Vysoká škola ekonomická v Praze

Fakulta informatiky a statistiky

Datová analýza českého realitního trhu

BAKALÁŘSKÁ PRÁCE

Studijní program: Aplikovaná informatika

Studijní obor: Aplikovaná informatika

Autor: Sívek Viktor

Vedoucí bakalářské práce: Ing. Chudán David, Ph.D.

Praha, květen 2023

Poděkování

Tímto děkuji panu Ing. Davidu Chudánovi, Ph.D. za vedení své bakalářské práce a za jeho trpělivost, odborné vedení a cenné rady. Dále děkuji všem, kteří mě při psaní mé práce podporovali.

Abstrakt

Cílem mé bakalářské práce je sesbírat konkrétní data a aplikovat na ně techniky data miningu. Tato práce se zaměřuje na získávání dat pomocí web scrapingu z webového realitního portálu. Získaná data mají za účel reflektovat aktuální stav českého realitního trhu. Dále se práce zabývá datovou analýzou pomocí Python knihoven a výsledky analýzy jsou vizualizovány. Nakonec jsou výsledky porovnány s daty z minulého roku a popsán vývoj dat v čase. Práce se dělí na teoretickou a praktickou část.

V teoretické části je představena oblast web scrapingu, jeho principy, existující řešení a etický aspekt. Dále práce popisuje čistění dat a možnosti ukládání dat a práce s nimi. Jsou zde také popsány populární metodiky data miningu a analýzy dat, s důrazem na metodiku CRISP-DM. Poslední část je věnována teorii vizualizace.

Praktická část práce začíná sběrem dat z webového realitního portálu www.bezrealitky.cz, včetně tvorby skriptu v jazyce Python. Následuje představení nástrojů použitých v průběhu analýzy a seznámením s doménovou oblastí realitních dat. Získaný dataset je poté předzpracován v prostředí Jupyter Notebook. Výsledná data jsou nejprve analyzována pomocí explorační analýzy a poté s využitím klasifikačních, regresních a popisných metod data miningu. Na závěr jsou výsledky porovnány s daty z minulého roku, popsán vývoj dat v čase a celá práce je shrnuta.

Klíčová slova

Web scraping, data mining, datová analýza, data cleaning

Abstract

The goal of my undergraduate thesis is to collect specific data and apply the data mining technique to it. This thesis focuses on data extraction using web scraping from a real estate portal. The data collected is intended to reflect the current state of the Czech real estate market. Furthermore, the thesis deals with data analysis using Python libraries and the results of the analysis are visualized in Dash application. Finally, the results are compared with the data from last year and the evolution of the data over time is described. The thesis is divided into theoretical and practical parts.

The theoretical part introduces the field of web scraping, its principles, existing solutions and ethical aspect. Furthermore, the thesis describes data cleansing and the possibilities of storing and working with data. Popular data mining and data analysis methodologies are also described, with emphasis on CRISP-DM methodology. The last section is devoted to visualization theory.

The practical part of the thesis starts with data collection from a web-based real estate portal www.bezrealitky.cz, including the development of a Python script. This is followed by an introduction of the tools used during the analysis and an introduction to the domain of real estate data... The obtained dataset is then preprocessed in the Jupyter Notebook environment. The resulting data is first analyzed using exploratory analysis and then using classification, regression and descriptive data mining methods. Finally, the results are compared with the previous year's data, the evolution of the data over time is described, and the entire paper is summarized.

Keywords

Web scraping, data mining, data analysis, data cleaning

Obsah

Úvod

Realitní trh představuje důležitý a dynamický sektor ekonomiky, který má zásadní dopad na životy jednotlivců, rodin i celé společnosti. Český realitní trh, stejně jako realitní trhy v jiných zemích, prochází neustálým vývojem a změnami, které ovlivňují jak nabídku, tak poptávku po nemovitostech. V této bakalářské práci se zaměřím na datovou analýzu českého realitního trhu s cílem sesbírat konkrétní data a aplikovat na ně techniky data miningu.

Mým hlavním motivem pro zvolení tohoto tématu je osobní zájem o realitní trh, který pramení z mého rodinného pozadí. Moje rodina se již řadu let pohybuje v realitním sektoru, a právě díky tomu jsem měl možnost získat hlubší pochopení o fungování trhu, potřebách zákazníků a významu analýzy dat pro úspěšné podnikání v oblasti realit.

V průběhu mé práce budu využívat metodu web scrapingu, která umožňuje shromažďování velkého množství dat z různých online zdrojů, jako jsou realitní portály a webové stránky realitních kanceláří. Tato metoda mi umožní získat aktuální a relevantní data o nabídkách nemovitostí, cenách, lokalitách a dalších důležitých parametrech. Následně budu data analyzovat a zpracovávat za pomoci statistických metod a vizualizací, které mi umožní identifikovat klíčové trendy a vzory na českém realitním trhu.

Struktura bakalářské práce je následující: V první části práce se zaměřím na teorii web scrapingu, na základní pojmy a principy spojené s metodou web scrapingu, včetně nástrojů a technik pro extrakci dat z online zdrojů a způsobů, jak zajistit kvalitu a spolehlivost získaných dat. Ve druhé části se zaměřím na teorii dobývání znalostí z databází, kde se budu věnovat teoretickému aspektu včetně procesu, metodiky, metod a technik používaných pro analýzu, interpretaci a vizualizaci dat. Ve třetí části se zaměřím na praktickou implementaci web scrapingu, kde popíši proces sběru dat z realitního portálu www.bezrealitky.cz, předzpracování a čištění dat pro následné analýzy. V poslední části práce budu aplikovat metody a techniky dobývání znalostí z databází na získaná data, provádět analýzu a interpretaci výsledků a zkoumat kvalitu zvolených modelů. V závěru zhodnotím, zda se mi podařilo naplnit definované cíle a shrnu výsledky práce.

Jako metodiku práce jsem zvolil pozorování, deskripci nasbíraných dat a teoretických částí práce, komparaci modelů, technik scrapování a modelů data miningu a analýzu konkrétních sesbíraných dat a výkonosti modelů.

1 Web scraping

V této části si popíšeme, co to je scrapování webových stránek. Poté se budeme věnovat různým technikám web scrapingu, parsování jazyka HTML, regulárním výrazům, headless prohlížečům a použití nástrojů Selenium a Webdriver. Neopomeneme ani etický a právní aspekt, včetně důležitosti dodržování pokynů robots.txt. Pro další ilustraci probíraných konceptů si předvedeme implementaci scraperu založeného na systému Selenium.

1.1 Úvod do web scrapingu

Web scraping je jedním ze základních procesů při získávání strukturovaných dat z webových stránek a hraje klíčovou roli v oblasti datové vědy a výzkumu (AYDIN O., 2018). S rozvojem digitálního prostředí se stále více tvůrců marketingových reportů obrací k metodám sběru dat z webu, aby získali cenné informace o chování spotřebitelů a trendech na trhu. V posledních letech se využívání webových dat výrazně rozšířilo a přispělo k velkému množství výzkumných publikací.

Scraping webových stránek je technika, která zahrnuje vytvoření softwaru pro automatické získávání informací, často s cílem sestavit rozsáhlé soubory dat pro analýzu. Analytici mohou například získávat data z online platforem, jako je Amazon, aby studovali recenze spotřebitelů a získali přehled o nákupních zvyklostech. Tato metoda může být obzvláště atraktivní, protože umožňuje shromažďovat veřejně přístupná data, aniž by bylo nutné přímé zapojení poskytovatelů dat. (BOEGERSHAUSEN, 2022, s. 2-5)

1.2 Techniky web scrapingu

Pro získání strukturovaných dan můžeme využít různé techniky, z nichž každá má své jedinečné funkce a schopnosti. Mezi běžné techniky využívané při web scrapingu patří např:

- HTML parsery
- Regulární výrazy
- Rozhraní API (APIs)
- Headless browsers (AYDIN O., 2018, s. 6)

HTML parsing je technika, která rozkládá zdrojový kód webové stránky do strukturovaného formátu, což usnadňuje extrakci konkrétních datových prvků. To lze provést pomocí specializovaných knihoven, jako je Beautiful Soup pro Python nebo Jsoup pro Javu, které poskytují metody pro navigaci a manipulaci se stromem DOM (Document Object Model) jazyka HTML (KHALIL, 2017).

Regulární výrazy jsou výkonnou technikou, jelikož umožňují extrahovat relevantní strukturované informace z volného textu. V kontextu web scrapingu lze regulární výrazy použít k identifikaci konkrétních vzorů ve zdrojovém kódu HTML, pro identifikaci a extrakci požadovaných dat, které jsou předmětem zájmu. V některých případech jsou informace ukryty v atomických hodnotách nebo rozptýleny v dokumentu HTML, takže je náročné využít strukturu dokumentu a jeho značení pro extrakci. Právě v tomto případě regulární výrazy ztrácí svou efektivitu (MUNZERT, 2017).

Některé webové stránky nabízejí rozhraní API (Application Programming Interfaces), která umožňují strukturovaný přístup k jejich datům, což umožňuje efektivní a kontrolované získávání dat (MUNZERT, 2017).

Podle studie z Národního statistického úřadu (NSI) se můžeme se podívat na efektivnost různých technik web scrapingu s cílem udržet si kontrolu nad procesem získávání dat. Aby se technika minimalizovala závislost na externích organizacích a režijní náklady, vyhodnotil NSI několik softwarových řešení s otevřeným zdrojovým kódem pro sběr, zpracování a ukládání dat. Mezi vybraná řešení patří Robot Framework, Scrapy, Apache Nutch a RSelenium. Výběr těchto řešení byl založen na specifických kritériích, která zajišťovala, že jsou volně dostupná, mají otevřený zdrojový kód a snadno se udržují. Kromě toho NSI upřednostňoval řešení s aktivní technickou podporou ze strany dynamické komunity uživatelů. Tento přístup byl inspirován předchozími zkušenostmi jiných statistik NSI v oblasti web scrapingu, které ukazují, jak je důležité zachovat si kontrolu nad produkčním procesem a využívat nákladově efektivní a spolehlivé nástroje. (OANCEA Bogdan, 2022)

1.3 Headless Browsers

Headless browsery jsou webové prohlížeče, které pracují bez grafického uživatelského rozhraní, což jim umožňuje přistupovat k webovým stránkám a získávat informace, aniž by se obsah zobrazoval jakémukoli lidskému uživateli. Tyto prohlížeče jsou využívány jinými programy k přístupu k obsahu webových stránek a jeho analýze.

Jednou z klíčových výhod headless browserů je jejich schopnost interpretovat webové stránky podobně jako tradiční prohlížeče. Headless browsery mohou analyzovat dynamické stránky náročné na JavaScript a vykreslovat konečný DOM, čímž poskytují přístup k plně načtenému dynamickému obsahu. Dále umožňují klikat na odkazy, vyplňovat formuláře, a dokonce spravovat stahování, což poskytuje cenné informace o tom, jak by webové stránky vypadaly a fungovaly pro koncové uživatele (GIRDWOOD, 2009).

