Replace the contents of this file with official assignment.
Místo tohoto souboru sem patří list se zadáním závěrečné práce.

Bakalářská práce

WEBOVÁ APLIKACE PRO SLEDOVÁNÍ VÝVOJE CEN KRYPTOMĚN.

Jan Koten

Fakulta informačních technologií
Katedra teoretické informatiky
Vedoucí: Ing. Stanislav Kuznetsov
15. dubna 2022

Obsah

Seznam obrázků

Seznam tabulek

Seznam výpisů kódu

Chtěl bych poděkovat především sit amet, consectetuer adipiscing elit. Curabitur sagittis hendrerit ante. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Cras pede libero, dapibus nec, pretium sit amet, tempor quis. Sed vel lectus. Donec odio tempus molestie, porttitor ut, iaculis quis, sem. Suspendisse sagittis ultrices augue.

Prohlášení

FILL IN ACCORDING TO THE INSTRUCTIONS. VYPLŇTE V SOULADU S POKYNY. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Curabitur sagittis hendrerit ante. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Cras pede libero, dapibus nec, pretium sit amet, tempor quis. Sed vel lectus. Donec odio tempus molestie, porttitor ut, iaculis quis, sem. Suspendisse sagittis ultrices augue. Donec ipsum massa, ullamcorper in, auctor et, scelerisque sed, est. In sem justo, commodo ut, suscipit at, pharetra vitae, orci. Pellentesque pretium lectus id turpis.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Curabitur sagittis hendrerit ante. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Cras pede libero, dapibus nec, pretium sit amet, tempor quis. Sed vel lectus. Donec odio tempus molestie, porttitor ut, iaculis quis, sem. Suspendisse sagittis ultrices augue. Donec ipsum massa, ullamcorper in, auctor et, scelerisque sed, est. In sem justo, commodo ut, suscipit at, pharetra vitae, orci. Pellentesque pretium lectus id turpis.

V Praze dne 15. dubna 2022 .....

Abstrakt

Tato práce se zabývá problematikou predikce vývoje hodnoty kryptoměn. V rámci ní je vytvořena webová aplikace poskytující uživateli informace, které mu usnadní vydělávání na trhu s kryptoměnami. Část těchto informací je tvořena predikcemi vývoje hodnot sledovaných kryptoměn. Data pro trénování predikčního modelu jsou periodicky stahována z internetu pomocí nástrojů pro scraping

Klíčová slova predikce hodnoty kryptoměn, webová aplikace, scraping webu, google trends, nestandardní ukazatele, python, tensorflow, vue.js

Abstract

This work deals with the problem of predicting the development of the value of cryptocurrencies. Within it, a web application is created that provides users with useful information making it easier for them to make money on the cryptocurrency market. Part of this information is formed by predictions of the development of the values respectively to the monitored cryptocurrencies. Data for training the prediction model are periodically downloaded from the Internet using scraping tools.

Keywords cryptocurrency value prediction, web application, web scraping, google trends, non-standard indicators, python, tensorflow, vue.js

Seznam zkratek

DFA	Deterministic Finite Automaton
FA	Finite Automaton
LPS	Labelled Prüfer Sequence
NFA	Nondeterministic Finite Automaton
NPS	Numbered Prüfer Sequence
XML	Extensible Markup Language
XPath	XML Path Language
XSLT	eXtensible Stylesheet Language Transformations
W3C	World Wide Web Consortium

Úvod

Kryptoměna je digitální forma měny. Jedná se o prostředek směny využívaný v online světě a její největší předností je decentralizovanost. Není regulovaná ani udržována žádnou centrální autoritou, jako jsou například vládní orgány nebo banky. Záznamy o vlastnictví a transakcích s kryptoměnami jsou uloženy v distribuovaných databázích. Typicky se jedná o využití technologie blockchain. Kryptoměny jsou kryptograficky chráněny a pro validaci používají metody jako proof-of-stake, při kterých se ověří platnost historie všech transakcí. Validace transakcí kryptoměn je zajištěna blockchainem, tedy neustále rostoucím seznamem navzájem propojených záznamů. Každý ze záznamů transakcí obsahuje hash odkazující se na předchozí záznam. Tímto způsobem se stávají odolné vůči neoprávněné modifikaci dat. Behem posledních let se začalo rozšiřovat povědomí o kryptoměnách prostřednictvím internetu i mimo něj. Umožňují způsob, jak obejít pokles hodnoty oficiální státní měny a zároveň poskytuje uživatelům lepší formu anonymity při provádění transakcí. Nejen Bitcoin, jakožto první veřejně známá kryptoměna totiž od svého vzniku výrazně získal na své hodnotě. Stejně tak postupně získávaly na hodnotě i ostatní kryptoměny. Od jejich vzniku počínaje kryptoměnou Bitcoin v roce 2009 se jedinci, komunity i společnosti snažili jejich prostřednictvím finančně obohatit. Přímým vlivem růstu jejich hodnoty začaly vznikat národní i nadnárodní společnosti, z nichž mnoho kapitalizovalo právě na vývoji hodnoty kryptoměn. Začaly vznikat kryptoměnové burzy, které nabízeli uživatelům velice efektivní způsob obchodování a změny hodnoty jejich konta.[1, 2, 3, 4]

Vstup na finanční trh může být pro mnohé nové obchodníky skličující zkušenost. Investování do vybraných asetů se nám nemusí vždy vyplatit. Pro dosažení výdělku při obchodování s kryptoměnami, je nutné správně odhadnout trend, kterým se daná kryptoměna ubírá. Tuto informaci je možné odhadnout z pouhého sledování dlouhodobého trendu kryptoměny. Pro dosažení nejoptimálnějších výsledků však je potřeba využít co nejvíce zdrojů, které mají prokazatelně vliv na vývoj hodnoty. Přístup k této problematice rozhodně není jednotný a existuje tedy mnoho velice odlišných řešení, se kterými se na kryptoměnový trh přistupuje.

Hledání optimálního přístupu k vydělávání na trhu s kryptoměnami je předmětem řešení této práce. Na základě analýzy trhu a současných řešení vytvoříme systém, který nám bude schopný poskytovat informace maximalizující naší šanci na výdělek. Výsledek a také důvod našeho snažení by pak měla být aplikace, která bude pro své uživatele zdrojem konstantního příjmu.

V následných kapitolách budeme analyzovat technologie využívané v současné době na finančním trhu. Nebude se jednat o postupy používané pouze při obchodování s kryptoměnami. Budeme se soustředit zpravidla na přístupy, které mají v našem poli zájmu potenciál využití. Následně budeme analyzovat technologie a metody, které v naší aplikaci využijeme v závislosti na jejích požadavcích na funkcionality, hardware a software. Jednotlivě budeme konstruovat systémy pro získávání dat z internetu, zpracování, predikci a zobrazování dat a po ukončení vývoje jednotlivých částí aplikace se budeme věnovat vytváření běhového prostředí pro jejich provoz a vzájemnou komunikaci.

Cíl práce

Fakt, že pro obchodování na trhu s kryptoměnami neexistuje žádný jednoznačný přístup, který by nám mohl garantovat výdělek pro nás znamená, že se na základě prozkoumání současných dostupných řešení můžeme pokusit o konstrukci našeho vlastního. Cílem práce je vytvoření jednoduché webové aplikace, která by zobrazovala vývoj ceny krytoměny pomocí nestandartních indikátorů, jako je například počet historických dotazu na kryptoměnu v Google Search, historický vývoj zlata nebo korelace kryptoměn v čase. Tyto indikátory ukážou na komplexní analýzu chování kryptoměny v minulosti. Dále budou prostřednictvím grafické části aplikace poskytovány vhodné rady ve smyslu nákupu, prodání a nečinosti v rámci jednotlivých kryptoměn. Na základě získaných dat o sledovaných kryptoměnách bude provedena predikce vývoje jejich cen. Budeme provádět rešerši současných trendů v alternativních indikátorech popisujících chování kryptoměn.

V rámci naší aplikace budeme také aktivně a periodicky získávat informace pro predikci z námi zvolených internetových zdrojů. Tyto informace budou po vhodném zpracování poskytovány uživateli v rámci webového grafického uživatelského rozhraní. Jejich poskytování bude realizováno v reálném *real* čase nebo *near real* čase se zpožděním a periodou obnovy zobrazení nových informací maximálně v jednotkách minut. Tyto informace se budou ukládat do databázového systému a budou dále zpracovány do podoby, ve které z nich bude uskutečněna predikce natrénovaným predikčním modelem. Především bude kladen důraz na zobrazování relevantních informací v dostatečně přehledném měřítku a také na jejich přesnost. Informace signalizující, že by měl uživatel ihned nakoupit nebo prodat kryptoměnu by měly vést k výdělku pro libovolně zvolenou kryptoměnu. Lze tedy počítat s tím, že nebude vybrána strategie, která by dosahovala nejlepšího výsledku napříč testovanými daty, ale ta, která bude dosahovat nejkonzistentnějšího výsledku.

Před konstrukcí samotného programu je nezbytné správně specifikovat a rozebrat požadavky na jeho funkce a užití. Bude třeba řešit běhové prostředí aplikace s ohledem na uživatelské, softwarové a hardwarové požadavky. Rozhodování pro jednotlivé akce bude realizováno na základě predikcí vyhodnocených zvoleným predikčním modelem. Na základě této predikce ke každé uživatelem specifikované kryptoměně bude možné učinit informovaný odhad vývoje kryptoměny a uskutečňovat transakce, které mu budou dlouhodobě přinášet zisk.

Kapitola 1

Současné přístupy k problematice obchodování s kryptoměnami

Obchodování s kryptoměnami může být jen zřídka kdy interpretováno jako forma lehkého výdělku. Samozřejmě se nechají najít výjimky v tomto tvrzení. Tím bývají zpravidla jedinci, kteří byli schopni, ať už cíleně nebo náhodou, zakoupit kryptoměnu před jejím náhlým zvýšením její hodnoty a zajistit si takto nemalý výdělek. Většinu času nelze pouze spoléhat na šťastnou náhodu a je třeba využít patřičných postupů, které zvýší šanci na zisk z našeho obchodování.

V této kapitole se seznámíme se současnými technologiemi používaných při obchodování na trhu s kryptoměnami, ale také s přístupy na ostatních trzích. Těmi budou například trhy akciové nebo se směnou cizích měn. Budeme se snažit analyzovat co nejvíce rozdílných přístupů, abychom mohli snáze určit náš vlastní způsob řešení, které bude stavět na úspěších zkoumaných prací.

Vývoj kryptoměn je velice nestálý. Hodnota je ovlivněna podílem měny, která se nachází v oběhu a částkou, kterou je za ní nakupující strana ochotná zaplatit. Jedná se tedy o vliv nabídky a poptávky, ale také o sentiment uživatelů a investorů, vládní nařízeni či regulace a mediální informace. Mnoho měn včetně Bitcoinu má konečný počet mincí, které budou kdy vydány. Lze očekávat, že čím blíže se počet mincí kryptoměn bude blížit ke svému konečnému počtu, tím bude jejich hodnota stoupat.[5]

Na základě této skutečnosti se ukazuje nemožné nalézt jednotný způsob, který by spolehlivě zaručil výdělek. Jednotlivé přístupy se často velice liší. Základní myšlenka však vždy zůstává stejná. Je třeba nakoupit kryptoměnu za menší cenu, než za kterou jí prodáme, abychom na ní mohli vydělat. Je také možné nalézt rozdílné kurzy kryptoměn mezi jednotlivými burzami a tedy transakcemi mezi nimi dosáhnout zisku. Proto se postupy při obchodování mohou odvíjet od společných základních principů, u kterých je obecně možné za specifických podmínek dosáhnout zisku. Nejpoužívanější z nich jsou HODL, HFT a Obchodování na základě predikce.

1.1 HODL

HODL je pojem odvozený od překlepu anglického slova „hold“. Jedná se o strategii, při které uživatel zakoupí kryptoměnu a dlouhodobě si jí ponechá bez ohledu na její současnou hodnotu. Prodej nastává pouze ve chvíli, kdy se její hodnota značně odchýlí od její kupní ceny. Tento dlouhodobý přístup je volen zejména ve chvíli, kdy se uživatelé chtějí co nejvíce oprostit od aktivního sledování vývoje ceny kryptoměny. Je volen zejména jedinci bez předchozích zkušeností s obchodováním kvůli jejich sníženým schopnostem správně načasovat výhodnou koupi nebo prodej. V tomto způsobu obchodování nefigurují žádné algoritmy nebo predikční modely a jedná se o nejjednodušší formu obchodování. Zároveň ale ne o nejvýnosnější.[6]

1.2 HFT

Vysokofrekvenční obchodování je metoda obchodování, která, jak už z názvu vypovídá, využívá množství příkazů k transakci provedených za sebou i v rozmezí jedné vteřiny. Využívá analýzy mnoha různých trhů a kryptoměnových burz. Při tomto typu obchodování platí, že obchodníci s vyšší rychlostí a získáváním dat dosahují více ziskových transakcí a celkově jsou tedy úspěšnější než ti s pomalejším prováděním transakčních operací. Pokud algoritmus určený k analýze trhu objeví posloupnost transakcí, která vede k pozitivnímu zisku, neprodleně tuto transakci provede. Tento způsob obchodování dodává trhům likviditu a eliminuje malé rozpětí mezi nabídkou a poptávkou. Jedná se o částku, o kterou vyvolávací cena převyšuje nabídkovou cenu kryptoměny na trhu. Jinými slovy jde o nejvyšší cenu, kterou je kupující ochoten za kryptoměnu zaplatit a nejnižší kupní cenou, kterou je prodávající ochoten přijmout. Přístup vysokofrekvenčního obchodování začal nabírat na popularitě, když burzy, jako například Newyorská burza cenných papírů, začaly vybízet společnosti, aby na trh přidali likviditu. Ta ve své podstatě udává, jak efektivně lze kryptoměnu, nebo jakékoliv jiné aktivum, či cenný papír směnit za hotovost, aniž by byla ovlivněna původní tržní cena[7]. Vysokofrekvenční obchodování mělo prokazatelně pozitivní vliv na likviditu trhu a snížilo rozpětí mezi nabídkou a poptávkou, která se kryptoměn týkala. Toto bylo otestováno například ve studii, která zkoumala trh po zavedení poplatků za transakce při vysokofrekvenčním obchodování, což mělo na vydělávání touto metodou negativní vliv. Studie zjistila, že tento krok zvýšil rozdíly mezi nabídkou a poptávkou o 13%.[8, 9]

Vysokofrekvenční obchodování má však svá úskalí a to především pro jednotlivé obchodníky. Tento přístup vyžaduje vysokorychlostní připojení pro rychlý a efektivní zisk informací. Je také důležité uvažovat i o čase, který trvá přenesení informace od serveru k uživateli. Ten by měl být připojený geograficky co nejblíže k serveru, ze kterého získává informace. Samotný přístup k informacím je také velice omezený. Poskytování informaci v reálném čase a v dostatečném množství je vždy buď zpoplatněno nebo přinejmenším penalizováno časem před odeslání informace nebo množstvím informací. Uživatel dále potřebuje výkonný server pro nalezení vhodných transakcí, které má uskutečnit pro zajištění zisku a to i v případě, že se rozhodne za poskytování dat platit namísto složitého obcházení omezení od poskytovatelů těchto dat.

