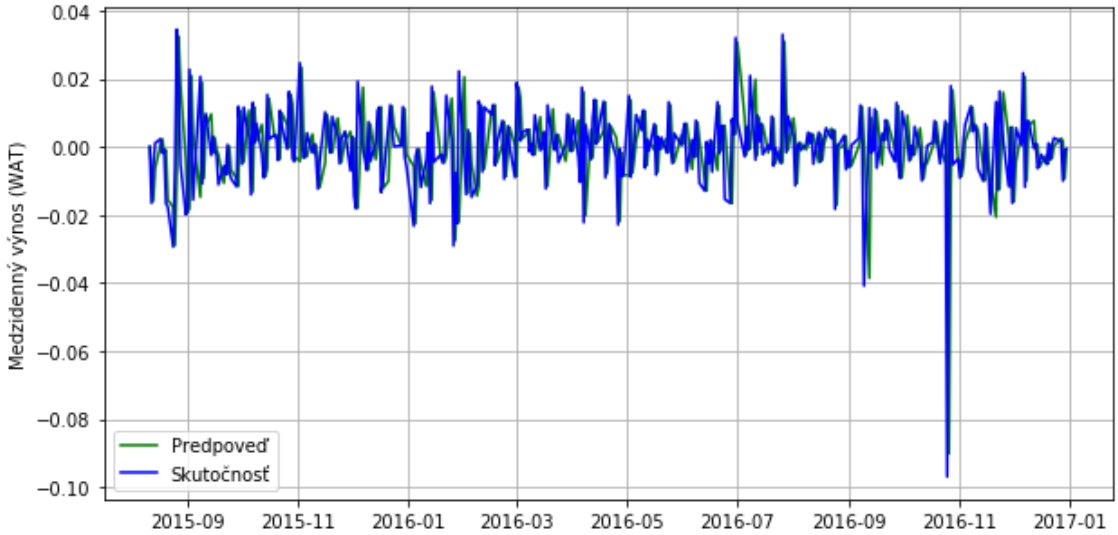

Predpoveď
Skutočnosť
Medzidenný výnos (COO)
0.04
0.02
0.00
−0.02
−0.04
−0.06
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01

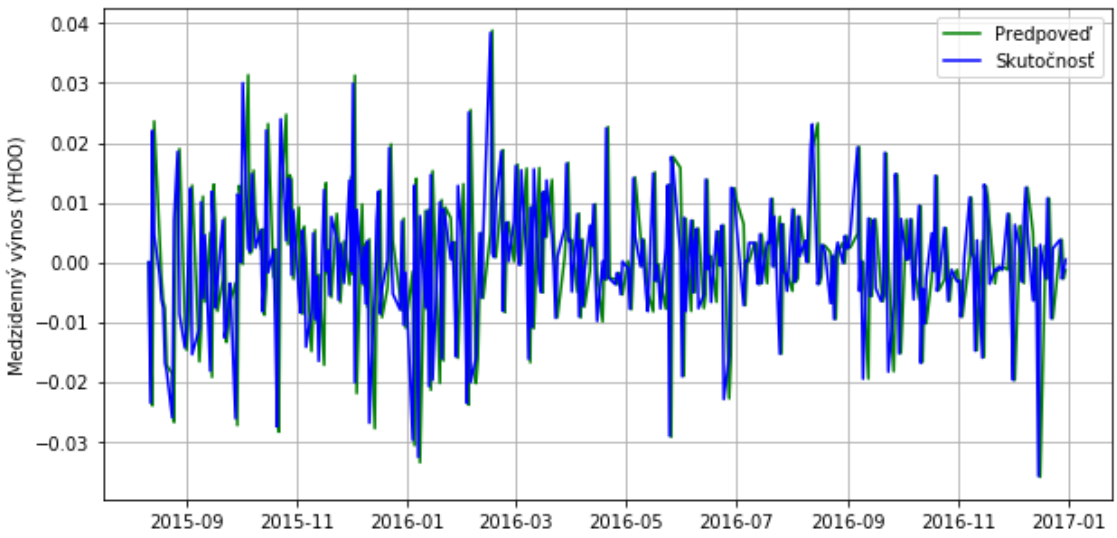

Predpoveď
Skutočnosť
Medzidenný výnos (CSCO)
0.08
0.06
0.04
0.02
0.00
−0.02
−0.04
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01