1.3.1 Selenium

Nástroj Selenium s otevřeným zdrojovým kódem poprvé vyvinul v roce 2004 Jason Huggins ve společnosti ThoughtWorks za účelem automatizace testování webových aplikací. Jeho hlavním účelem je umožnit testování chování front-endu ve více prohlížečích, což vedlo k jeho široké popularitě mezi vývojáři a testery. S rostoucí poptávkou po automatizovaném testování se Selenium dočkalo mnoha iterací a nyní zahrnuje sadu nástrojů, včetně

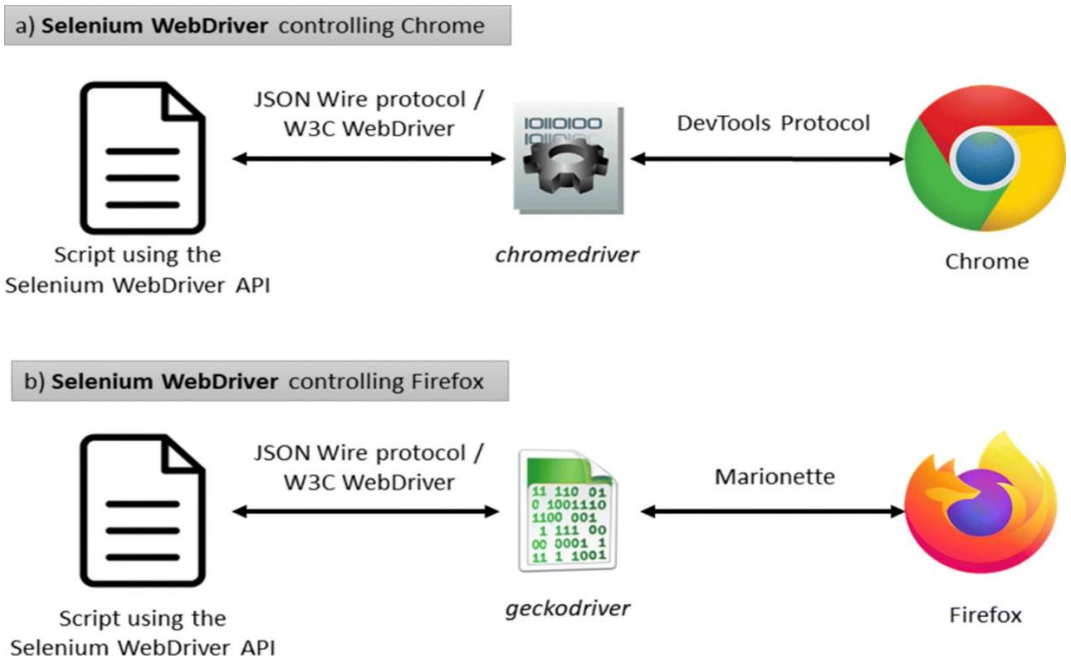
Selenium WebDriver, který automatizuje testovací případy webových aplikací.

Architektura nástroje Selenium se skládá ze tří hlavních součástí: Selenium klientské knihovny, ovladače prohlížeče a webové prohlížeče. Komunikace mezi těmito komponenty probíhá prostřednictvím protokolu JSON. Flexibilita nástroje Selenium umožňuje podporu různých operačních systémů, jako jsou Windows, MacOS, Unix a Linux, a různých programovacích jazyků, včetně Pythonu, Javy, Ruby a mnoha dalších. Nástroj je oblíbený pro svou snadnou implementaci, opakované použití a minimální nároky na hardwarové prostředky. Celkově je Selenium díky své open-source povaze, kompatibilitě napříč platformami a široké podpoře jazyků preferovanou volbou pro testování a scrapování webových aplikací (RAGHAVENDRA Sujay, 2021).

1.3.2 Webdriver

WebDriver je rozhraní API, které Selenium používá k interakci s různými webovými prohlížeči. Poskytuje konzistentní rozhraní pro automatizaci prohlížeče a umožňuje vývojářům psát skripty pro scraping webu, které lze spouštět ve více prohlížečích, například Chrome, Firefox a Edge. Tato kompatibilita napříč prohlížeči zvyšuje univerzálnost a robustnost úloh webového scrapování a zajišťuje, že skripty fungují konzistentně bez ohledu na preferovaný prohlížeč uživatele.

K dosažení této kompatibility používá WebDriver binární soubory závislé na platformě, nazývané ovladače, které slouží jako prostředníci mezi skripty Selenium a prohlížeči. Ovladače překládají příkazy skriptů Selenium do jazyků specifických pro prohlížeč, což umožňuje bezproblémovou interakci s různými webovými prohlížeči. Vývojáři tak mohou vytvářet skripty pro scrapování webu s větší flexibilitou a spolehlivostí, protože vědí, že jejich kód bude v různých prostředích prohlížečů fungovat konzistentně (GARCÍA, 2021).



Obr. 1.1 Selenium WebDriver (GARCÍA, 2021)

1.3.3 Dynamický obsah webové stránky

Dynamické webové stránky zobrazují různý obsah pro různé uživatele při zachování stejného rozvržení a designu. Běžně se implementují za účelem zobrazení často se měnících informací, jako jsou aktualizace počasí nebo ceny akcií. Tyto stránky používají technologie, jako jsou CGI, AJAX, ASP nebo ASP.NET. Na straně serveru, je databáze, což umožňuje oddělit design od obsahu. Existují dva typy dynamických webových stránek, a to se skriptováním na straně klienta a skriptováním na straně serveru. Skriptování na straně klienta generuje obsah v počítači uživatele, zatímco skriptování na straně serveru generuje obsah při načítání stránky (MKS., 2019).

Vyhledávání dynamických webových stránek může být náročné, protože obsah se mění nebo načítá asynchronně, aniž by bylo nutné obnovit celou stránku. Webové scrapery musí před extrakcí dat počkat, až se obsah načte a zpřístupní. Bezhlavé prohlížeče, jako je Selenium, mohou zpracovávat dynamický obsah pomocí explicitního nebo implicitního čekání, které umožňuje scraperu pozastavit načítání určitých prvků nebo splnění podmínek před pokračováním v procesu extrakce (MITCHELL, 2018).

1.4 Etický a právní aspekt

Při scrapování webu je potřeba brát ohled i na etickou a právní stránku. Pokud jsou údaje na internetu veřejně dostupné, lze obecně považovat za legální tyto údaje shromažďovat. (LAUREN KLEIN, 2021). I přesto terms of services (TOS) hrají zásadní roli při definování právnívh hranic web scrapingu. Nástroje používané k web scrapingu, jako jsou roboti, crawlery, pavouci a scrapery, jsou v dokumentech TOS často uváděny zaměnitelně, což vytváří nejasnosti ohledně konkrétních omezení a povolení. Zásadními faktory, které je třeba vzít v úvahu, jsou také kontext a účel web scrapingu. Zatímco některé dokumenty TOS mohou stanovit výjimky pro konkrétní účely, jako jsou internetové vyhledávače nebo nekomerční veřejné archivy (např. archive.org), většina ustanovení nerozlišuje mezi různými záměry scrapování, jako je akademický výzkum nebo komerční využití. Pro orientaci v etickém a právním prostředí web scrapingu je nezbytné, aby si uživatelé pečlivě prostudovali podmínky používání webových stránek, které hodlají scrapovat, a v případě potřeby si vyžádali písemný souhlas. To může pomoci předejít možným právním problémům a zajistit, aby činnosti spojené se scrapingem webu byly prováděny zodpovědně a eticky (FIESLER, 2021).

1.4.1 Robot.txt

Před procházením webu by měl každý robot nejprve zkontrolovat soubor robots.txt a zjistit, zda přístup není omezen. Soubor robots.txt musí být napsán malými písmeny a umístěn v kořenovém adresáři webu (např. hned za koncovkou .cz/ nebo .com/). Například na naší webové stránce má soubor robots.txt adresu http://www.bezrealitky.cz/robots.txt. Každý řádek souboru robots.txt určuje agenta uživatele (bota vyhledávače) a zakázané přístupové cesty (Disallow). Nastavením pravidel v souboru robots.txt mohou webmasteři řídit chování robotů vyhledávačů a chránit citlivé oblasti svých webových stránek

(FIESLER, 2021). Zde je ukázka souboru, pokud by webová stránka zakazovala scrapování webu.

Výpis 1.1 Ukázka rebots.txt file (vlastní zpracování)

```
1         User-agent: *
2         Disallow: /
```

2 Dobývání znalostí z databází

V této části teoretické práce se podrobně seznámíme s procesem dobývání znalostí z databází (KDD), který představuje základní kámen datové analýzy a těžby dat. Nejprve si vysvětlíme, co je to KDD a jaký je jeho význam v současném datově orientovaném světě. Následně se zaměříme na různé metodiky, jako jsou SEMMA, 5A a CRISP-DM, které nám pomohou vytvořit systematický přístup k KDD procesu.

Poté se přesuneme k různým metodám používaným v KDD, jako jsou prediktivní a deskriptivní metody, které nám umožní objevovat a analyzovat vzory v našich datech. Nakonec se seznámíme s několika klíčovými nástroji, které nám pomohou s aplikací KDD v praxi, jako jsou Pandas, Numpy, DataFrame a Scikit-learn. Tyto nástroje nám poskytnou efektivní způsob, jak manipulovat s daty, modelovat a vyhodnocovat naše analýzy.

2.1 Úvod do KDD

Pojem dobývání znalostí z databází (KDD, Knowledge Discovery in Databases) definoval Usama Fayyad jako proces netriviálního objevování implicitních, dopředu neznámých a potenciálně použitelných znalostí v datech (Fayyad a kol., 1996). Podle této definice někteří autoři považují termíny dolování znalostí z databází (KDD) a data mining za synonyma, zatímco někteří považují data mining pouze za jednu z nezbytných fází iterativního procesu objevování znalostí. S postupem technologického vývoje v oblasti ukládání a výpočetní techniky, spolu s exponenciálním růstem objemu dat, se dolování dat rozvinulo v široce rozšířený obor (HAN, 2012).

V procesu KDD se na rozdíl od jednoduchého použití statistických metod a metod strojového učení, klade velký důraz na přípravu a analýzu dat pro interpretaci. Původně byl zaveden technologický pohled na proces dobývání znalostí, avšak v současné době je uplatňován ve většině situací tzv. manažerský proces těžby znalostí (MRÁZOVÁ, 2014).

1. Příprava dat
    a. Selekce – z databáze vyhledají data relevantní pro analytickou úlohu.
    b. Předzpracování – eliminace šumu a nesrovnalostí v datech a sloučení dat z více zdrojů.
    c. Transformace – data jsou transformována a konsolidována do podoby vhodné pro vytěžování provedením souhrnných nebo agregačních operací
2. Data mining – základní proces, při němž se používají inteligentní metody k získání datových vzorů.
3. Interpretace dat
    a. Evaluace modelu – identifikace skutečně zajímavých vzorů představujících znalosti na základě měr zajímavosti

b. Prezentace znalostí – prezentace vytěžených znalostí uživatelům kdy se používají techniky vizualizace a reprezentace znalostí

2.2 Metodiky

Metodologie mají za úkol nabídnout uživatelům sjednocený rámec pro řešení různorodých úkolů v oblasti dobývání znalostí z databází. Díky těmto metodologiím je možné sdílet a předávat zkušenosti z jiných úspěšných projektů. Mezi nejrozšířenější metodologie patří CRISP-DM, SEMMA (SAS) a 5A (SPSS) (DANEL, 2014).

2.2.1 SEMMA

SEMMA, zkratka pro Sample, Explore, Modify, Model, and Assess, představuje metodiku používanou při realizaci projektů dobývání znalostí z databází (KDD). SAS Institute uvádí pětistupňový cyklus tohoto přístupu:

1. Sample (Vzorek) - zahrnuje získání podmnožiny dat, která je dostatečně velká, aby zachovala podstatné informace, a zároveň dostatečně malá pro efektivní manipulaci.
2. Explore (Prozkoumání) - zahrnuje zkoumání dat s cílem zjistit neočekávané trendy a anomálie, aby se získaly poznatky.
3. Modify (Modifikování) - zaměřuje se na změnu dat vytvářením a transformací proměnných s cílem zefektivnit proces výběru modelu.
4. Model (Modelování) - software automaticky vyhledává kombinaci dat, která spolehlivě předpovídá požadovaný výsledek.
5. Assess (Vyhodnocení) - vyhodnocuje užitečnost a spolehlivost zjištění z modelu a odhaduje jeho výkonnost.

Metoda SEMMA poskytuje srozumitelný rámec, který umožňuje dobře organizovaný vývoj a údržbu projektů KDD. Následně nabízí strukturu pro koncepci, tvorbu a vývoj projektů, pomáhá při řešení obchodních problémů a určování obchodních cílů (AZEVEDO, 2014).

