VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ALGORITMICKÉ OBCHODOVÁNÍ NA BURZE S VYUŽITÍM UMĚLÝCH NEURONOVÝCH SÍTÍ

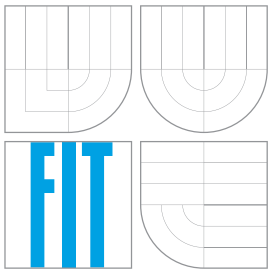
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. JAKUB BÁRTA

BRNO 2014

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ALGORITMICKÉ OBCHODOVÁNÍ NA BURZE S VYUŽITÍM UMĚLÝCH NEURONOVÝCH SÍTÍ

ALGORITHMIC TRADING USING ARTIFICIAL NEURAL NETWORKS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. JAKUB BÁRTA

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. IGOR SZÖKE, Ph.D.

BRNO 2014

Abstrakt

Tato práce se zabývá návrhem a implementací systému schopného automatizovaně obchodovat na burze. K predikci vývoje je využito neuronových sítí. Pro hledání vhodných kombinací vstupních parametrů je použit genetický algoritmus.

Abstract

This master thesis is focused on designing and implementing a software system, that is able to trade autonomously at stock market. Neural networks are used to predict future price. Genetic algorithm was used to find good combination of input parameters.

Klíčová slova

burza, neuron, neuronová síť, backpropagation, obchodování, genetický algoritmus

Keywords

stock market, neuron, neural network, backpropagation, trading, genetic algorithm

Citace

Jakub Bárta: Algoritmické obchodování na burze
s využitím umělých neuronových sítí, diplomová práce, Brno, FIT VUT v Brně, 2014

Algoritmické obchodování na burze s využitím umělých neuronových sítí

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Igora Szöke, Ph.D.

.....
Jakub Bárta
28. května 2014

Poděkování

Na tomto místě chci poděkovat vedoucímu mé diplomové práce za odborné vedení a konzultace.

Obsah

Kapitola 1

Úvod

Obchodování na burze má dlouho tradici - podle některých pramenů započalo před 6000 lety v Číně. První dochované doklady se objevily v Japonsku v 17. století. Pěstitelé rýže potřebovali na pěstování určitý základní kapitál, který ne každý farmář měl. Pokud mu chyběl, měl možnost ještě nevypěstovanou rýži předprodat. Uzavřel s nějakým velkým odběratelem (např. šlechticem) smlouvu, že mu v dohodnutém termínů dodá dohodnuté množství rýže za předem dohodnutou cenu. Z této dohody měli oba prospěch, neboť farmář věděl, že má jistého kupce pro svoji úrodu a odběratel věděl, kolik za rýži zaplatí. S touto smlouvu, kterou v dnešní době nazýváme *futures kontrakt*, mohl farmář získat úvěr pro financování pěstování.

Tento systém výborně fungoval v případě, že byla úroda víceméně průměrná. Pokud však byla velmi dobrá, bylo rýže hodně a její cena šla dolů - v tuto chvíli tratil odběratel, který byl futures kontraktem vázán odebrat rýži za vyšší cenu. V případě malé úrody cena vylétla nahoru a tratil farmář - musel dodat rýži za nižší, předem dohodnutou cenu. Lidi tedy napadlo obchodovat se samotnými futures kontrakty - pokud měl farmář podezření, že bude špatná úroda, nakoupil futures kontrakty ostatních farmářů a poté, co se opravdu urodilo špatně, mohl odkoupit rýži vázanou kontraktem za nízkou cenu a poté ji prodat za cenu trží, tj. vyšší.

Postupem času začali futures kontrakty nakupovat i lidé, kteří o pěstování rýže nic nevěděli a chtěli pouze vydělat na spekulaci na pokles nebo růst ceny [1].

A takto se zrodila burza...

Zdánlivá jednoduchost obchodování na burze s komoditami, akciemi či rozličnými deriváty s vidinou pohádkových zisků lákala a stále láká velké množství lidí. Jejich postupy a systémy pro úspěšné obchodování se různí - někteří spoléhají na své neomylné „oko“, podle kterého trhy analyzují, jiní se řídí podle globálních a lokálním ekonomických ukazatelů a další spoléhají na statistické zpracování historických cenových grafů.

Některé obchodníky časem napadlo, že by se dalo obchodování přenechat počítači - a začaly vznikat generátory automatických obchodních strategií, expertní systémy pro automatizované obchodování založené na statistice, neuronových sítích, genetických algoritmech atp. Čas ověřil, že prakticky použitelné automatizované způsoby je možné vytvořit, není to však jednoduché.

Předmětem této diplomové práce je seznámení se s principy fungování obchodování na burze a vytvoření systému, který do určité míry umožňuje predikovat vývoj cen na burzovním trhu.

[1] čerpáno z *http://www.financnik.cz/komodity/manual/komodity-jak-to-funguje.html*

Práce navazuje na semestrální projekt, ze kterého přejímá kapitolu o obchodování na burze a kapitolu věnovanou teorii.

První část práce obsahuje seznámení s principy fungování obchodování na burze, se základními pojmy a s některými přístupy, které používají reální obchodníci. Druhá část představuje teoretický popis nástrojů, které lze k predikci vývoje burzovního trhu použít. Třetí část popisuje data, která jsou k dispozici a představuje jednotlivé obchodní modely a jejich výsledky a finální porovnání jejich úspěšnosti. Čtvrtá část rozvádí návrh systému a popis implementace. Pátá část diskutuje možné směry dalšího vývoje a poslední část, závěr, přehledně shrnuje získané výsledky.

Kapitola 2

Základy obchodování na burze

V úvodu byl stručně popsán způsob obchodování s futures kontrakty. Na burzovních trzích je však možné obchodovat mnoha různými aktivy, které budou ve stručnosti představeny. Nejprve je nutné seznámit se se samotnými principy obchodování a prostředky, které obchodování zjednodušují.

2.1 Základní dělení obchodování

Existují 2 základní způsoby dělení- podle způsobu, jakým obchodník analyzuje trhy (**fundamentální analýza** a **technická analýza**), nebo podle doby, po jakou drží otevřené své pozice (**intradenní obchodování** a **poziční obchodování**).

2.1.1 Dělení podle způsobu analýzy

Fundamentální analýza

Je založena na znalostech globálním a lokálních ekonomických ukazatelů. Jedná se např. o investici do akcií firmy, pokud vím, že do firmy má přijít velký investor.

Fundamentální obchodníci se většinou specializují na několik málo (často i na jediný) trhů, o kterých mají rozsáhlé vědomosti - nejen o chování samotných trhů, ale i o odvětví, do kterého trh spadá (např. komoditní fundamentální obchodník obchodující kukuřici bude dobře informován o dopadu počasí na sklizeň kukuřice, o hlavních oblastech, kde se pěstuje, předpovědích počasí atd.)

Tento typ analýzy se špatně automatizuje a je hluboce spojen s pokrokem ve zpracování přirozeného jazyka. Automatizace lze např. provádět analýzou klíčovým slov v novinách a denících.

Technická analýza

Analýza pracuje pouze s historickými cenovými grafy, ze kterých se pokouší různými metodami predikovat další vývoj trhu.

Techničtí obchodníci se mohou specializovat na více trhů než jejich fundamentální kolegové, neboť nepotřebují tak velké znalosti. I tak ale musí věnovat nemalé množství času studiu historických cenových grafů svých trhů.

Technická analýza je častěji a jednodušeji automatizovatelná a tato práce na ní bude založena.

2.1.2 Dělení podle doby držení pozic

Pojem pozice bude vysvětlen dále, zjednodušeně lze říci, že otevřená pozice je nakoupený nebo prodaný kontrakt trhu.

Intradenní obchodování

Obchodník drží otevřené pozice v rámci jednoho obchodního dne, tj. maximálně v řádu hodin, častěji však minut. Tento způsob obchodování je časově náročnější, je nutné se po celý obchodní den soustředit. Umožňuje však potenciálně vyšší zhodnocení a brokerské poplatky jsou většinou nižší.

Tento způsob v naprosté většině volí lidé, kteří se obchodováním živí.

Poziční obchodování

Pozice mohou být otevřené dlouho dobu (v řádech měsíců i let). Trhy stačí kontrolovat několikrát týdně.

2.2 Aktiva

Aktiva jsou zjednodušeně řečeno instrumenty, které lze na různých burzovních trzích obchodovat. Dělí se na základní *podkladová aktiva* a odvozené *finanční deriváty*. Mezi podkladová aktiva patří např. akcie, komodity (obilí, ropa, zlato…) a zahraniční měny. Finančními deriváty se rozumí již dříve zmíněné futures kontrakty, opce, swapy, akciové indexy a další, které jsou nějakým způsobem odvozeny od jednoduššího derivátu nebo podkladového aktiva.

2.3 Cenové grafy

Cenové grafy jsou základním nástrojem technické analýzy, zobrazují vývoj ceny trhu v průběhu času. Jsou vždy vztaženy k nějakému časovému intervalu, v němž cenová data agregují (např. 2 minutový cenový graf agreguje data ze 2 minut do jedné úsečky grafu). Tento interval se anglicky nazývá *timeframe* a toto označení se bude nadále v práci používat.

Existuje několik druhů cenových grafů, mezi nejpoužívanější patří svíčkové grafy. Tyto grafy se skládají ze „svíček“, které jsou tvořeny 3 částmi (viz 2.1):

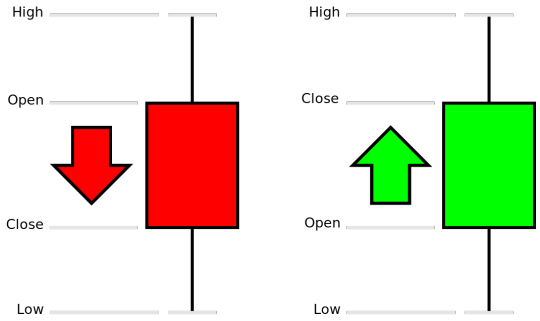
tělo určuje *open* a *close* svíčky, tj. hodnoty na kterých daný časový úsek začínal (= otevíral) a končil (= zavíral). Většinou je barevné, červená značí klesající svíčku (open je na horní hraně, close na spodní), zelená rostoucí svíčku (open je na spodní hraně, close na horní)

spodní a horní výběžek určuje *low* (resp. *high*) svíčky, tj. minimální (resp. maximální) hodnotu, na kterou se cenu v průběhu daného intervalu dostala

Z výše uvedeného je vidět, že svíčka je definovaná 4 hodnotami, přičemž je občas vhodné ji reprezentovat jedinou hodnotou. Pro tento účel se používá *typická hodnota* (angl. *typical price*, dále jen TP), která se spočítá následovně[1] :

$$TP = \frac{high + low + close}{3} \tag{2.1}$$

[1] čerpáno z *http://www.incrediblecharts.com/indicators/typical_price.php*



Obrázek 2.1: „Svíčka“

Obrázek 2.2: Ukázka svíčkového grafu z programu NinjaTrader

2.4 Supporty a resistance

Čerpáno z [**?**].

Finanční trhy jsou velmi volatilním a nepředvídatelným prostředím. V grafem se však často vyskytují útvary, jejichž další chování lze s rozumnou pravděpodobností předpovídat. Jedná se o tzv. *supporty* a *resistance*[2].

support je cena, pod kterou nikdo nechce prodávat.

resistance je cena, nad kterou nikdo nechce nakupovat

Z výše uvedeného vyplývá, že se jedná o body, které vzdorují dalšímu růstu, resp. pádu ceny (viz obrázek 2.3). Tyto bariéry jsou tím pevnější, čím déle cena na jejich hranici stagnuje.

Pokud je obchodník schopen rozlišit tyto útvary, může na nich s velkou pravděpodobností nemálo profitovat.

2.5 Trhy

Trhy jsou místem, kde se s aktivy uskutečňuje samotné obchodování. Mohou být komoditní nebo finanční, v závislosti na aktivu, na každém trhu se obchoduje právě jedno aktivum.

[2] čerpáno z *http://www.financnik.cz/komodity/manual/komodity-support-resistance.html*

Obrázek 2.3: Ukázka supportu a resistance (převzato z *http://www.financnik.cz/komodity/manual/komodity-support-resistance.html*)

Trh lze charakterizovat několika hledisky:

volatilita udává, jak rychlý a živý trh je [3], tzn. jak rychlé a velké jsou změny ceny v průběhu dne. Na více volatilních trzích lze rychle a hodně vydělat stejně tak jako ztratit.

cena každý trh má ve své specifikaci pevně definovanou cenu jednoho bodu. Bod se skládá z určitého počtu menších jednotek, *ticků* (taktéž definováno ve specifikaci). Tick je tedy nejmenši cena, o kterou se může trh pohnout. Z výše uvedeného vyplývá, že u dvou různých trhů může stejný bodový pohyb znamenat nestejný finanční zisk či ztrátu.

Z výše uvedeného vyplývá, že ke každému trhu je dobré přistupovat s jinou strategií a s různě velkým kapitálem.

2.6 Technické indikátory

Jedná se o pomocné hodnoty vypočítané z cenových grafů. V průběhu let se ustálila jistá množina těchto indikátorů, které se v praxi osvědčily a které se v různých kombinacích používají k predikci vývoje trhu.

[3] čerpáno z *http://www.financnik.cz/wiki/volatilita*

2.6.1 Klouzavý průměr (MA)

Někdy též označován jako jednoduchý klouzavý průměr - SMA (simple moving average)[4]. Používá se pro vyhlazení průběhu ceny v čase a tedy k redukci šumu. Parametrem je počet historických hodnot, ze kterých se klouzavý průměr počítá. Je definován následovně:

$$SMA_i(n) = \frac{P_{i-n} + P_{i-(n-1)} + ... + P_i}{n} \tag{2.2}$$

kde n je počet historických hodnot a P_i je hodnota close v čase i.[2]

2.6.2 Exponenciální klouzavý průměr (EMA)

Exponenciální klouzavý průměr je velmi podobný jednoduchému klouzavému průměru s tím rozdílem, že starší hodnoty mají pro výpočet menší váhu[5]. Je definován rekurzivně:

$$EMA_i(n) = \frac{2}{n+1}(P_i + EMA_{i-1}) + EMA_{i-1} \tag{2.3}$$

kde n, P_i má stejný význam jako u SMA a EMA_{i-1} je předchozí hodnota exp. průměru. EMA_0 může být zvolena 0.

2.6.3 Index relativní síly (RSI)

Z angl. relative strength index, tento index měří dynamiku vývoje ceny[6]. Pohybuje se v rozmezí 0 -100, trh je považován za překoupený, pokud je index víc než 70 a přeprodaný pokud je menší než 30. Je definován následovně:

$$AG_i(n) = \frac{AG_{i-1}(n)(n-1) + currentGain}{n} \tag{2.4}$$

$$AL_i(n) = \frac{AL_{i-1}(n)(n-1) + currentLoss}{n} \tag{2.5}$$

$$AG_0(n) = \frac{\sum_{i=0}^{n} currentGain}{n} \tag{2.6}$$

$$AL_0(n) = \frac{\sum_{i=0}^{n} currentLoss}{n} \tag{2.7}$$

$$RSI_i(n) = 100 - \frac{100}{1 + \frac{AG_i(n)}{AL_i(n)}} \tag{2.8}$$

kde *currentGain*, resp. *currentLoss*, představují aktuální zisk, resp. ztrátu, vypočítanou jako rozdíl aktuálního close a předchozího close.

[4] čerpáno z *http://stockcharts.com/school/doku.php? id=chart_school:technical_indicators:moving_averages*
[5] čerpáno z *http://stockcharts.com/school/doku.php? id=chart_school:technical_indicators:moving_averages*
[6] čerpáno z *http://stockcharts.com/school/doku.php? id=chart_school:technical_indicators:relative_strength_in*

2.6.4 Commodity Channel Indicator (CCI)

Tento indikátor měří odchylku ceny od své průměrné hodnoty[7]. Vysoká hodnota ukazuje, že cena je výše než je průměr, nízká hodnota signalizuje opak. Jeho rozsah podstatě není omezen, většinou se však pohybuje v rozmezí -100 až 100. Je definován následovně:

$$MD(n) = \frac{|TP_{i-n} - SMA_i^{TP}(n)| + |TP_{i-(n-1)} - SMA_i^{TP}(n)| + ... + |TP_i - SMA_i^{TP}(n)|}{n} \tag{2.9}$$

$$CCI_i(n) = \frac{TP_i - SMA_i^{TP}(n)}{0.015 \cdot MD(n)} \tag{2.10}$$

kde $MD(n)$ je střední odchylka pro n historických hodnot, TP_i je typická hodnota v čase i a $SMA_i^{TP}(n)$ je jednoduchý klouzavý průměr, ve kterém se používá TP místo close (viz 2.2).

