ZÁPADOČESKÁ UNIVERZITA V PLZNI

FAKULTA EKONOMICKÁ

Diplomová práce

Systémy pro podporu finančního rozhodování

Financial Decision Support Systems

Bc. Štěpán Havlovic

Plzeň 2021

Čestné prohlášení

Prohlašuji, že jsem diplomovou práci na téma

„Systémy pro podporu finančního rozhodování“

vypracoval samostatně pod odborným dohledem vedoucího diplomové práce za použití pramenů uvedených v přiložené bibliografii.

Plzeň dne 10. 5. 2021 *v. r. Štěpán Havlovic*

Poděkování

Rád bych poděkoval vedoucímu diplomové práce doc. RNDr. Mikuáši Gangurovi, Ph.D. za sdílení jeho hlubokých znalostí prostřednictvím rad, připomínek a za pomoc se strukturováním práce. Dále bych rád poděkoval mému bratrovi Ondřeji Baštaři za konzultace při vytváření skriptů a celé rodině, která mne při psaní práce podporovala.

Obsah

Úvod

Rozšiřování znalostí o jakékoliv podnikatelské činnosti je podmíněno získáváním a zpracováním informací. Každým rokem se množství publikovaných informací zvyšuje. Přináší to mnohá úskalí pro jednotlivce, ale i společnosti usilující o prohloubení znalostí v kýžené problematice. Nemusíme hovořit pouze o lidech nabízejících produkty a služby. I pro stranu poptávky je dostupné doslova nepřeberné množství informací. Většina lidí se v průběhu života musela rozhodnout o koupi nového mobilního telefonu. Každý den je možné v médiích nalézt reklamu na produkt nějaké společnosti. V případě, že si telefon musíme v nejbližší době koupit a nejsme předem rozhodnuti o konkrétní značce ani typu, začneme hlouběji analyzovat reklamy, recenze, online obchody nebo stránky výrobců. Zanedlouho zjistíme, že není v našich silách přečíst a zpracovat všechny dostupné informace. Diplomová práce se věnuje oblasti finančních trhů, na kterých působí velké množství subjektů a denně vzniká enormní množství dat a informací popisujících finanční trh.

Hlavním cílem práce je **vytvoření systémů pro podporu rozhodování při nákupu a prodeji aktiv na finančním trhu**. Pro splnění hlavního cíle je nezbytné splnit následující dílčí cíle:

- Teoretický popis možností pro získávání a ukládání informací o finančních trzích.
- Identifikace indikátorů technické, fundamentální analýzy a jejich implementace do obchodních strategií.
- Návrh a vytvoření systému pro zpětné testování obchodních transakcí.
- Návrh funkcionality webové aplikace a její vytvoření.

Systémem pro podporu rozhodování u diskrečního obchodování je interaktivní webová aplikace, sloužící investorovi jako zdroj informací při formulování rozhodnutí o alokaci finančních prostředků. Podobně jako lidé, poptávající mobilní telefon využívají server: https://www.heureka.cz, na kterém je možné nalézt recenze online obchodů a hodnocení jednotlivých výrobků. Nejlepší diskreční obchodníci se při formulování investičního rozhodnutí řídí tacitními znalostmi. Je obtížné přesně popsat všechny důvody, které vedly diskrečního obchodníka k nakoupení, nebo prodání daného aktiva. V případě, že by se nám povedlo identifikovat všechny důvody a přidělit jim váhu, je velice pravděpodobné, že u jiného aktiva by stejný člověk postupoval odlišným způsobem. Oproti tomu

postavíme obchodování kvantitativní, s přesně definovanými podmínkami pro vytvoření obchodu a jeho ukončení. Neměnné podmínky, při kterých dojde k nakoupení akcie formulují strategii. U jasně specifikované a neměnné strategie se pomocí zpětného testování pokusíme ověřit, zdali je možné pomocí různých indikátorů vytvořit profitabilní obchodní systém.

Pro uvedení do problematiky se v první části rešerše zaměříme na různé toky financí, proudící na finanční trhy. Poté se zaměříme na akciové trhy a popíšeme rozdíly mezi fundamentální a technickou analýzou a kvantitativním a diskrečním obchodováním. Po vysvětlení těchto termínů se ve stejné kapitole věnujeme generování obchodních signálů a obchodních strategií. Některé části práce budou aplikovatelné i na další finanční trhy, jako například: trh s devizovým kursem, dluhopisy a kryptoměnami. Pro zjednodušení jim nebude věnována velká pozornost. Základním stavebním kamenem, vedoucím k vytvoření systému pro podporu finančního rozhodování je získání potřebných dat. Popíšeme zpravodajské weby, sociální sítě, fundamentální a technická data. Konkrétně představíme možné způsoby získávání jednotlivých dat. Vzhledem k tomu, že máme zájem data v budoucnu využít, v další kapitole rešerše se zaměříme na jejich ukládání. V našem případě nebude dostačující oblíbený program pro uchovávání dat Microsoft Excel. Z toho důvodu pronikneme do problematiky zkoumající různé druhy databázových systémů. Před samotným uložením bude nutné data upravit do vhodné podoby, proto bude následovat kapitola věnovaná předzpracování dat.

Při výběru zmíněného mobilního telefonu se po nalezení a zpracování jednotlivých článků věnujeme, možná nevědomě, analýze sentimentu. Utváříme si názor na sdělované informace a rozlišujeme mezi pozitivní a negativní recenzí. Při vytváření systémů pro podporu finančního rozhodování se zabýváme pozitivitou a negativitou článků zveřejněných na zpravodajských webech a sociálních sítích. Člověk je schopen rozhodnout o sentimentu relativně bez obtíží. Jedná se ale vždy o subjektivní pohled. My se pokusíme pro kvantitativní vyjádření sentimentu použít výpočetní techniku. Jedná se o velmi obsáhlé a komplikované téma, vznikají celé diplomové práce věnující se pouze analýze sentimentu. Pokud by se v budoucnu objevily lepší přístupy, například výborně natrénovaná umělá inteligence, na námi získaných a uložených datech bude možné jednoduše využít nový výpočet sentimentu a jeho hodnoty upravit.

V předposlední části rešerše se pokusíme vybrat vhodný nástroj pro zpětné testování obchodních strategií. Pokud bude obchodování profitabilní, můžeme strategii využít pro

obchodování v reálném čase s reálnými penězi. Po vybrání vhodného nástroje, knihovny pro námi preferovaný programovací jazyk, principiálně popíšeme jeho využívání.

V poslední části rešerše se budeme věnovat vytvoření webové platformy, výběr vhodného nástroje pro prezentaci dat a popíšeme existující aplikace.

Následuje kapitola s formulací problému a popisem jeho řešení. Je identifikován přínos naší práce, postup při zpětném testování obchodních strategií, využívaná data a jsou definovány konkrétní strategie, které budou testovány. Také je popsán způsob tvorby webové aplikace a definovaná její funkcionalita.

V praktické části budou realizovány obchodní strategie s různými metodami pro generování obchodních signálů. Jak strategie, tak metody jsou identifikovány v kapitole 1.2. Následně budou prezentovány výsledky ve formě: zisku, ztráty, minimální hodnota portfolia a podobně. Jako další bude praktická část obsahovat postup vytvoření webové aplikace, ukázku její funkcionality a návrh konkrétních způsobů využití.

Před závěrem práce budeme diskutovat o přínosech, nedostatcích a možném rozšíření diplomové práce. V závěru autor posuzuje naplnění hlavního a dílčích cílů diplomové práce.

1 Teoretická východiska

1.1 Finanční trhy

Zajišťují základní ekonomickou funkci, kterou je přerozdělování bohatství od ekonomických subjektů s přebytkem příjmů k subjektům s nedostatkem zdrojů. Pokud člověk utrácí méně peněz, než vydělá, může vzniklé úspory využít k podpoře subjektu, který dodatečné zdroje, v ideálním případě, využije k výrobě užitečných komodit a generování zisku (Mishkin, 2019).

Tok přebytku financí od věřitelů (domácností, firem nebo vlády) k dlužníkům může být přímý nebo nepřímý. Strukturou těchto dvou možností interakce mezi věřitelem a dlužníkem se zabývají následující dvě podkapitoly.

1.1.1 Přímý tok financí

Dlužník si půjčuje finanční zdroje přímo od věřitelů vyskytujících se na finančním trhu. Věřitel nakupuje finanční instrumenty od dlužníka, čímž získává možnost podílet se na budoucí ztrátě, ale i výnosech, které plynou z ekonomické aktivity dlužníka. Existuje několik kategorií finančních trhů:

- **Dluhové a akciové trhy**
  Firma nebo jednotlivec může získat finanční prostředky nabídnutím dluhových nástrojů, které zavazují dlužníka splácet vypůjčenou částku spolu s danými úroky v předem domluveném intervalu. Další možností je emitování akcií. Pokud věřitel nakoupí 10 % akcií společnosti, stane se vlastníkem 10 % aktiv společnosti. Cena akcie se s plynutím času mění, to může být výhodou i nevýhodou. Analýza výkyvů cen akcií následkem zveřejnění fundamentálních veličin a po změně názoru lidí na danou společnost (sentimentu), jsou jedním z hlavních předmětů této práce. Nevýhodou oproti vlastnění dluhových nástrojů je to, že v případě úpadku musí společnost nejdříve splatit závazky dlužníkům a až poté uspokojuje akcionáře.
- **Primární a sekundární trhy**
  Na primárním trhu dochází k prodeji nově emitovaných akcií a dluhopisů. Nevýhodou je nízká likvidita. Výhodou nižší cena, vzhledem k neexistenci dalších subjektů podílejících se na vznikajícím obchodním vztahu. U většiny

účastníků dochází k interakci s finančním trhem až na sekundárním trhu. Důležitou součástí sekundárního trhu jsou makléři(*brokeři)*, kteří poptávce nabízejí finanční nástroje. Po sjednání ceny, při které má dojít k nákupu, *dealeři* propojí nakupující a prodávají uskutečněním vlastního nákupního a prodejního pokynu. Výhodou může být vysoká likvidita. Nevýhodou vyšší transakční náklady.

- **Burzovní a mimoburzovní trhy**

  V minulosti bylo jedinou možností vstoupit na finanční trh pouze fyzickým vstupem na podlahu směnárny (trading floor). S rozvinutím internetu se stal mimoburzovní (OTC) trh nejvyužívanějším způsobem pro nákup finančních instrumentů.

- **Peněžní a kapitálové trhy**

  Na peněžním trhu jsou obchodované dluhové nástroje se splatností do jednoho roku. Na kapitálovém trhu jsou obchodované dluhové nástroje se splatností od jednoho roku výše, spolu s kapitálovými nástroji. Na peněžním trhu je možné obchodovat s: *pokladničními poukázkami, bankovními certifikáty a depozity, komerčními papíry, dohodami o zpětném odkupu a fondy*. Na kapitálovém trhu s: *akciemi, hypotékami a cennými papíry zajištěnými hypotékou, korporátními dluhopisy, státními dluhopisy, cennými papíry vládních agentur, spotřebitelskými a bankovními půjčkami.* Jedná se o výpis možností, které se vyskytují ve Spojených státech amerických, v jednotlivých zemích se mohou odlišovat (Mishkin, 2019).

1.1.2 Nepřímý tok financí

U nepřímého toku financí se mezi střadatelem a dlužníkem vyskytuje prostředník, napomáhá s přenosem financí od jednoho subjektu ke druhému. Prostředník si půjčuje zdroje (úspory) od střadatele a využívá je k poskytnutí půjček dlužníkovi. Studie prováděné na rozvinutých zemí ukázaly, že podniky získávají většinu zdrojů využitím prostředníků, tedy nepřímého toku financí. Jednotlivé země se ale odlišují ve využití dluhových a akciových trhů, mezi kterými neexistuje jasná preference. Zajímavá je situace ve Spojených státech amerických, kde podniky získávají na dluhovém trhu desetinásobně více finančních prostředků než na trhu akciovém. Existuje několik typů

zprostředkovatelů. Jsou to: banky, spořitelny a investiční společnosti. Důvody, které vedou většinu firem k využití prostředníků jsou následující:

- **Transakční náklady**
  Výskyt podobných investičních příležitostí vede u finančního zprostředkovatele ke značným úsporám z rozsahu. Proto je například banka schopna půjčovat malé částky, navzdory tomu, že sepsání jedné smlouvy je velmi nákladné.
- **Sdílení rizik**
  Nízké transakční náklady umožňují zprostředkovatelům prodávat riziková aktiva a získané peníze využít na nákup aktiv ještě více rizikových, tomuto procesu se také říká transformace aktiv, jelikož zprostředkovatel v jistém smyslu transformuje riziková aktiva na bezpečnější. Další výhodou sdílení rizik je možnost nákupu portfolia aktiv s různým typem diverzifikace.
- **Asymetrické informace**
  Nemožnost jednotlivce zjistit všechny informace které souvisejí s jeho finančním rozhodnutím. S informacemi existujícími již před obchodní transakcí souvisí termín *nepříznivý výběr*. Ten se vyskytuje, vzhledem k tomu, že rizikoví poptávající vyhledávají dodatečný kapitál aktivněji než méně rizikoví dlužníci. Po uskutečnění investice je problémem *morální hazard* neboli situace kdy se dlužník po získání finančních prostředků může zapojit do aktivit, které věřitel předem nepředpokládal (Mishkin, 2019).

1.2 Akciové trhy

I když je možné použít některé poznatky z diplomové práce i na jiných trzích (komoditních nebo například nově vzniklého trhu s kryptoměnami), budeme se dále zabývat převážně trhem akciovým. Tato kapitola popisuje možné způsoby analýzy akciového trhu a přístupy k obchodování.

1.2.1 Fundamentální analýza

Určujeme, zdali je aktuální hodnota akcie předražena či podhodnocena na základě jak minulého, a současného tak předpovídaného budoucího průběhu fundamentálních veličin, těmi jsou například: *zisk, zadluženost, dividendy, makroekonomické indikátory, management firmy* nebo *sentiment* společnosti (Jean-Philippe Bouchaud, 2018). Získávání těchto dat je hlouběji zpravováno v kapitolách: 1.3.1, 1.3.2 a 1.3.3. Na internetu

existuje mnoho webových stránek, soustřeďujících se na zveřejňování zpráv a dat využívaných pro fundamentální analýzu, například: *https://finance.yahoo.com, https://atom.finance* a https://finviz.com. Zároveň jsou fundamentální informace velmi často publikovány v televizním nebo rádiovém vysílání a novinových článcích

1.2.2 Technická analýza

Zabývá se studiem průběhu ceny akcie. Klasická technická analýza využívá historických cen zobrazených v grafu a pomocí různých finančních indikátorů, s velkým důrazem na přímkové indikátory, se snaží předpovědět budoucí průběh. V současnosti je možné počítačově generovat velké množství indikátorů technické analýzy a zobrazovat je překrytím grafu: *Bollinger Band, klouzavý průměr.* Využívány jsou také oscilátory zobrazované v samostatném okně: *Klouzavý průměr konvergenční divergence* a mnoho dalších (Edwards, Magee, & Bassetti, 2018).

Hojně využívaným indikátorem technické analýzy je klouzavý průměr (MA). Na datech s vysokou zrnitostí a velkým rozptylem může být po vypočtení klouzavého průměru lépe pozorovatelný trend. Pro nás může být zajímavé použití MA například pro vyhlazení dat sentimentu a ceny akcie. Výpočet MA je následující:

$$MA = \frac{x_{t-1} + x_{t-2} + x_{t-3} + x_{t-4} \dots x_{t-n}}{n}$$

MA = Klouzavý průměr.

x_{t-1} = Hodnota ceny akcie nebo sentimentu v periodě t-1.

n = Počet period.

1.2.3 Kvantitativní obchodování

Investiční rozhodnutí závisí na počítačově generovaných signálech, které stojí na předem stanovených podmínkách, generujících obchodní signály a analýze velkého množství dat (Abis, 2017).

1.2.4 Diskreční obchodování

Investiční rozhodnutí závisí na rozhodnutí investičních manažerů, kteří mohou mít kvantitativní systémy pro generující obchodní signály, ale založení obchodního pokynu závisí čistě na jejich osobním rozhodnutí (Abis, 2017).

1.2.5 Generování obchodních signálů

V předchozích dvou kapitolách hovoříme o obchodních signálech. Nyní se musíme zaměřit na jejich identifikaci. Ke generování obchodních signálů můžeme použít například metody technické a fundamentální analýzy, ale i strojové učení.

Mezi metody technické analýzy patří:

- Trendové linie: jsou přímky, které znázorňují důležité cenové úrovně. Využívají se k identifikování hladiny rezistence a podpory. Pokud budeme hovořit o ceně akcie, podpora (support) je dolní hranice rozmezí, ve kterém se cena pohybuje a rezistence (resistant) je hranice horní.
- Grafické formace: jsou stejně jako trendové linie jednoduše vizuálně identifikovatelné, ale jedná se o subjektivní metody, jelikož každý člověk může nalézt odlišné formace. Jsou to například: Hlava a ramena, trojúhelník a vlajka.
- Klouzavé průměry: jsou nejčastěji využívány 20, 50 a 200denní. Jako obchodní signál se považuje přechod klouzavého průměru s nízkou periodou přes klouzavý průměr s vysokou periodou. Možné je také využít indikátory založené na klouzavých průměrech. Indikátor MACD vyjadřuje vychýlení hodnoty krátkodobého MA od hodnoty dlouhodobého MA. Mimo jednoduchý klouzavý průměr, jehož výpočet je možné nalézt v kapitole 1.2.2 se také využívá exponenciálního MA.
- Volume: vyjadřuje množství dat za danou periodu. V případě ceny akcie ukazuje množství obchodů, které proběhly za námi sledovanou periodu. V případě, že je množství obchodů vysoké, je podpořen růstový trend a naopak (Edwards, Magee, & Bassetti, 2018).

Metody fundamentální analýzy jsou například:

- Zisk na akcii: je cena akcie vydělena množstvím akcií.
- Volné hotovostní prostředky: je množství peněz, které jsou k dispozici pro vlastníky a akcionáře.
- Zadlužení vlastního kapitálu: je poměr cizích zdrojů vůči vlastnímu kapitálu.
- Množství vyplácených dividend: značí, jak velkou část čistého příjmu společnost využívá na vyplácení dividend (Nikolaev, 2019).

Metody strojového učení:

- Deep learning: předpokládá, že data jsou generována funkcemi s víceúrovňovou hierarchií. Umělá inteligence se snaží vytvořit jednoduché nelineární funkce. Kombinací těchto funkcí je možné naučit umělou inteligenci rozpoznávat komplexní funkce, které je možné použít i na jiná data. Využívá se také při rozpoznávání obrázků.
- Opakující se neuronové sítě: umožňují využít informace z předchozího pozorování při provádění následujícího výpočtu. Využívají se například při výpočtu sentimentu.
- Deep reinforced learning: do deep learningu implementují metody odměňování, které podporují formování správných funkcí. Při generování obchodních signálů můžeme odměňovat v případě, že generovaný signál má za následek přírůstek peněžních prostředků (Jansen, 2020).

1.2.6 Obchodní strategie

Pomocí obchodních signálů je možné vytvářet obchodní strategie. Existuje velké množství parametrů, které musíme zvážit při formulaci obchodní strategie, například:

- Využívané obchodní pokyny: dlouhé (znamenají nákup) a krátké (prodej).
- Doba držení aktiva: dělí strategie na krátkodobé a dlouhodobé.
- Používání příkazů pro uzavření obchodu v případě vysokého zisku nebo ztráty.
- Množství používaných signálů: můžeme použít jenom jeden pro nákup a držet akcii po neomezenou dobu. Nebo využít signálů více jak pro nákup, tak pro uzavření obchodu.
- Rozhodnutí o konkrétních signálech: můžeme využívat pouze signálů technických. Často se kombinují technické a fundamentální signály.
- Výběr aktiv, na kterých budeme strategii využívat: strategie můžeme využívat například jen pro obchodování s akciemi technologických společností.

Z těchto parametrů můžeme například zformulovat strategii, která využívá pouze dlouhé obchodní pokyny, doba držení aktiva je jeden den, obchod uzavře v případě ztráty 0,2% celkové hodnoty portfolia, k nakoupení dojde v případě přechodu krátkodobého průměru přes dlouhodobý a bude využíván pouze na akciích s vysokou volatilitou.

Kombinací parametrů je možné vytvořit téměř nekonečné množství strategií. V praktické části testujeme obchodní strategie pomocí skriptů, ve kterých je možné snadno měnit vybrané parametry.

1.3 Programovací jazyk

Pro vytvoření systému pro zpětné testování obchodních transakcí i webové aplikace musíme rozhodnout o využívaném programovacím jazyku. Preferujeme programovací jazyk Python, ale v případě identifikace výrazně lepších nástrojů, dostupných pouze pro jiný programovací jazyk, se nebráníme využití jiného programovacího jazyka.

1.4 Získávání dat

Data pro fundamentální a technickou analýzu je možné získat z nepřeberného množství zdrojů. V současné době je nejjednodušší využít zdroje internetové. Zejména u analýzy sentimentu se bohužel budeme muset oprostit od získávání dat z rádiového a televizního vysílání vzhledem k obtížnosti této problematiky. Možné způsoby získávání potřebných dat jsou popsány v následujících kapitolách.

1.4.1 Zpravodajské weby

První možností, jak získat články, v našem případě finanční, je využití web scrapingu neboli hledání a ukládání informací přímo z HTML kódu webové stránky. Zdrojový kód webových stránek obsahuje velké množství příkazů a informací, které například zajišťují, že se objekty nacházejí na správném místě. Proto je vhodné pro získání potřebných informací využít některou z knihoven dostupných pro většinu programovacích jazyků. Hojně využívanou knihovnou pro Python je Beautiful Soup. Umožňuje extrahování informací obsažených v jednotlivých částech HTML kódu (Sarkar, Text Analytics with Python, 2019). Například na webu Reuters jsou pro nás relevantní informace uloženy v těchto částech kódu: `<title> </title>` obsahuje nadpis stránky, `<div class="StandardArticleBody_body"> </div>` obsahuje text článku a `<div class="ArticleHeader_date"> </div>` obsahuje datum a čas zveřejnění článku. Bohužel se ale nejedná o standardizovaný zápis. Pokud bychom chtěli tyto informace sbírat i z jiných webů zjistíme, že jsou uloženy v jiných blocích.

Druhou možností je využití API. Pro získávání nejen finančních zpráv existuje mnoho firem nabízejících připojení k jejich API, pomocí kterého lze snadno stahovat články

obsahující klíčová slova, z různých webů. Společnosti s největším podílem na trhu s finančními daty jsou velmi dobře známé. Jedná se o Bloomberg (33,4 % podíl) a Reuters (23,1 % podíl). Obě společnosti nabízejí přístup k API, s jehož pomocí je možné stahovat nejen novinkové články. Cenu služeb neprezentují veřejně, ale z neověřených zdrojů můžeme zjistit ceny pohybující se kolem $20 000 ročně (Kolakowski, 2020). Pro naše účely využijeme API dostupné na https://newsapi.org. News API nabízí možnost stahovat články z 50 000 blogů a novinových serverů. Cena pro komerční využití je $449 respektive $849 měsíčně. Pro vývojáře je News API dostupné zdarma, ovšem s několika omezeními. V kapitole 2 je využíváno News API pro získávání dat ze zpravodajských webů a je v ní nastíněn i přístup k jednotlivým omezením tak, aby pro nás nebyly zásadní.

1.4.2 Sociální sítě

V roce 2020 je nejpopulárnější sociální sítí Facebook s 2,6 miliardami uživatelů, následují: YouTube (2 miliardy), Instagram (1,1 miliardy), Reddit (430 milionů) a Twitter (326 milionů) (Clement, Most popular social networks worldwide as of July 2020, ranked by number of active users, 2020). YouTube není pro naše účely relevantní, mohli bychom analyzovat komentáře pod videi zveřejňované danou společností, ale ty obsahují i názory týkající se kvality videa a nebo i osobní střety lidí. Zároveň ne každá společnost má svůj vlastní YouTube kanál, na kterém pravidelně zveřejňuje videa týkající se nových produktů. Zajímavé by mohlo být zjišťování sentimentu názorových videí (recenzí produktu), ale na tuto problematiku by bylo možné napsat samostatnou diplomovou práci.

Facebook v roce 2012 koupil společnost **Instagram** za 1 miliardu amerických dolarů (Rodriguez, 2019). Nyní nabízí API, ze kterého je možné stahovat vybraná data z obou aplikací. Instagram slouží ke sdílení obrázků a prezentaci životního stylu. Akciové společnosti jej využívají především pro marketingové účely. Můžeme se podívat na instagramový účet: https://www.instagram.com/intel/. Pokud nahlédneme do komentářů pod jednotlivými příspěvky, zjistíme, že vzhledem k jejich množství není problém tyto příspěvky přečíst. Následuje zjištění, že neobsahují téměř žádné užitečné informace. Facebook už při zadávání příspěvku nabádá ke zveřejňování: „Co se vám honí hlavou“. Uživatelé používají příspěvky k prezentaci názorů nejen na aktuální dění ve světě a bylo by vhodné jejich příspěvky obsahující námi sledovaná klíčová slova analyzovat. Facebook pomocí jeho Graph API umožňuje vyhledávání veřejných příspěvků, ale pouze vybraným médiím (Facebook, 2020). Možné je sbírat komentáře u příspěvků na

fanouškovských stránkách jednotlivých společností. Například stránka @Intel má v roce 2020 38 309 561 fanoušků, mezi 11.6 a 11.8 zveřejnila 14 příspěvků s mediánem 14,5 komentářů u každého příspěvku, obdržela průměrně 4,1 komentáře za den. Po zevrubném pročítání komentářů jsme zjistili, že ze všech 266 komentářů vypovídá o sentimentu společnosti pouhých 10.

Reddit je v roce 2020 šestou nejnavštěvovanější webovou stránkou ve Spojených státech amerických a patnáctou nejnavštěvovanější stránkou celosvětově (Alexa, 2020). Na Redditu existuje mnoho komunit (subreddit) ve kterých je možné zveřejňovat příspěvky a další uživatelé je mohou komentovat případně rozhodovat o jejich pořadí v subredditu pomocí pozitivních a negativních hlasů (Reddit, 2020). Existuje mnoho komunit, které může být vhodné sledovat pro naše účely. Největší subreddity zabývající se financemi jsou: r/personalfinance s 14,2 miliony čtenářů, r/wallstreetbets s 1,4 miliony čtenářů, r/investing s 1,1 miliony čtenářů anebo například r/news s 21,5 miliony čtenářů. Subreddit r/wallstreetbets obsahuje příspěvek: Daily Discussion Thread for August 11, 2020, tento příspěvek se objevuje každý den a průměrně obsahuje 20 000 komentářů. Pomocí programovacího jazyku Python je možné získávat informace z Redditu použitím knihovny PRAW.

Twitter slouží k zveřejňování příspěvků s maximálním počtem 280 znaků. Podobně jako na Redditu se na něm shromažďují komunity lidí zajímající se o určitá témata. U příspěvků je vhodné uvádět hashtagy (#) spolu s klíčovým slovem, které definuje příspěvek jako například: #INTC, #amazing. Bez využití programovacího jazyka lze podle hashtagů vyhledávat příspěvky zveřejněné jednotlivými uživateli. Pro získávání informací pomocí programovacích jazyků Twitter připravuje API v2. Z dostupných informací, ale pro naše účely nejspíše nepřinese žádné zásadní změny oproti aktuální verzi. V současnosti existují 3 verze Twitter API: Standard, Premium a Enterprise. Budeme využívat první zmíněnou, jelikož je zdarma. Standardní verze přináší několik omezení, se kterými se budeme muset vyrovnat, jako je možnost vyhledávání maximálně 7 dní starých příspěvků (Twitter, 2020). Pro připojení k API je možné využít dva způsoby autorizace. První je pomocí `AppAuthHandeler`, u kterého je pro připojení nutné zadat API klíč a API tajný klíč. V případě vyhledávání příspěvků umožňuje 450 požadavků za 15 minut, přičemž v každém požadavku je možné získat 100 příspěvků na Twitteru. Dohromady je tedy možné získat 45 000 příspěvků za 15 minut. Druhý je pomocí `OAuthHandeler`, který pro autorizaci požaduje navíc přístupový token a tajný

přístupový token. Tokeny slouží pro autorizaci uživatelského účtu, po tomto způsobu přihlášení je možné například: získat polohu uživatele při zveřejnění příspěvku nebo zveřejňovat vlastní příspěvky. U vyhledávání příspěvků je umožněno pouze 180 požadavků, celkem 18 000 příspěvků za 15 minut (Twitter, 2020). Pro práci s API je pro Python vytvořena knihovna Tweepy (Tweepy, 2020).

1.4.3 Fundamentální data

Předchozí dvě kapitoly se věnovaly jedné z částí fundamentální analýzy, analýze sentimentu různých zdrojů, které bude věnována stěžejní část diplomové práce. Nyní budou představeny možné způsoby získávání ostatních dat, o kterých pojednává kapitola 1.2.1 Fundamentální analýza.