2.2.2 5A

5A, zkratka pro Assess, Access, Analyze, Act, and Automate, představuje jednu z dalších používaných metodik při realizaci projektů dobývání znalostí z databází nabízený firmou SPSS (DANEL, 2014):

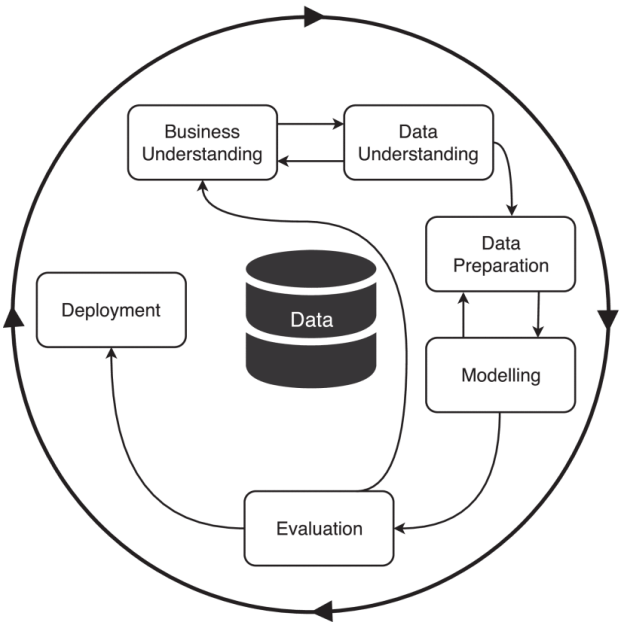
1. Assess – posouzení potřeb projektu
2. Access – shromáždění potřebných dat
3. Analyze – provedení analýz
4. Act – přeměna znalostí na akční znalosti
5. Automate – převedení výsledků analýzy do praxe

2.2.3 CRISP-DM

Cross-Industry Standard Process for Data mining (CRISP-DM) je metodika pokrývající kompletní proces data miningových úloh (Rauch, Šimůnek 2014, str. 19). Metodika byla vyvinuta v rámci výzkumného projektu Evropské komise. Projekt si klade za cíl navrhnout univerzální postup (tzv. standardní model procesu dobývání znalostí z databází), který bude uplatnitelný v různých komerčních aplikacích. Tato metodologie umožňuje rychlejší, efektivnější, spolehlivější a nákladově efektivnější řešení rozsáhlých úloh KDD. CRISP-DM má navíc poskytnout "průvodce" možnými problémy a řešeními, které se mohou objevit v reálných aplikacích. Projekt spolupracuje s firmami NCR (přední dodavatel datových skladů), DaimlerChrysler, ISL (tvůrce systému Clementine) a OHRA (velká holandská pojišťovna). Všechny tyto firmy mají bohaté zkušenosti s reálnými úlohami těžby znalostí z databází (PLCHÚT, 2014).

Základní etapy procesu dobývání jsou:

1. Co řešit – Business understanding
2. Kde vzít data – Data understanding
3. Jak data připravit – Data preparation
4. Jak data analyzovat – Modelling
5. Co jsme zjistili – Evaluation
6. Jak výsledky využít – Deployment



Obr. 2.1 CRISP-DM (MARTINEZ-PLUMED, 2021)

Business understanding

Pro úspěšnou realizaci projektu je důležité pochopit jeho doménovou oblast, cíle a požadavky. Je potřeba identifikovat obchodní cíle, pochopit požadované výsledky zákazníka z obchodního hlediska a stanovit kritéria úspěchu. Poté zhodnotit situaci zjištěním dostupnosti zdrojů, pochopením požadavků na projekt a posouzením rizik a nepředvídaných událostí. Dále je třeba provést analýzu nákladů a přínosů, abychom zajistili proveditelnost projektu.

Dále stanovíme cíle dolování dat a nastíníme, jak by měly vypadat úspěšné výsledky z technického hlediska. Na základě těchto informací vypracujeme komplexní plán projektu, který zahrnuje výběr vhodných technologií a nástrojů a vytvoření podrobných plánů pro jednotlivé fáze projektu, aby byla zajištěna jeho hladká realizace (HOLUBOVÁ, 2016).

Data understanding

Pochopení jednotlivých atributů a jejich významu pro analýzu je zásadní pro správnou přípravu dat a modelování (Hair et al., 2010). Například poloha nemovitosti může mít významný vliv na její cenu a poptávku, protože nemovitosti v žádanějších lokalitách mají zpravidla vyšší ceny. Podobně mohou hodnotu nemovitosti ovlivnit atributy jako počet místností, typ budovy a její stav (Clapp & Giaccotto, 1992). Data understanding můžeme rozčlenit do několika bodů:

1. Shromáždění počátečních údajů – Získání potřebných dat a jejich načtení do analytického nástroje.
2. Popis dat – Prozkoumání dat a zdokumentování jejich povrchových vlastností jako je formát dat, počet záznamů a identita polí.
3. Prozkoumání dat – Prozkoumání dat hlouběji. Dotazování se na data a identifikace vztahů mezi daty.
4. Ověření kvality dat – Jak čistá, nebo špinavá jsou data. Zdokumentování případných problémů s kvalitou dat (HOTZ, 2018).

Data preparation

Příprava dat hraje klíčovou roli v procesu vývoje statistických modelů. Navzdory svému významu je příprava dat často přehlížena, přestože zpracování špinavých nebo nekonzistentních dat je pro analytiky velkým problémem. Pro řešení problémů spojených s předzpracováním a analýzou dat ve vysoko dimenzionálních statistických modelech je nezbytné vytvořit účinné kanály, které si poradí s problémy, jako je extrakce struktury, imputace chybějících hodnot a zpracování nesprávných dat (KRISHNAN, 2016).

Tradiční přístupy k přípravě dat, jako jsou přístupy založené na integritních omezeních, statistice nebo strojovém učení, mohou být účinné, ale kvůli svým vnitřním omezením nemohou zaručit přesnost opravených dat. Pro zvýšení přesnosti metod čištění dat je nezbytné využívat externí informace, jako jsou znalost datasetu a lidské odborné znalosti, které lze získat od expertů v dané oblasti (CHU, 2015).

Tento segment procesu se dá rozčlenit do několika úkolů, přičemž přesná kategorizace jednotlivých činností nemusí být striktně dodržována. Například odstraňování redundance dat může být součástí čištění dat i redukce dat. Mezi úkoly patří:

1. Čištění dat – zahrnuje doplňování chybějících hodnot, identifikaci a odstranění odlehlých hodnot a nekonzistencí. Pokud uživatelé nedůvěřují datům, ze kterých vycházíme, nebudou důvěryhodné ani dosažené výsledky.
2. Integrace dat – data mohou pocházet z různých zdrojů, jako jsou databáze, datové kostky nebo soubory. Atributy mohou mít v různých databázích odlišná jména, i když označují stejnou vlastnost, což způsobuje redundanci a nekonzistenci dat. Je tedy nezbytná data integrovat do akceptovatelné podoby.
3. Transformace dat – některé metody preferují transformovaná data. Transformace zahrnuje například normalizaci, při níž se původní rozsah hodnot převede na vhodnější, např. [0.0, 0.1]. Další operací je agregace, která se často provádí před ukládáním dat do datového skladu.
4. Redukce dat – často nepotřebujeme všechna dostupná data. Celý proces lze urychlit vyjmutím nepotřebných a nerelevantních dat při zachování vlastností původního souboru hodnot. Běžně se používá agregace, redukce dimenzionality a snížení počtu hodnot.
5. Diskretizace dat – tato forma redukce dat spočívá v nahrazení původních numerických dat kategorickým atributem, čímž se výrazně snižuje objem dat při uspokojivém zachování původní informace (HAN, 2012).

Modelling

Během této fáze se vytváří a vyhodnocují různé modely s využitím několika modelovacích technik. Tato fáze se skládá ze čtyř úkolů:

1. Výběr technik modelování – Rozhodnutí, které algoritmy se budou používat (např. regrese, neuronová síť).
2. Vypracování návrhu testů – V závislosti na přístupu k modelování je nutné rozdělení dat na trénovací, testovací a validační množiny.
3. Sestavení modelů – Sestavení modelů pomocí provedení několika řádků kódu typu "reg = LinearRegression().fit(X, y)".
4. Vyhodnocení modelu: Obvykle mezi sebou soutěží několik modelů a datový vědec musí interpretovat výsledky modelu na základě odborných znalostí v dané oblasti, předem stanovených kritérií úspěšnosti a návrhu testu.

Ačkoli příručka CRISP-DM doporučuje iterovat tvorbu a vyhodnocování modelů, dokud není nalezen nejlepší model(y), v praxi týmy iterují, dokud neobjeví "dostatečně dobrý" model, pokračují v životním cyklu CRISP-DM a poté model v dalších iteracích vylepšují (HOTZ, 2018).

Evaluation

V této fázi dochází k vyhodnocení získaných znalostí, které jsou relevantní z pohledu data miningových metod. Zkoumá se širší pohled na to, který model nejlépe vyhovuje potřebám. Součástí vyhodnocení jsou:

1. Vyhodnocení výsledků – Kontrola, zda model splnil kritéria, výběr modelů, které budou schváleny.
2. Proces přezkoumání – Přezkoumání, zda nebylo něco přehlédnuto a zda byly kroky provedeny schválně.
3. Určení dalších kroků – Na základě předchozích kroků přechod k nasazení, nebo iteraci (HOLUBOVÁ, 2016).

Deployment

Složitost této závěrečné fáze se může značně lišit a zahrnuje čtyři úkony:

1. Plánování nasazení: Vytvoření a zdokumentování plánu implementace modelu.
2. Plánování monitorování a údržby: Vypracujte komplexní plán monitorování a údržby, abyste předešli problémům během provozní fáze modelu (nebo po skončení projektu).
3. Vytvoření závěrečné zprávy: Projektový tým připraví shrnutí projektu, které může zahrnovat prezentaci výsledků dolování dat.
4. Přezkoumání projektu: Provedení retrospektivní analýzy projektu, diskuse o tom, co se povedlo, jaké oblasti je třeba zlepšit a jak vylepšit budoucí projekty.

CRISP-DM jako projektový rámec neposkytuje pokyny pro činnosti po ukončení projektu (známé také jako "provoz"). Pokud však model přechází do výroby, zajistěte jeho řádnou údržbu. Často je nutné průběžné monitorování a občasné ladění modelu (HOTZ, 2018).

2.3 Metody

Dnes užívanými metodami dolování dat jsou:

1. Prediktivní metody
2. Deskriptivní metody

2.3.1 Prediktivní metody

Prediktivní metody představují předpověď hodnot dat na základě známých výsledků zjištěných z různých údajů. Prediktivní modelování může být provedeno na základě použití variantních historických dat. Úlohy prediktivního modelu pro dolování dat zahrnují regresi, analýzu časových řad, klasifikaci, predikci. Jedná se o techniku monitorování učení, která zahrnuje vysvětlení závislosti několika hodnot atributů na hodnotách jiných atributů V podobném předmětu a růst modelu, který může předpovídat tyto hodnoty atributů pro nedávné případy (DANEL, 2014).

Regrese

Regresní analýza je základní technikou pro řešení regresních problémů ve strojovém učení pomocí modelování dat. Tato technika spočívá v nalezení optimální přímky, která prochází všemi datovými body a minimalizuje vzdálenost mezi přímkou a každým bodem. Existují různé techniky regresní analýzy, přičemž výběr jednotlivých technik závisí na několika faktorech. Tyto faktory zahrnují povahu cílové proměnné, tvar regresní přímky a množství nezávislých proměnných. Regrese se od ostatních klasifikačních metod liší především typem výsledku. Výsledkem regresní metody je spojitá číselná hodnota, zatímco výsledkem klasifikační metody je kategorie.

Lineární regrese patří mezi nejzákladnější typy regrese ve strojovém učení. Zahrnuje predikovanou proměnnou a závislou proměnnou, které jsou lineárně propojeny. Lineární regresní model představuje následující rovnice ($y = mx + c + e$). Zde (m) představuje sklon přímky, (c) je intercept a (e) označuje chybu modelu. Optimální fit linie se určí úpravou hodnot (m) a (c). Chyba predikce je rozdíl mezi pozorovanými hodnotami a predikovanou hodnotou. Hodnoty (m) a (c) se volí tak, aby se minimalizovala chyba předpovědi. Je nezbytné si uvědomit, že základní lineární regresní model je náchylný k odlehlým hodnotám, takže nemusí být vhodný pro velké soubory dat (VADAPALLI, 2022).

Pokud data vykazují multikolinearitu, tj. pokud jsou nezávislé proměnné vysoce korelované, použije se technika ridge regrese. Odhady metodou nejmenších čtverců jsou sice při multikolinearitě nestranné, ale jejich rozptyly jsou dostatečně významné na to, aby způsobily odchylku pozorované hodnoty od skutečné hodnoty. Ridge regrese snižuje standardní chyby zkreslením regresních odhadů. Proměnná lambda (λ) v rovnici regrese řeší problém multikolinearity.