2.7 Obchodování

Pro úspěšné obchodování, tedy obchodování, které přináší zisk, je nutné vyřešit 2 zásadní otázky - kdy vstupovat do pozice a kdy z pozice vystupovat. Pro obě tyto otázky existuje řada rozličných metod, od velmi jednoduchých po velice sofistikované. Možná ještě důležitějších tématem, než jsou ta výše uvedená, je správa kapitálu (angl. *money management*).

2.7.1 Vstup do pozice

Do trhu lze vstupovat na základě cenových formací, technických indikátorů, dobrého pocitu, či kombinace výše uvedeného. Jeden z nejjednodušších způsobů je využití klouzavého průměru. Jakmile cena protne klouzavý průměr, jedná se o signál vstupu do pozice. Pokud průměr protne shora dolů, jedná se o signál pro vstup do pozice short, pokud zdola nahoru, je to signál pro vstup do pozice long (viz obrázek 2.4).

Obecně lze říci, že vstup do pozice je značně netriviální rozhodnutí a různé obchodní systému k němu používají různé prostředky.

Výstup je dokonce ještě důležitějším rozhodnutím, než vstup do pozice, neboť přímo ovlivňuje velikost zisku či ztráty obchodu. Stejně jako v případe vstupu do pozice, ani zde neexistuje žádné obecné pravidlo a každý systém řeší výstup z pozice po svém.

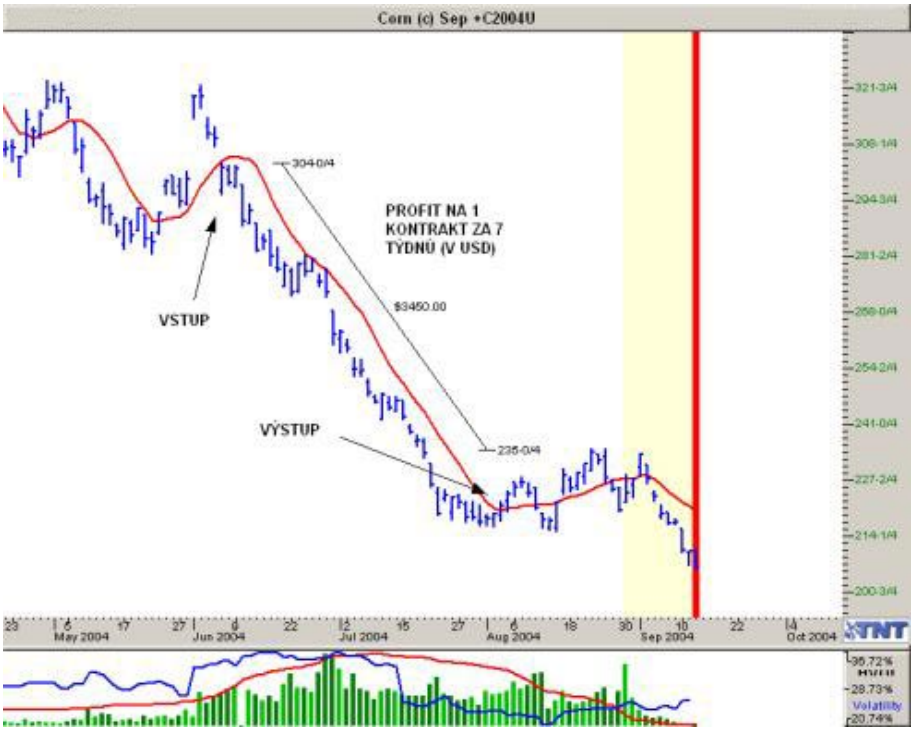
Často používanou technikou je tzv. *stop-loss*, dále jen *SL*. SL ilustruje obrázek 2.5. Jedná se o vědomé akceptování jisté, předem definované ztráty, jako platby za to, že se ztráta nestane nekontrolovatelnou. Funguje tak, že při zadávání příkazu nastavíme hranici, pod kterou cena nesmí spadnout. Pokud pod ni spadne, pozice je automaticky uzavřena a obchod tedy ukončen se předem definovanou ztrátou.

SL jako takový chrání před nekontrolovatelnou ztrátou, používá se však i v technikách pro generování zisku v případě, že se trh pohybuje správným směrem.

Profit-target

Dále jen *PT*, termín profit-target označuje techniku, kdy je dopředu stanovena hodnota zisku, na které je obchod uzavřen. PT se používá v kombinaci se SL. Výstup pomocí PT je

[7] čerpáno z *http://stockcharts.com/school/doku.php? id=chart_school:technical_indicators:commodity_channel_in*



Obrázek 2.4: Ukázka vstupu do pozice short při pozičním obchodování futures kontraktů kukuřice (převzato z *http://www.financnik.cz/komodity/manual/komodity-klouzave-prumery.html*)

prakticky použitelný v případě, že obchodník již delší dobu obchoduje a statisticky analyzuje své obchody a je tedy schopen odhadnout, na jaké hodnotě zisku nejčastěji uzavíral obchody.

Posouvání SL

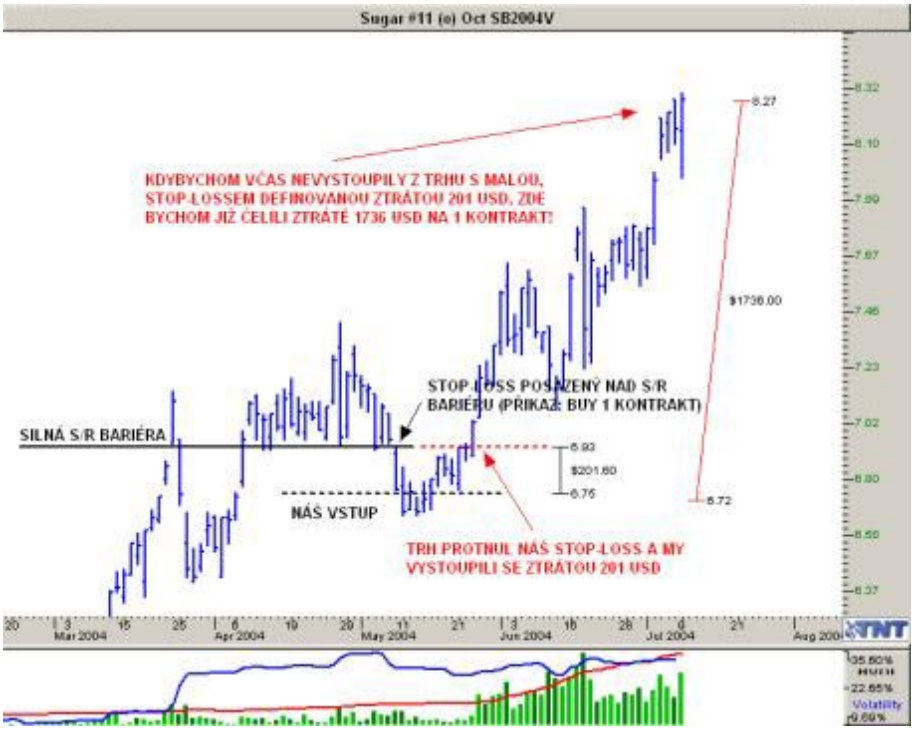
V případě, že se trh pohybuje správným směrem, lze výstup řídit postupným posouváním SL a tím chránit již dosažený zisk. Pravidla pro tyto posuny mohou být velmi složitá nebo velmi jednoduchá, záleží na obchodním systému. Opět neexistuje obecná poučka nebo pravidlo.

2.8 Money management

Často podceňovaná součást obchodování, která však fakticky odděluje stabilně vydělávající úspěšné obchodníky od těch, kteří znovu a znovu přichází o peníze[8].

Úspěšné obchodování není založené na schopnosti určit s úspěšností 90 % vývoj trhu. Běžně se dosahuje přesnosti kolem 50 %, je však nutné vybírat obchody, které přinášejí potenciálně vyšší zisk než ztrátu. Money management je tedy schopnost vybírat a plánovat obchody a strategie s vyšším potenciálem zisku, než je výše předem stanoveného risku. Pro měření poměru risku a zisku používá se *RRR (risk reward ratio)* (poměr mezi riskem a očekávaným ziskem), které se obchodníci snaží minimalizovat, tj. snaží se při předem definovaném riziku dosáhnout co největšího potenciálního zisku.

[8] čerpáno z *http://www.financnik.cz/komodity/manual/money-management.html*



Obrázek 2.5: Ukázka fungování SL. Obchod je uzavřen ještě předtím, než může ztráta znatelně narůst (převzato z *http://www.financnik.cz/komodity/manual/komodity-stop-loss.html*)

Risk lze dopředu definovat pomocí stop-lossů, očekávaný zisk se většinou počítá jako průměrný zisk za historii obchodování.

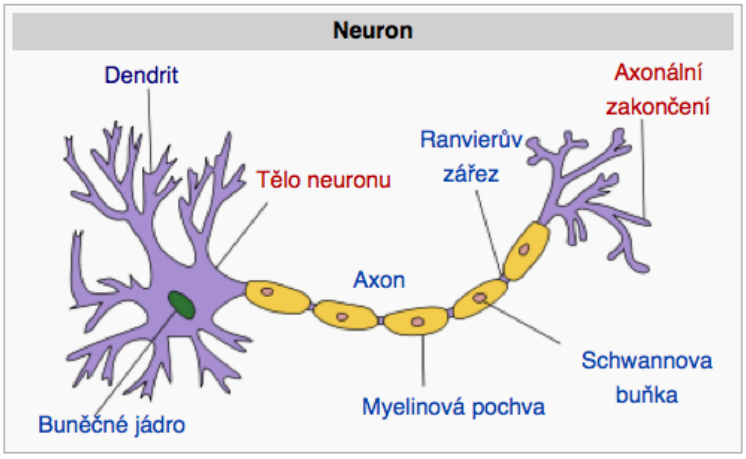
Kapitola 3

Teoretická část

V této části budou představeny základy fungování přirozených i umělých neuronových sítí i neuronů.

3.1 Biologický neuron

Neuron je základním prvkem nervové tkáně. Jedná se o vysoce specializovanou buňku schopnost přijímat, zpracovávat a rozesílat elektrické impulsy. Jeho strukturu popisuje obrázek 3.1. Skládá se z těla neuronu, krátkých výběžků (*dendritů*) a dlouhého výběžku (*axonu*).



Obrázek 3.1: Struktura biologického neuronu (převzato z *http://cs.wikipedia.org/wiki/Neuron*)

Neurony spolu komunikují na základě *synapsí* - propojů mezi axonem jednoho neuronu a dendrity jiných neuronů. Pokud napětí na dendritech překoná určitou limitní hodnotu, dojde v vyslání elektrického impulsu do axonu a tedy do ostatních neuronů, které jsou přes své dendrity připojeny k axonu.

Z výše uvedeného vyplývá, že dendrity jsou „vstupem“ neuronu, kdežto axon je „výstupem“.

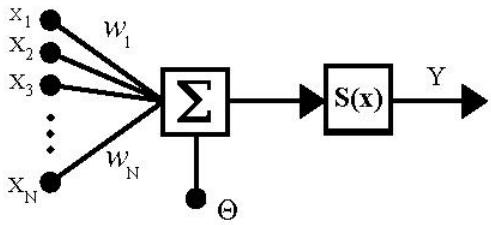
Spojením několika neuronů vzniká neuronová síť.

3.2 Umělý neuron

Matematický model, který napodobuje chování biologického neuronu[1]. Existuje jich celá řada, jeden z nejpoužívanějších je model popsaný McCullochem a Pittsem (viz obrázek 3.2):

$$Y = S(\sum_{i=1}^{N}(w_i x_i) + \Theta) \tag{3.1}$$

kde Y je výstup neuronu, x_i jsou vstupy neuronu, w_i jsou synaptické váhy, Θ je prahová hodnota a $S(x)$ je aktivační funkce.



Obrázek 3.2: Model umělého neuronu (převzato z *http://cs.wikipedia.org/wiki/Neuronov%C3%A1_s%C3%*

Synaptické váhy určují informaci uloženou v neuronu. Čím je hodnota váhy vyšší, tím je daný vstup důležitější.

Neuron tedy funguje tak, že na jeho vstup jsou vloženy hodnoty. Hodnoty jsou vynásobeny s váhou dané synapse a výsledné hodnoty jsou sečteny. Pokud je součet menší než prahová hodnota Θ, zůstává neuron v pasivním stavu. Pokud je práh překročen, výstup neuronu je vypočítán pomocí aktivační funkce.

Aktivační funkce může být v podstatě libovolná spojitá funkce, nejčastěji se však používají 4 následující:

skoková přenosová funkce pro vstup menší než daná mez vrací nulu, pro větší vrací jedničku.

$$f(x) = \begin{cases} 0 & x < \Theta \\ 1 & x \geq \Theta \end{cases} \tag{3.2}$$

sigmoidální přenosová funkce hodnoty se v minus nekonečnu blíží nule a v nekonečnu jedničce, v nule nabývá hodnoty 0,5. Její výhodou oproti skokové je existence derivace v každém bodě definičního oboru.

$$f(x) = \frac{1}{1 + e^{-kx}} \tag{3.3}$$

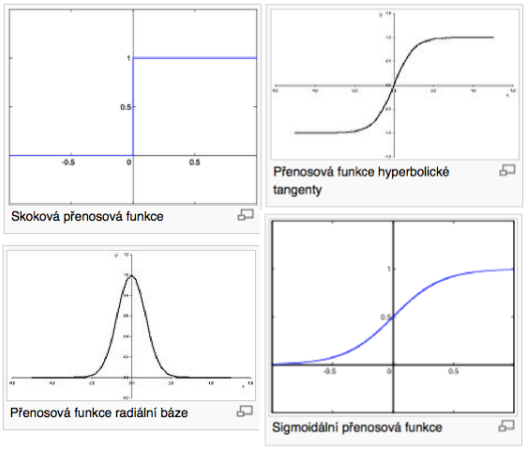
přenosová funkce hyperbolické tangenty v minus nekonečnu se blíží -1 a jedničce v nekonečnu, v nule nabývá hodnoty 0.

$$f(x) = \frac{2}{1 + e^{-kx}} - 1 \tag{3.4}$$

[1] čerpáno z *http://cs.wikipedia.org/wiki/Neuronov%C3%A1_s%C3%AD%C5%A5*

přenosová funkce radiální báze hodnoty se v minus i plus nekonečnu blíží 0, pro nulu dosahuje hodnoty 1.

$$f(x) = e^{-kx^2} \tag{3.5}$$



Obrázek 3.3: Přehled aktivačních funkcí (převzato z *http://cs.wikipedia.org/wiki/Neuronov%C3%A1_s%C3%AD%C5%A5*)

3.3 Neuronová síť

Spojením několika neuronů vzniká neuronová síť. Podle způsobu zapojení neuronů lze rozlišit několik topologií sítí, z nichž nejčastěji se používá dopředná vrstevná síť (feedforward neural network). Obecně se skládá ze 3 typů vrstev - vstupní vrstva, jedné nebo více skrytých vrstev, výstupní vrstva (viz obrázek 3.4). Vrstvy jsou plně propojené, tj. všechny neurony vstupní vrstvy jsou propojeny se všemi neurony skryté vrstvy a všechny neurony skryté vrstvy jsou propojeny se všemi neurony výstupní vrstvy. Na vstupní vrstvu jsou přivedeny vstupy. Z výstupní vrstvy vycházejí výstupy.

3.4 Učení neuronové sítě

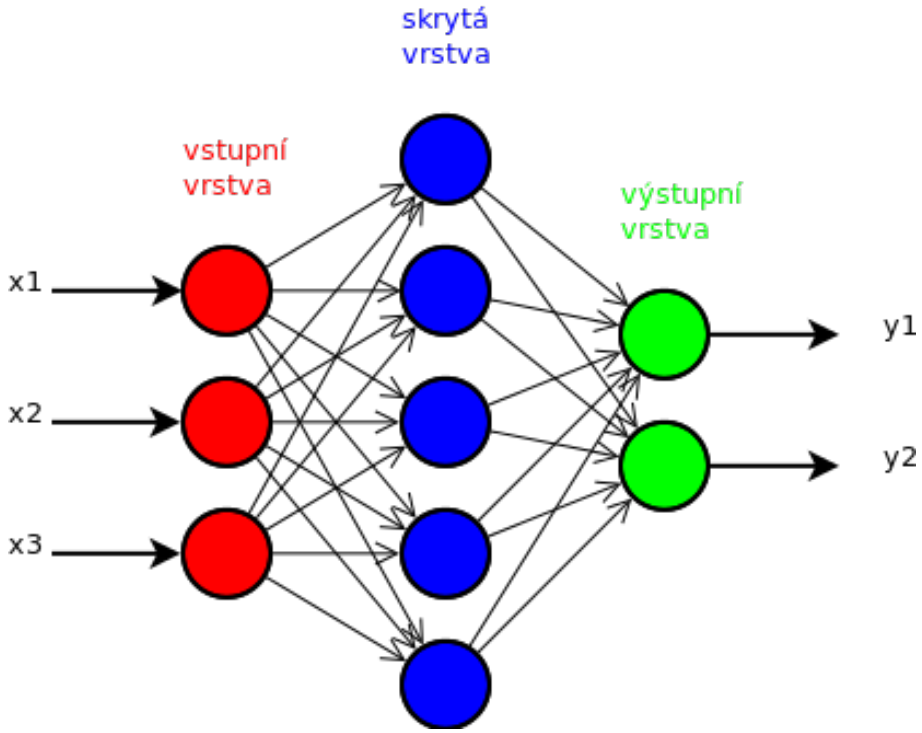
Aby neuronová síť k něčemu byla, je nutné ji naučit na datech. Obecně existují 2 druhy učení - učení bez učitele a učení s učitelem.