Vysokofrekvenční obchodování je často předmětem kritiky. Vysoká rychlost obchodování může být teoreticky schopná způsobit velký a zdánlivě bezdůvodný pohyb na trhu. V roce 2010 zažil 6. května akciový trh náhlý pokles Dow Jones Industrial Average o 1.5%. Zpočátku nevysvětlitelný pokles byl předmětem vládního vyšetřování, jehož výsledkem byl nález vysoké objednávky, která tento pokles vyvolala.[10]

1.3 Predikce

Nejčastěji využívanou metodou pro obchodování s kryptoměnami je provádění transakcí na základě predikce vývoje hodnoty kryptoměn. Obchodník v tomto případě využívá predikční model, který je schopný s dostatečnou přesností určit požadované vlastnosti kryptoměny v budoucnosti. Tím je nejčastěji predikce následného stoupání či klesání. Čím je využívaný model přesnější, tím je větší šance, že uživatel na zadané transakci vydělá. Problém zde je primárně ve vytvoření takového modelu, který by disponoval požadovanými vlastnostmi. V první řadě je zapotřebí definovat, jaké výsledky by měl přesně model poskytovat. Také je potřeba zvolit a předzpracovat data, na základě kterých bude model výsledky predikovat. V neposlední řadě musí být zvolen samotný predikční model, který se bude k predikci využívat. Po získání predikcí ze zvoleného modelu je následně nutné zvolit vhodnou strategii, se kterou se bude obchodovat na základě získaných informací.

Při přistupování k obchodování s kryptoměnami pomocí predikcí můžeme narazit na pojem teorie náhodné chůze. Ta uvádí, že ceny budoucích aktiv jsou náhodné a nejsou ovlivněny minulými událostmi. Tímto tvrzením se zamítá myšlenka, že by předchozí události měly mít

vliv na současný vývoj hodnoty a odráží se v ní pouze současný stav a informace. Toto tvrzení bylo zkoumáno například u vývoje hodnoty Bitcoinu. Analyzovalo se časově proměnlivé chování dlouhodobé paměti výnosů z Bitcoinu pomocí Hustova exponentu. Od roku 2014 bylo zjištěna nízká forma efektivnosti, tedy naznačuje chování podle teorie náhodné chůze. Toto mimo jiné znamená, že denní vývoj cen může být sám na sobě zcela nezávislý. Je tedy potřeba hledat vnější závislosti na děje s kryptoměnami spojené pro zvýšení přesnosti predikcí. Existují ovšem ale predikční postupy využívající jiné nástroje a modely než ve zmíněné studii, které tvrzení o nulové závislosti vyvrací.[11]

1.4 Využívané metody pro obchodování

1.4.1 Binární klasifikační problém

V práci „Short-term bitcoin market prediction via machine learning"[12], se řešil binární klasifikační problém. Výsledek tedy nabýval dvou hodnot 1 nebo 0. Tato hodnota se určila na základě následující funkce:

$$y_t^m = \begin{cases} 1, & \text{pokud } r_{t+m,t}^{cena} \geq Med(r_{u+m,u}^{cena}) \forall u \in \{train\} \\ 0, & \text{jinak} \end{cases}$$

V zadání je m velikost intervalu neboli počet minut, v rámci kterého se výsledná hodnota počítá. Výsledek pak v čase t spadá do třídy *1*, pokud jeho hodnota přesahuje medián hodnoty v tomto časovém intervalu. V opačném případě spadá do třídy *0*. Pro predikci byly využity data ze serveru Bloomberg, příspěvky ze sociální sítě Twitter, články ze serveru blockchain.com, ceny pohonných hmot a dále indexy S&P 500, VIX a MSIC. Z tweetů a novinových článků byl zpracován sentiment, který se následně použil při predikci společně s ostatními daty. Následně byla pomocí zadané funkce analyzována krátkodobá předvídatelnost trhu s kryptoměnou Bitcoin. V rámci této kryptoměny byly zkoumány různé efekty ovlivňující její cenu v tomto časovém horizontu. Zároveň bylo pro predikci využito více typů modelů. Jednalo se o modely s paměťovou funkci LSTM a GRU, ale také modely postavené na stromové struktuře jako například Random Forest nebo regresní modely, které paměťovou funkcí nedisponují a jejich predikce se odvíjí pouze od jednoho datového bodu.

Bylo zjištěno, že pro každý horizont a každý model byla úspěšnost predikce větší než 50% a tedy by tento postup mohl být použitelný v praxi. Dále z predikcí pro rozdílné časové horizonty vyplývá, že přesnost se zvyšuje pro zvyšující se časové intervaly. Predikce vývoje pro příští hodinu byla přesnější, než predikce pro příští minutu.

Pro samotné obchodování s kryptoměnou byl zvolen přístup pomocí kvantilů. Vypočítají se 99% kvantily pravděpodobností všech intervalů. Z výsledných dat bylo odvozeno, že pokud je předpokládaná pravděpodobnost v testovací sadě pro intervaly pod 10 minut větší, než pro ostatní, zaujímá se *long* pozice a měl by tedy nastat nákup před růstem. V opačném případě jde o *short* pozici a měl by se případně uskutečnit prodej, nebo zapůjčení aktiv a poté jejich splátka za nižší obnos. Na konci predikčního horizontu byla pozice uzavřena.

Kromě přesnosti byl zároveň měřen i vliv jednotlivých příznaků dat určených k predikci. Bylo zjištěno, že největší vliv na predikci má samotná hodnota kryptoměny společně s počtem transakcí s touto kryptoměnou v čase a počet tweetů. Samotný sentiment má na predikční přesnost pouze minimální vliv a u modelu GRU je hodnocen jako nejhorší zdroj informací pro predikci.

1.4.2 ARIMA

V některých přístupech, jako například v práci „Bitcoin Price Prediction: An ARIMA Approach"[13], se pro predikci volí ARIMA model. Zde se předpovídal trend kryptoměny Bitcoin v

následujícím dnu po použitém časovém okně vstupních dat. Obecně je postup s tímto modelem považována za zastaralý. Volí se zde algoritmický přístup pro predikci budoucích hodnot na základě předešlých. Využívá klouzavý průměr k vyhlazení dat časových řad. Model implicitně předpokládá, že budoucí data se bude podobat těm minulým.

V rámci „Seq 2 Seq RNNs and ARIMA models for Cryptocurrency Prediction : A Comparative Study“ pak byl prováděn výzkum, který vyšetřoval rozdíly mezi ARIMA modelem a RNN. Bylo zjištěno, že na krátké časové intervaly může být predikce pomocí ARIMA modelu přesnější, než použité RNN.

ARIMA vyžaduje pro své predikce data, která jsou stacionární. Stacionarita dat časových řad znamená, že tyto data se v průběhu času nemění. Jinými slovy jsou data nestacionární, pokud je u nich možné pozorovat trend nebo sezónnost. Převedením dat na jejich stacionární ekvivalenty dosáhneme konzistence průměru a rozptylu dat v čase. Předpoklad stacionarity je využíván v mnoha predikčních postupech včetně ARIMA modelu a množství dalších z něj benefituje. Pro ověření stacionarity a pomoc při převádění dat na stacionární se většinou volí autokorelační a parciálně autokorelační funkce ACT/PACF. Při použití modelu ARIMA se můžeme setkat s komplikacemi v podobě nalezení správných parametrů modelu pro zlepšení jeho predikčních schopností. Tomuto hledání lze však předejít zvolením automatického hledání ideálních parametrů na základě vstupních dat.

1.4.3 Analýza sociálních médií

Práce „Advanced sentiment analysis of social media for short-term cryptocurrency price prediction“[14] se zabývala neustálým vývojem ceny kryptoměn a vnějšími vlivy, které na ní mají dopad. Testoval se vliv sociální sítě Twitter jakožto marketingového nástroje, vliv četnosti vyhledávání na webu prostřednictvím Google Search Engine. Tyto data lze stahovat ze serveru Google Trends. Data ze sociální sítě Twitter byly využity pro analýzu sentimetu trhu a na základě získaných dat byla predikována budoucí hodnota pomocí jednoduchých modelů například se zapojením lineární a hřebenové regrese bez paměťových bloků. Z provedených experimentů vychází jasná spojitost mezi vývojem hodnoty Bitcoinu, sentimentem ze sociální sítě Twitter a dat o vyhledávání z Google Trends, ty byly vyhodnoceny jako nejlepší zdroj informací pro predikci společně s negativním sentimentem. Závěrem byl prokazatelný psychologický faktor obchodníků hrající roli při vývoji hodnot na trhu. Zároveň byly identifikovány nedostatky v predikcích touto metodou. Těmi byly zejména politická rizika spojená s omezeními vztahujícími se ke kryptoměnám a bankovní omezení.

1.4.4 CNN-LSTM

Při zvolení predikce jako rozhodujícího faktoru pro obchodování s kryptoměnami můžou obchodníci často narazit na problém s daty. Primárně se jedná o způsob jejich získávání, či jejich nedostatek. Zjednodušeně platí, že se zvýšením počtu trénovacích dat jsme schopní zkonstruovat přesnější a robustnější model, který je odolný vůči přeučení. Pokud ale data je vůbec možné získat, je to často pouze výměnou za finanční obnos. Proto se při bezplatném získávání dat musí často volit neoficiální postupy, při kterých se můžou uživatelé během stahování dat setkávat s verifikacemi lidského faktoru, nebo data získávat netradiční formou.

Jeden z takovýchto netradičních postupů využívá práce „Forecasting stock prices with a feature fusion LSTM-CNN model using different representations of the same data“[15]. Využívá zde predikce na základě snímků obsahujících se zaznamenaným vývojem. V této práci byly zkoumány možnosti predikce ceny akcií bez nutnosti konstrukce komplexních metod pro scraping dat. Navržený model využívá schopností a možností konvolučních neuronových sítí, které se primárně využívají na zpracování obrazu a poté zapojí LSTM model s pamětí využívaný pro predikci časových řad. Konvoluční neuronové sítě jsou používány díky své schopnosti využít vnitřní reprezentaci časových řad pro nalezení vzorů chování vývoje hodnoty. Pomocí těchto informací

pak LSTM identifikují krátkodobé i dlouhodobé závislosti v datech, na základě kterých se pak vyhodnotí výsledek predikce. Výstup z konvolučních vrstev tedy je tvořen vybranými vlastnostmi původních dat a představuje vstup do LSTM vrstev, kde se prověří krátkodobá a dlouhodobá závislost dat. Tyto zpracované závislosti tvoří vstup do vrstev neuronové sítě bez paměti, ve kterých se vyhodnotí výsledek.

V rámci tohoto experimentu byly provedeny analýzy optimálního obrázku burzovního grafu pro predikci cen akcií. Za optimální a nejúčinnější typ grafické reprezentace vývoje ceny byl prohlášen svíčkový graf, jehož vývoj v čase je tvořen barevnými bloky s nenulovou výškou a šířkou. Výsledkem tohoto experimentu byl predikční model, který dosahoval vysoké úspěšnosti při predikci s měřenou průměrnou absolutní percentální chybou pod 20%. Celkově dosahoval vyšší přesnosti, než tradiční predikční techniky například s použitím ARIMA modelu.

1.4.5 Predikce řízená událostmi

Práce „Event-Driven LSTM For Forex Price Prediction“[16] se zabývala predikcí časových řad s hodnotou reprezentující jednotlivé cizí měny. Namísto posloupnosti jednotlivých datových bodů v čase byly pro trénování modelu využity posloupnosti lokálních extrémů. Pro zlepšení predikce a omezení rušivých elementů v datech byl použit přístup implementace s využitím teorie Elliottovy vlny.

Teorie Elliottových vln se používá v technické analýze při řešení pohybu ceny aktiv na finančním trhu. Jedná se o formu technické analýzy, která hledá opakující se dlouhodobé chování ceny související s přetrvávajícími změnami v sentimentu a psychologii investorů. Tato teorie identifikuje impulsní vlny, které vytvářejí vzor a korekční vlny, které se následně šíří v opačném směru od korekčních. Každá sada vln je vnořena do větší sady vln, které se drží stejného impulsu nebo koreknčního vzoru. To znamená, že lze najít tyto vlny v mnoha intervalech a měřítkách datasetů.[17]

Predikovaná data byla specifikována jako konečná hodnota po první korekční vlně. Pro predikci této hodnoty byl využit LSTM model. Pro predikci počátku Elliottovy vlny byl využit ZigZag indikátor, což je uváděno jako jedna ze slabých stránek přístupu. Predikce byly prováděny pro porovnání celkem na 4 modelech. Ze získaných vyplývalo, že nejefektivnější predikční model je obousměrné LSTM.

ZigZag indikátor slouží jako ukazatel obrácení trendu. Prostřednictvím určení oblastí růstu a poklesu vyznačuje výraznější změny aniž by byl ovlivněn krátkodobými výkyvy. Uživatelům tento přístup většinou umožňuje nastavit procentuální hranici. Poté, co se nový trend odchýlí od původního minimálně o tuto hodnotu, zaznamená indikátor informaci o změně trendu.[18]

1.4.6 Změny směru trendu

V práci „Intelligent Dynamic Backlash Agent: A Trading Strategy Based on the Directional Change Framework“ je opět využíván trend pro predikci. Autoři se snažili vyvinout metodu predikce na základě změny jeho směru. Směrové změny jsou způsobem, kterým je možné popsat pohyb tržních cen. Pomocí tohoto přístupu se rozdělí vývoj kryptoměn podle změn tržní ceny na stoupající a klesající trendy. Uživatel si zvolí velikost změny směru, která udává vznik nového trendu. Trend tedy končí, pokud se cena změní o zadanou prahovou hodnotu.

Pro samotné obchodování byl vytvořen algoritmus na základě technologie Dynamic Backlash Agent. Tento přístup se nechá aplikovat pouze pokud se celkový trend hodnoty šíří směrem dolu. Je tvořen dvěma pravidly. Signál nákupu se vygeneruje ve chvíli, kdy trend k současnému bodu je méně strmý, než celkový současný trend. V opačném případě nastane signál k prodeji.

1.4.7 Využití optimalizačních metod

Práce „Electricity price prediction based on hybrid model of adam optimized LSTM neural network and wavelet transform“[19] využívá LSTM modely na predikci ceny elektřiny. Při jednotlivých predikcích tímto modelem porovnává využití optimalizačních metod Adam, SGD a RMSProp. Z výsledků jednotlivých měření vychází optimalizátor Adam jako nejlepší varianta při predikci ceny elektřiny. Tímto vychází najevo, že při volbě modelu pro predikci také záleží na volbě vhodných dedikovaných metod.

Kapitola 2

Výběr vhodného prostředí a nástrojů pro vývoj aplikace

Před počátkem vývoje samotné aplikace čelí vývojáři kritickému rozhodnutí. Tím není nic jiného, než výběr běhového prostředí, ve kterém bude výsledná aplikace existovat a nástrojů, které pro vývoj použije. Zmíněné volby jsou předmětem řešení této kapitoly. Nejedná se o problém s jednoznačným řešením. Na základě parametrů a specifikací aplikace se však nechá výběr minimalizovat na hrstku vhodných kandidátů. První otázkou, kterou se při vývoji musíme zabývat, je zda vůbec potřebujeme uměle vytvořené prostředí. Musíme zvážit, zda naše potřeby a požadavky odůvodňují konstrukci nadstavby nebo využití lokálního prostředí na předem specifikované platformě s předem známým operačním systémem a hardwarem. Přístup s lokálním vývojem může být validní, pokud je předem rozhodnuto, že aplikace bude existovat pouze na jednom specifikovaném zařízení.

Je také třeba vyřešit, zda aplikace poběží pouze jako jeden celek, nebo bude mít rozdělené komponenty, z nichž každá bude pracovat nezávisle na ostatních jako samostatný prvek. Toto rozhodnutí významně ovlivňuje způsob ukládání dat. V případě, že je naše aplikace rozdělena do více samostatných částí, je nevhodné nebo přinejmenším velice složité využít ukládání v podobě souborů. Jedná se totiž o přístup, který není ošetřený proti více současným požadavkům na zápis najednou a může dojít k nedefinovanému chování daných souborů a ke ztrátě dat.