Stejně jako u ridge regrese technika lasso (Least Absolute Shrinkage and Selection Operator) penalizuje absolutní velikost regresního koeficientu. Navíc technika lasso regrese využívá výběr proměnných, který vede ke smrštění hodnot koeficientů na absolutní nulu (SHARMA, 2022).

Klasifikace

Klasifikace je proces hledání modelu (nebo funkce), který popisuje a rozlišuje třídy nebo koncepty dat. Modely jsou odvozeny na základě analýzy souboru trénovacích dat (tj. datových objektů, pro které jsou známy značky tříd). Model se používá k předpovídání označení tříd objektů, u nichž není známo označení třídy.

Odvozený model může být reprezentován v různých formách, například jako klasifikační pravidla (tj. pravidla IF-THEN), rozhodovací stromy, matematické vzorce nebo neuronové sítě. Rozhodovací strom je stromová struktura podobná diagramu, kde každý uzel označuje test na hodnotu atributu, každá větev představuje výsledek testu a listy stromu představují třídy nebo rozdělení tříd (HAN, 2012).

2.3.2 Deskriptivní metody

Deskriptivní model rozlišuje vztahy nebo vzorce v datech. Na rozdíl od prediktivního modelu slouží deskriptivní model ke zkoumání vlastností zkoumaných dat, nikoliv k předpovídání nových vlastností, shlukování, sumarizace, přiřazování pravidel a objevování sekvencí jsou úlohy dolování dat pomocí deskriptivního modelu. Deskriptivní analytika Soustřeďte se na sumarizaci a převod dat na významné informace pro monitorování a reporting (DANEL, 2014).

Shlukování

Shlukování je úkol rozdělující neoznačená data nebo datové body do různých shluků tak, aby podobné datové body spadaly do stejného shluku než ty, které se od ostatních liší. Zjednodušeně řečeno, cílem procesu shlukování je oddělit skupiny s podobnými znaky a přiřadit je do shluků.

Hierarchický clustering si s velkými daty neporadí dobře, ale K Means ano. Je to proto, že časová složitost K Means je lineární, tedy O(n), zatímco hierarchického je kvadratická, tedy O(n2). Protože začínáme s náhodným výběrem shluků, mohou se výsledky získané vícenásobným spuštěním algoritmu při shlukování K Means lišit. Zatímco v případě hierarchického shlukování jsou výsledky reprodukovatelné. Zjistilo se, že metoda K Means funguje dobře, když je tvar shluků hypersférický (jako kruh ve 2D nebo koule ve 3D). Shlukování K Means vyžaduje předchozí znalost K, tj. počtu shluků, do kterých chcete data rozdělit. Při hierarchickém shlukování se však můžete zastavit na jakémkoli počtu shluků, který považujete za vhodný, a to interpretací dendrogramu (KAUSHIK, 2016).

Detekce anomálií

V analýze dat je detekce anomálií (označovaná také jako detekce odlehlých hodnot) obecně chápána jako identifikace vzácných položek, událostí nebo pozorování, které se výrazně odchylují od většiny dat a neodpovídají přesně definovanému pojetí normálního chování. Takové příklady mohou vzbuzovat podezření, že byly vytvořeny jiným mechanismem, nebo se jeví jako nekonzistentní se zbytkem daného souboru dat (CHANDOLA, 2009).

2.4 Použité nástroje

V této části si popíšeme, jaké nástroje jsme požívali v praktické části. Popíšeme knihovnu pythonu Pandas a její součást PandasDataFrame. Dále knihovnu Numpy a Scikit-learn.

2.4.1 Pandas

V této části si popíšeme, co to je příprava dat. Poté si popíšeme knihovnu pythonu Pandas a jeí součást PandasDataFrame. Pro další ilustraci probíraných konceptů si následně v praktické části předvedeme implementaci přípravy dat na předem vyscrapovaných datech.

Pandas, open-source knihovna jazyka Python, se stala základním zdrojem pro analýzu dat mezi profesionály používajícími Python. Knihovna pandas, kterou vyvinul především Wes McKinney a později s pomocí Siena Changa, byla vytvořena s cílem řešit potřebu jednoduché, ale výkonné knihovny pro zpracování, extrakci a manipulaci s daty. Knihovna pandas je postavena na základech knihovny NumPy, je kompatibilní s většinou ostatních modulů a využívá výhod vysoce výkonných výpočetních schopností NumPy. Knihovna také zavádí vlastní datové struktury určené pro práci s relačními nebo označenými daty, což z ní činí ideální volbu pro správu dat podobnou relačním databázím SQL nebo tabulkám Excelu (NELLI, 2015).

2.4.2 Numpy

NumPy (z anglického Numerical Python) slouží jako základní knihovna pro vědecké výpočty v jazyce Python. Tento balíček jazyka Python nabízí objekt vícerozměrného pole spolu s řadou souvisejících objektů (jako jsou maskovaná pole a matice) a kolekcí funkcí, které umožňují rychlé operace s poli. Tyto operace zahrnují matematiku, logiku, změny tvaru, třídění, výběr, vstupy a výstupy, diskrétní Fourierovy transformace, elementární lineární algebru, základní statistické postupy, náhodné simulace a spoustu dalších funkcí (IDRIS, 2013).

2.4.3 DataFrame

DataFrame, základní součást knihovny pandas, je tabulková datová struktura připomínající tabulky Excelu. Je navržena tak, aby rozšířila možnosti datové struktury Series na více dimenzí. DataFrame je v podstatě uspořádaná kolekce sloupců, přičemž každý sloupec může obsahovat hodnoty různých typů, například číselné, řetězcové nebo logické. Na rozdíl od struktury Series, která obsahuje jediné indexové pole obsahující popisky pro každý prvek, DataFrame využívá dvě indexová pole (NELLI, 2015).

2.4.4 Scikit-learn

Scikit-learn je široce používaná open-source knihovna jazyka Python pro aplikace strojového učení a datové vědy. Knihovna scikit-learn byla vyvinuta jako součást širšího ekosystému SciPy a poskytuje rozsáhlou škálu nástrojů pro předzpracování dat, extrakci příznaků, redukci dimenzionality a vyhodnocování modelů. Knihovna nabízí rozsáhlý výběr algoritmů učení pod dohledem i bez dohledu, včetně metod klasifikace, regrese, shlukování a detekce anomálií.

Scikit-learn je známý svým uživatelsky přívětivým rozhraním API, rozsáhlou dokumentací a kompatibilitou s dalšími vědeckými knihovnami jazyka Python, jako jsou NumPy, Pandas a Matplotlib. Jeho konzistentní principy návrhu usnadňují uživatelům přepínání mezi různými modely a technikami, což podporuje experimentování a umožňuje rychlý vývoj robustních pipeline strojového učení.

3 Implementace Selenium scraperu

Scraper se nachází ve třídě WebScraper a slouží ke scrapování dat z webového portálu www.bezrealitky.cz. V konstruktoru přijímá ovladač (driver) jako parametr, což umožňuje použití různých ovladačů, jako je Chrome, Firefox nebo Edge. Dále inicializuje instanci WebDriverWait, která slouží k čekání na načítání dynamických prvků na webových stránkách. Konstruktor také nastavuje proměnnou sleep_time, která určuje dobu pauzy mezi jednotlivými akcemi v procesu scrapingu.

3.1 Logging

Logging (česky protokolování) pomáhá identifikovat problémy a odstraňovat chyby, které mohou vzniknout během průběhu aplikace. V kontextu scraperu Selenium se nastavuje vlastní konfigurace protokolování.

Funkce configure_logging() slouží jako vstupní bod pro nastavení konfigurace logů. Tato funkce volá funkci setup_logging_configuration() pro vytvoření souboru protokolu a nastavení formátu protokolování. Kromě toho volá funkci silence_third_party_loggers(), která potlačuje výstup logů třetích stran a zajišťuje, že se zobrazují pouze relevantní zprávy ze scraperu.

Funkce setup_logging_configuration() vytvoří soubor pro ukládání logů s názvem "webscraper.log". Úroveň logů je nastavena na logging.INFO, což znamená, že se budou zaznamenávat pouze logy s úrovní INFO nebo vyšší. Formát záznamu je nastaven tak, aby obsahoval časovou hodnotu, úroveň záznamu a zprávu. Funkce také volá funkci add_console_handler(), která umožňuje zobrazení logů v konzoli během průběhu scrapování.

Funkce add_console_handler() vytvoří a nakonfiguruje zobrazení logů v konzoli. Tím je zajištěno, že se logy zobrazují v konzoli v reálném čase, což uživatelům umožňuje sledovat průběh aplikace.

A konečně funkce silence_third_party_loggers() potlačuje výstup logů z knihoven třetích stran, jako jsou Selenium a urllib3. Nastaví jejich úroveň logů na WARNING a tím zabrání tomu, aby jejich logy zahlcovaly výstup, a usnadní se tak soustředění na relevantní logy generované scraperem.

Výpis 3.2 Konfigurace logů (vlastní zpracování)

```
def configure_logging():
    # Set up the main logging configuration
    setup_logging_configuration()

    # Silence third-party logger output
```

```
    silence_third_party_loggers()

def setup_logging_configuration():
    # Create a log file and set logging format
    log_directory = os.path.dirname(os.path.abspath(__file__))
    log_file_path = os.path.join(log_directory, "webscraper.log")
    logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s -
%(message)s',
                        filename=log_file_path, filemode='a')

    # Add console handler for displaying logs in the console
    add_console_handler()

def add_console_handler():
    # Create and configure a console handler for logging
    console_handler = logging.StreamHandler()
    console_handler.setLevel(logging.INFO)
    formatter = logging.Formatter('%(asctime)s - %(levelname)s - %(message)s')
    console_handler.setFormatter(formatter)
    logging.getLogger().addHandler(console_handler)

def silence_third_party_loggers():
    # Set log levels for third-party loggers to suppress their output
    LOGGER.setLevel(logging.WARNING)
    logging.getLogger("selenium").setLevel(logging.WARNING)
    logging.getLogger("urllib3").setLevel(logging.WARNING)
```

3.2 Driver

Metoda init_driver vytváří novou instanci webdriveru. Statická metoda configure_chrome_options je zodpovědná za nastavení ovladače Chrome. Tato konfigurace zahrnuje vypnutí zobrazování obrázků, použití režimu inkognito, vypnutí upozornění, vypnutí rozšíření a další. Také nastavuje agenta prohlížeče na konkrétní hodnotu, která simuluje běžného uživatele.

V kódu je také zakomponovaná možnost --headless, která umožňuje spouštět ovladač bez zobrazování grafického rozhraní prohlížeče. Tuto možnost lze zakomentovat, pokud chceme rozhraní zobrazit.

Výpis 3.3 HeadLess chrome option

```
chrome_options.add_argument("--headless")
```

3.3 Cookies

Webové stránky často používají cookies k ukládání informací o návštěvnících a jejich preferencích, což vyžaduje jejich souhlas. Tento souhlas se obvykle vyžaduje prostřednictvím zobrazení banneru s tlačítkem pro přijetí cookies.

Metoda accept_cookies se nejprve pokusí najít tlačítko pro přijetí cookies pomocí WebDriverWait a XPath. WebDriverWait zajišťuje, že kód čeká na načtení tlačítka, než se pokusí s ním interagovat. Pokud se tlačítko na stránce objeví a je kliknutelné, metoda na něj klikne, čímž přijme cookies.

Pokud tlačítko pro přijetí cookies není na stránce nalezeno nebo pokud jeho načítání trvá příliš dlouho, metoda vyvolá výjimku TimeoutException. V tomto případě je do logu zaznamenáno varování, které informuje o problému s nalezením tlačítka nebo s jeho načítáním.

Výpis 3.4 Cookies (vlastní zpracování)

```
    def accept_cookies(self):
        """
        Accept cookies on the website if the cookies banner is present.
        """
        try:
            # Find the 'accept cookies' button and click it
            accept_button = self.wait.until(EC.element_to_be_clickable((By.XPATH,
"//button[@id='CybotCookiebotDialogBodyLevelButtonLevelOptinAllowAll']")))
            accept_button.click()
        except TimeoutException:
            logging.warning("Could not find the accept cookies button or it took too long
to load.")
```

3.4 Procházení URL

Metoda scrape_listings slouží k procházení URL adres a získávání informací z jednotlivých inzerátů. Tato metoda přijímá argument url, který je hlavní URL adresa, ze které se začne scrapovat. Metoda nejprve navštíví hlavní URL a přijme cookies pomocí metody accept_cookies. Poté začne procházet jednotlivé stránky s inzeráty.