3.4.1 Učení bez učitele

Způsob učení, při kterém není známý správný výstup neuronové sítě, tj. hledá skrytou strukturu v datech. Používá se např. pro shlukování, konkrétně algoritmy *self-orginizing maps (SOM)* a *adaptive resonance theory (ART)*.

3.4.2 Učení s učitelem

Učení, při kterém je znám správný výstup sítě. V této práci bude využit tento druh učení, bude tedy podrobněji probráno.



Obrázek 3.4: Schéma dopředné neuronové sítě

Backpropagation

Jeden ze základních algoritmů pro učení umělých neuronových sítí[2]. Je nutná datová sada, skládající se z množiny vstupů a očekávaných výstupů. Tento algoritmus také požaduje, aby aktivační funkce neuronů měla derivaci na celém svém definičním oboru.

Skládá se ze 2 fází - propagace a úprava vah.

Propagace Skládá se ze 2 kroků.

1. Dopředná propagace trénovacího vstupu neuronovou sítí a generování výstupů.
2. Zpětná propagace výstupů sítí a počítání rozdílů mezi hodnotami výstupů a skrytých neuronů ($\vec{\Delta}$).

Úprava vah

1. Pronásobení $\vec{\Delta}$ s hodnotami vstupních neuronů, tj. výpočet gradientu vah.
2. Odečtení násobku gradientu od vah.

Velikost násobku gradientu se nazývá učící konstanta. Její velikost ovlivňuje rychlost učení sítě - čím vyšší je konstanta, tím rychleji se síť učí.

Fáze propagace a úpravy vah se opakují, dokud síť nedosáhne potřebné přesnosti.

[2] čerpáno z *http://en.wikipedia.org/wiki/Backpropagation*

Algoritmus

```
inicializace vah sítě (nastavení na náhodné hodnoty)
do
    foreach trénovací vzorek v
        výstup = síť.spočítejVýstup(v)
        ideální = v.získejIdeálníHodnotu()
        spočítej chybu (výstup - ideální) výstupních neuronů
        spočítej rozdíl vah pro všechny váhy ze skryté vrstvy do výstupní vrstvy
        spočítej rozdíl vah pro všechny váhy ze vstupní vrstvy do skryté vrstvy
        uprav váhy
until síť nemá požadovanou přesnost
```

Resilient backpropagation

Tato metoda je vylepšením základního backpropagation algoritmu a je prezentována v [1].

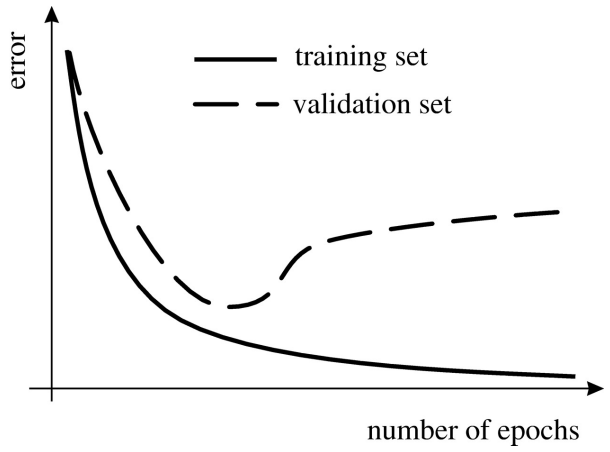
Narozdíl od původní metody, která k úpravě vah používala hodnotu parciální derivace, resilient backpropagation používá pouze znaménko derivace, které pak dále řídí danou iteraci učení.

3.4.3 Cross validace

Z angl. *cross-validation*. Metoda, která zvyšuje přesnost neuronové sítě (ale obecně libovolného modelu) pro neznámá (tj. ještě neviděná) data.

Množina dat, na které se síť učí, se rozdělí na určitý počet stejně velkých podmnožin. Tento počet je parametrem křížové validace. Každá podmnožina je poté postupně použita pro validaci (je na ní vyhodnocena přesnost sítě) a ostatní množiny jsou použity jako testovací data. Výsledné hodnoty jsou zprůměrovány.

Výše uvedený postup odstraňuje problém přeučení, tedy situaci, kdy síť dosahuje dobrých výsledků na trénovacích datech, na skutečných datech si však vede špatně (viz obrázek 3.5).



Obrázek 3.5: Problém přeučení (převzato z *http://rstb.royalsocietypublishing.org/-content/362/1479/339.full*)

3.5 Genetický algoritmus

Genetický algoritmus je heuristickou metodou prohledávání stavového prostoru založenou na napodobování evoluce. Pracuje s genetickými pojmy - genotyp, fenotyp, populace, chromozom atd. Používá se v případech, kdy je analytické řešení zcela neznámé nebo příliš výpočetně náročné a spadá do širší kategorie tzv. evolučních algoritmů.

3.5.1 Důležité pojmy

gen základní stavební jednotka chromozomu. Jeho hodnota spadá do definované abecedy (např. 0 a 1, celá čísla atd.)

genotyp skládá ze z jednotlivých genů, občas bývá nazýván chromozom nebo jedinec

fenotyp kandidátní řešení vytvořené na základě určitého genotypu

populace množina genotypů

fitness funkce definuje, nakolik je daný fenotyp kvalitní (odpovídá požadavkům zadání)

3.5.2 Stručný popis

Množina kandidátních řešení (nazývaná populace) je na počátku algoritmu inicializována náhodnými hodnotami (genotypy). Tyto genotypy představují (vetšinou číselnou formou) vlastnosti toho kterého kandidátního řešení a postupně se mění. Tyto změny představují evoluci jedinců.

Algoritmus pracuje tak, že na začátku je populace vytvořena z náhodných jedinců. Celá populace je ohodnocena tzv. fitness funkcí, která popisuje, jak moc kvalitní je dané řešení. Dále jsou z počáteční generace určitým způsobem (budou představeny dále) vybráni rodiče, kteří určitým způsobem vytvoří potomky (taktéž bude rozvedeno dále). Z potomků a současné generace je vytvořena generace nová. Je tedy vidět, že se jedná o iterativní proces, který může mít různá kritéria pro ukončení.

3.5.3 Pseudokód GA

```
inicializuj počáteční populaci
ohodnoť populaci

dokud není splněna ukončovací podmínka
    vyber vhodné jedince z populace
    pomocí genetických operátorů (mutace, křížení...) je modifikuj
    z původní populace a těchto nových potomků vytvoř novou populaci
    ohodnoť populaci

výstupem GA je řešení s nejlepší hodnotou fitness funkce
```

3.5.4 Metody výběru rodičů

Rodiče lze z populace na základě fitness funkce vybrat mnoha různými způsoby, které mohou potenciálně významně ovlivnit výsledek GA. Pro všechny by však mělo platit, že

jedinec s vyšší fitness funkcí bude jako rodič vybrán pravděpodobněji než jedinec s nižší fitness funkcí. Je též důležité zmínit, že jedinec může být vybrán vícekrát.

Představme některé běžně používané.

Ruleta

Každému jedinci je vypočítaná relativní fitness hodnota (definovaná jako podíl fitness jedince a součet fitness hodnot celé populace), která odpovídá pravděpodobnosti, se kterou bude jedinec vybrán. Tato metoda má zásadní nevýhodu, pokud se v populaci vyskytuje jedinec s výrazně vyšší fitness než zbytek populace. Tento jedinec je pak velice často vybírán jako rodič, takže populace v další generaci „zdegeneruje“ - ztratí se genetická rozmanitost.

Označení ruleta vychází z podobnosti se skutečnou ruletou - kružnice je rozdělena na výseky, jejichž velikost odpovídá relativní fitness hodnotě jedince. Při vybírání se ruletou zatočí a je vybrán příslušný jedinec.

Výběr podle pořadí

Jedinci se seřadí podle hodnoty fitness a vybere se potřebný počet nejlepších.

Turnaj

Z populace se náhodně vybere určitý počet jedinců (vetšinou 2), ze kterých je pak nejlepší vybrán jako rodič. Poté se náhodný výběr opakuje.

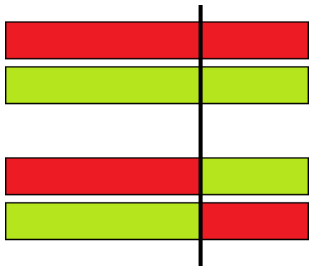
3.5.5 Genetické operátory

Křížení

Křížení je operátor, který z určitého počtu rodičů (vetšinou 2) generuje určitý počet potomků (taktéž většinou 2). Tento operátor se používá velmi často (s cca 80 % pravděpodobností). Konkrétní realizace tohoto operátoru závisí na abecedě genů. Pro binární geny se často používá jedno- či vícebodové křížení a uniformní křížení.

Jednobodové křížení

Potomci vznikají tak, že se náhodně vygeneruje bod křížení. První potomek má první část z prvního rodiče, druhou z druhého, druhý potomek přesně obráceně (viz obrázek 3.6).



Obrázek 3.6: Jednobodové křížení (převzato z *http://blog.dreasgrech.com/2011/02/stringevolver-evolving-strings-with.html*)

Vícebodové křížení

Bodů křížení se generuje víc, tvorba nových potomků probíhá analogicky jako u jednobodového křížení (viz obrázek 3.7).

Obrázek 3.7: Dvoubodové křížení (převzato z *http://blog.dreasgrech.com/2011/02/stringevolver-evolving-strings-with.html*)

Uniformní křížení

Hodnota každého genu potomka je vybrána z prvního rodiče se stejnou pravděpodobností jako ze druhého.

Mutace

Tento operátor se používá s nízkou pravděpodobností (kolem 10 %). Jeho realizace opět závisí na abecedě genů.

Mutace binárního genotypu

Binární genotyp se mutuje velmi jednoduše - provede se inverze náhodného genu genotypu.

Mutace reálného genotypu

Ke každému genu se přičte číslo, které je náhodně generováno podle Gaussova rozdělení pravděpodobnosti se středem v 0 a s odchylkou 0,5.

3.5.6 Metody tvorby nové populace

Existují 2 základní přístupy ke tvorbě nové populace - buď se do nové populace přenáší část staré populace, která se doplní nově vytvořenými potomky, nebo se celá nová populace tvoří pouze z potomků (tzv. generační populace).

3.5.7 Ukončovací podmínky

Při rozhodování, kdy algoritmus ukončit, se používá několik přístupů.

- definice požadované maximální/maximální (v závislosti na tom, zda se fitness funkce minimalizuje či maximalizuje) hodnoty fitness funkce
- ukončení po vytvoření pevně stanoveného počtu generací
- ukončení po uplynutí určité doby, příp. spotřebovaní určitého strojového času

- ukončení poté, co nové generace nezlepšují kvalitu řešení
- kombinace výše uvedeného

Kapitola 4

Praktická část

V této kapitole bude představena stěžejní část práce. Na začátku budou prezentována trénovací a testovací data a popsáno jejich zpracování. Poté budou popsány použité obchodní modely, u složitějších bude podrobně popsána jejich implementace. Nakonec budou předloženy a diskutovány výsledky jednotlivých modelů.

4.1 Trénovací a testovací data

Jak již bylo řečeno, práce se zabývá intradenním obchodováním. Je tedy vhodné vybrat takové trhy, které mají přes den dostatečný pohyb, aby bylo možné realizovat zisk (musí tedy být dostatečně volatilní). Zároveň však nesmí běžný denní pohyb činit tisíce USD (nesmí tedy být příliš volatilní) - aby na nich mohl obchodovat i začínající investor.

Trhy též musí být co nejméně ovlivněny fundamentálními faktory, což činí např. z akciových trhů zcela nevhodné kandidáty, neboť ceny akcií mohou být zásadně ovlivněny např. změnou vedení společnosti. Tyto vlivy se do modelu velmi špatně zahrnují.

Dalším z důležitých hledisek je likvidita trhu. Pokud není trh dostatečně likvidní (tj. neprobíhá dostatečné množství obchodů), je možné, že aktuální příkaz nebude okamžitě proveden, protože se nenajde partner transakce. To znamená pozdržení příkazu, což bude velmi pravděpodobně znamenat změnu ceny a tedy zásadní nekonzistenci mezi modelem a reálným trhem. Historická data sice neumožňují poznat, zda by byl příkaz okamžitě vykonán, v dostatečně likvidním trhu to však lze předpokládat.

Posledním hlediskem je dostupnost dostatečného množství historických dat. Většina takovýchto intradenních dat je k dispozici na internetu za finanční poplatek, tento faktor je tedy velmi limitujícím.

Podařilo se získat tickova data z období od půlky prosince 2010 do září 2013 pro futures index E-mini Russell 2000, který je mimojiné pro začínající obchodníky doporučován v [3].

4.1.1 E-mini Russell 2000

Jak již bylo zmíněno, jedná se o futures index, jehož hodnota je počítána z hodnot akcií 2000 firem z různých odvětví obchodovaných na burze. Je extrémně likvidní a splňuje i druhou podmínku, tj. malou závislost na fundamentálních faktorech (právě kvůli faktu, že je počítán z velkého množství hodnot). Splňuje i podmínku rozumné volatility.

Velikost kontraktu je 100 USD a jeden bod je složen z 10 ticků.

4.1.2 Zpracování dat

Data, která jsou k dispozici, jsou ticková. To přináší určité výhody a nevýhody. Výhodou je, že je možné vygenerovat cenové svíčky libovolného timeframu. Nevýhodou je, že je nutné tyto data agregovat do cenových svíček, neboť v tickové podobě je jich příliš velké množství. Zpracování probíhalo tak, že se shlukly ticky po daných časových intervalech a z každého shluku se vytvořila svíčka - první tick shluku značil její open, poslední close, největší high a nejmenší low.

Pro další práci byl zvolen timeframe 2 minuty, který je doporučován pro intradenní obchodování v [3].

4.2 Obchodní modely

V této části budou představeny rozebrány jednotlivé obchodní modely, které byly testovány. Jedná postupně o náhodný obchodní model, poté naivní lidský obchodní model založený na klouzavých průměrech a nakonec 2 modely s neuronovou sítí - první s náhodně zvolenými parametry, druhý s parametry zjištěnými genetickým algoritmem.

4.2.1 Metodika vyhodnocení obchodních modelů

Každý obchodní model bude vyhodnocen na druhé polovině datové sady, která je k dispozici (tj. 338 obchodních dní, které budou nadále označovány jako testovací data). První polovina bude využita modely, které používají neuronovou síť, na učení této sítě (tato data budou dále označována jako trénovací data).

Každý model bude vyhodnocen prvně bez stop-lossu, poté s použitím stop-lossu.

Počet ziskových a ztrátových obchodních dní je uveden bez započítání brokerských poplatků, stejně tak jako celkový zisk a celková ztráta. Tyto údaje tedy ukazují surovou úspěšnost systému. Konečný stav účtu i průměrná bilance z jednoho obchodu jsou uvedeny bez i s poplatky, které byly stanoveny na 4 USD na obchod.

4.2.2 Náhodný obchodní model

Tento model vychází z předpokladu, že pokud bude vstup do pozic a výstup z pozic náhodný, bude při dostatečném množství obchodu celková bilance (samozřejmě bez započítá brokerských poplatků) víceméně nulová. Je v implementován v podstatě jako test, zda je implementace nižších vrstev aplikace správná.

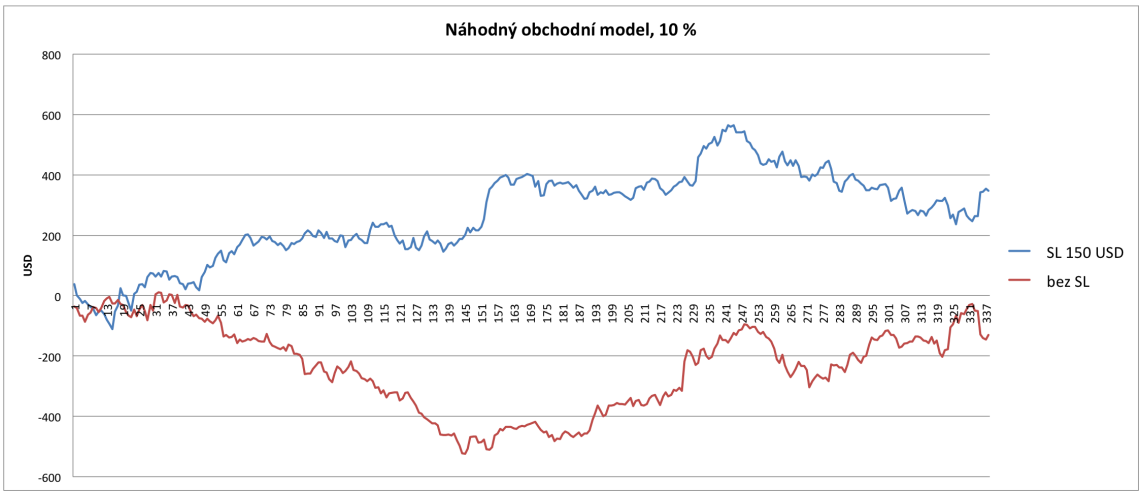
Funguje tak, že pro každou svíčku s určitou pravděpodobností vstoupí/vystoupí z pozice. Pokud vstupuje, je to s pravděpodobností 50 % na dlouhou stranu a se stejnou pravděpodobností na krátkou stranu.