Pokud budeme chtít zjišťovat data firem působících na trhu ve Spojených státech amerických, můžeme si podobně jako u zpravodajských webových stránek vybrat mezi web scrapingem, nejspíše pomocí knihovny Beautiful Soup, a využitím API pro námi zvolený programovací jazyk. Pokud se rozhodneme sami shromažďovat data z internetových stránek, můžeme se zaměřit na portál americké komise pro cenné papíry a burzy (SEC). Na internetové stránce: https://www.sec.gov/edgar.shtml jsou společnosti nuceni se registrovat a zveřejňovat výroční či čtvrtletní zprávy, které jsou na portálu dostupné zdarma. Podobně jako v České republice na webové stránce https://www.justice.cz. Získávání neupravených dat z portálu EDGAR (Electronic Data Gathering, Analysis, and Retrieval systems) je jednoduché. Existuje mnoho knihoven, sloužících k získání potřebných dokumentů a uložení ve formátu TXT. Oblíbené jsou 10–k fillings, obsahující finanční data a další informace o výkonnosti podniku. Pomocí následujícího skriptu je možné získat všechny 10–k výkazy.

```
from secedgar.filings import Filing, FilingType

filings = Filing(cik_lookup='název_akcie',
filing_type = FilingType.FILING_10Q)
filings.save('/kam/se/mají/soubory/uložit')
```

Většina článků se zabývá analýzou sentimentu těchto výkazů. Jako například: (Ashraf, 2017). Získání finančních dat (o zisku, příjmech, zadlužení a podobně) je složitější. SEC má v současnosti standardizovaný formát pro zveřejňování finančních výkazů XBRL. V minulosti nebylo výjimkou setkání se s různými formáty. Získávání dat z portálu justice.cz je na tom ještě hůře, nejsou výjimkou formáty PDF obsahující nekvalitní fotografie finančních výkazů. Standardizace formátu na XBRL je krokem kupředu, ale

pokud se zaměříme na obsah těchto souborů nalezneme u různých firem velké odlišnosti ve struktuře i názvech jednotlivých položek. Tyto odlišnosti velmi komplikují získávání požadovaných finančních dat (Lucey, 2020).

Existuje velké množství API, pomocí nichž je možné získat tato data. Poskytují je například společnosti: Quandl, YahooFinancials, Finnhub, Bloomberg, Reuters, SimFin a Financial Modeling Prep. Všechny zmiňované společnosti poskytují přístup ke kompletním historickým datům, je to ale podmíněno zaplacením poplatků za využití jejich služeb.

Poslední zmiňovaná společnost (Financial Modeling Prep) limituje data poskytovaná zdarma na pět let, respektive 5 kvartálů. Po zaplacení $14 měsíčně, umožní přístup k datům sahajícím třicet a více let do minulosti. Po získání API klíče je možné, pomocí knihovny FundamentalAnalysis, krátkým skriptem uložit požadovaná data z portálu EDGAR do různých formátů, například kompatibilního s programem Microsoft Excel (Financial Modeling Prep, 2021).

```
import FundamentalAnalysis as fa

ticker = "název_akcie"
api_key="klíč_pro_získávání_dat"

income_statement_quarterly = fa.income_statement(ticker,api_key,
                                    period="quarter")
income_statement_quarterly.to_excel("název_souboru.xlsx")
```

1.4.4 Data pro technickou analýzu

Historická data cen akcií se opět převážně nacházejí pod placenou bránou. Především pokud chceme získat data s nízkým intervalem OHLC cen. API jednotlivých poskytovatelů a s nimi spojené knihovny umožnují stažení dat v různých intervalech a periodách pod různými cenami. Dříve bylo možné získat historická data ve vysoké zrnitosti pomocí API Yahoo! Finance nebo Google Finance, tyto API už dnes nefungují. Populární knihovna yfinance, vytvořena z důvodu zrušení oficiálního Yahoo! Finance API, umožňuje získat data cen akcií pomocí tohoto skriptu:

```
import yfinance as yf

msft = yf.Ticker("název_akcie")
data = msft.history(period="Perioda", interval="Interval")
print(data)
```

V následující tabulce je možné vidět periodu a interval dat, které je možné získat pomocí výše ukázaného skriptu.

Tabulka 1: yfinance perioda a interval

Perioda	7d	60d	60d	60d	60d	730d	60d	730d	max
Interval	1m	2m	5m	15m	30m	60m	90m	1h	1d, 5d, 1wk, 1mo, 3mo

Zdroj: Vlastní zpracování

Pokud není perioda a interval dostačující, následující skript umožňuje průběžně stahovat a ukládat data do vlastní, v tomto případě MySQL databáze.

```
import yfinance as yf
from sqlalchemy import create_engine
import mysql.connector
import kody

sqlal = create_engine("mysql+mysqldb://jméno:heslo@adresa")
def stock_data(ticker, perioda, interval_ohlc, table):
    """
    ticker = název akcie
    period = časové okno pro celý graf
    interval_ohlc = OHLC se vytváří po dobu intervalu
    table = tabulka do které se mají data uložit
    """

    cnx = mysql.connector.connect(user="jméno",
                                  password='heslo',
                                  host='localhost',
                                  database='mydb',
                                  charset = 'utf8')
    cursor = cnx.cursor()
    stock = yf.Ticker(ticker)

    cursor.execute("""SELECT Datetime FROM "+ table +"
                      ORDER BY Datetime DESC LIMIT 1""")
    for row in cursor.fetchall():
       result = row
    df = stock.history(period=perioda,
                       interval=interval_ohlc,
                       start=result[0])
    df = df.drop(columns=["Dividends", "Stock Splits"])
    df.to_sql(table, con=sqlal, if_exists="append",
                index=True)

stock_data("AMD", "7d", "1m", "AMD")
```

1.5 Velká data

Analýzy dat, prováděné pro účely diplomové práce, stojí na pomezí mezi analýzou velkých dat (anglicky Big Data) a pouze velkého množství dat (Layton, 2017). Budeme ukládat například příspěvky na sociální síti Twitter (tweety) obsahující určitá klíčová slova definující název společnosti. V případě ukládání tweetů hovořících o jedné společnosti, je možné za jeden měsíc nashromáždit kolem 100 MB dat. Při ukládání příspěvků týkajících se akciového indexu S&P 500 bychom mohli stahovat a ukládat data o všech firmách, které jsou v indexu obsažené. Přibližně by bylo každý měsíc uloženo 50 GB dat. Takto velké množství dat není možné uložit do operační paměti (RAM) běžného počítače, v některých zdrojích tento milník definuje velká data. Po vytvoření skriptu pro ukládání dat nebude problém ho modifikovat pro získávání většího množství. Vzhledem k tomu budeme k datům přistupovat jako k velkým datům. Při zvýšení množství ukládaných dat se problémem stane velikost potřebného diskového úložiště a potřebný výpočetní výkon počítače. Výběr vhodného systému řízení báze dat nám umožní pracovat s daty efektivně, nabízí se dva systémy a to:

- Relační databázové systémy: na trhu se vyskytují již velmi dlouho a jsou standardizované.
- Nerelační databázové systémy: jsou novější, neexistují standardy a pro vývojáře jsou obtížné na naučení.

U obou systémů řízení báze dat musíme při zvýšení množství ukládaných dat počítat s nutností zvýšení výpočetního výkonu. To je možné jak změnou hardwaru na výkonnější anebo, v případě využití hostovaného systému v cloudu (například u Amazon Web Services), zaplacením více peněz za využitý strojový čas. V diplomové práci využíváme relační databázový systém, k jehož výběru přistoupíme v následující kapitole. Nerelační databázový systém by byl obtížnější vytvořit a vzhledem k tomu, že se pohybujeme na rozmezí mezi velkým množstvím dat a velkými daty, budou pro praktickou část dostačující. V případě Rozšíření diplomové práce o analýzu většího množství akcií by mohla nastat situace ve které bude nutné data přenést do nerelačního databázového systému (Tobin, 2020).

1.6 Uchovávání dat

Vzhledem k vybranému programovacímu jazyku Python se jako první vhodný nástroj pro ukládání velkého množství dat nabízí databázový systém SQLite. Není nutné doinstalovat další balíčky, Python tuto knihovnu obsahuje bez nutnosti její dodatečné instalace. SQLite je hojně využíván technologickými společnostmi například v mobilních telefonech, televizorech, kamerách nebo letadlech. Umožňuje ukládání dat ve formátech: *NULL, INTEGER, REAL, TEXT a BLOB* (SQLite, 2020). Velmi populární společností, periodicky zveřejňující hodnocení databázových systémů, je DB-Engines. Metodiku kalkulace skoré je možné nalézt na: https://db-engines.com/en/ranking_definition. V žebříčku hodnotícím relační DBMS neboli RDBMS (Relational Database Management Systems), se v červenci 2020 SQLite umístilo na šestém místě.

RDBMS budeme využívat při práci s daty. Relační databázový systém se dá představit jako tabulka v Excelu, jejíž první řádek obsahuje popisy dat, v následujících řádcích jsou konkrétní data. V terminologii relačních modelů se řádku říká *tuple*, nadpisu sloupce *attribute* a tabulce *entita* (Elmasri & Navathe, 2016). Zápis dat tímto způsobem je vhodný pro naší práci. Můžeme vytvořit entity s atributy jako jsou: *čas, autor, odkaz na web, text článku, vypočtený sentiment.*

SQLite není jediným relačním databázovým systémem. Vzestupně od pátého místa se v hodnocení RDBMS od DB-Engines dále vyskytuje: *IBM Db2, PostgreSQL, Microsoft SQL Server, MySQL* a *Oracle.* Vzhledem k tomu, že řešení od *IBM, Microsoftu* a *Oracle* jsou placená, zužuje se náš výběr na *SQLite, PostgreSQL* a *MySQL*.

1.6.1 Výběr RDBMS

SQLite je jediným z posledních tří jmenovaných databázových systémů, který nepotřebuje pro své fungování server, jedná se o serverless databázi. To přináší mnoho výhod i nevýhod, které závisí na konkrétním způsobu použití databáze. Výhodou je nízká velikost instalace (méně než 600KiB). Dalšími výhodami může být, že SQLite není nutno nakonfigurovat před spuštěním databáze a možnost jednoduchého přenášení souboru s daty. SQLite oproti ostatním systémům ukládá data pouze do jednoho, snadno dostupného souboru. Nevýhody SQLite mohou být: nemožnost správy uživatelů a jejich oprávnění a nutnost vlastního řešení pro zabezpečení databáze (Ostezer & Drake, 2019).

PostgreSQL se sám proklamuje jako nejvíce pokročilá open source relační databází (PostgreSQL Global Development Group, 2020). Dle oficiální dokumentace podporuje 160 funkcí standardu SQL z celkového počtu 179. Podporuje souběžný zápis a ze zkoumaných databázových systémů také největší počet formátů ukládaných dat. Vhodný je při nutnosti udržení integrity dat a použití komplexních příkazů. Mezi nevýhody patří: složitá konfigurace, vyšší nárok na paměť v případě připojení více klientů a nižší rychlost čtení (Ostezer & Drake, 2019).

MySQL byla vytvořena za účelem dosažení rychlosti a spolehlivosti. Tento databázový systém je využíván společnostmi: Twitter, Facebook, Netflix a Spotify. Není ovšem tak komplexní jako PostgreSQL. Pro mnohé aplikace obsahuje dostatečné množství funkcí ze standardu SQL. Nabízí možnost zabezpečení na různých úrovních a je vhodný, pokud očekáváme budoucí škálování systému. Nevýhodou může být licence pod GPLv2, některé funkce MySQL jsou přístupné až v placené verzi jako například technická podpora a některé doplňky (Ostezer & Drake, 2019).

Všechny tři popsané RDBMS by byly vhodné pro řešení ukládání dat. Při současném zápisu více procesy si musíme dávat pozor na uzamčení, dokud první proces nedokončí vkládání dat do tabulky, pro ostatní je uzamčena a čekají na dokončení prvního procesu. Vybraným databázovým systémem je řešení od MySQL. Jedná se o střední cestu v komplikovanosti nastavení a práce s databází. Vzhledem k jeho rozšířenosti existuje široké spektrum informací a podpory na různých internetových stránkách, například: https://stackowerflow.com.

1.7 Předzpracování dat

Získaná data mohou obsahovat nežádoucí informace, nebo být neúplná. Před samotnou analýzou je nutné, v nejlepším případě již před samotným uložením do databáze, upravit data do vhodné podoby (Han, Kamber, & Pei, 2012).

1.7.1 Odstranění nežádoucích informací

Zejména u dat, které nelze získat zpětně, je nutné předem pečlivě rozmyslet, zdali daná data budeme ukládat do databáze a jaká můžeme odstranit (Shmueli, Bruce, Gedeck, & Patel, 2020). Například u ukládaných tweetů je možné, spolu s textem tweetu, uložit velké množství informací (metadat) jako jsou: informace o autorovi (*datum a čas založení účtu, počet oblíbených příspěvků, počet sledujících, počet přátel, jméno, počet tweetů anebo*

zdali má uživatel povolenou geolokaci), informace o tweetu (*datum a čas tweetnutí, geolokace, jazyk, množství sdílení a oblíbených u tweetu, přístroj, na kterém byl zveřejněn, nebo URL*). Z těchto informací je možné ukládat všechny, nebo jen některé.

1.7.2 Oprava chybějících částí dat

Při ukládání do MySQL je nutné zkontrolovat nastavení tabulky, pokud nenastavíme výchozí hodnotu, nebude možné uložit do databáze celý řádek. Zároveň dojde k zastavení skriptu ukládajícího informace a bude ukončen zobrazením chybové hlášky.

1.8 Analýza sentimentu

V předchozích kapitolách se již několikrát objevilo slovo sentiment, přičemž při první zmínce (v kapitole 1.1.1) bylo definováno jako názor lidí na danou společnost. Sentiment se může měnit v průběhu života. Například v případě, že navštívíme novou restauraci a po objednání obdržíme vynikající jídlo, budeme mít v danou chvíli pozitivní sentiment. Když, ale při další návštěvě obdržíme jídlo, které není dobře uvařeno, změní se náš sentiment na negativní. I restaurace se snaží o získávání dat sentimentu, mnoho z nich má účet na https://www.google.com, pod který je lidem umožněno psát zpětnou vazbu. Zaměstnanci či majitelé jsou schopni přečíst si a zareagovat na recenze osobně. V našem případě budeme, pro zjišťování sentimentu u milionů dat, muset využít výpočetní techniky (Lane, Howard, & Hapke, 2019).

1.8.1 Hodnota sentimentu

Často se setkáme s knihovnami, u kterých se vypočtený sentiment pohybuje v rozmezí od -1 do 1 přičemž: *záporné hodnoty* značí negativní sentiment (například: "Pomalá obsluha, drahé a nedovařené jídlo."), *kladné hodnoty* pozitivní sentiment (například: "Vynikající ceny i pokrmy.") a *nulové hodnoty* neutrální sentiment (například: "Rychlost obsluhy srovnatelná s konkurencí."). Těmto hodnotám se říká *polarita (polarity)*, možné je získávat pouze *pozitivitu (positivity)* či *negativitu (negativity)* příspěvku. V našem případě budeme využívat jedné, *sloučené (compound)* hodnoty. Jako další můžeme u sentimentu sledovat také *subjektivitu (subjectivity)* příspěvku která se obvykle pohybuje v rozmezí od 0 (objektivní příspěvek) do 1 (subjektivní příspěvek) (Lane, Howard, & Hapke, 2019).

1.8.2 Slovníky sentimentu

Základním stavebním kamenem pro analýzu sentimentu textu jsou slovníky sentimentu. Je nutné vytvořit datové řady obsahující slova, nebo fráze a jejich pozitivitu či negativitu. Existuje velká řada již existujících slovníků: LIWC (Linguistic Inquiry and Word Count), ANEW (Affective Norms for English words), GI (General Inquirer), Twitter US Airline Sentiment, Lexicoder, Sentiment140 a desítky dalších. Druhou možností je vytvoření vlastních slovníků. Jedná se o velmi náročnou činnost. Můžeme k ní například přistoupit tak, že posbíráme velké množství tweetů a požádáme respondenty, pomocí online dotazníku, o přečtení a vyhodnocení sentimentu jejich obsahu. Tímto způsobem vznikl například slovník Twitter US Airline Sentiment. Další možností je využití strojového učení. Vychází z dat obsahující klasifikaci sentimentu, získanou od respondentů a následně se snaží klasifikovat data bez určeného sentimentu. Pro všechny popsané případy existují knihovny, pomocí nichž je možné analyzovat sentiment. Vybrané z nich, pro programovací jazyk Python, si nyní popíšeme.

VaderSentiment je, stejně jako ostatní popisované nástroje pro analýzu sentimentu, dostupný pod MIT (Massachusetts Institute of Technology) licencí. Programy nabízené pod touto licencí mají nejméně omezení pro využití, označují se jako open-sourced neboli otevřený zdroj (Goldstein, 2019). VaderSentiment je možné nalézt na adrese: https://github.com/cjhutto. Vychází ze slovníků: LIWC, ANEW a GI. Rozšiřuje je o emotikony, zkratky a slangové výrazy. Celý slovník se 7 520 výrazy je dostupný na GitHub stránce autora v souboru /vaderSentiment/vaderSentiment/vader_lexicon.txt. Pokud bychom chtěli přidat svá vlastní slova do slovníku, autor C. J. Hutto doporučuje mít pro každé slovo hodnocení jeho sentimentu od deseti nezávislých osob (Hutto, 2021). Na následujícím skriptu je vidět způsob využití této knihovny na konkrétních příkladech vět.

```
from vaderSentiment.vaderSentiment import
                                        SentimentIntensityAnalyzer

věty = ["The tea is horrible.",
        "The tea is ok.",
        "The tea was awesome"]
for x in věty:
    vader = SentimentIntensityAnalyzer().polarity_scores(x)
    print(x,"Sentiment= ",vader["compound"])
```

Výsledky po spuštění příkazu:

```
The tea is horrible. Sentiment = -0.5423
The tea is ok. Sentiment = 0.0
The tea was awesome. Sentiment = 0.6249
```

Můžeme vidět, že věta „Čaj je hrozný.“ získala negativní hodnotu sentimentu, „Čaj je ok.“ neutrální a „Čaj byl úžasný.“ pozitivní hodnotu sentimentu.

TextBlob. Tvůrcem je Steven Loria: https://github.com/sloria. Oproti předchozí knihovně obsahuje také modely sloužící pro klasifikaci pomocí strojového učení. Model Naive Bayes využívá předem ohodnocené věty, jedná se tedy o učení s učitelem (supervised learning). Klasicky se využívá filmových recenzí, které u každého slovního vyhodnocení filmu obsahují hodnocení pomocí hvězdiček. Pokud shromáždíme velké množství recenzí, hvězdičky se mohou využít k předvídání sentimentu jednotlivých slov, například: pokud hodnocení obsahuje 0 hvězdiček, jedná se o velmi negativní recenzi. TextBlob z těchto recenzí vytvoří slovník jednotlivých slov a jejich sentimentu, podobně jako předem zmíněný slovník u knihovny vaderSentiment (Lane, Howard, & Hapke, 2019). Na GitHub stránce autora, v souboru TextBlob/textblob/en/en-sentiment.xml můžeme nalézt slovník obsahující 2917 slov spolu s jejich významem. Pomocí následujícího skriptu je možné tento slovník využít k analýze sentimentu stejných vět jako u knihovny vaderSentiment.

```
from textblob import TextBlob

věty = ["The tea is disgusting.",
        "The tea is ok.",
        "The tea was awesome."]
for x in věty:
    tb = TextBlob(x).polarity
    print(x,"Sentiment= ", tb)
```

Výsledky po spuštění příkazu:

```
The tea is disgusting. Sentiment=  -1.0
The tea is ok. Sentiment=  0.5
The tea was awesome. Sentiment=  1.0
```

Můžeme vidět odlišnosti oproti knihovně vaderSentiment. Je velmi obtížné, ne-li nemožné vytvořit slovník, který bude sentiment objektivně vyjadřovat pomocí číselné hodnoty. Pokud bychom se zeptali několika lidí na číselné vyjádření sentimentu jedné věty, budou také odlišné.

SpaCy je dnes velmi populární knihovnou. Zatímco předchozí dvě přistupovali k měření sentimentu pomocí slovníků, strojového učení a lineárních vztahů mezi jednotlivými

slovy ve větách. Pomocí spaCy je možné využít k měření sentimentu nelineární vztahy a umělé neuronové sítě. Nejen deep learning je možné provádět i pomocí grafických karet Nvidia. SpaCy dokáže využít kromě procesoru také paralelní výpočetní platformu a aplikační programovací rozhraní CUDA (SpaCy, 2021). Pro naučení slovníku je nutné disponovat velkým množstvím ohodnocených slov, frází, vět, nebo celých dokumentů. Modely pomocí spaCy je možné natrénovat například na recenzích produktů na online obchodu Amazon a podobně.

1.9 Backtesting

Zpětné testování obchodních transakcí neboli backtesting slouží k otestování metod predikce obchodních signálů a obchodních strategií a jejich výkonnosti na minulých datech. Můžeme testovat výnosy generované strategií po změně různých podmínek jako je: časové okno pro data, granularita technických a fundamentálních dat nebo změna aktiva (Davda, 2021). Před výběrem konkrétního softwaru je nutné popsat několik termínů, týkajících se problematiky backtestingu.

1.9.1 Obchodní pokyny

Pro zahájení a ukončení obchodu je možné využít širokou škálu příkazů pro realizaci obchodní strategie. V seznamu níže jsou uvedeny anglické názvy, budeme je v textu dále využívat, jelikož jsou důrazné a v krátkosti interpretují podstatu pokynu.

- Market Order: obchod je uskutečněn téměř okamžitě po jeho zadání do systému. Pokud se okamžitě nenajde protistrana a dojde ke změně ceny, obchod se uskuteční i za pro nás nevýhodnou, nebo naopak výhodnou cenu.
- Limit Order: obchod je uskutečněn za přesně stanovenou cenu, nebo pro nás výhodnější. Nevýhodou může být, že obchod nemusí být nikdy uskutečněn.
- Stop Order: pro obchod stanovíme cenu, za kterou se uskuteční. Pokud se cena aktiva dostane na námi stanovenou hodnotu, obchodní pokyn stop order začne fungovat stejně jako market order.
- Stop-Limit Order: pokud se cena dostane na námi stanovenou, stane se z obchodního pokynu limit order.
- Trailing Stop Order: slouží pro stanovení variabilní ceny, při které je stop order a stop-limit order uskutečněn (Milton, 2020).

Často se setkáme s názvy stop-loss a take-profit. Stop-loss zajišťuje maximální ztrátu z obchodu a většinou se jedná o stop order, jelikož je většinou v našem zájmu obchod ukončit okamžitě. Take-profit určuje hodnotu zisku, při které dojde k ukončení obchodu a využívá obchodní pokyn limit order.

1.9.2 Transakční náklady

Uskutečnění obchodního pokynu s sebou nese různé náklady, které jsou snadno opomenutelné při testování obchodní strategie, ale na výsledky backtestingu mohou mít velký dopad:

- Provize a poplatky: makléř, umožňující nám přístup k trhu a vytváření obchodních pokynů poskytuje své služby nejčastěji formou poplatků za uskutečnění obchodu, držení aktiva nebo přístup k obchodní aplikaci.
- Skluz: doba mezi zadáním obchodního pokynu a jeho uskutečněním
- Likvidita: u aktiv s nízkou volatilitou může u HFT vzniknout problém s neuskutečněním obchodu vzhledem k nízkému množství nabízejících, nebo poptávajících.
- Spread: rozdíl mezi cenou nabídky a poptávky, často v sobě obsahuje i poplatky makléře za uskutečnění obchodu (QuantStart, 2021).

Při tvorbě strategie je možné transakční náklady implementovat jako:

- Fixní: poměrně přesně reprezentuje provize a poplatky placené u makléře při uskutečnění obchodu. Stanovujeme neměnící se hodnotu rozdílu mezi nabídkou a poptávkou a zanedbáváme skluz a likviditu.
- Lineární a kvadratické: lineární modely předpokládají fixní spread. V případě zvyšování nakupovaného, nebo prodávaného množství se nelineárně mění transakční náklady, proto je nejvhodnější využívat modely kvadratické. Jedná se o modely velmi obsáhlé a více informací je možné nalézt například ve zdroji (Chen, Lezmi, Roncalli, & Xu, 2019). Aplikace modelu s kvadratickým výpočtem transakčních nákladů by byla vhodná zejména u vysokofrekvenčního obchodování. U velkého množství obchodů se výrazně projevují jednotlivé náklady.

1.9.3 Kognitivní zkreslení

Nejen při testování strategií se člověk při získávání a zpracování informací potýká se slepými místy, které brání racionálnímu uvažování a vytváření objektivního názoru. Přehledně zobrazuje více než 180 kognitivních zkreslení Cognitive Bias Codex (Heick, 2020). My si představíme vybraná zkreslení, kterých se člověk může dopustit u problematiky backtestingu:

- testování na malém vzorku dat a formování konečných závěrů,
- pohlížení do budoucnosti a upravování strategie podle budoucích dat, která nebudou v případě implementace reálného obchodování předem dostupná,
- implementování pouze vybraných dat, která potvrzují naši hypotézu,
- využívání snadno(zdarma) dostupných dat, která nemusí být přesná,
- předpokládání, že neměnná strategie bude fungovat stejně na všech aktivech,
- zanedbání transakčních nákladů (Formula Stocks, 2017),
- vysoká optimalizace strategie dokonale fungující na jednom aktivu,
- zanedbání našeho vlivu na volatilitu (Liew, 2021).

1.9.4 Software

Pro zpětné testování existuje široký výběr softwaru. Při výběru záleží na osobních preferencích a finančních možnostech. Mezi nejznámější, které ihned, z důvodu vysoké ceny, opomeneme, patří Bloomberg terminál a Refinitiv Eikon (dříve Thomson Reuters Eikon). Další populární a dlouhou dobu existující software dostupný zdarma je **MetaTrader**, který pro backtesting využívá jazyk MQL5. **Zorro** je dle jejich vlastních internetových stránek ekosystém pro testování obchodních strategií. K optimalizaci parametrů používaných strategií umožňuje použít deep learning. Nabízí grafické rozhraní a podporuje programovací jazyky R a Python. **Quantopian** byla velmi propagovaná a populární online platforma, nabízející zdarma data a nástroje pro backtesting využitím programovacího jazyku Python (Konstantinovic, 2020). Jako alternativa pro Quantopian se nabízí **QuantConnect** s podobnými funkcemi podporující jazyky Python, C# a F#. Nevýhodou při využívání předpřipravených nástrojů a softwaru může být neznalost přesného fungování jednotlivých komponentů a následný vznik chyb při testování strategie.

Pro účely diplomové práce budeme využívat knihovny podporující programovací jazyk Python. Pro zpětné testování jich existuje celá řada, její výběr je subjektivní a závisí na požadované funkcionalitě nebo programovacích schopnostech uživatele. Několik z nich si nyní představíme:

- FinTA: není přímo nástroj pro backtesting, ale lze pomocí něj do strategie implementovat přes 80 technických indikátorů bez nutnosti jejich kompletního výpočtu.
- Ta-lib: populárnější než FinTa a umožňuje vypočítat více než 150 indikátorů.
- Zipline: byla základním stavebním kamenem pro Quantopian. Na portálu GitHub se mezi knihovnami pro backtesting pyšní zřejmě nejvyšší oblíbeností s 13 700 hvězdičkami. První zveřejnění bylo dne 17.2. 2013.
- Backtrader: pro srovnání obdržel 6 100 hvězdiček. Nabízí podobnou funkcionalitu jako Zipline. Na rozdíl od ní umožňuje snadný přechod z backtestingu na reálné obchodování a je novější knihovnou s prvním zveřejněním 10.1. 2015.
- TensorTrade: umožňuje při formulování strategie využívat strojové učení. knihovna je stále v beta verzi, v našem přehledu se jedná o nejnovější, zveřejněnou dne 28.7. 2019.

1.9.5 Formulace a testování strategie

Vzhledem ke zkušenostem autora budeme pro zpětné testování využívat knihovnu Backtrader. Umožňuje výpočet vybraných indikátorů obsažených v knihovně Ta-lib. Všechny Python knihovny je možné zakomponovat do backtestingu. Samozřejmě je možné obejít se i bez nich a veškeré skripty pro testování obchodní strategie vytvořit. K tomu by ovšem bylo zapotřebí větší skupiny osob. Z tohoto důvodu se hojně využívají knihovny.

V internetových zdrojích je možné najít velké množství příkladů strategií vytvořených a testovaných pomocí vybrané knihovny. Třída Cerebro je základní blok knihovny Backtrader. Slouží k nastavení zdrojových dat, strategií, protokolování, spuštění testování, nástrojů sloužící k analýze výsledků testované strategie a zobrazování výstupů pomocí grafů. Konkrétní způsob sestavení skriptů pro zpětné testování pomocí knihovny Backtrader bude možné nalézt v praktické části.