Metoda používá smyčku while pro procházení stránek s inzeráty, dokud nenarazí na poslední stránku nebo dokud nedosáhne maximálního počtu stránek k procházení. Na každé stránce metoda najde odkazy na inzeráty pomocí WebDriverWait a XPath, poté získá URL adresy jednotlivých inzerátů. Pro každou získanou URL adresu pak metoda extrahuje data pomocí metody extract_info.

Po zpracování všech inzerátů na stránce metoda pokračuje na další stránku kliknutím na tlačítko "Další". Pokud není další stránka nalezena, metoda ukončí procházení stránek

a vypíše zprávu o dokončení scrapování. Na konci metody je WebDriver uzavřen a metoda vrátí získaná data z inzerátů.

3.5 Extrakce dat

Metoda extract_info zajišťuje extrakci požadovaných informací z dané URL adresy. Tato metoda volá další metody pro extrakci různých typů dat. Hlavní metoda extract_data organizuje získaná data a slučuje je do jednoho slovníku, který obsahuje základní údaje, údaje z tabulky a sezbíraná data.

Metody extract_basic_data, extract_extra_data, extract_table_data a extract_poi_data jsou zodpovědné za extrakci specifických typů dat z webových stránek. Každá z těchto metod vrací slovník se získanými daty, který je následně aktualizován do hlavního slovníku v metodě extract_data. Tyto metody používají různé strategie pro extrakci dat, jako je využití XPath nebo CSS selektorů.

Konečně, metoda try_extract_element slouží k získání textu prvku z webové stránky pomocí zadaného lokátoru. Pokud prvek není nalezen, metoda vrací None. Tato metoda je často používána v ostatních metodách pro extrakci dat, což zvyšuje efektivitu a zjednodušuje práci s kódem.

Díky modularitě kódu a jeho organizaci do několika metod je tento přístup efektivní a snadno rozšiřitelný pro další potřeby.

3.6 Uložení do csv

Metoda save_to_csv zajišťuje uložení získaných dat do CSV souboru s předem definovaným schématem. Nejprve se data převedou do pandas DataFrame a přidá se sloupec 'Index' s řadou čísel od 1 do délky DataFrame. Následně se pomocí slovníku column_mapping přejmenují sloupce, aby byly konzistentní se schématem. Poté se definuje pevné schéma s požadovanými sloupci a vytvoří se nový prázdný DataFrame s tímto schématem.

Před spojením dvou DataFramů (jeden s extrahovanými daty a druhý s pevným schématem) se resetují indexy obou DataFramů. Pokud se vyskytnou duplicity ve jménech sloupců, přidají se příslušné přípony pro jejich rozlišení. Poté se oba DataFramy spojí, chybějící hodnoty se vyplní hodnotou 'NaN' a získaná data se uloží do CSV souboru.

Pro uložení dat do CSV souboru se používá Python modul csv. S jeho pomocí se vytvoří objekt csv.DictWriter, který zapisuje data do souboru podle zadaných sloupců. Pokud soubor neexistuje, nejprve se zapíše hlavička se jmény sloupců, poté se zapisují jednotlivé řádky dat.

Díky tomuto přístupu je možné snadno uchovávat a organizovat získaná data pro následné čištění a zpracování analýzy.

4 Implementace dobývání znalostí

V této části se budeme zabývat procesem analýzy dat o nemovitostech se zaměřením na pochopení domény, dat, přípravu dat, modelování a vyhodnocení výsledků. Analýza bude provedena na předem vyscrapovaném souboru dat se specifickými atributy.

4.1 Porozumění doménové oblasti

Nemovitosti jsou složitou doménou a pro úspěšnou analýzu dat je nezbytné porozumět složitostem trhu (McGraw-Hill, 2012). Realitní trhy jsou ovlivňovány různými faktory, jako jsou ekonomické podmínky, demografické trendy, vládní politika a dynamika místního trhu. Tyto faktory mohou mít významný vliv na ceny nemovitostí, poptávku a nabídku, a proto je pochopení jejich vzájemného působení klíčové pro přesnou analýzu a předpověď (Bourassa et al., 2003).

Různé studie se pokoušely modelovat a předpovídat ceny nemovitostí pomocí různých technik, jako je vícenásobná regrese, strojové učení a prostorová analýza (Pagourtzi et al., 2003). Pochopení základních faktorů ovlivňujících ceny nemovitostí je klíčové pro vývoj přesných modelů a interpretaci výsledků (Worzala et al., 1995).

4.2 Porozumění datům

Poskytnutý soubor dat obsahuje informace o jednotlivých nemovitostech s následujícími atributy:

- Index, URL, ČÍSLO INZERÁTU (informační hodnoty).
- CENA (cena)
- TYP NABÍDKY (koupě/pronájem)
- LOKACE (Kraj/mesto)
- DISPOZICE (kolik pokojů)
- STAV (podmínky)
- DOSTUPNÉ OD (datum, kdy lze koupit)
- VLASTNICTVÍ (osobní/družstevní)
- TYP BUDOVY (cihla, panel, ostatní)
- PLOCHA (krajina)
- VYBAVENO (ano/ne)
- PODLAŽÍ (ve kterém patře)
- PENBR
- Internet (ano/ne)
- Energie (ano/ne)

4.2.1 Shromáždění počátečních údajů

Shromáždění počátečních údajů začíná importem základních knihoven pro manipulaci s daty a vizualizaci, jako jsou Pandas, NumPy, Matplotlib a Seaborn. Poté načte data ze zadané cesty k souboru a vytvoří Pandas DataFrame (df), který nabízí řadu funkcí pro efektivní manipulaci s daty a jejich analýzu.

Tato část kódu poskytuje přehled o struktuře datové sady, například počet řádků a sloupců, několik prvních řádků a shrnutí datové sady. Souhrn (vytvořený pomocí df.info()) obsahuje datový typ každého sloupce, počet nenulových hodnot a využití paměti. Tyto informace jsou klíčové pro pochopení složení datové sady a identifikaci potenciálních problémů s datovými typy nebo chybějícími hodnotami.

```
df = pd.read_csv(file_path, encoding='utf-8', index_col=None)

# Shape of the dataset
print("Shape of the dataset:", df.shape)

# Display the first 5 rows of the dataset
print("\nFirst 5 rows of the dataset:")
display(df.head())

# Get the summary of the dataset (column data types, non-null values, memory usage)
print("\nSummary of the dataset:")
display(df.info())

Output:
Summary of the dataset:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5966 entries, 0 to 5965
Data columns (total 39 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Index                 5966 non-null   int64
 1   URL                   5966 non-null   object
 2   CENA                  5963 non-null   object
 3   POPLATKY ZA SLUŽBY    2117 non-null   object
 4   POPLATKY ZA ENERGII   0 non-null      float64
 5   VRATNÁ KAUCE          2832 non-null   object
 6   TYP NABÍDKY           5964 non-null   object
 7   LOKACE                5964 non-null   object
 8   ČÍSLO INZERÁTU        5211 non-null   float64
 9   DISPOZICE             5210 non-null   object
 10  STAV                  5051 non-null   object
 11  DOSTUPNÉ OD           2658 non-null   object
 12  VLASTNICTVÍ           4171 non-null   object
```

```
 13  TYP BUDOVY            5197 non-null   object
 14  PLOCHA                5210 non-null   object
 15  VYBAVENO              4591 non-null   object
 16  PODLAŽÍ               4929 non-null   float64
 17  PENB                  3029 non-null   object
...
 37  Hřiště                5719 non-null   object
 38  POPLATKY ZA ENERGIE   1063 non-null   object
dtypes: float64(13), int64(1), object(25)
memory usage: 1.8+ MB
```

Datová sada obsahuje 5 966 záznamů a 39 sloupců. Sloupec Non-Null Count (Počet nenulových hodnot) udává počet nenulových hodnot pro každý prvek. Například sloupec "CENA" má 5 963 nenulových hodnot, což znamená, že v tomto sloupci chybí tři hodnoty (protože celkový počet položek je 5 966). Naproti tomu sloupec "POPLATKY ZA ENERGIE" nemá žádné nenulové hodnoty (0 nenulových hodnot), což znamená, že všechny hodnoty v tomto sloupci chybí. Zde se pravděpodobně vyskytla chyba při scrapování dat. Sloupec Dtype představuje datový typ každého prvku. V této datové sadě jsou tři hlavní datové typy:

- int64: Představuje celočíselné hodnoty, například sloupec "Index".
- float64: Reprezentuje hodnoty s pohyblivou desetinnou čárkou, například sloupce "ČÍSLO INZERÁTU" a "PODLAŽÍ".
- Objekt: Reprezentuje nečíselné datové typy, obvykle řetězce nebo smíšené typy. V této datové sadě má většina sloupců datový typ "objekt", například "URL", "CENA", "TYP NABÍDKY", "LOKACE" a "DISPOZICE".

Paměťová náročnost datové sady je přibližně 1,8 MB.

4.2.2 Popis dat

Při popisu dat kód vypočítá a zobrazí různé popisné statistiky pro číselné sloupce (počet, průměr, směrodatnou odchylku, minimum, kvartily a maximum) pomocí df.describe(). To vám pomůže pochopit rozložení, centrální tendenci a rozptyl hodnot v rámci každého sloupce. Dále kód zobrazuje počet unikátních hodnot v každém sloupci pomocí df.nunique(), což poskytuje přehled o rozmanitosti datového souboru a pomáhá identifikovat kategoriální proměnné.

```
# Descriptive statistics of the dataset (only for numerical columns)
print("Descriptive statistics of the dataset:")
display(df.describe())

# Get the number of unique values in each column
print("\nNumber of unique values in each column:")
display(df.nunique())
```

Pomocí df.isnull().sum posuzujeme kvalitu dat kontrolou chybějících hodnot v každém sloupci a zobrazením počtu chybějících hodnot. Identifikace a pochopení rozsahu chybějících hodnot je zásadní, protože mohou významně ovlivnit procesy analýzy dat a modelování. V závislosti na povaze a rozsahu chybějících hodnot může být nutné použít techniky, jako je imputace, odstranění nebo dokonce sběr dalších dat.

```
# Count of missing values in each column
print("Missing values count in each column:")
display(df.isnull().sum())
```

4.2.3 Prozkoumání dat

Při prozkoumávání dat nejprve vybereme ze souboru dat sloupce, které jsou předmětem zájmu a vytvoříme seznam nazvaný selected_columns. Korelační matice se vypočítá pomocí metody corr(), která vypočítá párovou korelaci vybraných proměnných. Výsledek se uloží do proměnné corr_matrix. Pro horní trojúhelník korelační matice se vytvoří maska. Je to proto, že matice je symetrická, takže zobrazení pouze dolního trojúhelníku poskytne všechny potřebné informace bez nadbytečných chyb. Pomocí funkce Seaborn diverging_palette je vytvořena vlastní divergující barevná mapa, aby byla heatmapa vizuálně atraktivní a snadno interpretovatelná. Korelační heatmapa je vykreslena pomocí funkce Seaborn heatmap s vlastní colormapou, maskou a dalšími možnostmi, jako jsou šířky čar, poměr stran čtverce a zmenšení barevného pruhu. Nakonec se vytvoří párový graf Seaborn pomocí funkce pairplot(), která generuje grafy rozptylu každé proměnné vůči každé jiné proměnné a poskytuje vizuální znázornění jejich vztahů.

```
# Select columns for heatmap
selected_columns = ['CENA', 'POPLATKY ZA SLUŽBY','VRATNÁ KAUCE',
    'TYP NABÍDKY', 'LOKACE', 'DISPOZICE',
    'STAV', 'VLASTNICTVÍ', 'TYP BUDOVY', 'PLOCHA',
    'VYBAVENO', 'PODLAŽÍ', 'PENB', 'Balkón',
    'Terasa', 'Sklep', 'Lodžie', 'Bezbariérový přístup', 'Parkování',
    'Výtah', 'Garáž', 'MHD', 'Pošta', 'Obchod', 'Banka', 'Restaurace',
    'Lékárna', 'Škola', 'Mateřská škola', 'Sportoviště', 'Hřiště',
    'POPLATKY ZA ENERGIE']
df_selected = df[selected_columns]
corr_matrix = df_selected.corr()

# Create a mask for the upper triangle
mask = np.triu(np.ones_like(corr_matrix, dtype=bool))

# Set up the matplotlib figure
plt.figure(figsize=(12, 8))

# Generate a custom diverging colormap
cmap = sns.diverging_palette(230, 20, as_cmap=True)
```

```
# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(corr_matrix, annot=True, cmap=cmap, mask=mask, linewidths=0.5, square=True,
cbar_kws={"shrink": .5})

# Print the output of the correlation matrix
plt.title("Correlation Heatmap")
plt.show()
sns.pairplot(df_selected)
plt.show()
```