Model byl testován bez stop-lossu a poté se stop-lossem 150 USD (empiricky určeným) a s pravděpodobností vstupu/výstupu do/z pozice 10 % (zvolena víceméně náhodně, ale tak, aby průměrný denní počet obchodů nepřesáhl 15) a 3,2 % (určena tak, aby průměrný denní počet obchodů byl 3-4, jak je doporučeno v [3]), což dává 4 kombinace nastavení modelu.

Protože výsledky po jednom běhu simulace by byly ovlivněny šumem, bylo provedeno 1000 simulačních běhů a předkládané výsledky jsou jejich průměrem.

Tabulka 4.1: Výsledky náhodného obchodního modelu pro pravděpodobnost 10 %

Pravděpodobnost 10 %	bez SL	SL 250 USD
Počet obchodních dní celkem	338	338
z toho ziskových	169	169
z toho ztrátových	169	169
Průměrná bilance z jednoho obchodu		
bez započítání poplatků	-0,00003 USD	0,00009 USD
se započítáním poplatků	-4,03 USD	-3,9 USD
Celkový zisk	6874 USD	6876 USD
Celková ztráta	6887 USD	6842 USD
Průměrný denní počet obchodů	10,7	10,7
Konečný stav účtu		
bez započítání poplatků	-131 USD	347 USD
se započítáním poplatků	-14662 USD	-14184 USD



Obrázek 4.1: Vývoj stavu účtu po jednotlivých dnech v USD pro náhodný obchodní model s pravděpodobností 10 % bez započítaných poplatků

Výsledky náhodného obchodního modelu

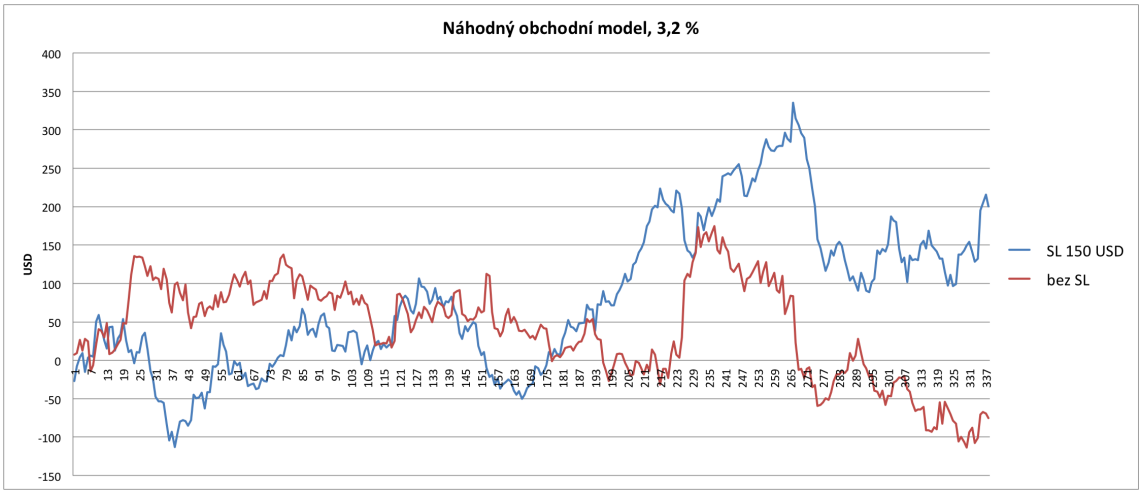
Z výsledků náhodného obchodního modelu s pravděpodobností 10 % (viz graf 4.1 a tabulka 4.1) lze soudit, že pokud nezapočítáváme brokerské poplatky, osciluje stav účtu okolo 0 (hodnoty bez započítání poplatků). Tomu odpovídá i průměrná bilance na obchod, která se též blíží 0 (bez poplatků) nebo hodnotě poplatku za obchod (pokud jsou započítány poplatky). Poměr ztrátových a ziskových dní je taktéž vyrovnaný. Stop-loss, podle očekávání, zlepšuje úspěšnost modelu.

Výsledky modelu s pravděpodobností 3,2 % (viz graf 4.2 a tabulka 4.2) jsou velmi podobné výsledkům 10% modelu s tím rozdílem, že poměr ztrátových a ziskových dnů není 1:1. Toto lze přisoudit faktu, že proběhlo menší množství obchodů, jsou tedy více ovlivněny šumem. Stop-loss opět znatelně zlepšil úspěšnost.

Celkové výsledky náhodného obchodního modelu odpovídají předpokladům.

Tabulka 4.2: Výsledky náhodného obchodního modelu pro pravděpodobnost 3,2 %

Pravděpodobnost 3,2 %	bez SL	SL 250 USD
Počet obchodních dní celkem	338	338
z toho ziskových	166	166
z toho ztrátových	172	172
Průměrná bilance z jednoho obchodu		
bez započítání poplatků	-0,00006 USD	0,0001 USD
se započítáním poplatků	-4,06 USD	-3,8 USD
Celkový zisk	5967 USD	5967 USD
Celková ztráta	5975 USD	5946 USD
Průměrný denní počet obchodů	3,2	3,2
Konečný stav účtu		
bez započítání poplatků	-75 USD	200 USD
se započítáním poplatků	-4495 USD	-4221 USD



Obrázek 4.2: Vývoj stavu účtu po jednotlivých dnech v USD pro náhodný obchodní model s pravděpodobností 3,2 % bez započítaných poplatků

4.2.3 Naivní obchodní model

V praxi používané obchodní modely bývají velmi sofistikované a obtížně implementovatelné, neboť do nich obchodník dává svoje dlouholeté zkušenosti, cit pro trh a intuici. Existuje však velmi jednoduchý a naivní model[1], který by měl být lepší než náhodný model.

Tato strategie je založena na klouzavých průměrech. Pro připomenutí uvedeme, že klouzavý průměr je parametrizovaný technický indikátor, kde parametr je počet posledních close hodnot, ze kterých se počítá aritmetický průměr. Obchodní interpretace je tedy ta, že vyhlazuje cenovou křivku a snižuje její zašuměnost.

Pokud, na cenovém grafu, protne cena klouzavý průměr směrem dolů, je to signál ke vstupu do krátké pozice (formálně viz výraz 4.1) , pokud směrem nahoru, jedná se o signál ke vstupu do dlouhé pozice (viz výraz 4.2).

[1] čerpáno z *http://www.financnik.cz/komodity/manual/komodity-klouzave-prumery.html*

Výraz, který definuje signál pro vstup do krátké pozice:

$$Close(i-1) > SMA(i-1) \wedge Close(i) < SMA(i) \tag{4.1}$$

Výraz, který definuje signál pro vstup do dlouhé pozice:

$$Close(i-1) < SMA(i-1) \wedge Close(i) > SMA(i) \tag{4.2}$$

Obrázek 4.3: Klouzavý průměr protíná cenový graf (převzato z *http://www.financnik.cz/komodity/manual/komodity-klouzave-prumery.html*)

Výstup z pozice je řízen stejným způsobem, tzn. pokud jsem v dlouhé pozici a model dá signál do krátké pozice, ukončuji svoji dlouhou pozici. Byla zvolena vcelku agresivní obchodní strategie, kdy se v tomto případě nejenže ukončí dlouhá pozice, ale zároveň se vstupuje do krátké pozice.

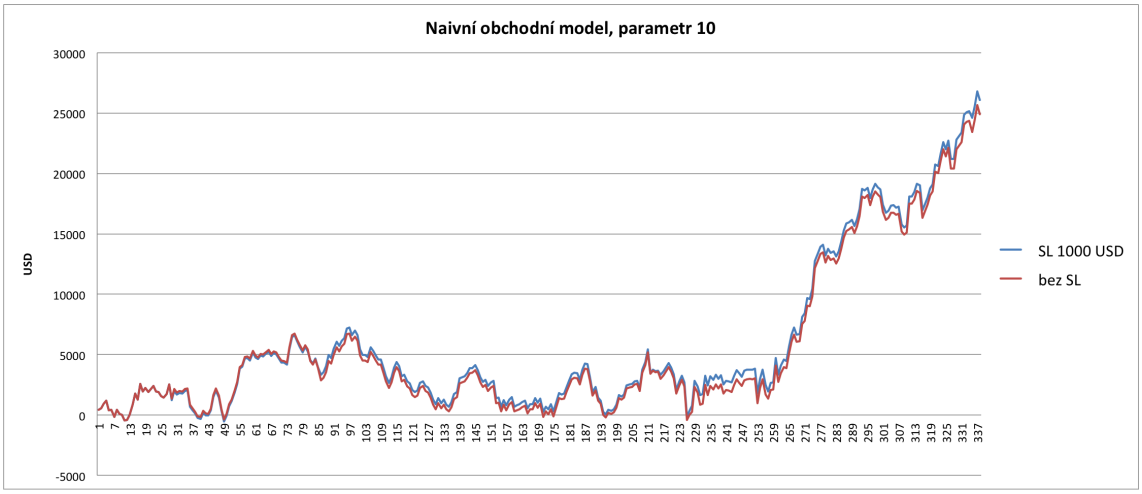
Další způsob, jak lze vystoupit z pozice, je případný stop-loss nebo pokud je daná svíčka poslední svíčkou dne (nechceme držet pozice přes noc, neboť se jedná o intradenní obchodování).

Tento model je parametrizován jedním parametrem, a to parametrem klouzavého průměru, tj. z kolika posledních hodnot se počítá. Budou představeny výsledky pro hodnoty 10, 40 a 30, která byla experimentálně vypozorována jako optimální. Ve spojení s variantou se stop-lossem a bez stop-lossu vzniká 6 kombinací nastavení modelu.

Tabulka 4.3: Výsledky naivního obchodního modelu pro parametr 10

Parametr 10	bez SL	SL 1000 USD
Počet obchodních dní celkem	338	338
z toho ziskových	194	194
z toho ztrátových	144	144
Průměrná bilance z jednoho obchodu		
bez započítání poplatků	2,38 USD	2,49 USD
se započítáním poplatků	-1,6 USD	-1,5 USD
Celkový zisk	10084 USD	10102 USD
Celková ztráta	7591 USD	7495 USD
Poměr zisku a ztrát	1,33	1,35
Průměrný denní počet obchodů	31	31
Konečný stav účtu		
bez započítání poplatků	24929 USD	26079 USD
se započítáním poplatků	-16938 USD	-15788 USD

Výsledky naivního obchodního modelu



Obrázek 4.4: Vývoj stavu účtu po jednotlivých dnech v USD pro naivní obchodní model s parametrem 10 bez započítaných poplatků

Z výsledků pro parametr 10 je vidět, že pro některé nastavení modelů není rozdíl mezi variantou se stop-lossem a bez stop-lossu velký.

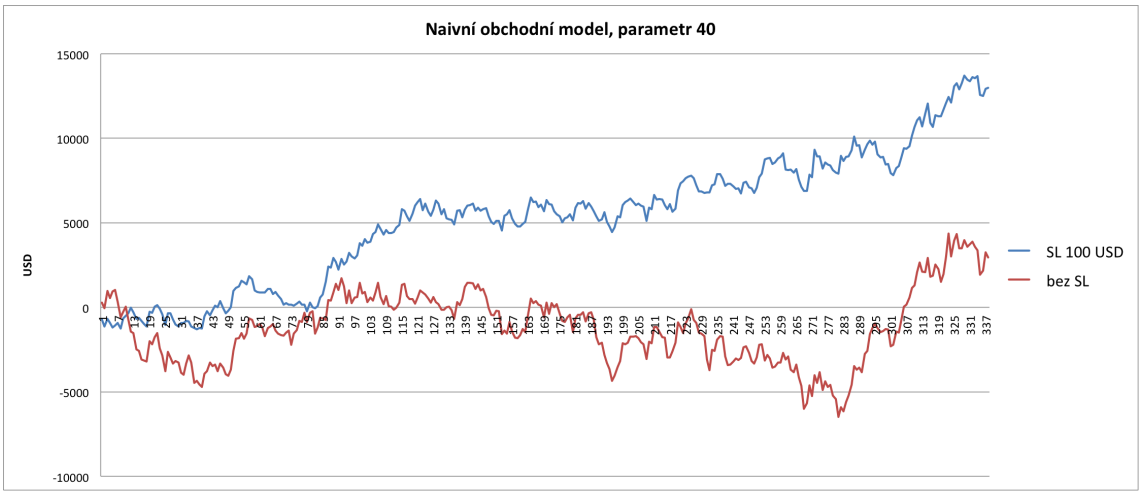
Při hodnotě stop-lossu menší než 100 USD byl výkon modelu podstatně horší než bez stop-lossu. To lze vysvětlit tím, že klouzavý průměr s takto nízkým parametrem je pořád dostatečně volatilní na to, aby dával velké množství obchodních signálů, model je tedy často v otevřené pozici. Takto nízký stop-loss způsobí, že i při malém pohybu na „špatnou“ stranu je pozice ukončena se ztrátou, přestože by za pár dalších svíček mohla přinést zisk.

Ukázalo se, že až hodnota stop-lossu 1000 USD přináší zlepšení výkonnosti modelu, ne však velké (viz tabulka 4.3). To lze vysvětlit tak, že denní pohyb 1000 USD je pro tento trh velmi neobvyklý (navíc je neobvyklé i to, že model sám o sobě nedal výstupní signál), stop-loss se tedy aplikuje velmi zřídka (viz graf 4.4).

Tabulka 4.4: Výsledky naivního obchodního modelu pro parametr 40

Parametr 40	bez SL	SL 100 USD
Počet obchodních dní celkem	338	338
z toho ziskových	164	177
z toho ztrátových	174	161
Průměrná bilance z jednoho obchodu		
bez započítání poplatků	0,69 USD	3 USD
se započítáním poplatků	-3,3 USD	-0,95 USD
Celkový zisk	7477 USD	5694 USD
Celková ztráta	7183 USD	4397 USD
Poměr zisku a ztrát	1,04	1,29
Průměrný denní počet obchodů	12,6	12,6
Konečný stav účtu		
bez započítání poplatků	2949 USD	12979 USD
se započítáním poplatků	-14098 USD	-4072 USD

Je též vidět, že vhodně zvolená hodnota stop-lossu zlepšuje poměr zisků a ztrát, neovlivňuje však průměrný denní počet obchodů.

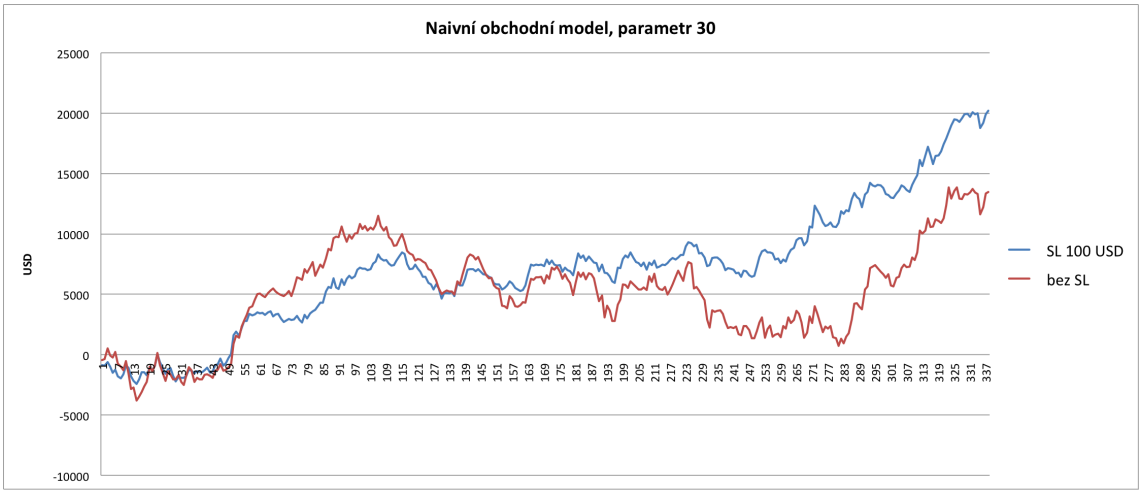


Obrázek 4.5: Vývoj stavu účtu po jednotlivých dnech v USD pro naivní obchodní model s parametrem 40 bez započítaných poplatků

Model s parametrem 40 již naplno ukazuje potenciál stop-lossu. Z grafu vývoje stavu účtu je vidět, že stop-loss zavedl vzestupnou tendenci a také značně vylepšil průměrnou bilanci na obchod a poměr zisku a ztrát. Stejně jako v minulém případě nezměnil průměrný denní počet obchodů.