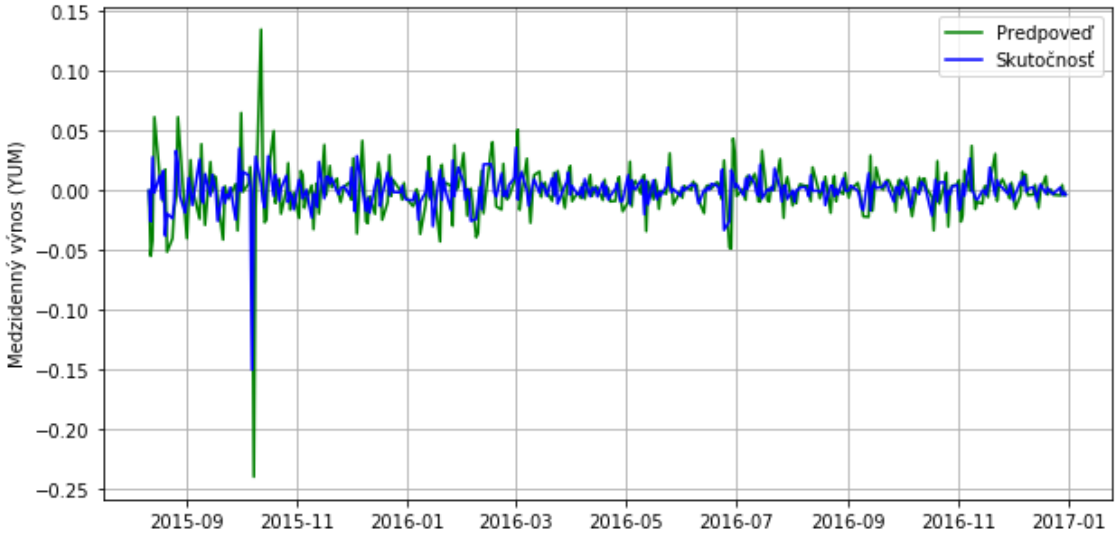

Predpoveď
Skutočnosť
Medzidenný výnos (GE)
0.06
0.04
0.02
0.00
−0.02
−0.04
−0.06
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01


Predpoveď
Skutočnosť
Medzidenný výnos (GOOG)
0.04
0.02
0.00
−0.02
−0.04
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01

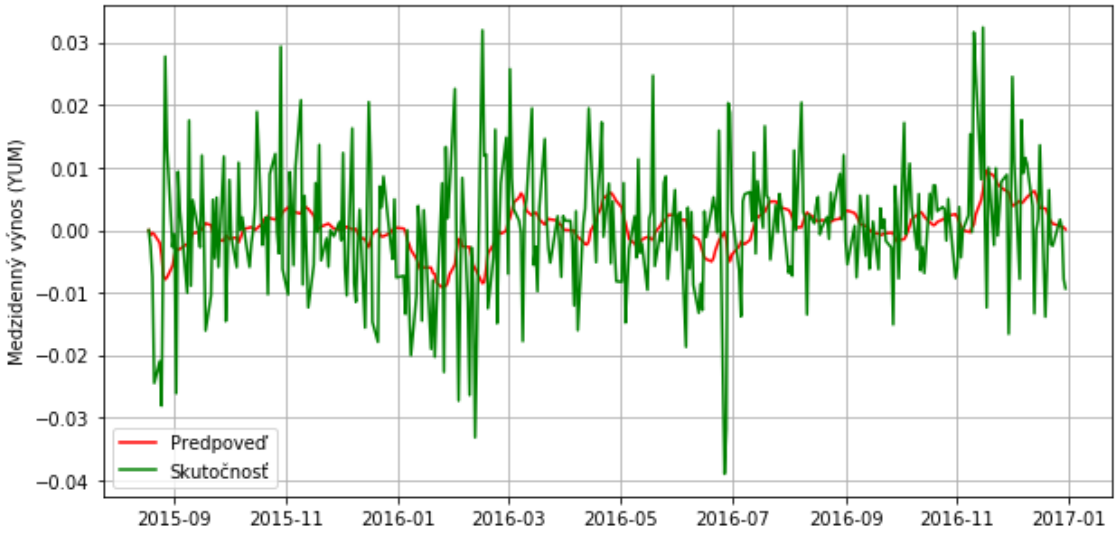

Predpoveď
Skutočnosť
Medzidenný výnos (UPS)
0.03
0.02
0.01
0.00
−0.01
−0.02
−0.03
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01