Dalším bodem při plánování konstrukce aplikace je volba programovacích jazyků pro backend a frontend. Na toto rozhodnutí se váže nejvyšší množství parametrů aplikace. Programovací jazyk pro backend je volen tak, aby odpovídal požadavkům na rychlost aplikace. Je také důležité aby veškeré další nástroje, které je nutné v projektu využít byly v tomto jazyce podporované. V neposlední řadě je důležitá znalost zvoleného programovacího jazyka programátorem. Omezené znalosti se můžou odrazit ve funkcionalitě a chování aplikace v krajních případech, nebo také v neefektivních řešeních, které jí zpomalují. Dále je třeba zvolit dílčí nástroje specifické pro zadané funkce aplikace.

2.1 Docker

Pro každou komerční aplikaci je výhodné, pokud jí lze spustit na co nejvíce zařízeních a operačních systémech. Zlepší to například její možnosti na získání většího počtu uživatelů a její šance na získání vyšších tržeb z prodeje. Tento krok je však v mnoha případech velice obtížný, protože vývojáři musí dbát nejen na jiné způsoby implementace pro běh na daném softwaru, ale i rozdílné hardwarové nároky.

Docker je nástroj navržený tak, aby vývojářům usnadnil vývoj, vydávání a spouštění aplikací

pomocí kontejnerů. Ty umožňují vývojářům specifikovat nároky a požadavky na kontejner, ve kterém bude běžet aplikace nebo její část. Při prvním spuštění aplikace nebo přesněji kontejneru se stáhne jeho image ze spravovaného repozitáře a na základě specifikace se nastaví tak, aby v něm mohl program být spuštěn nezávisle na platformě hostující docker. Docker se tedy postará jak o start prostředí, tak o instalaci závislostí pro aplikaci, která v tomto prostředí má běžet. Takto je možné až na výjimky zprovoznit jeden nebo více kontejnerů s aplikací na všech systémech docker podporujících nezávisle na operačním systému a jediným omezujícím faktorem jsou hardwarové nároky aplikace a hardwarové specifikace zařízení, na kterém docker běží.

Další vlastnost, kterou docker nabízí je rozdělení aplikace na více komponent a spouštění každé z nich v jiném kontejneru s možností neomezené komunikace mezi jednotlivými kontejnery a tedy i částmi aplikace. Rozdělení programu na více oddělených částí je výhodné zejména ve chvíli, kdy chceme nasazovat nové verze jednotlivých komponent bez nutnosti přerušit běh celé aplikace. Takto dosáhneme daleko vyšší účinnosti u programů a aplikací, které zpracovávají a vyhodnocují a zobrazují informace v reálném čase. Je tedy uzpůsoben pro vývoj s průběžnou integrací. Tohoto můžeme využít především pokud chceme aktivně vylepšovat systémy pro predikci, stahování dat a jejich zobrazování v naší aplikaci.

Naší aplikaci chceme vybudovat tak, aby byla co nejvíce odolná vůči chybějícím datům, zastavení odpovídání serverů, nebo třeba i výpadkům připojení či elektřiny. Docker umožňuje nastavení automatického restartování kontejneru po opětovném zapnutí zařízení, na kterém je aplikace provozována.[20, 21]

Vzhledem k nízkým nárokům na výkon a relativně nízké velikosti jednotlivých kontejnerů je pak Docker výhodný i pro systémy s omezenými hardwarovými specifikacemi.

2.2 PostgreSQL

Při výběru databázového systému se nám nabízí vysoká škála možností. Na výběr z typů databází máme například distribuované databáze, relační databáze nebo NoSQL databáze.[22]

Distribuované mají data rozloženy mezi různými databázovými systémy. Tyto databázové systémy jsou propojeny komunikačními linkami. Jejich příklady jsou Apache Cassandra nebo HBase. Jejich výhodou je modulární vývoj. Systém počítá s možností rozšíření zapojením více serverů.

Relační databáze jsou založeny na datovém modelu, který ukládá data ve formě řádků a sloupců a společně tvoří tabulku. Relační databáze používá jazyk SQL k administraci uložených dat. Každá tabulka v databázi nese unikátní klíč, kterým se jednotlivé údaje identifikují. Příklady takovýchto databází jsou MySQL a PostgreSQL.

Každá verze relačních databází sdílí vlastnosti známé jako ACID[23], na kterých jsou také založené. Jedná se o atomicitu, konzistenci, izolace a trvanlivost.

Atomicitou je zajištěno, že po dokončení operace budou změny v datech prospěšně uloženy, nebo změna neproběhne vůbec. Nemůže se tedy stát, že by se při přerušení operace poškodily data.

Konzistence udává, že pokud provedeme jakoukoli operaci s daty, jejich hodnoty před a po operaci zůstanou zachovány. To znamená, že žádnou operací nevzniknou bezúčelně nová data, se kterými uživatel nepočítal.

Izolace je využita při současném přístupu více uživatelů do databáze najednou. V této situaci by operace neměla ovlivnit data ostatních uživatelů přistupujících k databázi a projeví se až po ukončení každé transakce. Do té doby změny nejsou pro ostatní uživatele viditelné.

Trvanlivost značí, že po ukončení operací a potvrzení dat tyto změny zůstanou trvale v databázi uloženy.

NoSQL je typ databáze, která se používá pro ukládání široké škály datových sad. Na rozdíl od relačních databází nemá omezenou svojí strukturu na tabulky, sloupce a řádky, ale má schopnost ukládat data mnohdy různými způsoby. NoSQL databáze tedy můžeme dále dělit podle typů

ukládání dat na key-value databáze, dokumentově orientované databáze, grafové databáze a uložiště s širokými sloupci.

Key-value uložiště jsou nejjednodušším typem databázového uložiště, kde se ukládá každá jednotlivá položka pomocí klíčů, které udržují hodnoty dat.

Dokumentově orientovaná databáze se používá k ukládání dat jako dokumentů podobným formátu JSON. Tímto je dosaženo ukládání ve stejném formátu, který může být použit v kódu aplikace.

Grafová databáze se používá pro ukládání nadměrného množství dat do struktury podobné grafu. Tyto databáze jsou využívány například v souvislosti se sociálními sítěmi.

Uložiště se širokými sloupci se podobá struktuře v relačních databázích. Data jsou zde ale uložena ve sloupcích, namísto reprezentace pomocí řádků.

V naší aplikaci budeme využívat periodické ukládání a stahování dat. Předpokládá se, že data na sebe nebudou plynule navazovat, ale budou se často překrývat. Můžeme předpokládat, že vzhledem k neustálému stahování dat od více různých poskytovatelů nemůžeme určit, zda stažená data budou ve své finální verzi a zda se ještě v průběhu své životnosti u poskytovatele nezmění. Bude tedy potřeba vybrat takový databázový systém, který je schopný při každém vkladu nahradit současná data vkládanými, pokud mají stejnou časovou známku. Zároveň budeme využívat strukturovaná data v podobě tabulky, která budeme ještě před ukládáním upravovat pro získání jednotné podoby. Dává tedy smysl použít typ relační databáze. Konkrétně zvolíme PostgreSQL, která splňuje veškeré požadavky. PostgreSQL je objektově relační databáze na rozdíl od MySQL, která je čistě relační. Toto znamená mimo jiné, že PostgreSQL nabízí využití komplexních datových typů. Dále má ale lepší podporu pro aplikace s vysokým počtem čtení a zápisů na rozdíl od MySQL, která je orientovaná spíše na vylepšení rychlosti při čtení.

V práci si hodláme také manuálně zálohovat data do souborů pro jejich případnou obnovu během vývoje a pro případné snížení doby čekání na stažení dostatečného množství dat pro predikce. Z tohoto důvodu preferujeme PostgreSQL před ostatními variantami relačních databází.[24]

2.3 Python

Python je vysokoúrovňový programovací jazyk. Je navržen způsobem, který klade důraz na čitelnost kódu. Aktivně ve své syntaxi využívá odsazení řádků. Využívá garbage-collector a nevyžaduje tedy po programátorech manuální uvolňování paměti. Plně podporuje objektově orientované programování stejně tak jako funkcionální programování[25]. Namísto implementace veškerých svých funkcionalit do svého jádra byl Python od začátku konstruován s možností rozšíření prostřednictvím mnoha modulů. Díky tomuto přístupu se mu dostalo značné popularity[26]. Vzhledem k nesčetnému množství existujících knihoven a modulů jsou jeho možnosti prakticky neomezené a může nalézt využití ve vývoji grafických aplikací, data science, umělé inteligenci, animaci, zpracování obrazu a mnoho dalších. Pro řešení problémů strojového učení je Python tedy ideální programovací jazyk. Disponuje totiž jak knihovnami pro samotné strojové učení jako například PyTorch, Tensorflow nebo Scikit-learn, tak ale i knihovnou pro rychlou a efektivní práci s daty v tabulkové podobě jako je například Pandas nebo knihovnu Seaborn a Matplotlib pro vizualizaci dat. Jednoduchost tohoto programovacího jazyka navíc umožňuje vývojářům psát spolehlivější programy. Vývojáři se pak mohou věnovat lépe řešení samotného problému strojového učení a nejsou omezováni obtížností konstrukce nástrojů v daném jazyce.[27]

2.4 GUI

Vue.js je frontend framework typu MVVM pro vytváření uživatelských rozhraní fungující jako nadstavba nad standardy HTML, CSS a JavaScript. Mezi největší výhody Vue.js patří reaktivita, tedy schopnost automatického sledování změn stavu uložených dat a jejich následná

prostřednictvím uživatelem specifikovaných metod. Dále také disponuje deklarativním vykreslováním, což znamená, že rozšiřuje HTML o syntaxi šablony, která umožňuje uživateli deklarativně popsat výstup HTML. V rámci HTML kódu můžeme tedy využít dvojitých složitých závorek či jiných uživatelem specifikovaných identifikátorů, na základě kterých framework dynamicky aktualizuje obsah uvnitř těchto identifikátorů. Framework také nabízí obousměrné vázání a cykly, díky kterým jsme schopni propojit hodnoty prvků HTML s daty aplikace. Pro vývoji GUI aplikace tedy využijeme právě reaktivní vlastnost tohoto frameworku. To nám ulehčí zobrazování dat uživateli bez konstrukce složitých funkcí pro obnovu stavu grafických prvků. Vue dále nabízí možnost zasílání HTTP GET požadavků. Takto budeme schopni našemu frontendu zasílat na vyžádání nová data z backendu aplikace.[28, 29, 30]

2.5 Scraping

V aplikaci bude nutné vhodně implementovat stahování dat v reálném čase z internetu. Pro tento účel je možné využít například knihovnu requests. Ta poskytuje jednotlivé nástroje k odesílání HTTP požadavků a získávání odpovědí na ně. Dále implementuje metody, které přidají možnost zadání parametrů do HTTP požadavku na základě uživatelem specifikovaných klíčových slov. Odstraňuje tedy potřebu složitě specifikovat možnosti přímo do řetězce dotazu.[31]

Ač je tento přístup vcelku jednoduchý, má svá úskalí. Některé internetové služby ovšem nejsou ochotné udělit přístup ke svým datům touto metodou. Namísto toho jsou tato data využívána dynamicky. Jsou tedy předmětem konstantní změny v rámci nějakého předdefinovaného kontejneru a pro jejich získání musí být použity nástroje, které jsou schopny se těmto omezením přizpůsobit.

Tímto nástrojem je například Selenium. Jedná se o řadu nástrojů a knihoven, které jsou uzpůsobeny k automatizaci prohlížečů. Jsou schopny simulovat interakci uživatele s prohlížečem přes klasické grafické rozhraní. To zahrnuje mimo jiné klikání interaktivních tlačítek nebo vyplňování formulářů, ale také načítání prvků do paměti a zprostředkování jejich přístupu uživateli přes nějaký unikátní identifikátor. Pro psaní instrukcí Selenium využívá WebDriver. Ten prostřednictvím klientského rozhraní přijímá příkazy a odesílá je do zvoleného prohlížeče. Samotný prohlížeč pak musí mít implementovanou sadu nástrojů, která tento způsob komunikace podporuje. Pro naše účely budeme používat chromium. Bude tomu tak vzhledem k faktu, že se od něj odvíjí množství prohlížečů, jako například Google Chrome nebo Microsoft Edge. To má mimo jiné za následek jistou garanci stability a široké podpory také vzhledem k tomu, že na jeho vývoj dohlíží Google.[32]

Tento přístup je značně náročnější na výkon. Je tomu tak především z důvodu, že pro svojí funkci potřebuje Selenium plnohodnotný prohlížeč, prostřednictvím kterého přistupuje k internetu. Zvýšené nároky na výkon jsou ale daň za získání přístupu k datům, které nelze načíst pouhým zasláním HTTP požadavku.[33]

2.6 Server

Frontend naší aplikace bude vyžadovat konstantní přístup k datům, o které si bude periodicky žádat. Bude se jednat o historii vývoje ceny kryptoměn a dále podrobné údaje o současném stavu kryptoměn zprostředkováno predikcemi natrénovaného modelu. Tuto komunikaci můžeme řešit více způsoby. Prostředky pro komunikaci mezi serverem a klientem poskytuje například WebSocket API nebo Flask.

Websocket API je technologie umožňující obousměrnou interaktivní komunikaci mezi klientem a serverem. Tímto způsobem jsme schopni odesílat zprávy na server a přijímat odpovědi řízené událostmi, při kterých není potřeba dotazovat se serveru na odpověď.[34]

Další možností je využití Flask framework. Je označován za microframework, protože nevyžaduje pro svojí funkci žádné další nástroje nebo knihovny[35]. Framework závisí však na šablono-

vém enginu Jinja, díky kterému je možné generovat obsah v HTML šabloně obsah pomocí kódu podobajícímu se programovacímu jazyku Python, a sadě nástrojů Werkzeug WSGI, které zprostředkovávají komunikaci mezi webovou aplikací a serverem [36, 37].

Mimo samotné komunikace se serverem budeme v grafické části naší aplikace využívat načítání a zobrazování samotného rozložení v HTML souboru, sady stylů CSS a současně také kód v jazyce JavaScrypt, nebo přesněji ve frameworku Vue.js, který zprostředkuje dynamické chování aplikace. Tímto chováním je myšleno zobrazování vývoje hodnoty kryptoměn v grafu a zobrazování aktuálního stavu kryptoměn a budoucího stavu zprostředkovaného predikcemi. Flask disponuje možností sestavit takto grafické rozhraní z dílčích souborů. Je tedy v rámci našich potřeb ideální volbou.[38]

2.7 Umělá inteligence

Pří výběru knihoven, které nabízejí software pro strojové učení, umělou inteligenci a v první řadě konstrukci predikčních modelů s paměťovými bloky, jako jsou LSTM a GRU, máme na výběr primárně ze dvou alternativ. Těmi jsou TensorFlow a PyTorch. Jedná se o velice podobné nástroje. Oba konstruují model ve formě orientovaného acyklického grafu a liší se pouze jejich definování uživatelem.

TensorFlow je open-source knihovna vyvinutá Google Brain Team. Je velice dobře přizpůsobena pro využití jak nováčky tak profesionály. Predikční modely se zde vytváří staticky před jejich během v rámci definované session. Má velice rozsáhlou dokumentaci zahrnující mnoho ukázek a návodů, které umožní rychlé pochopení principů a zvýší efektivitu při vývoji. Mimo oficiální zdroje existuje pro TensorFlow mnoho tutoriálů, školení a volně dostupného kódu, což z něj pro většinu vývojářů činí první volbu pro strojové učení.[39]

PyTorch je knihovna disponující velice podobnými schopnostmi a přístupy, jako TensorFlow. Oproti ní je však novější a umožňuje dynamický přístup ke konstrukci k modelu. Je možné za jeho běhu je možné definovat, měnit a spouštět jednotlivé uzly v modelu. Jeho adaptace v širších kruzích nastala poněkud později a podle průzkumů stoupá jeho popularita mezi vývojáři. Přesto však se většina příspěvků na softwarově zaměřených sociálních sítí jako například Stack Overvlow stále týká TensorFlow.[40]

V našem modelu nebudeme potřebovat dynamické vlastnosti modelů. Počet historických datových bodů bude konstantní. Při predikci sentimentu by toto mohl být problém vzhledem k měnícímu se počtu slov ve větách. Tento problém s naším řešením ale nesouvisí. Při použití TensorFlow tedy budeme mít tedy přístup k více informacím o tomto nástroji, ale také k tfdbg, což je nástroj přizpůsobený pro debugging modelů. TensorFlow je tedy pro naše účely tou lepší volbou.[41]

Kapitola 3
Scraping

Proto, abychom mohli natrénovat model, který budeme moci používat pro predikce v naší aplikaci, budeme potřebovat data, na kterých jej chceme trénovat. Tato kapitola se zabývá různými technikami získávání dat z internetu a různými datasety, které jsme zvažovali v naší práci použít. Jedná se nejen o stejná data, která byla použita v analyzovaných projektech v teoretické části, ale také o námi zvolená, která jsme testovali pro jejich informační hodnotu při predikcích.