Představíme si další knihovny, které jsou vhodným doplňkem při backtestingu:

- Datetime: napomáhá pro práci s časovým obdobím v různých formátech.
- Pandas: nástroje pro práci se strukturovanými daty a jejich analýzu. Umožňuje zobrazovat data ve formě pandas.DataFrame, který je podobný tabulce v programu Microsoft Excel.
- Numpy: slouží pro provádění vědeckých výpočtů. V pandas.DataFrame umožňuje například výpočet kvadratické interpolace dat.
- Pyfolio: slouží k analýze risku a výkonnosti strategie.
- Quantstats: relativně nová knihovna zveřejněna 1.5. 2019. Může navazovat na Pyfolio a vytvářet přehledné zprávy a vizualizace výsledků strategie.
- Tensorflow: mohutná knihovna pro využívání strojového učení.
- Keras: je postaven na Tensorflow a poskytuje relativně jednoduché API pro vytváření neuronových sítí a využívání deep learningu (Terra, 2021).

Jako poslední v kapitole o backtestingu musíme zmínit benchmarking. Po otestování strategie a zjištění výsledků je vhodné porovnat výsledky. Ke srovnání se často využívá strategie Buy and Hold při které se nakoupí nejčastěji akciový index pouze jednou, a je držen stejné časové období, ve kterém testujeme námi vytvořenou strategii. Další možností je testování oproti náhodné strategii s podobnými parametry (Longmore, 2015).

1.10 Webová aplikace

Pro podporu rozhodování při diskrečním obchodování existuje velká řada aplikací, ať už desktopových, mobilních, nebo webových. Nabízí různé úrovně funkcionality, od informativní, po aplikace umožňující zadávání obchodních pokynů. Nejdříve se zaměříme na funkcionalitu existujících a následně identifikujeme nástroje využitelné pro vytvoření vlastní aplikace.

1.10.1 Existující webové aplikace

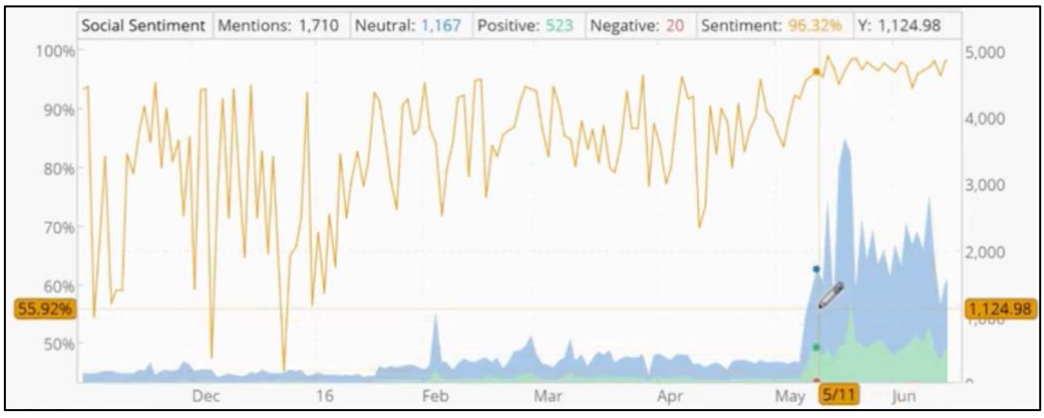
Mezi nejznámější aplikace propojené s makléřem, poskytující fundamentální informace a nástroje pro technickou analýzu patří například:

- Bloomberg terminál: je aplikace nabízející téměř veškerou představitelnou funkcionalitu, kterou zpřístupní po zaplacení ročního poplatku blížícího se k 500 000Kč. Uživatel má přístup k velkému množství nástrojů pro technickou

analýzu. Pro analýzu sentimentu poskytuje obsáhlé datové struktury. Bloomberg implementoval umělou inteligenci a strojové učení podobným způsobem, který popisujeme v kapitole 1.7, získává ohodnocené fundamentální zprávy a pomocí nich trénuje modely, které bez dozoru hodnotí sentiment nových zpráv a příspěvků na sociálních sítích (Bloomberg Finance L.P., 2021).

- Refinitiv Eikon: je podobně cenově dostupný, respektive nedostupný, jako Bloomberg terminál, kterému je v mnoha ohledech dobrým konkurentem. Pro analýzu sentimentu poskytuje velké množství dat. Ty už ale počítačově neanalyzuje, pouze k nim umožňuje přístup pomocí API. Zajímavostí je, že Refinitiv na svých internetových stránkách využívá pro analýzu sentimentu knihovny TextBlob (Ramchandani, 2019).
- Thinkorswim: platforma byla odkoupena společností TD Ameritrade, po vytvoření účtu u tohoto makléře je nabízena zdarma. V současné době není dostupná pro Evropany. Nabízí široké možnosti pro technickou analýzu. Pro fundamentální analýzu poskytuje informace ze čtvrtletních výkazů akciových společností (Thinkorswim, 2021). Pro analýzu sentimentu nabízí technické indikátory zobrazující „Social Sentiment“, umožňuje vizualizovat množství zpráv zveřejněných na sociálních sítích a jejich sentiment, a to pouze prostřednictvím desktopové aplikace. Neumožňuje zobrazení obsahu konkrétních příspěvků a hodnot jejich sentimentu (Thinkorswim, 2021). Na obrázku níže je vidět zobrazení sentimentu aplikací Thinkorswim.

Obrázek 1: Thinkorswim Social Sentiment



Zdroj: (Fitton, 2016)

Na pravé ose je zobrazen celkový počet příspěvků na sociálních sítích za daný časový interval. Nad grafem je v absolutním vyjádření množství neutrálních,

pozitivních a negativních příspěvků. Levá osa vyobrazuje relativní hodnotu sentimentu. Aplikace vypočítává „Sentiment“ jako poměr mezi negativními a pozitivními příspěvky, z výpočtu jsou vynechány příspěvky neutrální.

Další aplikace dostupné pro Evropany přímo napojené na brokera, s nástroji pro fundamentální a technickou analýzu:

- SaxoTrader: je platforma od společnosti Saxo bank. Poskytuje přístup k obchodování s více než 19 000 akciemi. Vyznačuje se relativně komplikovanou strukturou poplatků, například po 180 dnech neaktivity požaduje $100 a za každý měsíc minimálně $5 za otevřené obchody. Pro technickou analýzu nabízí podobnou paletu nástrojů a indikátorů jako další platformy zmíněné v této sekci. Pro fundamentální analýzu nabízí přehled výročních a čtvrtletních zpráv, neposkytuje přehled o sentimentu, jedná se o první zmíněnou platformu, kterou je možné vyzkoušet bez otevření účtu na internetové stránce: https://www.saxotrader.com/sim/instant-demo/InstantDemo-EN-GL/.
- Následující aplikace nabízí podobnou funkcionalitu nástrojů pro technickou analýzu. Pro fundamentální analýzu většinou nabízí přístup ke strukturovaným čtvrtletním a výročním zprávám a vybranému malému množství dalších fundamentálních dat. Žádná z nich nevěnuje pozornost analýze sentimentu. Struktura poplatků není složitá, všechny zmíněné nabízí platformu po vytvoření účtu zdarma:
    - E*Trade: umožňuje obchodovat se 6 000 akciemi (Carey, 2021).
    - Currency.com: umožňuje obchodovat se 2 000 akciemi (Currency.com, 2021).
    - XTB: umožňuje obchodovat s 1 700 akciemi (XTB, 2021).
    - eToro: umožňuje obchodovat s 800 akciemi (eToro, 2021).

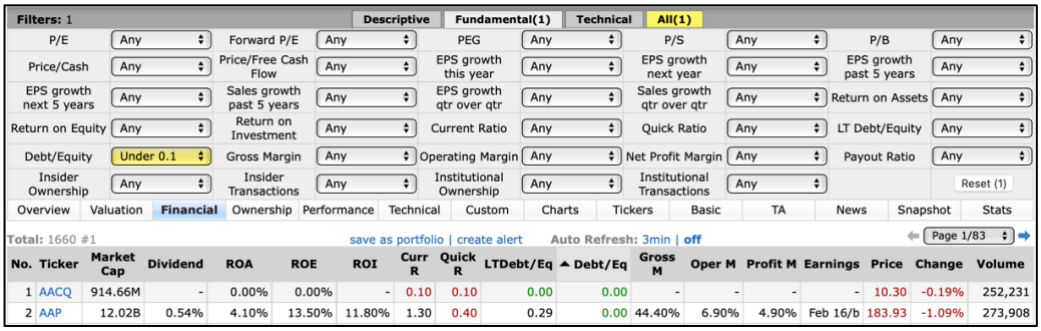
Aplikace dostupné zdarma bez nutnosti vytvoření účtu u makléře:

- TradingView: je využíván především pro technickou analýzu a nabízí pro ni širokou paletu indikátorů a dalších nástrojů. Zdarma umožňuje graficky zobrazit více než 100 ukazatelů vycházejících z výročních a čtvrtletních zpráv a to pouze 8 čtvrtletí zpět. Po zaplacení minimálně $14,95 zobrazí starší data (TradingView, 2021). Při vytváření vlastní stránky můžeme využít widgety aktuální ceny, grafů,

fundamentálního souhrnu, jejichž HTML kód je možné generovat na stránce: https://www.tradingview.com/widget/.

- FinViz: umožnuje filtrování akcií dle různých fundamentálních (z výročních nebo čtvrtletních zpráv) a technických kritérií, zpětně do tří let. Za $24,96 měsíčně umožňuje filtrování zpětně 8 let, exportování dat a nástroj pro backtesting pomocí technických indikátorů (FinViz, 2021). Na obrázku níže jsou zobrazeny možnosti pro filtrování akcií pomocí fundamentálních kritérií, konkrétně zadluženosti vůči vlastnímu kapitálu.

Obrázek 2: FinViz screening



Zdroj: (FinViz, 2021)

- Mezi další úzce zaměřené aplikace patří:
    - Sentiment Viz: pro vizualizaci sentimentu příspěvků na sociální síti Twitter, dle množství dat řádově do několika minut zpátky v čase (Sentiment Viz, 2021).
    - Sentiment Trader: pro grafickou vizualizaci sentimentu za poplatek od $39 měsíčně (Sentiment Trader, 2021).
    - Powrbot: pro získání základních fundamentálních dat o vybrané společnosti, ve formě .CSV souboru, za cenu dosahující až ke $249 (Powrbot, 2021).
    - Yahoo Finance a Google Finance: nabízí podobnou funkcionalitu, pro analýzu čtvrtletních a výročních zpráv. Zobrazení dat v širokém časovém horizontu je možné pouze u Yahoo Finance po zaplacení měsíčního poplatku $35 (Yahoo Finance, 2021).

1.10.2 Nástroje pro tvorbu vlastní webové aplikace

Při tvorbě vlastní webové aplikace je nutné rozhodnout o využívaném programovacím jazyku. V našem případě se rozhodneme pro v současné době velmi populární Python.

Na velmi oblíbeném serveru pro hostování zdrojových kódů, který má v současné době přibližně 62 milionů uživatelů (toto číslo je možné nalézt po zadání výrazu *type:user* do vyhledávacího pole na stránce https://github.com), byl v roce 2020 druhým nejoblíbenějším programovacím jazykem. Na prvním místě se umístil JavaScript (Choudhury, 2020). Python nabízí velké množství nástrojů pro analýzu a vizualizaci dat. YouTube je téměř zcela vytvořen pomocí tohoto jazyka. Mezi společnosti využívající Python patří také: Amazon, Google, Facebook, Dropbox, Netflix, Quora, Spotify a Reddit (Ozimek, 2018).

Pro Python existuje velká řada knihoven, které je možné využít při budování systémů pro back end (serverovou stranu) a front end (klientskou stranu). Z velké části bude záležet na osobních preferencích. Mezi nejpopulárnější aplikační rámce pro tvorbu webových aplikací (web framework) patří:

- Django: je komplexní nástroj pro back end i front end. Umožňuje implementovat přihlašovací dialogy, směrování URL adres, šablony a přenášet data z databáze do objektů využívaných ve webové aplikaci (Petlovana, 2020). Je využíván například společností YouTube, DropBox, Mozilla, Prezi a mnohými dalších (Korsun, 2021).
- Flask: je microframework. Na rozdíl od Djanga obsahuje pouze základní funkcionalitu. Slouží jako základní stavební kámen při tvorbě webové aplikace. Pro doplnění jeho funkcionality je možné využít celou řadu dostupných rozšíření, nebo vytvořit skripty samostatně (Petlovana, 2020).

Knihoven pro vizualizaci dat je mnoho a opět záleží na subjektivních preferencích. Mezi ty hojně využívané patří: Plotly, Matplotlib, Bokeh a Seaborn. Pro naše účely je možné kombinací nástrojů například: Django a Bokeh vytvořit webovou aplikaci sloužící k vizualizaci námi vybraných dat (McMahon, 2019).

V našem případě se rozhodneme pro využití frameworku pro vytváření analytických webových aplikací s názvem Dash. Jedná se o nadstavbu pro Flask a mezi hlavní výhody můžeme zařadit velké množství podporovaných komponentů, například:

- Callback: slouží k zavolání Python funkce, a to buď automaticky, nebo uživatelem. Pomocí callbacků je možné vytvářet interaktivní grafy a tabulky, aktualizovat hodnoty jednotlivých komponentů a mnoho dalších. Pokud bychom

se snažili o vytvoření funkcionality callbacků, jednalo by se o velmi komplikovanou činnost.

- Core components: je celá řada komponentů od checklistů, rozbalovacích seznamů, pole pro zadání textu, až po grafy a stav jejich načítání.
- HTML components: pro vytváření vizuálního vzhledu stránky je nutná alespoň částečná znalost programovacího jazyka HTML. Pomocí těchto komponentů je možné vytvářet nadpisy, tlačítka a odstavce. Po jejich přidání je možné aplikovat na ně různé styly pomocí CSS.
- Data Table: slouží pro vytváření tabulek. Všechny komponenty je možné nalézt na stránce: https://dash.plotly.com.

2 Formulace problému a popis jeho řešení

Kapitola je rozdělena do tří částí, vycházeních z dílčích cílů práce. Jako první se budeme zabývat získáváním a zpracováním dat, které budeme využívat u obou systémů. Následně je v kapitole 2.2 popsán způsob tvorby systému pro zpětné testování obchodních strategií a v kapitole 2.3 způsob řešení interaktivní webové aplikace. Na začátku každé části je popsán konkrétní přínos diplomové práce, v čem je zvolený postup nový, proč se nepoužijí postupy z literatury a jsou představeny vytvořené skripty. Jednotlivé skripty jsou zaneseny do blokového schéma. Následující části obsahují relativně velké množství zásadních částí vytvořených skriptů a odkazy na ně. Dle potřeby není nutné věnovat pozornost úryvkům kódů, základní podstata by měla být zřejmá i bez jejich studování. Důležité jsou interakce mezi jednotlivými skripty, a proto se jejich názvy v textu nezřídka objevují. Čtenář má vždy možnost vrátit se k blokovému schéma a zorientovat se v popisované problematice.

2.1 Získávání, uchovávání a zpracování dat

Kapitola 2.1.1 popisuje programovací jazyk využívaný při práci s daty uloženými v MySQL databázi. Další kapitoly jsou rozděleny podle zdrojů, ze kterých jsou data získávána. Bylo by možné nalézt existující skripty pro získávání a uchovávání dat a z řady z nich budeme také vycházet. Přínosem vypracování této části je získání dat v námi požadované kvalitě, možnost rozhodnout o tom jaká data a metadata budeme ukládat anebo možnost rozšíření skriptů o pro nás důležité funkce. Pro účely diplomové práce budeme získávat data o společnostech:

- Airbus (AIR)
- Boeing (BA)
- AMC Entertainment Holdings (AMC)
- AstraZeneca (AZN)
- Pfizer (PFE)
- Ford Motor (F)
- Ferrari (RACE)
- Toyota (TM)
- Cloudflare (NET)
- Oracle (ORCL)

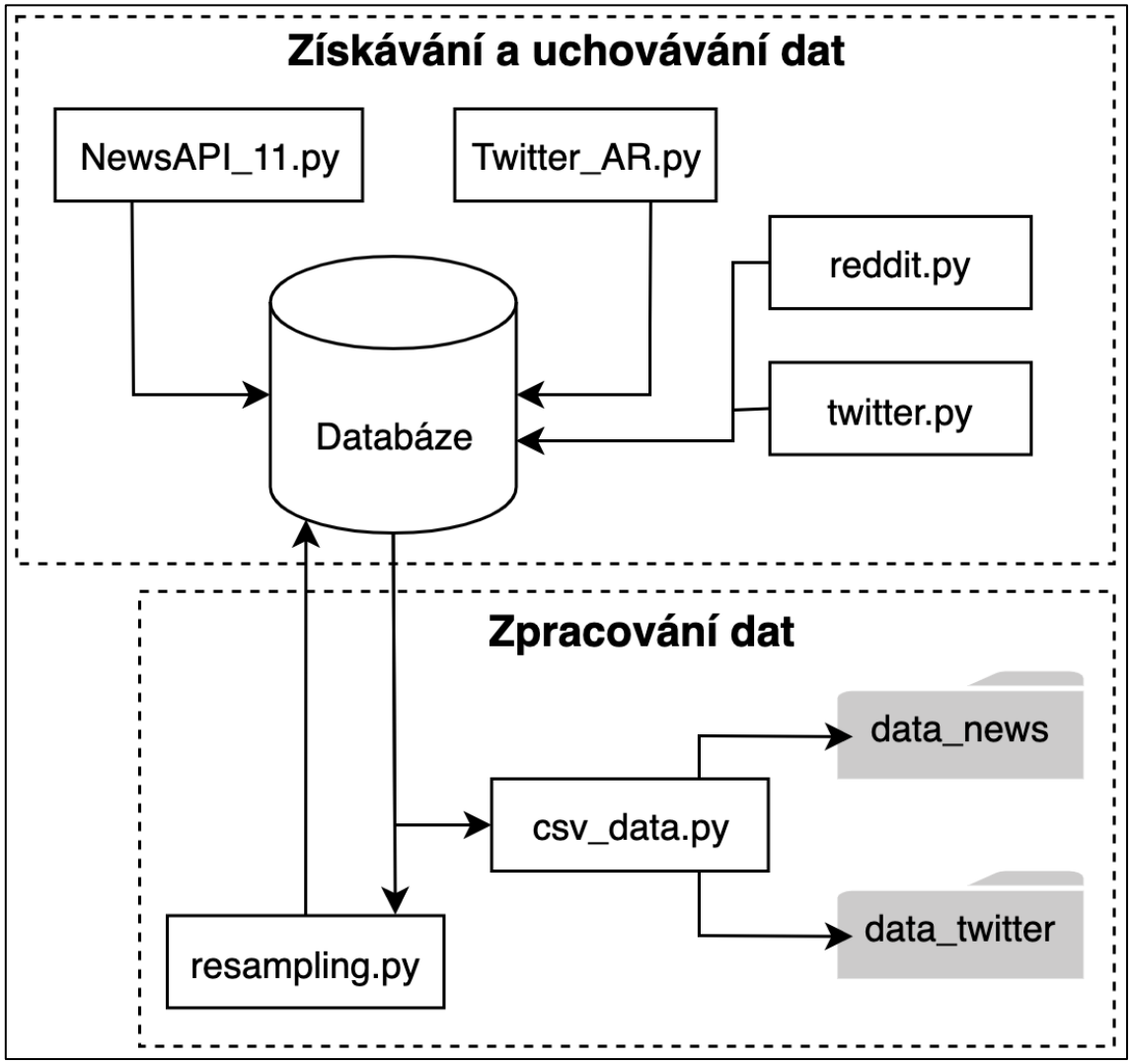
AIR, BA jsou výrobci letadel. AZN, PFE farmaceutické společnosti. F, RACE, TM výrobci automobilů. NET, ORCL jsou technologické společnosti. AMC je v tomto roce (2021) velmi volatilní akcií která se těší velkému zájmu. Snažili jsme se vytvořit diverzifikované portfolio vhodné zejména pro zpětné testování obchodních strategií.

Kompletní skripty je možné nalézt na adrese: https://github.com/HSTEP/FDSS. Pro získávání a zpracování dat využíváme skripty:

- **NewsAPI_11.py**: pro získání a uložení dat ze zpravodajských webů,
- **Twitter_AR.py**: pro získání a uložení tweetů pomocí Academic Research API,
- **reddit.py**: pro získání a uložení dat ze sociální sítě Reddit,
- **twitter.py**: pro získávání a zpracování tweetů pomocí knihovny Tweepy,
- **csv_data.py**: slouží k uložení ceny jednotlivých akcií a hodnoty sentimentu do souborů ve formátu CSV.
- **resampling.py:** snižuje množství dat pro možnost jejich zobrazení webovou aplikací.

Na následujícím obrázku jsou pomocí blokového schéma zobrazeny jednotlivé skripty využívané při získávání, uchovávání a zpracování dat. Můžeme vidět, že skripty ***NewsAPI_11.py***, ***Twitter_AR.py, twitter.py*** a ***reddit.py*** ukládají získaná data do databáze. Pro backtesting jsou data zpracována pomocí skriptu ***csv_data.py***, který generuje CSV soubory v podobě vyžadované knihovnou Backtrader.

Obrázek 3: Schéma získávání, uchovávání a zpracování dat



Zdroj: Vlastní zpracování

2.1.1 Programovací jazyk SQL

V kapitole 1.6.1 Výběr RDBMS byl několikrát zmíněn soulad se standardem SQL a jeho 179 funkcemi. Pro práci se všemi zmíněnými databázovými systémy je využíván standardizovaný programovací jazyk SQL. Příkazy je možné zadávat přes příkazový řádek (CLI). Druhou možností je využít aplikace umožňující interakci s databází pomocí uživatelského rozhraní (GUI). Existuje mnoho nástrojů pro práci s databází, které vykonávají příkazy na pozadí, bez nutné znalosti programovacího jazyka. Výběr aplikace s GUI závisí na preferencích konkrétního uživatele, liší se v množství funkcí a designu. Pro testovací účely na místním serveru můžeme využívat například aplikaci TablePlus nebo SQLPro. Pro správu databáze uložené na serveru budeme využívat webový nástroj phpMyAdmin. V obou zmíněných aplikacích je možné zadávat příkazy pomocí

programovacího jazyka SQL. Pro potřeby diplomové práce nebude nutná znalost všech 179 funkcí standardu SQL, nýbrž pouhého zlomku příkazů, vybrané z nich nyní představíme:

Vytvoření tabulky můžeme provádět pomocí GUI. Na následujícím obrázku z aplikace phpMyAdmin jsou vidět nejdůležitější parametry nutné k vytvoření tabulky s jedním sloupcem.

Obrázek 4: Vytvoření sloupce v MySQL tabulce

Zdroj: Vlastní zpracování

Po zadání Jména tabulky a názvu sloupce je nutné vybrat typ dat. Námi využívané *typy* dat budou: VARCHAR pro ukládání textových dat, TIMESTAMP pro ukládání data a času, BIGINT pro ukládání čísel až do hodnoty 2^63-1 (do datového typu INT je možné uložit číslo do hodnoty 2^31-1 = 2 147 483 647, to je méně než počet uživatelů Facebooku v roce 2018 (Clement, Number of monthly active Facebook users worldwide as of 1st quarter 2020, 2020) a FLOAT pro ukládání desetinných čísel. Pole *Délka/Množina* a pole *Porovnání* budeme vyplňovat pouze v případě VARCHAR, vyznačíme tím maximální počet písmen v jedné buňce a kódování textu. *Výchozí* může být například u TIMESTAMP aktuální hodnota nebo u VARCHAR slovní spojení „chybějící text“, v ostatních případech můžeme využívat defaultní hodnotu „Žádná“ (Elmasri & Navathe, 2016). Kód pro vytvoření tabulky z předchozího obrázku je možné získat z phpMyAdmin a dalších programů pro správu databází. Jeho modifikací lze rychleji vytvářet nové tabulky s podobným rozložením:

```
CREATE TABLE nová_tabulka2 (sloupec_s_textem varchar(30) COLLATE
utf8mb4_unicode_ci NOT NULL DEFAULT "chybějící text")
ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci
```

Zápis do tabulky budeme provádět pouze pomocí CLI a skriptů. Následující příklad ukazuje způsob zápisu textu "text" do tabulky:

```
INSERT INTO nová_tabulka (sloupec_s_textem) VALUES ("text")
```

V případě nutnosti vkládání nových řádků bez vzniku duplicitních řádků, je nutné příkaz INSERT INTO modifikovat například do podoby:

```
INSERT INTO nová_tabulka (sloupec_s_textem) SELECT * FROM
(SELECT "text_2") AS tmp WHERE NOT EXISTS (SELECT
sloupec_s_textem FROM nová_tabulka WHERE sloupec_s_textem =
"text_2") LIMIT 1
```

Příkaz vybere všechny sloupce z tabulky nová_tabulka. Následně zadáme text, který chceme zapsat do databáze (v našem případě "text_2") ten se uloží jako dočasný soubor. Pokud sloupec sloupec_s_textem neobsahuje "text_2" zapíše se tato položka do tabulky.

Pro **čtení z tabulky** budeme používat pouze jeden příkaz, obsažený i v předchozím příkladu:

```
SELECT sloupec_s_textem FROM nová_tabulka ORDER BY
sloupec_s_textem ASC
```

Při čtení z tabulky jsou navíc seřazeny hodnoty ze sloupce sloupec_s_textem vzestupně, pomocí příkazů ORDER BY a ASC. Další možností je seřadit data sestupně, příkazem DESC.

Python skripty, používané pro získávání dat obsahují příkazy pro čtení a zápis do tabulky. Do databáze se budeme připojovat pomocí knihovny mysql-connector-python. Nejdříve musíme definovat připojení a kursor:

```
import mysql.connector
cnx = mysql.connector.connect(user='jméno uživatele',
                              password='heslo',
                            host='adresa (localhost)',
                            database='databáze',
                            charset = 'znaková sada (utf8mb4)')
kursor = cnx.cursor()
```

Příkazy pro čtení a zápis můžeme po vytvoření kursoru spouštět v Python skriptu po jejich vložení do příkazu `cursor.execute()`.

2.1.2 Spouštění skriptů

Skripty v programovacím jazyce Python spouštíme v IDE Visual Studio Code, samozřejmě je možné použít jakékoliv jiné jako například PyCharm. Skripty, které je nutné nechat zapnuté dlouhodobě, například skript získávající data ze zpravodajských webů, budeme spouštět pomocí linuxového terminálu. K tomu jsou využívány způsoby představené v této kapitole.

Pro spuštění skriptů na vzdáleném serveru používáme protokol SSH. V případě, že chceme SSH připojení po spuštění skriptu ukončit, nesmíme skript spustit v terminálu otevřeném po zadání adresy. Dvě možnosti pro spuštění skriptů jsou:

- Screen: slouží k vytvoření více obrazovek (terminálů), mezi kterými je možné přepínat.
- Systemd: slouží k definování skriptu jako služby.

Novou obrazovku s názvem Nová_obrazovka lze v linuxovém terminálu vytvořit pomocí příkazu: `screen -S Nová_obrazovka`. Po vytvoření obrazovky jsme do ní okamžitě přepnuti. Odejít z ní je možné pomocí klávesové zkratky Ctrl-a-d. Pro zobrazení existujících obrazovek slouží příkaz: `screen -r` a pro otevření vytvořené obrazovky: `screen -r Nová_obrazovka` (Linuxize, 2020).

Sofistikovanější možností je vytvoření služby pomocí systemd. Výhodou je možnost nastavení spuštění skriptu po restartu počítače, nebo automatické opětovné spuštění skriptu po ukončení následkem vzniklé chyby. Pro vytvoření služby je nejprve nutné vytvořit konfigurační soubor (demo.service), níže je jeho obsah s popisy jednotlivých příkazů:

```
# Název služby
Description=Python Demo Service

[Install]
# Spuštění po restartu počítače
WantedBy=default.target

[Service]
# zobrazení celého výstupu z Python skriptu
Environment=PYTHONUNBUFFERED=1
# Spuštěný příkaz při spuštění služby
ExecStart=/cesta/k/python3 /cesta/k/demo_skript.py
# Opětovné spuštění po vzniku chyby
Restart=on-failure
```

Nebudeme vytvářet systémovou službu, a proto můžeme uložit konfigurační soubor do domácí složky uživatele, konkrétně:

`~/.config/systemd/user/demo.service`

Při každé změně v konfiguračním souboru je nutné spustit příkaz: `systemctl --user daemon-reload`. Následně je možné spustit/zastavit službu pomocí: `systemctl --user start/stop demo.service` (Torfsen, 2020). Zobrazení protokolu

o chybách je umožněno pomocí Journalctl. Pokud budeme chtít přečíst celý protokol u demo.service po spuštění skriptu demo_skript.py, můžeme do terminálu zadat příkaz: `journalctl --user-unit demo.service --no-pager`. Převážně budeme využívat tento příkaz, další možné příkazy pro Journalctl je možné nalézt ve zdroji (Ellingwood, 2018).