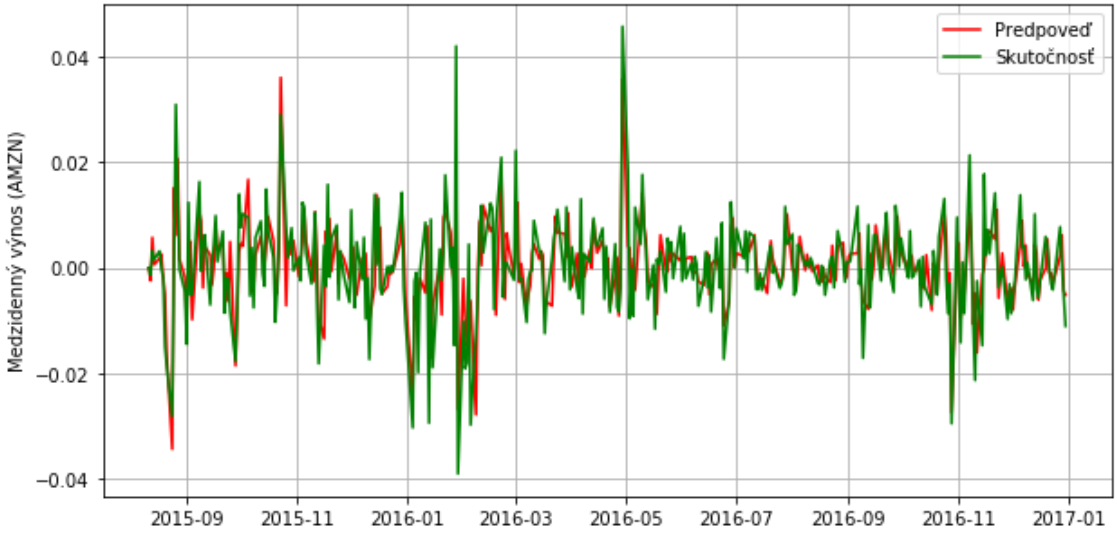

Medzidenný výnos (WAT)
0.04
0.02
0.00
−0.02
−0.04
−0.06
−0.08
−0.10
Predpoveď
Skutočnosť
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01