- CENA a PLOCHA – Mezi cenou nemovitosti a její rozlohou existuje pozitivní korelace (0,382377). To znamená, že s rostoucí plochou nemovitosti má tendenci růst i její cena.
- CENA a VRATNÁ KAUCE – Mezi cenou nemovitosti a vratnou kaucí existuje negativní korelace (-0,474472). To znamená, že s rostoucí cenou nemovitosti má vratná kauce tendenci klesat.
- POPLATKY ZA SLUŽBY a VRATNÁ KAUCE – Mezi poplatky za služby a vratnou kaucí existuje silná pozitivní korelace (0,672332). To naznačuje, že nemovitosti s vyššími servisními poplatky mají tendenci mít také vyšší vratné kauce.
- Výtah a Bezbariérový přístup – Mezi přítomností výtahu a bezbariérovým přístupem existuje pozitivní korelace (0,386329). To znamená, že nemovitosti s výtahem mají s větší pravděpodobností také bezbariérový přístup.
- Balkón (balkon) a Sklep (sklep): Mezi přítomností balkonu a sklepa existuje pozitivní korelace (0,225306). To znamená, že nemovitosti s balkonem mají s větší pravděpodobností také sklep.

4.2.4 Průměrný byt k pronájmu

Po prozkoumání dat jsem vytvořil profil typického bytu k pronájmu v celé ČR a v Praze. Zjistilo se, že pronajmout byt s plochou 55 m2 v České republice stojí přibližně 13217 Kč. V Praze má střední byt skoro stejnou plochu (56 m2) a cenu větší o 42 % – tady majitelé bytů vyžadují 18825 Kč/měsíc. Měsíční poplatky v Praze jsou větší o 58 % – 1925 Kč oproti poplatkům mimo Prahu. Nejdražší byt k pronájmu se nachází v Praze a stojí téměř 149000 Kč/měsíc.

4.2.5 Průměrný byt na prodej

Přesuneme-li se k inzerátům o prodeji bytů, pak z datasetu vyplývá, že průměrný byt na prodej jak v ČR, tak i v Praze, má plochu 65 m2. V Praze je tato hodnota asi 7 300 353 Kč, v celé ČR je to 4 141 189 Kč. Nejdražší byt ke koupi stojí téměř 88 050 000 Kč.

4.3 Příprava dat

Pro další ilustraci probíraných konceptů si následně v praktické části předvedeme implementaci přípravy dat na předem vyscrapovaných datech.

Proces přípravy dat se nachází ve třídě DataHandler a slouží k vyčištění sesbíraných dat z webového portálu www.bezrealitky.cz. V konstruktoru přijímá název datasetu jako parametr, což umožňuje zaměnění datasetu. Funkce clean_data() slouží jako vstupní bod a volá všechny ostatní funkce jako odstranění duplicit, odstranění nepotřebných slopců, standardizování názvů, převod na správný datový typ a vyplňování NaN hodnot.

4.3.1 Duplicitní hodnoty

Metoda drop_duplicates() slouží k odstranění duplicitních řádků z DataFrame. Zadáním parametru subset s hodnotou ["URL"] metoda zajistí, že duplicity budou identifikovány na základě sloupce "URL". S argumentem inplace=True metoda přímo upraví původní DataFrame a odstraní duplicity, aniž by vytvořila nový DataFrame. Tento účinný přístup k odstraňování duplicit je zásadním krokem v procesu čištění dat, protože pomáhá zachovat integritu dat a zajišťuje přesnou analýzu v dalších fázích.

```
    def _remove_duplicates(self, df):
        """
        Args:
            df (pd.DataFrame): The input DataFrame to remove duplicates from.
        """
        # Remove duplicate rows based on the 'URL' column
        df.drop_duplicates(subset=['URL'], inplace=True)
```

4.3.2 Nepotřebné hodnoty

Metoda nejprve identifikuje a smaže všechny sloupce obsahující pouze hodnoty NaN voláním metody dropna() s parametry axis=1, how='all' a inplace=True. Tento přístup zajišťuje, že každý sloupec, jehož všechny hodnoty jsou NaN, je odstraněn přímo z původního DataFrame, aniž by byl vytvořen nový. Dále metoda vypustí sloupec "URL" pomocí metody drop(), přičemž zadá parametry labels="URL" a axis=1 spolu s inplace=True, aby se DataFrame upravil na místě.

4.3.3 Standardizace hodnot

Metoda začíná definováním seznamu tuplů umístění, kde každý tupl obsahuje starý název a standardizovaný nový název. Poté odstraní všechny řádky s hodnotami NaN ve sloupci "LOKACE" pomocí metody dropna() s subset=['LOKACE'] a inplace=True. Dále metoda iteruje přes tuply umístění a pomocí metody str.contains() identifikuje řádky obsahující ve sloupci "LOKACE" starý název umístění (bez ohledu na velikost písmen). Tyto řádky jsou poté aktualizovány novým standardizovaným názvem umístění. Nakonec metoda filtruje

DataFrame tak, aby zachovala pouze řádky, kde je hodnota "LOKACE" v seznamu standardizovaných názvů umístění.

4.3.4 Převod hodnot

Metoda nejprve odstraní řádky s hodnotami NaN ve sloupci "CENA" a poté převede zbývající nečíselné znaky ve sloupci "CENA" na celá čísla. Podobně u ostatních sloupců, jako jsou "POPLATKY ZA SLUŽBY", "POPLATKY ZA ENERGII" a "VRATNÁ KAUCE", metoda nahradí nečíselné znaky, hodnoty NaN doplní nulami a převede sloupce na celočíselné datové typy. Pro seznam logických sloupců metoda vyplní hodnoty NaN nulami a převede sloupce na celočíselné datové typy.

Kromě toho metoda převede sloupec "DOSTUPNÉ OD" na datový objekt pomocí funkce pd.to_datetime() se zadaným formátem a ošetřením chyb. Nakonec zpracuje seznam sloupců vzdálenosti odstraněním nečíselných znaků a převede sloupce na celočíselné datové typy pomocí funkce applymap().

4.3.5 NaN hodnoty

Metoda nejprve odstraní řádky s hodnotami NaN ve sloupci "STAV". Poté nahradí hodnoty NaN řetězcem "unknown" v seznamu zadaných sloupců.

4.3.6 Zpracování kategoriálních proměných

Zpracování proměnných demonstruje proces kódování kategoriálních proměnných do číselných hodnot pomocí nástroje LabelEncoder. Jedná se o nezbytný krok při přípravě dat, protože většina algoritmů strojového učení vyžaduje číselné vstupy.

Nejprve vytvoříme seznam s názvem cat_cols, který obsahuje názvy kategoriálních sloupců v datové sadě. Inicializujeme prázdný slovník s názvem label_mappings, který bude uchovávat mapování z původních kategoriálních hodnot na zakódované číselné hodnoty. Kód iteruje přes každý kategoriální sloupec v cat_cols a provede následující kroky. Zavolá instanci třídy LabelEncoder a přiřadí ji proměnné le. Použije metodu fit_transform na kategoriální hodnoty sloupců v DataFrame. Tato metoda transformuje hodnoty sloupců na číselné hodnoty.

```
from sklearn.preprocessing import LabelEncoder

# List of categorical columns
cat_cols = ['TYP NABÍDKY', 'LOKACE', 'DISPOZICE', 'STAV', 'VLASTNICTVÍ', 'TYP BUDOVY',
'VYBAVENO', 'PENB']

# Initialize an empty dictionary to store the mappings
label_mappings = {}

# Apply LabelEncoder for each categorical column and store the mappings
```

```
for col in cat_cols:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])
    label_mappings[col] = dict(zip(le.classes_, le.transform(le.classes_)))

# Print the mappings
for col, mapping in label_mappings.items():
    print(f"{col}: {mapping}")
```

4.4 Modelování

4.4.1 Regrese

Tento kód představuje praktickou implementaci regresních modelů pro úlohu predikce cen nájmů. Účelem je porovnat výkonnost různých regresních modelů (lineárního, Ridgeova a Lasso) na problému predikce cen nájmů jejich trénováním a vyhodnocením pomocí křížové validace. Výstupní metriky zahrnují skóre R2, střední kvadratickou chybu, střední kvadratickou chybu, upravené skóre R2 a střední skóre R2 na základě křížové validace.

Definice funkcí:

- Funkce printRegMetrics vypíše R2 skóre, střední kvadratickou chybu a střední kvadratickou chybu pro předpovědi daného modelu.
- Funkce preprocess_data předběžně zpracuje vstupní data filtrováním nabídek pronájmu, imputací chybějících hodnot a škálováním funkcí.
- Funkce train_evaluate_regression_model trénuje a vyhodnocuje daný regresní model pomocí křížového ověřování.
- Funkce preprocess_data slouží k předzpracování dat a vrací škálované rysy a cílovou proměnnou.
- Funkce train_evaluate_regression_model trénuje a vyhodnocuje regresní modely.

```
# Function definitions
def printRegMetrics(y_test: pd.Series, y_pred: np.ndarray) -> None:
    """
    Prints out basic evaluation metrics of regression models.
    """
    score = r2_score(y_test, y_pred)
    print(f'R2 ------------------------- {round(score, 3)} ({int(round(score * 100, 0))}
%)')
    print(f'Mean squared error ---------- {round(mean_squared_error(y_test, y_pred), 1)}')
    print(f'Root mean squared error ----- {round(np.sqrt(mean_squared_error(y_test,
y_pred)), 1)}')

def preprocess_data(df: pd.DataFrame) -> pd.DataFrame:
```

```
    """
    Preprocesses the data by filtering, imputing missing values, and scaling features.
    """
    df_rent = df[df['TYP NABÍDKY'] == 1].copy()
    df_rent.drop(['TYP NABÍDKY'], axis=1, inplace=True)

    X_rent = df_rent.drop('CENA', axis=1)
    Y_rent = df_rent['CENA']

    # Impute missing values
    imputer = SimpleImputer(missing_values=np.nan, strategy='mean')
    X_rent_imputed = imputer.fit_transform(X_rent)

    # Scale the features
    scaler = StandardScaler()
    X_rent_scaled = scaler.fit_transform(X_rent_imputed)

    return X_rent_scaled, Y_rent

def train_evaluate_regression_model(model, X, y, cv=5):
    """
    Trains and evaluates a regression model using cross-validation.
    """
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=12)

    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)

    printRegMetrics(y_test, y_pred)

    adjusted_r2 = 1 - (1 - model.score(X, y)) * (len(y) - 1) / (len(y) - X.shape[1] - 1)
    print(f"Adjusted R2 score: {adjusted_r2}")

    cv_r2_scores = cross_val_score(model, X, y, cv=cv, scoring='r2')
    mean_cv_r2_score = np.mean(cv_r2_scores)

    print(f"Mean R2 score based on {cv}-fold cross-validation: {mean_cv_r2_score:.3f}
({mean_cv_r2_score*100:.0f}%)")

# Preprocess the data
X_rent, Y_rent = preprocess_data(df)

# Train and evaluate Linear Regression model
print("\nLinear Regression:")
```

```
lin_reg_rent = LinearRegression()
train_evaluate_regression_model(lin_reg_rent, X_rent, Y_rent)

# Train and evaluate Ridge Regression model
print("\nRidge Regression:")
ridge_reg_rent = Ridge(alpha=1.0)
train_evaluate_regression_model(ridge_reg_rent, X_rent, Y_rent)

# Train and evaluate Lasso Regression model
print("\nLasso Regression:")
lasso_reg_rent = Lasso(alpha=1.0)
train_evaluate_regression_model(lasso_reg_rent, X_rent, Y_rent)
```

4.4.2 Klasifikace

Tento kód představuje praktickou implementaci klasifikačních modelů na hodnocení výkonnosti různých klasifikačních modelů na základě průměrného skóre přesnosti.. Je definováno pět klasifikačních modelů (Support Vector Machine, K-Nearest Neighbors, Gaussian Naive Bayes, Decision Tree a Random Forest) a jejich instance jsou uloženy v seznamu.)