3.1 Získávání dat kryptoměn

3.1.1 Průzkum

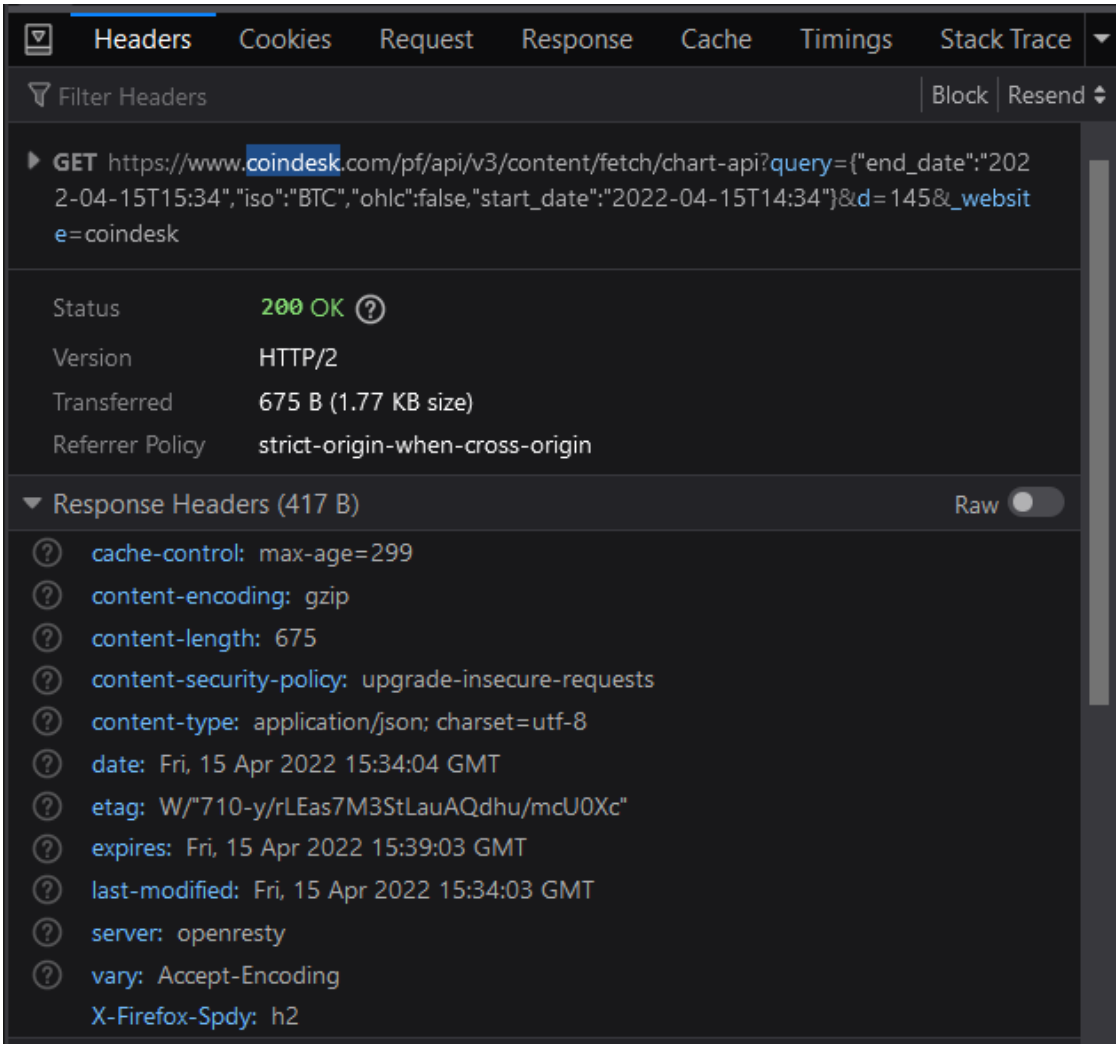
Získání historických cen kryptoměn v minutových intervalech není problém alespoň pro nejznámější kryptoměny. Server `https://www.cryptodatadownload.com` nabízí jejich bezplatné stažení ve formátu .csv a v časovém intervalu přesahujícím jeden rok. Mnohem složitější je získávání real-time minutových dat. Od většiny poskytovatelů lze získat tyto informace zdarma pouze omezeně ať už v rámci volání za pevný časový interval, nebo v rámci celkového povoleného počtu volání, která má například CryptoCompare API[42]. Mimo to pak vyžadují registraci uživatele a přístup přes předem definované údaje. Je tedy potřeba prozkoumat přístupy k získávání těchto dat v reálném čase, které nejsou zprostředkovány žádným oficiálním API.

Mnoho internetových stránek zabývajících se kryptoměnami nabízí také vizualizace a přehledy mnoha kryptoměn. Konkrétně se jedná o grafy, které na základě dat obdržených ze serveru vykreslují zadaným způsobem nějaké vlastnosti kryptoměn. Toto znamená, že před tím, než se vizualizace zobrazí uživateli, je vygenerována na klientské straně z číselné reprezentace předané prostřednictvím komunikačního kanálu. Existují stále výjimky v podobě vykreslování statických obrázků namísto interaktivních grafů. V poslední době je však většina vizualizací prováděna interaktivně prostřednictvím programovacího jazyka JavaScript.

Jeden z poskytovatelů dat o kryptoměnách je CoinDesk. Na svých stránkách poskytuje přístup k mnoha kryptoměnám a zároveň zobrazuje interaktivní grafy o vývoji jejich ceny s více různými časovými měřítky[43]. Pro otestování, zda jsme schopni získat data využívaná ke generování zobrazených grafů jsme postupovali podle návodu „Scrape ANY Chart, Graph, Visualization or Interactive with JSON Data Sources - Free Web Extractor“[44] na platformě YouTube.

Prohlížeče jako například Google Chrome nebo Microsoft Edge poskytují vývojářské nástroje pro usnadnění vývoje webových aplikací. Mimo jiné je zde možné zobrazit i síťovou komunikaci mezi otevřenou internetovou stránkou a ostatními zdroji. Naším úkolem bylo identifikovat data, která jsou této stránce zasílána k vykreslení grafu. Jednalo se o nalezení komunikace na obrázku 3.1 Předpokladem bylo, že data nejsou zasílána periodicky, ale pouze na základě

Obrázek 3.1 Komunikace obsahující hledaný dataset



požadavku z front-endu. Tedy je potřeba pozorovat komunikaci po kliknutí na časová měřítka v grafu. Pro naše účely jsme potřebovali data s minutovou přesností, tedy klikneme na nejnižší časový údaj nabízený v grafu. V záznamu komunikace sítě můžeme zpozorovat komunikaci s adresou `https://www.coindesk.com/pf/api/v3/content/fetch/chart-api?query=%7B%22end_date%22%3A%222022-04-02T08%3A23%22%2C%22iso%22%3A%22BTC%22%2C%22ohlc%22%3Afalse%2C%22start_date%22%3A%222022-04-02T07%3A23%22%7D&d=142&_website=coindesk`. V adrese vidíme, že se v ní vyskytuje časové razítko začátku a konce intervalu požadavku a zkratka kryptoměny, jejíž graf jsme zkoumali. Po zobrazení tohoto obsahu je možné vidět dataset ve formátu JSON přehled vývoje kryptoměny společně s časovými razítky příslušných hodnot. Data mají stejnou strukturu, jako v kódu 3.1.

Výpis kódu 3.1 Formát datasetu s historickými hodnotami kryptoměn

```
{
    "iso":"BTC",
    "name":"Bitcoin",
    "slug":"bitcoin",
```

```
    "ingestionStart":"2014-11-03",
    "interval":"1-minute",
    "src":"tb",
    "entries":[
        [164239000,46539.628207],
        ...
        [164888770,46585.082078]
    ],
    "_id":"245...386"
}
```

3.1.2 Návrh

Při získávání dat máme možnost vyzkoušet získávání dat pomocí requests API. Jedná se o jednoduchý způsob spočívající v zaslání HTTP GET požadavku na adresu s daty. Do adresy požadavku jsme přidali ISO zkratku kryptoměny a správně formátovaný časový údaj. Požadavek jsme následně odeslali. Získanou odpověď jsme pak převedli z textového řetězce do formátu JSON a vyextrahovali jsme z něj dvojice časových údajů a hodnot kryptoměny. Časové údaje jsme pak převedli z textového formátu na datetime zaokrouhlený na minuty. Pomocí nástroje Pandas jsme tyto údaje převedli do tabulkové podoby a uložili v naší PostgreSQL databázi.

3.1.3 Výsledky

Tento postup se osvědčil jako funkční. Bylo možné bez omezení stahovat data o uživatelem specifikovaných kryptoměnách. Finální verze aplikace každých 30 vteřin zašle požadavek na server a získaná data uloží do databáze tak, že novými daty nahradí původní, pokud se překrývají. Stahování ceny kryptoměn běží v nekonečném cyklu a ve vlastním docker kontejneru. Při běhu se výjimečně vyskytly problémy s odpovědí serveru. Ty jsou validovány pomocí podmínky kontrolující návrat úspěšné odpovědi s kódem 200 a při neúspěchu jsou ošetřeny pomocí *try-catch* výrazu, který po zaznamenané chybě obnoví běh po 30 vteřinách.

3.2 Twitter

3.2.1 Průzkum

Při průzkumu současných technologií bylo zjištěno, že mnoho řešení obsahuje příspěvky ze sociální sítě Twitter. Zde jsme měli možnost využít oficiální API, které je nabízeno pro stahování a analýzu tweetů. Toto API však má mnoho omezení, z nichž největší je možnost stahování tweetů s maximálním stáří jeden týden[45]. Z tohoto důvodu jsme museli použít vlastní nástroje pro stahování tweetů, pokud jsme tento přístup chtěli používat v naší aplikaci.

Vzhledem k tomu, že příspěvky na Twitteru jsou přidávány a mazány dynamicky na základě scrollování uživatele, museli jsme pro jejich získání použít Selenium. Takto jsme využili prohlížeč Chromium, ve kterém jsme algoritmicky scrollovali a následně stahovali prvky, které se v ten daný moment nacházely v okně. Jako základ našeho nástroje pro scraping jsme použili neoficiální API na stahování tweetů Scweet[46]. Toto API ovšem vůbec nevyhovovalo našim účelům. Nebylo v něm možné zadat časový interval přesnější, než jeden den a stažení tweetů trvalo příliš dlouho v důsledku aplikování mnoha flitrů na jednotlivé tweety. Mělo ale správně implementovaný přístup na Twitter Wall a vyhledávání podle hashtagu. Této funkcionality jsme tedy využili.

3.2.2 Návrh

Ze studie „Event-Driven LSTM For Forex Price Prediction“ [16] vyplývá, že sentiment tweetů je oproti jejich počtu mnohem méně informativní. Ukládali jsme tedy jen počet nově publikovaných tweetů za každou minutu. Pro zvýšení rychlosti jsme z tweetů získávali pouze čas zveřejnění a nijak jsme je nefiltrovali. To znamená, že jsme započítávali i tweety s reklamou, což by nemělo vést ke zhoršení kvality dat z důvodu předpokladu konstantního počtu reklam na zadané časové okno pro každý hashtag. V kódu jsme následně implementovali nový systém hledání tweetů pomocí identifikátoru *XPATH* specifikovaný cestou *'//article[@data-testid="tweet"]'* a také jsme vytvořili nový scrollovací systém, který po každém stažení tweetu posouval Twitter wall o velikost zobrazeného okna. Ve chvíli, kdy se našemu programu po mnoha pokusech nepodařilo scrollovat, byla současná pozice považována za konec stránky a stahování bylo ukončeno.

Webová aplikace Twitter nepovoluje specifikování času hledaných tweetů v časovém měřítku přesnějším, než je jeden den. To znamená, že při stahování historických tweetů byl nejnovější příspěvek poslední, který byl zveřejněný před půlnocí a pomocí Selenia a Chromia se scrollovalo dále do minulosti. Při vyhodnocování tweetů v reálném čase jsme však byli schopni blíže specifikovat jak daleko do minulosti má scraper scrollovat.

Pro zamezení nebo alespoň snížení pravděpodobnosti odhalení nelidské aktivity ochrannými mechanizmy se po každém scrollování čekalo po dobu náhodně zvoleného časového intervalu mezi *0.5* a *1.5* vteřinami.

3.2.3 Výsledky

Stahování tweetů se ukázalo jako velice nevýhodné. Využívání prohlížeče pro opakující se stahování tweetů bylo jednou z výkonnostně nejtěžších úloh v aplikaci. Ačkoliv jsme byli schopní vytvořit rychlý systém pro stahování na základě hashtagů, zvyšovala se doba stahování tweetů až nad časový úsek, který jsme chtěli stahovat. Při stahování historických dat navíc Twitter přibližně po stažení 7 hodin odhalí naší činnost a zablokuje nám přístup k tweetům.

Tento problém bylo teoreticky možné odstranit s použití různých VPN serverů, mezi kterými by se střídalo po kratších časových intervalech. Taky jsme se mohli pokusit zpomalit stahování pomocí prodloužení čekání. Vzhledem k tomu, že stahování bylo ale už za současné situace pomalé a načítání tweetů vyžadovalo vysoké využití paměti i výkonu, nebylo vhodné stahování a zpracování tweetů zahrnout do naší aplikace.

3.3 Google

3.3.1 Průzkum

Google Search Engine je nejvyužívanější vyhledávací engine na světě s podílem využití přesahujícím 90%[47]. Vzhledem k jeho návštěvnosti by byla jeho data o vyhledávání ideálním ukazatelem přispívajícím k predikci vývoje kryptoměn. Tato data jsou dostupná uživatelům prostřednictvím stránky Google Trends. Zde je poskytována analýza popularity různých dotazů vyhledávání v Google Search Engine v různých regionech a jazycích. Tato popularita nabývá hodnoty od 0 do 100. Každý datový bod je vydělen celkovým počtem vyhledávání v zadaném časovém rozsahu a reprezentuje relativní popularitu vyhledávání termínů.

Google ovšem pro získávání těchto dat nenabízí žádné oficiální API. Proto jsme využili stejného přístupu, jaký využívá projekt Pytrends[48] pro získání dat o vyhledávání. Pro samotné dotazování na data jsme využili zasílání HTTP požadavku pomocí requests API na předem specifikovanou adresu obohacenou o sérii doprovodných parametrů.

3.3.2 Návrh

Odesílání dotazů na Google Trends z naší aplikace probíhá v nekonečném cyklu a každých 30 vteřin je zaslán nový požadavek na stažení dat. Termíny, na které se doptáváme jsou ISO zkratky jmen kryptoměn. Při spuštění se napřed zjistí, jaké bylo poslední uložené datum u jednotlivých stažených kryptoměn a pak je od tohoto časového údaje obnoveno stahování dat o vyhledávání. Pokud se jednalo o první spuštění pro zadanou kryptoměnu, stáhly se napřed data z předešlé hodiny. Nastavení stahování jedné hodiny lze však modifikovat na libovolnou časovou vzdálenost počátku stahování. Architektura aplikace nám tedy umožňuje stahovat data v libovolném časovém intervalu s přesností na minuty. Počátek časového intervalu však nemůže být starší, než půl roku. Touto dobou pravděpodobně došlo k modifikaci datového formátu a nedostupnosti starších dat touto metodou.