2.1.3 Kódy

Data jsou získávána pomocí API, ke kterým bylo nutné získat přístupové kódy. Následně jsou ukládána a čtena z databáze, ke které je také nutné vlastnit přístupové kódy. Veškeré kódy jsou uloženy v souboru `kody.py`. Jedná se o jediný soubor, který není veřejně dostupný. Níže je uveden celý jeho obsah, přičemž místo kódů je název skriptu, který daný kód využívá anebo odkaz na internetovou adresu, kde je možné o přístupové kódy zažádat, případně je vygenerovat:

```
API_key = ("twitter.py - https://developer.twitter.com")
API_secret_key = ("twitter.py")
Access_token = ("twitter.py")
Access_token_secret = ("twitter.py")

Twitter_AR_bearer_token = ("Twitter_AR.py -
https://developer.twitter.com"")

mysql_password = ("heslo")
mysql_username = ("jméno")

import mysql.connector
cnx = mysql.connector.connect(user='jméno', password='heslo',
                              host='localhost',
                              database='twitter',
                              charset = 'utf8mb4')
cnx.autocommit = True #jinak nefungují callbacky
newsAPI_key = ('NewsAPI_11.py - https://newsapi.org')

import praw
reddit = praw.Reddit(client_id='reddit.py -
https://www.reddit.com/prefs/apps',
                     client_secret='reddit.py‘,
                     user_agent='název vyvynuté aplikace',
                     username='uživatelské jméno',
                     password='heslo')

sqlalchemy_psswd = "mysql+pymysql://jméno:heslo@localhost/databáze"
```

2.1.4 Zpravodajské weby

V kapitole 1.4.1. Zpravodajské weby byla identifikovaná knihovna NewsAPI. Byla nalezena omezení, se kterými se budeme muset vypořádat. V následujícím seznamu je jejich výčet a jsou nastíněna řešení, která použijeme, aby nás knihovna neomezovala:

- **Vyhledávání maximálně měsíc starých článků.**
  Články je možné průběžně ukládat do databáze, ve které je možné nalézt data zpětně od prvotního spuštění skriptu.
- **Stahování článků s hodinovým zpožděním.**
  Bylo by omezující v případě reálného obchodování dle algoritmem generovaných signálů. K tomu je možné se uchýlit až v případě vytvoření funkčního systému generujícího zisk a v tom případě by bylo možné zakoupit jednu z placených verzí. V našem případě můžeme při testování na minulých datech bez problémů ukončit obchodování o jednu hodinu dříve.
- **V jednom požadavku je možné zobrazit posledních 100 článků.**
  V případě vyhledávání často vyskytujícího se termínu, jako například: Intel, není možné stahovat články ze všech 50 000 webů. Pokud se, ale podíváme na nejznámější finanční weby, kterými jsou například: https://finance.yahoo.com a https://www.reuters.com, vidíme, že za jeden den vytvoří jen několik málo jednotek článků. Články je možné vyhledávat pouze na vybraných finančních webech o kterých můžeme rozhodnout například podle seznamu z: https://www.alexa.com/topsites/category/Top/Business. API dovoluje vybrat maximálně 20 webových serverů anebo kategorii "Business". Pokud kategorie "Business" generuje více než 100 článků, přistoupíme k vyhledávání z vlastního seznamu webů. Pokud ovšem vyhledáváme méně častá klíčová slova, například souběžně GILD nebo Gilead, je možné ukládat do databáze články ze všech 50 000 zdrojů, jelikož ani takto velké množství serverů nevygeneruje více než 100 článků za daný *časový interval*.
- **Maximálně 100 požadavků za den.**
  Dle množství sledovaných firem budeme muset upravit *časový interval* mezi jednotlivými zavoláními na News API. Pokud bychom sledovali pouze jednu společnost, bylo by možné získat (100 * 100) / 24 = 416 článků za hodinu, respektive 10 000 článků za jeden den.

Byl vytvořen skript **NewsAPI_11.py**, ukládající název webové stránky, datum publikace, nadpis zprávy, odkaz na zprávu a částečný obsah zprávy. Obsah celé zprávy je možné uložit pouze částečně, pokud bychom chtěli získat obsah celé zprávy, museli bychom přistoupit k web scrapingu, nebo zakoupit placenou verzi NewsAPI. Zároveň bylo poprvé přistoupeno k výpočtu sentimentu, který je vypočten pomocí knihovny TextBlob a knihovny VaderSentiment. Sentiment byl vypočítán jak pro nadpis zprávy, tak pro jeho částečný obsah.

Skript umožňuje ukládat data průběžně dle nastavení příkazu `time.sleep()`. Průměrné množství získaných zpráv, věnujících se námi vybraným společnostem je 200 za jeden den, toto číslo je ale velmi závislé na mnoha faktorech. Příkaz je nastaven podle limitu požadavků za den. Chceme limit co nejvíce vytížit. Při současné formulaci skriptu je jej možné spustit maximálně jednou za 21 600 vteřin. Vzhledem k tomu, že také vytváříme interaktivní webovou aplikaci, můžeme se později rozhodnout toto číslo změnit tak, aby se webová stránka obnovovala častěji v denních hodinách oproti hodinám nočním. Zajímavostí je, že v minulém roce (2020) NewsAPI umožňoval 500 zavolání na API v průběhu jednoho dne, v současnosti (2021) je limit 100 zavolání na API denně.

Skript spouštíme pomocí funkce, u které je třeba definovat databázi a klíčové slovo:

```
def news(database, keywords):
```

Po spuštění této funkce je jako první načten čas nejaktuálnějšího příspěvku, který byl uložen do databáze a je uložen do proměnné *last_article_in_db*:

```
cursor.execute("""SELECT published FROM """+ database +"""
                ORDER BY published DESC LIMIT 1""")
for row in cursor.fetchall():
    result = row
last_article_in_db = result[0].strftime(
                                        '%Y-%m-%dT%H:%M:%S')
```

Získaný čas je následně vložen do příkazu, který získává data z NewsAPI. Při volání na NewsAPI získáváme zprávy, které v titulku obsahují požadované klíčové slovo, případně slova. Příspěvky získáváme ze všech zdrojů, pouze v anglickém jazyce a v časovém rozmezí mezi aktuálním časem -1 hodina a *last_article_in_db*. Po získání maximálního množství článků, které byly vydány v časovém rozmezí od *last_article_in_db* a současností, jsou data o jednotlivých článcích ukládána do proměnných a následně do definované databáze. V případě, že v daném časovém rozmezí bylo zveřejněno více než 100 článků, funkce opět zavolá na NewsAPI a vyžádá dalších 100 zpráv do té doby, než jsou načteny všechny chybějící zprávy. Příklad zavolání funkce, kde databáze, do které

se data ukládají, je newsAZN a zpráva musí obsahovat alespoň jedno klíčové slovo AZN anebo AstraZeneca:

```
news('newsAZN', "AZN OR AstraZeneca")
```

2.1.5 Twitter

V kapitole 1.4.2 byl identifikován způsob získávání příspěvků na Twitteru pomocí knihovny Tweepy. Jedná se o veřejně dostupnou možnost získávání tweetů. Byl sestaven skript **twitter.py**, využitím tohoto skriptu je možné získat zdarma tweety s klíčovým slovem pojednávajícím o přibližně jedné akcii (v závislosti na množství tweetů) pomocí knihovny Tweepy. Způsob je podobný tomu, který popisujeme v této kapitole, a proto se jím nebudeme zabývat. Po výměně přibližně sedmi e-mailových zpráv byl získán přístup k datům pomocí Academic Research API a Twitter umožnil stahovat 10 milionů tweetů měsíčně. Data získáváme po přihlášení pomocí získaného přístupového tokenu (bearer token), pomocí HTTP požadavků, které umožňuje zasílat knihovna Requests. Skript pro získávání dat pomocí Academic Research API je pojmenován **Twitter_AR.py**. Nejdříve jsme vytvořili parametry, dle kterých získáváme data:

```
params = {'max_results' : 10,
          'start_time' : '2021-01-29T00:00:00Z',
          'query' : '(boeing OR $BA) lang:en',
          'end_time' : '2021-03-17T00:00:00Z',
          'tweet.fields' : 'created_at,public_metrics,geo',
          'expansions' : 'author_id',
          'user.fields' : 'name,username,public_metrics,
                                              verified,location'
         }
```

V parametrech je definovaný čas od a do kterého chceme data získávat. Je definováno klíčové slovo, případně slova jako `'query'`, současně je na stejném místě definován i jazyk příspěvku, který je opět anglický, vzhledem k používaným knihovnám pro měření sentimentu. Dále v HTTP požadavku získáváme: čas vytvoření příspěvku, kolik like má příspěvek, geologaci uživatele, jméno, přezdívku, kolik lidí dalo uživateli follow, zdali se jedná o ověřeného uživatele a jeho geolokaci. Samotný požadavek je generován pomocí příkazu `response`:

```
bearer_token = "Bearer token"
search_url = "https://api.twitter.com/2/tweets/search/all"
headers = {"Authorization": "Bearer {}".format(bearer_token)}
response = requests.request("GET", search_url, headers=headers,
                            params=params)
```

K získání dat dojde například po spuštění funkce:

```
twitter_ar("2021-04-24T11:00:00Z","2021-04-24T12:00:00Z",
           "airbus", "tweetTable_AR_AB",)
```

Spuštěním funkce s těmito parametry získáme tweety ze dne 24. 4. 2021, v čase mezi 11:00 a 12:00, obsahující klíčové slovo Airbus a následně jsou uloženy do databáze tweetTable_AR_AB.

2.1.6 Reddit

Byl vytvořen skript ***reddit.py***, který pro získávání dat ze sociální sítě Reddit využívá knihovnu PRAW. Po spuštění skriptu jako první dojde k vyhledání subredditu pomocí funkce `reddit.subreddit(sub_reddit)`. Následovně jsou pomocí funkce `subreddit.search(query=query)` vyhledány příspěvky na subredditu obsahující požadovaná klíčová slova. Příspěvků na subredditu je relativně malé množství, mnohem zajímavější je pro nás získávání komentářů k jednotlivým příspěvkům. Jako jeden z nejzajímavějších příspěvků se jeví Daily Discussion Thread. Jak bylo zmíněno v rešeršní části, tento příspěvek se na subredditu r/wallstreetbets objevuje každý den a na konci dne obsahuje přibližně 20 000 komentářů.

Komentáře získáváme pomocí `for` smyčky, která najde všechny komentáře. Na komentáře mohou lidé odpovídat dalšími komentáři. Pro získání všech odpovědí slouží funkce:

```
submission.comments.replace_more(limit=None)
```

Všechny získané komentáře jsou analyzovány pomocí `for` smyčky sloužící k nalezení komentářů obsahujících požadovaná klíčová slova:

```
comment_keywords=["GILD", "Remdesivir", "Gilead"]
if any(s in topic_comment for s in comment_keywords):
```

Po nalezení požadovaných komentářů je vypočten sentiment a komentáře jsou do databáze.

2.1.7 Fundamentální data ze čtvrtletních zpráv

Fundamentální data z výročních ani čtvrtletních zpráv společností nebudeme získávat, ani používat. Získaná data ze zpravodajských webů a sociálních sítí jsou v příliš krátkém časovém okně, aby se daly kombinovat se čtvrtletními, nebo dokonce ročními daty jako je zadluženost, zisk a podobně.

2.1.8 Zpracování dat

Aby bylo možné získaná a uložená data využít při zpětném testování obchodních strategií a tvorbě webové aplikace, byly vytvořeny funkce, které je formátují do vhodné podoby. Při backtestingu budeme využívat pouze data zpracovaná skriptem ***csv_data.py***. Webová aplikace bude využívat data přímo z databáze a tabulky upravené skriptem ***resampling.py***.

V kapitole 2.1 jsme do této chvíle získali a uložili sentimentální data, viz blok Získávání a uchovávání dat na obrázku 3. Data uložená v databázi mimo jiné obsahují datum a čas zveřejnění, sentiment vypočítaný pomocí knihovny VaderSentiment a sentiment vypočítaný pomocí knihovny TextBlob. Právě tyto tři sloupce dat získáme z databáze pomocí funkce:

```
def bt_data_sentiment_ffill_news(database, time_from, resampling):
```

Tato funkce je obsažena ve skriptu ***csv_data.py***. Jak už její název napovídá, slouží k získávání dat z databáze a doplnění chybějících dat. Pro doplnění chybějících dat využíváme funkci `fillna(method="ffill")`, jelikož množství dat získaných zejména ze zpravodajských webů je relativně nízké a při zpětném testování je nutné, aby data neobsahovala žádné chybějící hodnoty. V závislosti na konkrétní akcii může dojít k zveřejnění nové zprávy i pouze jednou za několik hodin. Funkce vybere zmíněné sloupce z databáze, uloží je do proměnné jako `pd.DataFrame`, přidá sloupec *volume* a u každého řádku nastaví jeho hodnotu jako 1. Následně jsou data převzorkována pomocí:

```
df = df.resample(""+resampling+"in").agg(
                                {
                                'sentiment_vader': np.average,
                                'sentiment': np.average,
                                'volume': np.sum
                                })
```

Pokud jako proměnnou `+resampling+` použijeme například `"2m"`, převzorkováním získáme `pd.DataFrame`, jehož řádky jsou ve dvouminutové frekvenci. V případě, že po převzorkování chybí v některých řádcích hodnoty sentimentu, jsou doplněny pomocí zmíněné funkce a možnosti `"ffill"`. Ukázka původních dat a dat doplněných je v následující tabulce.

Tabulka 2: Doplnění chybějících dat

Původní data:	0	0,2			1	0,9	0,8			0,5	0,2	
Doplněná data:	0	0,2	0,2	0,2	1	0,9	0,8	0,8	0,8	0,5	0,2	0,2

Zdroj: Vlastní zpracování

Jako další je v souboru ***csv_data.py*** definována funkce:

```
def get_bt_data(ticker, database, time_from,interval):
```

Tato funkce využívá předchozí funkci a k datům přidává data s cenou akcie. Pokud spustíme funkci s parametry:

```
get_bt_data("AIR", "newsAIRBUS", "2021-02-21", "2m")
```

Dojde k uložení CSV souboru ***newsAIRBUS.csv*** do složky ***backtrader/stocks_data/***, který obsahuje sloupec s časem ve formátu UTC, sloupce s OHLC daty získanými pomocí knihovny yfinance a dva sloupce s hodnotami sentimentu. Data jsou ve dvouminutové frekvenci a v období od 21. 2. 2021 do současnosti.

Časy vygenerovaných dat přesně odpovídají skutečnosti (jsou v čase UTC). Při obchodování pomocí signálů generovaných ze sentimentálních dat může být vhodné vygenerovat data, ve kterých existuje časová prodleva mezi zveřejněním zprávy a vznikem signálu pro nákup, nebo prodej. Časovou prodlevu si můžeme představit jako dobu mezi zveřejněním zprávy a jejím přečtením větším množstvím lidí. Je možné, že časová prodleva neexistuje, vzhledem k vyspělým obchodním terminálům a robotům využívaných hedgeovými fondy s velkým kapitálem a možností okamžitě zareagovat na zveřejněnou zprávu. Použitím funkce `datetime.timedelta()` na časový index dat sentimentu je možné čas posunout o libovolné časové období, například o hodinu zpět:

```
df.index = df.index - timedelta(hours=1)  # časová prodleva
```

Tím docílíme zpoždění mezi změnou sentimentu a reakcí ceny. Získaná data se budou tvářit jako kdyby byla zveřejněna o dané časové okno dříve. Vytvořili jsme časové okno, které může být potřebné pro rozšíření zprávy mezi větší počet lidí. Funkce pro nastavení časové prodlevy je obsažena v souboru ***csv_data.py***. Alternativně je možné posunout čas pomocí stejné funkce u získaných cen akcie.

Posledním skriptem, vytvořeným pro zpracování dat je skript ***resampling.py***. Skript funguje podobně jako předchozí. Načte data z databáze do pd.DataFrame, data převzorkuje na desetiminutová a uloží je zpět do nové tabulky databáze. Takto převzorkovaná data jsou využívána webovou aplikací.

2.2 Backtesting

Při vypracovávání rešeršní části diplomové práce jsme nalezli velké množství nástrojů a strategií pomocí nich implementovaných. Využití existujících nástrojů a strategií je jedna z možností. Ale pokud chceme nastavovat vlastní parametry a využívat vlastních dat, musíme přistoupit k sestrojení vlastního systému pro zpětné testování obchodních strategií. Výhodou vlastních systémů je možnost implementování všech požadovaných funkcionalit, které požadujeme:

- Využití dat cen jednotlivých akcií a možnost vložení vlastních,
- vytváření technických indikátorů,
- zadání a optimalizaci daných parametrů,
- formulování podmínek pro nákup a prodej,
- generování grafických a textových výstupů o průběhu strategie.

Hlavním přínosem je možnost využití dat sentimentu pro generování signálů k nakoupení, prodání, nebo držení akcie. Signály jsou generovány dle zadaných parametrů, jako například hodnota stop-loss, které je možné optimalizovat. Například stop-loss v intervalu celočíselných hodnot od 0 % z ceny akcie do 100 %. Generované signály využíváme ve strategii. Možné je srovnávat strategii s různými parametry signálů anebo úplně vynechat některé parametry. V případě, že nastavíme hodnotu stop-loss jako 10 000 % z ceny akcie, s největší pravděpodobností nikdy nedojde k uzavření obchodu pomocí tohoto signálu.

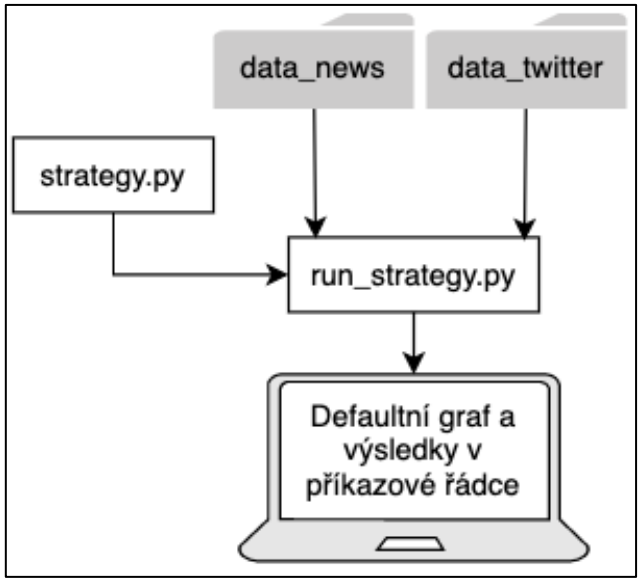
Kompletní skripty vytvořené v kapitole 2.2 jsou dostupné ke stažení na adrese: https://github.com/HSTEP/FDSS/tree/master/backtrader. Složka obsahuje skripty:

- ***run_strategy.py***: pro testování strategie na datech jedné akcie,
- ***strategy.py***: kde jsou definované obchodní signály a podmínky pro otevření obchodu,
- ***optimize_strategy.py***: slouží k optimalizaci parametrů a uložení výsledků testu do jednoho souboru ve formátu CSV,
- ***run_strategy_multistocks.py***: pro testování na datech více akcií a uložení výstupů do souborů ve formátu CSV,
- ***optimization_analysis.py***: pro analýzu a vizualizaci výstupů z optimalizace parametrů,

- ***multistocks_analysis.py***: pro analýzu a vizualizaci výsledků strategie spuštěné se stejnými parametry na datech všech sledovaných akcií.

Vazby mezi jednotlivými skripty opět nejdříve znázorníme pomocí blokového schéma. Vždy vycházíme z dat uložených do jednotlivých složek pomocí skriptu ***csv_data.py*** viz obrázek 4. Spuštění importované strategie ze skriptu ***strategy.py*** na datech jedné společnosti je po zadání cesty k souboru ve skriptu ***run_strategy.py*** možné následovně:

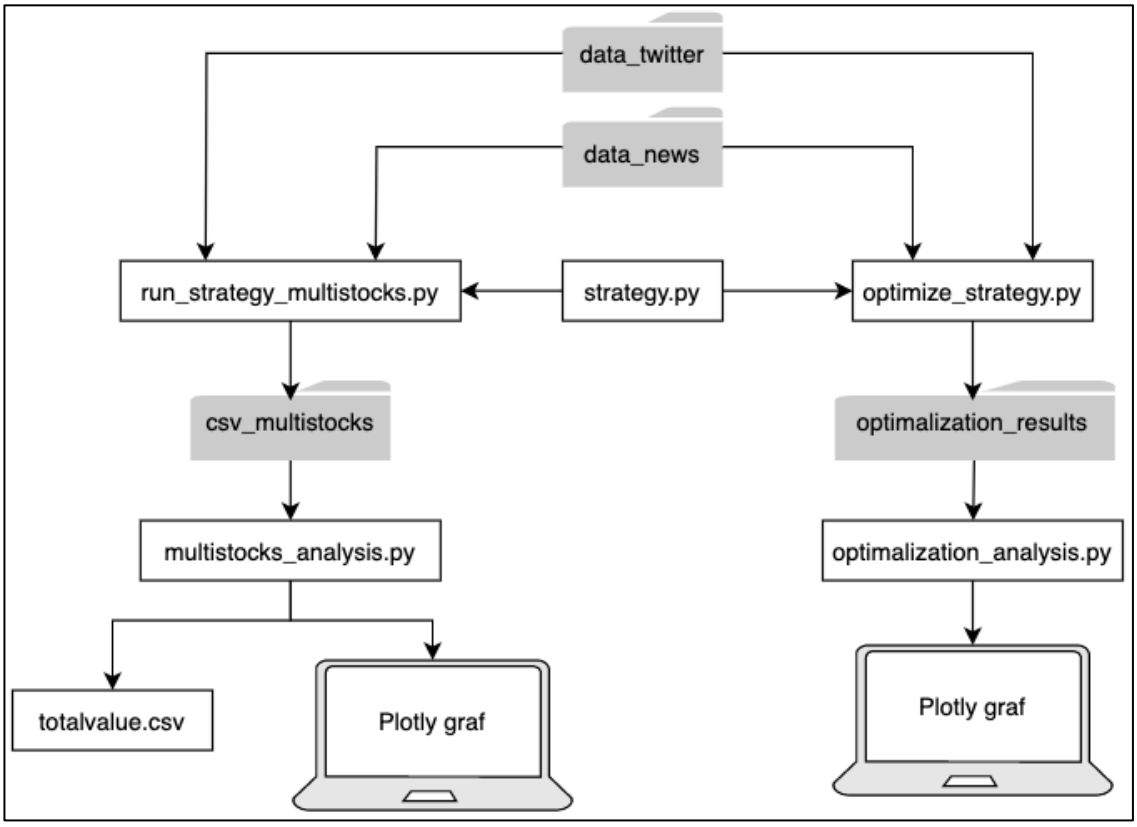
Obrázek 5: Schéma skriptu run_strategy.py



Zdroj: Vlastní zpracování

V kapitole 2.2.4 se věnujeme optimalizaci strategie, ke které slouží skript ***run_strategy_multistocks.py***. V kapitole 2.2.5 spuštění strategie na datech více akcií pomocí skriptu ***optimize_strategy.py***. Analýzou výsledků strategie se zabývají skripty ***multistocks_analysis.py*** a ***optimalization_analysis.py***. Využití těchto skriptů je zobrazeno na následujícím obrázku.

Obrázek 6: Schéma skriptu run_strategy_multistocks.py a optimize_strategy.py



Zdroj: Vlastní zpracování

2.2.1 Cerebro

Před definicí strategie, obsahující dané obchodní signály, dle kterých dochází k otevření a uzavření obchodu, je nutné vytvořit skript, sloužící ke spouštění jednotlivých strategií anebo jejich optimalizaci. Pro spuštění slouží skripty ***run_strategy.py***, ***run_strategy_multistocks.py*** a ***optimize_strategy.py***. Skripty se mírně odlišují dle jejich účelu, základní struktura je v této kapitole vysvětlena na skriptu ***run_strategy.py***.

Ve skriptu je importovaná knihovna Backtrader jako `bt`. Slouží k nastavení základních prvků třídy `cerebro = bt.Cerebro()`. V cerebru nastavuje množství peněz na obchodním účtu, například 1 000 USD:

```
cerebro.broker.setcash(1000)
```

Velmi důležité je nastavit hodnotu poplatků. V případě, že bychom pro reálné obchodování využili brokera eToro, všechny provize a poplatky by byly započítány ve spreadu, a proto se eToro vyznačuje relativně vysokým spreadem. Jeho průměrná hodnota je vypočtena v následující tabulce.

Tabulka 3: Průměrný spread eToro

Akcie	Sell [$]	Buy [$]	Spread [$]	Spread [%]
AIR	103,63	103,74	0,11	0,1060
AMC	Není dostupná k obchodování u brokera eToro			
AZN	50,66	50,73	0,07	0,1380
BA	248	248,28	0,28	0,1128
F	12,22	12,24	0,02	0,1634
NET	74,09	74,26	0,17	0,2289
ORCL	78,88	78,95	0,07	0,0887
PFE	38,53	38,57	0,04	0,1037
RACE	211,77	212,29	0,52	0,2449
TM	157,34	157,67	0,33	0,2093
			Průměrný spread [%]:	**0,1551**

Zdroj: http://etoro.com

Dále eToro požaduje fixní poplatek za výběr z účtu. Ve výpočtu transakčních nákladů jsme zanedbali likviditu a skluz a zaokrouhlili jejich hodnotu na 0,2 %. Tato hodnota byla v cerebru nastavena následovně:

```
cerebro.broker.setcommission(commission=0.002)
```

Pro možnost vložení vlastních dat ve formátu CSV, byla vytvořena třída `GenericCSV_X()`, která dědí třídu `bt.feeds.GenericCSVData`. Kromě OHLC cen může po následující úpravě obsahovat i sloupec se sentimentem:

```
class GenericCSV_X(bt.feeds.GenericCSVData):
    lines = ("sentiment",)
    params = {"sentiment": 6}
```

Data obsahující sloupce s cenou a sentimentem (konkrétně se jedná o sentiment vypočtený pomocí knihovny VaderSentiment, v případě že číslo 6 změníme na 5, bude využíván sentiment vypočtený pomocí knihovny TextBlob), podle kterých bude Backtrader nakupovat a prodávat akcie, vkládáme do cerebra následovně:

```
data = GenericCSV_X(
    dataname="csv_NET.csv",
    dtformat="%Y-%m-%d %H:%M:%S",
    datetime=0,
    high=2,
    low=3,
    open=1,
    close=4,
```

```
    volume=5,
    openinterest=-1,
    sentiment=6,
    timeframe=bt.TimeFrame.Minutes,
)
cerebro.adddata(data)
```

Do proměnné `dataname` ukládáme cestu k souboru vygenerovaného pomocí skriptu ***csv_data.py***. Následně je nastaven formát času a číselně označeny jednotlivé sloupce. Důležité je nastavení `timeframe`. Pokud využíváme soubor s minutovými daty a nastavíme `timeframe = bt.TimeFrame.Days`, obchod není uzavřen v minutě po výskytu signálu, ale až na konci dne.

Poté přidáme strategii, která je definována v následujících kapitolách a uložena ve skriptu ***strategy.py***:

```
cerebro.addstrategy(MA_controll_strategy)
```

V neposlední řadě můžeme strategii spustit pomocí `cerebro.run()`.

Po ukončení zpětného testování jsme informováni o celkovém zisku, nebo ztrátě. K tomu slouží:

```
print(cerebro.broker.getvalue() - 1000)
```

V případě, že chceme získat informací více, musíme do cerebra přidat nějaký analyzátor. Námi využívané jsou:

```
cerebro.addanalyzer(bt.analyzers.TradeAnalyzer, _name="ta")
cerebro.addanalyzer(bt.analyzers.DrawDown, _name="DD")
```

Pomocí nich získáváme:

- Celkový počet obchodů,
- počet dlouhých a krátkých obchodů, které byly profitabilní,
- počet dlouhých a krátkých obchodů se záporným ziskem,
- počet konsekutivních obchodů, které přinesly zisk,
- počet konsekutivních obchodů, které zapříčinily ztrátu,
- drawdown neboli procentní vyjádření největšího poklesu zůstatku na účtu.

2.2.2 Strategie MA sentimentu

Ve skriptu ***strategy.py*** byla vytvořena strategie `MA_cross_Sentiment` a přidána do třídy `bt.Strategy`. Jedná se o první vysvětlovanou strategii, proto je způsob jejího vytvoření popsán nejvíce podrobně. Fungování strategie je také popsáno pomocí

komentářů ve skriptu. Může se zdát komplikovaná, ale po jejím spuštění například pomocí webové aplikace a vygenerování grafu zobrazujícího průběh backtestingu je možné fungování strategie a generování obchodních signálů jednoduše vypozorovat.

```
class MA_cross_Sentiment(bt.Strategy)
```

Jako první jsou ve strategii definovány parametry. V kapitole 2.2.2 můžeme vidět parametr `period`, sloužící k určení periody klouzavého průměru. Místo konkrétního čísla budeme zadávat proměnnou, kterou je možné měnit právě v parametrech:

```
params = dict(
    period_long = 233,
    )
```

Parametry zadané jako `params` je možné optimalizovat.

Z hodnot sentimentu jsou vytvořeny dva klouzavé průměry. Jeden klouzavý průměr s krátkou periodou a druhý s periodou delší. Pomocí klouzavých průměrů budeme generovat obchodní signály pro prodej, nákup anebo držení akcie.

```
self.sma_long_s = bt.indicators.SimpleMovingAverage(
                      self.data.lines.sentiment,
                      period=self.params.period_long)
```

V `def __init__(self):` je kromě klouzavých průměrů definován stop-loss, take-profit, množství nakupovaných akcií a zdali je možné při spuštění strategie realizovat obchodní pokyny. Stop-loss a take-profit nastavíme v procentech jako `params`, aby je bylo možné optimalizovat.