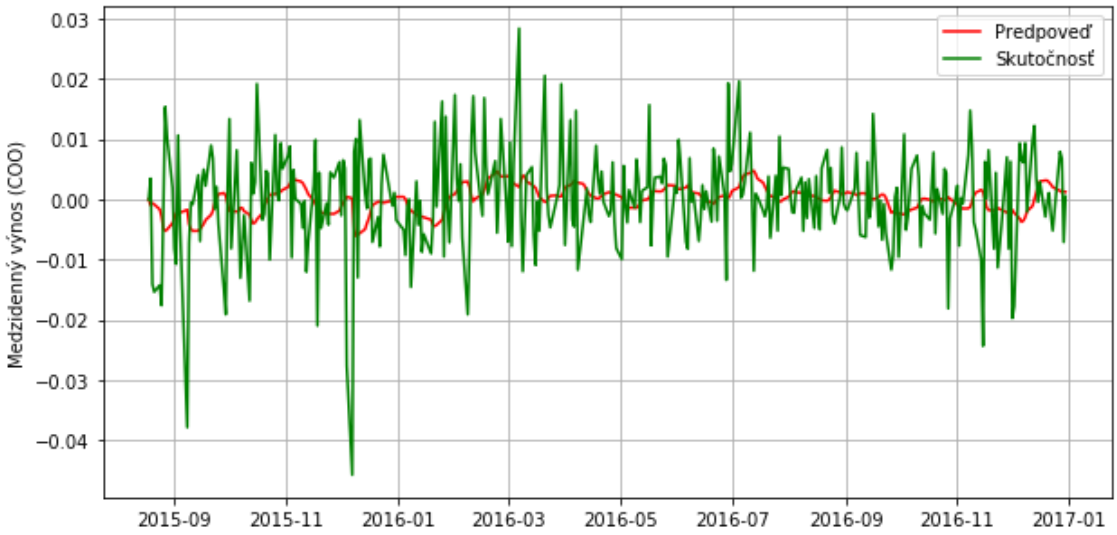


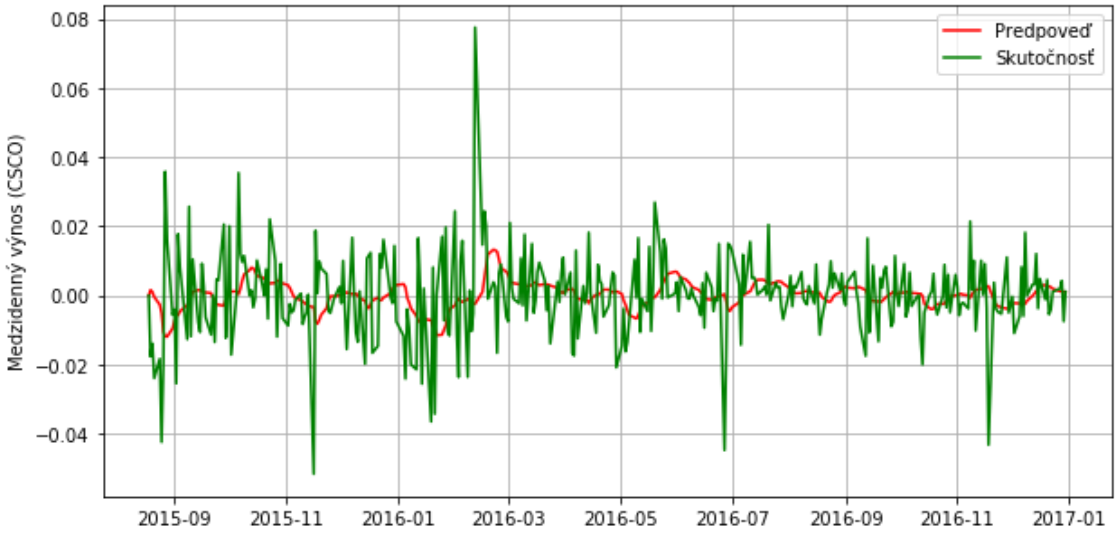


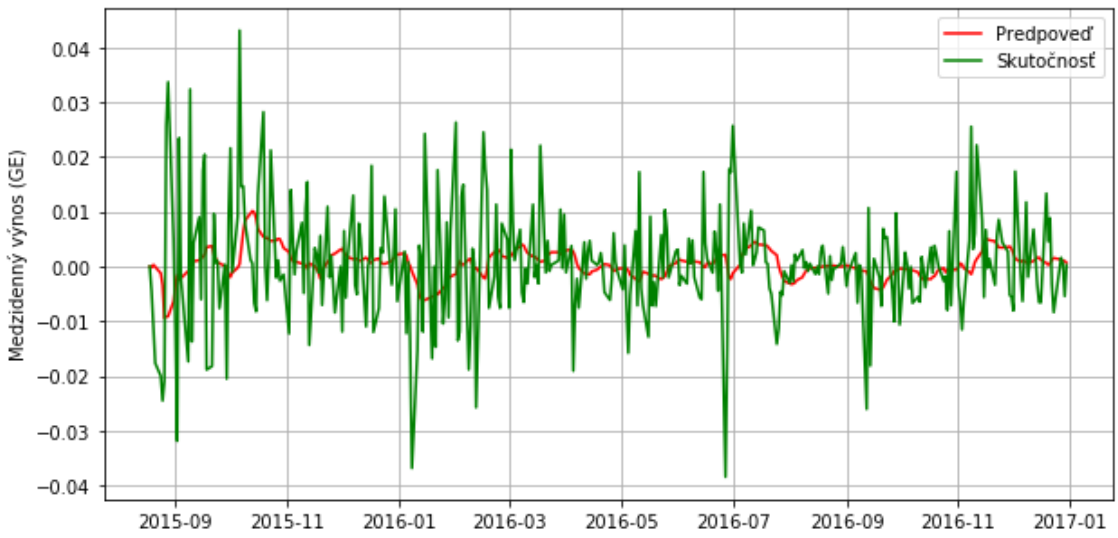
Výsledky LSTM modelu

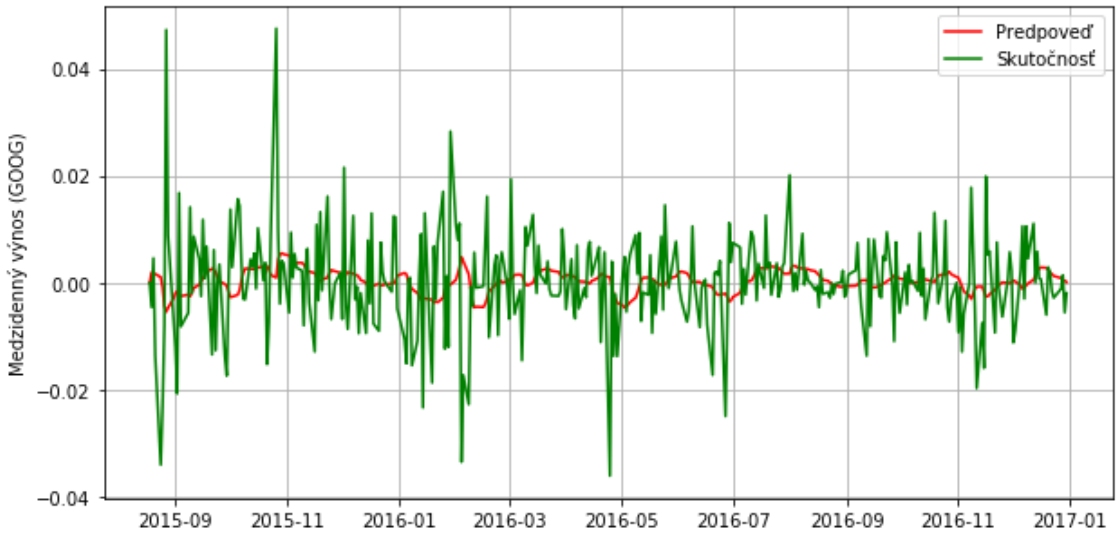