Na počátku stahování se nastaví parametry kategorie, lokality, časové zóny, hledané slovo, jazyk a časové okno. Ty jsou postupně počínaje kategorií *finance*, *”*, *UTF-0*, *jméno kryptoměny*, *en-US*. Časové okno je specifikováno jako čas počátku a konce intervalu v požadovaném formátu. Velikost každého intervalu bude maximálně 270 minut. Jedná se o největší časové okno, ve kterém Google Trends poskytuje data za každou minutu vyhledávání. Takto získaná data se převedou do tabulkového formátu o dvou sloupcích, kde první sloupec a zároveň index je datum a druhý sloupec je popularita v dané minutě. V případě chyby při provádění *GET* požadavku pomocí requests API, bude aplikace čekat půl minuty, než se požadavek zopakuje.

3.3.3 Výsledky

Tento postup se prokázal jako zcela funkční a je implementován v konečné verzi aplikace. Při nastavení kratší doby čekání na odeslání požadavku, než je 30 vteřin je nám po určité době odepřen přístup ke Google Trends a vrácena je pouze chyba typu *429* značící příliš mnoho uskutečněných požadavků na server. Rychlost Jednoho požadavku za půl minuty je však dostačující a není zde potřeba tento problém řešit. Pokud by ovšem do budoucnosti bylo potřeba zvýšit frekvence zasílání požadavků, je možné tento problém vyřešit pouhým přepínáním serverů poskytovatele VPN.

3.4 Zlato, VIX, S&P 500 a další indexy

3.4.1 Průzkum

Při průzkumu současných řešení predikce hodnoty kryptoměn bylo zjištěno, že některá z nich používají například vývoj ceny zlata, výkonnost 500 největších firem a společností uvedených na burzách ve spojených státech S&P 500 index, Ukazatele nestálosti a změny na S&P 500 na trhu CBOE Volatility Index, hodnotu amerického dolaru nebo hodnoty 30 dlouhodobě výnosných společností DJIA. V naší aplikaci jsme využili 3 z těchto indikátorů na základě účinku na přesnost predikce.[49, 50, 51]

Při konstrukci scraperu pro uvedené hodnoty jsme postupovali podobně, jako u získávání hodnot kryptoměn. Grafy těchto indikátorů jsme nalezli na stránkách `https://www.investing.com/`. Odtud jsme získali data pomocí sledování internetové komunikace při změně náhledu grafu stejně, jako v případě dat o kryptoměnách.

3.4.2 Návrh

Hledaný odkaz na síťovou komunikaci má formát `https://tvc4.investing.com/68865803aac4b4410f7c8caf7483921c/1648917052/1/1/8/history?symbol=166&resolution=1&from=1648485061&to=1648917121`. Prvním údajem je náhodně vygenerovaný hash, který nemá efekt

Výpis kódu 3.2 Selenium ovladač

```
    options = Options()
    options.add_argument('--no-sandbox')
    options.add_argument('--headless')
    options.add_argument('--disable-gpu')
    options.add_argument('--disable-dev-shm-usage')
    options.add_argument('--profile-directory=Default')

    driver = webdriver.Chrome(options=options)
    driver.set_page_load_timeout(self.page_load_timeout)
    return driver
```

na obsah stránky s odkazem. Dalším údajem je časová známka, která značí přesný čas, kdy je k datům přistupováno. Prvním důležitým identifikátorem v odkazu je až symbol, kterým je značen ukazatel, který chceme stáhnout. Číslo 166 je identifikátor S&P 500. Data přístupná přes odkaz pak mají JSON formát s odděleným polem s časem následovaný 4 poli s nejvyššími, nejnižšími počátečními a koncovými hodnotami v rámci minut.

3.4.3 Výsledky

V tomto případě již nebylo možné využít requests API. Data byla generovaná nestandardním blíže neznámým způsobem, což znamená, že bylo nutné využít Selenium v kombinaci s prohlížečem Chromium. Jednalo se však pouze o malou úpravu v kódu, protože Selenium s requests API sdílí podobný princip volání. Stačí tedy pouze vytvořit ovladač, jako je tomu v kódu 3.2, kterým budeme komunikovat s prohlížečem, získat text z obsahu webdriveru a ovladač zavřít. Takto jsme získali výsledek v podobě, ve které by byl vrácen z requests API. Získávání těchto dat opět běželo v cyklu po půlminutových pauzách. Stejně jako u ostatních informací jsou získaná data ukládána do databáze ve formě tabluky o dvou sloupcích. Indexem je čas a rozdílné hodnoty v rámci minut jsme sloučili do jedné použitím průměru.

Ačkoliv jsme byli pro stahování dat nuceni využít Selenium, nebyla tato operace příliš náročná. Jednalo se totiž pouze o načtení stránky bez žádné doprovázející interakce. Při testování nebyly odhaleny žádné chyby spojené s četností požadavků na server. Maximální velikost časového intervalu, který se nechá stáhnout v rámci jednoho volání je jeden týden. Při postupném rozdělení získávání historie po těchto časových úsecích však není problém stáhnout údaje i z roku 2021. Ty jsou však pouze dostupné se zkreslenou přesností a data jsou hodnoty ukazatelů v pětiminutových intervalech.

Kapitola 4
Predikce

4.1 Preprocessing

Tato kapitola řeší volbu, trénování a používání predikčních modelů pro naší aplikaci. Předtím, než vůbec můžeme uvažovat nad využitím různých postupů strojového učení, je nutné převést data do formátu, ve kterém je bude moci predikční model zpracovat a efektivně využít. Je třeba zejména vyřešit problém s chybějícími nebo nekonzistentními daty a s nalezením vhodné reprezentace datových bodů. Také je důležité se rozhodnout, jaké nástroje pro jejich zpracování použijeme.

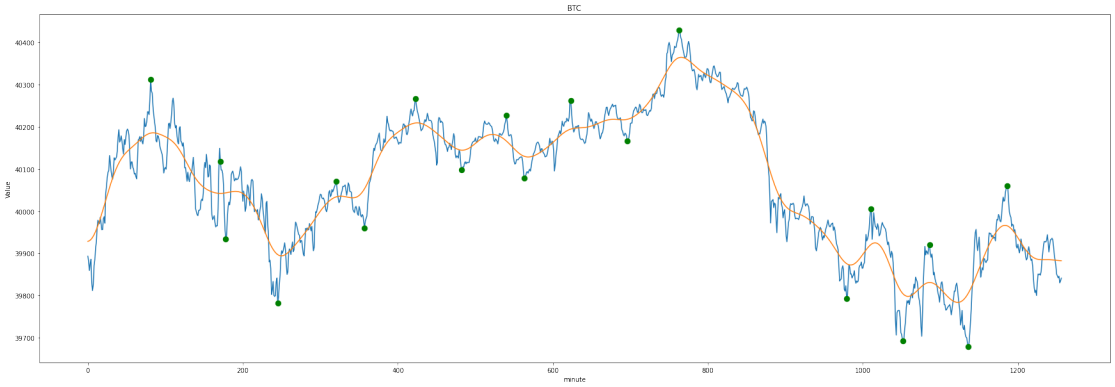
4.1.1 Doplnění hodnot

Problémy s chybějícími hodnotami se projevovalz zejména v datasetech získaných před rokem 2022 ze serverů investing.com. Zde jsme měli možnost stahovat pouze data s přesností jednoho datového bodu na pět minut. Zároveň jsme museli vyřešit problém pravidelného zavírání obchodu se zlatem a ostatními ukazateli, které jsme chtěli využít k predikci. Cena zlata je dostupná pouze ve všední dny, kdy se s touto komoditou také obchoduje. Stejné problémy se týkají také například S&P 500 a VIX indexů. Chybějící data jsme tedy doplňovali s využitím nástrojů databázového systému PostgreSQL. Zde jsme vygenerovali časové známky po minutách mezi prvním a posledním časovým údajem v datasetu. Následně jsme je sloučili s indexy datasetu. Výsledný dataset pak neměl chybějící časové údaje. Stále se v něm ovšem mohli vyskytovat chybějící nebo nulové hodnoty. Pokud se v datasetu vyskytovaly nulové hodnoty, přenastavili jsme je na chybějící data a následně jsme je doplnili. Tento proces jsme realizovali pomocí knihovny Pandas, ve které je možnost doplnění chybějících hodnot na základě existujících údajů. Všechny chybějící údaje v datech kromě hodnot kryptoměn jsme nahradili poslední známou hodnotou v datasetu. Hodnoty kryptoměn jsme paka vyplňovali lineárně, tedy váženým průměrem mezi hodnotou předcházející a následující za úsekem s chybějícími hodnotami. Váha zde byla poměr počtu minut mezi poslední předcházející a první nadcházející hodnotou. Tento postup doplnění dat jsme aplikovali pokaždé, kdy bylo nutné data z databáze použít.

4.1.2 Gaussův filtr

Gaussovy filtry mají tu vlastnost, že se u nich nevyskytuje překročení maximální nebo minimální hodnoty vstupu a zároveň minimalizují dobu stoupání a klesání. Toto chování úzce souvisí se skutečností, že Gaussův filtr má maximální možnou adaptaci trendu. Tento filtr je na základě

Obrázek 4.1 Vyhlazení grafu a nalezení extrémů



uvedených specifikací jeden z mála, který nikdy nevytvoří nové lokální extrémy, které se nevyskytovaly v původním vstupu. To z něj činí ideální filtr pro zpracování časového signálu a jeho očištění od rušivého signálu.[52]

Naše vstupní data jsou tvořena jedním sloupcem reprezentujícím vývoj hodnoty v čase. Jedná se tedy o jednodimenzionální data. Pro zpracování této posloupnosti jsme využili Gaussův filtr pro jednodimenzionální data v následující podobě:

$$g_\sigma(x) = \frac{1}{2\pi\sigma}^{(-\frac{x^2}{2\sigma^2})}$$

4.1.3 Lokální extrémy

Při průzkumu současných trendů v predikcích kryptoměn nás zaujal postup, který pro predikci volil lokální extrémy v posloupnosti Elliottových vln[16]. Tímto přístupem se autorům podařilo velice zredukovat problém predikce a dosahovali překvapivě dobrých výsledků. Proto jsme chtěli podobný přístup aplikovat na predikci kryptoměn. Pro predikci jsme nepoužívali teorii Elliottových vln. Využili jsme ale pro predikci lokální extrémy z původního datasetu.

Pro nalezení lokálních extrémů jsme využili dataset s hodnotami kryptoměn a gaussův filtr, který je implementovaný v knihovně scipy jako `scipy.ndimage.gaussian_filter1d`. Hodnotu sigma jsme nastavili na 15, což nám zaručilo nalezení lokálních extrémů ve vyhlazeném grafu v rozmezí desítek minut. Následně jsme nalezli lokální minima a maxima, která jsme umístili do dvou datasetů. Poté jsme vytvořili zvlášť dvojice sousedních lokálních minim a maxim. Mezi dvojicemi lokálních minim jsme v původním datasetu před použitím gaussova filtru hledali maximální hodnotu. Stejný postup jsme provedli s dvojicí lokálních maxim a hledali jsme minimální hodnotu. Výsledný dataset se poté skládal ze seřazené posloupnosti nalezených minimálních a maximálních hodnot. Tyto hodnoty tvořili skutečné lokální extrémy a tvořili základ výsledného datasetu. Vzhledem k tomu, že skutečné lokální extrémy se hledají mezi lokálními extrémy vyhlazených dat, vyskytuje se poslední extrém příliš daleko od konce časové řady. Tento problém jsme vyřešili umístěním posledního lokálního extrému mezi současný poslední lokální extrém a konec dat. Nalezení těchto extrémů je reprezentováno obrázkem 4.1. Dataset s předzpracovanými daty je pak tvořen hodnotami s časovým razítkem nalezených lokálních extrémů v datech s kryptoměnami.

Dále výsledný dataset tvoří časová razítka jednotlivých extrémů s přesností na minuty. Z původního datasetu jsou pak vytvářena vyhlazená data s použitím Gaussova filtru s hodnotou sigmy 15, 60 a 240. Z těch jsou vybírány body, ve kterých se vyskytují skutečné lokální extrémy z původního datasetu. Tento postup se třemi různými okny je prováděn pro data s hodnotami kryptoměn, data z Google Trends a pro data stažená z investing.com. Následně z nich stejným

Výpis kódu 4.1 ADF

```
from statsmodels.tsa.stattools import adfuller
def ADF_Cal(x):
    result = adfuller(x)
    ADF_stat = result[0]
    p = result[1]
    print("ADF␣Statistic:␣%f" % ADF_stat)
    print("p-value:␣%f" % p)
    print("Critical␣Values")
    levels = [.01, .05, .1]
    i = 0
    for key,value in result[4].items():
        print('\t%s:␣%.3f' % (key,value))
        hyp = p < levels[i]
        if ADF_stat < value:
            cert = (1-levels[i])*100
            print("{}%␣certain␣this␣is␣staionary".format(cert))
            print('Reject␣H0:␣{}'.format(hyp))
            break
        i = i+1
        if i >= 3:
            print("Less␣than␣90%␣certain␣that␣data␣is␣stationary")
            print('Reject␣H0:␣{}'.format(hyp))
print("Calculating␣ADF␣test␣for␣X...")
ADF_Cal(X)
```

způsobem vybíráme body s časovým razítkem nalezených skutečných lokálních extrémů. U dat stažených z investing.com už však do výsledného datasetu zahrnujeme pouze vyhlazené hodnoty po aplikaci gaussova filtru a původní data již nepoužíváme. Tímto jsme vyřešili zkreslení dat z této stránky před počátkem roku 2022, které máme uložené pouze s přesností na pět minut.

4.1.4 Stacionarita

Časové řady jsou považovány za stacionární, pokud se jejich statistické vlastnosti, jako například momenty typu průměr nebo rozptyl, nemění v čase a zůstávají konstantní. Stacionarita zaručí, že každý bod je svojí hodnotou nezávislý na ostatních bodech v datasetu. Toto neznamená, že všechny body v datasetu musí být stejné, ale figuruje zde garance zachování stejného chování vývoje hodnoty napříč celým datasetem. Při pohledu na graf stacionárních časových řad pak nebude možné pozorovat opakující se nebo dlouhodobý trend. Mnoho predikčních modelů zpracovávajících časové řady stacionaritu přímo vyžaduje. Jedná se například o řešení pomocí ARIMA modelu.[53]

Způsobů pro ověření stacionarity existuje více. Nejzákladnější z nich je rozhodnutí na základě vizualizace. Pokud je možné po vytvoření grafu dat v čase pozorovat, že daty se šíří trend, nejedná se o data stacionární. Dalším způsobem je využití autokorelační funkce. Ta porovnává data s jejich zpožděnou kopií nazývanou lag. Takto je možné odhalit, jak velký lag je nutné zvolit pro snížení korelace dat. Pokud jsou zkoumaná data stacionární, hodnoty pro zvyšující se lag rychle konvergují k nulové hodnotě. Také můžeme využít ADF, rozšířený Dickey-Fullerův statistický test, který nám umožní prokázat hypotézu o pravděpodobnosti stacionarity. Tento test přesněji uvádí, do jaké míry může být naše hypotéza o stacionaritě zamítnuta. Pro samotné otestování hypotézy na našich datech použijeme ADF s kódem 4.1 získaným na webu `towardsdatascience.com` v článku „Why Does Stationarity Matter in Time Series Analysis?“.

Data potřebujeme převádět do stacionárního stavu právě v důsledku jejich předešlé závislosti.