Prodej akcie (`self.sell`) je definován jako funkce `crosdown()`, stejně tak nákup (`self.buy`), jako `crossup()`:

```
def crossup(self):
    self.order = self.buy(size=self.stake)
    self.can_sell = True
```

Cerebro po spuštění strategie postupuje po jednotlivých řádcích obsažených v souboru s daty. Na každém řádku je volána funkce `def next(self):`, obsahující podmínky pro otevření a uzavření obchodu. Jako první jsme nastavili, že pokud máme otevřený obchodní pokyn, nebudou se otevírat žádné další obchody:

```
    if self.order:
        return
```

Pokud je tato podmínka splněna, dojde k nakoupení akcie v případě, že MA s krátkou periodou má hodnotu větší nebo rovnu MA s periodou delší. Klouzavý průměr s delší

periodou zobrazuje dlouhodobý trend sentimentu. Krátkodobý průměr rychle reaguje na změny sentimentu. V tomto případě se generovanými signály snažíme reagovat na krátkodobou změnu sentimentu:

```
    if (
        self.ema50_sentiment[-1] >= self.ema200_sentiment[-1]
        and self.position.size < 1
        and self.can_buy
    ):
        self.crossup()
        return
```

Podmínku je možné obrátit, v tom případě by došlo k nakoupení v případě, že dlouhodobý klouzavý průměr má vyšší hodnotu než klouzavý průměr krátkodobý. A opačným způsobem je definována podmínka pro prodej akcie.

Jako poslední v `def next(self):` voláme dvě funkce:

```
    self.check_stoploss()
    self.check_takeprofit()
```

První funkce funguje jako stop-loss, druhá jako take-profit. Vysvětlena bude jenom jedna, jelikož jsou si hodně podobné. `self.check_stoploss()` zajišťuje, že pokud je pozice buy (koupit) a cena se dostane na hodnotu stop-loss, dojde k uzavření obchodu a dokud MA s kratší periodou nepřejde přes hodnotu MA s periodou delší, není umožněno koupit akcii. Zároveň je ve stejné funkci nastavena podobná podmínka pro případ, že pozice je sell (prodat).

Po spuštění strategie nás v příkazové řádce o jejím průběhu informuje funkce:

```
def notify_trade(self, trade):
```

Zobrazuje stav obchodních pokynů, který může být:

- Submitted: obchodní pokyn je vytvořen
- Accepted: obchodní pokyn je přenesen k brokerovi
- Completed: obchodní pokyn je zadán. Námi využívané obchody jsou typu Market Order a stav completed znamená, že akcie byla nakoupena, případně prodána.

Po uzavření obchodu se zobrazí zisk nebo ztráta z každého obchodu. Spolu se stavem obchodního pokynu zobrazujeme také čas, ve kterém ke změně stavu došlo. Tento čas získáváme ve funkci `def log(self, txt, dt=None):`.

Aby bylo možné spustit jakoukoliv strategii na datech více než jedné akcie, byl vytvořen skript ***run_strategy_multistocks.py.*** Obsahuje slovník `datapath_filename_dict`,

který obsahuje cesty k souborům, generovaných skriptem ***csv_data.py***. Pomocí `for` cyklu postupně vkládá jednotlivá data do třídy `GenericCSV_X()`. V cerebru je pro ukládání výsledků strategie, spuštěné se stejnými parametry na datech všech vybraných akcií nastaven `writer`, aby po dokončení každého testu uložil výsledky do CSV souboru s názvem testované akcie:

```
cerebro.addwriter(bt.WriterFile, csv=True, out=
                  "backtrader/csv_multistocks/"+file_name)
```

Výstupy jednotlivých strategií jsou pro možnost jejich analýzy pomocí skriptu ***multistocks_analysis.py*** ukládány do složky `backtrader/csv_multistocks/`.

2.2.3 Kontrolní strategie

V rešeršní části byla identifikována strategie Buy and Hold. Koupíme akcii na začátku období, ve kterém testujeme ostatní obchodní strategie, a budeme ji držet až do konce období. Je tedy otevřen pouze jeden obchod a na konci období uzavřen.

V předchozí kapitole 2.2.2 vytváříme klouzavé průměry z hodnot sentimentu, které slouží jako obchodní signál. Z tohoto důvodu je navržena druhá kontrolní strategie, která jako obchodní signály využívá klouzavé průměry vytvořené z ceny akcie, přičemž ostatní parametry ponecháme stejné jako u strategie kontrolované. Tím zjistíme, zdali pomocí získaných dat sentimentu generujeme lepší obchodní signály, než pomocí tradičně využívaných a snadněji získatelných klouzavých průměrů počítaných z ceny akcie.

Kontrolní strategie `MA_controll_strategy` se jako všechny ostatní strategie nachází ve skriptu ***strategy.py***. Dlouhodobý klouzavý průměr je možné generovat z OHLC cen, my jsme se rozhodli pro využití C (close) ceny, cena na konci intervalu je pro technickou analýzu využívána také ve zdroji (Edwards, Magee, & Bassetti, 2018). Klouzavý průměr s periodou 233 (například u 2minutových dat se jedná o klouzavý průměr s periodou 466 minut) byl vytvořen následovně:

```
self.sma_long_p = bt.indicators.SimpleMovingAverage(
    (self.datas[0].close), period=233, plotname="SMA_long_p"
)
```

Stejným způsobem vytváříme krátkodobý průměr, který má hodnotu `period` nižší.

2.2.4 Strategie sentiment 0

Jako poslední byla ve skriptu ***strategy.py*** vytvořena třída obsahující strategii sentiment_0_strategy. Tato strategie se od strategie popisované v kapitole 2.2.2 liší podmínkou pro nákup akcie, která byla nastavena následovně:

```
if (
    self.sma_sentiment[-1] > 0
    and self.position.size < 1
    and self.can_buy
):
    self.crossup()
    return
```

A podmínkou pro prodej akcie, která byla nastavena opačně. U této strategie by nemělo být vhodné zaměňovat tyto dvě podmínky. Teoreticky by měla cena akcie růst v případě, že sentiment je pozitivní.

K nakoupení a prodání akcie dochází pomocí signálu:

```
self.sma_sentiment = bt.indicators.SimpleMovingAverage(
                                    self.data.lines.sentiment,
                                    period=self.params.period_short)
```

Signálem je pouze jeden klouzavý průměr, který je pro účely využití webovou aplikací tvořen pomocí parametru period_short. K nakoupení akcie dojde v případě, že hodnota sentimentu je větší než 0 a v opačném případě k prodání. Klouzavý průměr může být vhodné využít u tweetů, jejichž sentiment přechází přes hodnotu 0 velmi často a bez využití klouzavého průměru by bylo tvořeno velmi velké množství obchodů vedoucích k velkým transakčním nákladům. V případě využití dat ze zpravodajských webů je možné nastavit periodu klouzavého průměru 1, tím dojde k využívání neupravených dat získaných pomocí skriptu ***csv_data.py***.

2.2.5 Optimalizace strategie

Ve strategiích byly zmíněny parametry (params), které je možné optimalizovat. Pro účely optimalizace obchodních strategií byl vytvořen skript ***optimize_strategy.py***. V cerebru je oproti skriptu ***run_strategy.py*** nastaveno rozmezí optimalizovaných parametrů:

```
cerebro.optstrategy(MA_cross_Sentiment,
                    period_long=range(180,220),
                    period_short=range(20,50),
                    stop_loss=range(1,10)
                    )
```

V tomto případě je optimalizovanou strategií `MA_cross_Sentiment` a optimalizované parametry jsou perioda dlouhého MA v intervalu <180, 219>, perioda krátkého MA v intervalu <20, 50), hodnota pro stop-loss v intervalu <1, 9>. Intervaly jsou v celých číslech. Hodnota periody 180 znamená, že cerebro počítá klouzavý průměr ze 180 řádků v souboru s daty. Pokud jsou data dvouminutová, jedná se o šestihodinový klouzavý průměr. V případě optimalizování těchto parametrů je strategie spuštěna 10 179krát. Předpokládáme, že cena akcie bude rychleji reagovat na příspěvky na Twitteru, kterých jsme získali mnohem větší množství a jsou zveřejňovány lidmi v mnoha případech jako reakce na přečtení novinové zprávy. Konkrétní nastavení šíře intervalu periody, pomocí kterého se snažíme nalézt nejlepší hodnotu periody, je výpočetně velmi náročné. V kapitole 3 Výsledky se tomuto tématu věnujeme u výsledků optimalizace. Po dokončení optimalizace každé strategie s danou hodnotou parametrů ukládáme hodnoty jednotlivých parametrů a vybrané výstupy z analyzátorů do CSV souboru z důvodu velkého množství kombinací jednotlivých parametrů a pozdější analýzy výsledků. Soubor se nachází ve složce `backtrader/optimalization_results/`.

2.2.6 Spouštění strategií a generování výsledků

Jako první je spuštěn skript ***optimize_strategy.py*** na datech jedné akcie. Získáme soubor ve formátu CSV, obsahující informace o použitých parametrech, zisk (případně ztrátu) a drawdown. Seřadit soubor například od nejvyššího zisku po největší ztrátu a používání dalších skriptů je možné jednoduše například pomocí aplikace Microsoft Excel. Z tohoto souboru můžeme snadno vytvořit `pd.DataFrame`:

```
pd.read_csv("/cesta/k/souboru.csv")
```

Výsledky optimalizace analyzujeme pomocí skriptu ***optimization_analysis.py***. Obsahuje funkci `def scatter3D():`, která slouží k multidimenzionální vizualizaci výsledků optimalizace.

Vybereme vhodné parametry a nastavíme je ve strategii, následně jsou po spuštění skriptu ***run_strategy_multistocks.py*** postupně generovány soubory obsahující výstupy strategie. Ke zpracování těchto souborů slouží skript ***multistocks_analysis.py***. Pomocí `for` cyklu získáváme list obsahující `pd.DataFrame`. Skript obsahuje funkci `def mustistocks_value_chart()`, sloužící k vizualizaci zisku, případně ztráty v průběhu testovaného období. Pro získání konečné hodnoty portfolia používáme funkci `def final_value_to_csv():`, pomocí které získáváme konečnou hodnotu

obchodního účtu po spuštění strategie na datech všech akcií, také slouží k porovnání výsledků se strategií Buy and Hold.

2.3 Webová aplikace

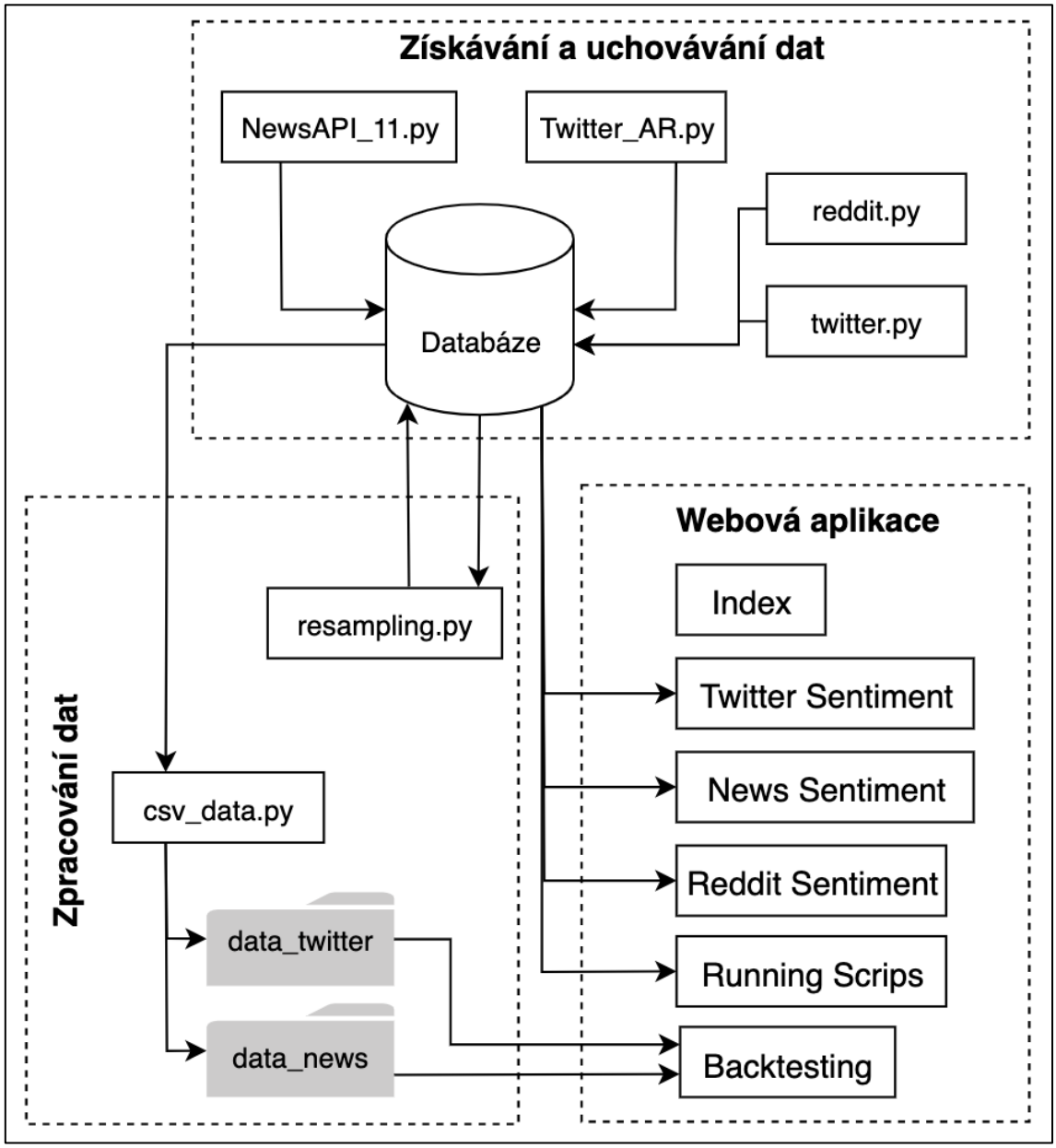
V poslední kapitole rešeršní části je identifikované velké množství existujících webových aplikací pro podporu rozhodování při diskrečním obchodování. Zjistili jsme, že podrobnou analýzou sentimentu se zabývá jen malá část z nich. Přínosem diplomové práce proto bude interaktivní vizualizace dat sentimentu. Do interaktivní webové stránky budeme implementovat následující funkcionality:

- Zobrazení konkrétních příspěvků na sociálních, zpravodajských sítích a jejich sentimentu ve formě tabulky,
- možnost filtrování dat dle různých kritérií,
- grafické zobrazení dat a technických indikátorů,
- další možné nástroje podporující formování rozhodnutí o investici:
    - zobrazení geografické pozice, ze které byl příspěvek odeslán,
    - Word Cloud neboli obrázek tvořený ze slov, kde velikost každého slova je dána množstvím výskytu slova v textu.

Webová aplikace se skládá z rozložení (layout) a Python funkcí zvaných callback. Kompletní rozložení a callback funkce jsou dostupné v souboru ***web.py*** na adrese: https://github.com/HSTEP/FDSS. Na stejné adrese se nacházejí i doplňkové skripty využívané webovou aplikací na jejichž umístění je v příslušené kapitole odkázáno.

Na následujícím obrázku je blokové schéma webové aplikace. Vidíme všechny části webové aplikace, od skriptů, ukládajících sentimentální data do databáze, až po jednotlivé podstránky webové aplikace.

Obrázek 7: Schéma webové aplikace



Zdroj: Vlastní zpracování

2.3.1 Rozložení webové aplikace

Před samotným vytvořením skriptu s webovou aplikací byl na Linux server s operačním systémem Ubuntu nainstalován HTTP server Apache2. Na internetové stránce http://freenom.com, kde je možné zdarma zaregistrovat vybrané domény, byla zaregistrována adresa http://www.stepanh.gq stejně jako http://stepanh.gq. Aby se bylo možné připojit na webový server z vnější sítě, je vhodné mít od poskytovatele internetového připojení nastavenou statickou veřejnou IP adresu, která byla spolu s názvem webové adresy nastavena u poskytovatele domény, ten umožňuje přesměrování

vnějšího uživatele na naši veřejnou IP adresu. Dále bylo na Wi-Fi routeru na lokální síti nastaveno přesměrování portu 80 na IP adresu serveru na kterém je spuštěna webová aplikace. IP adresa serveru s webovou aplikací byla na Wi-Fi routeru také nastavena jako neměnná. Aby se po zadání adresy http://stepanh.gq objevilo rozložení webové aplikace, byl vytvořen konfigurační soubor pro Apache2. Je dostupný na GitHub stránce jako ***stepanh.gq.conf***. Soubor je na serveru uložen ve složce ***/etc/apache2/sites-available***. Následně byl ve složce s webovou aplikací vytvořen soubor ***index.wsgi***, WSGI (Web Server Gateway Interface) popisuje jakým způsobem webový server (Apache2) komunikuje s webovou aplikací. Po provedení těchto nastavení můžeme postoupit k vytvoření skriptu ***web.py***.

Jako první ve skriptu ***web.py*** definujeme aplikaci `app`, do které budeme postupně zabudovávat funkcionalitu webové aplikace. Nastavujeme také `server`, který jsme importovali ve WSGI souboru:

```
app = dash.Dash(__name__, suppress_callback_exceptions=True)
server = app.server
```

Dash může využít externí CSS soubor. My jsme se rozhodli pro lokální uložení souboru a vycházíme z volně dostupného předdefinovaného stylu, který byl zkopírován z adresy: https://codepen.io/chriddyp/pen/bWLwgP.css a následně uložen ve složce ***assets/*** jako ***reset.css***. V souboru byla například změněna barva pozadí:

```
body {
  background-color: rgb(255, 128, 0); }
```

Po otevření internetové stránky je na liště internetového prohlížeče zobrazen název a ikona webu zvaná favicon. Byla vytvořena ikona ***favicon.ico*** a stejně jako CSS soubor uložena do složky ***assets/***, kterou Dash automaticky využívá.

Vracíme se do souboru ***web.py***, kde byl nastaven název webové aplikace:

```
app.title = "StepanH"
```

Barvy kromě CSS souboru definujeme přímo ve skriptu s webovou aplikací. K tomu byl na začátku skriptu definován slovník s barvami:

```
colors = {
    "background": "black",
    "text": "#ff8000",
    "button_background" : "#663300",
    "button_text" : "#01ff70",
    "button_border" : "1px solid #ff8000",
}
```

Barvy a další styly jsou v případě potřeby nastavovány přímo u jednotlivých HTML komponentů. Například u nadpisu `'index'` je nastavena barva ze slovníku a následně zarovnání textu na střed:

```
html.H1(
    children='index',
    style={
        "color": colors["text"],
        "textAlign": "center"
        }),
```

Vytvořená webová aplikace se skládá z více podstránek. Dash k tomu umožňuje použít základní komponent `dcc.Location`:

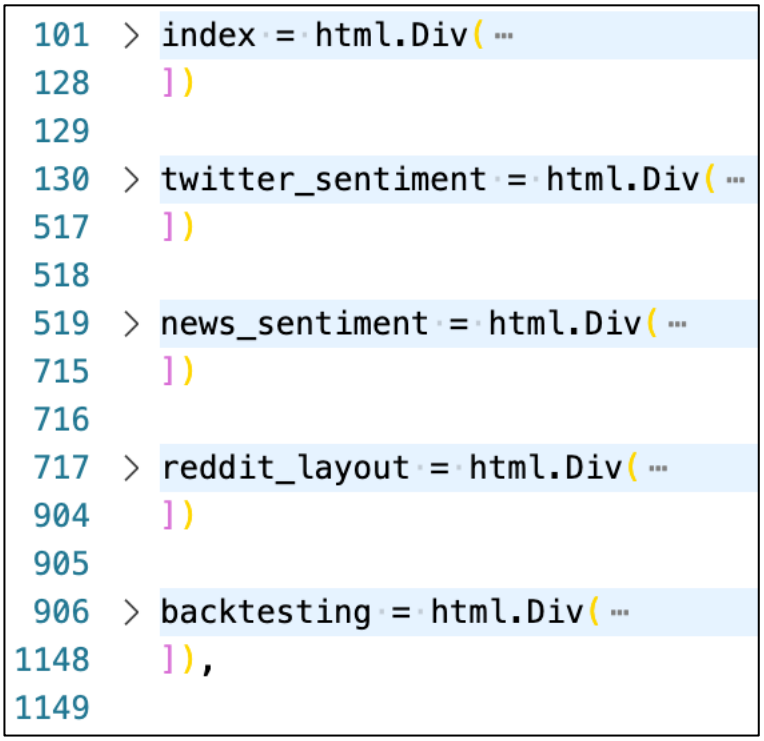
```
app.layout = html.Div([
    dcc.Location(id='url', refresh=False),
    html.Div(id='page-content')
])
```

K přesměrování na správnou podstránku slouží callback, který při zadání adresy http://stepanh.gq vrátí index a při zadání adresy http://stepanh.gq/twitter_sentiment vrátí podstránku s Twitter sentimentem:

```
#--------callback pro otevření cesty k jiné Dash stránce--------
@app.callback(Output('page-content', 'children'),
              [Input('url', 'pathname')])
def display_page(pathname):
    if pathname == '/twitter_sentiment':
        return twitter_sentiment
    else:
        return index
```

Rozložení jednotlivých podstránek je možné definovat v samostatném skriptu, naimportovat je na začátku skriptu ***web.py*** a odkazovat na ně v callbacku pro otevření dané stránky. My jsme se rozhodli pro nevytváření dílčích souborů, všechna rozložení jsou definována ve skriptu ***web.py***. Záleží na osobních preferencích, námi vytvořený skript je relativně dlouhý, ale vzhledem k tomu, že jednotlivé části je v IDE jednoduše možné skrýt, nepovažuje autor obsáhlost skriptu za problém. Ve skriptu ***web.py*** jsou definovaná rozložení pro podstránky, které je možné přehledně vidět při sbalení všech bloků:

Obrázek 8: Bloky s podstránkami



Zdroj: Vlastní zpracování

2.3.2 Index

Na většině internetových stránek je index první stránka, která se objeví po zadání základní adresy. V našem případě se po zadání adresy http://stepanh.gq zobrazí rozcestí obsahující odkazy, které uživatele přesměrují na danou podstránku. Téměř všechny odkazy ve formě tlačítek jsou uvedeny v záhlaví všech podstránek. Uvádíme jeden odkaz, který uživatele přesměruje na podstránku twitter_sentiment:

```
    html.Button(
        dcc.Link(
            'Twitter sentiment',
            href='/twitter_sentiment',
            style={
                "color" : colors["button_text"]
                }
        ),
        style={
            "background-color" : colors["button_background"],
            "border" : colors["button_border"]
        }
    ),
```

2.3.3 Twitter sentiment

Celé rozložení podstránky Twitter Sentiment se nachází ve skriptu ***web.py***. Skládá se ze tří částí:

1. Záhlaví
2. Graf
3. Tabulka

Záhlaví je skoro stejné jako na stránce Index, jediný rozdíl je v chybějícím odkazu na Twitter Sentiment, jelikož se na této stránce uživatel nachází.

Část s grafem začíná základním komponentem knihovny Dash `dcc.Dropdown`. pomocí tohoto komponentu je definována rozbalovací nabídka, jejíž defaultní hodnota je nastavena:

```
value='tweetTable_AR_NET_r'
```

Jedná se o název tabulky v databázi, ze které získáváme data v podobě `pd.DataFrame` a vytváříme graf pomocí knihovny `plotly.graph_objects` importované do skriptu jako `go`. Vzhledem k tomu, že není možné vytvořit responzivní graf s několika miliony body, byla vytvořena z původní tabulky `'tweetTable_AR_NET'` tabulka `'tweetTable_AR_NET_r'`, která obsahuje dvouminutová data. Tato a všechny ostatní tabulky, které je možné nastavit pomocí rozbalovací nabídky, byla vytvořena pomocí skriptu ***resampling.py***. Hodnota rozbalovací nabídky slouží k výběru dat v grafu pomocí callbacku, který je ve skriptu označen jako:

```
#---------callback pro update grafu z MySQL tweetTable----------
```

V callbacku je také nastaven způsob vytváření klouzavých průměrů pomocí komponentu `dcc.Input` a tlačítka `dcc.Button`. Po kliknutí na tlačítko jsou vykresleny zadané klouzavé průměry. Jako poslední je v grafu možné zobrazovat klouzavé průměry vypočtené pomocí knihovny TextBlob a VaderSentiment. Sentiment je možné, jak již bylo zmíněno, zobrazit pomocí klouzavého průměru a nebo bodově (scatter) zobrazující všechna data v převzorkované databázi. Zaškrtávací pole byla vytvořena pomocí komponentu `dcc.Checklist`. Jako poslední se pod grafem nachází posuvník vytvořený pomocí komponentu `dcc.Slider`, kterým je možné vybrat časové okno zobrazovaných dat. Defaultně je u posuvníku nastaven datum a čas 10 dní od aktuálního času. Doba načítání grafu se odvíjí od počtu zobrazovaných dat. Aby měl uživatel odezvu o tom, zdali se data načítají, byl graf zabalen do komponentu `dcc.Loading`.

Poslední část s tabulkou je generovaná z dat získaných a uložených pomocí skriptu ***Twitter_AR.py***. Načtení několika milionů tweetů do jedné tabulky je velmi náročné, proto byl implementován systém pomocí základních komponentů knihovny Dash, kterým je možné generovat data dle zadaných pravidel. Jako první je možné pomocí rozbalovací nabídky vybrat tabulku z databáze obsahující zobrazená klíčová slova. Následně zadáváme časové okno pro data, které je defaultně nastaveno na 20 dní. Dále vybíráme sloupec tabulky, podle kterého se jednotlivé řádky tabulky seřadí. Jako poslední zadáme počet dat s nejvyšší hodnotou, defaultně tweet_count (množství tweetů které uživatel zveřejnil), a množství dat s nejnižší hodnotou. Po kliknutí na tlačítko Search je vytvořen list obsahující pd.DataFrame, v listu každý pd.DataFrame obsahuje data získaná za jednu hodinu. Z tohoto listu je vybráno množství tweetů s nejvyšší hodnotou (with highest value), případně s nejnižší hodnotou a následně jsou tweety souhrnně zobrazeny ve webové aplikaci pomocí komponentu `dash_table.DataTable`.

2.3.4 News sentiment

Rozložení podstránky News sentiment se nachází ve skriptu ***web.py***. Skládá se ze čtyř částí:

1. Záhlaví
2. Graf
3. Word Cloud
4. Tabulka

Záhlaví vytvořeno stejným způsobem jako v předchozí kapitole.

Další tři části jsou interaktivně propojeny. Vzhledem k tomu, že zprávy na zpravodajských webech jsou zveřejňovány méně intenzivně, jsou využívána data přímo z jednotlivých tabulek databáze, která jsou postupně doplňována o nové zprávy. Velmi důležitá je rozbalovací nabídka `dcc.Dropdown` s unikátním identifikátorem `id ='news-dropdown'`. Podle hodnoty v rozbalovací nabídce jsou zobrazena data na celé stránce. Defaultně je hodnota nastavena pro zobrazování `value ='newsGILD'`, neboli dat z tabulky newsGILD, která obsahuje zprávy z 2 000 zpravodajských webů, s klíčovými slovy GILD, nebo Gilead, nebo Remdesivir v titulku zprávy. Po vybrání tabulky je možné v grafu zobrazovat klouzavé průměry sentimentu nebo přímo jednotlivé zprávy. Dle hodnoty posuvníku (datum je možné nalézt pod posuvníkem, jedná se

o Slider Value) je z textu nadpisů článků zveřejněných do daného data tvořen Word Cloud, pomocí callbacku:

```
#--------------callback pro update wordcloudu news--------------
```

Velikost slov ve Word Cloud je dána množstvím výskytu daného slova. Jeho barva je dána nastavením barevného schéma `colormap`. Jako poslední je na webové stránce tabulka obsahující všechna data z databáze dle vybrané tabulky pomocí rozbalovací nabídky. Data v tabulce je možné filtrovat pomocí šipek nacházejících se v první řádce tabulky. Filtrování je umožněno nastavením:

```
dash_table.DataTable(
    filter_action='native')
```

2.3.5 Reddit Sentiment

Podstránka byla vytvořena podobným způsobem jako dříve popisované. Popisujeme proto důležité části kódů, ze kterých se skládá, a ještě jsme se o nich nezmínili.