Tabulka 4.5: Výsledky naivního obchodního modelu pro parametr 30

Parametr 30	bez SL	SL 100 USD
Počet obchodních dní celkem	338	338
z toho ziskových	166	174
z toho ztrátových	172	164
Průměrná bilance z jednoho obchodu		
bez započítání poplatků	2,5 USD	3,8 USD
se započítáním poplatků	-1,4 USD	-0,1 USD
Celkový zisk	8881 USD	6638 USD
Celková ztráta	7536 USD	4616 USD
Poměr zisku a ztrát	1,17	1,43
Průměrný denní počet obchodů	15,3	15,3
Konečný stav účtu		
bez započítání poplatků	13459 USD	20219 USD
se započítáním poplatků	-7300 USD	-544 USD



Obrázek 4.6: Vývoj stavu účtu po jednotlivých dnech v USD pro naivní obchodní model s parametrem 30 bez započítaných poplatků

Model s parametrem 30 se nejeví bez stop-lossu jako kvalitní nastavení. Po jeho zavedení se však úspěšnost modelu rapidně zvyšuje, což je vidět hlavně ze vzrůstu poměru zisku a ztrát o 22 %. I v ostatních kritériích toto nastavení systému dominuje - má nejlepší průměrnou bilanci na obchod jak se započítáním poplatků, tak bez nich i nejvyšší konečný stav účtu po započítání poplatků.

Vhodná hodnota stop-lossu tedy opět zásadně zlepšila kvalitu modelu a opět nezměnila průměrný počet obchodů za den.

Vyhodnocení naivního obchodního modelu

Z výsledků je zřetelně vidět, že, v souladu s předpoklady, naivní obchodní model je kvalitnější ve smyslu průměrné bilance na obchod a poměru zisků a ztrát. V některých nastavením dává náhodný model lepší konečný stav účtu se započítáním poplatků, což je umožněno faktem, že dělá méně obchodů za den, poplatky jsou tedy nižší. Důležitějším výsledkem je v

tomto případě konečný stav účtu bez poplatků, ve kterém naivní model jasně vede.

4.2.4 Základní obchodní model využívající neuronovou síť

Tento model bude dále také nazýván modelem se základní (neuronovou) sítí.

V této části bude popsán základní model založený na neuronové síti, bude představena jeho základní myšlenka, způsob předzpracování dat a poté výsledky simulovaného obchodování.

Myšlenka

Základní myšlenka je prostá - naučit neuronovou síť predikovat budoucí hodnotu trhu nebo alespoň tendenci trhu. Jakmile se toto podaří, lze velmi efektivně obchodovat.

Vstupy sítě budou výhradně historické ceny a technické indikátory, fundamentální faktory by bylo těžké zakódovat a navíc nemají na intradenní obchod ani zdaleka tak velký vliv jako na poziční obchodování.

Vhodnou kombinací historických cen a technických indikátorů by se neuronová síť měla naučit predikovat budoucí cenu. Činí tak na základě již viděných cenových formací (z trénovací sady), se kterými hledá podobnost v aktuálním cenovém grafu.

Předzpracování dat

Hodnoty cenových dat se pohybují v řádech desítek, stovek a tisíců bodů. Tento rozsah je však pro učení neuronové sítě nevhodný. Při pokusu o učení sítě v bodovém rozsahu 730 - 800 se chyba sítě s postupnými iteracemi zlepšovala jen velice pomalu, tento způsob byl tedy zavržen. Vstupní data jsou normalizována do rozsahu -1 až 1, což nejenže zlepšuje a v podstatě umožňuje učení sítě, ale navíc má i zobecňující efekt - v reálném trhu se stejné cenové útvary objevují na všech cenových hladinách, tj. i když je trh např. na úrovní 200 bodů i 300 bodů. Normalizace v rámci dní umožňuje síti učit se cenové útvary nezávisle na tom, na jaké cenové hladině se útvar objevil.

Dalším důvodem, proč byla zvolena normalizace v rámci dnů a ne v rámci celé datové sady, je fakt, že datová sada zahrnuje více než rok a za tento rok se trh mohl pohnout o několik stovek bodů. Pokud by tedy -1 v normalizované datové sadě odpovídalo minimu a 1 by odpovídalo maximu, jejichž vzdálenost by mohla být několik set bodů, pak by denní pohyby v rámci desetin a jednotek bodů staly velice nesignifikantními a síť by nebyla schopná naučit se intradenní nuance pohybů specifické pro daný trh.

Je použita modifikovaná min-max normalizace. Přímo tuto metodu nelze použít, neboť normalizuje do intervalu 0 až 1 (viz vzorec 4.3).

$$MinMaxNorm(x) = \frac{x - min}{max - min} \tag{4.3}$$

kde x je normalizovaná hodnota a max, resp. min je maximum, resp. minimum intervalu, který se normalizuje.

Pro normalizaci do intervalu -1 až 1 je nutné použít následující vzorec (jeho odvození je triviální):

$$ModifiedMinMaxNorm(x) = MinMaxNorm(x) - 1 \tag{4.4}$$

Vzhledem k tomu, že na vstup neuronové sítě přijdou i technické indikátory odvozené z cen, je zřejmé, že bude nutné normalizovat i indikátory. Klouzavé průměry (jednoduchý i

exponenciální) není třeba nijak upravovat, neboť jsou počítány jako průměr cen, a pokud jsou ty normalizovány, jsou automaticky normalizovány i vypočtené průměry.

Jiná situace nastává pro CCI a RSI indikátory. Oba jsou počítány jako poměry, normalizace cen je tedy neovlivní.

CCI indikátor je nastaven tak, aby cca 80 % hodnot padlo do intervalu -100 až 100. Je tedy normalizován pomocí modifikovaného min-max vzorce tak, že jako minimum intervalu je bráno -150 a jako maximum 150. Do tohoto intervalu padne naprostá většina hodnot a těch pár extrémních se promítne mimo interval -1 až 1. Toto však lze prohlásit za dostatečně dobrou aproximaci.

RSI indikátor se ze své definice pohybuje mezi 0 a 100, tyto hodnoty jsou tedy voleny jako parametry pro modifikovanou min-max normalizaci.

Je zřejmé, že při využití v reálném provozu nebude možné přijímaná cenová data korektně normalizovat, neboť nebude znám rozsah celého obchodního dne. Toto lze řešit mnoha způsoby, např.:

- převzetí minima a maxima předchozího dne
- převzetí minima a maxima a jejich lineární transformace
- sofistikovanější výpočet minima a maxima na základě průběhu minulého obchodního dne (na základě statistiky, neuronové sítě atd.)

Struktura neuronové sítě

Bude použita třívrstvá dopředná neuronová síť, počet vstupních neuronů bude roven délce vstupního vektoru, počet neuronů skryté vrstvy bude postupně nastaven na experimentální hodnoty 15, 20 a 25 neuronů, ve výstupní vrstvě bude jediný výstupní neuron, který bude predikovat budoucí hodnotu trhu.

Horizontem bude nazýván časový bod (svíčka), pro který se dělá aktuální predikce. Hodnoty starší budou označována jako historická data (případně poslední hodnoty), hodnoty mladší budou budoucnost.

Vstupní vektor bude složen ze 4 složek:

TP 20 posledních typických hodnot, normalizovaných v intervalu obchodního dne

CCI 15 posledních hodnot indikátoru s parametrem 20, normalizovaných výše uvedených způsobem

RSI 15 posledních hodnot indikátoru s parametrem 14, normalizovaných výše popsaným způsobem

EMA 20 posledních hodnot indikátoru s parametrem 12, hodnoty jsou již normalizovány, neboť jsou počítány z normalizovaných cen

Výstupní vektor pro vstupní vektor vytvořen z trénovacích dat jako průměr normalizovaných typických hodnot ze 7 nejbližších budoucích svíček. Pro úspěšné obchodování není totiž nutné znát přesnou hodnotu budoucí svíčky, stačí úspěšně předpovědět směr, kterým se bude trh ubírat. Navíc volba většího časového okna (v tomto případě 7) částečně omezuje šum, kterým se reálná data vyznačují.

Tyto parametry byly experimentálně nalezeny jako vyhovující, evoluce lepších pomocí genetického algoritmu je předmětem další podkapitoly.

Učení neuronové sítě

Je využit algoritmus resilient backpropagation. Učení probíhá dávkově, tzn. váhy jednotlivých synapsí nejsou měněny po každé iteraci, ale až po 1000 iteracích. Toto rozhodnutí bylo provedeno na základě faktu, že při tzv. online učení (váhy jsou upravovány po každé iteraci) může chyba sítě po dokončení iterace dokonce vzrůst[2].

Pro počáteční inicializaci sítě je použit Nguyen-Widrow algoritmus[3] - na počátku jsou váhy a prahy inicializovány náhodnými hodnotami. Tyto hodnoty jsou poté upraveny pomocí koeficientu vypočteném z počtu neuronů skrytých vrstev a počtu výstupních neuronů. Nguyen-Widrow algoritmus umožňuje nastavit takové počáteční rozložení vah, že následné učení sítě je rychlejší než při použití zcela náhodného rozložení.

Síť je učena na párech vstupní vektor/výstupní vektor (viz výše). Burzovní data mají timeframe 2 minuty, tj. každá svíčka je agregací dat ze 2 minut. Toto časové měřítko je vhodné z několika důvodů - je doporučované v [3], odstraňuje velké množství šumu, zmenšuje objem zpracovávaných dat a přitom zachovává rozumné množství testovacích vektorů.

Množina trénovacích vektorů je vytvořena následovně - z první poloviny dostupných dat jsou vytvořeny vektory. Poté je každý desátý (tj. celkem 10 %) vložen do množiny validačních vektorů, ze zbytku je složena trénovací množina. Pořadí vektorů v ní je poté náhodně promícháno z toho důvodu, aby učení sítě bylo co nejplynulejší. V případě, kdy by se jedna část trhu chovala výrazně jinak než druhá (např. jedna část silně rostoucí trh, druhá část silně klesající trh), by po započetí učení na druhé části trhu chyba výrazně zhoršila. Promíchání pořadí tento jev do velké míry eliminuje.

Učení sítě probíhá tak dlouho, dokud se signifikantně zlepšuje přesnost sítě na validačních datech. Je tedy provedena iterace učení na trénovacích datech a poté je spočítána chyba na validačních datech. Dokud síť zlepšuje o více než 0.00001 na iteraci, učení pokračuje. Jakmile toto přestane platit, učení končí.

Tento způsob má za úkol zajistit, aby učení sítě netrvalo příliš dlouho (v této fázi by ještě nebyl problém, pokud by učení sítě trvalo desítky minut, v další fázi je však využit genetický algoritmus, který potřebuje natrénovat velké množství sítí a tam by byl takto dlouhý čas učení nezvládnutelný) a také předejít overfittingu, tj. přeučení. Jedná se o situaci, kdy se přesnost sítě na trénovacích datech stále zlepšuje, přesnost na testovacích datech však klesá. Lze to interpretovat tak, že síť se příliš přizpůsobila dané trénovací sadě a ztrácí schopnost generalizovat.

Přesnost sítě je měřena střední kvadratickou chybou (dále jen MSE - z angl. mean squared error), která se počítá následujícím způsobem:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(predicted_i - real_i)^2 \tag{4.5}$$

kde n je počet vektorů, $predicted_i$ je hodnota, kterou predikuje výstupní neuron pro i-tý vektor a $real_i$ je ideální hodnota.

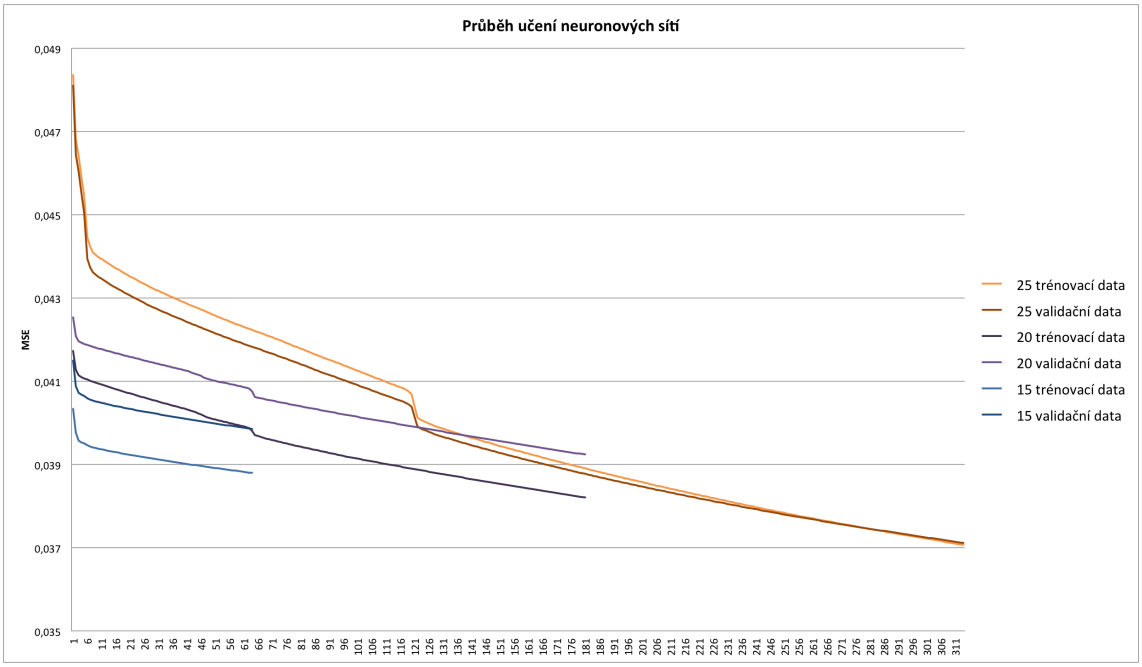
Cílem učení je tedy minimalizace této chyby a v důsledky toho co nejpřesnější predikce.

Výsledky učení

Na obrázku 4.7 je graf průběhu střední kvadratické chyby v závislosti na proběhlých trénovacích epochách. Jsou zobrazena data pro trénovací množinu i validační množinu pro různé

[2] http://www.webpages.ttu.edu/dleverin/neural_network/neural_networks.html
[3] http://www.heatonresearch.com/encog/articles/nguyen-widrow-neural-network-weight.html

počty neuronů skryté vrstvy.



Obrázek 4.7: Průběh chyby v závislosti na proběhlých epochách. Číslo označuje počet neuronů skryté vrstvy.

Z grafu je zřetelně vidět, že síť s menším počtem neuronů ve skryté vrstvě se učí rychleji, což je očekávaný výsledek. Je též vidět, že chyba pro trénovací a validační množiny pro jednotlivé sítě velmi silně koreluje.

Také je vidět, že průběh chyby je pro všechny sítě a množiny velmi podobný - z počáteční chyby velmi rychle během několika málo epoch výrazně klesá, poté se pokles zpomaluje čím dál více.

Tabulka 4.6: Konečné chyby (měřené MSE) pro jednotlivé sítě a množiny

	15 neuronů	20 neuronů	25 neuronů
Trénovací množina	0,038795035	0,038207722	0,037067822
Validační množina	0,039851685	0,039238443	0,037113335

Tabulka 4.6 přehledně shrnuje výsledné hodnoty pro jednotlivé počty neuronů a množiny. Je vidět, že pro tento způsob trénování sítě nemá počet neuronů skryté vrstvy velký vliv na chybu sítě, má však velký vliv na rychlost trénování (viz tabulka 4.7).

Tabulka 4.7: Počet epoch potřebných k natrénování sítě

	15 neuronů	20 neuronů	25 neuronů
Počet epoch	64	181	314

Obchodní model

Obchodování probíhá s využitím natrénovaných sítí. Z druhé poloviny dostupných dat jsou vytvořeny testovací vektory (analogickým způsobem jako trénovací vektory), kterou jsou postupně přikládány na vstup sítě. Výstup sítě (tj. predikovaná budoucí hodnota) je porovnána s aktuální hodnotou trhu. Pokud se liší o větší než prahovou hodnotu, model generuje doporučení pro vstup do pozice (zda do dlouhé či krátké závisí na znaménku rozdílu mezi aktuální hodnotou a predikovanou hodnotou).