Pokud vstup do našeho predikčního modelu tvoří nestacionární data, mohou se velice lišit přesnost predikovaných výsledků v závislosti na tom, kde se vstupní data nachází v původním datasetu. Nestálost přesnosti predikcí je tedy také jedním z ukazatelů stacionarity, protože k ní může docházet z toho důvodu, že zpracovaný úsek dat nereprezentuje korektně strukturu datasetu. U dat časových řas se však můžeme pokusit o řešení tohoto problému a převodu dat na stacionární s konstantní odchylkou a nulovou střední hodnotou.

První a zároveň nejjednodušší metodou pro převod nestacionárních dat na stacionární je diferencování. Jedná se o odečtení dat od jejich předcházejících hodnot popsaných následující rovnicí:

$$\Delta y(t) = y(t) - y(t-1)$$

Konkrétně se jedná o diference první třídy. Těchto tříd můžeme využít neomezený počet pokud to naše potřeby vyžadují. Druhá třída od sebe odečítá sousední datové body, které byly výsledkem první diference. Jedná se o tento vzorec:

$$\Delta y(t) = y(t) - 2y(t-1) + y(t-2)$$

Ostatní třídy jsou od předchozích odvozeny stejným způsobem. Existuje také časové řady, které nelze převést na stacionární pomocí diferencování. Jedná se například o divergující časové řady, ve kterých se bude vyskytovat rostoucí nebo klesající trend po libovolném množství diferencování. Na tyto řady lze aplikovat logaritmickou transformaci, která data převede na podobu vhodnou pro diferencování.[54]

Data s hodnotami kryptoměn máme v našem předzpracovaném datasetu v současné době zaznamenané pouze jako hodnoty v daných bodech. Tyto data stále disponují trendem a na základě testu ADF zamítáme tvrzení, že se jedná o stacionární data. Data získaná z Google Trends a investing.com se po předzpracování také jevila být nestacionární. Aplikovali jsme na ně tedy první třídu diferencí. Časové údaje, které jsme přidali do výsledného datasetu se však svojí strukturou liší od vývoje hodnot. Narozdíl od výše jmenovaných typů dat se tyto hodnoty vyvíjí pouze kladným směrem. Po provedení diference první třídy jsme převedli tyto údaje do podoby, kde reprezentují velikost časového okna mezi jednotlivými extrémy. Po dalším provedení diference jsme získali rozdíly jednotlivých časových oken. Nad těmito daty jsme tedy provedli diferenci druhé třídy.[55]

4.1.5 Závěr

Rozšířený Dickey-Fullerův test prokázal, že data mohou být stacionární. To vyplývá i z vizuální kontroly datasetu kryptoměny Bitcoin na následujícím grafu. Konstrukce datasetu je vytvořena tak, aby bylo možné vytvářet jednoduše nové sloupce na základě nových dat, které chceme přidat do datasetu. Naším dalším krokem bylo vytvoření modelu, který bude na základě předchozích zpracovaných dat predikovat následující lokální extrém.

4.2 Modely

4.2.1 LSTM

//todo

4.2.2 GRU

//todo

4.2.3 Dropout

//todo

4.2.4 L2

//todo

4.3 Predikce dat

Vstupní data do predikčního modelu jsou tvořena datasetem zkonstruovaným pomocí předzpracovacích metod. Popřípadě z něj také mohou být přímo odvozena. Pro získání výsledných dat používáme alespoň dva modely. Jsou využívány pro samostatnou predikci časového rozpětí mezi posledním lokálním extrémem a nadcházejícím. Také predikujeme změnu hodnoty kryptoměny v následujícím lokálním extrému oproti minulé hodnotě. Jednotlivé kryptoměny obvykle následují ve stejném čase stejný trend. Mnoho z nich bylo totiž odvozeno od kryptoměny bitcoin nebo se s nimi často směna za bitcoin provádí[56]. To znamená, že jejich chování ve stejném časovém úseku je velice podobné. Byli jsme tedy schopní zkonstruovat jeden model pro predikci všech kryptoměn. Tento krok nám velice usnadnil a urychlil postupy při přidávání nových kryptoměn uživatelem. Nevyskytla se pak potřeba vytvářet nový model s každou novou kryptoměnou.

Ačkoliv se ale některé kryptoměny vyvíjí podobně, mají jiné hodnoty. Tento problém jsme byli schopni vyřešit pomocí transformace hodnot na společné měřítko. Pro tento účel jsme využili knihovnu `sklearn` a z ní nástroj na předzpracování `PowerTransformer`. Ten se hlavně používá k tomu, aby distribuce dat přibližně odpovídala Gaussově distribuci. Využijeme přitom výchozí parametry. Na grafu //todo můžeme vidět, že transformace opravdu převedla data na podobné hodnoty a je tedy opravdu možné použít pro jejich predikci stejný model.

Ještě před tím, než jsme začali učit model na našich datech, bylo potřeba správně rozdělit data pro vylepšení odhadu přesnosti našeho modelu. Pokud by se například u našeho modelu projevilo přeučení, mohlo by se stát, že přesnost na datech určených k učení bude mít model vysokou na úkor přesnosti při predikci nových dat, která při učení použita nebyla. Z tohoto důvodu jsme rozdělili data na tři části. První část byla použita pro učení modelu. Další pro určení chyby na datech, ke kterým model při učení neměl přístup a na základě kterých byly vybrány výsledné váhy modelu. Poslední třída dat byla využita pro určení výsledné přesnosti modelu s vybranými váhami. Dále bylo potřeba vybrat způsob měření chyby, které se model dopouští při predikcích. Pro měření predikce spojité hodnoty se nejčastěji používá průměrná absolutní odchylka MAE a průměrná kvadratická odchylka MSE popsané těmito vzorci:

$$MSE = \frac{1}{n}\sum(y - \hat{y})^2$$

$$MAE = \frac{1}{n}\sum|y - \hat{y}|$$

Z nich jsme vybrali MAE, protože dosahoval nižší výsledné průměrné absolutní chyby na testovacích datech. Rozhodli jsme se pro sledování tohoto ukazatele, protože chceme hlavně snížit velikost celkového rozptylu chyb.

4.3.1 Dedikované modely

Jako první návrh jsme zkonstruovali model zobrazený v kódu 4.2. Ten následoval rozložení po vzoru projektu „LSTM Neural Network for Time Series Prediction“[57] z repozitáře na síti `github.com`. Model byl autory používán pro predikci trendu v následujících časových bodech.

Výpis kódu 4.2 1. model pro predikci

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 49, 256)           268288

dropout (Dropout)            (None, 49, 256)           0

lstm_1 (LSTM)                (None, 49, 256)           525312

dropout_1 (Dropout)          (None, 49, 256)           0

lstm_2 (LSTM)                (None, 49, 256)           525312

dropout_2 (Dropout)          (None, 49, 256)           0

lstm_3 (LSTM)                (None, 128)               197120

dropout_3 (Dropout)          (None, 128)               0

dense (Dense)                (None, 1)                 129

=================================================================
Total params: 1,516,161
Trainable params: 1,516,161
Non-trainable params: 0
_________________________________________________________________
```

Tato predikce byla prováděna nad všemi daty z datasetu. Na rozdíl od nich my však používáme lokální extrémy. Při predikci s tímto modelem se projevilo přeučení. Přesnost na trénovacích datech měřena pomocí průměrné absolutní chyby byla 0,00192572901. Chyba na validačních datech se ovšem ke konci trénování zvyšovala na hodnotu nad 1,0. Proto bylo nutné zvolit jiný přístup k predikci.

Tímto přístupem bylo vytvoření mnoha kompaktních modelů, z nichž každý by predikoval část dat. Ty byly rozděleny z původního předzpracovaného datasetu a jednotlivé modely zvlášť predikovaly následující lokální extrém kryptoměny například podle informací z Google Trends, S&P 500 nebo ostatních monitorovaných ukazatelů. Tyto modely byly poté vrstvou v TensorFlow sloučeny do jednoho celku, který vracel jednu hodnotu, kterou mohlo být na základě vysvětlované proměnné interval nebo hodnota následujícího lokálního extrému. Každý z těchto modelů měl architekturu zobrazenou v kódu 4.3. Jejich úspěšnost se výrazně zlepšila a v průběhu tréninku nedocházelo k přeučení, čehož bylo dosaženo zavedením Dropout vrstvy a l2 regularizace. Výsledná hodnota MAE predikce pro časové intervaly byla 0,601726986939748 a pro hodnoty extrémů 0,328593894398835

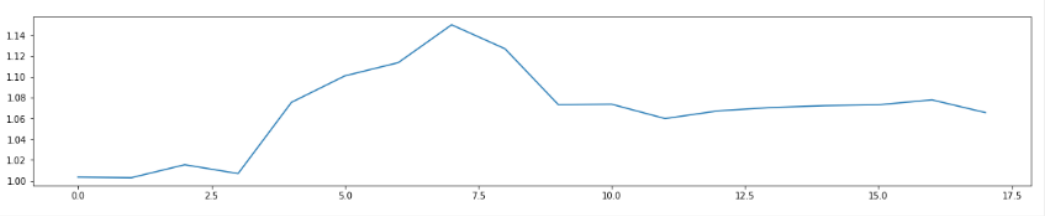
4.3.2 Společný model

Při použití LSTM je možné aplikovat trénování ve směru dopředném i zpětném při použití obousměrné vrstvy, jak je to v této práci „Cryptocurrency price prediction using LSTMs — TensorFlow for Hackers (Part III)“[58]. Ta pro predikci hodnot využívá jeden kompaktní model s LSTM obousměrnými vrstvami. Tento přístup jsme také otestovali a pomocí konstrukce modelu 4.4 jsme dosáhli průměrné absolutní chyby intervalů 0,47399976953694684 a 0,19839759873870

Výpis kódu 4.3 2. model pro predikci

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 lstm (LSTM)                  (None, 49, 20)            2080

 lstm_1 (LSTM)                (None, 4)                 400

 dense (Dense)                (None, 1)                 5

=================================================================
Total params: 2,485
Trainable params: 2,485
Non-trainable params: 0
_________________________________________________________________
```

Obrázek 4.2 Vývoj zisku po jednotlivých transakcích



u hodnot extrémů. Toto řešení bylo dále méně časově náročné na trénování a to až čtyřnásobně. Také se zlepšil čas predikce dat natrénovaným modelem vzhledem k jeho kompaktnosti. Model zároveň neprojevoval známky přeučení a jeho architektura tedy je využita ve finální verzi aplikace.

4.3.3 Genetický algoritmus

Genetický algoritmus je vyhledávací heuristika, která je inspirována teorií přirozené evoluce publikované Charlesem Darwinem. V tomto algoritmu se odráží proces přírodního výběru, kdy jsou nejvýkonnější jedinci vybráni pro pokračování v evoluci. Tento proces začíná zavedením úvodního člena nebo členy, které tvoří první generaci populace. Následně se v rámci této generace provádí dva typy operací. První z nich je křížení. Při tomto procesu se vyberou prvky z populace a jejich vlastnosti se ve specifikovaném měřítku zkříží. Výsledkem je nový jedinec, který je následně přidán do populace. Dalším typem operace je mutace. Jedná se o proces, při němž se z jednoho člena populace vytvoří na základě předem specifikovaných parametrů člen nový. [59]

Pro ohodnocení výkonu jedince je potřeba zavést fitness funkci, která bude určovat, jak si jedinec vede v porovnání s ostatními. Po otestování jedince mu bude uděleno touto fitness funkcí skóre určující pravděpodobnost, se kterou bude vybrán do nové generace.

Na základě predikcí našeho modelu jsme byli schopni určit, kdy nastane další lokální extrém a jaká bude jejich hodnota. Tyto predikce však nemusely být zcela přesné. Jejich nedostatek plynoucí z této nepřesnosti není možné nikdy zcela vyřešit a může se negativně podepsat na našem finančním zisku. Toto můžeme pozorovat na obrázku 4.2 znázorňující výdělek náhodně vybrané kryptoměny v průběhu obchodování. Jeho hodnota znázorňuje poměr vůči původní částce a částce po ukončení dvojice operací nákup a prodej. Pro nákupy byly využity predikované údaje na základě zvolení rozhodnutí v kódu 4.5.

Výpis kódu 4.4 Finální model pro predikci

```
    Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 bidirectional (Bidirectiona  (None, 49, 28)           3248
 l)

 dropout (Dropout)           (None, 49, 28)            0

 bidirectional_1 (Bidirectio  (None, 6)                768
 nal)

 dropout_1 (Dropout)         (None, 6)                 0

 dense (Dense)               (None, 1)                 7

 activation (Activation)     (None, 1)                 0

=================================================================
Total params: 4,023
Trainable params: 4,023
Non-trainable params: 0
_________________________________________________________________
```

Výpis kódu 4.5 Přiřazení akce na základě konfigurace

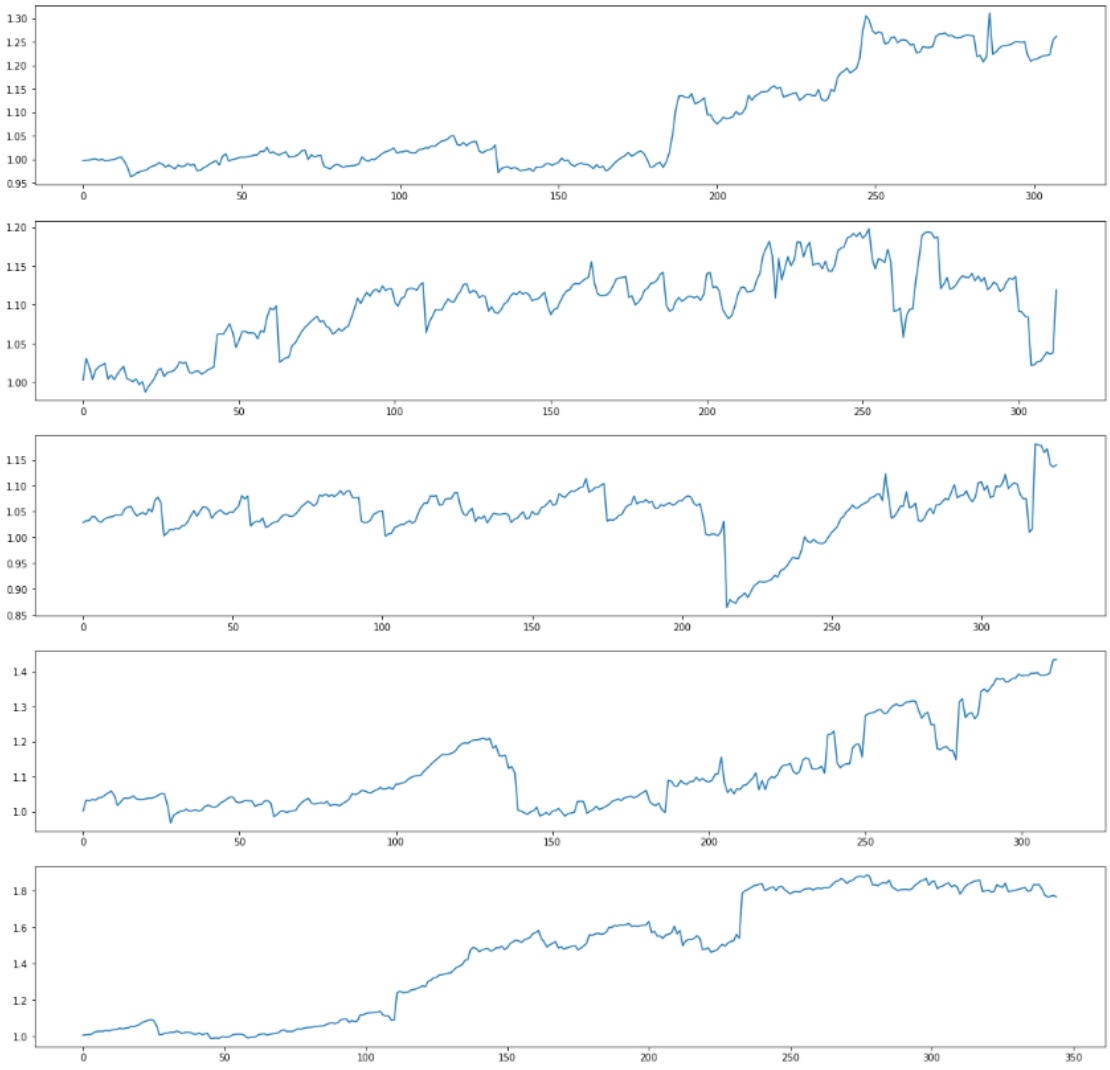
```
rising = config[0]
sinking = config[1]
time_buy_rising = config[2]
time_buy_sinking = config[3]
time_sell_rising = config[4]
time_sell_sinking = config[5]

advice = 'wait'

if growth > rising:
    if duration > time_buy_rising:
        advice = 'buy'
    if duration <= time_sell_rising:
        advice = 'sell'
if growth <= sinking:
    if duration < time_buy_sinking:
        advice = 'buy'
    if duration >= time_sell_sinking:
        advice = 'sell'
```