Téměř v každé tabulce generované webovou aplikací se v každém řádku nachází odkazy na zdrojová data. Aby bylo možné zobrazit zkrácený odkaz na který je možné kliknout a přesměrovat se na danou internetovou stránku, bylo nutné odkazy upravit, jelikož `dash_table.DataTable` dokáže zobrazit odkazy pouze v jazyce Markdown. Následující způsobem je každý odkaz v `pd.DataFrame` přeformátován do jazyka Markdown:

```
links = df['url'].to_list()
rows = []
for x in links:
    link = '[link](' +str(x) + ')'
    rows.append(link)#
df['url'] = rows
```

Jako poslední popíšeme funkcionalitu základního komponentu knihovny Dash, kterým je `dcc.interval`. Umožňuje automatické obnovování jednotlivých částí webové aplikace. Na podstránce Reddit Sentiment slouží k automatickému obnovování grafu a tabulky. Je definován v rozložení stránky:

```
dcc.Interval(id='interval-component-reddit',
             interval=70*1000, # in milliseconds
             n_intervals=0)
```

Následně je jeho id využíváno v callback funkcích, které generují graf a tabulku. Na příkladu je použit k automatické obnově tabulky:

```
#---------callback pro update tabulky z MySQL redditGILD--------
@app.callback(
        Output('table_redditGILD', 'data'),
    [
        Input('reddit-dropdown', 'value'), #call pro dropdown
        Input('interval-component-reddit', 'n_intervals')
    ])
```

2.3.6 Informace o spuštěných skriptech

Ve webové aplikaci je uživatel na podstránce s názvem Running Scripts informován o času posledního spuštění skriptu sloužícího k získání sentimentální dat. K tomu byla vytvořena univerzální funkce, kterou je možné přidat do jakéhokoliv souboru:

```
def is_it_running():
    script_name = "název skriptu"
    now = datetime.now().isoformat()
    cursor.execute("""
                    UPDATE
                        running_scripts
                    SET
                        script = %s, time = %s
                    WHERE
                        script = %s""",
                    (script_name, now, script_name))
    kody.cnx.commit()
```

2.3.7 Backtesting

Byla vytvořena podstránka, ve které je možné testovat obchodní strategie pomocí skriptu ***run_strategy.py***, který byl spolu se skriptem ***strategy.py*** mírně modifikován pro umožnění spuštění přímo z webové aplikace. Oba tyto soubory je možné nalézt ve složce ***backtrader/backtrader_web/***. Soubory v této složce jsou upraveny pro možnost spuštění skriptu a zadáním parametrů pomocí linuxového terminálu. Je například možné zpětné testování spustit v příkazové řádce následujícím způsobem:

```
python3 run_strategy.py "strat_id=2" "stopLoss=0"
```

Dojde k testování strategie `MA_cross_Sentiment` s parametrem stop-loss = 0 % a všechny ostatní parametry zůstanou stejné, jako jsou definovány ve skriptu ***strategy.py*** ve slovníku `params`.

Poslední soubor, který umožnil backtesting ve webové aplikaci je uložen ve stejné složce jako webová aplikace a má název ***bt_for_web.py***. Tento skript obsahuje dvě funkce, první funkce je:

```
def run_strategy(*args):

proc = subprocess.Popen(["/cesta/k/python3",
                         /cesta/k/run_strategy.py"]+list(args),
                         stdout=subprocess.PIPE, cwd="..")
   return proc
```

Funkce slouží ke spuštění skriptu ***run_strategy.py*** dle zandaných argumentů `(args)` ve webové aplikaci.

Druhou funkcí je `def bt_make_chart():`, která slouží k vygenerování grafu pomocí knihovny Plotly. Pro interaktivní zobrazení grafu nebylo možné využít knihovnou Backtrader defaultně generovaného Matplotlib grafu. Dash není s knihovnou Matplotlib kompatibilní. Výstupy strategie, ze kterých je tvořen Plotly graf jsou generovány skriptem ***backtrader/backtrader_web/run_strategy.py*** po nastavení funkce `addwriter` v cerebru:

```
cerebro.addwriter(bt.WriterFile,
                  csv=True,
                  out="/cesta/k/web_writer_backtrader.csv")
```

Při každém spuštění backtestingu pomocí webové aplikace dojde k přepsání souboru ***/backtrader/backtrader_web/web_writer_backtrader.csv***, ze kterého je následně tvořen Plotly graf.

Podstránka s názvem Backtesting je tvořena ze dvou částí:

1. Výběr parametrů pro backtesting, jeho spuštění a generování výstupů
2. Graf

V první části je možné pomocí rozbalovací nabídky vybrat data jedné akcie, získaná pomocí skriptu ***csv_data.py***. Na levé straně se nachází pole pro výběr konkrétní strategie a pro zadání parametrů, pomocí kterých jsou generovány obchodní signály. Po vybrání strategie je možné pomocí tlačítka Start Backtest spustit zpětné testování obchodní strategie. Po jejích spuštění se začnou objevovat výstupy o průběhu strategie. Spuštění strategie, zobrazení průběhu backtestingu a hodnot jednotlivých parametrů je definováno pomocí callbacků, nacházejících se pod komentářem:

```
#-------------------callbacky pro backtesting-------------------
```

Po spuštění backtestingu je možné vygenerovat Plotly graf zobrazující průběh strategie. K vykreslení grafu dojde po stisknutí tlačítka Make Chart.

3 Výsledky

3.1 Získaná a uchovávaná data

Byly vytvořeny dva systémy pro podporu finančního rozhodování využívající data uchovávaná v MySQL databázi. V následujících podkapitolách představujeme data získaná a uložená pomocí skriptů vytvořených v kapitole 2.1. Data je nutné přiložit k diplomové práci jako přílohu, ale vzhledem k tomu, že některé skripty průběžně získávají a uchovávají data, není možné zajistit aktuálnost přiloženého souboru. Databáze byla exportována dne 8. 5. 2021 ve formátu SQL. Exportovaný soubor je dostupný ke stažení na adrese:

- https://drive.google.com/file/d/13ZgQLq2XDBC-xoi-g4jmUZsx6nBFig0-/view?usp=sharing

Soubor je možné importovat na MySQL server například pomocí linuxového terminálu spuštěním příkazu:

```
mysql -u root -p DSS_sentiment < NET_ORCL_PFE_RACE.sql
```

Po spuštění příkazu dojde k vytvoření databáze s názvem DSS_sentiment obsahující jednotlivé tabulky s daty. V následujících podkapitolách jsou představeny základní informace o jednotlivých tabulkách. K tomuto účelu byl vytvořen skript ***database_info.py*** generující výstupy v linuxovém terminálu, které jsou prezentovány prostřednictvím obrázků. Skript je dostupný na adrese: https://github.com/HSTEP/FDSS.

3.1.1 Zpravodajské weby

Obrázek 9: Tabulky s daty ze zpravodajských webů

	Název tabulky	Počet řádků	Nejstarší příspěvek	Nejnovější příspěvek	Velikost [MB]
0	newsAIRBUS	353	2021-02-18 13:48:36	2021-05-07 21:45:46	0.17
1	newsAMC	574	2021-03-01 01:30:50	2021-05-07 22:15:14	0.30
2	newsAZN	6906	2021-02-18 12:45:17	2021-05-08 12:02:18	3.52
3	newsBOEING	1467	2021-02-18 13:42:00	2021-05-08 01:55:00	1.52
4	newsF	2787	2021-02-18 14:05:00	2021-05-08 00:00:27	1.52
5	newsGILD	1757	2020-07-17 19:14:47	2021-05-08 05:04:55	0.28
6	newsNET	152	2021-02-21 13:12:55	2021-05-07 19:26:18	0.09
7	newsORCL	832	2021-02-18 14:12:00	2021-05-08 10:50:09	0.41
8	newsPFE	3036	2021-02-28 19:22:21	2021-05-08 12:03:10	1.52
9	newsRACE	691	2021-02-18 16:00:00	2021-05-08 05:30:42	0.34
10	newsTOYOF	1262	2021-02-18 13:45:00	2021-05-08 10:24:47	1.52

Celková velikost= 11.19 MB
Celkový počet řádků= 19817

Zdroj: Vlastní zpracování

Získávání dat ze zpravodajských webů je prováděno průběžně a s téměř žádnými problémy.

3.1.2 Twitter

Data ze sociální sítě Twitter byla získávána pomocí API pro akademický výzkum. Stejná data by byla možná získat využitím zpoplatněného API. Data byla získávána zpětně a pro jejich doplnění je nutné ručně spustit skript ***Twitter_AR.py***. V průběhu získávání dat dochází k uzavření spojení ze strany Twitteru. Nejčastěji je příčinou chyba s kódovým označením 503 (Service Unavailable), neboli servery Twitteru jsou přetíženy. Vzhledem k tomu, že API pro akademický výzkum není možné využívat věčně, nebylo pomocí tohoto API implementováno získávání živých dat. Data byla nejdříve, v průběhu přibližně pěti dnů, získána na notebooku s procesorem Apple M1 a SSD. Následně byla přenesena do databáze na server s procesorem Intel Core i5-2500K, 8GB RAM a přibližně 12 let starým HDD. Data byla nutná uložit na server zejména pro účely jejich využití webovou aplikací.

Na následujícím obrázku je vidět množství získaných tweetů, které se pohybuje v miliónech. V tabulkách můžeme nezřídka nalézt dva tweety, které byly vydány ve stejnou vteřinu. Pro možnost jejich zobrazení v grafu vytvářeného webovou aplikací byly data v tabulkách převzorkována na desetiminutová pomocí skriptu ***resampling.py*** a uložena do nových tabulek jejichž název končí „_r“.

Obrázek 10: Tabulky s daty ze sociální sítě Twitter

	Název tabulky	Počet řádků	Nejstarší příspěvek	Nejnovější příspěvek	Velikost [MB]
0	tweetTable_AR_AB	228619	2021-01-05 03:13:19	2021-05-06 11:59:54	67.11
1	tweetTable_AR_AB_r	17477	2021-01-05 03:10:00	2021-05-06 11:50:00	1.84
2	tweetTable_AR_AMC	1987309	2021-01-28 16:32:28	2021-05-06 11:59:56	510.56
3	tweetTable_AR_AMC_r	14085	2021-01-28 16:30:00	2021-05-06 11:50:00	1.78
4	tweetTable_AR_AZN	2211273	2021-03-07 23:30:50	2021-05-06 11:59:52	273.34
5	tweetTable_AR_AZN_r	8571	2021-03-07 23:30:00	2021-05-06 11:50:00	0.69
6	tweetTable_AR_BOEING	788453	2020-12-01 00:00:54	2021-05-06 11:57:44	198.27
7	tweetTable_AR_BOEING_r	22536	2020-12-01 00:00:00	2021-05-06 11:50:00	1.92
8	tweetTable_AR_F	3753347	2021-01-19 22:35:09	2021-05-06 11:59:58	1000.58
9	tweetTable_AR_F_r	15345	2021-01-19 22:30:00	2021-05-06 11:50:00	1.81
10	tweetTable_AR_GILD	112674	2020-12-01 00:01:56	2021-05-06 11:59:42	32.08
11	tweetTable_AR_GILD_r	22536	2020-12-01 00:00:00	2021-05-06 11:50:00	1.92
12	tweetTable_AR_NET	110269	2020-12-01 00:01:52	2021-04-24 11:48:05	31.08
13	tweetTable_AR_NET_r	20807	2020-12-01 00:00:00	2021-04-24 11:40:00	1.89
14	tweetTable_AR_ORCL	1282803	2020-12-01 00:00:20	2021-04-24 11:59:55	351.45
15	tweetTable_AR_ORCL_r	20808	2020-12-01 00:00:00	2021-04-24 11:50:00	1.89
16	tweetTable_AR_PFE	2136758	2020-12-01 12:21:12	2021-04-24 11:59:58	560.56
17	tweetTable_AR_PFE_r	20734	2020-12-01 12:20:00	2021-04-24 11:50:00	1.89
18	tweetTable_AR_RACE	745091	2020-12-01 00:00:42	2021-04-24 11:59:56	188.25
19	tweetTable_AR_RACE_r	20808	2020-12-01 00:00:00	2021-04-24 11:50:00	1.89
20	tweetTable_AR_TOYOF	1192780	2020-12-01 00:00:12	2021-04-24 11:59:58	309.41
21	tweetTable_AR_TOYOF_r	20808	2020-12-01 00:00:00	2021-04-24 11:50:00	1.89

Celková velikost= 3542.0999999999995 MB
Celkový počet řádků= 14753891

Zdroj: Vlastní zpracování

Můžeme také pozorovat neaktuálnost dat (dnes je 7. 5. 2021). Skript ***Twitter_AR.py*** byl na serveru spuštěn, aby data doplnil, ale zřejmě z důvodu zastaralých komponentů dochází k doplňování velmi pomalu. Například doplnění chybějících dat mezi 24. 4. 2021 a 6.5. 2021 u akcie společnosti Ford trvalo přibližně 32 hodin.

Pro získávání živých dat přímo v čase jejich zveřejnění byl vytvořen skript ***twitter.py***. Tento skript ovšem s použitím účtu Twitter API, který je zdarma, dokáže získat pouze data o jedné akcii.

3.1.3 Reddit

Data získaná skriptem ***reddit.py*** nebudou využívána při backtestingu, jelikož nebyla získána úplná data z důvodu dlouhé doby potřebné k získání historických dat. Skript byl sestaven tak, aby pomocí něj bylo možné získat všechny komentáře u všech příspěvků obsahujících zadané klíčové slovo. Problém nastal na subredditu r/wallstreetbets u příspěvku Daily Discussion Thread. Předpokládáme, že právě tento příspěvek bude obsahovat velké množství pro nás důležitých sentimentálních dat bez kterých není vhodné přistoupit k backtestingu. Skript data stáhnout dokáže, ale Reddit má nastavené neznámý limit `time.sleep()`, který získávání dat mohutně zpomaluje. Eventuálně by došlo ke stažení všech kýžených dat, ale netroufáme si ani odhadnout, kolik hodin, nebo dní by musel být skript spuštěn. Kromě dlouhé doby potřebné pro získání dat je skript protkaný bloky, které kód testují na výskyt chyb.

Skript byl proto upraven pro získání dat pouze o jedné akcii s vynecháním příspěvku Daily Discussion Thread. Časové okno pro získávání dat je možné nastavit ve funkci `subreddit.search()`, pokud ho nastavíme na jeden rok, data jsou získána za přibližně osmnáct minut. Při prvním spuštění byla získána všechna data, jejich množství je možné vidět na následujícím obrázku.

Obrázek 11: Tabulka s daty ze sociální sítě Reddit

```
  Název tabulky Počet řádků  Nejstarší příspěvek Nejnovější příspěvek  Velikost [MB]
0    redditGILD        1326  2012-09-27 12:23:09  2021-05-07 13:14:46          1.52
Celková velikost=  1.52 MB
Celkový počet řádků=  1326
```

Zdroj: Vlastní zpracování

Po prvotním spuštění byl skript upraven tak, by získával data průběžně každou hodinu. Takto získaná data jsou využívána webovou aplikací.

3.1.4 Zpracovaná data

Pomocí skriptu ***csv_data.py*** je možné vytvořit soubory, které využíváme při zpětném testování obchodních transakcí. Po spuštění skriptu dojde k převzorkování dat dle nastavené hodnoty a případně k doplnění dat chybějících. Frekvence výsledných dat může být například minutová, hodinová, nebo denní. Pro vyzkoušení skriptů sloužících k backtestingu není nutné stahovat celou databázi. Data vytvořená skriptem ***csv_data.py*** jsou dostupná ve složce ***/data_news*** a ***/data_twitter*** na následující adrese:

https://github.com/HSTEP/FDSS/tree/master/backtrader.

Na následujícím obrázku můžeme vidět podobu dat zpracovaných skriptem csv_data.py. Zobrazená data obsahují sloupce s OHLC cenami, sentiment (sentiment vypočtený pomocí knihovny TextBlob), sentiment_vader (sentiment vypočtený pomocí knihovny VaderSentiment) a datum a čas který je převzorkovaný na dvouminutovou frekvenci.

Obrázek 12: Zpracování dat pro backtesting

Datetime	Open	High	Low	Close	sentiment	sentiment_vader
2021-03-19 14:22:00	13.99	14.06	13.88	13.92	-0.05	-0.13
2021-03-19 14:24:00	13.91	14	13.85	13.97	-0.05	-0.13
2021-03-19 14:26:00	13.97	14	13.89	13.92	-0.05	-0.13
2021-03-19 14:28:00	13.93	13.94	13.78	13.81	0	0.42
2021-03-19 14:30:00	13.81	13.86	13.73	13.81	0	0.42

Zdroj: Vlastní zpracování

Po spuštění skriptu resampling.py dojde k převzorkování dat na desetiminutová a následně jsou uložena do databáze. Na dalším je zobrazen úryvek dat z tabulky tweetTable_AR_AB_r získaný pomocí phpMyAdmin.

Obrázek 13: TweetTable_AR_AB_r

created_at	sentiment_vader	sentiment_textblob	volume
2021-01-05 03:10:00	0.0857	0.37380957142857146	7
2021-01-05 03:20:00	0	0.2714285	4
2021-01-05 03:30:00	0	0	1
2021-01-05 03:40:00	0.07452222222222223	0.22619044444444444	9
2021-01-05 03:50:00	0.4589600000000001	0.1233334	5

Zdroj: Vlastní zpracování

Data byla získána z tabulky TweetTable_AR_AB, převzorkována a následně uložena do zmíněné tabulky.

3.2 Backtesting

Bylo získáno velké množství dat, které je možné využít vytvořeným systémem pro zpětné testování obchodních strategií. Obchodní signály jsou generovány pomocí velkého množství kombinací jednotlivých parametrů. V kapitole jsou představeny výstupy, které je možné generovat pomocí vytvořeného systému. Jedním z výstupů jsou interaktivní grafy, které se obtížně prezentují pomocí statických obrázků. Prezentované grafy je možné zobrazit v interaktivní podobě na adrese http://stepanh.gq/backtesting_results. Na této podstránce jsou grafy setříděny do záložek, pojmenovaných dle názvu jednotlivých obrázků prezentovaných v následujících kapitolách. Pro ukázku výstupů systému bylo pro sjednocení kapitoly využíváno pětiminutových dat vygenerovaných skriptem ***csv_data.py*** a pro vytváření klouzavých průměrů ze sentimentálních dat jsou využívána data vypočtená pomocí knihovny VaderSentiment. Pokud hovoříme o klouzavém průměru s periodou 20, jedná se o 100minutový klouzavý průměr.

3.2.1 Strategie MA sentimentu

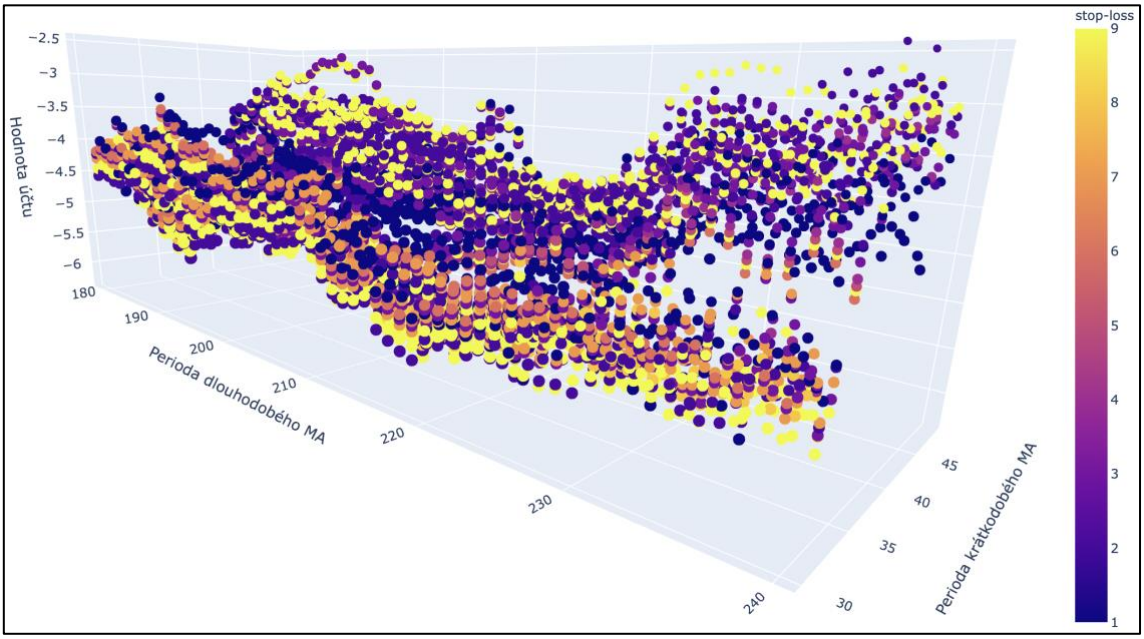
Strategie MA sentimentu byla optimalizována na datech ze zpravodajských webů společnosti Ford (newsF.csv) s parametry:

- dlouhého klouzavého průměru v intervalu <180, 240)
- krátkého klouzavého průměru v intervalu <30, 50)
- stop-loss v intervalu <1 %, 10 %)
- take-profit = 8 %

Doba trvání této optimalizace byla 32 minut a 57 vteřin. V průběhu optimalizace bylo na disk zapsáno 178,21 GB dat a z disku přečteno 172,3 GB dat.

Na následujícím obrázku každý bod zobrazuje jeden test. Body jsou barevně odlišeny podle toho, jak byl velký stop-loss a velikost bodu závisí na velikosti drawdown.

Obrázek 14: Strategie MA sentimentu opt. 1



Zdroj: Vlastní zpracování

S parametry nejméně ztrátového testu ($-2,52) jsme pomocí skriptu ***run_strategy.py*** získali podrobnější informace o jeho průběhu. Parametry byly nastaveny v souboru ***strategy.py*** jako:

- dlouhodobý klouzavý průměr = 238
- krátkodobý klouzavý průměr = 42
- stop-loss = 2 %
- take-profit = 8 %

Výstupy z příkazové řádky jsou na následujícím obrázku:

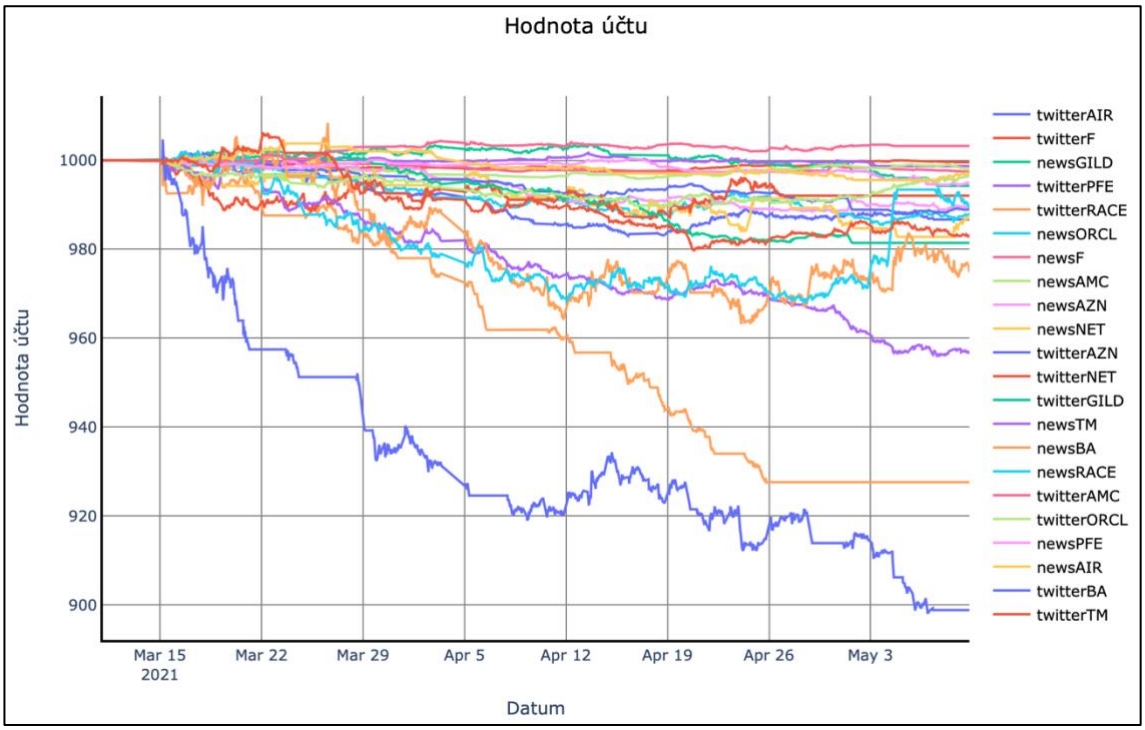
Obrázek 15: run_strategy.py newsF

```
startcash:  1000
PnL:  -2.5208583335877393
PnL percentage:  -0.25208583335877394 %
total trades: 70
long trades won: 12
long trades lost: 24
short trades won: 11
short trades lost: 22
DrawDown: 0.2878095566167377
winning streak: 3
losing streak: 14
```

Zdroj: Vlastní zpracování

Celkem bylo uskutečněno 70 obchodních pokynů. Celý výstup z příkazové řádky spolu s grafickým výstupem je po zadání zmíněných parametrů možné vygenerovat ve webové aplikaci na adrese http://stepanh.gq/backtesting. Je možné, že výsledky budou odlišné vzhledem k aktuálnějším datům vygenerovaných skriptem ***csv_data.py***.

Strategie se zadanými zmíněnými fixními parametry byla testována na všech datech pomocí skriptu ***run_strategy_multistocks.py***. Průběh hodnoty peněžních prostředků na účtu je na následujícím obrázku. Po jeho zobrazení ve webové aplikaci je mimo jiné možné skrývat jednotlivé křivky a zjistit konkrétní hodnotu ve vybraném čase.



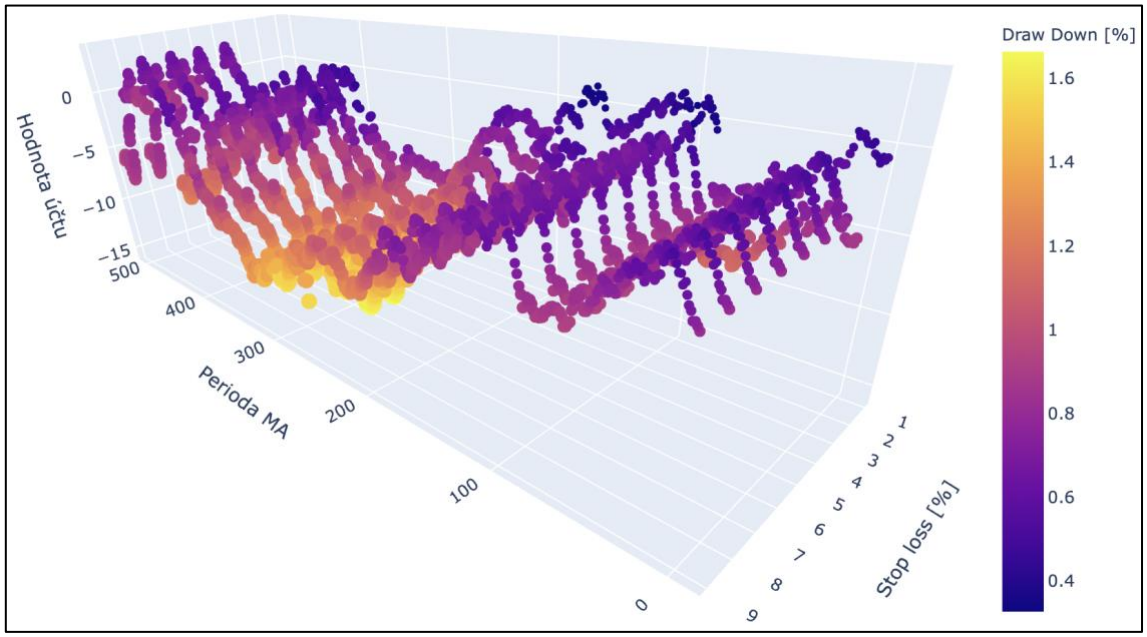
3.2.2 Strategie sentiment 0

Jako první bylo testování strategie spuštěno na datech newsGILD se 4 491 kombinacemi parametrů, konkrétně se jednalo o parametry:

- klouzavý průměr v intervalu <1, 500)
- stop-loss v intervalu <1 %, 10 %)
- take-profit = 10%

Na následujícím obrázku můžeme vidět grafický výstup:

Obrázek 16: Strategie Sentiment 0 opt. 1



Zdroj: Vlastní zpracování

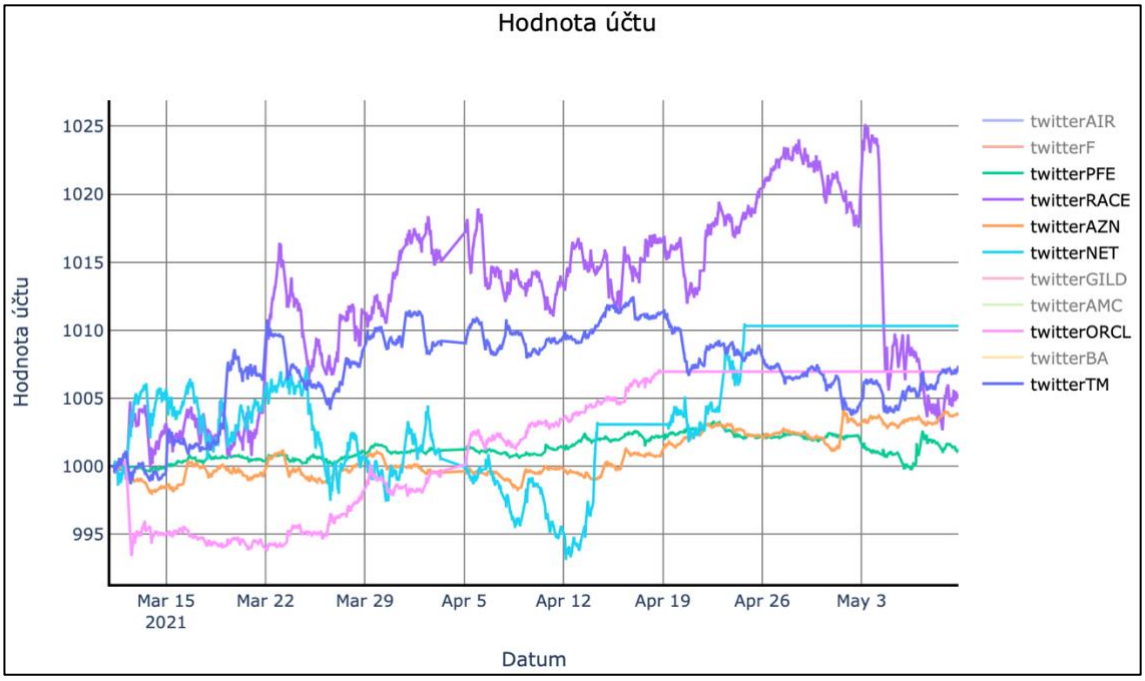
Data sloužící k vygenerování grafu byla analyzována pomocí MacOS aplikace Numbers. Byl použit filtr pro nalezení dat s periodou klouzavého průměru nižší než 100 a zobrazeny pouze ziskové výsledky, kterých bylo 24.