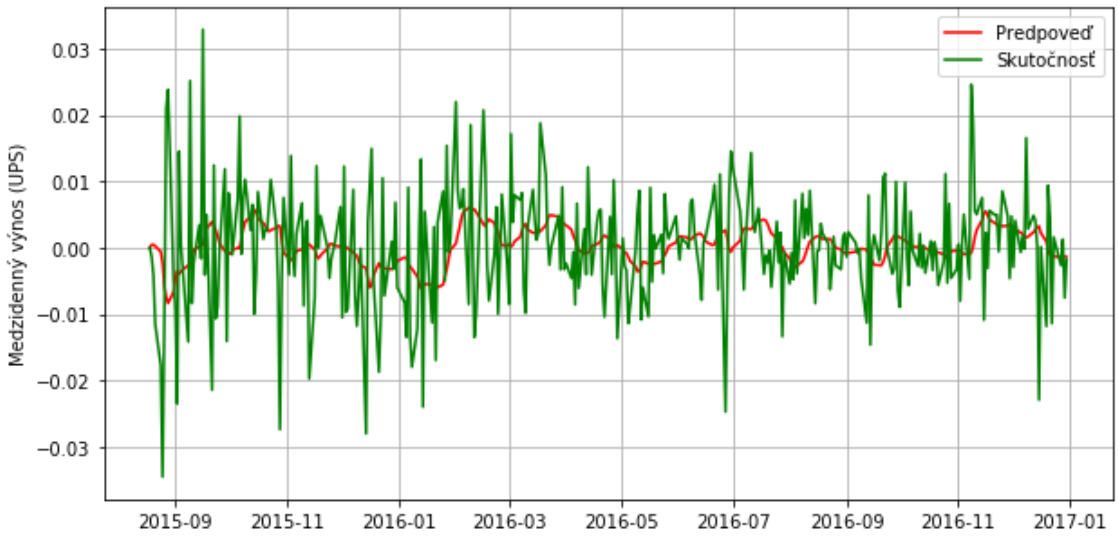

Predpoveď
Skutočnosť
Medzidenný výnos (AMZN)
0.04
0.02
0.00
−0.02
−0.04
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01