V případě, že se model vyskytuje v opačné pozici, než která vyplývá z výstupu sítě (např. model má otevřenu dlouhou pozici a od sítě vzejde doporučení pro vstup do krátké pozice), je příslušná pozice ukončena a model vstupuje do pozice doporučené sítí (pro náš případ by model ukončil dlouhou pozici a vstoupil do krátké).

Tento algoritmus popisuje následující pseudokód:

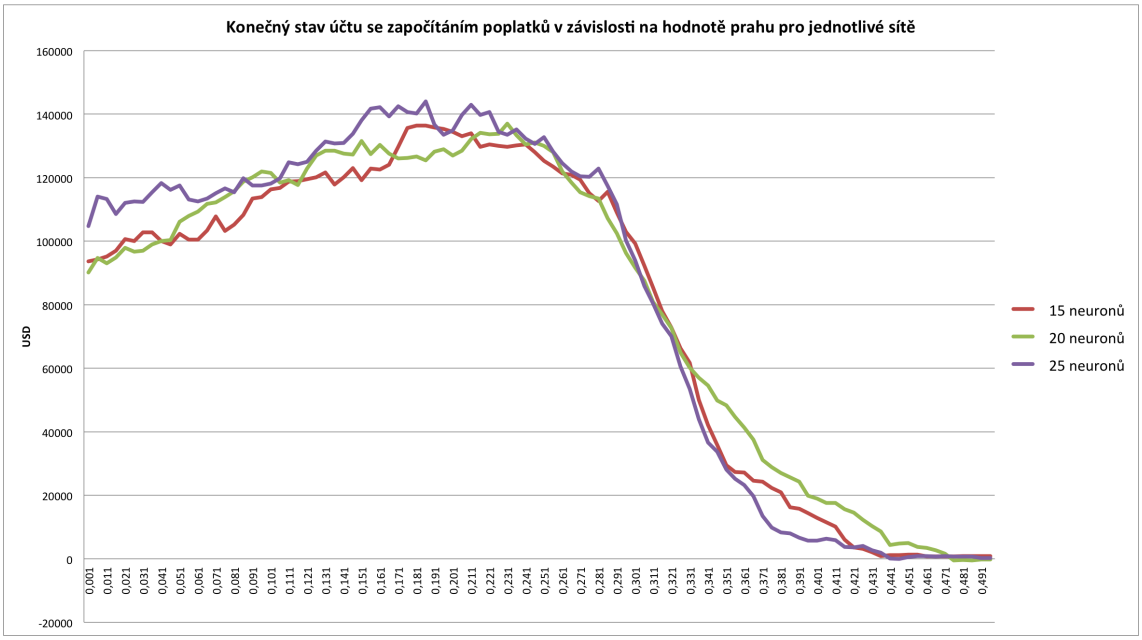
```
aktualniCena = vektor.ziskejAktualniCenu()
predikovanaCena = sit.spocitejVystup(vektor)

trhPoroste = ((predikovanaCena - aktualniCena) > (prah))
trhKlesne = ((aktualniCena - predikovanaCena) > (prah))

if (trhPoroste) {
    if (vKratkePozici()) {
        ukonciPozici()
    }
    vstupDoDlouhe()
}
else if (trhKlesne) {
    if (vDlouhePozici()) {
        ukonciPozici()
    }
    vstupDoKratke()
}
```

Hodnota prahu může zcela zásadně měnit výsledky obchodování, proto není možné ji volit náhodně. Pro optimální hodnotu je zkoumán interval 0,001 až 0,5 s krokem 0,005. Pro každou zkoumanou hodnotu prahu jsou obchodnímu algoritmu postupně předkládána testovací data a je vypočten celkový zisk včetně započítání poplatků za obchody. Pokud by tyto poplatky nebyly započítány, byly by zvýhodněny ty velikosti prahu, při kterých model dělá velké množství obchodů a průměrná bilance obchodu je nízká. Tyto hodnoty prahu by mohly dávat zajímavé výsledky pro celkový stav účtu bez započítaných poplatků, byl by to však odklon od reálného obchodování, ke kterému se tato práce pokouší přiblížit.

Výsledky pro všechny tři sítě ukazuje graf 4.8.

Z grafu je vidět, že všechny sítě dosahují maxima v podobném intervalu a jejich křivky obecně velmi silně korelují. Vypočítaná maxima jsou přehledně shrnuta v tabulce 4.8.

Pro optimalizované hodnoty jsou v dalším kroku pro každou síť vypočítány podrobnosti - stejně jako pro předchozí modely (tedy náhodný a naivní). Jsou uvedeny pouze hodnoty pro strategii bez stop-lossů, neboť strategie se stop-lossy byla sice vyzkoušena, nicméně



Obrázek 4.8: Konečný stav účtu se započítáním poplatků v závislosti na hodnotě prahu pro sítě s 15, 20, 25 neurony ve skryté vrstvě.

Tabulka 4.8: Vypočítaná obchodní maxima pro jednotlivé sítě

	15 neuronů	20 neuronů	25 neuronů
Maximální zůstatek	136940 USD	136832 USD	144026 USD
Hodnota prahu	0,181	0,231	0,186

žádná hodnota stop-lossu nezlepšila výsledky obchodování pro žádnou síť [4]. Hodnoty pro strategii se stop-lossy tedy nejsou uvedeny.

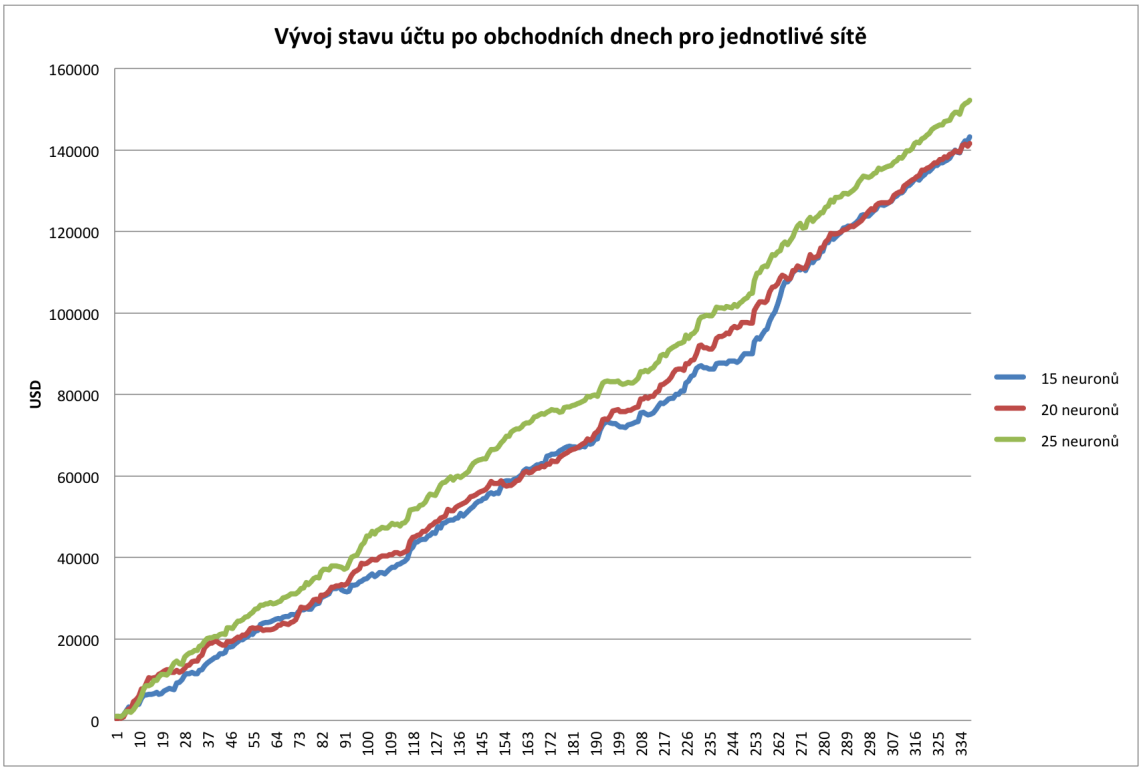
Údaje shrnuje graf 4.9, který ukazuje vývoj stavu účtu po jednotlivých obchodních dnech bez započítaných poplatků, a tabulka 4.9, která ukazuje shrnující informace pro jednotlivé sítě.

Z grafu je zřetelně vidět, že stav účtu plynule a stabilně roste pro všechny sítě a že křivky velmi silně korelují.

Z tabulky, která poskytuje podrobnější údaje, je vidět, že síť se 20 neurony ve skryté vrstvě dosahuje nejvyššího počtu nulových dní, tj. dní, ve kterých neproběhl žádný obchod. Vysvětlení je nasnadě - pro tuto síť byl jako optimální vypočítán nejvyšší (konkrétně 0,231) ze všech tří prahů a čím vyšší práh, tím méně obchodních signálů (což mj. potvrzuje i vypočítaný průměrný denní počet obchodů). Nižší počet obchodů je ovšem kompenzován lepší průměrnou bilancí na obchod.

Celkový zisk a celková ztráta jsou pro sítě s 15 a 20 neurony ve skryté vrstvě velice podobné, lehce vybočuje síť s 25 neurony. Totéž platí pro konečný stav účtu bez započítání poplatků i se započítáním. Nejpravděpodobnějším vysvětlením se jeví být fakt, že síť s 25 neurony byla natrénována na větší přesnost než ostatní dvě sítě (viz tabulka 4.6). Vzhledem k tomu, že hlavní výsledek (tj. konečný stav účtu se započítáním poplatků) je pro sítě s 15 a

[4] V reálném obchodování by bylo ovšem rozumné stop-loss využít i tak a zvolit takovou hodnotu, která negativně neovlivní obchodování. Může nastat výpadek proudu, případně internetového připojení a té chvíli je lepší mít stop-loss nastaven a vystavit se tak pouze omezené ztrátě.



Obrázek 4.9: Vývoj stavu účtu bez započítaných poplatků po jednotlivých dnech v USD pro obchodní model využívající neuronové sítě

Tabulka 4.9: Výsledky obchodního modelu s neuronovými sítěmi

	15 neuronů	20 neuronů	25 neuronů
Počet obchodních dní celkem	338	338	338
z toho ziskových	249	254	261
z toho ztrátových	71	54	64
z toho nulových	18	30	13
Průměrná bilance z jednoho obchodu			
bez započítání poplatků	84,99 USD	119,7 USD	75 USD
se započítáním poplatků	81,3 USD	115,7 USD	71 USD
Celkový zisk	158470 USD	156970 USD	169520 USD
Celková ztráta	15340 USD	15410 USD	17390 USD
Poměr zisku a ztrát	10,33	10,18	9,74
Průměrný denní počet obchodů	4,98	3,49	5,99
Konečný stav účtu			
bez započítání poplatků	143130 USD	141560 USD	152130 USD
se započítáním poplatků	136940 USD	136832 USD	144026 USD

20 neurony, který byly natrénovány na téměř stejnou přesnost, v podstatě totožný, je možné usuzovat, že počet neuronů skryté vrstvy nemá na výsledek příliš velký vliv. Mnohem větší vliv bude mít přesnost natrénované sítě.

Z toho důvodu budou v následující modifikaci tohoto modelu využity sítě s 15 neurony ve skryté vrstvě.

4.2.5 Modifikace modelu s ANN využívající genetický algoritmus

Tento model bude dále také nazýván jako model s optimalizovanou sítí.

Jak již bylo řečeno, je při využití neuronové sítě cílem naučit síť na trénovacích datech tak, aby její chyba na testovacích byla co nejmenší. Tohoto se snažíme dosáhnout kombinací historických dat a z nich odvozených indikátorů. Problém je ten, že stavový prostor všech kombinací délek historických dat a případně délek identifikátorů je příliš rozsáhlý na procházení hrubou silou.

Pro nalezení optimálních parametrů se nabízí využití genetického algoritmu. Jak již bylo řečeno, genetický algoritmus je heuristická metoda inspirovaná přírodní evolucí, která vyžaduje, aby byly definovány následující položky:

- způsob zakódování problému (resp. kandidátního řešení) do genomu
- vytvoření počáteční populace
- fitness funkce, která hodnotí kvalitu kandidátních řešení
- ukončovací podmínka

Tyto položky budou nyní podrobně představeny.

Zakódování kandidátního řešení do genomu

V našem případě je kandidátní řešení kombinace délky vstupního okna a množství historických hodnot jednotlivých indikátorů, které jsou zároveň nepovinné.

Toto lze nejjednodušeji zakódovat do pole celých čísel o 4 prvcích (vstupní okno + 3 indikátory), kde hodnota 0 značí nepřítomnost indikátoru (pro vstupní okno to není povolená hodnota). Strukturu genomu shrnuje tabulka 4.10.

Tabulka 4.10: Struktura genomu

okno	CCI	RSI	EMA

Povolené hodnoty pro jednotlivá pole jsou následující:

okno libovolně přirozené číslo (tj. $\mathbb{Z}^+ - \{0\}$)

CCI, RSI, EMA libovolné nezáporné celé číslo (tj. $\mathbb{Z}^+$)

Proces převodu z genomu na parametry je velice jednoduchý - pole okno se použije jako délka vstupního okna, další pole, pokud jsou nenulová, se použijí jako počet historických hodnot daného indikátoru. Opačný převod se provede analogicky.

Vytvoření počáteční populace

Počáteční populace není generována zcela náhodně, neboť existují principiální omezení hodnot genomu (např. nedává smysl velikost okna v záporných číslech) i optimalizační omezení (např. pro velikost okna 5000 by se nikdy žádný obchod neprovedl, neboť tolik dat obchodní den ani neobsahuje).

Pro velikost vstupního okna je tedy použito náhodné číslo z intervalu 1 až 100. Každý indikátor je použit s pravděpodobností 50 % a pokud je použit, je pro počet použitých historických hodnot vybráno náhodně číslo z intervalu 1 až 100.

Fitness funkce

Hodnotící funkce má za úkol vyjádřit, jak kvalitní dané kandidátní řešení. Na základě pozorování z předchozího modelu je možné říct, že výsledek (tj. konečný stav účtu se započítanými poplatky) obchodního modelu využívajícího neuronovou síť je nepřímo úměrný chybě sítě na trénovacích datech, tedy čím nižší chyba, tím lepší výsledek.

Z tohoto důvodu není nutné hledat parametry pro maximální hodnotu stavu účtu, stačí nalézt parametry pro minimální chybu sítě.

Tato optimalizace umožní velmi výrazně zkrátit časovou náročnost výpočtu, neboť pro výpočet maximální hodnoty stavu účtu by bylo třeba:

1. natrénovat neuronovou síť
2. pro každou kandidátní hodnotu prahu (viz minulý obchodní model) vypočítat výsledek obchodování (velmi náročná operace)
3. maximální hodnotu vrátit jako hodnotu fitness funkce pro zadané parametry

Pokud se však omezíme na výpočet chyby sítě, změní se postup následovně:

1. natrénovat neuronovou síť
2. její výslednou chybu vrátit jako výsledek fitness funkce

Je vidět, že velmi výpočetně náročný bod 2 odpadl.

Ukončovací podmínka

Genetický algoritmus končí, dokud se ve 3 po sobě jdoucích generacích nezmění hodnota fitness funkce nejlepšího jedince.

Parametry genetického algoritmu

Populace se skládá ze 60 jedinců. 30 % nejlepších jedinců automaticky beze změny přechází do další generace (princip elity), ostatní jsou vybráni turnajovým algoritmem.

Pro tvorbu nových potomků je použito dvoubodové křížení.

Mutace probíhá s pravděpodobností 20 % tak, že ke každému prvku genomu náhodně přičte či odečte náhodné číslo z intervalu 1 až 5.

Výsledky genetického algoritmu

Jako nejlepší byl vyselektován jedinec s genomem [10, 0, 0, 0], tedy velikost vstupního okna 10, bez indikátorů. Příslušná síť byla natrénována na střední kvadratickou chybu 0,018.

Pro síť byl použit stejný postup jako v předchozím modelu, tj. byla opět hledána optimální hodnota prahu na intervalu 0,001 až 0,5 s krokem 0,005. Výsledné hodnoty jsou zobrazeny v grafu 4.10. Nejvyšší hodnoty dosahuje model pro hodnotu prahu 0,061.

Pro optimální hodnotu tedy byla stejně jako v minulém případě vypočítána podrobná dat. Vývoj stavu účtu po jednotlivých dnech zobrazuje graf 4.11 a shrnutí statistických údaje je obsaženo v tabulce 4.11.

V grafech 4.10 a 4.11 lze pozorovat jistou podobnost s grafy 4.8 a 4.9. První dvojice grafů (hodnoty prahů) nejprve pomalu roste, poté dosáhne svého maxima a nakonec klesá. Druhá dvojice grafů (stavy účtu) roste téměř lineárně.