Pro ukázku dalšího možného využití systému pro backtesting jsme spustili optimalizaci na datech twitterORCL.csv. Ze ziskových výsledků předchozího testu byly vybrány parametry:

- klouzavý průměr = 19
- stop-loss v intervalu = 9
- take-profit = 10

S těmito parametry byla strategie spuštěna na všech datech získaných ze sociální sítě Twitter a byl vygenerován graf. V grafu jsou zobrazeny pouze testy s konečnou hodnotou portfolia vyšší, než byla hodnota počáteční. Veškeré křivky je možné zobrazit ve webové aplikaci.

Obrázek 17: Strategie sentiment 0 multistocks



Zdroj: Vlastní zpracování

3.2.3 Porovnání se strategií Buy and Hold

Porovnávat budeme výsledky provedených testů s výsledky strategie Buy and Hold, jejíž výsledky jsou vypočteny odečtením hodnoty konečné ceny akcie od počáteční hodnoty akcie pomocí funkce `final_value_to_csv()`, která generuje výsledky do příkazové řádky a je dostupná ve skriptu ***multistocks_analysis.py***. Odečíst hodnoty akcií je možné, jelikož u všech strategií máme v jednu chvíli nakoupenou, nebo případně prodanou pouze jednu akcii, z toho důvodu není nutné provádět s výsledky složitější početní operace. Veškeré číselné hodnoty zobrazené na obrázcích v této kapitole jsou v USD.

Jako první se budeme věnovat výsledkům z kapitoly 3.2.1 a to testu spuštěnému na všech datech. Konkrétní parametry můžeme nalézt pod obrázkem 14. Na následujícím obrázku vidíme, že konečná hodnota účtu testovaného se strategií Strategie MA sentimentu byla $-376, zatímco při využití strategie Buy and Hold bychom vydělali i po započtení poplatků za otevření a uzavření všech obchodů $40.

Obrázek 18: Strategie MA sentimentu

Data akcie	Konečná hodnota	Počáteční cena akcie	Konečná cena akcie
twitterF	999.5588554496763	12.780000	11.820000
twitterPFE	998.6989746551515	34.759899	39.570000
newsAMC	998.1498339729311	11.410000	9.520000
newsF	997.4791416664123	12.780000	11.820000
newsAIR	997.1045156631467	41.959999	40.369999
twitterORCL	996.4824961853013	73.040001	80.389999
newsPFE	994.8959040145871	34.759899	39.570000
newsGILD	994.2767137374872	64.400002	66.540001
twitterNET	992.0408713760374	71.160004	71.610001
newsRACE	989.6286586914063	193.729996	200.630005
newsAZN	989.4545352478035	50.020000	53.810001
twitterAZN	988.9668586730957	50.020000	53.810001
newsORCL	987.8186077270506	73.040001	80.389999
newsNET	986.9320324325565	71.160004	71.610001
twitterAIR	986.6798175888046	41.959999	40.369999
twitterTM	982.734528625489	150.830002	153.679993
twitterGILD	981.4235230560314	64.400002	66.540001
newsBA	974.912132873535	235.500107	235.470001
newsTM	956.5526189270015	150.830002	153.679993
twitterRACE	927.5623075561522	193.729996	200.630005
twitterBA	898.7840812377933	235.500107	235.470001
twitterAMC	1003.2442711124421	11.410000	9.520000

Součet konečných hodnot= -376.61872
Konečná hodnota portfolia při použití strategie Buy and Hold= 40.02797996

Zdroj: Vlastní zpracování

Na následujícím obrázku můžeme vidět výsledky Strategie sentiment 0, které byly zobrazeny v grafu na obrázku 17.

Obrázek 19: Strategie sentiment 0 Twitter

Data akcie	Konečná hodnota	Počáteční cena akcie	Konečná cena akcie
twitterAMC	998.4573622760772	11.410000	9.520000
twitterF	996.9399831619261	12.780000	11.820000
twitterAIR	995.9638580017091	41.959999	40.369999
twitterBA	992.5580185241695	235.500107	235.470001
twitterGILD	990.7833503189083	64.400002	66.540001
twitterNET	1010.3086996002198	71.160004	71.610001
twitterTM	1007.3506400146488	150.830002	153.679993
twitterORCL	1006.9474200134277	73.040001	80.389999
twitterRACE	1005.0796859130859	193.729996	200.630005
twitterAZN	1003.8101998443603	50.020000	53.810001
twitterPFE	1001.2065407409666	34.759899	39.570000

Součet konečných hodnot= 9.405758
Konečná hodnota portfolia při použití strategie Buy and Hold= 20.01398998

Zdroj: Vlastní zpracování

3.2.4 Druhá kontrolní strategie

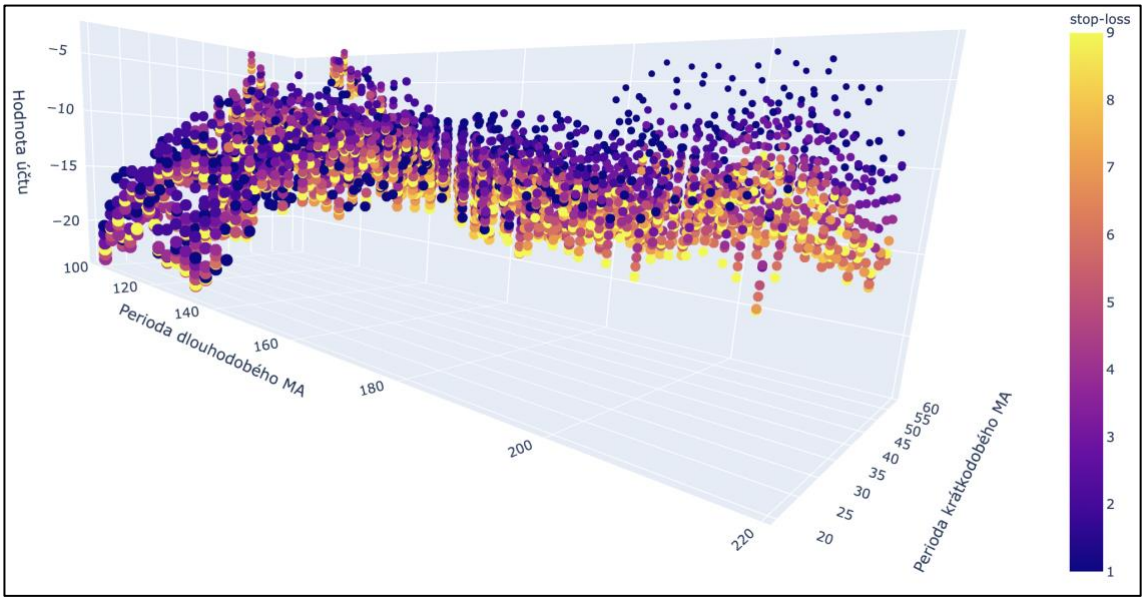
Druhá kontrolní strategie byla spuštěna s následujícími parametry:

```
cerebro.optstrategy(MA_controll_strategy,
                    period_long=range(100, 220, 3), # 3 = krok
                    period_short=range(20, 60, 3), # 3 = krok
```

```
stop_loss=range(1,10),)
```

Na následujícím obrázku můžeme opět vidět výsledky zanesené v grafu.

Obrázek 20: Druhá kontrolní strategie opt. 1



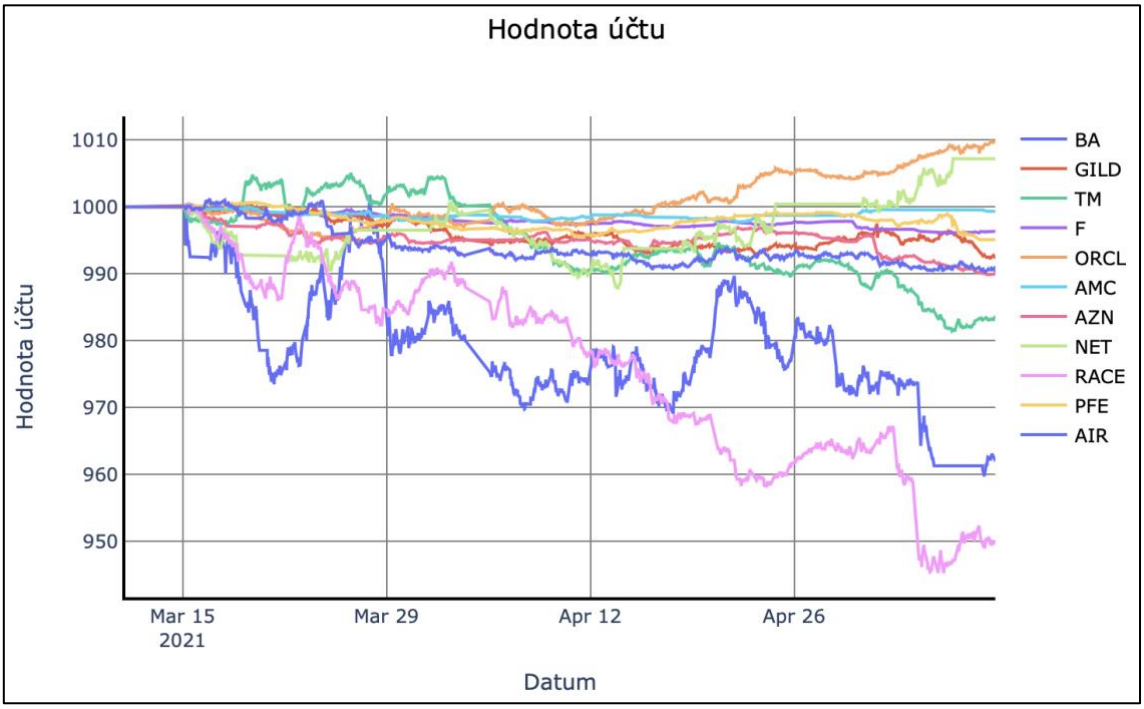
Zdroj: Vlastní zpracování

Jako poslední jsme strategii spustili se shodnými fixními hodnotami parametrů, jako v kapitole 3.2.1. Zopakujeme je:

- dlouhodobý klouzavý průměr = 238
- krátkodobý klouzavý průměr = 42
- stop-loss = 2 %
- take-profit = 8 %

Na následujícím obrázku je opět grafický výstup vygenerovaný pomocí skriptu ***multistocks_analsis.py***. Jako v každém z předešlých obrázků tohoto druhu můžeme vidět, že nejméně jedna akcie a na ní spuštěná strategie měla na konci období kladný zůstatek na obchodním účtu.

Obrázek 21: Druhá kontrolní strategie multistocks



Zdroj: Vlastní zpracování

Na posledním obrázku věnujícím se představení systému pro backtesting je zobrazen výstup v příkazové řádce po spuštění funkce `final_value_to_csv`, obsažené ve skriptu ***multistocks_analysis.py***.

Obrázek 22: Druhá kontrolní strategie 3.2.1

```
--------------------------------------------------------------------------------
              Konečná hodnota  Počáteční cena akcie  Konečná cena akcie
Data akcie
AMC          999.3259011535647             11.410000            9.520000
F            996.3318247966769             12.780000           11.820000
PFE          995.0745636138922             34.759899           39.570000
GILD         992.3042998199462             64.400002           66.540001
AIR          990.7610981140145             41.959999           40.369999
AZN          989.9672287673949             50.020000           53.810001
TM           983.6094250488278            150.830002          153.679993
BA           962.5955281677248            235.500107          235.470001
RACE         949.8637506408691            193.729996          200.630005
ORCL        1009.7832139434813             73.040001           80.389999
NET         1007.1720131225582             71.160004           71.610001
--------------------------------------------------------------------------------
Součet konečných hodnot=  -123.211153
Konečná hodnota portfolia při použití strategie Buy and Hold=  20.01398998
--------------------------------------------------------------------------------
```

Zdroj: Vlastní zpracování

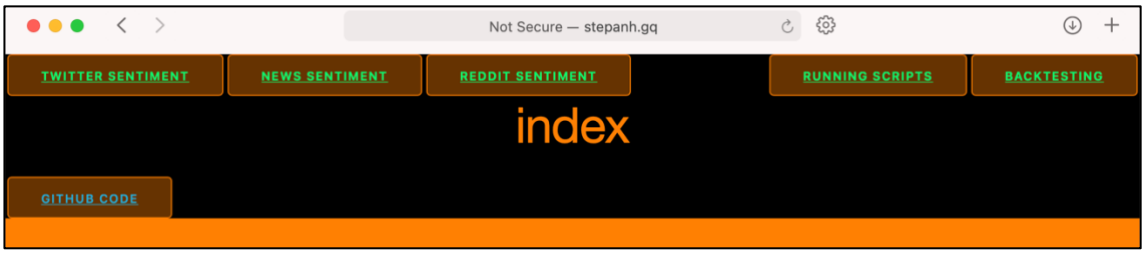
3.3 Webová aplikace

Kapitola se věnuje představení vytvořené webové aplikace pomocí snímků pořízených z internetového prohlížeče a ukazuje možné způsoby jejího využití. Webová aplikace je dostupná na adrese http://stepanh.gq, případně http://www.stepanh.gq. Správné zobrazení webové aplikace bylo testováno ve webových prohlížečích s renderovacím jádrem Webkit (Safari) a Blink (Chrome, Brave, Vivaldi).

3.3.1 Index

Jedná se o úvodní stránku, za které se pomocí tlačítkových odkazů můžeme přesměrovat na jednotlivé podstránky. Zároveň je na této stránce také odkaz na GitHub, kde jsou veřejně přístupné všechny kódy využívané jak při backtestingu, tak webovou aplikací.

Obrázek 23: Webová aplikace – Index



Zdroj: Vlastní zpracování

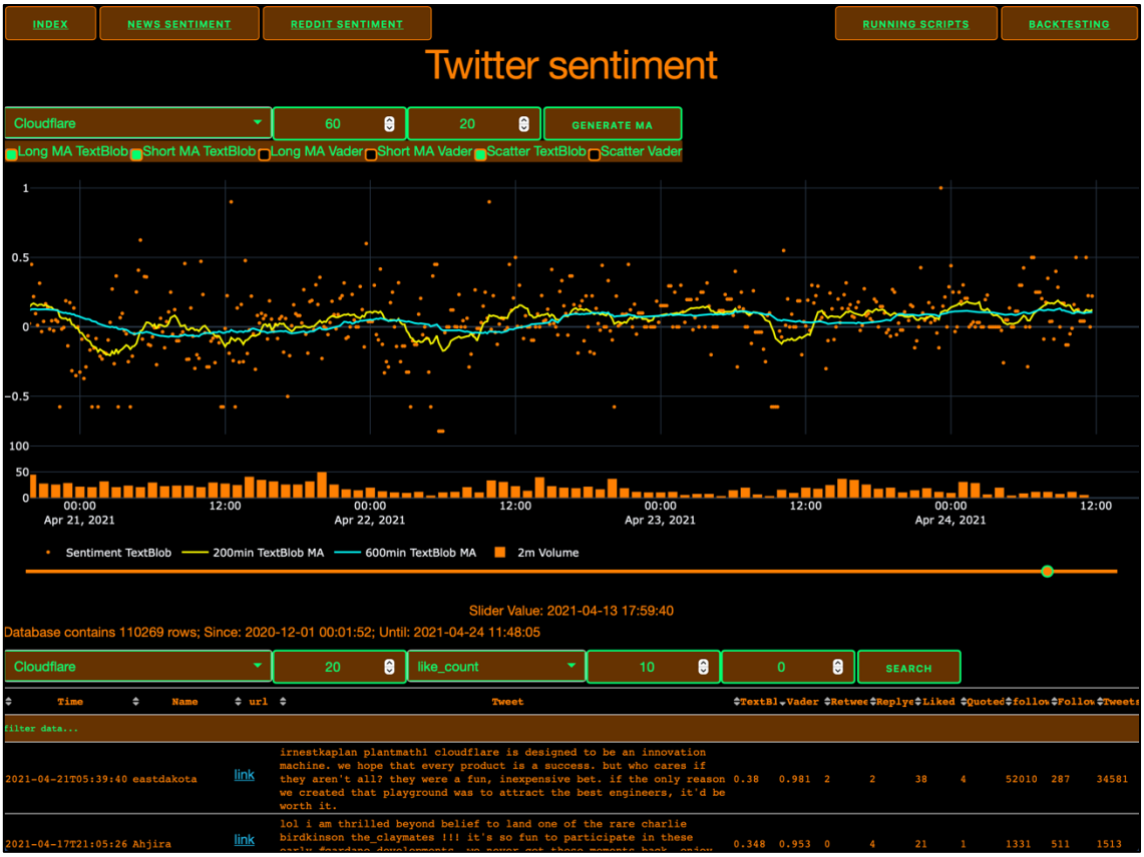
3.3.2 Twitter sentiment

Můžeme vidět vygenerované komponenty popisované v kapitole 2.2.3. Na obrázku jsou zobrazena data tweetů, která obsahují klíčové slovo Cloudflare. V grafu můžeme vidět bodový graf průměrného sentimentu za 10 minut, vypočteného pomocí knihovny TextBlob. Bodový graf byl vykreslen po vybrání možnosti Scatter TextBlob z horizontálního zaškrtávacího seznamu. V seznamu byly vybrány také možnosti Long MA TextBlob a Short MA Textblob, které slouží pro vykreslení klouzavých průměrů zadaných do příslušných polí nacházejících se pod záhlavím na pravé straně od rozbalovací nabídky. Zadané klouzavé průměry byly vygenerovány po kliknutí na tlačítko Generate MA. Mezi grafem a posuvníkem se nachází legenda grafu.

Pod hodnotou posuvníku můžeme vidět, že vybraná tabulka z databáze obsahuje 110 269 řádek. Největší tabulkou, kterou je možné v rozbalovací nabídce vybrat je Ford s 3 190 037 řádky. Vedle množství dat můžeme také zjistit datum a čas prvního (Since) a posledního (Until) záznamu v databázi. Tento text je generovaný po změně hodnoty

rozbalovacího seznamu nacházejícího se pod grafem nejblíže levému okraji webového prohlížeče. V tabulce pod poli sloužícími k vyhledávání ve vybrané tabulce jsou zobrazena data za posledních dvacet dnů. V těchto dnech bylo v každé hodině vybráno 10 nejlajkovanějších tweetů. Tabulka byla zobrazena po zadání parametrů do jednotlivých polí a kliknutí na tlačítko Search. Doba vyhledávání je závislá na velikosti příslušné databáze. Vyhledání dat zobrazených na obrázku trvá přibližně jednu vteřinu. Pokud necháme parametry stejné a zvolíme tabulku Ford, vyhledání trvá přibližně dvanáct vteřin.

Obrázek 24: Webová aplikace – Twitter Sentiment



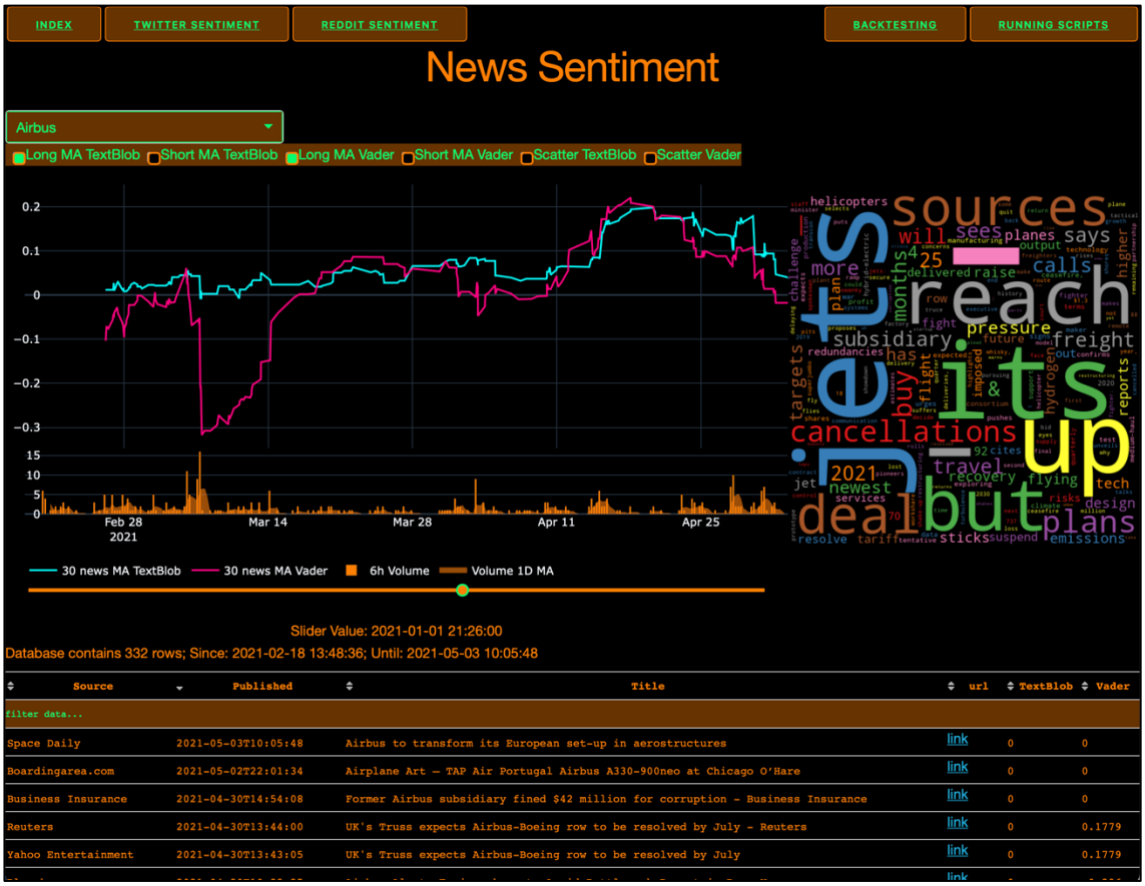
Zdroj: Vlastní zpracování

3.3.3 News sentiment

Tvorba vygenerovaných komponentů je popsána v kapitole 2.3.4. Můžeme vidět klouzavé průměry sentimentu vypočteného pomocí knihovny TextBlob a knihovny VaderSentiment. V rozbalovací nabídce je možné vybrat mezi jedenácti tabulkami a následně vygenerovat vybrané hodnoty z horizontálního seznamu. Na pravé straně se dle hodnoty posuvníku interaktivně generuje Word Cloud. Na obrázku je generován z titulků článků zveřejněných do 1.1. 2021 21:26:00. Vzhledem k tomu, že pod grafem

můžeme zjistit, že databáze obsahuje hodnoty pouze do 18.2. 2021, je Word Cloud generován ze všech řádků databáze obsahujících titulek ze zpravodajských webů. Jako poslední se na této podstránce nachází tabulka obsahující všechny řádky. Sloupce je možné filtrovat, například zobrazením pouze zpráv z webu Reuters. Současně je možné jednotlivé sloupce řadit vzestupně, případně vzestupně pomocí šipek nacházejících se v prvním řádku tabulky.

Obrázek 25: Webová aplikace – News Sentiment

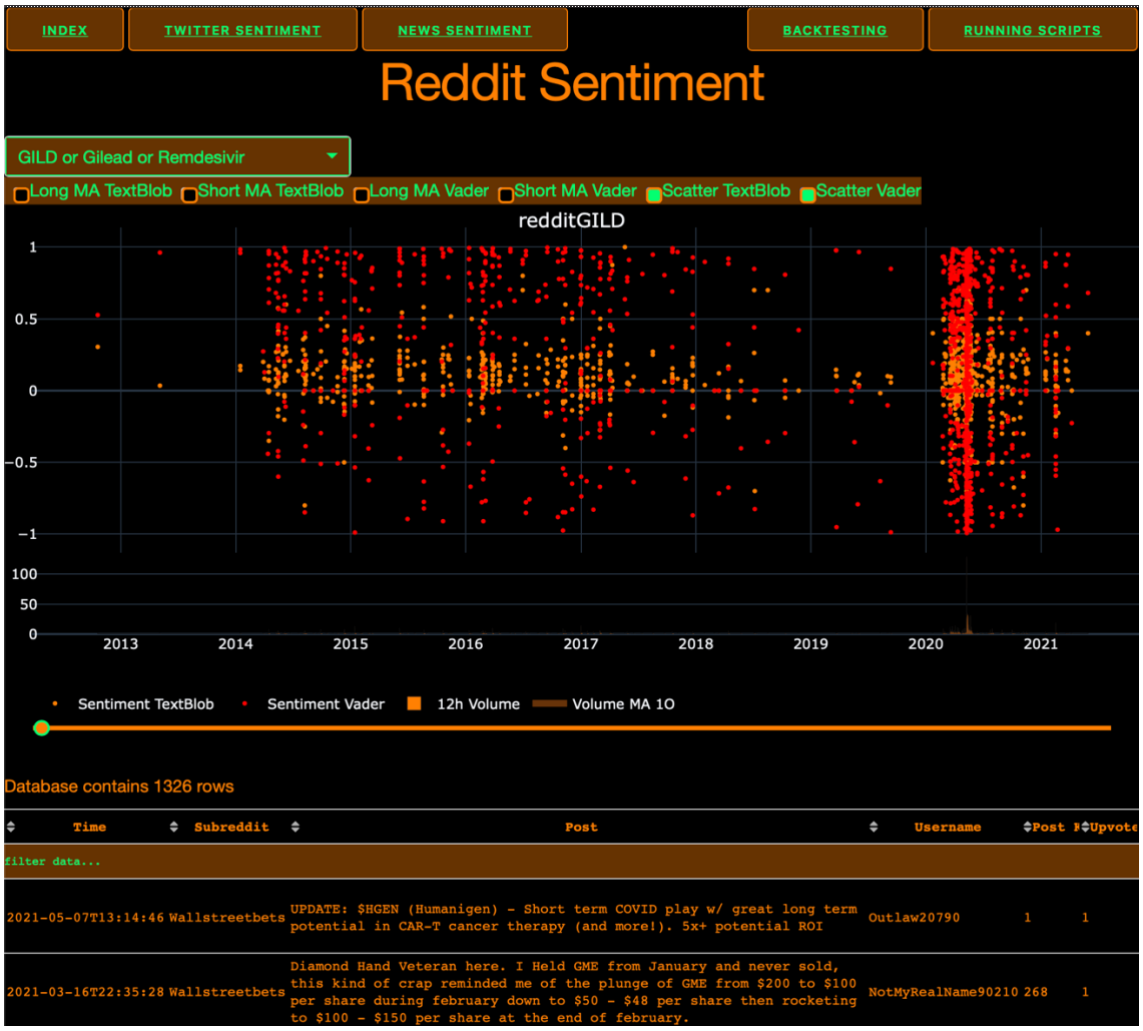


Zdroj: Vlastní zpracování

3.3.4 Reddit Sentiment

I přes to, že byla získána data pouze o jedné společnosti, webová aplikace zobrazuje zajímavá data. Na následujícím obrázku můžeme vidět výrazné zvýšení množství komentářů věnujících se společnosti Gilead. Vidíme, že nejvíce komentářů v jeden den, konkrétně 206, od roku 2014 bylo zveřejněno 17. 4. 2020.

Obrázek 26: Webová aplikace - Reddit Sentiment



Zdroj: Vlastní zpracování

Zajímavé může být, že 17. 4. 2020 dosáhla hodnota akcie jedné z nejvyšších hodnot, přibližně $85, od roku 2016. Vývoj ceny akcie GILD mezi lety 2016 a 2021 můžeme vidět na následujícím obrázku.