Pomocí genetického algoritmu jsme usilovali o nalezení takové konfigurace pro nakupování kryptoměn, při které uskutečněné nákupy v rámci obchodovacího období dosáhly kladného finančního zisku pro všechny kryptoměny. Náš genetický algoritmus byl pozměněn oproti své předepsané konstrukci vzhledem ke spojitým hodnotám jedinců populace. Struktura jedince je tvořena šesti hodnotami. Ten je pak použit v kódu 4.5 pod názvem `config`. Pomocí algoritmu se tedy hledají parametry prodeje a nákupu, díky kterým se aplikace může rozhodnout, kolik času musí uplynout po predikovaném lokálním minimu nebo kolik času musí zbývat u predikovaného lokálního maxima, aby mohla kryptoměny nakoupit. Prodej se pak řeší obdobným způsobem. Heuristická funkce, která určuje výkonnost jedinců vyhodnotí pro každou kryptoměnu zisk, kterého lze s danou konfigurací dosáhnout za zadané období. Z jednotlivých zisků kryptoměn se následně vybere nejnižší, kterého tato konfigurace dosáhla a toto číslo je vráceno jako její skóre. Nesnažíme se tedy maximalizovat celkový výdělek v rámci testovaných kryptoměn. Naším cílem je najít strategii, která nám umožní dosáhnout co nejlepšího výsledku napříč všemi kryptoměnami. Tímto se budeme snažit maximalizovat schopnost aplikace doporučovat výdělečnou strategii za každých okolností.

Data byla tvořena predikcemi z necelých 107 po sobě jdoucích dní. Pro predikce jsme využívali náš natrénovaný model a predikovali jsme následující lokální extrém v každé minutě našich dat. Celkem jsme tedy měli přibližně 107*24*60 predikcí seřazených v čase. Každá predikce se vztahovala k času poslední minuty, která byla zahrnuta v časovém okně dat tvořících vstupy pro predikci. Po každé predikci se posunulo toto časové okno o jednu minutu do budoucnosti.

Toto řešení vedlo k pozitivnímu výsledku. Bylo nalezeno množství konfigurací, které maximalizovali minimální výdělek při obchodování s jednotlivými kryptoměnami. Následně bylo vizuálně porovnáno 25 nejvýdělečnějších konfigurací, pro výběr konečné konfigurace, která bude použita pro uskutečňování nákupu a která vykazovala dlouhodobě stoupající trend výdělku při jednotlivých transakcích. Jedná se o konfiguraci `[0,69;-0,8;-96,7;57,43;281,61;15,63]`. Jedná se tedy o konfiguraci, která umožní nákup, pokud je predikovaný růst trendu nad 0.69 a počet minut, do kdy má extrém podle predikce nastat je vyšší, než -96.7. Záporná hodnota je zde právě z důvodu jisté nepřesnosti při predikování časového údaje extrému a podle vyhodnocení genetického algoritmu je toto číslo ze zkoumaných konfigurací nejvýhodnější. Graf, který zaznamenával výdělek při obchodování po každé transakci je znázorněn na obrázku 4.3. Hodnota tohoto grafu je násobek původní hodnoty po provedení dvojice akcí nákupu a prodeje. Celkově pak činil náš dosažený zisk téměř 40%.

Obrázek 4.3 Výdělek nejlepší vybrané konfigurace



Kapitola 5
Uživatelské rozhraní

V této kapitole budeme konstruovat webové grafické uživatelské rozhraní. Budeme řešit rozložení grafických prvků, kaskádové styly, které použijeme, ale také komunikaci se serverem a optimalizaci přenosu dat. Cílem našeho snažení má být přehledná grafická aplikace, která uživateli poskytne dostatečně podrobný přehled informací aniž by působila dezorientačně. Aplikace bude mít možnost zobrazovat jednotlivě i ve vyšším počtu grafy jednotlivých sledovaných ukazatelů. Dále bude disponovat tabulkou, ve které budou zaznamenány predikce našeho natrénovaného modelu a rady uživateli, jak má na trhu s kryptoměnami postupovat na základě těchto predikcí.

5.1 Frontend

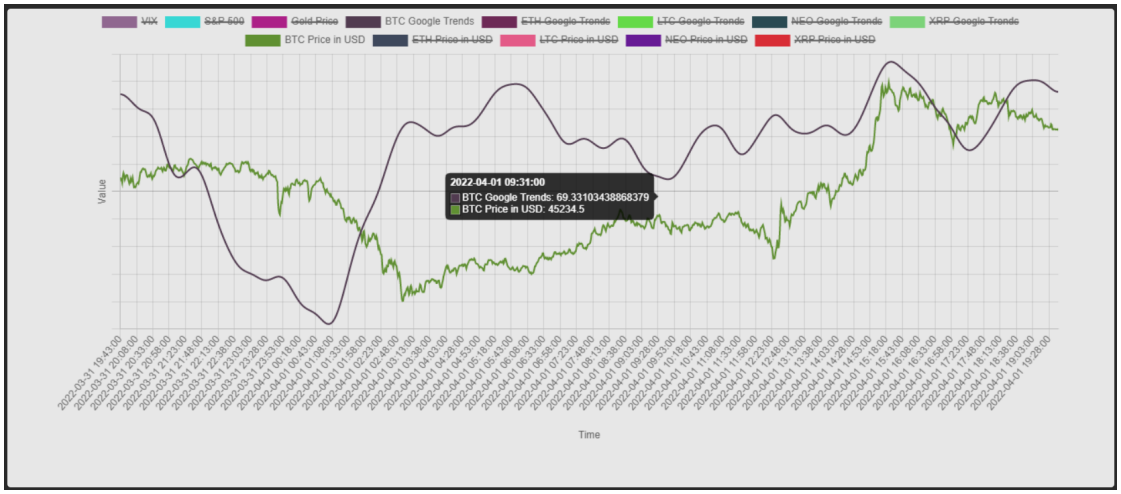
Pro zajištění orientace uživatele v naší aplikaci bylo třeby vytvořit grafické uživatelské rozhraní, které by poskytovalo přehled o vývoji veškerých sledovaných indexů, hodnot a ukazatelů. Soustředili jsme se na to, aby naše stránka poskytovala veškeré informace uživateli bez nutnosti přecházení mezi podstránkami a bez nutnosti scrollování. Tento přístup byl ale vzhledem k množství dat, kterými aplikace disponuje značně optimistický. Přesto jsme ale byli schopni co možná nejvíce omezit nutnost aktivní interakce uživatele s naším programem. Dále jsme se snažili poskytovat informace v takové podobě, aby uživateli přinesly co možná největší informační hodnotu a neznepřehledňovali přitom ostatní obsah na stránce. Výslednou podobu našeho uživatelského rozhraní je možné vidět na obrázcích 5.1 a 5.2.

5.1.1 Vue.js

Dynamická část grafického rozhraní byla konstruována pomocí Vue.js. Prostřednictvím tohoto rozhraní je pro uživatele generován graf, na kterém lze zobrazovat vývoj veškerých sledovaných indexů v reálném čase. Dále Vue.js zobrazuje tabulku, ve které jsou zaneseny predikce budoucích lokálních extrémů a poslední lokální extrémy pro jednotlivé kryptoměny. Ve spodní části aplikace je tlačítko na přidání další sledované kryptoměny, po jehož kliknutí se uživateli zobrazí formulář pro zadání nové kryptoměny.

Graf, se uživateli zobrazuje v horní části aplikace. Zobrazuje přehled událostí v reálném čase v rámci minut. Je tedy možnost v něm zobrazit například vyvíjející se hodnotu kryptoměn, jejich popularitu v rámci vyhledávání na Google Trends nebo hodnoty sledovaných komodit na investing.com. Hodnoty jednotlivých datasetů mají jiné měřítko. Cena Bitcoinu se pohybuje řádově v desítkách tisíc USD. Vyhledávání klíčových slov zase nabývá hodnoty mezi 0 a 100 a hodnota zlata se zase pohybuje v tisících. Oproti tomu DogeCoin má hodnotu desetin USD. Nedávalo by tedy smysl všechny tyto údaje uvádět v rámci jednoho společného měřítka. Místo

Obrázek 5.1 Graf zobrazující vývoj hodnot sledovaných ukazatelů



toho jsme v grafu pro každý dataset vytvořili unikátní svislou osu, čímž bylo zaručeno, že veškerá zobrazená data mají maximální možnou viditelnost rozdílů mezi hodnotami jejich datových bodů v čase. Dále je takto možné snáze pozorovat závislosti mezi jednotlivými datasety. Například jsme schopni pozorovat závislost některých kryptoměn na vývoji hodnoty kryptoměny Bitcoin nebo vývoji kryptoměn na datech z Google Trends. Požadavek na nová data je zasílán každých 20 vteřin v rámci asynchronní funkce, ve které se dále čeká pomocí klíčového slova *await* na odpověď serveru a následné zpracování dat.

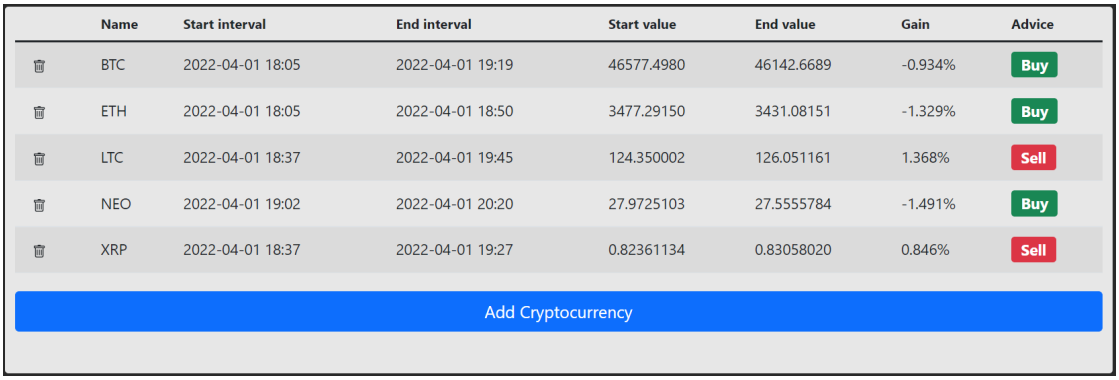
Tento přístup získávání dat ze serveru je označován jako promise-based HTTP klient. Promise neboli příslib označuje získání hodnoty, která nemusí být nutně známá při jeho vytvoření. Namísto okamžitého vrácení konečné hodnoty po volání se tedy vrátí jen jakási záruka, že data budou obdržena. Tento přístup nevylučuje neúspěch, nicméně řeší problém s čekáním na odpověď serveru. Nenastává tedy kolize mezi procesem získání dat a procesem pro jejich zpracování.[60]

Protože v grafu se zobrazují minutová data a funkce v našem frontendu odesílají požadavek a nová data každých 20 vteřin, bylo nutné optimalizovat náš kód. Nemělo smysl aktualizovat náš graf ani odesílat totožná data, pokud během posledních 20 vteřin nedošlo k jejich změně. Tato změna byla řízena posledním zaznamenaným časovým údajem u kryptoměn. Na rozdíl od ostatních dat nejsou data kryptoměn doplňováni poslední zaznamenanou hodnotou mezi posledním časovým údajem a současnou minutou. Zajistili jsme tedy, že data byla odeslána a jejich vizualizace následně aktualizovány pouze v případě, kdy se v datech kryptoměn vyskytuje novější časový údaj, než v předchozím volání. Pokud se tedy stane, že se data nezmenila, odesílá se pouze prázdná šablona jejich rozložení společně s hodnotou proměnné, která nabývá hodnot 0 a 1. Tyto hodnoty se mění pokaždé, kdy se datum v datech aktualizuje. V kódu Vue.js našeho front-endu jsme nastavili sledování změny proměnné v datech našeho grafu. Pokud tato proměnná změnila svojí hodnotu, aktualizovali jsme data v grafu. V opačném případě je takto signalizováno, že obdržená odpověď neobsahovala data a nedošlo k jejich změně. Tedy není potřeba provádět aktualizaci grafu a data budou zahozena.

Toto provedení umožňuje vytvoření komplexního způsobu aktualizace dat a jeho použití bylo popsáno v článku „Getting Started with vue-chartjs“[61]. Výslednou podobu našeho grafu je možné vidět na obrázku 5.1.

V dolní části aplikace se zobrazuje tabulka s přehledem. Je v ní zobrazen poslední zaznamenaný lokální extrém a predikovaný následující lokální extrém. Oba z nich jsou reprezentovány hodnotou a časem. V případě posledního zaznamenaného lokálního extrému se jedná o jeho přesný čas a hodnotu. V případě následujícího lokálního extrému je v tabulce zobrazena jeho predikovaná hodnota a čas, kdy by měl nastat. Dále se v tabulce vyskytuje jméno

Obrázek 5.2 Tabulka s predikcemi a radami pro uživatele



kryptoměn, pro které byly lokální extrémy zvoleny. Jména těchto kryptoměn lze do tabulky přidávat prostřednictvím tlačítka pro přidání kryptoměny vyskytujícího se pod tabulkou. Pokud si uživatel přeje některou evidovanou kryptoměnu odebrat, lze kliknout na ikonu odpadkového koše v tabulce. Po kliknutí na tuto ikonu se odešle dotaz na server s požadavkem odebrání této kryptoměny. V tabulce se pak její odebrání projeví okamžitě. V tabulce se dále zobrazuje vyhodnocení akce prostřednictvím konfigurace získané naším genetickým algoritmem. Na základě tohoto vyhodnocení se uživateli zobrazují nápisy *Buy*, *Wait* a *Sell*. Dotaz na nová data tabulky se stejným způsobem jako u grafu odesílá obdobně každých 20 vteřin. Výslednou podobu naší tabulky je možné vidět na obrázku 5.2.

5.1.2 CSS

Pro docílení výsledného vzhledu uživatelského rozhraní bylo potřeba použít správných vizuálních prvků. K tomuto účelu jsme využili kaskádových stylů. Ty reprezentují způsob úpravy vzhledu dokumentů napsaných v jazyce HTML. Popisují, jakým způsobem mají být prvky vykresleny na obrazovce. Umožňuje například změny barev a stylů písma, ale také zaoblení rohů nebo reaktivitu prvků po jejich interakci v podobě například změny barvy.[62]

Pro úpravu vzhledu naší aplikace jsme použili předpřipravené kaskádové styly z frameworku bootstrap. Pod tímto názvem je označován open-source CSS framework zaměřený na responzivní front-end webový vývoj. My jsme konkrétně použili CSS specifikaci `https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css` napříč celým projektem. To nám umožnilo získat a použít vzhled pro kontejnery tvořící pozadí prvků, tabulky, tlačítka text a textová pole.