Obrázek 4.10: Vývoj stavu účtu se započítáním poplatků pro jednotlivé hodnoty prahu pro obchodní model využívající GA

Obrázek 4.11: Vývoj stavu účtu bez započítaných poplatků po jednotlivých dnech v USD pro obchodní model využívající GA

Rozdíly se ovšem nachází v tabulce 4.11. Poměr ziskových a ztrátových dní prudce vzrostl, stejně jako poměr celkových zisků a ztrát. Průměrný denní počet obchodů klesl, naproti tomu se průměrná bilance obchodu rapidně zvýšila.

Tabulka 4.11: Výsledky obchodního modelu využívajícího genetický algoritmus

	15 neuronů
Počet obchodních dní celkem	338
z toho ziskových	305
z toho ztrátových	11
z toho nulových	22
Průměrná bilance z jednoho obchodu	
bez započítání poplatků	260,3 USD
se započítáním poplatků	256,3 USD
Celkový zisk	265180 USD
Celková ztráta	2810 USD
Poměr zisku a ztrát	94,3
Průměrný denní počet obchodů	2,98
Konečný stav účtu	
bez započítání poplatků	262370 USD
se započítáním poplatků	258338 USD

Pro stop-loss platí totéž co v minulém případě, tj. nebyl použit z toho důvodu, že nezlepšoval výkon modelu.

4.3 Porovnání obchodních modelů

V této části budou porovnány jednotlivé obchodní modely. Srovnání proběhne pomocí grafů vývoje stavu účtu po jednotlivých obchodních dnech bez započítání poplatků i s jejich započítáním (neboť model, který má lepší výsledky bez započítání poplatků než jiný model, může dávat horší výsledky, pokud se poplatky započítají) a pomocí shrnující přehledové tabulky.

Graf 4.12 ukazuje vývoj stavu účtu bez započítání poplatků po jednotlivých obchodních dnech.

Je vidět, že oba náhodné modely (tj. náhodný model s pravděpodobností 10 % a 3,2 %, pro oba nastaven stejný stop-loss 150 USD) po celou dobu oscilují okolo 0, což odpovídá předpokladům.

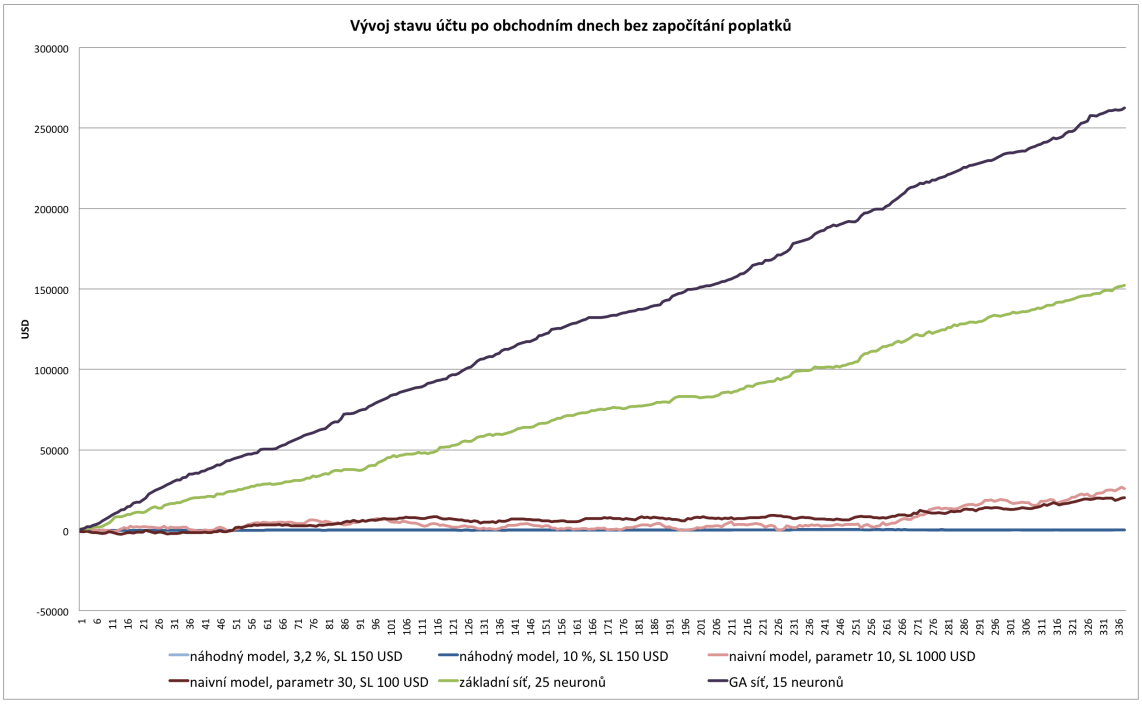
Naivní obchodní model s parametrem 10 je dle očekávání více volatilní než ten s parametrem 30, neboť delší parametr redukuje šum dat a volatilitu. Na konci však model s parametrem 10 překonává druhý model.

Model využívající základní neuronovou síť dosahuje mnohonásobně lepšího výkonu než naivní i náhodné modely, vykazuje též velmi nízkou míru volatility, neboť roste téměř lineárně.

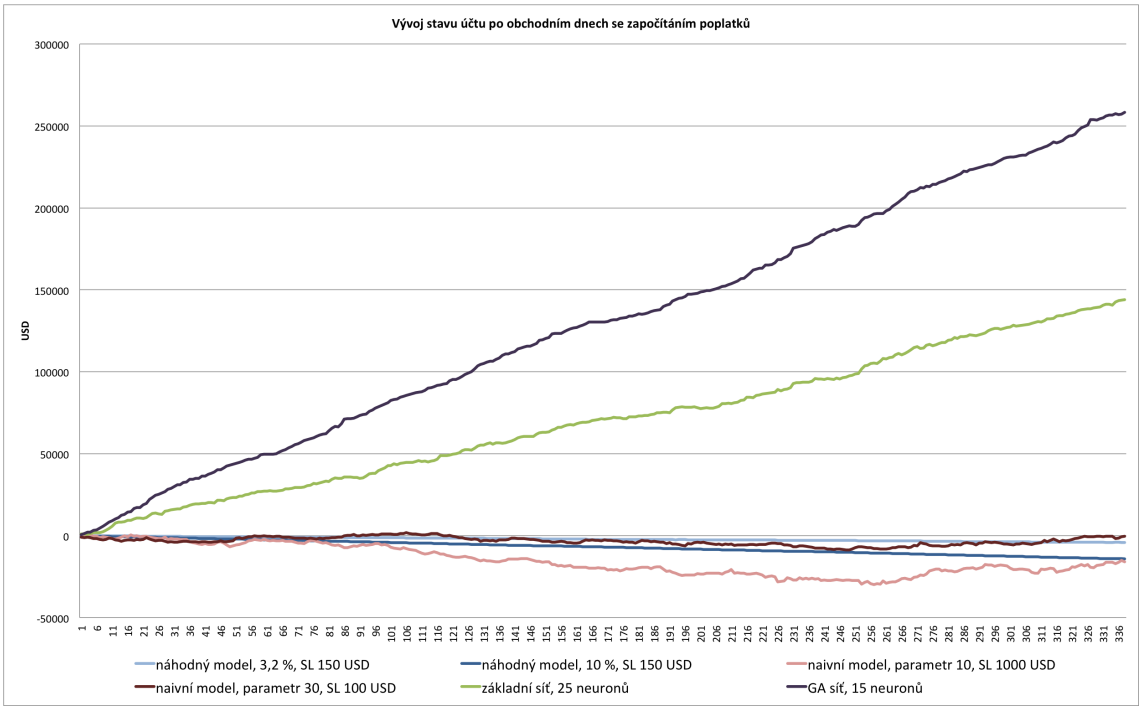
Model využívající neuronovou síť s parametry zjištěnými genetickým algoritmem má podobný průběh stavu účtů co se týče volatility a lineárního růstu jako předchozí model. Dosahuje však téměř dvojnásobně lepšího výsledku, což lze vysvětlit tím, že byla natrénována na nižší chybu a tedy vyšší přesnost.

Graf 4.13 zachycuje totéž co předchozí graf s tím rozdílem, že jsou započítány poplatky.

Oba náhodné modely lineárně klesají, ten s pravděpodobnostním parametrem 10 % klesá rychleji, neboť za den udělá více obchodů a bez započítání poplatků se drží okolo 0. Po započítání poplatků bude tedy lineárně klesat kvůli jejich kumulaci. Druhý náhodný



Obrázek 4.12: Srovnání průběhů stavů účtů po obchodních dnech bez započítání poplatků pro jednotlivé obchodní modely



Obrázek 4.13: Srovnání průběhů stavů účtů po obchodních dnech se započítáním poplatků pro jednotlivé obchodní modely

model klesá ze stejných důvodů.

Naivní model s parametrem 30 překonává druhý naivní model díky tomu, že v něm výborně funguje stop-loss a dělá méně obchodů. Bez započítání poplatků je lepší model s parametrem 10, jenže jeho bilance na jednotlivý obchod je horší a svého výkonu dosahuje velkým množství obchodů. Jakmile se započítají poplatky, velké množství obchodů se projeví na zásadním poklesu výkonu modelu.

Oba modely využívající neuronové sítě se chovají velice podobně, jako bez započítání poplatků - lineárně rostou a nejsou příliš volatilní.

Tabulka 4.12 shrnuje důležité výsledky jednotlivých modelů.

Vysvětlivky pro řádky:

náhodný, 3,2 % náhodný obchodní model, pravděpodobnost 3,2 %, SL 150 USD

náhodný, 10 % náhodný obchodní model, pravděpodobnost 10 %, SL 150 USD

naivní, param. 10 naivní obchodní model, parametr 10, SL 1000 USD

naivní, param. 30 % naivní obchodní model, parametr 30, SL 100 USD

základní sít obchodní model využívající základní neuronovou síť

optim. síť obchodní model využívající neuronovou síť optimalizovanou GA

Vysvětlivky pro sloupce:

B- průměrná bilance na jeden obchod bez započítání poplatků

B+ průměrná bilance na jeden obchod se započítáním poplatků

R poměr celkového zisku a celkové ztráty

průměrný počet obchodů za den

F- konečný stav účtu bez započítání poplatků

F+ konečný stav účtu se započítáním poplatků

Tabulka 4.12: Srovnání jednotlivých obchodních modelů

	B-	B+	R	#	F-	F+
náhodný, 3,2 %	0,0001 USD	-3,8 USD	1	3,2	200 USD	-4221 USD
náhodný, 10 %	0,000009 USD	-3,9 USD	1	10,7	347 USD	-14184 USD
naivní, param. 10	2,49 USD	-1,5 USD	1,35	31	26079 USD	-15788 USD
naivní, param. 30	3,8 USD	-0,1 USD	1,43	15,3	20219 USD	-544 USD
základní sít	75 USD	71 USD	9,74	5,99	152130 USD	144026 USD
optim. síť	260,3 USD	256,3 USD	94,3	2,98	262370 USD	258338 USD

Z tabulky je patrno, že oba náhodné modely modely stále oscilují okolo nuly, neboť jejich bilance na jeden obchod je bez poplatků téměř nulová (v případě modelu s pravděpodobností 10 % se více blíží 0, neboť provedl více obchodů a čím více obchodů, tím více bude výsledek konvergovat k 0). Bilance se započítáním poplatků se téměř rovná poplatku za obchod (ve

výši 4 USD), což dává smysl. Poměr zisků a ztrát je v obou případech roven 1, což taktéž odpovídá předpokladům.

Oba naivní modely vykazují lepší výsledky než náhodné modely, což je vidno na vyšším poměru zisků a ztrát. Oba modely jsou bez započítání poplatků relativně ziskové, po započítání poplatků se oba stávají prodělečnými.

Model využívající základní neuronovou síť dosahuje mnohem lepších výsledků než všechny předchozí modely. Je též vidět, že u něj klesl průměrný denní počet obchodů, přestože poměr zisků a proher a konečný stav účtu se výrazně zlepšily. Lze tedy usuzovat, že tento model si více vybíral obchodní příležitosti (tj. generoval méně obchodních signálu), ale o to byly příležitosti lepší.

Model využívající síť s parametry optimalizovanými genetickým algoritmem si vedl ještě lépe. Je vidět další snížení počtu obchodů denně a výrazné navýšení poměru zisků a ztrát. Znamená to tedy, že model si ještě lépe vybíral kvalitní a vhodné obchodní příležitosti a díky tomuto výrazně předčil všechny ostatní modely.

Kapitola 5

Implementace

Tato kapitola se zabývá popisem implementace systému a jeho důležitých funkčních částí. Na začátku budou představeny některé základní informace, poté architektura systému, následovat bude podrobnější popis důležitých balíčků a jejich tříd a nakonec bude zmínka o důležitých třídách z externích knihoven.

5.1 Obecné informace

Pro implementaci byl zvolen jazyk Java ve verzi 7.0. Volba jazyka je ovlivněna jednak jeho nezávislostí na platformě a jednak faktem, že pro něj existuje velké množství knihoven pro všechny možné účely (a tedy i ty implementující neuronové sítě a genetické algoritmy).

5.1.1 Použité knihovny

Systém využívá knihovnu JSAP[1] pro zpracování argumentů příkazové řádky, z knihovny Encog [2] používá neuronovou síť, genetický algoritmus a učící algoritmus resilient backpropagation a pro parsování konfiguračního JSON souboru používá knihovnu GSON[3].

5.1.2 Architektura systému

Diagram na obrázku 5.1 představuje pohled na architekturu systému. Je z něj zřejmé, že jsou využity 3 nezávislé knihovny (Encog, JSAP a GSON), a že samotný systém je rozdělen na více balíčků.

StockMarketPredictor název celého systému a hlavní třída

GA balíček obsahující třídy nezbytné pro genetický algoritmus

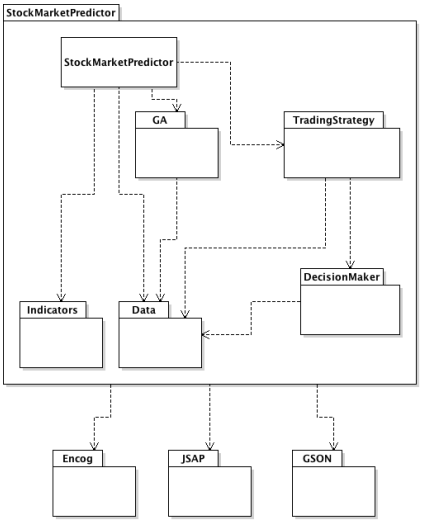
Indicators balíček obsahující indikátory technické analýzy

Data balíček obsahující třídy pro transformaci dat (např. z ticků na obchodní dny)

TradingStrategy balíček popisující jednotlivé obchodní strategie (např. strategie bez/se stop-lossem)

DecisionMaker obsahuje rozhodovací třídy

[1] the Java Simple Argument Parser, http://www.martiansoftware.com/jsap/
[2] Encog Machine Learning Framework, http://www.heatonresearch.com/encog
[3] google-gson, https://code.google.com/p/google-gson/



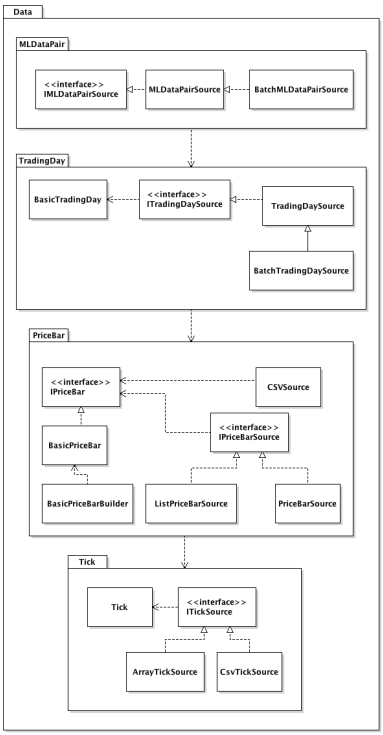
Obrázek 5.1: Zjednodušený pohled na architekturu systému

5.1.3 Popis balíčku a jejích tříd

Balíček Data

Jak již bylo řečeno, tento balíček se stará o transformaci nízkoúrovňových dat na složitější logické celky.

Obrázek 5.2 zobrazuje diagram tříd balíčku Data.



Obrázek 5.2: Diagram tříd balíčku Data

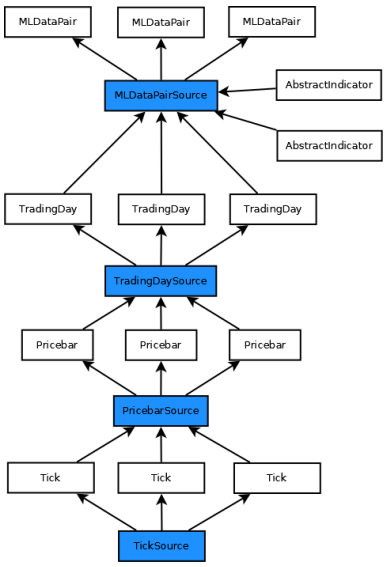
Je z něj patrné, že každá úroveň dat má vlastní balíček, který obsahuje alespoň jednu

třídu implementující rozhraní IXXXSource, kde XXX je označení úrovně dat. Tato třída má na starost generování dat svého typu (vetšinou je generuje z dat nižšího typu), funguje jako iterátor (tj. implementuje rozhraní Iterable<XXX>).