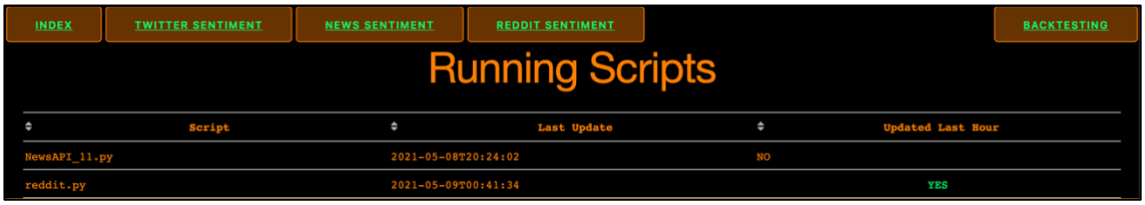
Obrázek 27: Vývoj ceny akcie GILD

Zdroj: (Trading View, 2021)

3.3.5 Running Scripts

Kapitola 2.3.5 obsahuje funkci využívanou na této podstránce. Na následujícím obrázku vidíme, že k poslednímu spuštění skriptu ***NewsAPI_11.py*** došlo 4. 5. 2021 v 22:35:30. a že skript byl spuštěn v poslední hodině.

Obrázek 28: Webová aplikace – Running Scripts



Zdroj: Vlastní zpracování

3.3.6 Backtesting

Na podstránce Backtesting je možné spustit strategie definované v kapitole 2.2.2, 2.2.3 a 2.2.4. Jako první můžeme vybrat data, vygenerovaná skriptem ***csv_data.py***, pomocí rozbalovací nabídky. Následně je možné zvolit strategii pomocí číselné hodnoty. Název strategie a další parametry se po zvolení číselné hodnoty objeví daným polem. Strategie jsou očíslovány následovně:

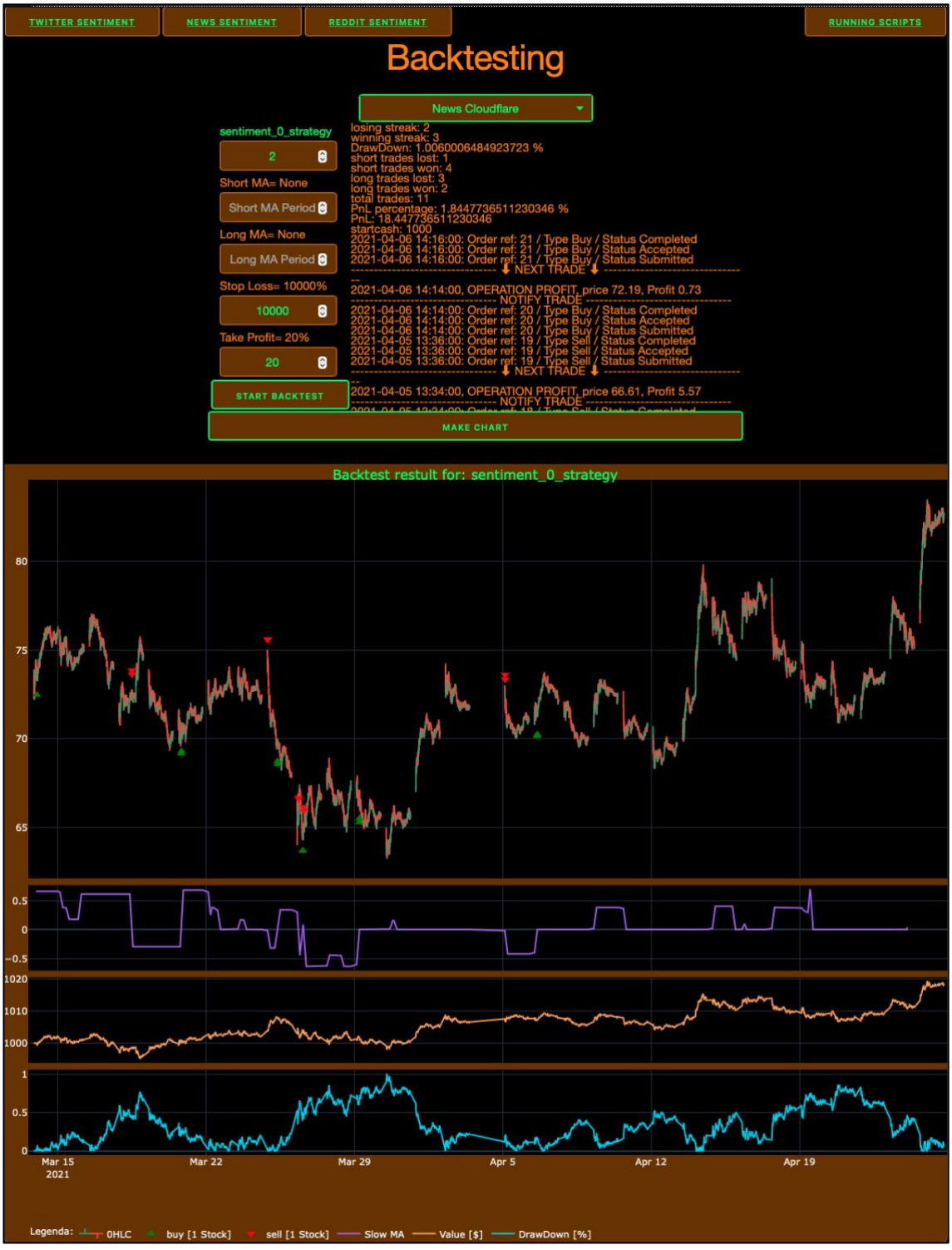
- `MA_cross_Sentiment` (kapitola 2.2.2) je vybrána po zadání hodnoty 0,
- `MA_controll_strategy` (kapitola 2.2.3) je vybrána po zadání hodnoty 1,
- `Sentiment_0_strategy` (kapitola 2.2.4) je vybrána po zadání hodnoty 2.

Na obrázku vidíme, že byla vybrána data sentimentu ze zpravodajských webů a pro backtesting byla vybrána strategie 2. Vzhledem k tomu, že tato strategie využívá pouze jeden klouzavý průměr, je potřebné zadat pouze hodnotu Short MA. Následně byly vyplněny hodnoty parametrů stop-loss a take-profit a kliknutím na tlačítko Start Backtest byl spuštěn backtesting. Jeho průběh a výsledky je možné nalézt na pravé straně od zadaných parametrů. V případě, že stránku používá více uživatelů, každý uživatel vidí stejný průběh backtestingu. Po každém kliknutí na tlačítko Start Backtest je strategie spuštěna znovu pro všechny uživatele. Backtesting pomocí webové aplikace není optimalizován pro možnost spuštění více strategií paralelně, jelikož se jedná o výpočetně relativně náročnou činnost.

Poslední tlačítko Make Chart umožňuje vygenerovat Plotly graf zobrazující průběh backtestingu. Po spuštění strategie pomocí tlačítka Start Backtest jsme o jeho ukončení

informovány zobrazením zisku, případně ztráty prostřednictvím bloku napodobujícího linuxový terminál. Po ukončení testování je nutné webovou aplikaci obnovit kliknutím na příslušné pole internetového prohlížeče. Po obnovení webové aplikace je načten nově vytvořený soubor, který slouží pro vygenerování Plotly grafu. Po obnovení webové aplikace můžeme kliknout na tlačítko Make Chart, zobrazí se pole oznamující načítání grafu a zanedlouho dojde k vykreslení grafu. Legenda vykresleného grafu je zobrazena pod grafem.

Obrázek 29: Webová aplikace – Backtesting



Zdroj: Vlastní zpracování

4 Diskuze výsledků

V průběhu vytváření jednotlivých systémů se autor práce musel vypořádat s různými problémy a omezeními, které jsou společné pro oba systémy a jejich eliminace by zvýšila kvalitu výsledků. V bodech je představíme a pokusíme se navrhnout řešení:

- **Nutnost znalosti využívaných programovacích jazyků.**
  Autor práce neměl před vypracováním diplomové práce žádné zkušenosti s programovacími jazyky využívanými pro analýzu dat. S programovacími jazyky se seznamoval v průběhu vytváření jednotlivých systémů. Je možné, že některé skripty by bylo možné formulovat jinak a lépe. I autor sám po získání větších zkušeností často skripty upravoval do vhodnější podoby. Věříme, že jak začátečník, tak zkušený programátor se dokáže zanedlouho ve skriptech zorientovat a případně generovat vlastní výsledky.
- **Limity API.**
  Bylo využíváno pouze API dostupných zdarma s různými omezeními. V případě využití jednotlivých skriptů pro reálné obchodování, které generuje zisk, by mohla být zaplacena prémiová verze jednotlivých API. Konkrétní cenu by bylo nutné vypočítat dle požadavků uživatele, ale zajisté by byla nižší než cena existujících obchodních terminálů poskytujících sentimentální data.
- **Výkon počítače.**
  Oba vytvářené systémy dokáží využít několikanásobně výkonnější hardware počítače, než je autorovi dostupný.

Dále se budeme věnovat výsledkům, problémům a omezením u jednotlivých systémů.

4.1 Backtesting

Byl vytvořen ucelený systém skriptů, které je možné využít pro zpětné testování obchodních strategií. Skript ***strategy.py*** obsahuje systém funkcí, po jejichž jednoduché modifikaci je možné měnit nastavení parametrů, nebo generovat obchodní signály pomocí jiných podmínek. Tímto způsobem bylo vytvořeno několik strategií. Můžeme diskutovat o tom, zdali se jedná o různé strategie, nebo pouze o různé metody generování obchodních signálů. Jedná se o velmi úzce propojené pojmy. V námi vytvořených strategiích je možné otáčet podmínky pro generování obchodních signálů, případně zadat nebo přidat podmínky nové. Dále je možné měnit parametry, jako například periodu

klouzavého průměru. Po nastavení parametru stop-loss na hodnotu 100 % je jisté, že nikdy nedojde k uzavření obchodu z důvodu jeho vysoké ztrátovosti.

V kapitole 3.2. byly prezentovány výsledky generované systémem pro backtesting. Existuje velké množství kombinací parametrů. Nalezení nejlepších parametrů a strategie by bylo výpočetně velmi náročné. Samotný backtesting vyžaduje velmi dobré hardwarové specifikace. Generuje velké množství dat, která jsou zapisována na disk a rychlost provedení testu závisí na počtu jader procesoru a jeho výkonu. Je možné, že skript ***optimize_strategy.py*** by bylo možné upravit tak, aby pracoval efektivněji. Při současné podobě skriptu není pro autora možné na pro něj dostupném počítači otestovat všechny strategie, se všemi kombinacemi parametrů na všech získaných datech. Z tohoto důvodu se po představení výsledků systému pro backtesting nevěnujeme vybrání nejlepší strategie a parametrů.

4.2 Webová aplikace

Webová aplikace by také benefitovala z výkonnějších komponentů serveru, které by umožnily rychlejší odezvu na uživatelem zadané požadavky. S větším počtem uživatelů připojených na webovou aplikaci celkově klesá rychlost její odezvy. Řešením by mohlo být například hostování webové aplikace na cloudovém serveru, k tomu se v dnešní době využívá například služeb Amazon Web Services. K tomuto řešení nebylo přistoupeno, jelikož předpokládáme, že využití cloudového serveru by z hlediska relativně velké výpočetní náročnosti webové aplikace nebylo levné. Webová aplikace generuje výsledky z MySQL databáze dle zadaných parametrů uživatelem. Bylo získáno velké množství metadat, která by bylo možné využít i jiným způsobem, například nebylo využito geografických dat pro vytvoření mapy zobrazující polohy uživatele při zveřejnění příspěvku na sociální síti Twitter. Webová aplikace by dokázala pracovat s mnohonásobně větším množstvím dat, pokud by byla prostřednictvím jednotlivých API získána.

Závěr

Cílem diplomové práce bylo vytvoření systémů pro podporu rozhodování při nákupu a prodeji aktiv na finančním trhu. Před vytvořením systémů byly teoreticky popsány způsoby sloužící k získávání, uchovávání a zpracování informací o finančních trzích, které byly následně implementovány. Byly také identifikovány nástroje technické a fundamentální analýzy využívané například při generování obchodních signálů. Tím byly splněny první dva dílčí cíle a mohli jsme přistoupit k vytváření systémů. Byly vytvořeny dva systémy.

Systém pro zpětné testování obchodních transakcí byl vytvořen pomocí Python knihovny Backtrader. Existuje mnoho příkladů skriptů vytvořených touto knihovnou. Přínos autora je ve vytvoření uceleného a komplexního systému s přesným popisem funkcionalit a jejich využití. Systémem je možné testovat velké množství strategií a optimalizovat různé parametry. V případném pokračování práce by bylo možné zaměřit se na výběr nejlepší strategie, parametrů, případně implementovat komplexnější systém pro výpočet poplatků a přistoupit k obchodování v reálném čase se skutečnými penězi. Dále by bylo možné do systému implementovat strojové učení a umělou inteligenci, například pomocí Python knihovny TensorTrade.

Jako druhý systém byla vytvořena webová aplikace pomocí Python knihovny Dash. V teoretické části byly popsány existující aplikace. Následně jsme přistoupili k vlastnímu řešení, které jsme se snažili odlišit od existujících aplikací. Vytvořená webová aplikace nabízí funkcionalitu, která není dostupná v žádných z identifikovaných existujících aplikacích dostupných zdarma. V případném pokračování práce by do tohoto systému bylo možné implementovat větší množství dat, rozšířit její funkcionalitu, pokusit se o rozšíření aplikace mezi větší množství uživatelů a případně některé funkce skrýt za placenou bránu.

Oba systémy pracují s daty generovanými pomocí různých API. Z hlediska získávání dat by bylo možné práci rozšířit o využití web-scrapingu. Po získání dat jsou sentimentální data vypočtena pomocí vybraných Python knihoven. V případném pokračování práce by bylo možné využít pro výpočet sentimentu například Python knihovny SpaCy a strojového učení.

Seznam použitých zdrojů

Petrovský, B. J. (22. 5 2016). Získávání a analýza textových dat pro oblast finančních trhů. Brno, Jihomoravský, Česká republika.

Mishkin, F. S. (2019). *The Economics of Money, Banking, and Financial Markets.* Global Edition: Pearson.

Jean-Philippe Bouchaud, J. B. (2018). *Trades, Quotes and Prices.* Cambridge: Cambridge University Press.

Abis, S. (2017). *Man vs. Machine: Quantitative and Discretionary Equity Management.* Columbia: Columbia University.

SQLite. (18. 6 2020). *Documentation*. Načteno z SQLite: https://www.sqlite.org/docs.html

Elmasri, R., & Navathe, S. B. (2016). *Fundamentals of Database Systems.* Pearson.

PostgreSQL Global Development Group. (25. 6 2020). *PostgreSQL*. Načteno z PostgreSQL: https://www.postgresql.org

Clement, J. (30. 4 2020). *Number of monthly active Facebook users worldwide as of 1st quarter 2020*. Načteno z Statista: https://www.statista.com/statistics/264810/number-of-monthly-active-facebook-users-worldwide/

Layton, R. (2017). *Learning Data Mining with Python.* Birmingham, Mumbai: Packt Publishing.

Han, J., Kamber, M., & Pei, J. (2012). *Data Mining Concepts and Techniques.* Amsterdam: Elsevier.

Shmueli, G., Bruce, P. C., Gedeck, P., & Patel, N. R. (2020). *Data Mining for Business Analytics.* Hoboken: Wiley.

Sarkar, D. (2019). *Text Analytics with Python.* Bangalore: Apress.

Kolakowski, M. (11. 2 2020). *Investopedia*. Načteno z Bloomberg vs. Reuters: What's the Difference?: https://www.investopedia.com/articles/investing/052815/financial-news-comparison-bloomberg-vs-reuters.asp

Clement, J. (24. 7 2020). *Most popular social networks worldwide as of July 2020, ranked by number of active users*. Načteno z Statista: https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/

Rodriguez, S. (24. 9 2019). *As calls grow to split up Facebook, employees who were there for the Instagram acquisition explain why the deal happened*. Načteno z CNBC: https://www.cnbc.com/2019/09/24/facebook-bought-instagram-because-it-was-scared-of-twitter-and-google.html

Facebook. (1. 1 2020). *Public Feed API*. Načteno z Facebook for Developers: https://developers.facebook.com/docs/public_feed/

Alexa. (11. 8 2020). *The top 500 sites on the web*. Načteno z Alexa: https://www.alexa.com/topsites

Reddit. (11. 8 2020). *About*. Načteno z Reddit: https://www.redditinc.com

Twitter. (13. 8 2020). *Tap into what's happening*. Načteno z Twitter Developer: https://developer.twitter.com/en/products/twitter-api

Twitter. (13. 8 2020). *Rate Limits*. Načteno z Twitter Developer: https://developer.twitter.com/en/docs/twitter-api/v1/rate-limits

Tweepy. (13. 8 2020). *Tweepy*. Načteno z Tweepy: https://www.tweepy.org

Linuxize. (30. 5 2020). *How To Use Linux Screen*. Načteno z Linuxize: https://linuxize.com/post/how-to-use-linux-screen/

Ellingwood, J. (20. 2 2018). *How To Use Journalctl to View and Manipulate Systemd Logs*. Načteno z Digital Ocean: https://www.digitalocean.com/community/tutorials/how-to-use-journalctl-to-view-and-manipulate-systemd-logs

Bharathi.Sv, S., & Geetha, A. (1. 6 2017). *Sentiment Analysis for Effective Stock Market Prediction.* Načteno z Research Gate: https://www.researchgate.net/publication/317214679_Sentiment_Analysis_for_Effective_Stock_Market_Prediction

Lane, H., Howard, C., & Hapke, H. M. (2019). *Natural Language Processing in Action.* Shelter Island: Manning.

Hilpisch, Y. (2014). *Python for Finance.* Sebastopol: O'Reilly.

Sarkar, D. (2019). *Text Analytics with Python.* Bangalore: Apress.

Ashraf, R. (12. 5 2017). Scraping EDGAR with Python. Atlanta, Georgia, United States.

Lucey, D. (12. 10 2020). *Finding the Dimensions of sec database. com from 2010-2020 - Part 2*. Načteno z Redwall Analytics: https://redwallanalytics.com/2020/10/12/finding-the-dimensions-of-secdatabase-com-from-2010-2020-part-2/

Financial Modeling Prep. (14. 3 2021). *Market Data Subscription Plans*. Načteno z Financial Modeling Prep: https://financialmodelingprep.com/developer/docs/pricing

Goldstein, A. (24. 1 2019). *Open Source Licenses Explained*. Načteno z White Source: https://resources.whitesourcesoftware.com/blog-whitesource/open-source-licenses-explained#MIT%20License

Hutto, C. J. (15. 3 2021). *VADER-Sentiment-Analysis*. Načteno z GitHub: https://github.com/cjhutto/vaderSentiment#resources-and-dataset-descriptions

SpaCy. (20. 3 2021). *Industrial-Strength Natural Language Processing*. Načteno z spaCy: https://spacy.io

Davda, J. (17. 1 2021). *What is Backtesting? 3 Aims of Backtesting*. Načteno z AlgoTrading101 Blog: https://algotrading101.com/learn/backtesting-guide/

Milton, A. (30. 7 2020). *Trading Order Types*. Načteno z The Balance: https://www.thebalance.com/trading-order-types-1031050#stop-orders-stp

QuantStart. (3. 30 2021). *Successful Backtesting of Algorithmic Trading Strategies - Part II*. Načteno z QuantStart: https://www.quantstart.com/articles/Successful-Backtesting-of-Algorithmic-Trading-Strategies-Part-II/

Chen, P., Lezmi, E., Roncalli, T., & Xu, J. (15. 11 2019). A Note on Portfolio Optimization with Quadratic Transaction Costs.

Heick, T. (19. 10 2020). *The Cognitive Bias Codex: A Visual Of 180+ Cognitive Biases*. Načteno z teachthought: https://www.teachthought.com/critical-thinking/the-cognitive-bias-codex-a-visual-of-180-cognitive-biases/

Formula Stocks. (12. 1 2017). *Backtesting bias and how we avoid it.* Načteno z Keeping Stock: https://keepingstock.net/backtesting-bias-and-how-we-avoid-it-fe598930cb1

Liew, L. (30. 3 2021). *Backtesting Biases and Risks*. Načteno z AlgoTrading101 Wiki: https://algotrading101.com/wiki/backtesting-biases-and-risks/

Konstantinovic, D. (9. 11 2020). *How Quantopian's open-source investment dream died*. Načteno z The Business of Business: https://www.businessofbusiness.com/articles/how-quantopian-died-shut-down-quant-investment-robinhood/

Terra, J. (19. 2 2021). *Keras vs Tensorflow vs Pytorch: Understanding the Most Popular Deep Learning Frameworks*. Načteno z Simpli Learn: https://www.simplilearn.com/keras-vs-tensorflow-vs-pytorch-article

Longmore, K. (18. 10 2015). *Benchmarking backtest results against random strategies*. Načteno z Robot Wealth: https://robotwealth.com/benchmarking-backtest-results-against-random-strategies/

Bloomberg Finance L.P. (1. 1 2021). *Bloomberg Professional Services*. Načteno z Bloomberg: https://www.bloomberg.com/professional/tech-decoded/?utm_medium=Adwords&utm_campaign=Tech&utm_source=pdsrch&utm_content=TechDecoded&tactic=435238&gclid=CjwKCAjwpKCDBhBPEiwAFgBzjwE6MHhK67QOdFu0HDMGyZSnB0s3XUHaq3perpUhj1IXWPB3ue-pcRoC82kQAvD_BwE

Ramchandani, J. (7. 5 2019). *News Sentiment Analysis with Eikon Data APIs*. Načteno z Refinitiv: https://www.refinitiv.com/perspectives/future-of-investing-trading/news-sentiment-analysis-with-eikon-data-apis/

Thinkorswim. (2. 4 2021). *Social Sentiment*. Načteno z How to thinkorswim: https://tlc.thinkorswim.com/center/howToTos/thinkManual/charts/Useful-Tools/Social-Sentiment

Thinkorswim. (2. 4 2021). *Fundamentals*. Načteno z How to thinkorswim: https://tlc.thinkorswim.com/center/howToTos/thinkManual/Analyze/Fundamentals

Fitton, N. (12. 12 2016). *How To Use Social Sentiment In Thinkorswim part 3: How To Buy & Short Stocks w/ Social Sentiment!* Načteno z YoutTube: https://www.youtube.com/watch?v=xnweDrp3jWw

Carey, T. W. (29. 1 2021). *E*TRADE Review*. Načteno z Investopedia: https://www.investopedia.com/e-trade-review-4587893

Currency.com. (4. 4 2021). *Why Currency.com*. Načteno z Currency.com: https://currency.com

XTB. (4. 4 2021). *Stock CFDs*. Načteno z XTB: https://www.xtb.com/int/shares

eToro. (4. 4 2021). *Help Center*. Načteno z eToro: https://www.etoro.com/customer-service/help/73082359/how-to-find-stocks-in-the-platform/

TradingView. (4. 4 2021). *Try any of our paid plans*. Načteno z TradingView: https://www.tradingview.com/gopro/?source=fundamentals_dialog&feature=deepFundamentalsNotification#

FinViz. (5. 4 2021). *Financial Markets Are Within Your Reach*. Načteno z Finviz: https://finviz.com/elite.ashx

FinViz. (5. 4 2021). *Screener*. Načteno z FinViz: https://finviz.com/screener.ashx?v=161&f=fa_debteq_u0.1&ft=2&o=debteq

Sentiment Viz. (5. 4 2021). *Timeline*. Načteno z Sentiment Viz: https://www.csc2.ncsu.edu/faculty/healey/tweet_viz/tweet_app/

Sentiment Trader. (5. 4 2021). *Pricing*. Načteno z Sentiment Trader: https://sentimentrader.com/pricing/

Powrbot. (5. 4 2021). *Pricing*. Načteno z Powrbot: https://powrbot.com/pricing/

Yahoo Finance. (5. 4 2021). *Yahoo Finance Plans*. Načteno z Yahoo Finance: https://www.yahoo.com/plus/finance

Choudhury, A. (4. 12 2020). *Top 10 Programming Languages Used By GitHub Repo Contributors In 2020*. Načteno z Analytics India Magazine: https://analyticsindiamag.com/top-10-programming-languages-used-by-github-repo-contributors-in-2020/

Ozimek, S. (18. 6 2018). *12 Top Python App Examples from Top-notch Companies*. Načteno z Netguru: https://www.netguru.com/blog/python-app-examples

Petlovana, Y. (1. 1 2020). *Top 13 Python Web Frameworks to Learn in 2020*. Načteno z Steel Kiwi: https://steelkiwi.com/blog/top-10-python-web-frameworks-to-learn/

Korsun, J. (6. 4 2021). *10 Popular Websites Built With D*. Načteno z Django Stars: https://djangostars.com/blog/10-popular-sites-made-on-django/

McMahon, K. (9. 2 2019). *Integrating Bokeh visualisations into Django Projects.* Načteno z Hackernoon: https://hackernoon.com/integrating-bokeh-visualisations-into-django-projects-a1c01a16b67a

Torfsen. (12. 6 2020). *python-systemd-tutorial*. Načteno z GitHub: https://github.com/torfsen/python-systemd-tutorial

Ostezer, & Drake, M. (19. 3 2019). *SQLite vs MySQL vs PostgreSQL: A Comparison Of Relational Database Management Systems*. Načteno z Digital Ocean: https://www.digitalocean.com/community/tutorials/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-management-systems

Nikolaev, K. (9. 7 2019). *Top 10 Fundamental Analysis Indicators for All Investors*. Načteno z Investor Academy: https://investoracademy.org/top-10-fundamental-analysis-indicators-for-all-investors/

Jansen, S. (2020). *Machine Learning for Algorithmic Trading.* Birmingham; Mumbai: Packt.

Tobin, D. (10. 6 2020). *Which Modern Database Is Right for Your Use Case?* Načteno z Xplenty: https://www.xplenty.com/blog/which-database/

SQLite. (10. 4 2021). *SQLite*. Načteno z Begin Concurren: https://sqlite.org/src/doc/begin-concurrent/doc/begin_concurrent.md

F., K. (2020. 9 2020). *Masking With WordCloud in Python: 500 Most Frequently Used Words in German*. Načteno z Medium: https://medium.com/swlh/masking-with-wordcloud-in-python-500-most-frequently-used-words-in-german-c0e865e911bb

Edwards, R. D., Magee, J., & Bassetti, W. (2018). *Technical Analysis of Stock Trends.* London: Taylor & Francis Ltd.

Trading View. (8. 5 2021). *GILD Stock Price and Chart*. Načteno z Trading View: https://www.tradingview.com/symbols/NASDAQ-GILD/

Seznam tabulek

Seznam obrázků

Seznam použitých zkratek

AIR	Airbus
AMC	AMC Entertainment Holdings
ANEW	Affective Norms for English words)
API	Application Programming Interface
AZN	AstraZeneca
BA	Boeing
CLI	Command Line Interface
CSS	Cascading Style Sheets
CSV	Comma-separated values
CUDA	Compute Unified Device Architecture
F	Ford Motor
GB	Gigabyte
GI	General Inquirer
GILD	Gilead Sciences
GUI	Graphical User Interface
HFT	High-frequency trading
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IBM	International Business Machines Corporation
IDE	Integrated Development Environment
LIWC	Linguistic Inquiry and Word Count
MA	Moving Average
MACD	Moving Average Convergence Divergence
MB	Megabyte
MIT	Massachusetts Institute of Technology

NET	Cloudflare
OHLC	Open, High, Low, Close
ORCL	Oracle
OTC	Over the Counter
PDF	Portable Document Format
PFE	Pfizer
PRAW	Python Reddit API Wrapper
RACE	Ferrari
RAM	Random Access Memory
RDBMS	Relational Database Management Systems
S&P 500	Standard & Poor's Index
SEC	Securities and Exchange Commission
SQL	Structured Query Language
SSH	Secure Shell
TM	Toyota
TXT	Text
URL	Uniform Resource Locator
UTC	Coordinated Universal Time
WSGI	Web Server Gateway Interface
XBRL	Extensible Business Reporting Language

Abstrakt

Havlovic, Š. (2021). *Finanční systémy pro podporu rozhodování* (Diplomová práce), Západočeská univerzita v Plzni, Fakulta ekonomická, Česko.

Klíčová slova: Obchodní strategie, Backtrader, obchodní signály, webová aplikace, Dash, MySQL, Python, sentiment, zpravodajské weby, sociální sítě.

Diplomová práce se zabývá vytvořením systémů pro podporu rozhodování při nákupu a prodeji aktiv na finančním trhu. Téma bylo vybráno z důvodu zájmu autora o prohloubení jeho znalostí o velkých datech a zpětném testování obchodních strategií. Pro řešení problému je využíváno zejména programovacího jazyku Python a analýzy sentimentu. Byly vytvořeny dva systémy. Prvním je systém sloužící pro zpětné testování obchodních strategií, který by mohlo být možné využít i při reálném obchodování. Druhým systémem je webová aplikace prezentující získaná data a slouží pro podporu rozhodnutí investora obchodujícího diskrečně.

Abstract

Havlovic, Š. (2019). *Financial Decision Support Systems* (Master's Thesis). University of West Bohemia, Faculty of Economics, Czech Republic.

Key words: Trading strategy, Backtrader, trading signals, web application, Dash, MySQL, Python, sentiment, news sites, social networks.

The master's thesis deals with the creation of systems to support financial decision-making about purchasing or selling an asset on the financial market. The topic was chosen due to the author's interest in deepening his knowledge about big data and backtesting of trading strategies. Sentiment analysis with the Python programming language is used to solve the problem. Two systems were created. The first is used for backtesting of trading strategies which could be possible to use in live quantitative trading. The second is a web application for presenting the obtained data that serves to support the decision of an investor who is trading discretionary.