Definice funkcí:

- Funkce convert_to_dummies() přijímá DataFrame a seznam atributů a převádí kategoriální proměnné na dummy/indikátorové proměnné, přičemž vypouští první kategorii, aby se zabránilo multikolinearitě.
- Funkce mean_accuracy_score() vypočítá průměrné skóre přesnosti pro různé modely pomocí křížové validace.
- Funkce eval_classification_by_offer_type() vyhodnotí klasifikační modely pro různé typy nabídek (pronájem, prodej, všechny) a vypíše výsledky.

```
def convert_to_dummies(df, dummies_attr):
    return pd.get_dummies(df, columns=dummies_attr, drop_first=True)

def mean_accuracy_score(X, y, models, cv, plot_feature_importance=False):
    for model in models:
        clf = model
        cv_scores = cross_val_score(clf, X, y, cv=cv)
        mean_accuracy = np.mean(cv_scores)
        print(f"{model.__class__.__name__}: Mean Accuracy: {mean_accuracy:.4f}")


def eval_classification_by_offer_type(models: list, data: list, dummies_attr: list,
target: str) -> None:
    """
```

```
    Prints the evaluation of classification models divided by offer type (rent, sell,
all).
    """

    for i, elem in enumerate([df_rent, df_sell, df_all]):
        df_dummies = convert_to_dummies(elem, dummies_attr)

        X = df_dummies.drop(target, axis=1)
        Y = df_dummies[target]

        offer_type = {0: 'RENT', 1: 'SELL', 2: 'ALL'}

        print(f'****************************** {offer_type[i]}
******************************\n')
        mean_accuracy_score(X, Y, models, 5, plot_feature_importance=False)
        print('\n')

# Models

SVC_model = SVC()
KNN_model = KNeighborsClassifier(n_neighbors=5)
GNB_model = GaussianNB()
DTC_model = DecisionTreeClassifier()
RFC_model = RandomForestClassifier(n_estimators=100)

models = [SVC_model, KNN_model, GNB_model, DTC_model, RFC_model]

df_rent = df[df['TYP NABÍDKY'] == 0].copy()
df_rent.drop(['TYP NABÍDKY'], axis=1, inplace=True)

df_sell = df[df['TYP NABÍDKY'] == 1].copy()
df_sell.drop(['TYP NABÍDKY', 'POPLATKY ZA SLUŽBY', 'VRATNÁ KAUCE'], axis=1, inplace=True)

df_all = df.copy()
df_all.drop(['TYP NABÍDKY', 'POPLATKY ZA SLUŽBY', 'VRATNÁ KAUCE'], axis=1, inplace=True)

dfs = [df_rent, df_sell, df_all]

dummies_attr = ['LOKACE', 'DISPOZICE', 'STAV', 'VLASTNICTVÍ', 'TYP BUDOVY', 'PLOCHA',
'VYBAVENO', 'PODLAŽÍ']

eval_classification_by_offer_type(models, dfs, dummies_attr, 'PENB')
```

4.4.3 Clustering

Tento kód představuje praktickou implementaci shlukovacího modelu na analýzu pomocí algoritmu K-means. Hlavním cílem je najít optimální počet shluků pro danou sadu dat, vyhodnotit model a vytvořit DataFrame s informacemi o shlucích.

Definice funkcí:

- Funkce find_optimum_n_clusters_elbow(data, kmodes) najde optimální počet shluků pro model K-means. Lze ji také použít pro algoritmus KModes nastavením parametru kmodes na hodnotu True. Funkce vykreslí součet čtvercových vzdáleností (WCSS) v závislosti na počtu shluků, aby pomohla určit "bod lokte", který představuje optimální počet shluků.
- Funkce evaluate_kmeans(data, n) vyhodnotí model K-means s daným počtem shluků n pomocí skóre siluety. Siluetové skóre měří podobnost objektu s jeho vlastním shlukem ve srovnání s ostatními shluky. Čím vyšší je skóre, tím lepší je shlukování. Funkce vrací přizpůsobený model K-means.
- Funkce create_df_with_clusters(data, attr_type) vytvoří DataFrame se shluky na základě vstupního DataFrame se sloupcem "cluster". Výsledný DataFrame bude obsahovat buď hodnoty mediánu pro každý shluk (pokud je attr_type 'num'), nebo hodnoty módu pro každý shluk (pokud je attr_type 'cat').

```
def find_optimum_n_clusters_elbow(data: pd.DataFrame, kmodes: bool = False) -> None:
    """
    Finds optimum number of clusters for K-means model using the "elbow" method.
    """

    ssd = []

    if kmodes:
        K = range(1, 10)
        for k in K:
            kmodes = KModes(n_clusters=k, n_init=5)
            kmodes.fit_predict(data)
            ssd.append(kmodes.cost_)
    else:
        K = range(1, 30)
        for k in K:
            kmeans = KMeans(n_clusters=k)
            kmeans.fit(data)
            ssd.append(kmeans.inertia_)


    plt.plot(K, ssd, 'bx-')
    plt.xlabel("Distance Residual Sums for K Values (WCSS)")
    plt.title("Elbow Method for Optimum Number of Clusters")
```

```
    plt.rcParams['figure.figsize'] = (12, 5)

    plt.show()

def evaluate_kmeans(data: pd.DataFrame, n: int) -> KMeans:
    """
    Prints the evaluation of K-means model with n-clusters.
    """

    kmeans = KMeans(n_clusters=n).fit(data)
    score = silhouette_score(data, kmeans.labels_, metric='euclidean')

    print(f'Silhouette score for {n} clusters --------- {round(score, 3)}')

    return kmeans

def create_df_with_clusters(data: pd.DataFrame, attr_type: str) -> pd.DataFrame:
    """
    Creates a DataFrame with clusters. The result depends on attribute type:
        - attr_type = 'num' -> return median values for each cluster
        - attr_type = 'cat' -> return mode values for each cluster

    Note: Provided DataFrame should aready contain 'cluster' column.
    """

    clusters = []

    for i in range(len(data['cluster'].unique())):
        cluster = data[data['cluster'] == i]

        if attr_type == 'num':
            values = cluster.drop('cluster', axis=1).describe().loc['50%']
        elif attr_type == 'cat':
            values = cluster.drop('cluster', axis=1).describe().loc['top']
        else:
            print(f'{attr_type} is not a valid attribute type. Could be only "num" or
"cat".')
            return

        count = len(cluster)
        row = [f'cluster_{i+1}', count] + values.tolist()
        clusters.append(row)

    clusters_df = pd.DataFrame(clusters, columns=['cluster', 'count'] +
data.drop('cluster', axis=1).columns.tolist())
```

```
    return clusters_df


# Assuming you already have a DataFrame named df_sell.
df_sell_part = df_sell[['CENA', 'PLOCHA']]

find_optimum_n_clusters_elbow(df_sell_part)

kmeans2 = evaluate_kmeans(df_sell_part, 2)

# Create a copy of df_sell_part to avoid modifying it directly.
df_sell_part_clusters = df_sell_part.copy()
df_sell_part_clusters['cluster'] = kmeans2.fit_predict(df_sell_part)

clusters_df = create_df_with_clusters(df_sell_part_clusters, attr_type='num')
print(clusters_df)

df_sell_clusters = df_sell.copy()
df_sell_clusters['cluster'] = kmeans2.fit_predict(df_sell_part)
print(df_sell_clusters)
```

4.5 Evaluace

Regrese

```
Lineární regrese:
R2: 0,64 (64 %)
MSE: 14 013 326,1
RMSE: 3 743,4
Upravené skóre R2: 0,7426
Průměrné skóre R2 na základě pětinásobné křížové validace: 0.706 (71%)
Hřebenová regrese:
R2 skóre: 0,64 (64 %)
MSE: 14 011 806,7
RMSE: 3 743,2
Upravené skóre R2: 0,7426
Průměrné skóre R2 na základě pětinásobné křížové validace: 0.706 (71%)
Lasso regrese:
R2 skóre: 0,641 (64 %)
MSE: 14 009 310,3
RMSE: 3 742,9
Upravené skóre R2: 0,7426
Průměrné skóre R2 na základě pětinásobné křížové validace: 0.706 (71%)
```

Po vyhodnocení výstupů vidíme, že všechny tři modely mají podobný výkon, pokud jde o skóre R2, MSE a RMSE. Skóre R2 se u každého modelu pohybuje kolem 64 %, což znamená, že vysvětlují přibližně 64 % rozptylu v datech. Upravené skóre R2 a průměrné skóre R2 založené na pětinásobném křížovém ověřování jsou u všech modelů také velmi podobné.

Je však důležité poznamenat, že model Lasso Regression si vede o něco lépe než ostatní dva modely, má nepatrně vyšší skóre R2 (0,641) a nižší hodnoty MSE a RMSE. To může naznačovat, že model Lasso Regression je pro tuto konkrétní datovou sadu lepší volbou, ale rozdíly ve výkonnosti jsou poměrně malé.

Klasifikace

```
RENT:
Support Vector Machine (SVC): 71.35%
K-Nearest Neighbors (KNN): 68.09%
Gaussian Naive Bayes (GNB): 22.89%
Decision Tree (DTC): 70.44%
Random Forest (RFC): 80.85%
SELL:
Support Vector Machine (SVC): 41.37%
K-Nearest Neighbors (KNN): 36.59%
Gaussian Naive Bayes (GNB): 32.30%
Decision Tree (DTC): 40.01%
Random Forest (RFC): 54.29%
ALL:
Support Vector Machine (SVC): 53.28%
K-Nearest Neighbors (KNN): 52.30%
Gaussian Naive Bayes (GNB): 23.98%
Decision Tree (DTC): 55.09%
Random Forest (RFC): 66.65%
```

Z výsledků vyplývá, že pro datovou sadu RENT dosahuje nejlepšího výsledku klasifikátor Random Forest s průměrnou přesností 80,85 %, následovaný klasifikátorem Support Vector Machine s přesností 71,35 %. Nejnižší přesnost 22,89 % vykazuje Gaussův Naive Bayes.

U datové sady PRODEJ si nejlépe vede klasifikátor Random Forest s průměrnou přesností 54,29 %. Všechny ostatní modely mají přesnost pod 50 %, přičemž model Gaussian Naive Bayes má nejnižší přesnost 32,30 %.

U datové sady ALL, která kombinuje nabídky RENT i SELL, dosahuje Random Forest Classifier opět nejlepších výsledků s průměrnou přesností 66,65 %. Klasifikátor rozhodovacího stromu je na druhém místě s 55,09 %. Nejnižší přesnost 23,98 % má model Gaussian Naive Bayes.

Souhrnně lze říci, že klasifikátor Random Forest dosahuje konzistentně nejlepších výsledků ve všech třech datových sadách. Před výběrem nejlepšího modelu pro daný problém je však nutné zvážit i další faktory, jako je například přiléhavost, interpretovatelnost a výpočetní čas.

Clustering

Siluetové skóre pro model se dvěma shluky je 0,61, což naznačuje přiměřenou úroveň separace mezi oběma shluky. Siluetové skóre se pohybuje v rozmezí od -1 do 1, přičemž hodnoty bližší 1 znamenají dobré oddělení shluků a hodnoty bližší -1 znamenají špatné oddělení. Skóre 0,61 naznačuje, že shluky jsou přiměřeně dobře odděleny, ale je zde prostor pro zlepšení.

Souhrnná statistika shluků ukazuje mediánové hodnoty pro oba shluky. Shluk 1 má 1 732 datových bodů s mediánem ceny (PRICE) 10 000 a mediánem plochy (AREA) 50 metrů čtverečních. Shluk 2 má 550 datových bodů s mediánem ceny 20 490 a mediánem plochy 63 metrů čtverečních.

Celkově výstup shlukování naznačuje, že algoritmus K-means identifikoval dvě odlišné skupiny nemovitostí s různými cenovými a plošnými charakteristikami. Siluetové skóre naznačuje, že oddělení shluků je přiměřené, ale může být prostor pro zlepšení.

Závěr

Cílem této práce byla datová analýza českého realitního trhu se zaměřením na sběr konkrétních dat a následná aplikace technik data miningu na sesbíraná data. Práce je rozdělena na 2 části a to na teoretickou a praktickou.

V teoretické části jsme si popsali teorii web scrapingu. Vysvětlili jsme základní principy a pojmy související s touto metodou, včetně nástrojů a technik pro extrakci dat z online zdrojů. Dále jsme si popsali metody zajištění kvality a spolehlivosti získaných dat. Ve druhé části jsme se zaměřili na teoretický aspekt dobývaní znalostí z databází, tedy na proces, metodiky, metody a techniky používané při analýze a interpretaci dat.

V praktické části jsme prakticky implementovali teorii web scrapingu, s jejíž pomocí jsme sbírali data z realitního portálu www.bezrealitky.cz. Tato data jsme následně zkoumali pomocí metod a technik dobývání znalostí z databází a vyhodnocovali kvalitu zvolených modelů.

Jedním z hlavních přínosů práce bylo odhalení pozoruhodných hodnot v datech. Jako osobní příspěvek mi projekt poskytl seznámení s různými metodami dolování dat, praktické zkušenosti a hlubší porozumění oblasti nemovitostí. V rámci rozšíření práce by bylo možné zvážit využití více zdrojů dat s pravidelně aktualizovanými informacemi, aby byly inzeráty ve výsledcích aktuální. Z mého pohledu nemusí být samotné výsledky modelování prakticky využitelné a je potřeba lepší předzpracování dat podrobnějším prozkoumáním oblasti domény pro výběr a použití nejvhodnějších modelů dolování dat.

Použitá literatura

BOEGERSHAUSEN, Johannes, Hannes DATTA, Abhishek BORAH a Anew T. STEPHEN, 2022. Fields of Gold: Scraping Web Data for Marketing Insights. *Journal of marketing* [online]. **86**(5), 1–20. ISSN 0022-2429. Dostupné z: https://go.exlibris.link/w0g4KK4d

CHAPAGAIN, Anish, 2019. *Hands-On Web Scraping with Python: Perform Advanced Scraping Operations Using Various Python Libraries and Tools Such As Selenium, Regex, and Others* [online]. Birmingham: Packt Publishing, Limited. Book, Whole. ISBN 1789533392. Dostupné z: https://go.exlibris.link/2QHWx5HH

AYDIN, Olgun, 2018. *R Web Scraping Quick Start Guide: Techniques and Tools to Crawl and Scrape Data from Websites* [online]. Birmingham: Packt Publishing, Limited. Book, Whole. ISBN 1789138736. Dostupné z: https://go.exlibris.link/Td5HPbWv

OANCEA, Bogdan a Marian NECULA, 2019. Web scraping techniques for price statistics - the Romanian experience. *Statistical journal of the IAOS* [online]. **35**(4), 657–667. ISSN 1874-7655. Dostupné z: https://go.exlibris.link/NPY0XQnH

RAGHAVENDRA, Sujay a SPRINGERLINK (ONLINE SLUŽBA), 2021. *Python testing with Selenium: learn to implement different testing techniques using the Selenium WebDriver* [online]. New York: Apress. Book, Whole. ISBN 9781484262498. Dostupné z: https://go.exlibris.link/B1cDwdj7

LAUREN KLEIN, 2021. *Web Scraping — Part 1 — Introduction to Cultural Analytics & Python* [online]. Dostupné z: https://melaniewalsh.github.io/Intro-Cultural-Analytics/04-Data-Collection/02-Web-Scraping-Part1.html

FIESLER, Casey, Nathan BEARD a Brian C. KEEGAN, 2020. No Robots, Spiders, or Scrapers: Legal and Ethical Regulation of Data Collection Methods in Social Media Terms of Service. *Proceedings of the International AAAI Conference on Web and Social Media* [online]. **14**, 187–196. ISSN 2334-0770, 2162-3449. Dostupné z: https://ojs.aaai.org/index.php/ICWSM/article/view/7290

MITCHELL, Ryan E., 2018. *Web scraping with Python: collecting more data from the modern web* [online]. Second. Beijing;Farnham;Sebastopol;Tokyo;Boston; O'Reilly. Book, Whole. ISBN 1491985577. Dostupné z: https://go.exlibris.link/2QMx0lcS

KHALIL, Salim a Mohamed FAKIR, 2017. RCrawler: An R package for parallel web crawling and scraping. *SoftwareX* [online]. **6**(Journal Article), 98–106. ISSN 2352-7110. Dostupné z: doi:10.1016/j.softx.2017.04.004

MUNZERT, Simon, Christian RUBBA, Peter MEISSNER a Dominic NYHUIS, 2014. *Automated data collection with R: a practical guide to web scraping and text mining* [online]. New York: WILEY. Book, Whole. ISBN 111883481X. Dostupné z: https://go.exlibris.link/HlChRq5V

GARCÍA, Boni, Mario MUNOZ-ORGANERO, Carlos ALARIO-HOYOS a Carlos Delgado KLOOS, 2021. Automated driver management for Selenium WebDriver. *Empirical software engineering : an international journal* [online]. **26**(5). ISSN 1382-3256. Dostupné z: doi:10.1007/s10664-021-09975-3

ANDREW R H GIRDWOOD, 2009. *What is a headless browser?* [online]. [vid. 2023-04-24]. Dostupné z: https://blog.arhg.net/2009/10/what-is-headless-browser.html

MKS, 2019. Difference between Static and Dynamic Web Pages. *GeeksforGeeks* [online]. [vid. 2023-04-25]. Dostupné z: https://www.geeksforgeeks.org/difference-between-static-and-dynamic-web-pages/

Documentation. logging — Logging facility for Python. *Python documentation* [online] [vid. 2023-04-25]. Dostupné z: https://docs.python.org/3/library/logging.html

KRISHNAN, Sanjay, Jiannan WANG, Eugene WU, Michael J. FRANKLIN a K. GOLDBERG, 2016. ActiveClean: Interactive data cleaning for statistical modeling. In: *Proceedings of the VLDB Endowment* [online]. s. 948–959. ISBN 2150-8097. Dostupné z: https://go.exlibris.link/XKsXft8W

CHU, Xu, John MORCOS, Ihab ILYAS, Mourad OUZZANI, Paolo PAPOTTI, Nan TANG a Yin YE, 2015. KATARA: A Data Cleaning System Powered by Knowledge Bases and Crowdsourcing. In: *Proceedings of the 2015 ACM SIGMOD International Conference on management of data* [online]. B.m.: ACM, s. 1247–1261. ISBN 0730-8078. Dostupné z: doi:10.1145/2723372.2749431

NELLI, Fabio, 2015. *Python Data Analytics: Data Analysis and Science Using Pandas, Matplotlib and the Python Programming Language* [online]. Berkeley, CA: Apress L. P. Book, Whole. ISBN 1484209591. Dostupné z: https://go.exlibris.link/1hBfr9XX

HAN, Jiawei, Micheline KAMBER, Jian PEI Ph.D, a SCIENCEDIRECT (ONLINE SLUŽBA), 2012. *Data mining: concepts and techniques* [online]. Third. San Francisco;New York;San Diego;Singapore;Amsterdam;Tokyo;London;Heidelberg;Oxford;Paris;Sydney;Boston; Morgan Kaufmann is an imprint of Elsevier. Book, Whole. ISBN 0123814804. Dostupné z: https://go.exlibris.link/hXZPVzDP

PLCHÚT, Martin. Dobývání znalostí z databází: Úvod a oblasti aplikací. In: euromise.vse.cz [online prezentace]. [cit. 2014-05-13]. Dostupné z: http://www.fit.vutbr.cz/study/courses/ZZD/public/seminar0304/Uvod.pdf

MRÁZOVÁ, Iveta. Dobývání znalostí. Přednáška [online]. Praha, UK-MFF, [cit. 2014-05-13]. Dostupné z: http://ksvi.mff.cuni.cz/~mraz/datamining/lecture/Dobyvani_Znalosti_Prednaska_Uvod.pdf

IDRIS, Ivan, 2013. *NumPy: beginner's guide : an action packed guide using real world examples of the easy to use, high performance, free open source NumPy mathematical library, Second edition* [online]. 2nd vyd. Birmingham, England: PACKT Publishing. Book, Whole. ISBN 1782166084. Dostupné z: https://go.exlibris.link/fD6MWFkY

DANEL, Roman. Dolování dat. Přednáška [online]. Ostrava, VŠB-TU, 2010. [cit. 2014-05-11]. Dostupné z: http://homel.vsb.cz/~dan11/is_skripta/IS%202010%20-%20Danel%20-%20Dolovani%20dat.pdf

Azevedo, A. and Santos, M. F.., 2019. Wayback Machine [online] [vid. 2013-01-09]. Dostupné z: https://web.archive.org/web/20190210044429/https://pdfs.semanticscholar.org/7dfe/3bc6035da527deaa72007a27cef94047a7f9.pdf

RAUCH, Jan a Milan ŠIMŮNEK. Dobývání znalostí z databází, LISp-Miner a GUHA. Praha: Oeconomica, nakladatelství VŠE, 2014. Odborná kniha s vědeckou redakcí. ISBN 9788024520339.

PLCHÚT, Martin. Dobývání znalostí z databází: Úvod a oblasti aplikací. In: euromise.vse.cz [online prezentace]. [cit. 2014-05-13]. Dostupné z: http://www.fit.vutbr.cz/study/courses/ZZD/public/seminar0304/Uvod.pdf

MARTINEZ-PLUMED, Fernando, Lidia CONTRERAS-OCHANDO, Cesar FERRI, Jose HERNANDEZ-ORALLO, Meelis KULL, Nicolas LACHICHE, Maria Jose RAMIREZ-QUINTANA a Peter FLACH, 2021. CRISP-DM Twenty Years Later: From Data Mining Processes to Data Science Trajectories. *IEEE transactions on knowledge and data engineering* [online]. **33**(8), 3048–3061. ISSN 1041-4347. Dostupné z: doi:10.1109/TKDE.2019.2962680

Doc. RNDr. Irena HOLUBOVÁ, Ph.D. Data Science Life Cycle, CRISP-DM Methodology [online prezentace]. [cit. 2016]. Dostupné z: https://www.ksi.mff.cuni.cz/~holubova/NDBI048/slajdy/03.1_CRISP-DM_Methodology.pdf

HOTZ, Nick, 2018. What is CRISP DM? *Data Science Process Alliance* [online]. Dostupné z: https://www.datascience-pm.com/crisp-dm-2/

VADAPALLI PAVAN, 2022. 6 Types of Regression Models in Machine Learning You Should Know About. *upGrad blog* [online]. Dostupné z: https://www.upgrad.com/blog/types-of-regression-models-in-machine-learning/

SHARMA, Prashant, 2022. Different Types of Regression Models. *Analytics Vidhya* [online]. Dostupné z: https://www.analyticsvidhya.com/blog/2022/01/different-types-of-regression-models/

KAUSHIK, 2016. Clustering | Introduction, Different Methods, and Applications (Updated 2023). *Analytics Vidhya* [online]. Dostupné z: https://www.analyticsvidhya.com/blog/2016/11/an-introduction-to-clustering-and-different-methods-of-clustering/

CHANDOLA, Varun, Arindam BANERJEE a Vipin KUMAR, 2009. Anomaly detection: A survey. *ACM Computing Surveys* [online]. **41**(3), 15:1-15:58. ISSN 0360-0300. Dostupné z: doi:10.1145/1541880.1541882

CHANDOLA, Varun, Arindam BANERJEE a Vipin KUMAR, 2009. Anomaly detection: A survey. *ACM Computing Surveys* [online]. **41**(3), 15:1-15:58. ISSN 0360-0300. Dostupné z: doi:10.1145/1541880.1541882

Přílohy

Příloha A: Název první přílohy

Příloha obsahuje okomentovaný zdrojový kód v jazyce Python.

1. app/main.py
2. app/webscraper.py
3. app/webscraper.log
4. app/pycache
5. app/venv/etc
6. app/venv/include
7. app/venv/lib
8. app/venv/Scripts
9. app/venv/share
10. app/venv/pyvenv.cfg
11. .gitignore
12. Analysis.ipynb
13. Listings_data.csv
14. README.md
15. Requirements.txt