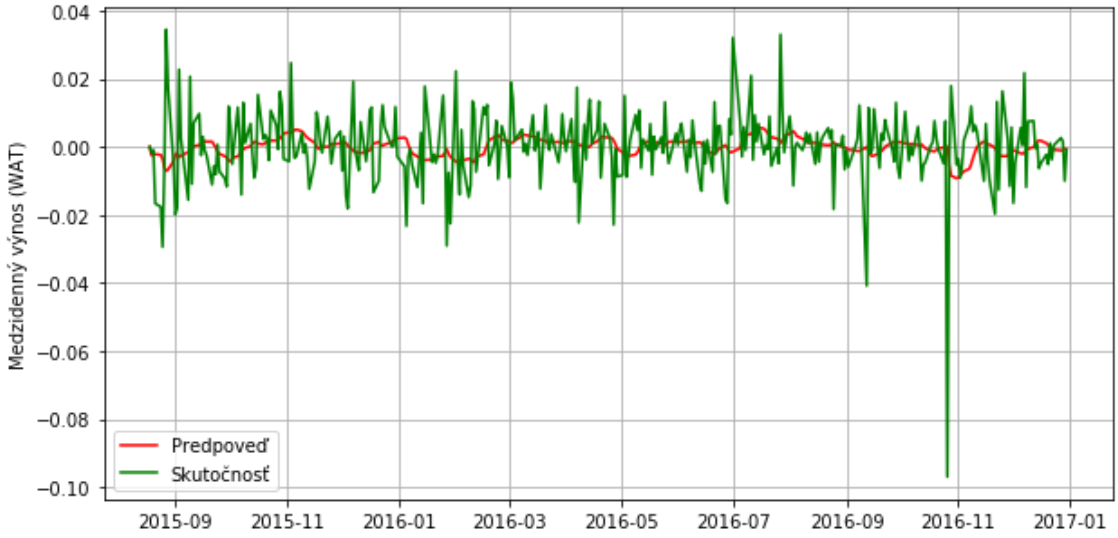

Predpoveď
Skutočnosť
Medzidenný výnos (COO)
0.03
0.02
0.01
0.00
−0.01
−0.02
−0.03
−0.04
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01

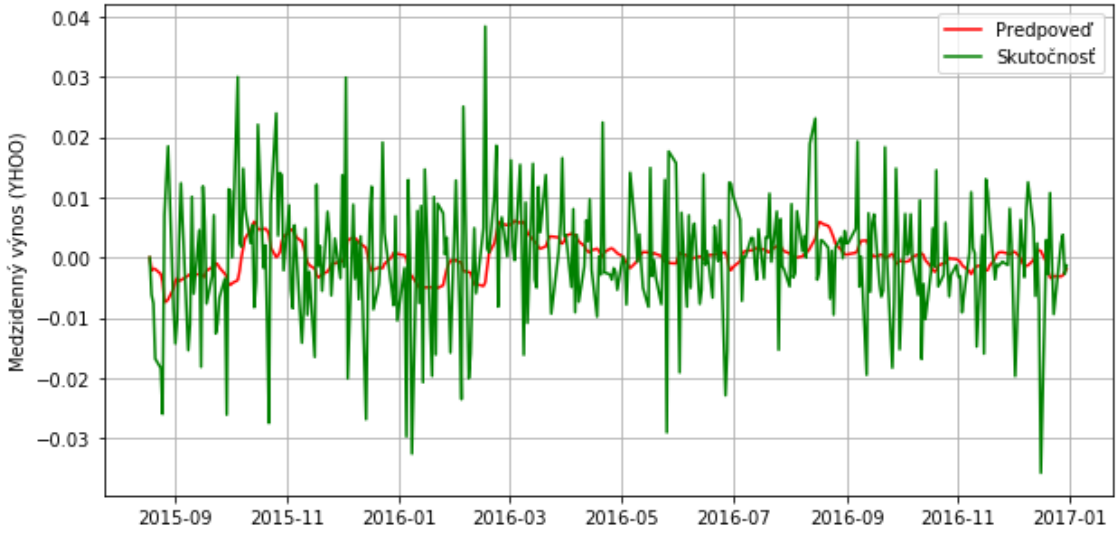

Predpoveď
Skutočnosť
Medzidenný výnos (CSCO)
0.08
0.06
0.04
0.02
0.00
−0.02
−0.04
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01


Predpoveď
Skutočnosť
Medzidenný výnos (GE)
0.04
0.03
0.02
0.01
0.00
−0.01
−0.02
−0.03
−0.04
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01


Predpoveď
Skutočnosť
Medzidenný výnos (GOOG)
0.04
0.02
0.00
−0.02
−0.04
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01


Predpoveď
Skutočnosť
Medzidenný výnos (UPS)
0.03
0.02
0.01
0.00
−0.01
−0.02
−0.03
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01


Medzidenný výnos (WAT)
0.04
0.02
0.00
−0.02
−0.04
−0.06
−0.08
−0.10
Predpoveď
Skutočnosť
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01


Medzidenný výnos (YHOO)
0.04
0.03
0.02
0.01
0.00
−0.01
−0.02
−0.03
Predpoveď
Skutočnosť
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01

Medzidenný výnos (YUM)
0.025
0.000
−0.025
−0.050
−0.075
−0.100
−0.125
−0.150
Predpoveď
Skutočnosť
2015-09
2015-11
2016-01
2016-03
2016-05
2016-07
2016-09
2016-11
2017-01