Pro obchodní dny a data pro neuronovou síť (tj. TradingDays a MLDataPair) je implementována ještě třída BatchXXXSource, která je potomkem třídy implementující příslušný generátor dat. Tato třída získá z generátoru všechna dostupná data a vrátí je jako seznam.

Třída MLDataPairSource navíc umožňuje vkládání indikátorů (podrobněji viz příloha), takže pro neuronovou síť jsou již korektně vygenerována data i s indikátory.

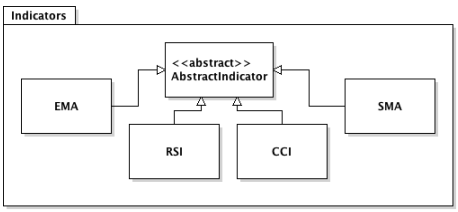
Proces transformace dat z nízkoúrovňových na data vyšší úrovně ilustruje obrázek 5.3.



Obrázek 5.3: Schéma transformace dat

Balíček Indicators

Obrázek 5.4 zobrazuje diagram tříd balíčku Indicators.



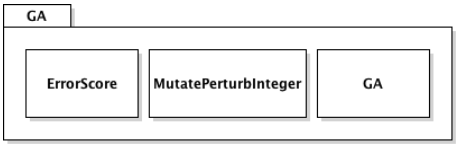
Obrázek 5.4: Diagram tříd balíčku Indicators

Obsahuje 5 tříd, z toho jednu abstraktní, ze které dědí všechny indikátory. Tato abstraktní třída obsahuje společné prostředky pro práci s historickými hodnotami indikátorů a posouvání časového horizontu.

Jednotlivé indikátory pak obsahují metody pro výpočet své hodnoty.

Balíček GA

Obrázek 5.5 zobrazuje diagram tříd balíčku GA.

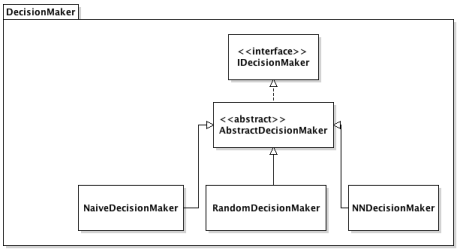


Obrázek 5.5: Diagram tříd balíčku GA

Třída GA obsahuje inicializaci počáteční populace a metody pro převod mezi genomem a parametry. ErrorScore definuje fitness funkci pro GA a MutatePerturbInteger je genetický operátor, který popisuje mutaci vektoru celých čísel (takový operátor Encog neobsahuje, bylo nutné jej doimplementovat).

Balíček DecisionMaker

Obrázek 5.6 zobrazuje diagram tříd balíčku DecisionMaker.



Obrázek 5.6: Diagram tříd balíčku DecisionMaker

Balíček obsahuje rozhraní, které umožňuje dvě základní operace - zjistit, kterou akci rozhodovací třída doporučuje (vstup do dlouhé/krátké pozice, ukončení pozice, nedělat nic) pomocí metody getAction() a posunutí časového horizontu metodou moveHorizon(). Abstraktní rozhodovací třída obsahuje operace pro zjišťování aktuálního stavu systému (zda-li je v pozici a pokud ano, tak v jaké).

Jednotliví potomci abstraktní třídy poté implementují výše uvedené operace - třída RandomDecisionMaker implementuje náhodné rozhodování, NaiveDecisionMaker se rozhoduje podle naivního modelu založeném na klouzavých průměrech (viz předchozí kapitoly) a NNDecisionMaker (zkratka pro NeuralNetworkDecisionMaker) se rozhoduje na základě výstupu naučené neuronové sítě.

Balíček TradingStrategy

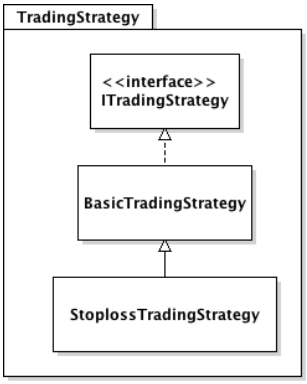
Obrázek 5.7 zobrazuje diagram tříd balíčku TradingStrategy.

Rozhraní definuje některé důležité metody, hlavně tedy metodu trade(), která provede obchodování v rámci jednoho obchodního dne.

Třída BasicTradingStrategy jednoduše obchoduje na základě výstupu rozhodovací třídy. StoplossTradingStrategy obchoduje stejným způsobem, navíc ale hlídá nastavený stop-loss a v případě, že je překročen, ukončuje pozici.

Třída StockMarketPredictor

Tato hlavní třída zpracovává pomocí knihovny JSAP parametry příkazové řádky, načítá konfigurační soubor a provádí požadovanou činnost. Lze požadovat natrénování neuronové

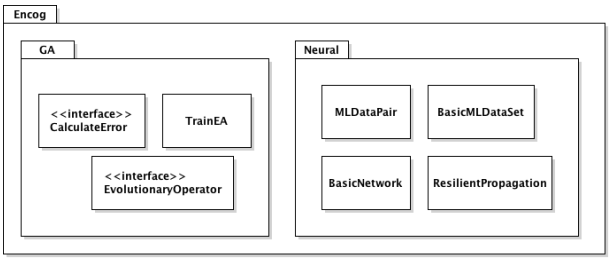


Obrázek 5.7: Diagram tříd balíčku TradingStrategy

sítě, obchodování s daty v definovaném souboru a spuštění genetického algoritmu pro optimalizaci parametrů neuronové sítě.

Knihovna Encog

Obrázek 5.8 zobrazuje upravený a zjednodušený (kvůli přehlednosti) diagram tříd balíčku Encog.



Obrázek 5.8: Zjednodušený diagram tříd balíčku Encog

Balíček GA obsahuje samotný genetický algoritmus, který je implementován ve třídě TrainEA. Výchozí nastavení používá pro výběr rodičů turnajový systém a přesouvá 30 % nejlepších jedinců rovnou do další generace na základě principu elity. Rozhraní CalculateError definuje metody, které musí implementovat fitness třída. Ty jsou hlavně 2 důležité - shouldMinimize(), která říká, zda se má fitness funkce minimalizovat či maximalizovat, a pak calculateScore(), která vypočítá zadanému jedinci hodnotu fitness funkce. Rozhraní EvolutionaryOperator definuje metody, které musí implementovat evoluční operátor - nejdůležitější je metoda performOperation(), která na zadaných jedincích provede danou operaci. Další metody jsou parentsNeeded() a offspringProduced(), která definují, kolik rodičů je potřeba a kolik potomků danou operací vznikne (pro mutace je potřeba 1 rodič a vznikne 1 potomek).

Balíček Neural obsahuje třídy pro práci s neuronovou sítí. Třída MLDataPair je datový kontejner nesoucí pár vstupní vektor a příslušný výstupní vektor, BasicMLDataSet je taktéž kontejner nesoucí seznam MLDataPair. BasicNetwork je samotná neuronová síť, které lze nastavovat počty neuronů v jednotlivých vrstvách, přidávat a odebírat vrstvy, měnit váhy synapsí a prahy neuronů, definovat aktivační funkce a provádět samotné výpočty. ResilientPropagation je třída implementující učící algoritmus resilient backpropagation odvozený

od backpropagation. Umožňuje volit jednotlivé varianty (RPROP-, RPROP+, iRPROP-, iRPROP+), nastavovat velikost dávky, po které se upravují váhy sítě a v neposlední řadě samozřejmě trénovat neuronovou síť.

Knihovna JSAP

Tato knihovna slouží pouze pro zjednodušení práce při zpracování parametrů příkazové řádky, nebude zde tedy detailněji popsána.

Knihovna GSON

Platí pro ni totéž co pro JSAP.

Kapitola 6

Závěr

V této kapitole budou představy možné směry dalšího vývoje a shrnuty dosažené výsledky.

6.1 Směry dalšího vývoje

Jednou z možných cest, kterou se vydat, je pokusit se přiblížit se co nejvíce realitě, tj. implementovat chování brokera. V této práci se obchoduje na úrovni agregovaných svíček - bere se, že je možné vstoupit do trhu na close minulé svíčky. Toto však nemusí být pravda a pokud jsou k dispozici ticková data, bylo by možné provádět test, zda byl daný obchod vykonán na dané cenové hladině.

Další věcí, kterou je možné zavést v rámci implementaci brokera, je rozlišovaní mezi jednotlivými typy příkazů (např. rozlišování mezi MARKET a LIMIT příkazy) a jejich příslušné vykonání. Toto může zásadně ovlivnit výsledky obchodní strategie.

Ideálním způsobem, jak implementovat brokera, je samozřejmě využít reálného brokera, bohužel nebyl nalezen broker, který by nabízel demoúčet s možností napojení na API a reálná realtime data, to vše zdarma. Je dokonce velmi nepravděpodobné, že by takový broker vůbec existoval, neboť už jen za samotná realtime data se platí nemalé sumy.

Dalším možným směrem vývoje by mohlo být testování přístupu použitého v této práci na vetším množství dat. V této práci byly použity data z cca dvou let, což však bylo období velmi klidné a trhy měly rostoucí tendenci. Existuje riziko, že při změně globální ekonomické situace by se takto naučený a nastavený systém nebyl schopen v rozumné době adaptovat, jeho přesnost by rapidně klesla a to by mohlo přinést obrovské finanční ztráty.

Stejně tak by bylo zajímavé otestovat systém natrénovaný na těchto datech na jiném trhu, ideálně mnohem volatilnějším - např. forex.

Dva předchozí odstavce se tedy dají shrnout následovně - pro reálné využití by bylo nezbytné provést důkladné a zevrubné testy robustnosti systému.

Další cestou by mohlo být zakomponování určitých fundamentálních dat (např. úroková sazba FEDu mění globální ekonomickou situaci). V tomto případě by však mohlo být obtížné tato data správně a vhodně zakódovat.

Zajímavé výsledky by mohlo dávat změna timeframu - 2 minuty se často používají pro lidské obchodování. Radikálním snížením (na úroveň setin sekund) bychom se dostali na pole HFT (high frequency trading), což je čistě strojová doména, neboť člověk není schopen v takto krátkých časem ani zareagovat. V HFT je ovšem obrovská konkurenci bank a velkých fondů s prakticky neomezeným kapitálem a přístupem k nejlepším mozkům planety. Naopak zvýšení timeframu umožňuje ještě více omezit šum dat a vypozorovat nějaké zákonitosti,

což by mohlo být zajímavé např. pro forex, známý svou volatilitou.

Taktéž úprava celé obchodní strategie by mohlo značně zlepšit výsledky - aplikace profit targetů, sofistikovanější money management, možnost otevření více pozic zároveň atd.

6.2 Shrnutí výsledků

Cílem práce bylo navrhnout a implementovat systém, který bude schopen autonomně obchodovat, a poté vyhodnotit jeho účinnost.

Systém byl implementován, jako referenční model byl zvolen náhodný obchodní model. Pro srovnání byl implementován i naivní model založený na klouzavých průměrech.

Ukázalo se, že referenční náhodný model dosahuje nejhorších výsledků, těsně následován naivním modelem. Model využívající neuronovou síť dával několikanásobně lepší výsledky. Poté byl pro hledání optimální kombinace vstupních parametrů využit genetický algoritmus, který úspěšně našel parametry lepší než ty původně zvolené.

Model využívající síť, jejíž vstupní parametry byly nalezeny GA, dává téměř dvakrát lepší výsledky než model se základní sítí. Ověřilo se tedy, že GA je schopen najít lepší parametry než ty, které byly v podstatě náhodně zvoleny.

Literatura

[1] Igel, C.; Hüsken, M.: Improving the Rprop Learning Algorithm. 2000.

[2] Lukáš Putna: PREDIKCE VÝVOJE KURZU POMOCÍ UMĚLÝCH NEURONOVÝCH SÍTÍ. diplomová práce na FIT VUT, 2011.

[3] Petr Podhajský, Tomáš Nesnídal: *Kompletní průvodce úspěšného finančníka*. Centrum finančního vzdělávání, 2009, ISBN 9788090387454.

Příloha A

Konfigrační soubor

Konfigurační soubor je ve formátu JSON [1]. Skládá se ze 2 povinných a 4 nepovinných polí.

Povinná pole jsou *inputWindow* a *outputWindow*, která udávají velikost vstupního a predikčního okna.

Nepovinná pole:

indicators obsahuje pole objektů definující indikátory (podrobnosti viz readme soubor na DVD)

pricebarFile udává název datové souboru

hiddenNeuronCount udává počet neuronů ve skryté vrstvě

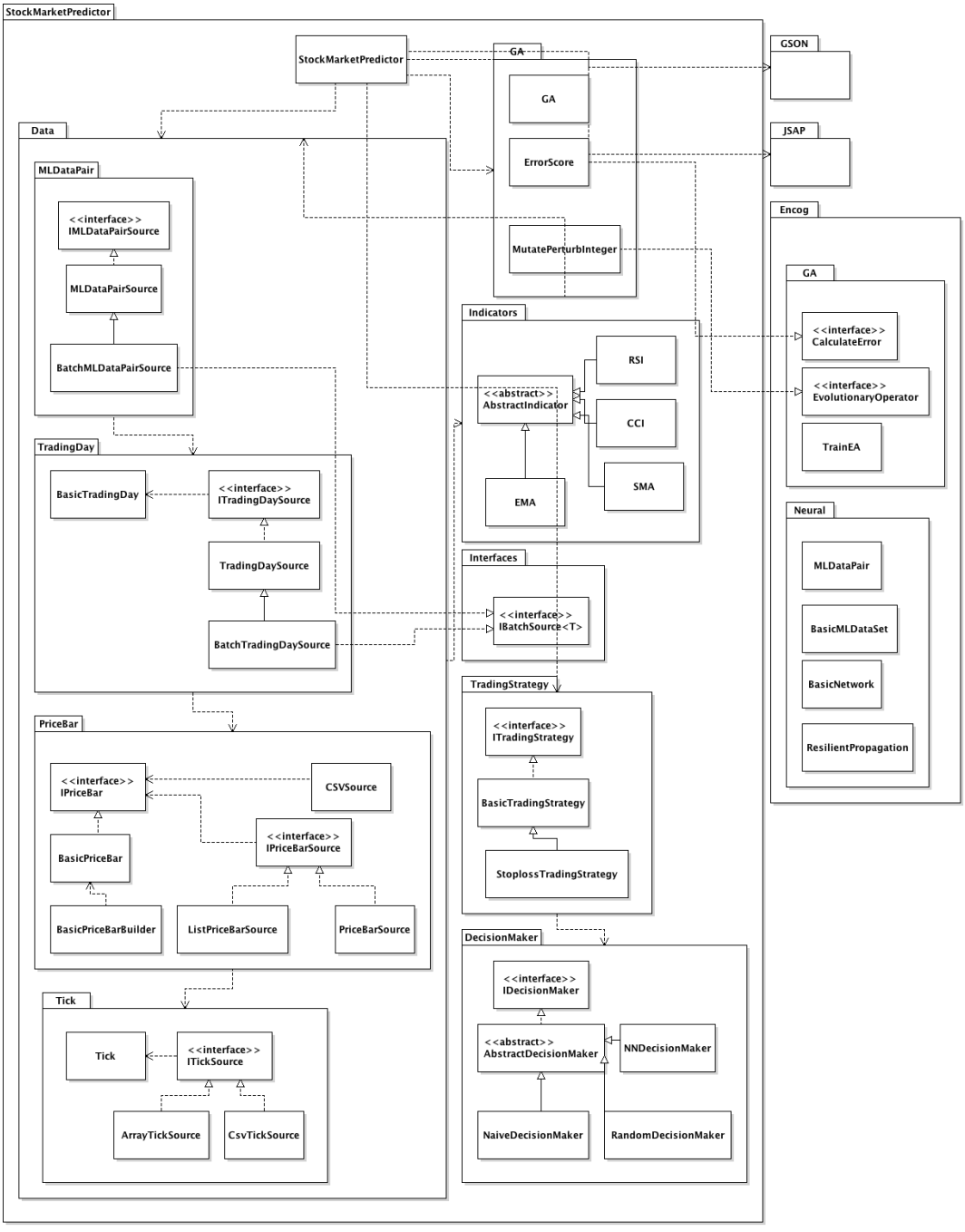
tradingThreshold velikost prahu, tj. o kolik se musí lišit současná a predikovaná hodnota, aby byl generován obchodní signál

[1] http://www.json.org/

Příloha B

Diagram tříd

Obrázek B.1 zobrazuje podrobný diagram významných tříd. Názvy balíčků se mohou lišit od názvů balíčků ve zdrojovém kódu, některé balíčky byly zjednodušeny kvůli přehlednosti.



Obrázek B.1: Podrobný diagram tříd