5.1.3 HTML

HTML je markup jazyk, s jehož pomocí můžeme definovat struktury obsahu naší webové aplikace. Právě prostřednictvím tohoto jazyka můžeme správně umístit do grafické části naší aplikace graf, tabulku a potřebné ovládací prvky. Každý z těchto prvků je vložen do kontejneru, který je tvořen strukturou div a od tmavého jednobarevného pozadí odděluje prvky v popředí. Samotný kontejner, na kterém se naše grafické prvky nachází má světlou barvu a zaoblené rohy. Jednotlivé kontejnery od sebe dále odděleny viditelným paddingem. Tento vzhled jednak logicky odděluje grafické prvky a zároveň na uživatele působí moderně a přívětivě. V tabulce je definovaný v-for cyklus, díky kterému můžeme pomocí Vue.js aktualizovat hodnoty v ní. Posledním prvkem v HTML dokumentu, který specifikuje rozklad našich prvků je formulář na přidání nových kryptoměn. Ten se zobrazí pouze ve chvíli, kdy uživatel klikne na tlačítko přidání. Možnost skrýt prvky a zobrazit je pouze po akci uživatele nám poskytuje BootStrap CSS knihovna. Pro využití

Výpis kódu 5.1 Vykreslování šablony

```
@app.route('/')
def get_view():
    return render_template('index.html')
```

této schopnosti stačí pouze nastavit třídu kontejneru mizícího prvku jako *modal fade* a specifikovat jeho id v tlačítku pro jeho zviditelnění.[63]

5.2 Backend

Pokud chceme používat Flask framework, musíme napřed vytvořit instanci naší webové aplikace. Následně se tato aplikace spustí příkazem "run", v rámci kterého se také specifikuje adresa, na které má být webové rozhraní přístupné. Tímto způsobem budeme ve výsledku schopni zobrazit naší aplikaci na adrese `http://localhost` a výchozím portu 5000. Máme také možnost prostřednictvím tohoto frameworku zobrazovat naší webovou aplikaci. Musíme ale napřed specifikovat, jakým způsobem k ní chceme přistupovat. K tomuto účelu slouží dekorátory, v rámci kterých jsme definovali cestu k jednotlivým částem aplikace.

Kromě načítání a vykreslování souborů s HTML strukturou můžeme také stejným způsobem přistupovat k dynamickým prvkům v jazyce JavaScript a kaskádovým stylům. Flask disponuje metodou `render_template`, která nám právě tento přístup umožnila. Její použití včetně dekorátoru je v kódu 5.1.

Specifikovali jsme zde cestu `'/'`. Tato cesta specifikuje strukturu za zadanou adresou serveru a portu. Plnohodnotná adresa má tedy tvar `http://localhost:5000/`. Soubory s kódem pro frontend musí být umístěny v přesně specifikované adresářové struktuře, kterou Flask framework respektuje. Tu jsme vytvořili následovně:

```
static
    js
        chart.js
        table.js
    styles
        styles.css
templates
    index.html
```

Obdobně, jako funkci pro vykreslení stránky jsme také definovali funkce, které na frontend zasílají data. Jejich návratové hodnoty jsou ovšem předzpracovaná data ve formátu JSON. Na adrese s cestout */data/* jsou přístupná data pro konstrukci nebo aktualizaci tabulky s predikcemi, a návrhy na akci. Na adrese */add/* se zasílají požadavky na přidání kryptoměny do aplikace. Jednotlivé kryptoměny jsou pak specifikovány v textovém řetězci požadavku. Adresa */remove/* je volána frontendem poté, co uživatel kliknul na tlačítko k odebrání. Dále adresa */chart/* poskytuje veškerá data potřebná pro konstrukci a aktualizaci grafu s vývojem hodnot.

Kapitola 6

Architektura a vývoj aplikace

6.1 Oddělení kontejnerů

Po naší aplikaci jsme vyžadovali, aby byla odolná vůči případným selháním jejích dílčích částí. Také jsme chtěli mít možnost během jejího chodu nasazovat nové podpůrné funkce a řešení. Z tohoto důvodu jsme si pro běh zvolili kontejnerovou strukturu, která je zprostředkována platformou Docker. Jednotlivé části jsou schopny fungovat jako samostatný celek a je tedy potřeba je vhodně využívat v rámci jedné aplikace. Důležité bylo zařídit mezi těmito samostatnými aplikacemi a databází komunikaci, v rámci které probíhá jejich interakce.

Konfiguraci databáze PostgreSQL jsme specifikovali v souboru docker-compose.yml, který se v příkazu k sestavení `docker-compose -f .devcontainer/docker-compose.yml up -d` předá jako parametr. Tímto jsme vytvořili funkční databázi viditelnou a přístupnou pouze ostatními kontejnery definovanými v souboru. Pomocí parametru `restart: unless-stopped` jsme zajistili, že pokud by nastala situace, při které by se vypnul image nebo počítač, pak se kontejner s touto částí aplikace automaticky restartuje. Tento parametr zavedeme u všech kontejnerů.

Ostatní části aplikace nemají definovanou konfiguraci jejich image přímo v konfiguračním souboru docker-compose.yml. Nachází se ve svých dedikovaných složkách. Protože se kód všech částí aplikace až na databázi spouští v programovacím jazyce Python, má jejich image `python:3.8` již předpřipravené prostředí pro tento programovací jazyk. Jedná se o verzi Pythonu, kterou podporují veškeré použité nástroje a která je stále v aktivní podpoře a údržbě. Konfigurace jsou v rámci svých dedikovaných složek specifikovány v souborech Dockerfile a společně s nimi se ve stejném adresáři vyskytují specifikované moduly, knihovny a balíčky vyžadované kódem spouštěným daným kontejnerem. Ty se po prvním spuštění kontejneru automaticky nainstalují. Kontejner pro stahování dat z Google Trends tedy má v požadavcích na python specifikovanou mimo jiné knihovnu requests pro zasílání požadavků. Dockerfile pro kontejner stahující data ze stránky investing.com navíc oproti ostatním obsahuje instalaci aplikací chromium a chromium-driver, které mu umožní přístup na internet prostřednictvím nástroje Selenium. Na konci každé specifikace kontejneru je blokující příkaz, který zapne dílčí program.

Připojení k databázi z kontejnerů probíhá podobným způsobem, jako při přístupu mimo ně. Knihovna psycopg2 poskytuje rozhraní pro připojení a komunikaci. Při pokusu o připojení se specifikuje uživatel, heslo, host, port a databáze. Pokud bychom přistupovali mimo kontejner, nastavili bychom adresu hostu na adresu databáze. Zde je místo adresy specifikované jméno kontejneru, ve kterém se databáze nachází. Docker sám tuto závislost vyřeší a dosadí za jméno kontejneru adresu databáze.

Pro možnost přistupovat a zobrazovat grafické uživatelské rozhraní mimo oddělené prostředí kontejnerů bylo potřeba napřed specifikovat porty, které byly vystaveny pro vnější komunikaci. Tím jsme dosáhli, že se adresa, na které mělo být webové rozhraní přístupné, přepsala na `0.0.0.0`. Tímto způsobem se adresa naší aplikace veřejně vystavila v rámci sítě, na které se naše zařízení vyskytuje. To v praxi znamená, že naše aplikace je přístupná i mimo síť kontejneru na zařízení, které docker knontejnery a tedy i naší aplikaci hostuje.

Kapitola 7

Závěr

//todo

Příloha A

Nějaká příloha

Sem přijde to, co nepatří do hlavní části.

Bibliografie

1. [B.r.]. Dostupné také z: https://web.archive.org/web/20180514234543/https://www.investopedia.com/terms/b/binance-exchange.asp.
2. [B.r.]. Dostupné také z: https://www.trendmicro.com/vinfo/us/security/definition/cryptocurrency#:~:text=A%20cryptocurrency%20is%20an%20encrypted,buying%2C%20selling%2C%20and%20transferring..
3. [B.r.]. Dostupné také z: https://en.wikipedia.org/wiki/Cryptocurrency.
4. [B.r.]. Dostupné také z: https://money.usnews.com/investing/cryptocurrency/articles/what-is-proof-of-stake-and-why-is-ethereum-adopting-it.
5. [B.r.]. Dostupné také z: https://www.investopedia.com/articles/investing/052014/why-bitcoins-value-so-volatile.asp#:~:text=Bitcoin%27s%20price%20fluctuates%20because%20it,together%20to%20create%20price%20volatility..
6. [B.r.]. Dostupné také z: https://www.investopedia.com/terms/h/hodl.asp.
7. [B.r.]. Dostupné také z: https://en.wikipedia.org/wiki/Market_liquidity.
8. [B.r.]. Dostupné také z: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2183806.
9. [B.r.]. Dostupné také z: https://www.iflr.com/article/b1sjckt2md0cg2/primer-high-frequency-trading#:~:text=High%20frequency%20trading%20(HFT)%2C%20or%20systematic%20trading%2C%20is,orders%20at%20extremely%20high%20speeds..
10. [B.r.]. Dostupné také z: https://www.sec.gov/news/testimony/2010/ts051110mls.htm.
11. [B.r.]. Dostupné také z: https://www.sciencedirect.com/science/article/abs/pii/S0165176517303804.
12. [B.r.]. Dostupné také z: https://www.sciencedirect.com/science/article/pii/S2405918821000027.
13. [B.r.]. Dostupné také z: https://arxiv.org/pdf/1904.05315.pdf.
14. [B.r.]. Dostupné také z: https://www.researchgate.net/publication/337442647_Advanced_social_media_sentiment_analysis_for_short-term_cryptocurrency_price_prediction.
15. [B.r.]. Dostupné také z: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0212320.
16. [B.r.]. Dostupné také z: https://www.researchgate.net/publication/348981358_Event-Driven_LSTM_For_Forex_Price_Prediction.
17. [B.r.]. Dostupné také z: https://www.investopedia.com/terms/e/elliottwavetheory.asp.

18. [B.r.]. Dostupné také z: https://www.investopedia.com/ask/answers/030415/what-zig-zag-indicator-formula-and-how-it-calculated.asp#:~:text=The%20zig%20zag%20indicator%20is,noise%20of%20everyday%20market%20conditions..
19. [B.r.]. Dostupné také z: https://www.sciencedirect.com/science/article/abs/pii/S0360544219314768.
20. [B.r.]. Dostupné také z: https://www.jobsity.com/blog/8-reasons-why-docker-matter-for-devs#:~:text=Docker%20is%20a%20tool%20designed,it%20as%20a%20single%20package..
21. [B.r.]. Dostupné také z: https://codeburst.io/how-to-start-docker-containers-automatically-ec0545c392e4.
22. [B.r.]. Dostupné také z: https://www.javatpoint.com/types-of-databases.
23. [B.r.]. Dostupné také z: https://en.wikipedia.org/wiki/ACID.
24. [B.r.]. Dostupné také z: https://www.fivetran.com/blog/postgresql-vs-mysql#:~:text=PostgreSQL%20is%20an%20object%2Drelational,%2C%20ACID%2Dcompliant%20storage%20engine..
25. [B.r.]. Dostupné také z: https://web.archive.org/web/20120623165941/http:/cutter.rexx.com/~dkuhlman/python_book_01.html.
26. [B.r.]. Dostupné také z: https://www.artima.com/articles/the-making-of-python.
27. [B.r.]. Dostupné také z: https://steelkiwi.com/blog/python-for-ai-and-machine-learning/#:~:text=Python%20offers%20concise%20and%20readable,technical%20nuances%20of%20the%20language..
28. [B.r.]. Dostupné také z: https://jasonwatmore.com/post/2020/04/30/vue-fetch-http-get-request-examples.
29. [B.r.]. Dostupné také z: https://www.rascasone.com/cs/blog/co-je-framework-vuejs.
30. [B.r.]. Dostupné také z: https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93viewmodel.
31. [B.r.]. Dostupné také z: https://docs.python-requests.org/en/latest/.
32. [B.r.]. Dostupné také z: https://en.wikipedia.org/wiki/Chromium_(web_browser).
33. [B.r.]. Dostupné také z: https://www.selenium.dev/documentation/.
34. [B.r.]. Dostupné také z: https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API.
35. [B.r.]. Dostupné také z: https://web.archive.org/web/20171117015927/http:/flask.pocoo.org/docs/0.10/foreword.
36. [B.r.]. Dostupné také z: https://palletsprojects.com/p/jinja/.
37. [B.r.]. Dostupné také z: https://wsgi.readthedocs.io/en/latest/what.html.
38. [B.r.]. Dostupné také z: https://stackoverflow.com/questions/22259847/application-not-picking-up-css-file-flask-python.
39. [B.r.]. Dostupné také z: https://www.tensorflow.org/tutorials.
40. [B.r.]. Dostupné také z: https://medium.com/analytics-vidhya/pytorch-is-growing-tensorflow-is-not-6986c5e52d6f#:~:text=In%202019%E2%80%9301%E2%80%9301%2C,Tensorflow%20and%2020%25%20for%20Pytorch..
41. [B.r.]. Dostupné také z: https://towardsdatascience.com/pytorch-vs-tensorflow-spotting-the-difference-25c75777377b#:~:text=So%2C%20both%20TensorFlow%20and%20PyTorch,from%20which%20you%20may%20choose..

42. [B.r.]. Dostupné také z: https://blog.cryptocompare.com/a-complete-starter-guide-to-the-cryptocompare-api-29b4bb1ca25.
43. [B.r.]. Dostupné také z: https://www.coindesk.com/price/bitcoin/.
44. [B.r.]. Dostupné také z: https://www.youtube.com/watch?v=i9N_LrnDUnY&t=146s.
45. [B.r.]. Dostupné také z: https://developer.twitter.com/en/docs/twitter-api/rate-limits.
46. [B.r.]. Dostupné také z: https://github.com/Altimis/Scweet.
47. [B.r.]. Dostupné také z: https://www.statista.com/statistics/216573/worldwide-market-share-of-search-engines/#:~:text=Google%20has%20dominated%20the%20search,share%20as%20of%20June%202021..
48. [B.r.]. Dostupné také z: https://github.com/GeneralMills/pytrends.
49. [B.r.]. Dostupné také z: https://www.spglobal.com/spdji/en/indices/equity/sp-500/#overview.
50. [B.r.]. Dostupné také z: https://www.ig.com/en/indices/what-is-vix-how-do-you-trade-it.
51. [B.r.]. Dostupné také z: https://www.investopedia.com/terms/d/djia.asp.
52. [B.r.]. Dostupné také z: https://en.wikipedia.org/wiki/Gaussian_filter#:~:text=In%20electronics%20and%20signal%20processing,would%20have%20infinite%20impulse%20response)..
53. [B.r.]. Dostupné také z: https://stats.stackexchange.com/questions/19715/why-does-a-time-series-have-to-be-stationary.
54. [B.r.]. Dostupné také z: https://discuss.analyticsvidhya.com/t/what-are-the-ways-through-which-we-can-make-a-time-series-stationary/471/2.
55. [B.r.]. Dostupné také z: https://otexts.com/fpp2/stationarity.html.
56. [B.r.]. Dostupné také z: https://www.commpro.biz/why-does-bitcoin-have-such-a-big-influence-on-other-cryptocurrencies/.
57. [B.r.]. Dostupné také z: https://github.com/jaungiers/LSTM-Neural-Network-for-Time-Series-Prediction/blob/master/config.json.
58. [B.r.]. Dostupné také z: https://towardsdatascience.com/cryptocurrency-price-prediction-using-lstms-tensorflow-for-hackers-part-iii-264fcdbccd3f.
59. [B.r.]. Dostupné také z: https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3#:~:text=A%20genetic%20algorithm%20is%20a,offspring%20of%20the%20next%20generation..
60. [B.r.]. Dostupné také z: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise.
61. [B.r.]. Dostupné také z: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise.
62. [B.r.]. Dostupné také z: https://developer.mozilla.org/en-US/docs/Web/CSS.
63. [B.r.]. Dostupné také z: https://getbootstrap.com/docs/4.0/components/modal/.

Obsah přiloženého média

- `readme.txt` ..... stručný popis obsahu média
- `exe` ..... adresář se spustitelnou formou implementace
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF