

Vysoká škola ekonomická v Praze

Fakulta informatiky a statistiky



Využití data miningu pro analýzu českého realitního trhu

BAKALÁŘSKÁ PRÁCE

Studijní program: Aplikovaná informatika

Studijní obor: Aplikovaná informatika

Autor: Ilya Tsakunov

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

Praha, květen 2022

Prohlášení

Prohlašuji, že jsem bakalářskou práci Využití data miningu pro analýzu českého realitního trhu vypracoval samostatně za použití v práci uvedených pramenů a literatury.

V Praze dne 9. května 2022

.....

Ilya Tsakunov

Poděkování

V první řadě bych chtěl poděkovat Ing. Davidu Chudánovi, Ph.D. za jeho čas a cenné připomínky, které mi pomohly při zpracování této práce. Dále bych rád poděkoval své rodině a blízkým za podporu během mého bakalářského studia.

Abstrakt

Cílem této bakalářské práce je získat data z českého realitního portálu pomocí web scrapingu a následně tato data analyzovat s využitím explorační analýzy a vybraných metod data miningu za účelem vyhledání zajímavých vztahů. Práce se dělí na teoretickou a praktickou část.

Teoretická část předně představuje oblast data miningu, včetně popisu nezbytných pojmu, vybraných metod a způsobů jejich evaluace. Kromě toho jsou popsány populární metodiky data miningu, obzvlášť je věnována pozornost metodice CRISP-DM. Pak je popsána technologie web scrapingu, jeho principy, existující řešení a také etický aspekt.

Praktická část začíná představením nástrojů, které byly použity během analýzy. Pak obsahuje stručné seznámení s doménovou oblastí realitních dat. Následuje popis sběru dat z webových stránek realitního portálu, včetně hledání API a tvorby skriptu v jazyce Python. Získaný dataset je dále předzpracován v prostředí Jupyter Notebook. Výsledná data jsou nejprve analyzována pomocí explorační analýzy. Potom následuje analýza s využitím klasifikačních, regresních a popisních metod data miningu. Na konci práce jsou prodiskutovány výsledky analýzy a na závěr je shrnutá celá práce.

Klíčová slova

Data mining, web scraping, realitní trh, explorační analýza.

Abstract

The aim of this bachelor thesis is to obtain data from the Czech real estate portal using web scraping and then analyze this data using exploratory analysis and selected data mining methods in order to find interesting relationships. The thesis is divided into theoretical and practical part.

The theoretical part primarily represents the area of data mining, including the description of the necessary concepts, selected methods and ways of its evaluation. In addition, popular data mining methodologies are described, in particular, attention is paid to the CRISP-DM methodology. Then the technology of web scraping, its principles, existing solutions, and also the ethical aspect are described.

The practical part begins with the introduction of the tools that were used during the analysis. Then it contains a brief introduction to the domain area of real estate. This is followed by a description of data collection from the real estate portal website, including API search and script creation in Python. The obtained dataset is further pre-processed in the Jupyter notebook environment. The resulting data is first analyzed using exploratory analysis. Then follows the analysis using classification, regression and descriptive methods of data mining. At the end of the thesis, the results of the analysis are discussed, and in conclusion the whole thesis is summarized.

Keywords

Data mining, web scraping, real estate market, exploratory analysis

Obsah

Úvod	4
1 Data mining.....	5
1.1 Definice a historie vzniku	5
1.2 Metodiky.....	6
1.2.1 SEMMA	6
1.2.2 5A.....	6
1.2.3 CRISP-DM	7
1.3 Metody data miningu.....	9
1.3.1 Prediktivní metody.....	9
1.3.2 Deskriptivní metody	10
1.4 Vyhodnocení modelu.....	11
1.4.1 Rozdělení dat.....	11
1.4.2 Klasifikační vyhodnocení.....	12
1.4.3 Regresní vyhodnocení.....	15
2 Web scraping	18
2.1 Definice a principy	18
2.2 Struktura webové stránky.....	19
2.2.1 XML	19
2.2.2 XPath.....	19
2.3 Existující řešení.....	21
2.3.1 API	21
2.3.2 Cloudová řešení	21
2.3.3 Knihovny pro Python	22
2.4 Problémy	23
2.4.1 CAPTCHA	24
2.4.2 Honeypot.....	24
2.4.3 Dynamický obsah	24
2.4.4 Změny struktury webový stránek.....	24
2.4.5 Mnoho častých HTTP požadavků.....	25
2.4.6 IP blokování	25
2.5 Etický aspekt	25
2.5.1 Web scraping a GDPR.....	26
2.5.2 Robots.txt	26
3 Získání dat a analýza.....	27
3.1 Použité nástroje.....	27
3.1.1 Python a knihovny.....	27
3.1.2 Jupyter Notebook	27
3.1.3 Tableau	27
3.1.4 Github.....	28
3.2 Porozumění doménové oblasti.....	28

<i>3.3 Získání dat</i>	<i>28</i>
3.3.1 Robots.txt.....	29
3.3.2 Hledání API.....	29
3.3.3 Zkoumání API	31
3.3.4 Scraping.....	32
<i>3.4 Porozumění datům</i>	<i>35</i>
3.4.1 Přehled atributů	35
3.4.2 Příklad záznamů.....	38
<i>3.5 Předpracování dat</i>	<i>39</i>
3.5.1 Odstranění sloupců a řádků.....	39
3.5.2 Transformace (převod na jiné datové typy).....	40
3.5.3 Chybějící hodnoty.....	41
3.5.4 Inzeráty o pronájmu	46
3.5.5 Inzeráty o prodeji	48
<i>3.6 Explorační analýza.....</i>	<i>49</i>
3.6.1 Mapa nabídek	50
3.6.2 Průměrný byt k pronájmu	51
3.6.3 Průměrný byt na prodej	52
3.6.4 Histogram ceny	53
3.6.5 Cena a plocha dle dispozicí	55
3.6.6 Poplatky.....	61
3.6.7 Populární byty	62
3.6.8 Informace o budovách.....	63
3.6.9 Popis bytu.....	64
3.6.10 Korelace atributů	66
<i>3.7 Modelování.....</i>	<i>70</i>
3.7.1 Regresní úloha	72
3.7.2 Klasifikační úloha	78
3.7.3 Shluková analýza	79
3.7.4 Detekce anomalií	89
<i>3.8 Diskuze výsledků.....</i>	<i>90</i>
Závěr	92
Použitá literatura	93
Přílohy.....	96
<i>Příloha A: Zdrojový kód web scraperu</i>	<i>96</i>
<i>Příloha B: Zdrojový kód Jupyter Notebooks.....</i>	<i>96</i>
<i>Příloha C: Data</i>	<i>96</i>

Seznam obrázků

Obrázek 1: Přehled Data Mining metodik. Zdroj: (4).....	6
Obrázek 2: Cyklus metodiky CRISP-DM. Zdroj: (6)	7
Obrázek 3: Matice záměn. Zdroj: (16)	12
Obrázek 4: Gain a Lift křivka. Zdroj: (19)	14
Obrázek 5: ROC křivka. Zdroj: (21)	15
Obrázek 6: Zjednodušená ukázka stromu HTML dokumentu (autor)	20
Obrázek 7: Soubor robots.txt pro webovou stránku Bezreality (autor)	29
Obrázek 8: Hledání API v přenášených souborech (autor)	30
Obrázek 9: Nalezené API (autor).....	31
Obrázek 10: Příklad záznamů z API (autor)	31
Obrázek 11: Příklad záznamů z API po rozdelení a odstranění atributů (autor)	32
Obrázek 12: Implementace funkce add_column_with_links() (autor)	32
Obrázek 13: Příklad stránky s inzerátem (autor)	33
Obrázek 14: Určení XPath vybraných elementů (autor)	34
Obrázek 15: Ukázka funkce pro scrapování inzerátů (autor)	35
Obrázek 16: Příklad získaných dat (autor)	38
Obrázek 17: Odstranění nerelevantních sloupců (autor)	39
Obrázek 18: Odstranění nerelevantních záznamů (autor)	39
Obrázek 19: Odstranění inzerátů ze Slovenska (autor)	39
Obrázek 20: Odstranění inzerátů v jiných měnách než CZK (autor)	39
Obrázek 21: Odstranění inzerátů s nulovou plochou (autor)	40
Obrázek 22: Převod vybraných atributů do binárních hodnot (autor)	40
Obrázek 23: Převod textových atributů do numerických (autor).....	40
Obrázek 24: Převod atributu „měna“ do numerického (autor).....	41
Obrázek 25: Seskupení málo četných hodnot atributu „dispozice“ (autor)	41
Obrázek 26: Převod vybraných atributů do kategoriálních (autor)	41
Obrázek 27: Odstranění inzerátů s chybějícími hodnotami u atributů o okolí (autor).....	43
Obrázek 28: Vyplnění chybějících hodnot u atributu „podlaží“ (autor)	44
Obrázek 29: Vyplnění chybějících hodnot u atributu „vybavenost“ (autor)	45
Obrázek 30: Rozdělení kategorií atributu „vybavenost“ (autor).....	45
Obrázek 31: Vyplnění chybějících hodnot u atributu „stav_budovy“ (autor)	45
Obrázek 32: Vyplnění chybějících hodnot u atributu „typ_vlastnictví“ (autor)	45
Obrázek 33: Odstranění inzerátů s cenou menší než 1500 Kč (autor).....	46
Obrázek 34: Přidání nového atributu „deposit_by_price“ (autor)	46
Obrázek 35: Identifikace outlierů u atributu „deposit_by_price“ (autor)	47
Obrázek 36: Graf ceny a kauce (autor)	47
Obrázek 37: Identifikace outlierů u atributu „poplatky“ (autor)	48
Obrázek 38: Graf ceny a poplatků (autor).....	48
Obrázek 39: Odstranění inzerátů s cenou menší než 200 000 Kč (autor).....	49
Obrázek 40: Mapa inzerátů (autor)	50
Obrázek 41: Inzeráty s vysokými poplatky v Moravskoslezském kraji (autor)	51
Obrázek 42: Průměrný byt k pronájmu v České republice (autor)	52
Obrázek 43: Průměrný byt k pronájmu v Praze (autor)	52
Obrázek 44: Průměrný byt na prodej v České republice (autor).....	53

Obrázek 45: Průměrný byt na prodej v Praze (autor)	53
Obrázek 46: Histogram cen bytů k pronájmu (autor).....	54
Obrázek 47: Histogram cen bytů na prodej (autor)	54
Obrázek 48: Rozložení ceny, plochy a poplatků bytů k pronájmu v ČR (autor)	56
Obrázek 49: Rozložení ceny, plochy a poplatků bytů k pronájmu v Praze (autor)	57
Obrázek 50: Rozložení ceny, plochy a ceny za m ² bytů na prodej v ČR (autor)	58
Obrázek 51: Rozložení ceny, plochy a ceny za m ² bytů na prodej v Praze (autor).....	59
Obrázek 52: Přehled cen bytů dle vybavenosti, stavu a typy budovy (autor).....	61
Obrázek 53: Přehled poplatků dle energetické třídy PENB (autor)	62
Obrázek 54: Populární byty (autor).....	63
Obrázek 55: Přehled inzerátů dle informací o budově (autor).....	63
Obrázek 56: Přehled inzerátů dle podlaží (autor)	65
Obrázek 57: Charakteristika bytů k pronájmu (autor)	65
Obrázek 58: Charakteristika bytů na prodej (autor)	66
Obrázek 59: Převod vybraných atributů do dummies proměnných (autor)	66
Obrázek 60: Korelace vybraných atributů (autor)	67
Obrázek 61: Korelace vybraných atributů u bytů na prodej (autor)	68
Obrázek 62: Korelace vybraných atributů u bytů k pronájmu (autor).....	69
Obrázek 63: Korelace ceny, plochy, poplatků a kouce (autor)	70
Obrázek 64: Funkce pro převod atributů do dummies proměnných (autor)	70
Obrázek 65: Pomocné funkce pro modelování 1 (autor)	71
Obrázek 66: Pomocné funkce pro modelování 2 (autor)	71
Obrázek 67: Příprava a rozdělení dat pro regresi (autor)	72
Obrázek 68: Vytvoření a evaluace lineární regrese (autor).....	72
Obrázek 69: Vytvoření a evaluace lasso regrese (autor)	73
Obrázek 70: Určení důležitosti jednotlivých atributů pomocí lasso regrese (autor)	74
Obrázek 71: Vytvoření a evaluace hřebenové regrese (autor)	75
Obrázek 72: Určení důležitosti jednotlivých atributů pomocí hřebenové regrese (autor) ..	76
Obrázek 73: Evaluace modelů pro predikci PENB (autor).....	78
Obrázek 74: Evaluace modelů pro predikci typu budovy (autor)	79
Obrázek 75: Evaluace modelů pro predikci stavu budovy (autor)	79
Obrázek 76: Funkce pro hledání optimálního počtu clusterů (autor)	80
Obrázek 77: Funkce pro evaluaci modelu K means (autor)	80
Obrázek 78: Loketní metoda pro určení optimálního počtu clusterů (autor).....	81
Obrázek 79: Clustery dle polohy (autor)	81
Obrázek 80: Přehled cen bytů na prodej dle clusteru (autor)	82
Obrázek 81: Přehled cen bytů k pronájmu dle clusteru (autor)	83
Obrázek 82: Cluster s dražšími a většími byty (autor).....	84
Obrázek 83: Cena a plocha bytů dle clusteru (autor).....	84
Obrázek 84: Příprava dat pro modelování (autor).....	85
Obrázek 85: Cluster s byty s velkými poplatky (autor)	85
Obrázek 86: Příprava dat pro použití algoritmu K-modes (autor)	87
Obrázek 87: Určení optimálních parametrů modelu Isolation Forest (autor)	89
Obrázek 88: Použití modelu Isolation Forest (autor)	89
Obrázek 89: Nalezené anomálie 1 (autor)	89
Obrázek 90: Nalezené anomálie 2 (autor).....	90

Seznam tabulek

Tabulka 1: Přehled atributů v datasetu (autor)	35
Tabulka 2: Přehled chybějících hodnot v datasetu (autor)	42
Tabulka 3: Přehled inzerátů podle podlaží (autor)	43
Tabulka 4: Přehled mediánních cen za m ² v ČR a Praze (autor).....	59
Tabulka 5: Přehled nevybavených bytů dle stavu budovy (autor)	64
Tabulka 6: Výsledky predikce ceny bytu k pronájmu pomocí regresí (autor)	77
Tabulka 7: Výsledky predikce poplatků bytu k pronájmu pomocí regresí (autor).....	77
Tabulka 8: Výsledky predikce ceny bytu na prodej pomocí regresí (autor).....	77
Tabulka 9: Přehled clusterů dle ceny a plochy (autor).....	83
Tabulka 10: Přehled clusterů dle ceny, plochy, poplatků a kauce (autor).....	85
Tabulka 11: Přehled clusterů dle atributů o okolí (autor)	86
Tabulka 12: Přehled clusterů dle kategoriálních atributů (autor).....	88
Tabulka 13: Přehled typických bytů v ČR a Praze (autor)	91

Úvod

Data jsou jednou z klíčových hodnot v dnešní době a jejich objem neustále roste. Nové technologie přináší nová data, nová data generují další data, vznikají nové technologie a tento cyklus se zase opakuje. Rozsáhlá digitalizace tento růst jen urychluje. Ale s růstem objemu dat vzniká i s tím spojený problém, a to zhoršení jejich kvality. Data jsou cenné aktivum, které je potenciálně schopno přinést jak užitek, tak i ztrátu. Kvalitní data s větší pravděpodobností přinesou užitek. Nekvalitní data v nejlepším případě nepřinesou nic. Aby tedy data přinesla užitek, je nutné umět s nimi správně manipulovat a umět transformovat nekvalitní data v kvalitní.

Tato bakalářská práce se věnuje analýze reálných dat z realitního trhu České republiky. Ceny nemovitostí rostou, během posledních několika let byl tento růst nezvykle citelný, developerských projektů je málo, a proto je doména realitního trhu ČR relativně zajímavá z pohledu analýzy. Je možné, že skrývá nějaké zajímavé vztahy. Odhalení podobných vztahů je jednou z úloh data miningu. Před samotnou analýzou je ale nutné získat data. Během svého vyměněného pobytu Erasmus v Eindhovenu (Nizozemsko) jsem se setkal s web scrapingem. Tato technologie mě zaujala a rozhodl jsem ji použít ve své bakalářské práci, abych analyzoval ta nejaktuálnější data z velkého českého realitního portálu. Cílem práce je tedy extrahovat data z jednoho ze předních českých realitních portálů a následně tato data analyzovat s využitím explorační analýzy a klasifikačních, regresních a popisních metod data miningu.

Práce se obecně dělí na teoretickou a praktickou část. Teoretická část se skládá ze 2 kapitol. V první kapitole jsou představeny metodiky a principy data miningu. Zvláštní pozornost je věnována metodice CRISP-DM, která bude také využita v praktické části. Další kapitola je o web scrapingu a zahrnuje popis jeho procesu, přehled a srovnání existujících řešení, představení základní struktury webových stránek a také etický aspekt web scrapingu.

Praktická část předpokládá aplikace znalostí z předchozí teoretické části na reálných datech. Skládá se z 8 subkapitol. První subkapitola krátce popisuje nástroje použité pro analýzu. Další subkapitoly v podstatě představují jednotlivé kroky metodiky CRISP-DM – od fáze porozumění doménové oblasti až po fázi evaluace výsledků. Mezi těmito subkapitolami je také část věnovaná procesu získání dat z webových stránek realitního zprostředkovatele s využitím web scrapingu. Většina analýzy je provedena v jazyce Python.

1 Data mining

Tato kapitola má za cíl seznámit čtenáře s oblastí data miningu. Obsahuje tak definice pojmu, krátkou historii data miningu. Dále je popsáno několik metodik procesu dobývání znalostí z databází: 5A, SEMMA a CRISP-DM. Metodika CRISP-DM je popsána nejvíce, jelikož právě ta bude použita v praktické části. Na konci kapitoly jsou představeny vybrané metody data miningu a způsoby jejich vyhodnocení.

1.1 Definice a historie vzniku

Růst objemu dat ve světě přispěl k vytvoření konceptu Data Driving Decision Making, což znamená rozhodování na základě dat, faktů a metrik spíše než na základě intuice. Tento přístup umožňuje přijímat nejlepší rozhodnutí pro firmu. K využití tohoto konceptu je však potřeba mít obrovské množství kvalitních dat. A pokud firmy většinou nemají problémy s objemem dat, tak s kvalitou je to stále složitější. Získat znalosti z dat firmám pomáhá proces, který je známý jako Knowledge Discovery in Databases (KDD).

Knowledge Discovery in Databases, KDD (česky dobývání znalostí z databází) – proces netriviálního objevování implicitních, dopředu neznámých a potenciálně použitelných znalostí v datech. (1)

Tady také stojí za zmínku, že pojmy „data“, „informace“ a „znanost“ nejsou vždy synonyma. Podle cambridgeského slovníku jsou tyto pojmy definovány takto (2):

Data je soubor textu, čísel nebo symbolů v neuspořádané formě a bez významu.

Informace je výsledkem zpracování dat. Vznikají tak fakta, která umožňují zpracovávaná data používat v kontextu a mít význam. Jinými slovy informace jsou data, která mají význam a kontext.

Znanost je získání informací osobou nebo porozumění informacím, například jak řešit problémy.

KDD se skládá z 5 kroků (1):

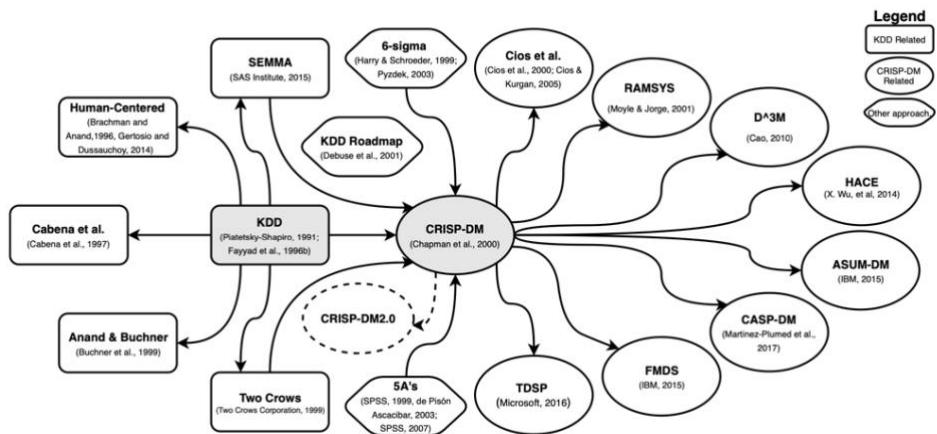
1. Selection (selekce);
2. Preprocessing (předzpracování);
3. Transformation (transformace);
4. Data mining (dolování z dat);
5. Evaluation (Vyhodnocení).

Data mining (česky dolování z dat) – jeden z kroků procesu KDD, zahrnuje aplikaci vybraných analytických metod a algoritmů pro vyhledávání zajímavých vztahů, patternů ve velkých objemech dat. (1)

1.2 Metodiky

KDD proces se stal základem pro vývoj dalších metodik. Daná kapitola představuje nejpopulárnější z nich. Důraz je kladen na metodiku CRISP-DM, která bude použita v praktické části. Ještě před tím, než budou představeny metodiky, je potřeba definovat samotný pojem.

Metodika je popis systematických postupů pro dosažení nějakého cíle. (3)



Obrázek 1: Přehled Data Mining metodik. Zdroj: (4)

1.2.1 SEMMA

SEMMA (angl. zkrátka Sample, Explore, Modify, Model, and Assess) je metodika vyvinutá společností SAS Institute, která vytváří analytický software. Popisuje implementaci data miningu a skládá se z 5 kroků (5):

- 1) Sample – vybírání vhodných objektů;
- 2) Explore – vizuální explorace a redukce dat;
- 3) Modify – seskupování objektů a hodnot atributů, datové transformace;
- 4) Model – analýza dat;
- 5) Assess – porovnání modelů a interpretace.

SEMMA se zaměřuje především na technický aspekt procesu dolování dat a vynechává byznys aspekt. Kromě toho je SEMMA vyvinuta tak, aby pomáhala uživatelům softwaru SAS Enterprise Miner. Proto může být použití SEMMA mimo Enterprise Miner nejednoznačné.

1.2.2 5A

Metodika společnosti IBM SPSS vyvinutá v roce 1999. Název metodiky napovídá, že se skládá z 5 kroků (6):

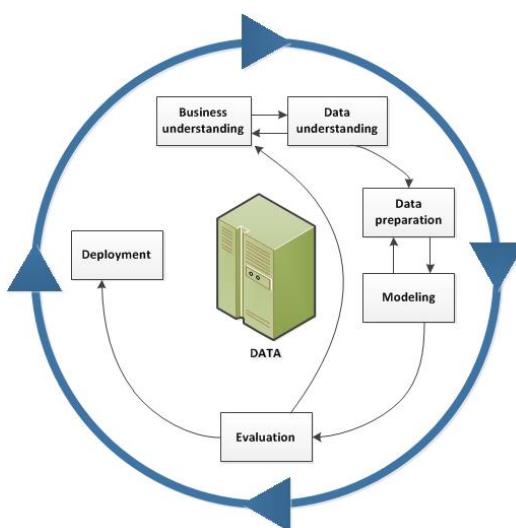
- 1) Assess – posouzení potřeb projektu;
- 2) Access – shromáždění potřebných dat;
- 3) Analyze – provedení analýz;

- 4) Act – přeměna znalostí na akční znalosti;
- 5) Automate – převedení výsledků analýzy do praxe.

1.2.3 CRISP-DM

CRISP-DM (Cross Industry Standard Process for Data Mining) je výsledkem evropského projektu ESPRIT z roku 1996. Autory metodiky jsou 5 firem z různých odvětví: Integral Solutions Ltd, Teradata, Daimler AG, NCR Corporation a OHRA. Na rozdíl od SEMMA CRISP-DM také zahrnuje do procesu byznys aspekt. CRISP-DM stejně jako KDD Process lze považovat za kanonický přístup a standard, ze kterého se vyvinula většina následujících metodik. Metodika předpokládá rozdělení procesu do šesti fází tvořících cyklus (4):

- 1) Business understanding – porozumění doménové oblasti;
- 2) Data understanding – porozumění datům;
- 3) Data preparation – příprava dat;
- 4) Modeling – modelování;
- 5) Evaluation – vyhodnocení modelu;
- 6) Deployment – využití modelu.



Obrázek 2: Cyklus metodiky CRISP-DM. Zdroj: (6)

Business understanding

První a zároveň nejdůležitější fázi CRISP-DM je porozumění doménové oblasti, jelikož se během právě této fáze definuje cíl projektu. Je potřeba identifikovat všechny stakeholdery (tj. zainteresované strany) a jejich požadavky v rámci projektu. Potíž je v tom, že většinou stakeholderi mají různé cíle, požadavky a vidění projektu. Ne všichni vidí stejně věci stejným způsobem a bez jasných cílů nelze v projektu pokračovat. Definování všech požadavků umožňuje určit, jaká data budou použita k vyřešení zadaného problému. (7)

Data understanding

Po fázi porozumění problematice následuje další fáze spojená s porozuměním, ale tentokrát s porozuměním dat. Porozumění datům závisí na porozumění problematice. V této fázi

procesu se shromažďují data. Daná fáze zahrnuje všechny činnosti související s vytvářením datasetu (česky datové sady): sběr požadavků na data, hledání zdrojů, získání dat a samotné porozumění datům. V podstatě určuje, jestli jsou data vhodná pro zadaný problém. Čím více se s problémem a daty pracuje, tím více se učí, a tedy tím více zpřesňování lze v rámci modelu provést, což nakonec vede k lepšímu řešení problému. (7)

Data preparation

Poté co byla data získána, nastává fáze přípravy dat. Spolu s předchozí fází je příprava dat obvykle časově nejnáročnější fází projektu. Příprava dat je proces, jak dostat data do stavu, kdy se s nimi může snadněji pracovat. Ne vždy jsou získaná data ihned připravená k použití. Většinou nejsou v použitelném formátu, to znamená že obsahují chybné nebo chybějící hodnoty, nerelevantní atributy, duplicitní záznamy apod. Fáze přípravy dat má za úkol zbavit se podobných vad a připravit data k modelování. (7)

Mezi kroky předzpracování dat patří například:

- vyplnění chybějících hodnot – mohou být vyplněny jak jednoduchými (střední hodnotou, nejčetnější hodnotou atd.), tak i pokročilejšími způsoby s využitím metod data miningu;
- převod numerických atributů do kategoriálních;
- převod kategoriálních proměnných do tzv. dummies proměnných – dummy proměnná je numerická hodnota, která uvádí, zda záznam spadá do určité kategorie, či ne. Tento převod umožňuje zahrnout kategoriální proměnné do některých modelů;
- standardizace/normalizace dat – rozsah numerických atributů se může velmi lišit a cílem daného kroku je zajistit, aby hodnoty všech numerických atributů byly téměř ve stejném měřítku, aby bylo snazší je zpracovat většinou metod data miningu.

Modeling

Jakmile jsou data připravena k použití, metody data miningu mohou být aplikovány. Je ale důležité si uvědomit, že neexistuje žádný univerzální model pro řešení všech problémů. Fáze modelování se zaměřuje na vývoj modelů, které jsou buď deskriptivní, nebo prediktivní. Ke každému problému je třeba vymyslet vlastní přístup, zkoušet různé modely, srovnat je a vybrat ten nejhodnější. Použití modelů odhaluje znalosti v datech, což je hlavním cílem data miningu. (7)

Evaluation

Fáze modelování a vyhodnocování se provádějí iterativně. Vyhodnocení modelu se provádí během jeho vývoje a před jeho nasazením. Vyhodnocení umožňuje posoudit kvalitu modelu a také zjistit, zda model splňuje původní požadavky. Vyhodnocení modelu zahrnuje výpočet různých metrik a další výstupy, jako jsou tabulky a grafy, jež umožňují interpretovat kvalitu modelu a jeho účinnost při řešení problému. Navíc je možné modelu přiřadit testy statistické významnosti jako další důkaz jeho kvality. (7)

Deployment

Posledním krokem metodiky CRISP-DM je využití modelu. V daném kroku je model použit na nových datech, případně i novými stakeholdery. Nové interakce v této fázi mohou odhalit nové proměnné a potřeby pro datovou sadu a model. To by pak mohlo zahájit revizi buď požadavků stakeholderů, nebo modelu a dat, nebo obojího.

CRISP-DM je vysoko flexibilní a cyklická metodika. Flexibilita je vyžadována v každé fázi spolu s komunikací, aby dosáhl projekt co nejlepších výsledků. V kterékoli ze šesti fází může nastat moment, kdy je nutné se vrátit na předchozí fázi a provést změny. (7)

1.3 Metody data miningu

Tato kapitola popisuje některé existující metody data miningu. Nejprve je definován pojem „metoda“, dále jsou představeny vybrané metody s uvedením příkladů algoritmů. Na konci kapitoly jsou také popsány některé z metrik pro vyhodnocení metod.

Oxfordský slovník velmi stručně popisuje metodu :

„**Metoda** je konkrétní způsob, jak něco udělat.“

Není ani potřeba zvlášť definovat metodu data miningu, jelikož i tak je zřejmé, že tím pojmem se označuje postup čili technika, která se používá k řešení konkrétního problému.

Metody se obecně dělí na *prediktivní* a *deskriptivní*. (8)

1.3.1 Prediktivní metody

Prediktivní metody jsou určeny k předpovídání hodnoty cílové proměnné vytvořením modelu založeného na jednom nebo více prediktorech.

Klasifikace

Klasifikační metody slouží k předpovídání hodnoty kategorické proměnné (cílový atribut) vytvořením modelu založeného na jedné nebo více proměnných (prediktorech). Cílem podobného modelu je vytvořit systém pro automatické rozhodování. (9)

Jeden názorný příklad klasifikačního modelu je z bankovního odvětví: systém, který rozhoduje, zda je klient bonitní na základě různých parametrů (věk, plat, počet dětí apod.). Cílovou proměnnou je v daném případě bonita klienta.

K významným klasifikačním algoritmům patří rozhodovací Decision Trees, K-Nearest Neighbors, Support Vector Machine, Random Forest atd.

Regrese

Regresní metoda předpovídá možné hodnoty chybějících nebo budoucích dat. Liší se od klasifikačních metod především typem výsledku. Výsledkem regresní metody je spojité číselná hodnota, zatímco výsledkem klasifikační metody je kategorie (třída). Stejně jako klasifikační má také regresní metoda cílovou (závislou) proměnnou a proměnnou případně proměnné nezávislé. (10)

Příkladem použití regresní metody může být predikce ceny bytu na základě jeho parametrů (lokace, plocha, počet místností, vzdálenost od metra apod.). Cílovou proměnnou je v daném případě cena bytu.

Existuje více typů regresí, nejpoužívanější z nich jsou lineární a logistická. Logistická regrese je však výjimkou, jelikož je primárně určena k předpovídání kategoriální proměnné. Většinou je jejím cílovým atributem binární hodnota, existuje ale i multinomiální logistická regrese, která umožňuje predikovat proměnnou s více než 2 třídami.

Kromě zmíněných typů regresí existují také Ridge regrese a Lasso regrese, které se počítají s důležitostí nezávislých atributů. (11)

Někdy se regresní metody označují jako predikční metody, ale jiné zdroje vyznačuje rozdíl mezi těmito metodami. Tento rozdíl se spočívá v tom, že predikční metody zahrnují časovou dimenzi. To znamená, že predikční metody slouží k předpovídání opravdu budoucích hodnot, zatímco regresní metody slouží spíš k předpovídání hodnot chybějících. (9), (8), (12)

Analýza časové řady

Časová řada je posloupnost časově uspořádaných hodnot. Jejich analýza zahrnuje metody pro extrahování užitečných vzorů, trendů, pravidel a statistik. Predikce akciového trhu je příkladem aplikací analýzy časových řad.

1.3.2 Deskriptivní metody

Deskriptivní metody popisují nalezené vzory a vztahy v datech, které mohou ovlivnit rozhodování.

Asociační pravidla

Asociační pravidla umožňují z velkého počtu záznamů stanovit pravidla, která odhalují závislost a vztahy mezi různými objekty. Kupříkladu obchodní řetězec má údaje o transakcích zákazníků. Pomocí asociačních pravidel může najít mezi položkami v transakcích takový vztah, že přítomnost jedné nebo více položek v transakci implikuje výskyt jiných položek. Tento problém je dost populární a známý jako analýza nákupního košíku (angl. MBA – Market Basket Analysis).

Shlukování

Také se používají pojmy *segmentace* a *clusterová analýza*. Shlukování se spočívá ve vytváření shluků (skupin) objektů na základě hodnot uvažovaných nezávislých proměnných tak, aby byla zajištěna co nejvyšší podobnost mezi objekty v jednom shluku, při současném zachování maximálních možných rozdílů mezi jednotlivými shluky. Na rozdíl od klasifikační metody shlukování nemá cílovou proměnnou. (9)

Shlukovou analýzu často využívají mobilní operátoři k segmentaci svých klientů, aby pak navrhli určitým skupinám tarif právě podle jejich preferencí. Pro shlukovou analýzu se ve většině případů používá algoritmus K-means (pro numerické atributy) a K-modes (kategoriální atributy). Základním vstupním parametrem pro tyto algoritmy je K, který vyjadřuje do kolika clusterů by původní data měla být rozdělena. Jedním způsobem, jak určit hodnotu K je postupně zvětšovat jeho hodnotu.

Existuje však i více optimální způsob, který je známý jako loketní metoda (angl. Elbow method). Loketní metoda vykresluje hodnotu WCSS (průměrná vzdálenost všech bodů ve shluku k centroidu shluku) funkce vytvořenou různými hodnotami K. Pokud se hodnota K zvýší, hodnota WCSS se sníží, každý shluk bude obsahovat méně jednotlivých bodů a ty body budou blíže k příslušným centroidům. Hodnota K, při které WCSS nejvíce klesá, se nazývá loket, a tato hodnota by měla být optimální. (13)

Detekce anomalií

Detekce anomalií (někdy také detekce outlierů) je proces identifikace datových bodů, které nespadají do běžného chování. Přístupů, kterými lze identifikovat outliersy, je více: statistický přístup (Z-skóre, IQR), vizualizace, shluková analýza apod. (14)

Algoritmů, které se dají použít pro detekci anomalií, je také více, k nejvýznamnějším patří Isolation Forest, K-means, Support Vector Machine a jiné.

1.4 Vyhodnocení modelu

Jak už víme z metodiky CRISP-DM, po kroku s vytvořením modelu následuje krok vyhodnocení modelu, jež je nedílnou součástí procesu vývoje modelu. Pomáhá najít nejlepší model, který reprezentuje data a vykazuje to, jak dobře bude vybraný model fungovat v budoucnu. Vyhodnocení modelů lze rozdělit do dvou skupin: klasifikační a regresní.

1.4.1 Rozdělení dat

Pro vyhodnocení modelu je potřeba rozdělit původní data na trénovací a testovací.

Trénovací dataset je dataset používaný k vytváření modelu.

Testovací dataset je dataset určený pro posouzení kvality modelu.

Existují 4 základní možnosti rozdělení dat (15):

- *resubstituce* (angl. *resubstitution*) – trénovací a testovací dataset obsahují stejná data;
- *náhodný výběr s opakováním* (*bootstrap*) – je založen na n-krát provedeném náhodném výběru subjektů s opakováním z původního datového souboru (s n subjekty), které se použijí jako testovací sada. Subjekty, které nebyly vybrány ani jednou, tvoří testovací dataset;
- *zádrže* (angl. *Hold-Out*) – v této možnosti jsou data náhodně rozdělena na trénovací a testovací dataset v poměru 3 : 1. Někdy se kromě zmíněných datasetů vytváří validační dataset. Ten se používá během optimalizace modelu;
- *křížová validace* (angl. *Cross-Validation*) – při k-násobné křížové validaci se data rozdělují do k podmnožin stejné velikosti, přičemž vždy jedna podmnožina tvoří testovací dataset a zbylých (k-1) podmnožin trénovací dataset. Potom se postup opakuje tak, že každá část je použita pro testování právě jednou.

1.4.2 Klasifikační vyhodnocení

Matice záměn

Matice záměn (angl. Confusion Matrix) ukazuje počet správných a nesprávných predikcí provedených klasifikačním modelem ve srovnání se skutečnými výsledky (cílovou proměnnou) v datech. Matice je rozměru $N \times N$, kde N je počet cílových proměnných (tříd). (17)

Následující tabulka zobrazuje matici záměn rozměru 2×2 , tedy pro dvě třídy (pozitivní a negativní).

Confusion Matrix		Target			
		Positive	Negative		
Model	Positive	a	b	Positive Predictive Value	a/(a+b)
	Negative	c	d	Negative Predictive Value	d/(c+d)
		Sensitivity	Specificity	Accuracy = (a+d)/(a+b+c+d)	
		a/(a+c)	d/(b+d)		

Obrázek 3: Matice záměn. Zdroj: (16)

Matice záměn zahrnuje několik metrik (17):

1. Správnost (angl. accuracy) – kvalita modelu v jednom čísle, podíl z celkového počtu předpovědí, které byly správné.
2. Přesnost (angl. precision) – podíl správně klasifikovaných případů z celkového počtu klasifikovaných. Obvykle se počítá pro jednotlivé třídy:
 - Pozitivní prediktivní hodnota (angl. positive predictive value) – podíl správně klasifikovaných pozitivních případů z celkového počtu klasifikovaných případů.

- Negativní prediktivní hodnota (angl. negative predictive value) – podíl správně klasifikovaných negativních případů z celkového počtu klasifikovaných případů.
3. Úplnost (angl. recall) – podíl správně klasifikovaných instancí ze všech s danou třídou v testovacích datech. Také se počítá pro jednotlivé třídy:
- Sensitivita (angl. sensitivity) – podíl skutečných pozitivních případů, které byly správně identifikovány.
 - Specifičnost (angl. specificity) – podíl skutečných negativních případů, které byly správně identifikovány.

F1 skóre

F1 skóre (angl. F1 score) je metrika pro klasifikační modely, která je v podstatě zlepšenou verzí dvou jednodušších metrik: přesnosti a úplnosti. F1 skóre bylo navrženo tak, aby dobře fungovalo na nevyvážených datech, tj. na datech se zkreslenými proporcemi tříd. (18)

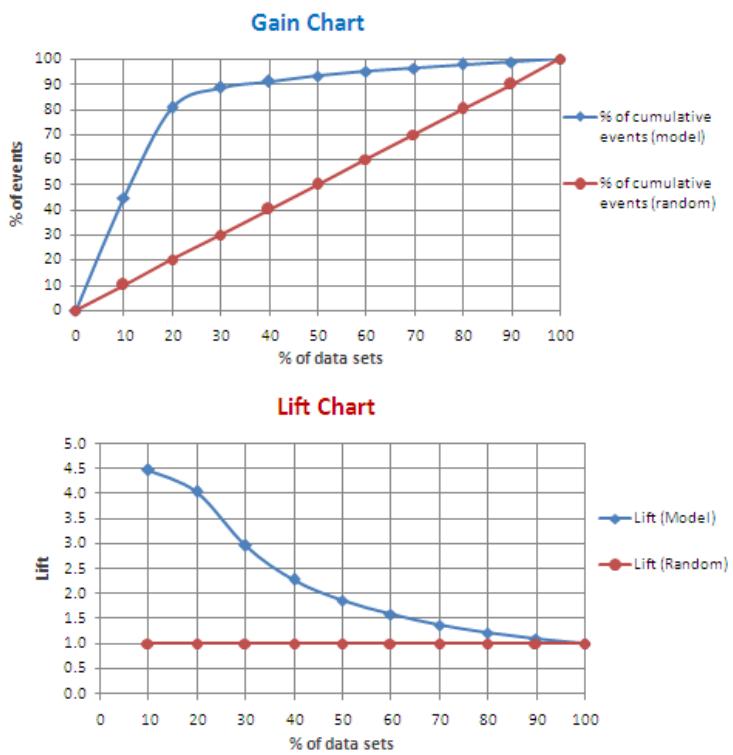
$$F1 \text{ score} = 2 \times \frac{\text{Přesnost} \times \text{Úplnost}}{\text{Přesnost} + \text{Úplnost}}$$

Gain křivka

Gain křivka je vizuální pomůckou pro hodnocení výkonnosti klasifikačních modelů. Na rozdíl od matic záměn, která vyhodnocuje modely na základě celého datasetu, Gain křivka vyhodnocuje výkonnost modelu v části datasetu. (16)

Lift křivka

Lift křivka měří účinnost modelů výpočtem poměru mezi výsledkem získaným s modelem a výsledkem získaným bez modelu. Výsledek získaný bez modelu je založen na náhodně vybraných záznamech. V grafu je znázorněn náhodnou křivkou. (16)

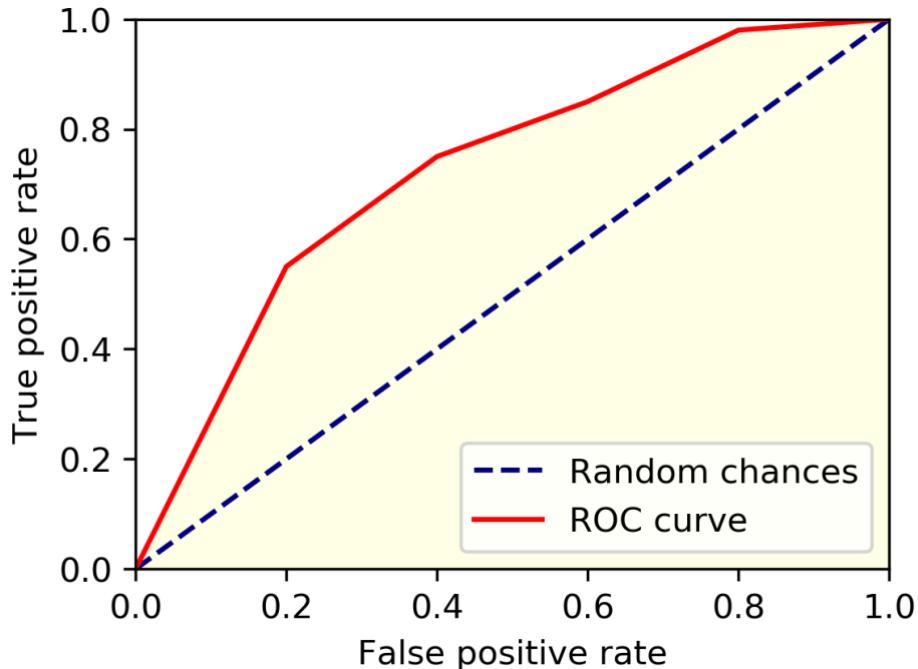


Obrázek 4: Gain a Lift křivka. Zdroj: (19)

ROC křivka

Křivka ROC (angl. receiver operating characteristic) ukazuje míru falešně pozitivních výsledků (specificita) na ose X, oproti skutečné pozitivní míře (sensitivita) na ose Y. V ideálním případě bude křivka rychle stoupat směrem k levému hornímu rohu, což znamená, že model správně předpověděl případy. Diagonální tečkovaná čára je pro náhodný model.

Plocha pod křivkou AUC (angl. area under the curve) se často používá jako měřítko kvality klasifikačních modelů. Náhodný model má plochu pod křivkou 0,5, zatímco plocha pro dokonalý model je rovna 1. V praxi má většina klasifikačních modelů AUC mezi 0,5 a 1. (20)



Obrázek 5: ROC křivka. Zdroj: (21)

1.4.3 Regresní vyhodnocení

RMSE

RMSE (angl. Root Mean Square Error) je populární vzorec pro měření chybovosti regresního modelu. Lze jej však porovnávat pouze mezi modely, jejichž chyby jsou měřeny ve stejných jednotkách.

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (\hat{y}_i - y_i)^2}{n}}$$

kde \hat{y}_i – predikovaná hodnota, y_i – skutečná hodnota, n – celkový počet pozorování.

Čím nižší je hodnota RMSE, tím lepší je model se svými predikcemi. Vyšší RMSE znamená, že existují velké odchylky mezi předpokládanou a skutečnou hodnotou. (22)

RSE

Na rozdíl od RMSE lze relativní čtvercovou chybu RSE (angl. Relative Squared Error) porovnávat mezi modely, jejichž hodnoty jsou měřeny v různých jednotkách. (22)

$$RSE = \frac{\sum_{i=1}^n (\hat{y}_i - y_i)^2}{\sum_{i=1}^n (\bar{y}_i - y_i)^2}$$

kde \hat{y}_i – predikovaná hodnota, y_i – skutečná hodnota, \bar{y}_i – střední hodnota, n – celkový počet pozorování.

Pokud je hodnota RSE větší než 1, pak to znamená, že vytvořený model není ani tak dobrý jako model, který využívá průměr jako predikce pro každé pozorování.

MAE

MAE (angl. Mean Absolute Error) je podobný RMSE v tom smyslu, že se i zde používá rozdíl mezi predikovanou a skutečnou hodnotou, který je následně vydělen počtem hodnot, avšak namísto umocnění tohoto rozdílu se bere jeho absolutní hodnota. (22)

$$MAE = \frac{\sum_{i=1}^n |\hat{y}_i - y_i|}{n}$$

kde \hat{y}_i – predikovaná hodnota, y_i – skutečná hodnota, n – celkový počet pozorování.

Díky použití absolutních hodnot se MAE dokáže vypořádat s tzv. *outliery*¹ lépe než RMSE.

RAE

Stejně jako RSE lze relativní absolutní chybu (RAE) porovnávat mezi modely, jejichž hodnoty jsou měřeny v různých jednotkách.

$$RAE = \frac{\sum_{i=1}^n |\hat{y}_i - y_i|}{\sum_{i=1}^n |\bar{y}_i - y_i|}$$

kde \hat{y}_i – predikovaná hodnota, y_i – skutečná hodnota, \bar{y}_i – střední hodnota, n – celkový počet pozorování.

Koefficient determinace

Koefficient determinace je nejdůležitější metodou hodnocení regresního modelu a označuje se jako R^2 . R^2 popisuje podíl rozptylu závislé proměnné vysvětlený regresním modelem. Jednoduše řečeno, R^2 vyjadřuje, kolik variací nezávislé proměnné je vysvětleno závislými proměnnými. (22)

Koefficient determinace se vypočítá ze součtu čtverců:

$$Koefficient\ determinace\ (R^2) \rightarrow R^2 = \frac{SSR}{SST} = 1 - \frac{SSE}{SST}$$

$$Celkový\ součet\ čtverců\ (SST) \rightarrow SST = \sum_{i=1}^n (y_i - \bar{y}_i)^2$$

¹ Outlier (česky odlehlá, extremní) – hodnota, která se výrazně liší od ostatních

$$\text{Regresní součet čtverců (SSR)} \rightarrow SSR = \sum_{i=1}^n (\hat{y}_i - \bar{y}_i)^2$$

$$\text{Zbytkový součet čtverců (SSE)} \rightarrow SSE = \sum_{i=1}^n (\hat{y}_i - y_i)^2$$

kde \hat{y}_i – predikovaná hodnota, y_i – skutečná hodnota, \bar{y}_i – střední hodnota, n – celkový počet pozorování.

R^2 může nabývat hodnot od 0 od 1. Pokud je regresní model „dokonalý“, SSE je nula a R^2 je 1. Pokud regresní model není vůbec vhodný, SSE se rovná SST, regresí není vysvětlena žádná odchylka a R^2 je nula.

2 Web scraping

Tato kapitola je věnována technologii web scrapingu. Nejprve jsou definovány pojmy a krátce popsány jeho principy. Následně je představena základní struktura webové stránky. Dále jsou prezentována existující řešení a nakonec je web scraping posuzován z hlediska etiky.

2.1 Definice a principy

Web scraping je proces extrakce dat z webových stránek a jejich uložení do souborového systému nebo databáze pro pozdější vyhledání nebo analýzu. Web scraping může být proveden ručně, avšak tento pojem se většinou používá, když jde o automatizovanou extrakci dat. (23)

Proces stahování dat z internetu lze rozdělit do dvou po sobě jdoucích kroků:

1. získání webových zdrojů (zdrojů webové stránky),
2. extrahování požadovaných informací ze získaných zdrojů.

Konkrétně proces web scrapingu začíná odesíláním požadavku HTTP na získání zdrojů cílené webové stránky. Tento požadavek může být naformátován buď jako adresa URL obsahující dotaz GET, nebo jako část zprávy HTTP obsahující dotaz POST.

Jakmile je požadavek úspěšně přijat a zpracován cílenou webovou stránkou, požadovaný zdroj bude z webové stránky načten a poté odeslán zpět. Zdroj může být v několika formátech, například ve formátu HTML, XML nebo JSON. Zdroj může také obsahovat multimediální data, jako jsou obrázky, audio nebo video soubory.

Krátce řečeno, web scraping transformuje nestrukturovaná data z webových stránek v strukturovaná.

Po získání webových zdrojů proces extrakce pokračuje v analýze, přeformátování a uspořádání dat strukturovaným způsobem. Důležitou součástí web scraperu jsou datové lokátory (neboli selektory), které slouží k vyhledání dat, která je potřeba z HTML souboru extrahovat – obvykle se používá XPath, selektory CSS, regulární výrazy nebo jejich kombinace.

Zjednodušený proces web scrapingu lze popsat následujícími kroky:

1. Identifikace cílové webové stránky;
2. Odesílání požadavku na URL adresu cílové webové stránky;
3. Získání HTML kódu stránky;
4. Vyhledání informace v HTML kódu;
5. Uložení dat do strukturovaného formátu (JSON, CSV apod.).

2.2 Struktura webové stránky

Většinou je web scraping založen na analýze struktury webové stránky, a proto je důležité pochopit ten princip. Daná subkapitola představí stručný přehled o technologiích, které se používají během web scrapingu.

2.2.1 XML

Existují tzv. značkovací jazyky, které pomocí speciálních značek vysvětlují význam (sémantiku) různých částí textu nebo určují vzhled (formát) jednotlivých částí textu. K nejvýznamnějším značkovacím jazykům patří HTML a XML.

XML (angl. zkrátka Extensible Markup Language) je jednoduchý textový formát pro reprezentaci strukturovaných informací: dokumenty, data, konfigurace, knihy, transakce, faktury a mnoho dalšího. XML byl odvozen ze staršího standardního formátu SGML, aby byl vhodnější pro použití na webu. (24)

I když jsou XML a HTML značkovací jazyky, mají i několik rozdílů. XML byl navržen tak, aby přenášel data, tedy se zaměřením na to, co data jsou, zatímco HTML je více zaměřen na to, jak data vypadají. XML je více přísný jazyk, to znamená, že pokud v XML dokumentu bude chyba, tak se prostě nespustí. Tak například všechny tagy v XML dokumentu musí být vždy uzavřeny, ale na druhou stranu může XML dokument obsahovat jakékoli názvy tagů, což v HTML dokumentech není možné – tam jsou tagy předdefinovány.

I když byl XML vyvinut později než HTML (první verze v roce 1998), lze říct, že XML je metajazyk, tj. jazyk, který se používá pro popis jiných jazyků. Čili i v současné době je XML základem pro nové webové technologie. (25)

2.2.2 XPath

Z předchozích odstavců lze shrnout, že HTML a XML dokumenty používané pro ukládání strukturovaných dat, případně pro tvorbu vizuálního pohledu webových stránek. Ale tyto formáty definují obsah celé webové stránky, zatímco jedním z kroků web scrapingu je vyhledání konkrétní informace ve zdrojovém kódu. Pro tento krok lze použít XPath.

XPath (angl. zkrátka XML Path Language) je dotazovací jazyk, který je užitečný pro identifikaci a extrahování částí z dokumentů HTML/XML. (26)

Každý HTML či XML dokument si lze představit jako strom. XPath umožňuje vyhledávání v podobných dokumentech pomocí dotazu. Výsledkem dotazu je uzel (angl. node), případně sada uzlů. Uzly v HTML/XML mohou být i tagy.

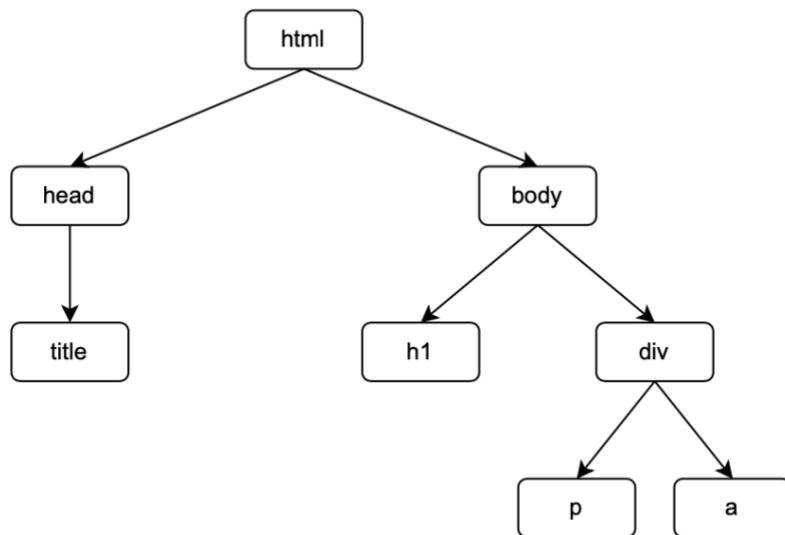
```
1 <html>
2   <head>
2     <title>My page</title>
3   </head>
4   <body>
```

```

5      <h1>This is my first page</h1>
6      <div class="wrapper">
7          <p>I'm studying at FIS</p>
8          <a href="https://fis.vse.cz">FIS website</a>
9      </div>
10     </body>
11  </html>

```

Z výše uvedené ukázky HTML dokumentu lze vytvořit následující strom.



Obrázek 6: Zjednodušená ukázka stromu HTML dokumentu (autor)

Podobný strom připomíná genealogický strom: stejně je z něj zřejmá viditelná dědičnost jednotlivých tagů. Tag `<html>` je kořenovým tagem a má 2 děti: `<head>` a `<body>`. Tag `<head>` zase má vlastní dítě – tag `<title>`, který je zároveň vnoučetem tagu `<html>` čili jeho potomkem. Stejně tak i tag `<body>` má 2 děti a pak 2 vnoučata. Tag `<div>` je rodičem tagů `<p>` a `<a>` a zároveň dítětem tagu `<body>`. Tagy `<head>` a `<body>` jsou navíc sourozenci, jelikož se nachází na stejném úrovni a mají společného rodiče.

Dotaz XPath v podstatě hledá, kde se nachází ten, nebo jiný tag. Dělá to pomocí buď absolutní, nebo relativní cesty. Absolutní cesta vždy projde začátkem dokumentu a skončí až u hledaného tagu, zatímco relativní cesta začíná u aktuálního tagu. Jednotlivé tagy v XPath dotazu se rozdělují buď lomítkem („/“), anebo dvojitým lomítkem („//“). Dvojité lomítko vyjadřuje, že hledá kdykoliv v dokumentu. Relativní cesta se v XPath značí jako „./“.

Dalšími důležitými prvky XPath dotazu jsou atributy tagů. V XPath se značí pomocí zavináče (@). Symbol hvězdičky (*) v Xpath vyjadřuje libovolný element. XPath také umožňuje zadávat podmínky do dotazu. Podmínky se zapisují do hranatých závorek. Příkladem XPath dotazu s podmínkou je:

```
//div[@class="wrapper"]/a/@href
```

Tento dotaz lze napsat i slovně: „najdi kdekoli v souboru tag <div>, který má atribut class s hodnotou ‚wrapper‘, pak najdi tag <a>, který je jeho dítětem, a nakonec vypiš atribut href(odkaz) u nalezeného tagu <a>“.

Tag <p> ze stromu lze pomocí XPath najít takto:

- /html/body/div/p (absolutní cesta)

2.3 Existující řešení

S příchodem popularity disciplín, jako je datová analytika, ve které je sběr dat nezbytným krokem, se také rychle rozvinul i web scraping. V dnešní době existuje mnoho nástrojů pro sběr dat z internetu. Mezi hlavní nástroje patří API, cloudová řešení (Software as a Service) a knihovny pro programovací jazyky.

2.3.1 API

API (angl. Application Programming Interface) je sada definovaných pravidel, která vysvětlují, jak spolu počítače nebo aplikace komunikují. Rozhraní API jsou umístěna mezi aplikací a webovým serverem a fungují jako zprostředkovatelská vrstva, která zpracovává přenos dat mezi systémy. (27)

Velké firmy, kupříkladu Amazon, Linkedin, eBay, mají vlastní API pro získání dat z jejich stránek. API lze z většího pohledu rozdělit do 4 typů (28):

1. Veřejné – API, které může využívat kdokoliv. Zatímco některá veřejná API jsou bezplatná, jiná vyžadují předplatné za použití.
2. Partnerské – API sdílené externě, ale pouze mezi těmi, kteří mají obchodní vztah se společností poskytující daný API.
3. Privátní (interní) – API, které jsou k dispozici pouze pro použití uvnitř organizace a je určeno k efektivnímu přenosu dat mezi týmy a systémy.
4. Kompozitní – API, které kombinuje více typů API a umožňuje získávat data z různých aplikací či zdrojů.

Před použitím nástrojů pro web scraping se vždy vyplatí zjistit, zda má web vlastní API, jelikož tento způsob získání dat je nejjednodušší, navíc se dá říct, že je legální a k tomu ještě nebude zbytečně zatěžovat samotný web.

2.3.2 Cloudová řešení

Cloudová řešení budou pro většinu problémů tím nejsnadnějším, neboť není potřeba znát žádný programovací jazyk. K dodavatelům podobných řešení patří společnosti Zyte, Octoparse, Apify a jiné. Tyto firmy nabízí předplatné a v podstatě působí jako SaaS (angl.

Software as a Service). Navíc mají i bezplatnou verzi, avšak pro potřeby velkého projektu asi stačit nebude.

Funkcionalita většiny SaaS nástrojů pro web scraping je relativně rozsáhlá. Mezi výhody podobných nástrojů patří mimo jiné možnost extrakce dynamického obsahu, možnost nastavit automatickou extrakci podle rozvrhu a automatická změna IP, aby se zabránilo blokování ze strany webu. Další důležitou výhodou využívání SaaS nástrojů je, že není třeba mít znalosti z programování.

2.3.3 Knihovny pro Python

Pro rozšíření možností jazyka se v programovacích jazycích používají knihovny. Knihovna je soubor funkcí, tříd, metod atd. shromážděných na jednom místě, které jsou pak využívány jinými programy. Vzhledem k tomu, že v praktické části této bakalářské práce bude použit Python, budou dále zvažovány web scraping knihovny pro tento jazyk.

Requests ²

Jak již bylo zmíněno na začátku dané kapitoly, jedním z prvních kroků web scrapingu je odesílání požadavku na webovou stránku s cílem získat zdrojový kód. Requests je jednoduchá knihovna, která umožňuje odesílat HTTP požadavky pomocí Pythonu. HTTP požadavek pak vrátí objekt odpovědi se všemi daty (obsah, kódování, stav atd.). Není to knihovna právě pro web scraping, ale přestavuje pro něj osnovu.

Beautiful Soup ³

Beautiful Soup je jedním z nejjednodušších nástrojů pro web scrapingu v Pythonu. Tato knihovna vyvinutá v roce 2004 poskytuje několik jednoduchých metod pro vyhledávání a extrafování potřebných dat. Občas funkcionality této knihovny bohatě stačí k vyřešení problému, a zároveň výsledný skript nebude obsahovat mnoho kódů.

Zde však stojí za zmínku, že moderní webové stránky lze rozdělit na 2 typy: stránky se statickým obsahem a stránky s dynamickým obsahem. S posledním typem stránek se Beautiful Soup nevypořádá. To znamená, že pokud webová stránka obsahuje JavaScript nebo JQuery elementy, Beautiful Soup prostě nedokáže vyexportovat obsah uvnitř ně.

Na druhou stranu má Beautiful Soup výhodu oproti jiným nástrojům, a tou je jeho schopnost automaticky detekovat kódování, což umožňuje zpracovávat HTML dokumenty se speciálními znaky. Umí tak převádět příchozí dokumenty na Unicode a odchozí dokumenty na UTF-8.

² <https://docs.python-requests.org/en/latest/>

³ <https://www.crummy.com/software/BeautifulSoup/bs4/doc>

Selenium⁴

Selenium je původně automatizovaný testovací rámec používaný k ověřování webových aplikací napříč různými prohlížeči a platformami. Selenium umožňuje automatizovat webové prohlížeče a má knihovny pro různé programovací jazyky, včetně Python.

Selenium používá WebDriver k ovládání webových prohlížečů, jako jsou Chrome, Firefox nebo Safari. Postupem času se však začala tato knihovna využívat nejen k testování aplikací, ale k web scrapingu, a to díky své funkcionality a kompatibilitě s JavaScriptem.

Selenium je užitečný, když je potřeba provést nějakou akci na webu, např.:

- Vyplňování polí nebo formulářů;
- Rolování stránky;
- Kliknutí na tlačítka;
- Pořízení snímku obrazovky.

Další výhodou Selenia je možnost fungování s JavaScriptem. Umí například načíst obsah vnořený do prvků JavaScriptu.

Selenium také podporuje tzv. „headless“ prohlížeče, což jsou prohlížeče bez GUI, které se spouští v příkazové řádce. K výhodám podobných prohlížečů patří větší rychlosť a menší spotřeba paměti.

Scrapy⁵

Scrapy je open source a kolaborativní rámec pro extrahování dat z webových stránek rychlým, jednoduchým, a přitom rozšířitelným způsobem (Scrapy.org).

V podstatě jde o nejkomplexnější řešení pro web scraping, které poskytuje nástroje k procházení webových stránek, stahování dat, jejich analýze a ukládání. Scrapy podporuje rozšíření, což přináší možnost přidání proxy, zpracování souborů s cookies a ovládání hloubky procházení.

Dalším rysem Scrapy je jeho asynchronní způsob zpracování požadavků. To umožňuje extrahovat data rychle i z více stránek najednou. Je však zřejmé, že jelikož tento nástroj umožňuje více, je také obtížnější jej nastavit.

2.4 Problémy

Web scraping je velice populární způsob získávání dat, avšak má i vlastní slabé stránky čili problémy. Některé weby se snaží zbavit web scraperů a používají tak různé techniky, jak se

⁴ <https://www.selenium.dev>

⁵ <https://scrapy.org>

vypořádat s nežádoucím scrapováním webu. Moderní řešení umožňují takové nástroje detekovat a blokovat.

2.4.1 CAPTCHA

CAPTCHA (angl. zkrátka Completely Automated Public Turing test to tell Computers and Humans Apart) se často používá k oddělení lidí od web scraperů zobrazením obrázků nebo logických problémů, které jsou lidé schopni snadno vyřešit, ale scrapery ne. (29)

Ačkoli některé scrapery mají možnost překonat podobné problémy, CAPTCHA stále může trochu zpomalit proces scrapingu.

2.4.2 Honeypot

Honeypot je past, kterou majitel webu umístí na stránku, aby chytil scrapery. Pasti mohou být například odkazy, které jsou neviditelné pro lidi, ale viditelné pro scrapery. Jakmile se scraper dostane do pasti, webová stránka může použít informaci, kterou obdrží (jako např. IP adresa), k zablokování web scraperu. (29)

Používání XPath k přesné lokalizaci elementů webové stránky, na které lze kliknout nebo extrahat, do značné míry snižuje šance pádu do pasti.

2.4.3 Dynamický obsah

Dynamický obsah je prvek webové stránky, který se mění podle údajů a chování uživatele. Většinou jej používají značky k zobrazování přizpůsobeného obsahu na základě historických dat uživatelů a vyhledávacích dotazů. Netflix například sleduje preference uživatelů a čas strávený na obrazovce, aby mohl vytvářet personalizovaná doporučení podle jejich chování. Dynamický obsah představuje výzvu pro roboty pro procházení webu, které jsou naprogramovány tak, aby exportovaly statické prvky HTML, avšak robot pro procházení může být naprogramován tak, aby posouval stránku dolů, našel cílová data a extrahal, co je potřeba. (29)

Mnoho webových stránek používá AJAX k aktualizaci dynamického webového obsahu. Příklady jsou líné načítání obrázků, nekonečné posouvání a zobrazení dalších informací kliknutím na tlačítko prostřednictvím volání AJAX. Pro uživatele je pohodlné prohlížet si více dat na takovýchto webech, ale ne pro scrapery.

2.4.4 Změny struktury webový stránek

Někdy je potřeba exportovat data z několika webů, nikoli z jednoho, a občas se dá obejít i jedním scraperem, a to například používáním regulárních výrazů pro identifikaci elementů. Avšak různé weby mají většinou různou strukturu, tedy pro každý web musí být vytvořen samostatný scraper, jindy i pro každou webovou stránku v rámci jednoho webu.

To se stává například kvůli tomu, že tvůrci webu používají různé id nebo třídy u HTML tagů. Kromě toho pokročilejší webové stránky používají nesmysluplné id, aby nebylo možné nastavit regulární výraz na zachycení elementů.

Tvůrci webových stránek navíc pravidelně aktualizují svůj obsah, aby zlepšili uživatelskou zkušenosť nebo přidali nové funkce, což často vede ke strukturálním změnám na webové stránce. Vzhledem k tomu, že web scrapery jsou nastaveny podle určité struktury webové stránky, nebudou fungovat pro aktualizované stránky. Někdy i malá změna na cílovém webu vyžaduje úpravu web scraperu, aby odpovídalo novým změnám.

Jedna věc, která s tím může částečně pomoci, je použití identifikace elementů podle atributu „id“ místo „třídy“, jelikož „id“ by měl být jedinečným v rámci jedné stránky, zatímco stejných tříd může být několik.

2.4.5 Mnoho častých HTTP požadavků

Webové stránky mohou odpovídat pomalu, nebo se dokonce nemusí načítat, když obdrží příliš mnoho požadavků na přístup. To není problém, když lidé procházejí web, protože stačí znova načíst webovou stránku a počkat, až se web obnoví. Ale web scraper se tím může rozbít, protože neví, jak se s takovou situací vypořádat.

Řešením může být umělé zpomalení, aby požadavky vypadaly spíše jako běžný uživatel nebo nastavení automatického obnovení stránky za určitých podmínek.

2.4.6 IP blokování

Jedním z hlavních rizik web scrapingu je blokování IP adres, což je běžná metoda, jak zabránit web scraperům v přístupu k datům webu. Obvykle k tomu dochází, když web detekuje vysoký počet požadavků ze stejné IP adresy, přičemž je docela zřejmé, že by to lidé nedokázali udělat. Webová stránka v takovém případě buď zakáže IP zcela, nebo omezí její přístup. (29)

Používání proxy či změna IP adres by mohly vést k vyřešení tohoto problému, aby web scraper pokaždé posílal požadavky na přístup z jiné IP adresy.

2.5 Etický aspekt

Web scraping je extrakce dat z veřejných webových stránek, ale občas tyto stránky obsahují data, která patří právě jejich vlastníkům, na základě čehož vznikla diskuse, zda je web scraping legální. Stojí za to říct, že na tuto otázku stále neexistuje definitivní odpověď. Na jedné straně jsou údaje zveřejněny na stránkách veřejně dostupné, každý je může najít.

Na druhou stranu to, že jsou tato data zveřejněna na internetu, neznamená, že je může kdokoli používat. To platí zejména pro citlivá data či osobní údaje, které podléhají GDPR.

2.5.1 Web scraping a GDPR

GDPR (angl. General Data Protection Regulation) je právní rámec ochrany osobních údajů v evropském prostoru s cílem hájit co nejvíce práva občanů EU proti neoprávněnému zacházení s jejich daty včetně osobních údajů. (30)

Jak již napovídá definice, GDPR určuje právní omezení zejména pro osobní údaje. Dle Článku 4 GDPR osobními údaji jsou (31):

„veškeré informace o identifikované nebo identifikovatelné fyzické osobě (dále jen „subjekt údajů“); identifikovatelnou fyzickou osobou je fyzická osoba, kterou lze přímo či nepřímo identifikovat, zejména odkazem na určitý identifikátor, například jméno, identifikační číslo, lokaci údaje, síťový identifikátor nebo na jeden či více zvláštních prvků fyzické, fyziologické, genetické, psychické, ekonomické, kulturní nebo společenské identity této fyzické osoby.“

V rámci této práce dojde k získání dat o nemovitostech. Z výše uvedených typů osobních údajů mohou být omezeny adresy objektů. Na základě téhož článku č. 4 jsou však osobními údaji takové údaje, jejichž prostřednictvím lze identifikovat konkrétní osobu, což je problematické udělat pouze s adresou objektu. Z hlediska GDPR je tedy web scraping veřejných realitních dat legální.

2.5.2 Robots.txt

Před samotným web scrapingem stojí za malý průzkum cílové webové stránky. Občas vlastníci webových stránek píšou o tom, zda lze kopírovat obsah jejich webu přímo na stránkách, a to zpravidla dole, v tzv. footeru stránky. Pokud tam není nic nenapsáno, dalším vhodným krokem bude otevřít soubor robots.txt, který lze najít na adrese webova_stranka.cz/robots.txt.

Robots.txt je textový soubor, který webmasteři vytvářejí, aby dali pokyn webovým robotům (obvykle robotům vyhledávačů), jak procházet stránky na jejich webu. Soubor robots.txt dále udává, zda určití uživatelští agenti (software pro procházení webu) mohou, nebo nemohou procházet části webových stránek. Tyto pokyny pro procházení jsou specifikovány slovy „disallow“ nebo „allow“. Základní formát pokynů v souboru robots.txt je:

User-agent: [název user-agentu] Disallow: [URL stránky, která se nemá procházet]

3 Získání dat a analýza

Tato kapitola má za cíl ukázat aplikaci teoretických znalostí z předchozí kapitoly na reálném problému. Začne se krátkým představením použitých nástrojů, další dílčí subkapitoly budou navázány na metodiku CRISP-DM, popsanou v teoretické části. To znamená, že se postupně přejde od porozumění doménové oblasti a datům přes přípravu dat a modelování až po vyhodnocení modelů i jejich využití s tím, že proces získání dat bude popsán mezi fázemi porozumění doménové oblasti a porozumění datům.

3.1 Použité nástroje

3.1.1 Python a knihovny

Základním nástrojem pro zpracování praktické části dané práce je programovací jazyk Python, jenž byl vyvinut v roce 1991 a v současné době se považuje za nejpoužívanější jazyk pro práce s daty. Jedním z důvodů, proč je to tak, jsou různé knihovny rozšiřující jeho funkcionalitu.

V rámci práce se používaly zejména knihovny:

- Pandas – manipulace s daty;
- Numpy – práce s vektory a maticemi;
- Matplotlib / Seaborn – vizualizace dat;
- Selenium – scraping;
- Scikit-learn – modelování.

3.1.2 Jupyter Notebook

Jupyter Notebook je webová aplikace pro vytvoření a sdílení výpočetních dokumentů. Kromě Pythonu podporuje také R, Java, Julia, Matlab a další programovací jazyky.

V rámci své práce jsem používal Jupyter Notebook pro předzpracování dat a tvorbu modelů.

3.1.3 Tableau

Tableau je nástroj pro vizualizaci dat, který se hodně používá v oblasti Business Intelligence. Umožňuje vytvářet interaktivní dashboardy a různé reporty. Tento nástroj jsem využil pro vizualizaci získaných dat.

3.1.4 Github

Github je internetový nástroj pro uložení a správu verzí zdrojového kódu. Pro danou práci byl vytvořen repositář, kde je uložena finální verze zdrojového kódu vytvořeného během zpracování praktické části spolu s tzv. community (uložené změny).

3.2 Porozumění doménové oblasti

Jak je zřejmé z názvu dané práce, samotná analýza bude probíhat na realitních datech. S pojmem „realitní“ se ale dá pokrýt obrovská oblast. K nemovitostem patří kupříkladu byty, garáže, pozemky apod. Pokud je nemovitost určena především k podnikání (restaurace, obchody, hotely atd.), nazývá se komerční.

Tato práce se bude zabývat pouze nekomerčními nemovitostmi, jelikož nabídek komerčních nemovitostí je relativně málo, navíc nejsou tak zajímavé pro obyčejné lidi z pohledu analýzy.

Dále oblast analýzy byla omezena pouze na byty, takže nabídky jiných nemovitostí nebyly brány v ohled. Kromě toho celý realitní trh lze rozdělit na dvě části podle typu nabídek. První část zahrnuje byty určené k prodeji, zatímco ta druhá představuje byty k pronájmu.

3.3 Získání dat

Realitní data se často používají pro analýzu, a proto již existuje hodně datasetů z této oblasti. Většinou se ale tato data týkají větších zemí (USA, Německo, Francie, Rusko atd.), navíc nejsou vždy aktuální.

Nabídky nemovitostí se nejčastěji dají najít na webových stránkách realitních kanceláří anebo na webových stránkách zprostředkovatelů. Stránky jednotlivých kanceláří obsahují menší počet nabídek, to znamená, že proces sběru dat by byl složitější.

Oproti stránkám kanceláří zprostředkovatele mají také tu výhodu, že se informace o nemovitosti ukládají strukturovaně čili všechny nabídky na těchto stránkách mají povinnou strukturu. K nejvýznamnějším zprostředkovatelům v České republice patří Sreality.cz a Bezrealitky.cz. Na webu Sreality ve footeru stránky se však nachází hláška:

„Jakékoli užití obsahu internetového serveru www.sreality.cz, včetně převzetí, šíření či dalšího zpřístupňování inzerátů a fotografií je bez souhlasu Seznam.cz, a.s., zakázáno.“

Dále jsem kontaktoval technickou podporu s tím, jestli můžu vyexportovat inzeráty pro akademické účely a dostal jsem odpověď: „... data však použít nelze, jelikož uvedené údaje nepatří nám, ale zadavatelům inzerce ...“ Tím pádem jsem musel web Sreality bohužel vynechat.

Bezrealitky.cz nemá na svých stránkách žádné hlášky o převzetí materiálu, nemá je tak ani ve smluvních podmínkách, takže jsem se soustředil na tento web.

3.3.1 Robots.txt

Soubor robots.txt webu bezrealitky.cz je na obrázku níže. Majitele webu zakazují robotům procházet stránkami, které ve svém URL obsahují speciální marketingové parametry, určené ke sledování efektivity marketingových kampaní. Jednotlivé inzeráty či stránky s výpisem všech nabídek mají ovšem jinou strukturu URL, neobsahující žádný podobný parametr, tedy dle souboru majitele nezakazují robotům procházení inzeráty.

```
# www.robotstxt.org/
# www.google.com/support/webmasters/bin/answer.py?hl=en&answer=156449

User-agent: *
Disallow: /*?utm_*
Disallow: /*?reload=*
Disallow: /*?kampan=*
Disallow: /*?source=*

Sitemap: https://www.bezrealitky.cz/sitemap/sitemap.xml
```

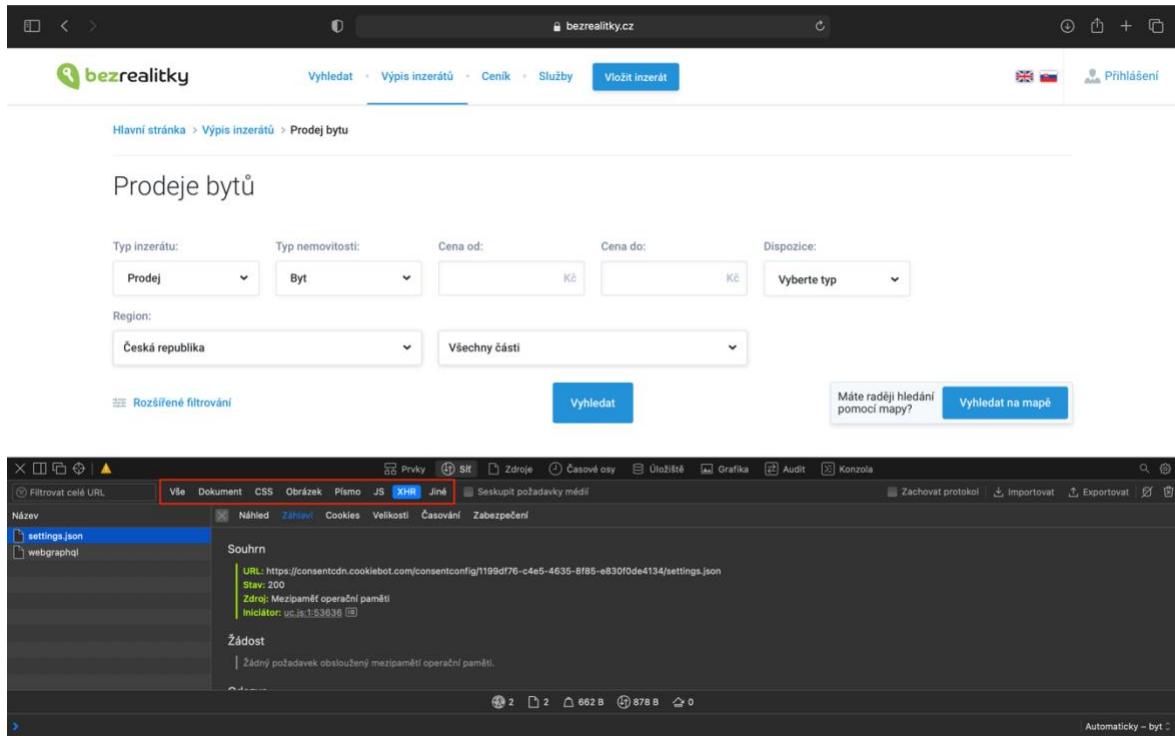
Obrázek 7: Soubor robots.txt pro webovou stránku Bezrealitky (autor)

3.3.2 Hledání API

Jak již bylo zmíněno v teoretické části, vhodným krokem by bylo zjistit, zda web nemá vlastní API. Hledáním ve vyhledávačích dotazu s klíčovými slovy „bezrealitky“ a „api“ se žádná veřejná API nenašla.

Výpis všech inzerátů lze najít na stránce „<https://www.bezrealitky.cz/vypis/nabidka-prodej/byt>“, kde se také dá zadat filtrování. Pomocí nástrojů vývojáře ve webovém prohlížeči se při načtení stránky dá zachytit soubory předávané přes síť. Z výsledků se dají vyfiltrovat pouze soubory typu XHR⁶, kde však žádné užitečné soubory také nejsou.

⁶ XHR (angl. zkrátka XMLHttpRequest) – rozhraní pro komunikaci mezi serverem a klientem, umožňující získání dat z URL, aniž by stránka měla být načtena znovu. (32)



Obrázek 8: Hledání API v přenášených souborech (autor)

Další možností bylo vyhledání API přímo v souborech, což se nakonec ukázalo jako úspěšný pokus. Po zadání dotazu „api/“ do vyhledávacího pole se ve výsledcích několikrát vyskytl URL „<https://www.bezrealitky.cz/api/record/markers>“ u klíče *marker_endpoint*. Endpointem se v API značí vstupní bod do komunikačního kanálu, když dva systémy interagují (33). To znamená, že nalezený formát API by měl být funkční. Po zadání API adresy do prohlížeče se zpět vrátil JSON soubor se všemi inzeráty.

Nalezené API je s velkou pravděpodobností určeno pro uložení tzv. markerů, které se využívají pro zobrazení elementů na mapě. Jednotlivé elementy souboru obsahují základní informaci o inzerátu: id, URI, cenu, koordináty umístění atd. Kromě toho obsahují i typ nemovitosti. Tyto parametry lze zadat do původního URL API a tím získat vyfiltrovaný výpis, tj. pouze nemovitosti typu byt, tedy všechny inzeráty potřebné pro analýzu.

I když nalezené API není primárně určeno pro zobrazení detailů inzerátů, tedy nestačí pro analýzu, přesto se podařilo zjednodušit proces sběru dat, protože již nebude potřeba sbírat odkazy na inzeráty pomocí web scrapingu.

```

    ...
    marker_endpoint: "https://www.bezrealitky.cz/api/record/markers",
    ...
  }
}

```

Obrázek 9: Nalezené API (autor)

3.3.3 Zkoumání API

Dalším krokem bude zkoumání získaných dat v prostředí Jupyter Notebook. Po nahrání dat do objektu DataFrame je vidět, že získaná data obsahují 7 sloupců, ovšem sloupce *timeOrder* a *advertEstateOffer* obsahují vnořené atributy, tedy by bylo vhodné je rozdělit do několika sloupců.

```

In [1]: import pandas as pd

In [2]: # Loading data
url = 'https://www.bezrealitky.cz/api/record/markers?&estateType=byt'
data = pd.read_json(url)
data.shape

Out[2]: (3427, 7)

In [3]: # First 5 rows
data.head()

Out[3]:
   id          uri keyAdvertType type      timeOrder  orderPriority advertEstateOffer
0  710488 710488-nabidka-prodej-bytu-evropska-hlavní-mes... estate_offer  {'date': '2022-03-15 06:57:45.000000', 'timezo... 0  [{"gps": {"lat": 50.09913, "lon": 14.3737}, "pri...}
1  710484 710484-nabidka-pronajem-bytu-tynska-ulicka-praha estate_offer flatio  {'date': '2022-03-15 00:35:18.000000', 'timezo... 0  [{"lat": 50.08790339999999, "lon": 14.3737}, {"gps": {"lat": 50.08790339999999, "lon": 14.3737}, "pri...}
2  710483 710483-nabidka-pronajem-bytu-kmchova-usti-nad... estate_offer  {'date': '2022-03-15 00:16:11.000000', 'timezo... 0  [{"gps": {"lat": 50.67319, "lon": 14.0536}, "pri...}
3  710481 710481-nabidka-pronajem-bytu-dohradek-i-hla... estate_offer  {'date': '2022-03-15 00:06:13.000000', 'timezo... 0  [{"lat": 50.05702105, "lon": 14.2922624}, {"gps": {"lat": 50.05702105, "lon": 14.2922624}, "pri...}
4  710480 710480-nabidka-pronajem-bytu-general-heringy-h... estate_offer  {'date': '2022-03-14 23:39:02.000000', 'timezo... 0  [{"lat": 48.94525, "lon": 16.72328}, {"gps": {"lat": 48.94525, "lon": 16.72328}, "pri...

```

Obrázek 10: Příklad záznamů z API (autor)

Po normalizaci se počet sloupců očividně zvýšil, nové sloupce teď obsahují i užitečnější informaci.

Následně lze odstranit ty sloupce, které nejsou relevantní pro další zpracování. Výsledná data pak obsahují 11 sloupců. Nejdůležitějším z nich je sloupec *uri*, ze kterého pak lze vytvořit odkaz na inzerát, a sloupce *lat* a *lng*, které ukazují koordináty polohy bytu.

	id	uri	date	lat	lng	price	currency	keyOfferType	keyDisposition	surface	surfaceLand
0	710488	710488-nabídka-prodej-bytu-evropská-hlavní-mes...	2022-03-15 06:57:45.000000	50.09913	14.3737	13500000	CZK	prodej	4-1	86	0.0
1	710484	710484-nabídka-pronajem-bytu-tynská-ulicka-praha	2022-03-15 00:35:18.000000	50.08790339999999	14.4227015	26250	CZK	pronajem	2-kk	76	NaN
2	710483	710483-nabídka-pronajem-bytu-kmochova-usti-nad...	2022-03-15 00:16:11.000000	50.67319	14.0536	8500	CZK	pronajem	2-1	65	0.0
3	710481	710481-nabídka-pronajem-bytu-dozahradek-i-hla...	2022-03-15 00:06:13.000000	50.057021	14.292262	17500	CZK	pronajem	2-kk	52	0.0
4	710480	710480-nabídka-pronajem-bytu-generalna-periny-h...	2022-03-14 23:39:02.000000	48.94525	16.72328	12500	CZK	pronajem	2-kk	40	0.0

Obrázek 11: Příklad záznamů z API po rozdělení a odstranění atributů (autor)

Samotný proces web scrapingu teď bude spočívat v procházení jednotlivými inzeráty a vyexportování jejich detailů.

3.3.4 Scraping

Python skript pro web scraping bude obsahovat jedinou třídu WebScraper, která má jeden atribut – *input_table*. Při inicializaci se instance třídy do tohoto atributu zapíše cestou do tabulky z předchozího kroku.

Funkcionalita scraperu bude rozdělena do jednotlivých funkcí. Nejprve je nutné vytvořit odkazy na inzerát. Funkce *add_column_with_links()* přidává do tabulky *input_table* nový sloupec *url*, který je tvořen spojením řetězce „<https://www.bezrealitky.cz/nemovitosti-byty-domky/>“ a sloupce *uri*.

```
@staticmethod
def add_column_with_links(input_data: pd.DataFrame) -> pd.DataFrame:
    """
    Creates new column with URL of listing.
    :param input_data: DataFrame with IDs and URLs of listings
    :return: data with new column
    """

    url_base = 'https://www.bezrealitky.cz/nemovitosti-byty-domky/'

    input_data['url'] = input_data['uri'].apply(lambda x: url_base + x)

    return input_data
```

Obrázek 12: Implementace funkce *add_column_with_links()* (autor)

Jak již bylo zmíněno v teoretické části, pokud se bude během scrapingu odesílat na web hodně požadavků ze stejné IP adresy za krátkou dobu, web s velkou pravděpodobností zablokuje tuto IP adresu. Aby se tomu vyhnulo, lze nastavit rotaci proxy, tj. změnu IP adresy

při každém načtení webové stránky. Na internetu se dá najít seznam bezplatných proxy, které jsou ovšem pomalejší. Jinou variantou je pronajmout si vlastní proxy, což jsem i udělal.

Po nastavení prohlížeče se dále definuje funkce pro vykonávání extrahování dat ze stránky. Z každého řádku tabulky *input_table* vezmeme hodnoty *id* a *url* a vytvoříme prázdný DataFrame. Následně se otevře prohlížeč a bude hledat určité elementy na stránce.

Stránky jednotlivých inzerátů obsahují sekce „Popis“, „Parametry“ a „V okolí najdete“. Sekce „Parametry“ je představena seznamem parametrů bytu, včetně dispozice, plochy, podlaží, vybavenosti, typu budovy atd. Sekce „V okolí najdete“ pak má v sobě informace o vzdálenosti do různých míst, například do banky, pošty, lékárny, školy apod. Třetí sekce „Popis“ uvádí slovní popis inzerátu a určité by se tady dala provést i textová analýza, ale pro danou práci bude tato sekce vynechána. Z každé stránky tedy budou extrahovány parametry bytu a údaje o okolí.

The screenshot shows a real estate listing for a 2+kk apartment in Holečkova, Praha - Smíchov. The listing includes a photo of the interior, a price of 5,990,000 Kč, and a link for a mortgage starting from 22,850 Kč per month. Below the listing, there's a section titled "Parametry" (Parameters) which lists various details about the apartment. At the bottom, there's a section titled "V okolí najdete" (Nearby) which lists nearby landmarks with their distances and directions.

Parametry	Value	Parametry	Value
Dispozice	2+kk	Stav	Novostavba
Plocha	32 m ²	Cena	5.990.000 Kč
Typ vlastnictví	Osobní	Typ budovy	Cihla
Vybavenost	Nevybavený	Podlaží	1
Balkón	Ne	Terasa	Ano
Sklep	Ne	Lodžie	Ne
Parkování	Ne	Výtah	Ano
Garáž	Ne	K dispozici od	16. 03. 2022
PENB	G	Energie	Přepis energií online, snadno a rychle
Investiční rádce	Tento byt pronajmete přibližně za 14.791 Kč	Internet	Zjistěte rychlosť O2 internetu u vás doma.

V okolí najdete	Distance	V okolí najdete	Distance
Zastávka veřejné dopravy: Švandovo divadlo	134 m (2 min)	Pošta: Praha 51	337 m (4 min)
Obchod:	142 m (2 min)	Banka: Evropsko-ruská banka	234 m (3 min)
Restaurace: The Burg	127 m (2 min)	Lékárna: Manor	282 m (3 min)
Škola: Gymnázium Christiana Dopplera	443 m (5 min)	Mateřská škola: Mateřská škola Hellichova	764 m (9 min)
Sportoviště: Sokol Malá Strana	584 m (7 min)	Hřiště: Kinská - Dolní Brána	195 m (2 min)

Obrázek 13: Příklad stránky s inzerátem (autor)

Následně na definujeme XPath elementů. Pro sekce „Parametry“ lze začít od tagu „div“ s id „detail-parameters“. Tento tag obsahuje určitý počet tagů „div“ s id „row param“. Každý

ten „tag“ je představen párem „název parametru: hodnota parametru“. XPath se tím pádem dá najít takto:

- 1) Nejprve najdeme kdekoli na stránce blok s parametry
`//div[@id=„detail-parameters”];`
- 2) Pak s využitím relativní cesty najdeme všechny řádky s parametry
`.//div[@class=„row param”];`
- 3) Následně najdeme všechny děti předchozího elementu a pomocí indexu určíme název a hodnotu parametru
`./*[1]`
`./*[2]`

Parametry			
Dispozice	2+kk	Stav	Velmi dobrý
Plocha	51 m ²	Cena	5.855.000 Kč
Typ vlastnictví	Oсобні	Typ budovy	Ostatní
Vybavenost	Nevybavený	Podlaží	3
Balkón	Ne	Terasa	Ne
Sklep	Ano	Lodžie	Ne
Parkování	Ano	Výtah	Ano
Garáž	Ne	K dispozici od	01. 01. 2022
Stáří	10 - 30 let	Provedení	standardní: běžné materiály a technologie
Typ vytápění	centrální	Podlaží v rámci domu	přízemí
Rekonstrukce	výška	PFNR	D

```


<div class="row pl-md-4">
    <div class="col col-12 col-md-6">
      <div class="row param">
        <div class="col col-6 param-title">Dispozice</div> = $0
        <div class="col col-6 param-value">
          2+kk
        </div>
      </div>
    </div>
  </div>


```

Obrázek 14: Určení XPath vybraných elementů (autor)

Víceméně stejným způsobem lze určit i XPath pro sekce „Okolí“.

- 1) Kdekoli na stránce najdeme blok s popisem okolí
`//div[@id="detail-pois"];`
- 2) U nalezeného tagu se spustíme o pár úrovní níže a získáme všechny řádky
`.//div[@class=„poi-item__content”];`
- 3) Nadefinujeme místo a vzdálenost
`./span[@class=„poi-item__name-type”]`
`./div[@class=„poi-item__walking” or @class=„poi-item__driving”]/strong.`

Získané hodnoty vyexportujeme do tabulky.

```

##### SCRAPING PARAMETERS SECTION #####
try:
    parameters_area = browser.find_element(By.XPATH, '//div[@id="detail-parameters"]')
    params = parameters_area.find_elements(By.XPATH, './/div[@class="row param"]')

    for param in params:
        try:
            title = param.find_element(By.XPATH, './[1]').text.strip().replace('\n', '') # Parameter's title
            value = param.find_element(By.XPATH, './[2]').text.strip().replace('\n', '') # Parameter's value

            # Drop irrelevant parameters
            if title not in ['Investiční rádce', 'Energie', 'Internet']:
                parameters.update({title: value}) # Add title:value to listing details
        except:
            continue
except Exception as e:
    logging.error(f'Error while scraping details on page {i} ({listing_id})')
    print(f'{datetime.datetime.now()} Error while scraping details on page {i} ({listing_id}) - {e}')
    continue
##### END OF SCRAPING PARAMETERS SECTION #####
##### SCRAPING NEIGHBORHOOD INFO SECTION #####
try:
    neighborhood_area = browser.find_element(By.XPATH, '//div[@id="detail-pois"]')
    neighborhood_params = neighborhood_area.find_elements(By.XPATH, './/div[@class="poi-item__content"]')

    for param in neighborhood_params:
        try:
            title = param.find_element(By.XPATH, './span[@class="poi-item__name--type"]').text.replace(':', '') # Parameter's title
            value = param.find_element(By.XPATH, './div[@class="poi-item__walking" or @class="poi-item__driving"]/strong').text # Parameter's value

            parameters.update({title: value}) # Add title:value to listing details
        except:
            continue
except:
    logging.error(f'Error while scraping area info on page {i} ({listing_id})')
    print(f'{datetime.datetime.now()} Error while scraping area info on page {i} ({listing_id})')
    continue
##### END OF SCRAPING NEIGHBORHOOD INFO SECTION #####

```

Obrázek 15: Ukázka funkce pro scrapování inzerátů (autor)

3.4 Porozumění datům

Pomocí web scrapingu se podařilo vyexportovat 3546 nabídky. Výsledný dataset má 54 sloupců. Odstraníme-li duplicitní (cena, plocha, dispozice) a nerelevantní sloupce (developer, kód jednotky, administrační poplatek apod.), zbude jich 35. Dále všechny názvy atributů přeložíme do angličtiny.

3.4.1 Přehled atributů

Tabulka 1: Přehled atributů v datasetu (autor)

Atribut	Datový typ	Počet chybějících hodnot	Význam
id	int	0	unikátní identifikátor inzerátu
uri	string	0	unikátní jednotný identifikátor zdroje
url	string	0	odkaz na inzerát

lat	float	0	zeměpisná šířka polohy bytu
lng	float	0	zeměpisná délka polohy bytu
price	int	0	cena bytu (prodej, měsíční plat)
currency	string	0	měna ceny
key_offer_type	string	0	typ inzerátu: prodej, pronájem
key_disposition	string	1	dispozice bytu
surface	int	0	plocha bytu v m ²
utilities	string	1273	poplatky za byt (pouze pro nájemné byty)
facilities	string	439	vybavenost bytu
floor	float	134	podlaží, ve kterém se nachází byt
balcony	string	0	má-li byt balkón
terrace	string	0	má-li byt terasu
cellar	string	0	má-li byt sklep
loggia	string	0	má-li byt lodžii
parking	string	0	má-li byt parkoviště
elevator	string	0	má-li byt výtah
garage	string	0	má-li byt garáž
public_transport_stop	string	0	vzdálenost do nejbližší zastávky veřejné dopravy
post_office	string	1	vzdálenost do nejbližší pošty

store	string	0	vzdálenost do nejbližšího obchodu
bank	string	67	vzdálenost do nejbližší banky
restaurant	string	0	vzdálenost do nejbližší restaurace
pharmacy	string	14	vzdálenost do nejbližší lékárny
school	string	12	vzdálenost do nejbližší školy
kinder_garten	string	42	vzdálenost do nejbližší mateřské školy
sport_field	string	35	vzdálenost do nejbližšího sportoviště
playground	string	5	vzdálenost do nejbližšího hřiště
condition	string	434	stav budovy
property_type	string	115	typ vlastnictví (osobní, družstevní apod.)
building_type	string	108	typ budovy, ve které se nachází byt
penb	string	575	energetická náročnost provozu bytu dle klasifikace PENB
deposit	string	1208	vratná kauce

3.4.2 Příklad záznamů

utilities	facilities	floor	balcony	terrace	cellar	loggia	parking	elevator	garage	public_transport_stop	post_office	store	bank	restaurant	pharmacy
7.500 Kč	Vybavený	6.0	Ano	Ne	Ne	Ne	Ne	Ano	Ano	202 m	523 m	119 m	828 m	248 m	936 m
NaN	Nevybavený	4.0	Ano	Ne	Ano	Ano	Ano	Ano	Ne	63 m	111 m	311 m	2.8 km	359 m	428 m
4.000 Kč	Vybavený	1.0	Ano	Ne	Ano	Ne	Ano	Ano	Ne	177 m	1,193 m	300 m	543 m	54 m	531 m
NaN	Částečně	4.0	Ano	Ne	Ano	Ne	Ano	Ano	Ne	408 m	774 m	839 m	1,037 m	713 m	690 m
6.000 Kč	Vybavený	2.0	Ne	Ne	Ne	Ano	Ne	Ne	Ne	128 m	650 m	96 m	653 m	133 m	531 m

Obrázek 16: Příklad získaných dat (autor)

Některé z atributů se vztahují pouze k inzerátům o pronájmu bytu. K takovým atributům patří *utilities* a *deposit*.

Atributy *balcony*, *terrace*, *cellar*, *loggia*, *parking*, *elevator* a *garage* nabývají pouze hodnot Ano/Ne, tedy by bylo vhodně je během fáze předzpracování dat převést do dichotomických hodnot 1 (Ano) a 0 (Ne).

Atributy, které se týkají vzdálenosti do jednotlivých míst, mají datový typ *string*, i když je zřejmé, že musí obsahovat číselné hodnoty. Po zkoumání dat je však vidět, že u některých inzerátů jsou hodnoty těchto atributů v kilometrech, nikoliv v metrech. Tím pádem převod atributů do číselních hodnot musí být udělán s ohledem na měrné jednotky.

Také je důležité brát v potaz i měnu u ceny za byt. V datasetu se vyskytuje jak české koruny, tak i eura.

Atributy *key_disposition*, *property_type*, *building_type* a *penb* mají omezený počet hodnot a lze z nich udělat kategoriální proměnné.

U atributu *key_disposition* se vyskytují pro český realitní trh netypické hodnoty: 1–izb, 2–izb, 3–izb a 4–izb. To jsou označení dispozice bytů na Slovensku, a jelikož se tato práce zabývá analýzou českého realitního trhu, inzeráty s podobnými hodnoty mohou být odstraněny z datasetu.

Další pohled na data lze získat pomocí EDA, tj. vizualizace, ale ještě před tím stojí za úpravu datasetu.

3.5 Předzpracování dat

Další fází metodiky CRISP-DM je předzpracování dat. Výsledkem této fáze by měl být dataset, který lze použít pro samotnou analýzu.

3.5.1 Odstranění sloupců a řádků

Smyslem odstranění atributů je vynechat ty, které nejsou pro analýzu relevantní. Jako první jsem odstranil atributy *uri* a *url*, protože pro identifikaci inzerátu stačí sloupec *id*.

```
# Drop columns
df.drop(['uri', 'url'], axis=1, inplace=True)
```

Obrázek 17: Odstranění nerelevantních sloupců (autor)

Zjistilo se, že se kromě inzerátů o prodeji a pronájmu v datasetu vyskytují i inzeráty o spolubydlení. Podobné řádky jsem odstranil.

```
# Drop rows with type of ad "spolubydlení"
df = df[df['key_offer_type'] != 'spolubydlení']
df['key_offer_type'].value_counts()
pronajem    2456
prodej      1000
Name: key_offer_type, dtype: int64
```

Obrázek 18: Odstranění nerelevantních záznamů (autor)

Stejně jsem odstranil i řádky, které u atributu *key_disposition* mají slovenské označení dispozice bytu. Předtím jsem ale ověřil, zda opravdu jde o byty na Slovensku, nikoliv o byty, které se nachází v České republice, ale majitel inzerátu je ze Slovenska. Tohle jsem ověřil pomocí odkazů na inzeráty. U všech nalezených inzerátů se v odkazu vyskytovala buď Bratislava, anebo jiné slovenské město, tyto inzeráty lze tedy odstranit.

```
# Drop rows with ads from Slovakia
df = df[~df['key_disposition'].isin(['1-izb', '2-izb', '3-izb', '4-izb'])]
```

Obrázek 19: Odstranění inzerátů ze Slovenska (autor)

Kromě toho jsem zjistil, že inzeráty, které jsou uvedené v eurech, se také nachází na Slovensku, a proto byly odstraněny z datasetu.

```
# Leave only records in CZK
print(f'Before: {len(df)} rows')
df = df[df['currency'] == 'CZK']
print(f'After: {len(df)} rows')

Before: 3426 rows
After: 3348 rows
```

Obrázek 20: Odstranění inzerátů v jiných měnách než CZK (autor)

Některé inzeráty obsahovaly nesmysluplnou informaci o ploše bytu, měly tak u tohoto atributu hodnotu 0. Tyhle inzeráty jsem také odstranil.

```
# Drop rows where surface is 0
print(f'Before: {len(df)} rows')
df = df[df['surface'] != 0]
print(f'After: {len(df)} rows')

Before: 3348 rows
After: 3337 rows
```

Obrázek 21: Odstranění inzerátů s nulovou plochou (autor)

3.5.2 Transformace (převod na jiné datové typy)

Některé kategoriální atributy (balkón, terasa, lodžie, výtah, garáž, parkoviště a sklep) mohou nabývat pouze 2 hodnot, a proto by bylo vhodné je převést do numerických hodnot 1 a 0, aby se pak dalo s nimi pracovat při modelování (některé data mining modely neumějí pracovat s kategoriálními proměnnými).

```
# Transform binary attributes to 0/1
binary_attributes = ['balcony', 'terrace', 'cellar', 'loggia', 'parking', 'elevator', 'garage']

for attribute in binary_attributes:
    df[attribute] = df[attribute].map({'Ano': 1, 'Ne': 0})

df.head()
```

sposition	surface	utilities	facilities	floor	balcony	terrace	cellar	loggia	parking	elevator	garage	public_transport_stop	post_office	store	bank	restar
1-kk	35	7.500 Kč	Vybavený	6.0	1	0	0	0	0	1	1	202 m	523 m	119 m	828 m	2
3-1	74	NaN	Nevybavený	4.0	1	0	1	1	1	1	0	63 m	111 m	311 m	2.8 km	3
1-1	38	4.000 Kč	Vybavený	1.0	1	0	1	0	1	1	0	177 m	1,193 m	300 m	543 m	
3-kk	80	NaN	Částečně	4.0	1	0	1	0	1	1	0	408 m	774 m	839 m	1,037 m	7
3-kk	78	6.000 Kč	Vybavený	2.0	0	0	0	1	0	0	0	128 m	650 m	96 m	653 m	1

Obrázek 22: Převod vybraných atributů do binárních hodnot (autor)

Atributy popisující vzdálenost do jednotlivých míst jsou představeny textem, je proto potřeba převést je do numerických hodnot s ohledem na měrné jednotky. To znamená, že pokud se u atributu vyskytuje „km“, výsledná hodnota musí být vynásobena 1000.

```
# Transform string attributes with area info to float
area_info_attributes = ['public_transport_stop', 'post_office', 'store', 'bank', 'restaurant',
                        'pharmacy', 'school', 'kindergarten', 'sport_field', 'playground']

for attribute in area_info_attributes:
    df[attribute] = df[attribute].apply(lambda x: float(str(x).replace(' m', '').replace(',', '')))
        if ' m' in str(x)
        else float(str(x).replace(' km', '')) * 1000

df.head()
```

garage	public_transport_stop	post_office	store	bank	restaurant	pharmacy	school	kindergarten	sport_field	playground	condition	property_type	buildir
1	202.0	523.0	119.0	828.0	248.0	936.0	219.0	756.0	1073.0	529.0	NaN	NaN	
0	63.0	111.0	311.0	2800.0	359.0	428.0	259.0	645.0	665.0	170.0	Velmi dobrý	Osobní	
0	177.0	1193.0	300.0	543.0	54.0	531.0	554.0	212.0	777.0	1017.0	Velmi dobrý	Osobní	
0	408.0	774.0	839.0	1037.0	713.0	690.0	797.0	623.0	4000.0	198.0	Novostavba	Osobní	
0	128.0	650.0	96.0	653.0	133.0	531.0	348.0	604.0	641.0	358.0	Dobrý	Osobní	

Obrázek 23: Převod textových atributů do numerických (autor)

Atributy obsahující měny lze také převést na numerické hodnoty, a to odstraněním textových řetězců „Kč“ a „€“.

```
attributes_with_currency = ['utilities', 'deposit']

for attribute in attributes_with_currency:
    df[attribute] = df[attribute].apply(lambda x: int(x.replace('Kč', '')).replace('.', '').replace('€', '').strip())
        if not pd.isnull(x)
        else None)

df.head()
```

	id	lat	lng	price	currency	key_offer_type	key_disposition	surface	utilities	facilities	floor	balcony	terrace	cellar	loggia	parl
0	710805	50.069999	14.382026	22500	CZK	pronajem	1-kk	35	7500.0	Vybavený	6.0	1	0	0	0	
1	710804	49.750050	13.415810	4550000	CZK	prodej	3-1	74	NaN	Nevybavený	4.0	1	0	1	1	
2	710803	49.188190	16.588290	10000	CZK	pronajem	1-1	38	4000.0	Vybavený	1.0	1	0	1	0	
3	710799	50.052230	15.751670	6690000	CZK	prodej	3-kk	80	NaN	Částečně	4.0	1	0	1	0	
4	710797	50.051150	14.457670	16000	CZK	pronajem	3-kk	78	6000.0	Vybavený	2.0	0	0	0	1	

Obrázek 24: Převod atributu „měna“ do numerického (autor)

V získaném datasetu počet bytů s dispozicí 5-kk, 5-1, 6-kk, 6-1, 7-kk a „ostatní“ dohromady činil méně než 35 bytů, a proto jsem zmíněné kategorie spojil do jedné.

```
df['key_disposition'].value_counts()

2-kk      854
2-1       594
1-kk      555
1-1       379
3-1       356
3-kk      335
garsoniera   108
4-kk       62
4-1       60
ostatni     11
5-1        9
5-kk        9
6-kk        2
7-kk        1
6-1        1
Name: key_disposition, dtype: int64
```

```
# Merge small groups

df['key_disposition'] = df['key_disposition'].replace(['5-kk', '5-1', '6-kk', '6-1', '7-kk', 'ostatni'], 'ostatní')
```

Obrázek 25: Seskupení málo četných hodnot atributu „dispozice“ (autor)

Poslední transformací bylo převést vybrané atributy do kategoriálních proměnných.

```
categorical_attributes = ['currency', 'key_offer_type', 'key_disposition', 'facilities', 'floor',
                           'condition', 'property_type', 'building_type', 'penb'] + binary_attributes

for attribute in categorical_attributes:
    df[attribute] = df[attribute].astype('category')
```

Obrázek 26: Převod vybraných atributů do kategoriálních (autor)

3.5.3 Chybějící hodnoty

Některé atributy v datasetu mají chybějící hodnoty, což by mohlo zhoršit finální výsledky analýzy. Na první pohled se zdá, že některé atributy (*utilities* a *deposit*) obsahují příliš vysoký počet chybějících hodnot. Pokud ale rozdělíme tento počet podle typu inzerátu, bude

zřejmé, že všechny inzeráty o prodeji bytu nemají atributy *utilities* a *deposit*, což zní docela logicky, vždyť pokud jde o nákup bytu, tak poplatky a vratná kauce se nikdy neuvádějí. V datové analýze se podobná situace nazývá MAR (angl. Missing at Random), tj. chybějící hodnoty lze objasnit jiným atributem – dispozicí bytu.

Tabulka 2: Přehled chybějících hodnot v datasetu (autor)

Atribut	Celkem	Prodej	Pronájem
key_disposition	1	1	0
utilities	1273	994	279
facilities	439	123	316
floor	134	34	100
post_office	1	0	1
bank	67	28	39
pharmacy	14	3	11
school	12	4	8
kinder_garten	42	16	26
sport_field	35	14	21
playground	5	2	3
condition	434	17	417
property_type	115	0	115
building_type	108	0	108
penb	575	128	447
deposit	1208	994	214

Z tabulky je také vidět, že inzeráty o prodeji bytů jsou detailnější, tj. mají menší procento nevyplněných hodnot. I když to spíš není žádné zvláštní zjištění, lze předpokládat, že se majitelé snaží popsát své byty více v případě prodeje než v případě pronájmu.

Atributy o okolí

Podíl inzerátů s nevyplněným alespoň jedním atributem o okolí z celkového počtu inzerátů se rovná asi 3 %. Tato hodnota je relativně malá, a proto v daném případě lze řádky prostě odstranit, nikoliv se snažit chybějící hodnoty nějak vyplnit.

```
: area_info_attributes = ['post_office', 'store', 'bank', 'pharmacy', 'school',
                           'kindergarten', 'sport_field', 'playground']

# Rate of missing values
missing_values_count = df[area_info_attributes].isna().any(axis=1).sum()

f'Missing values rate: {round(missing_values_count/len(df)*100, 2)} %'

: 'Missing values rate: 2.97 %'

: print(f'Before: {len(df)} rows')
df.dropna(axis=0, how='any', subset=area_info_attributes, inplace=True)
print(f'After: {len(df)} rows')

Before: 3337 rows
After: 3238 rows
```

Obrázek 27: Odstranění inzerátů s chybějícími hodnotami u atributů o okolí (autor)

Atribut floor

V datasetu se lze setkat s byty, které se nachází na podlaží od -1 až po 48.

Tabulka 3: Přehled inzerátů podle podlaží (autor)

Hodnota	Počet
1	760
2	700
3	604
4	389
5	266
6	151
7	68
8	60
9	25
10	25
11	17
-1	20

12	10
13	8
14	2
19	2
17	1
21	1
48	1

Z tabulky vyplývá, že inzeráty s podlažím od 1 po 5 pokrývá skoro 85 % datasetu. Navíc vysoké budovy nejsou v ČR tak běžné, nejvyšší budova (AZ Tower v Brně) má 30 podlaží. Vzhledem k těmto skutečnostem jsem se rozhodl vyplnit chybějící pole náhodnou hodnotou z rozmezí [1, 5].

```
df_1_to_5_floor = df[df['floor'].isin([1, 2, 3, 4, 5])]
print(f'1st to 5th floor: {len(df_1_to_5_floor)}')
print(f'All floors: {len(df)}')
print(f'1st to 5th floor share: {round(len(df_1_to_5_floor) / len(df)*100, 2)} %')

1st to 5th floor: 2719
All floors: 3238
1st to 5th floor share: 83.97 %

# Filling missing values
floors = [1, 2, 3, 4, 5]
random.seed(17)

df_floor_missing_values = df['floor'].isnull()    # Missing values
df_floor_missing_values_count = df_floor_missing_values.sum()    # Count of missing values
random_floors = random.choices(floors, k=df_floor_missing_values_count)    # List of random floors
df.loc[df_floor_missing_values, 'floor'] = random_floors    # Assigning list

df['floor'].isnull().sum()

0
```

Obrázek 28: Vyplnění chybějících hodnot u atributu „podlaží“ (autor)

Atribut facilities

Tento atribut je kategoriální a jednou možností bylo vyplnit chybějící hodnoty nejčetnější kategorií. V takovém případě by mohlo ale dojít ke zkreslení kategorií. Jinou variantou je přidat novou kategorii a přiřadit k ní všechny chybějící hodnoty. Takže jsem u atributu *vybavenost* vytvořil novou kategorii *None*, kterou jsem vyplnil o chybějící hodnoty.

```

df['facilities'].value_counts()
Částečně      1368
Vybavený       945
Nevybavený     497
Name: facilities, dtype: int64

# Add new category
df['facilities'] = df['facilities'].cat.add_categories('None').fillna('None')

```

Obrázek 29: Vyplnění chybějících hodnot u atributu „vybavenost“ (autor)

Po přidání nové kategorie rozdělení vypadalo následovně:

```

df['facilities'].value_counts()
Částečně      1452
Vybavený       1126
Nevybavený     523
None           445
Name: facilities, dtype: int64

```

Obrázek 30: Rozdělení kategorií atributu „vybavenost“ (autor)

Atributy condition, building_type a penb

Stejně tak jsem přidal novou kategorii i pro stav budovy, typ budovy a energetickou třídu PENB.

```

df['condition'].value_counts()
Velmi dobrý    1651
Dobrý          607
Novostavba     535
Špatný         20
Ve výstavbě    11
Name: condition, dtype: int64

# Add new category
df['condition'] = df['condition'].cat.add_categories('None').fillna('None')

```

Obrázek 31: Vyplnění chybějících hodnot u atributu „stav_budovy“ (autor)

Atribut property_type

Chybějící hodnoty u atributu *typ vlastnictví* lze také vyřešit přidáním nové kategorie, ovšem tady je počet unikátních hodnot 3. Nejčetnější z nich je „Osobní“, přičemž se vyskytuje u více než 90 % inzerátů. Touto hodnotou jsem vyplnil chybějící řádky.

```

df_rent['property_type'].value_counts()
Osobní        1970
Družstevní    133
Ostatní       33
Name: property_type, dtype: int64

df_rent['property_type'].fillna('Osobní', inplace=True)

```

Obrázek 32: Vyplnění chybějících hodnot u atributu „typ_vlastnictví“ (autor)

Další předzpracování dat bude probíhat zvlášť pro inzeráty o prodeji a pronájmu, protože tyto typy inzerátů obsahují různé atributy a jejich hodnoty.

3.5.4 Inzeráty o pronájmu

Atribut price

V datasetu se vyskytovaly inzeráty s měsíčnou cenou nižší než 1500, což není z ekonomického hlediska realistické. Takovou malou hodnotu nájemného si lze představit, pokud jde o pronájem pokoje, nikoliv celého bytu. Je možné, že jde o chyby během přidání inzerátu, anebo o situaci, kdy majitel inzerátu se ještě nerozhodl ohledně ceny, takže cena je po domluvě. Každopádně podobné inzeráty jsem z datasetu odstranil.

```
# Remove meaningless records
print(f'Before: {len(df_rent)} rows')
df_rent = df_rent.loc[df_rent['price'] > 1500]
print(f'After: {len(df_rent)} rows')

Before: 2304 rows
After: 2302 rows
```

Obrázek 33: Odstranění inzerátů s cenou menší než 1500 Kč (autor)

Atribut deposit

Atribut *deposit* nelze považovat za význačný, jelikož závisí na atributu *price*. Většinou očividně platí, že čím větší je nájemné, tím je větší i kauce. Proto jsem přidal další atribut *deposit_by_price*, který udává podíl kauci na ceně. Z osobní zkušenosti jsem předpokládal, že se vratná kauce obvykle rovná dvojnásobné měsíční ceně za byt. Z datasetu ovšem vychází, že mediánní hodnota pro atribut *deposit_by_price* je 1.6. Takže jsem vyplnil všechny chybějící hodnoty podle vzorce:

$$\text{deposit} = \text{price} * 1.6$$

```
# Adding new attribute deposit_by_price = deposit / price
df_rent['deposit_by_price'] = df['deposit'] / df['price']
df_rent.head()
```

price	store	bank	restaurant	pharmacy	school	kindergarten	sport_field	playground	condition	property_type	building_type	penb	deposit	deposit_by_price
23.0	119.0	828.0	248.0	936.0	219.0	756.0	1073.0	529.0	None	Osobní	None	None	NaN	NaN
33.0	300.0	543.0	54.0	531.0	554.0	212.0	777.0	1017.0	Velmi dobrý	Osobní	Cihla	G	10000.0	1.0000
50.0	96.0	653.0	133.0	531.0	348.0	604.0	641.0	358.0	Dobrý	Osobní	Cihla	G	21000.0	1.3125
50.0	15.0	494.0	73.0	336.0	519.0	216.0	542.0	280.0	None	Osobní	None	None	NaN	NaN
51.0	567.0	967.0	555.0	723.0	170.0	329.0	734.0	120.0	Velmi dobrý	Osobní	Panel	D	19700.0	2.0000

Obrázek 34: Přidání nového atributu „deposit_by_price“ (autor)

Dalším krokem jsem dopočítal Z-skóre pro atribut *deposit_by_price* a našel záznamy, u kterých hodnota Z-skóre je větší než 3. Vyšly mi tak 23 inzeráty, které lze považovat za outliery a jež bude vhodné pak odstranit.

```
df_rent['deposit_by_price'].describe()

count    2110.00000
mean      1.642455
std       0.552674
min      0.000053
25%     1.180015
50%     1.666667
75%     2.000000
max      4.255319
Name: deposit_by_price, dtype: float64

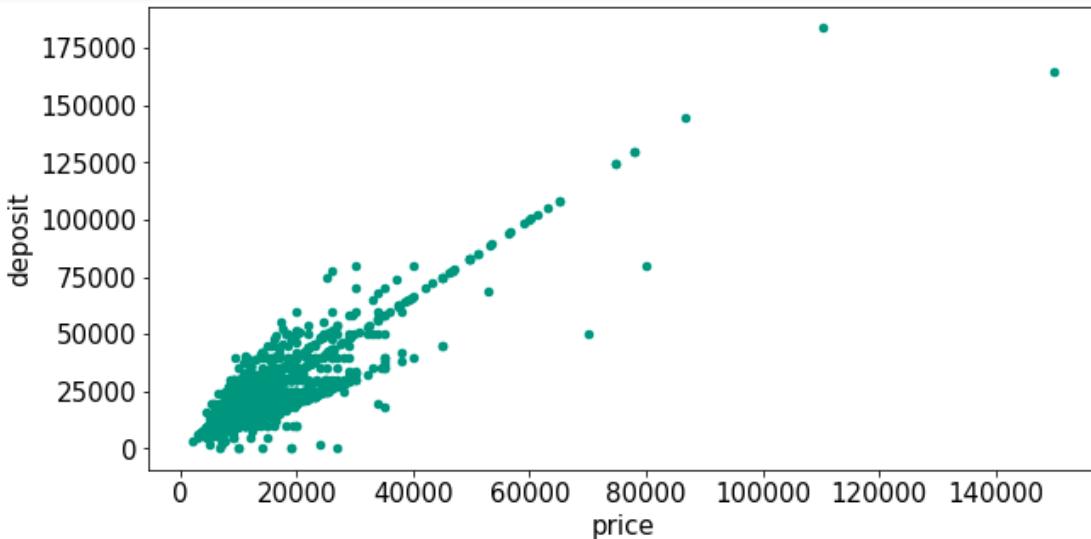
# Filling missing values
median_deposit_by_price = df_rent['deposit_by_price'].median()
df_rent['deposit'].fillna(df_rent['price']*median_deposit_by_price, inplace=True)
df_rent['deposit_by_price'].fillna(median_deposit_by_price, inplace=True)

# Outliers
z_score_deposit_by_price = np.abs(stats.zscore(df_rent['deposit_by_price']))
df_rent_deposit_by_price_outliers = df_rent[z_score_deposit_by_price > 3]

print(f'Number of outliers: {len(df_rent_deposit_by_price_outliers)}')
df_rent_deposit_by_price_outliers.head()

Number of outliers: 23
```

Obrázek 35: Identifikace outlierů u atributu „deposit_by_price“ (autor)



Obrázek 36: Graf ceny a kauce (autor)

Atribut utilities

Stejně jako u atributu *deposit* jsem pro vyplnění chybějících řad použil mediánní hodnotu z celého sloupce a následně jsem dopočítal Z-skóre. Tentokrát se počet outlierů rovná 40, ale i tak to není moc velké číslo, a proto jsem je odstranil. Navíc jsem odstranil inzeráty s poplatky méně než 100 Kč.

```

# Filling missing values
median_utilities = df_rent['utilities'].median()
df_rent['utilities'].fillna(df_rent['price']*median_utilities, inplace=True)

# Outliers
z_score_utilities = np.abs(stats.zscore(df_rent['utilities']))
df_rent_utilities_outliers = df_rent[z_score_utilities > 3]
print(f'Number of outliers: {len(df_rent_utilities_outliers)}')

Number of outliers: 40

print(f'Before: {len(df_rent)} rows')
df_rent = pd.concat([df_rent, df_rent_utilities_outliers]).drop_duplicates(keep=False)
print(f'After: {len(df_rent)} rows')

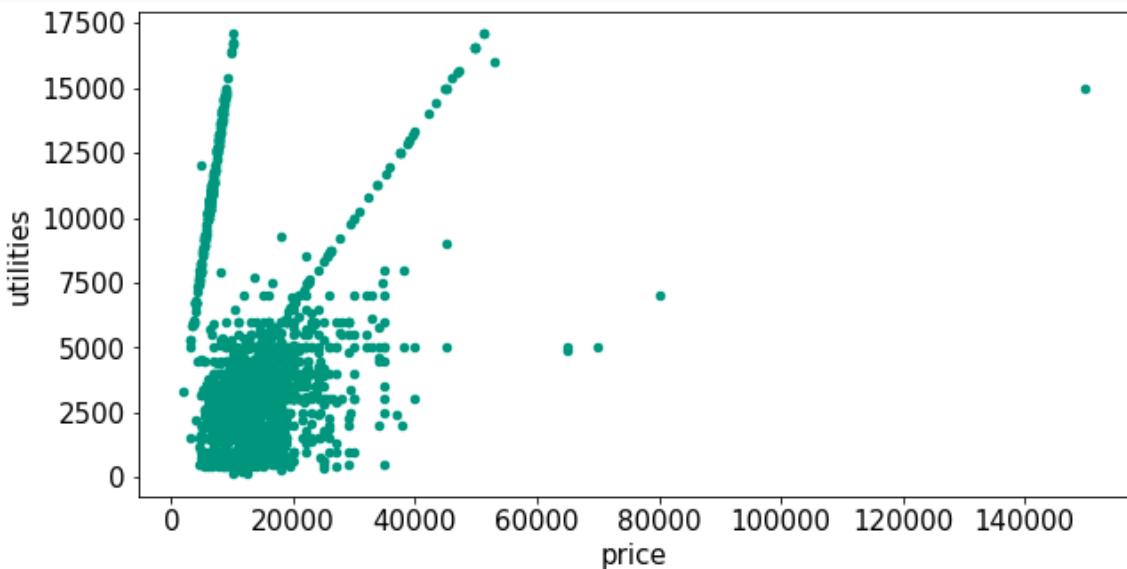
Before: 2279 rows
After: 2239 rows

print(f'Before: {len(df_rent)} rows')
df_rent = df_rent.loc[df_rent['utilities'] > 100]
print(f'After: {len(df_rent)} rows')

Before: 2239 rows
After: 2220 rows

```

Obrázek 37: Identifikace outlierů u atributu „poplatky“ (autor)



Obrázek 38: Graf ceny a poplatků (autor)

Stojí ještě za zmínu, že tento atribut není vždy jednoznačný. Tím myslím to, že byt může pronajímat i více lidí a v takovém případě budou poplatky určitě vyšší. To znamená, že v datasetu se můžou vyskytovat inzeráty, u kterých je hodnota poplatků uvedena za člověka, nikoliv za celý byt.

3.5.5 Inzeráty o prodeji

U inzerátů typu prodej zůstal jeden záznam s chybějící hodnotou dispozice. Tento řádek jsem odstranil stejně jako inzeráty s prodejní cenou nižší než 200 000 Kč.

df_sell[df_sell['key_disposition'].isnull()]																
	id	uri	lat	lng	price	currency	key_offer_type	key_disposition	surface	utilities	facilities	floor	balcony	terrace	cellar	lc
2248	691939	691939-nabídka-prodej-bytu-cukrovar	49.244742	17.464484	1449000	CZK	prodej		NaN	221	NaN	None	3.0	0	0	0

```
print(f'Before: {len(df_sell)} rows')
df_sell.dropna(axis=0, subset=['key_disposition'], inplace=True)
print(f'After: {len(df_sell)} rows')
```

Before: 934 rows
After: 933 rows

Price

```
# Remove meaningless records
print(f'Before: {len(df_sell)} rows')
df_sell = df_sell.loc[df_sell['price'] > 200_000]
print(f'After: {len(df_sell)} rows')
```

Before: 933 rows
After: 931 rows

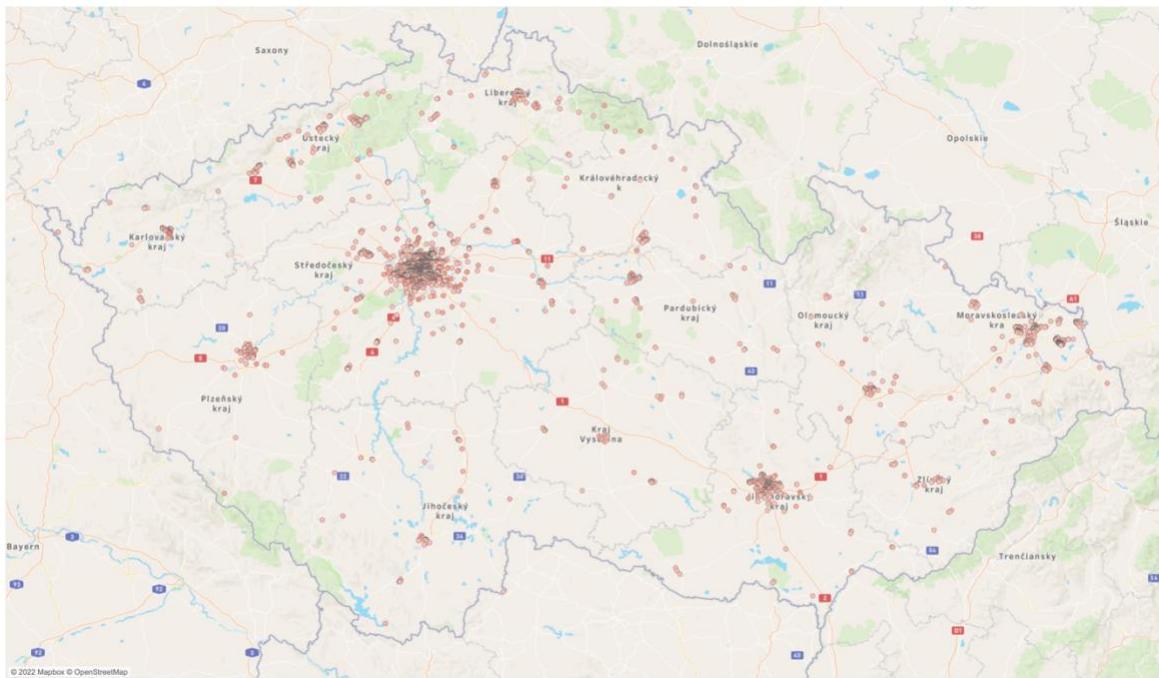
Obrázek 39: Odstranění inzerátů s cenou menší než 200 000 Kč (autor)

Ve výsledku zbylo z původního datasetu 3151 záznamů: 2220 inzerátů o pronájmu a 931 inzerát o prodeje. Po explorační analýze může nastat situace, kdy bude potřeba provést opakovou úpravu datasetu. Tohle je ovšem v souladu s předpokladem cyklickosti metodiky CRISP-DM.

3.6 Explorační analýza

I když není explorační analýza krokem metodiky CRISP-DM, považuje se většinou za nezbytnou součást analýzy. Umožňuje získat další pohled, případně i najít nějaké zajímavé vztahy.

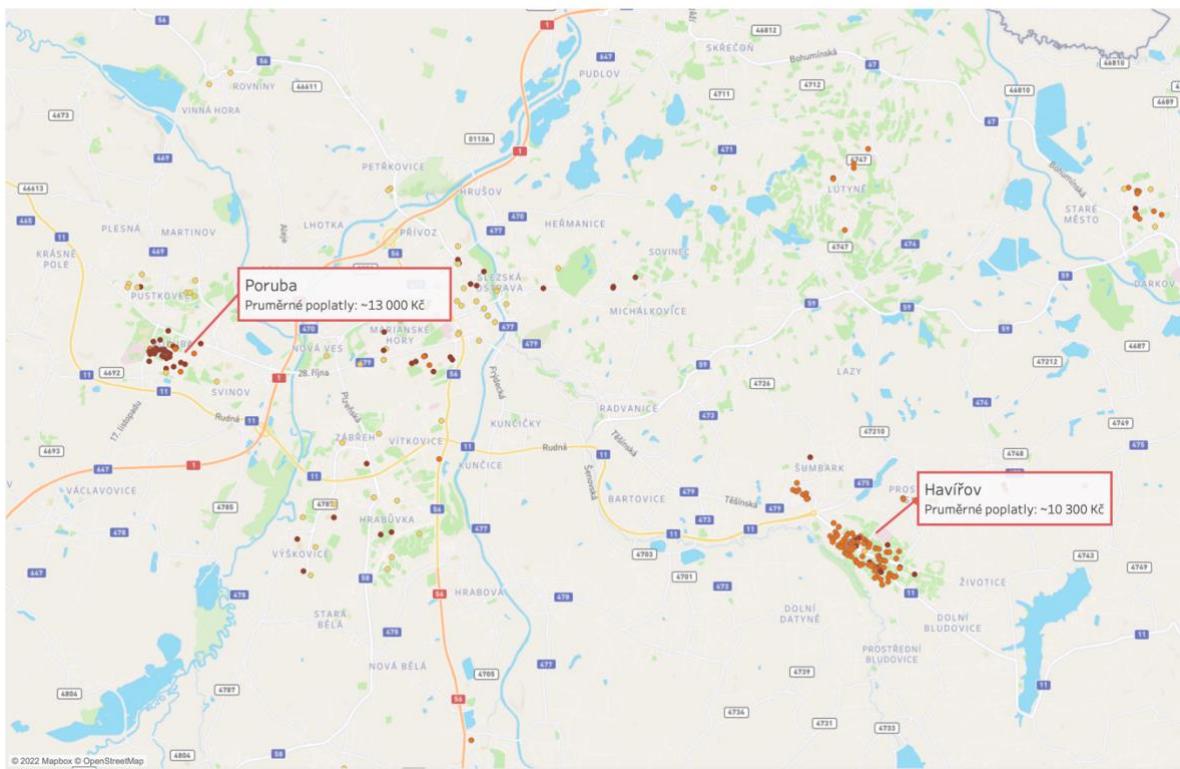
3.6.1 Mapa nabídek



Obrázek 40: Mapa inzerátů (autor)

Nejprve jsem zobrazil všechny nabídky na mapě. Samozřejmě se většina nabídek soustředila v hlavním městě a bylo by rozumné je vyčlenit, aby dále byly porovnávány byty z pohledu ČR/Praha.

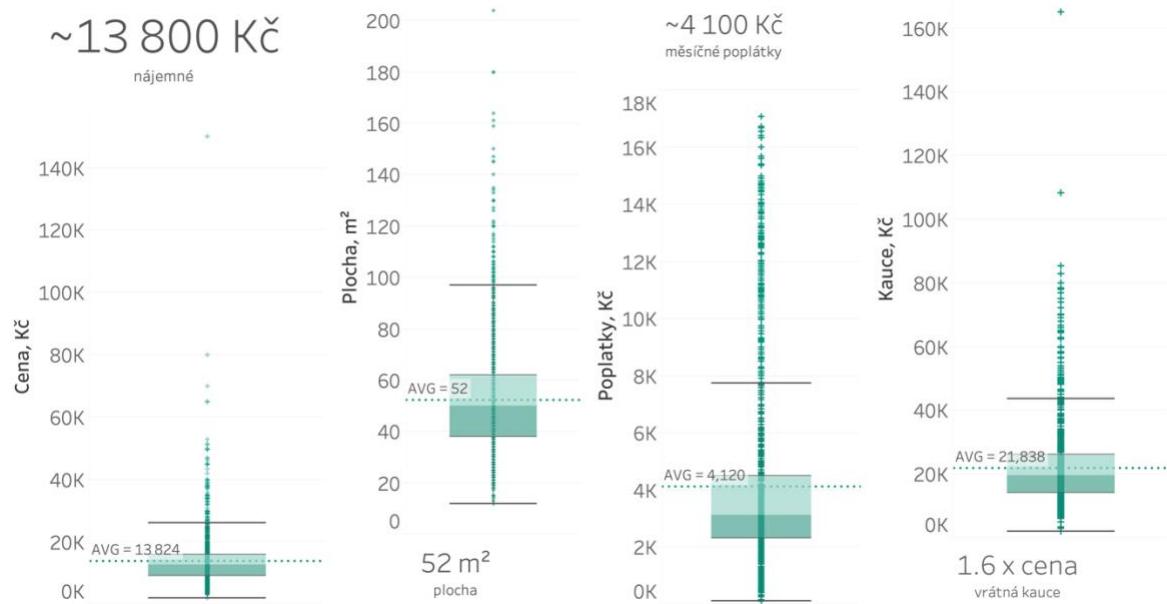
Na mapě se dají zobrazit nabídky podle hodnoty poplatků, co jsem i udělal, a zjistil jsem, že v Moravskoslezském kraji jsou města (Havířov a Poruba), ve kterých byty mají obzvlášť vysoké poplatky.



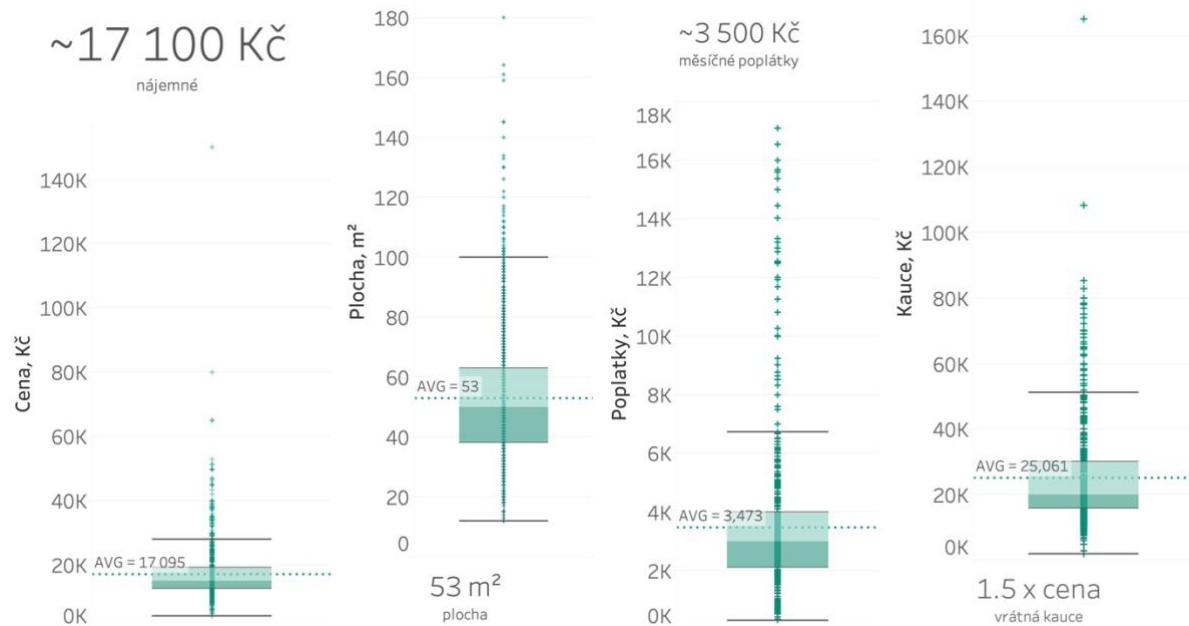
Obrázek 41: Inzeráty s vysokými poplatky v Moravskoslezském kraji (autor)

3.6.2 Průměrný byt k pronájmu

Po rozdelení nabídek jsem vytvořil profil typického bytu k pronájmu v celé ČR a v Praze. Zjistilo se, že pronajmout byt s plochou 52 m² v České republice stojí přibližně 13 800 Kč. V Praze má střední byt skoro stejnou plochu (53 m²) a cenu větší o 23 % – tady majitelé bytů vyžadují 17 100 Kč/měsíc. Na druhou stranu měsíční poplatky ve městech mimo Prahu jsou větší o 17 % – 4100 Kč oproti 3 500 Kč v Praze. Nejdražší byt k pronájmu se nachází v Praze a stojí téměř 150 000 Kč/měsíc. Co se týká vratné kauce, tady jsou hodnoty v ČR a Praze skoro stejné – všude nájemce musí zaplatit majiteli 1,5× ceny.



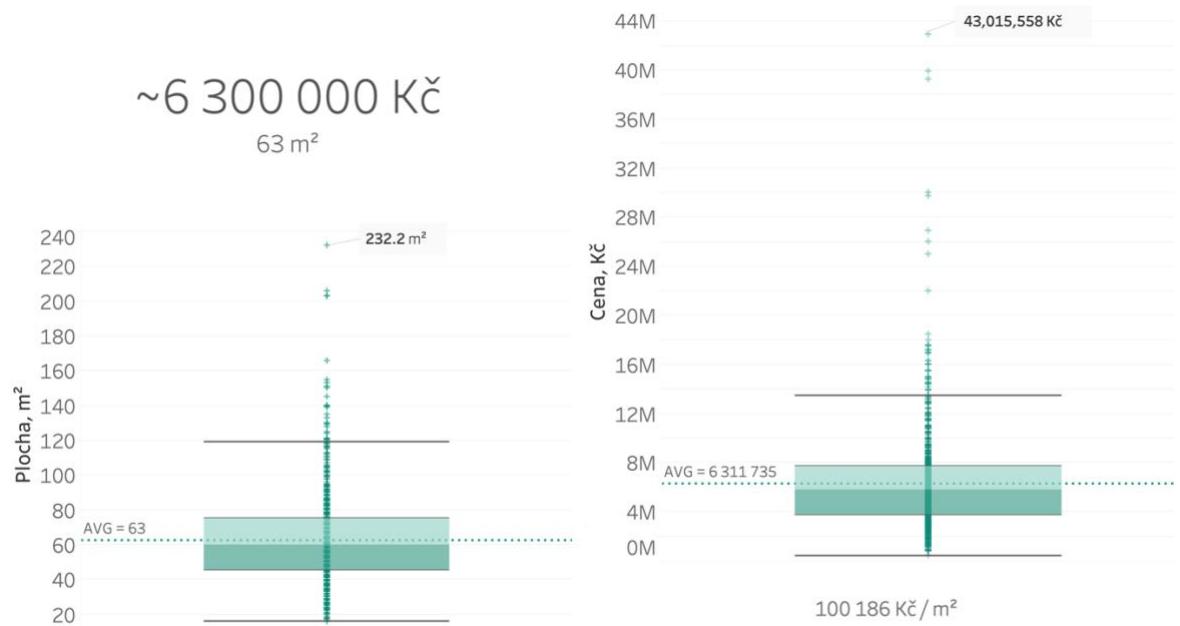
Obrázek 42: Průměrný byt k pronájmu v České republice (autor)



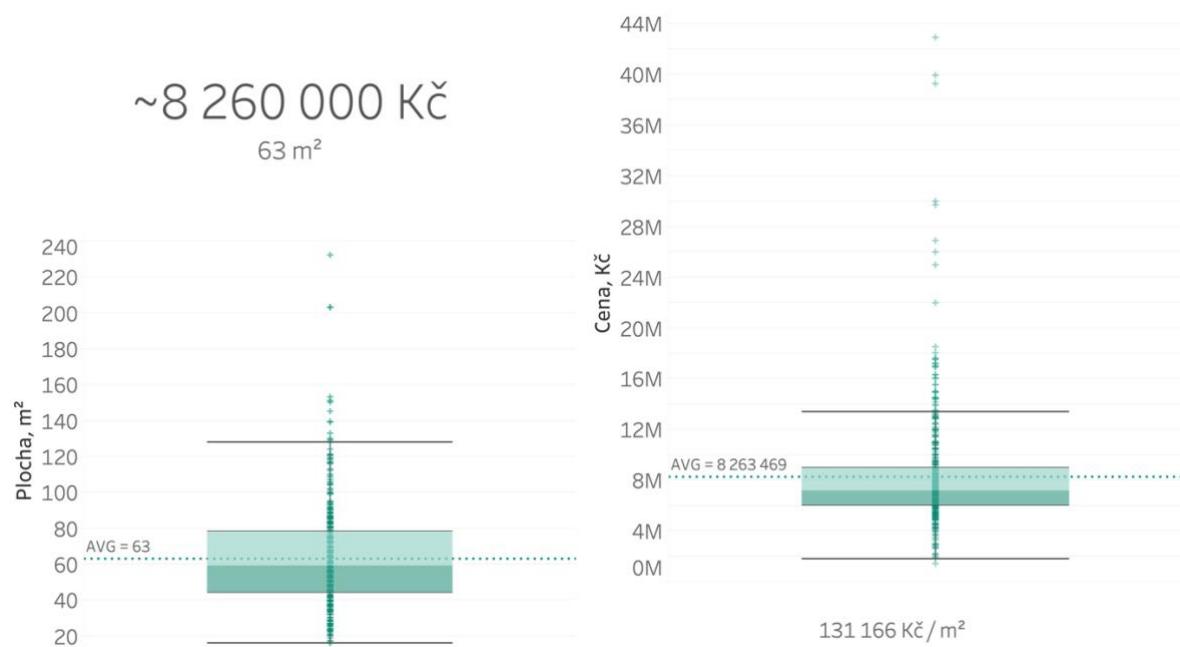
Obrázek 43: Průměrný byt k pronájmu v Praze (autor)

3.6.3 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 63 m². Cena se ale liší více než o 30 %. V Praze je tato hodnota asi 8 260 000 Kč, v celé ČR je to 6 300 000 Kč. Pokud vezmeme absolutní ukazatel, čímž je cena za jeden čtvereční metr, pak kupující v Praze musejí vynaložit 131 166 Kč/m², zatímco průměrně v České republice stačí 100 186 Kč/m.



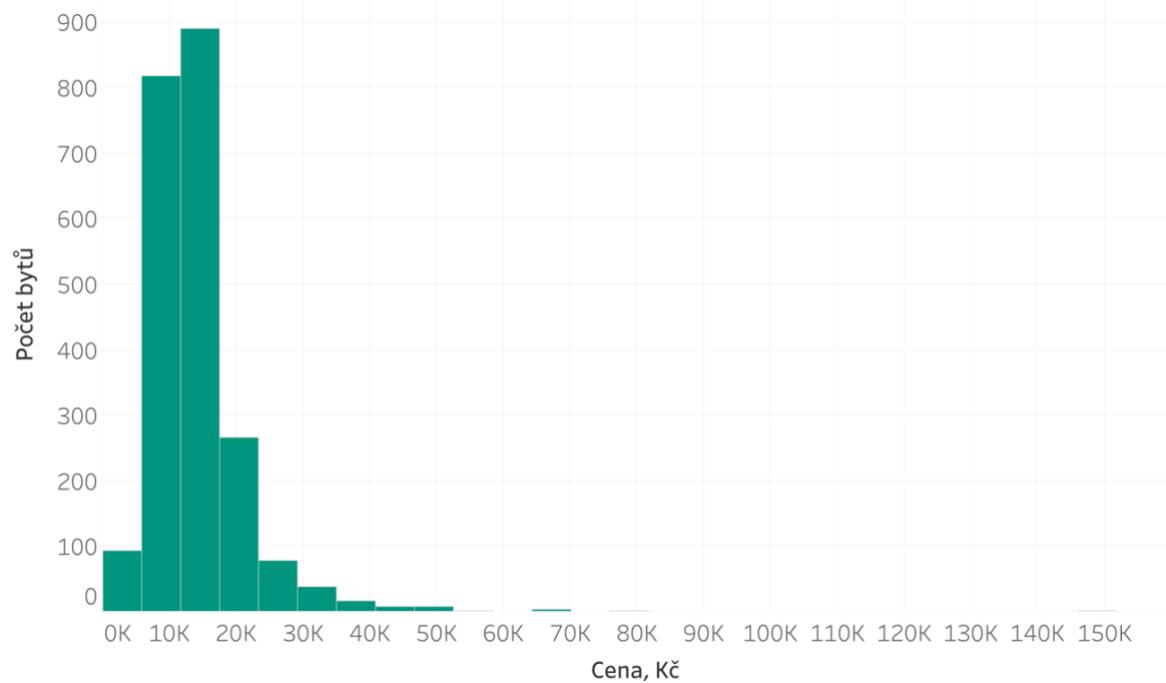
Obrázek 44: Průměrný byt na prodej v České republice (autor)



Obrázek 45: Průměrný byt na prodej v Praze (autor)

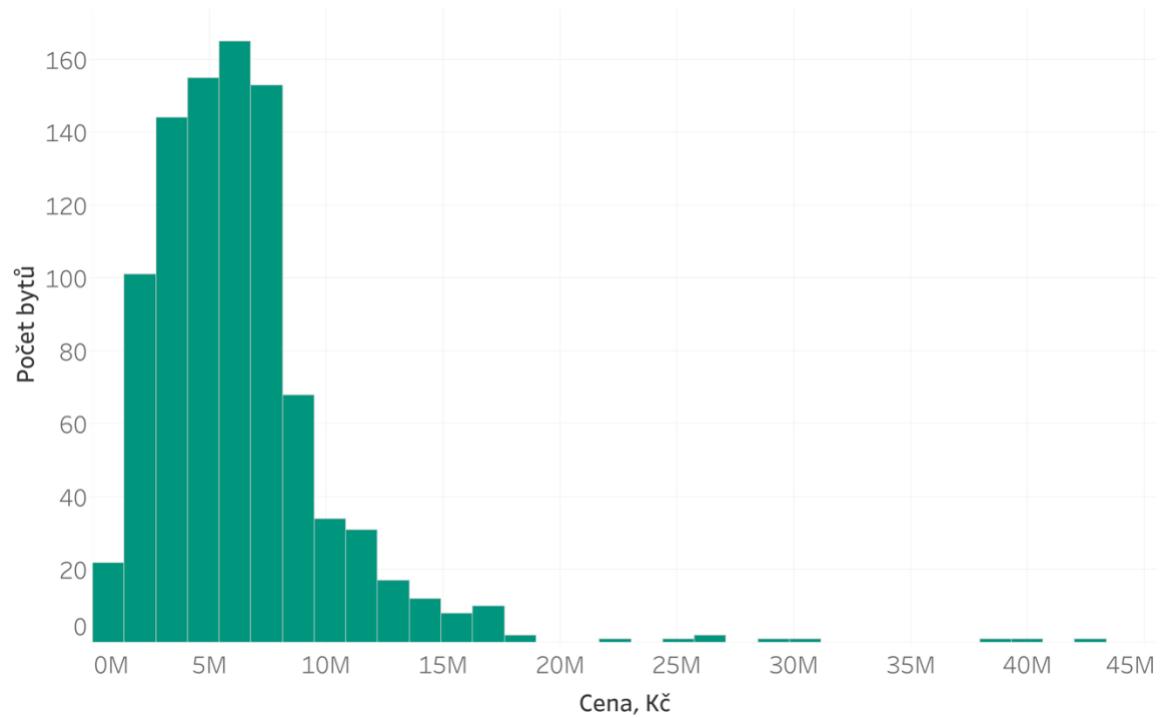
3.6.4 Histogram ceny

Rozložení cen bytů k pronájmu v České republice vypadá následovně. Opět je vidět, že se hodnota nájemného většinou nachází v rozmezí 10 000 až 15 000 Kč.



Obrázek 46: Histogram cen bytů k pronájmu (autor)

U bytů na prodej je toto rozmezí 6 až 8 milionů Kč.



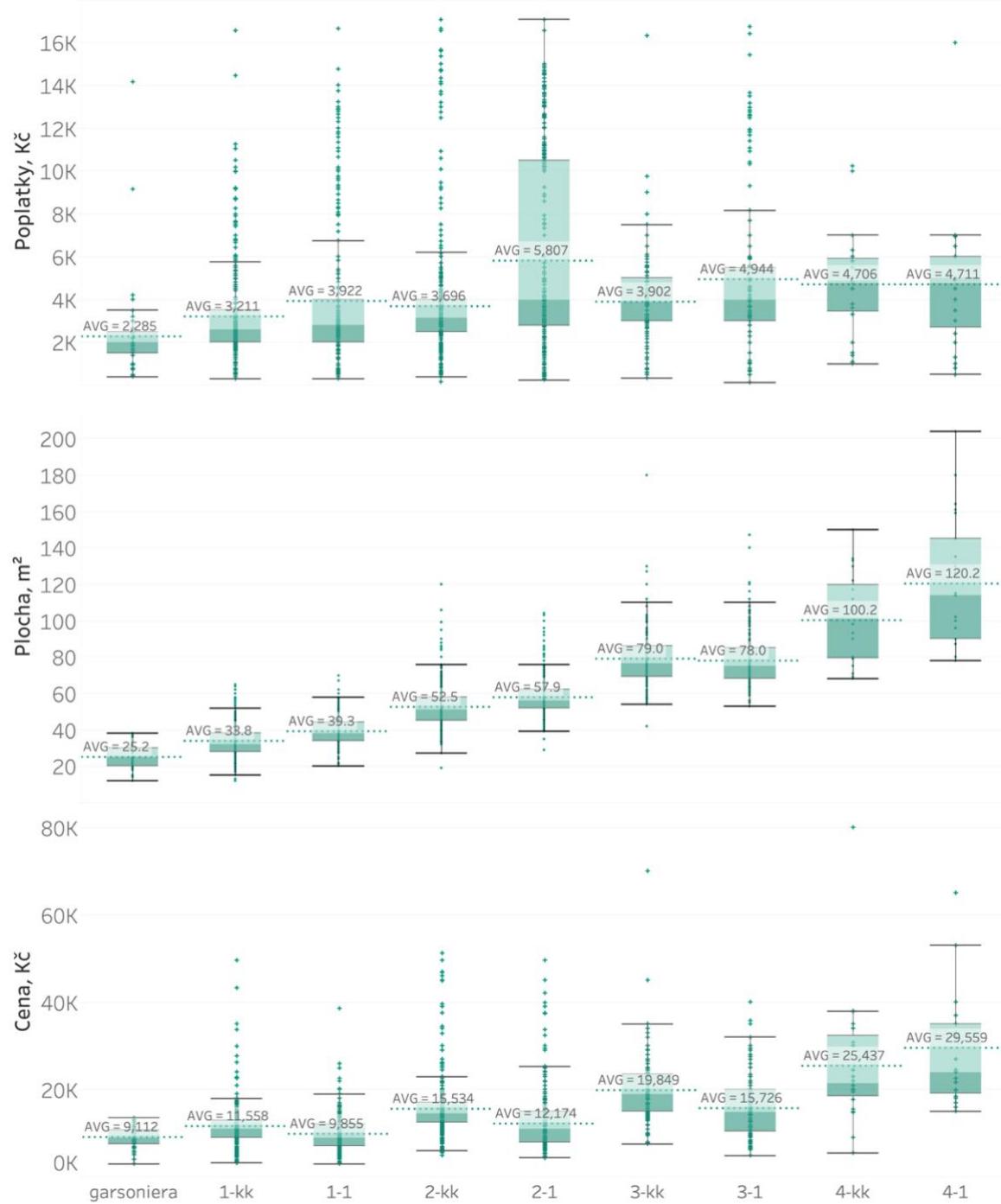
Obrázek 47: Histogram cen bytů na prodej (autor)

3.6.5 Cena a plocha dle dispozicí

Byty k pronájmu

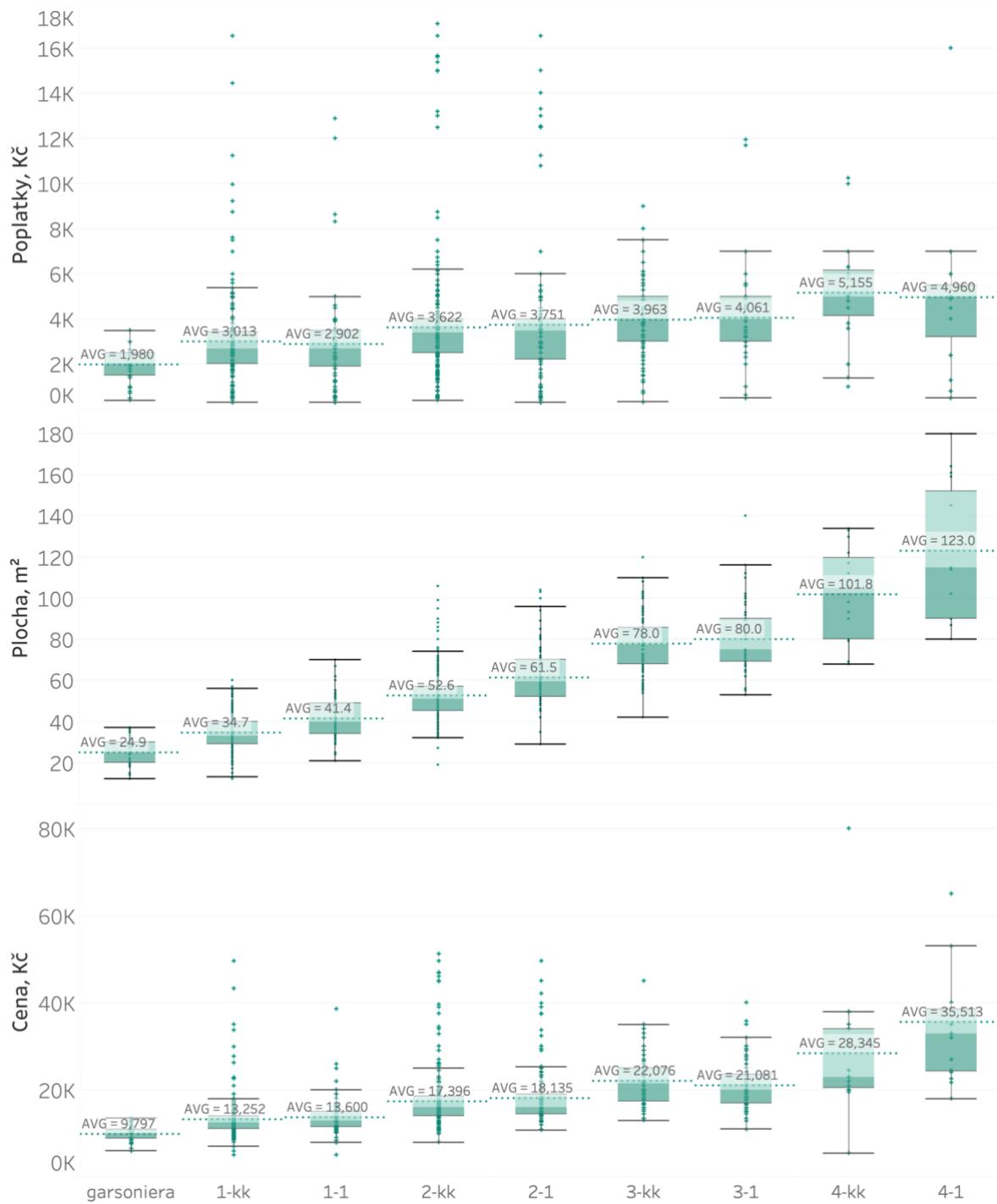
Dále jsem připravil rozložení ceny a plochy podle dispozice bytu. První obrázek reprezentuje celou ČR a tady jsem narazil na několik zajímavostí. První z nich je zvláštní rozptyl poplatků u bytů s dispozicí 2–1, které navíc mají největší střední hodnotu – 5 807 Kč. Kupříkladu byty s dispozicí 4–1 a plochou 120 m² mají poplatky 4 711 Kč. Je dost pravděpodobné, že je to způsobeno malým počtem bytů 4–1 v datasetu, protože pokud místo střední hodnoty budeme porovnávat mediální, tak byty 4–1 a 4–kk budou dražší z pohledu poplatků. Ale na druhou stranu byty 3–kk a 3–1 mají průměrně poplatky nižší anebo stejně velké jako byty 2–1 (i když porovnáváme mediální hodnoty).

Druhou zajímavostí je, že v některých případech stojí větší byty méně. Například cena bytu 1–1 je menší než cena bytu 1–kk. To samé platí i pro dvojice 2–kk a 2–1 a 3–kk a 3–1. Navíc mediální ceny bytů s dispozicí 2–1 (56 m²) a 1–kk (32 m²) jsou stejné – 11 000 Kč.



Obrázek 48: Rozložení cen, plochy a poplatků bytů k pronájmu v ČR (autor)

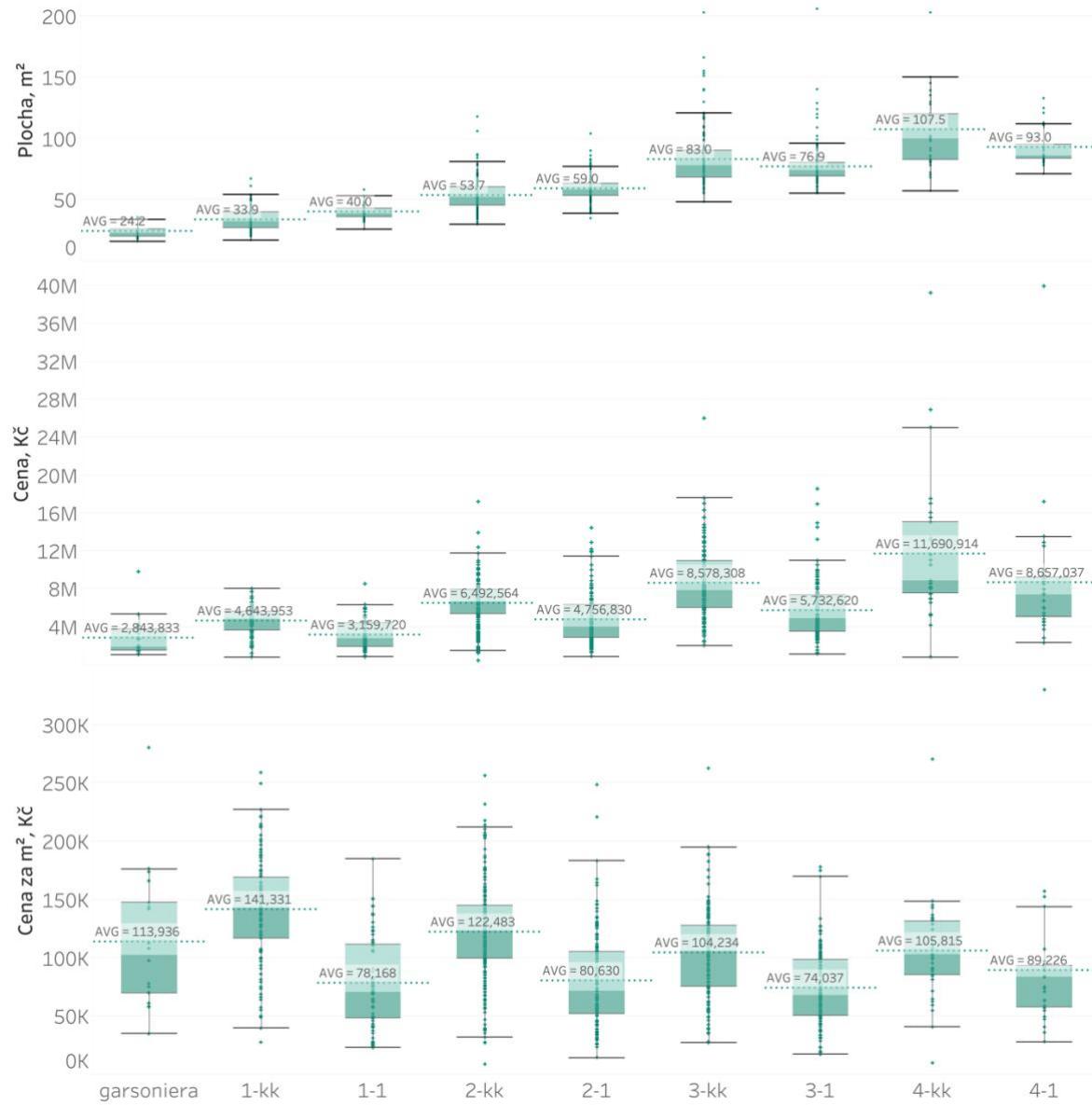
Rozebereme-li ceny dle dispozice v Praze, tady daný nález platí pouze pro dvojici 3-1 a 3-kk. Byty 2-kk a 2-1 ale mají stejnou mediánní hodnotu – 16 000 Kč. Poplatky v Praze se zvyšují současně s plochou, pokud jde o mediánní hodnoty.



Obrázek 49: Rozložení ceny, plochy a poplatků bytů k pronájmu v Praze (autor)

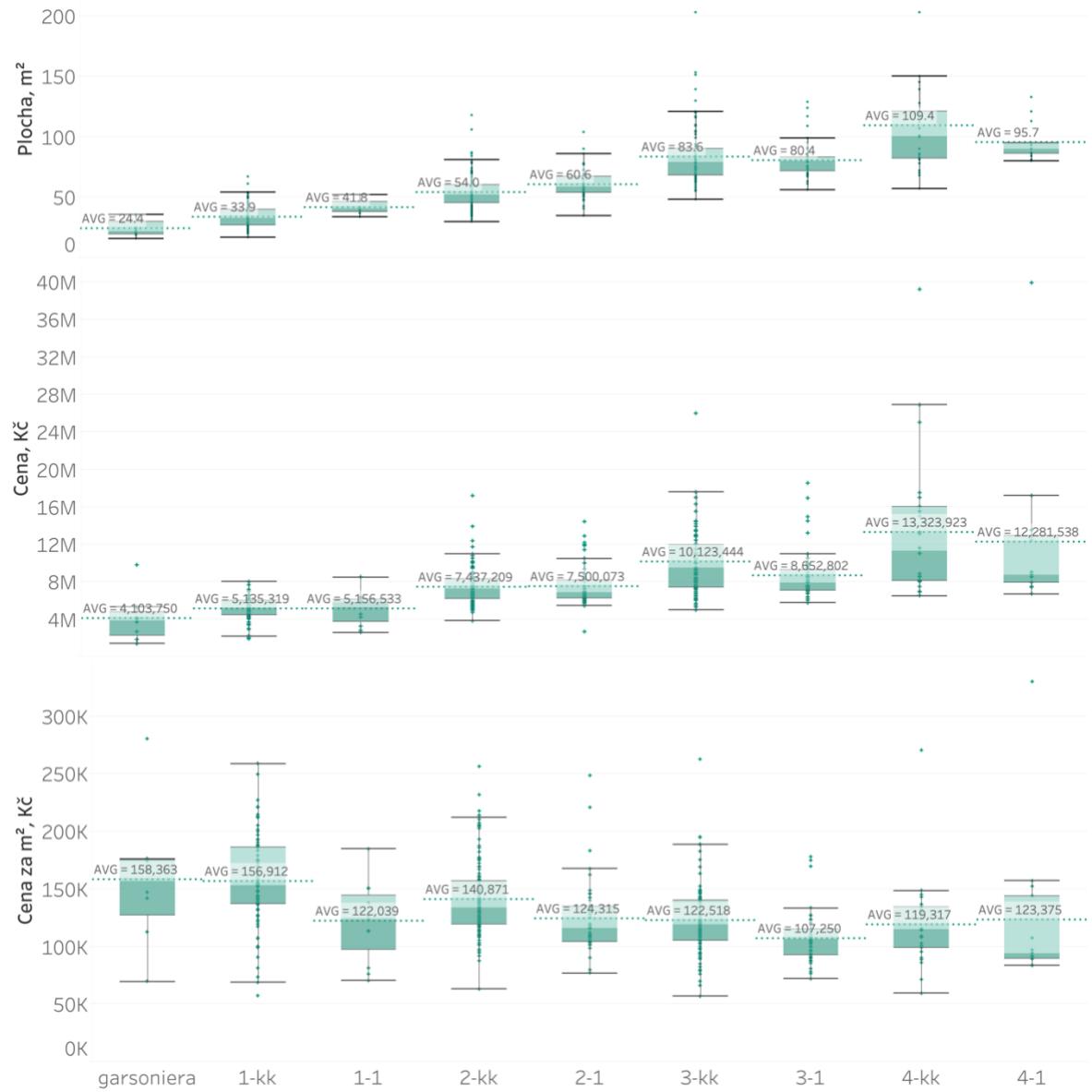
Byty na prodej

Co se týče bytů na prodej v ČR, tak i zde platí, že byty s větší dispozicí jsou levnější. Stojí ale za zmínu, že v datasetu byty s dispozicí 3–1 mají menší plochu ve srovnání s byty 3–kk. To samé platí i pro byty 4–1 a 4–kk. Tím pádem jsem zavedl nový parametr, který vyjadřuje cenu za metr čtvereční. Podle tohoto parametru se předchozí předpoklad potvrzuje.



Obrázek 50: Rozložení ceny, plochy a ceny za m² bytů na prodej v ČR (autor)

V Praze je cena více vyvážená, ale i tak například byty 2–1 mají nižší cenu za metr čtvereční než byty 1–kk.



Obrázek 51: Rozložení ceny, plochy a ceny za m² bytů na prodej v Praze (autor).

Tabulka dole znázorňuje přehled o mediánní ceně za m² v České republice a v Praze.

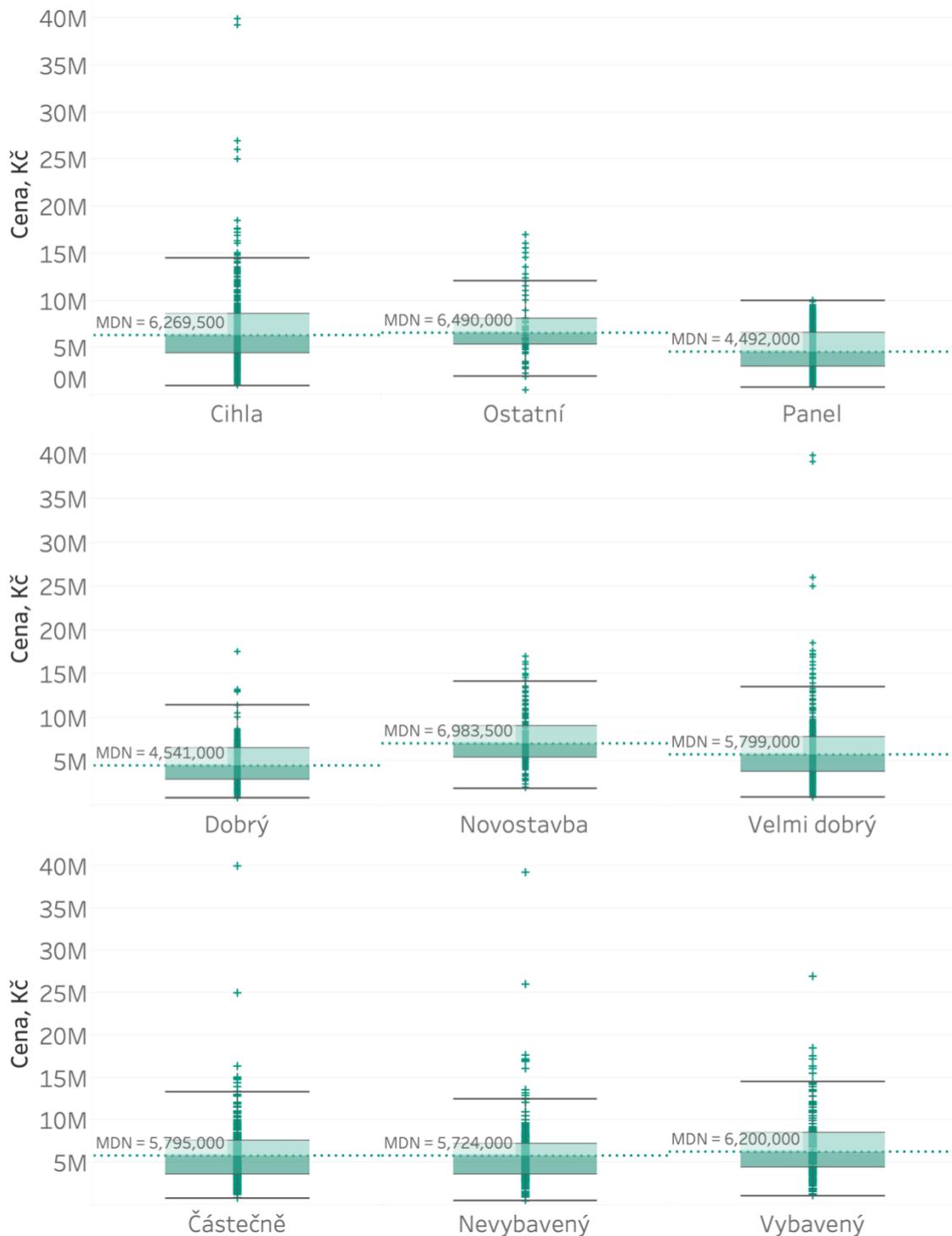
Tabulka 4: Přehled mediánních cen za m² v ČR a Praze (autor)

Dispozice	Cena za m ² (ČR)	Cena za m ² (Praha)
garsoniéra	102 611	156 611
1-kk	143 120	153 000
1-1	70 379	124 797
2-kk	123 520	133 833

2–1	71 429	115 917
3–kk	105 868	119 231
3–1	67 919	106 627
4–kk	102 895	114 744
4–1	83 333	94 086

Je vidět, že nejdražšími byty na prodej v ČR v absolutním ukazateli jsou byty 1–kk, zatímco v Praze jsou to garsoniéry. Hlavním důvodem, proč to tak je, by mohla být vysoká likvidita podobných bytů.

Dále se lze podívat na rozložení ceny s ohledem na vybavenost bytu, stav a typ budovy. Zde ale žádná překvapení nejsou.

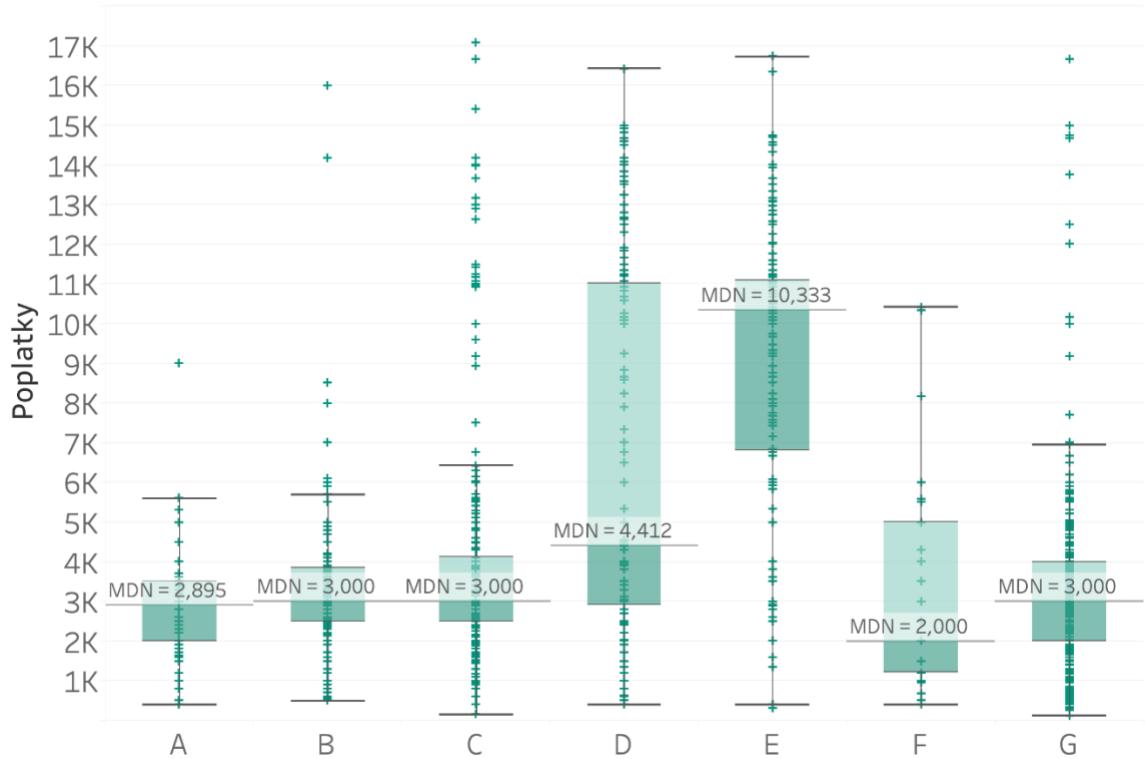


Obrázek 52: Přehled cen bytů dle vybavenosti, stavu a typy budovy (autor)

3.6.6 Poplatky

Třída energetické náročnosti budovy PENB slouží k porovnání objektů mezi sebou na základě zatřídění do klasifikačních tříd A (mimořádně úsporná) – G (mimořádně nehospodárná) (34). Z toho vyplývá, že čím vyšší je třída PENB, tím nižší by měly být poplatky. Ze získaných dat to ale tak není. Na obrázku dole je vidět, že byty v budovách třídy

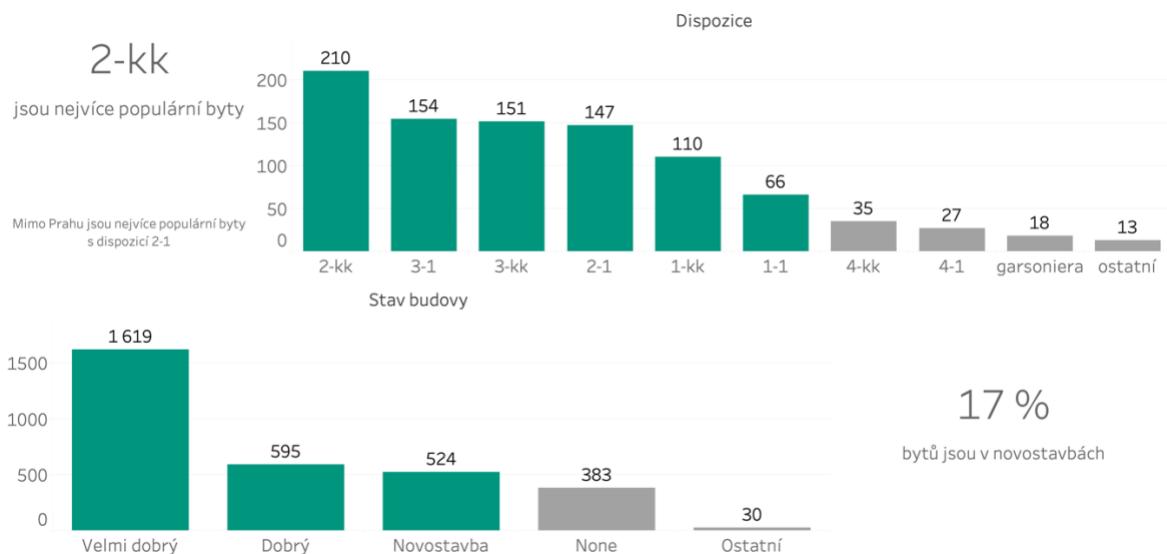
G mají stejné poplatky jako byty v budovách třídy B či C. Nejnižší poplatky jsou v budovách třídy F.



Obrázek 53: Přehled poplatků dle energetické třídy PENB (autor)

3.6.7 Populární byty

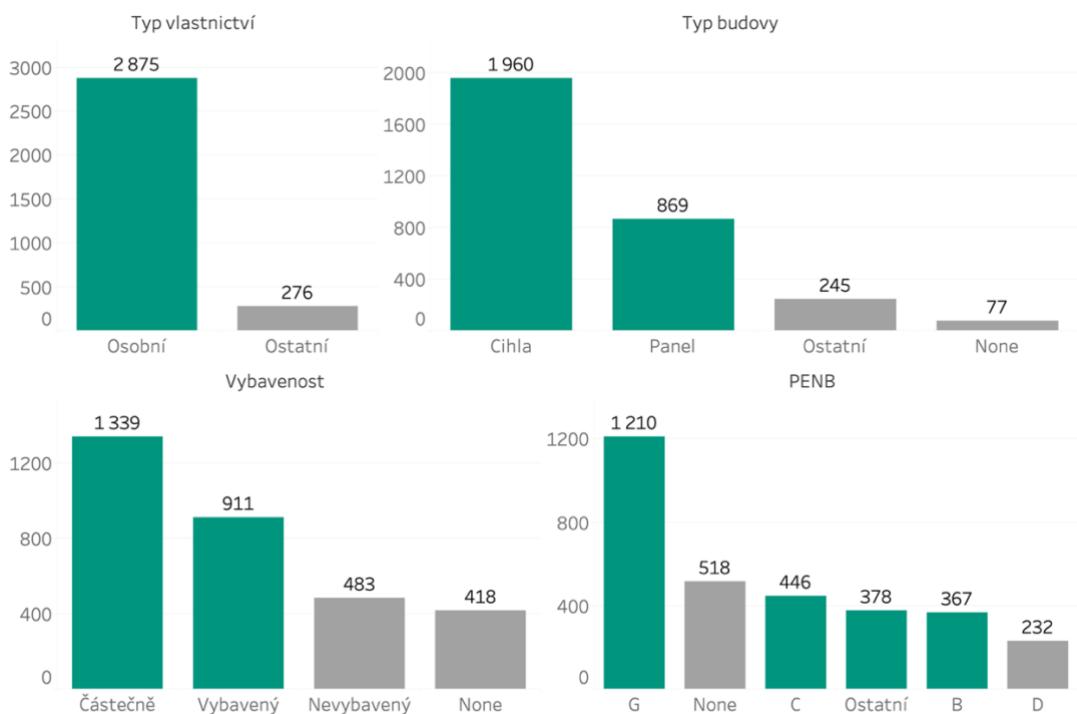
Ze všech nabídek (pronájem a prodej) jsou nejpopulárnější byty s dispozicí 2–kk. Většina budov, kde se nachází byty, je v dobrém stavu. Novostavby představují 17 % z celkového počtu nabídek.



Obrázek 54: Populární byty (autor)

3.6.8 Informace o budovách

Následně se podíváme na vizualizaci atributů popisujících byt a budovu, kde se byt nachází. Více než 90 % inzerátů představují byty v osobním vlastnictví. Do jiných typů bychom mohli zařadit družstevní vlastnictví, které se ovšem v datasetu přímo nevyskytuje. Nejčetnějším typem budovy je cihla s energetickou třídou G dle PENB. Třída G znamená nejhorší energetickou náročnost budovy a je dost překvapivé, že zhruba 40 % bytů v ČR se nachází v podobných budovách. Určitě získaný dataset nepředstavuje přehled o všech bytech, ale i tak se dá tento nález považovat za zajímavý.



Obrázek 55: Přehled inzerátů dle informací o budově (autor)

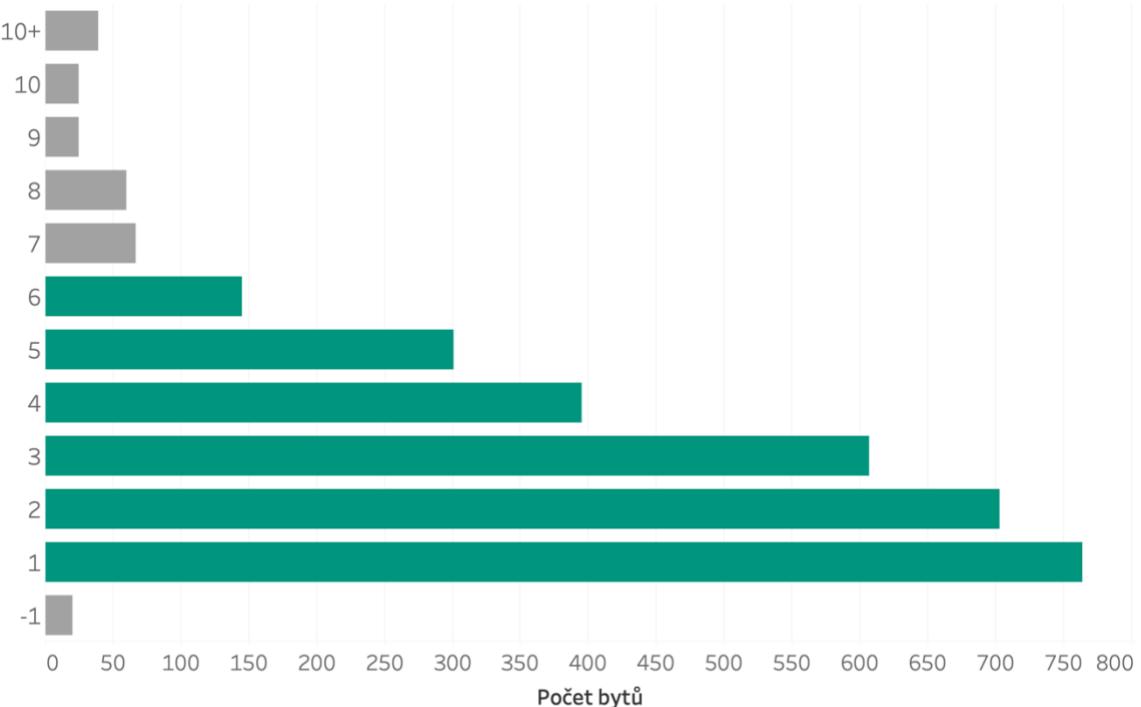
Většina bytů je buď vybavená částečně, anebo je vybavená kompletně. Co jsem slyšel, v mnohých evropských státech je obvyklé pronajímat byt bez nábytku a myslí jsem, že nevybavené byty jsou většinou v novostavbách anebo v budovách, které se právě nacházejí ve výstavbě. Pro ověření svého předpokladu jsem vytvořil tabulku, do níž jsem uvedl počty nevybavených bytů podle stavu budovy a podíl takových bytů na celkovém počtu bytů v budovách ve stejném stavu. Vyšlo tak, že největší podíl mají budovy ve špatném stavu a ve výstavbě. Trochu zvláštní je, že se budovy ve výstavbě v datasetu vyskytují i s hodnotami jinými než „nevybavený“. Budovy ve stavu „dobrý“ a „novostavba“ mají téměř stejný podíl nevybavených bytů (~18 %). O něco menší podíl mají i budovy ve stavu „velmi dobrý“, takže lze dospět k závěru, že nevybavené byty se vyskytují skoro stejně často jak v novostavbách, tak i v budovách, které se nacházejí ve stavu „dobrý“ a „velmi dobrý“.

Tabulka 5: Přehled nevybavených bytů dle stavu budovy (autor)

Stav budovy	Vybavenost	Počet bytů	Podíl
Velmi dobrý	Nevybavený	226	14 %
Dobrý	Nevybavený	111	18,7 %
Novostavba	Nevybavený	91	17,4 %
None	Nevybavený	40	10,4 %
Špatný	Nevybavený	10	52,6 %
Ve výstavbě	Nevybavený	5	45,5 %

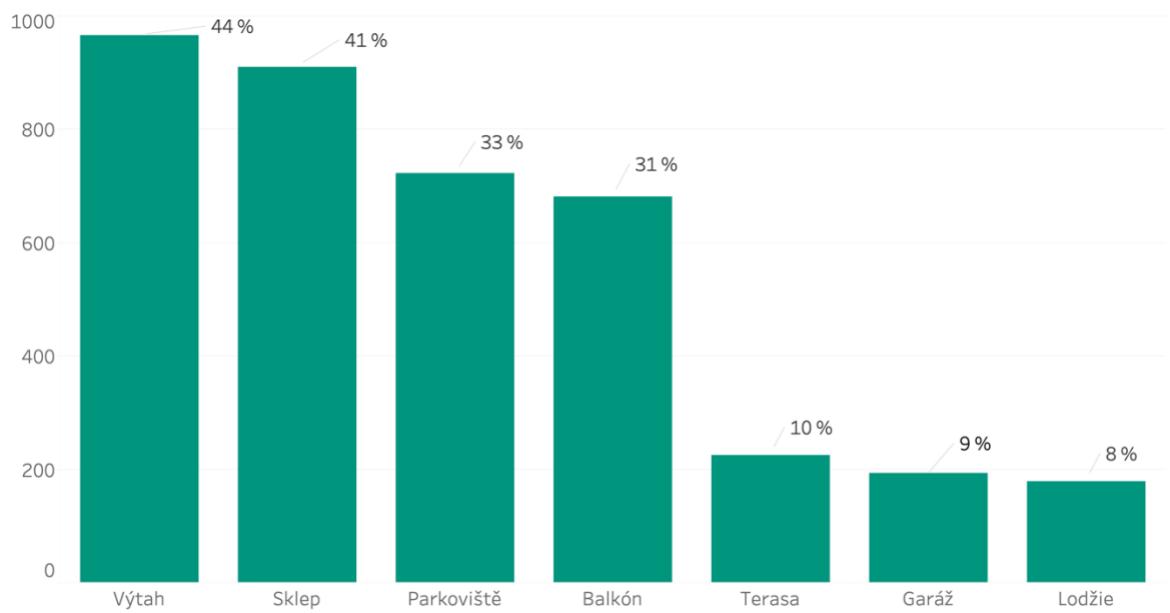
3.6.9 Popis bytu

Jak již bylo zmíněno, většina bytů se nachází na podlaží 1–6. Vyskytují se ale i byty na vyšších patrech a také byty v přízemí.



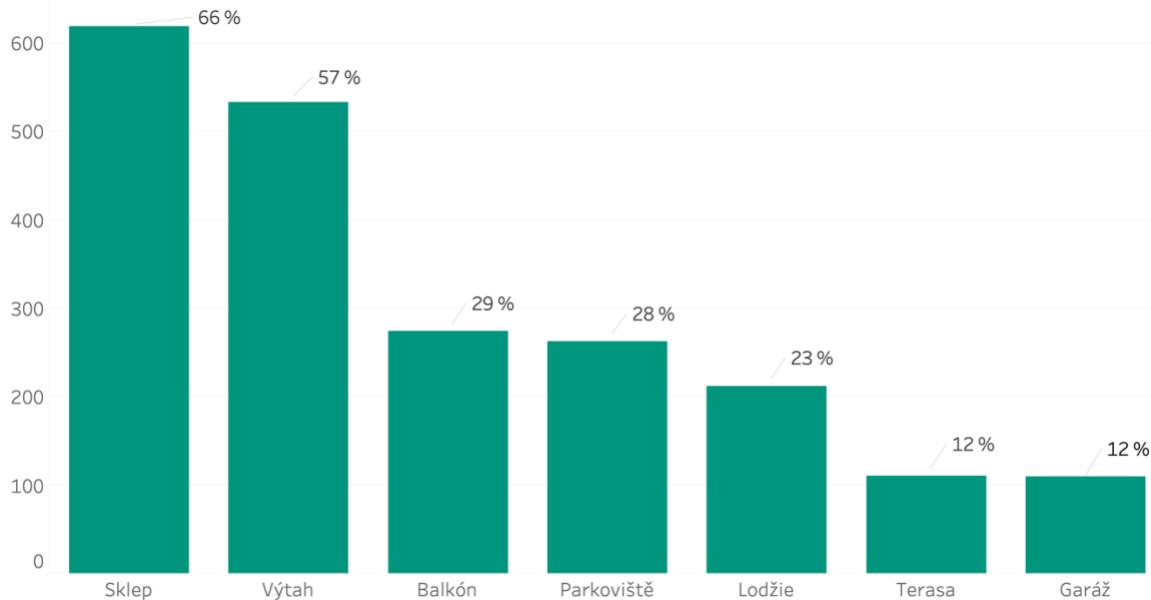
Obrázek 56: Přehled inzerátů dle podlaží (autor)

Nejčastěji se v popisu bytů k pronájmu můžeme setkat s výtahem, sklepem, parkovištěm a balkónem.



Obrázek 57: Charakteristika bytů k pronájmu (autor)

U bytů na prodej jsou tyto charakteristiky stejné, pouze v trochu jiném pořadí.



Obrázek 58: Charakteristika bytů na prodej (autor)

3.6.10 Korelace atributů

Pro výpočet korelace mezi atributy jsem provedl další zpracování dat, a to konkrétně převod kategoriálních atributů do tzv. dummies proměnných.

```
df_all_part = df_all[['price', 'key_offer_type', 'facilities', 'condition', 'property_type',
                      'building_type', 'penb']]

df_all_part_dummies = pd.get_dummies(df_all_part[['facilities', 'condition', 'property_type',
                                                 'building_type', 'penb']])

df_all_part.drop(['facilities', 'condition', 'property_type', 'building_type', 'penb'], axis=1, inplace=True)
df_all_part = pd.concat([df_all_part, df_all_part_dummies], axis=1)

df_all_part.drop(['facilities_Nevybavený', 'condition_None', 'property_type_Ostatní',
                  'building_type_Ostatní', 'building_type_None', 'penb_None'], axis=1, inplace=True)

df_all_part.head()
```

facilities_Nevybavený	facilities_Vybavený	facilities_Částečně	condition_Dobrý	condition_Novostavba	condition_Ve výstavbě	condition_Velmi dobrý	condition_Špatný	property_ty
0	1	0	0	0	0	0	0	0
0	1	0	0	0	0	1	0	0
0	1	0	1	0	0	0	0	0
0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0

Obrázek 59: Převod vybraných atributů do dummies proměnných (autor)

Výsledky korelační analýzy bohužel nejsou význačné, pokud jde o korelací ceny a jiných atributů. Největší hodnoty Pearsonova korelačního koeficientu mezi prodejnými nabídkami mají:

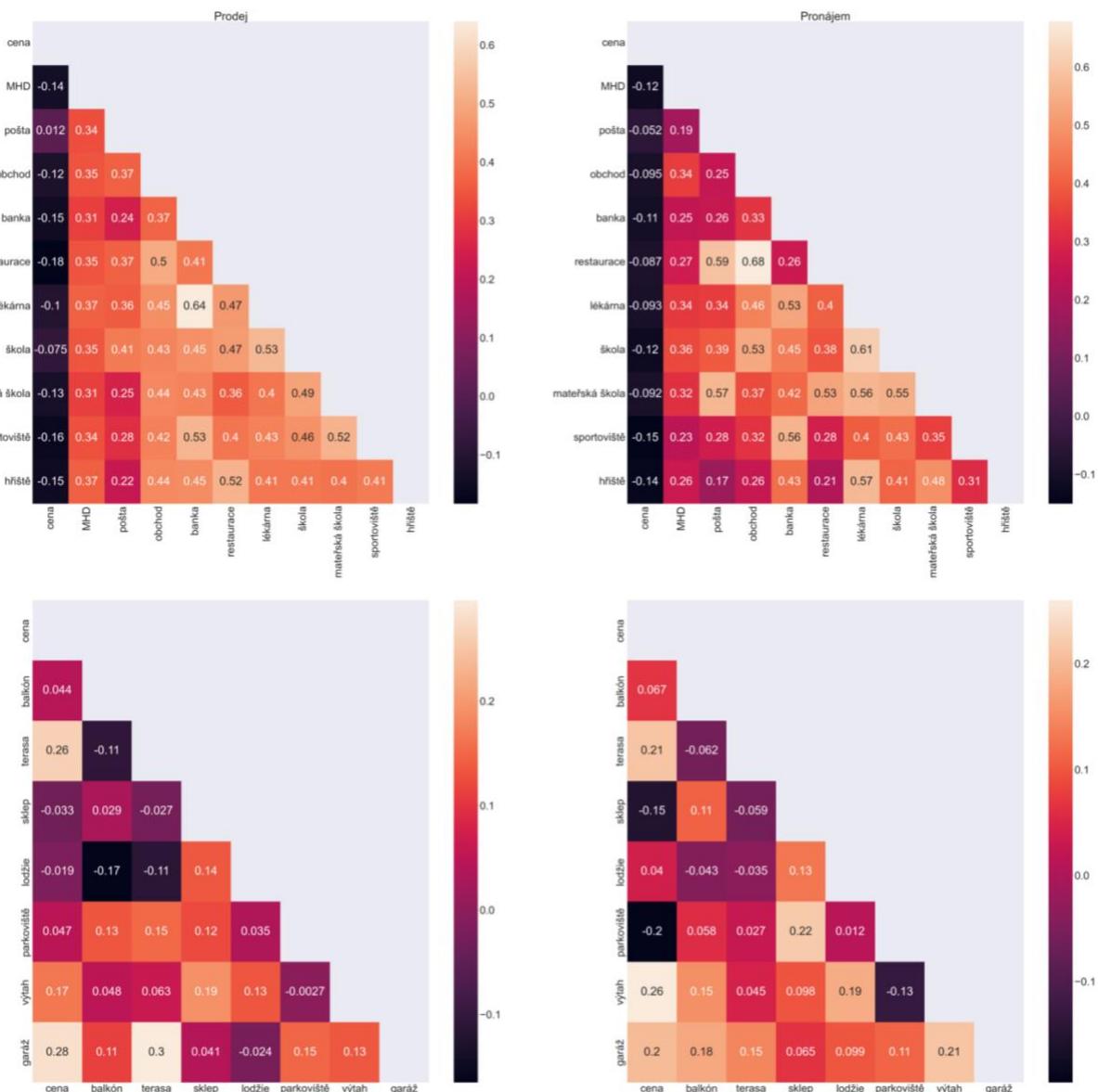
- cena a typ budovy (Panel): **-0,3**
- cena a garáž: **0,28**
- cena a terasa: **0,26**

Nabídky o pronájmu to mají trochu jinak:

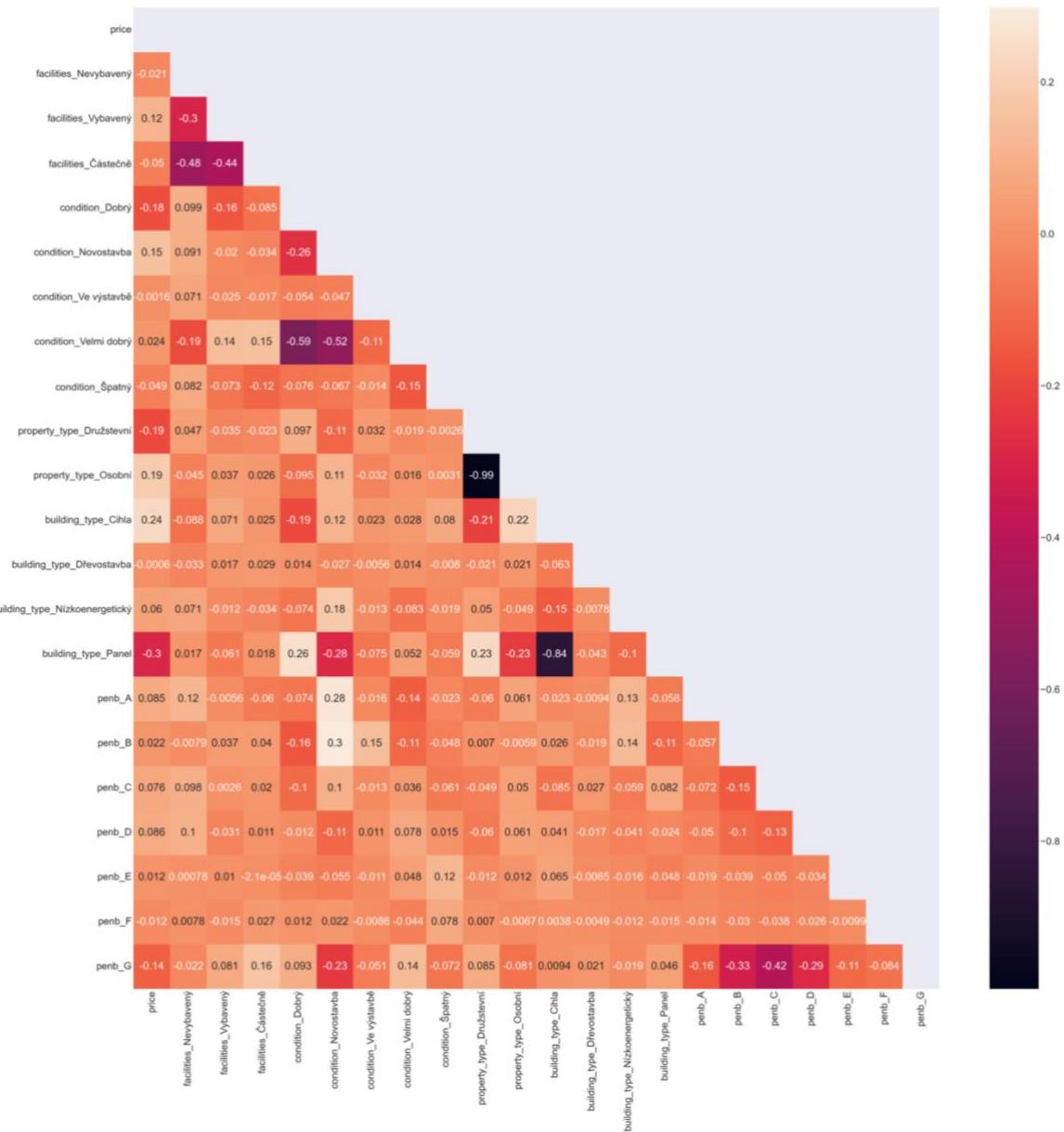
- cena a výtah: **0,26**
- cena a vybavenost (Vybavený): **0,23**
- cena a stav budovy (Dobrý): **-0,25**
- cena a PENB (E): **-0,25**

Z větších hodnot korelace lze vyčlenit následující atributy:

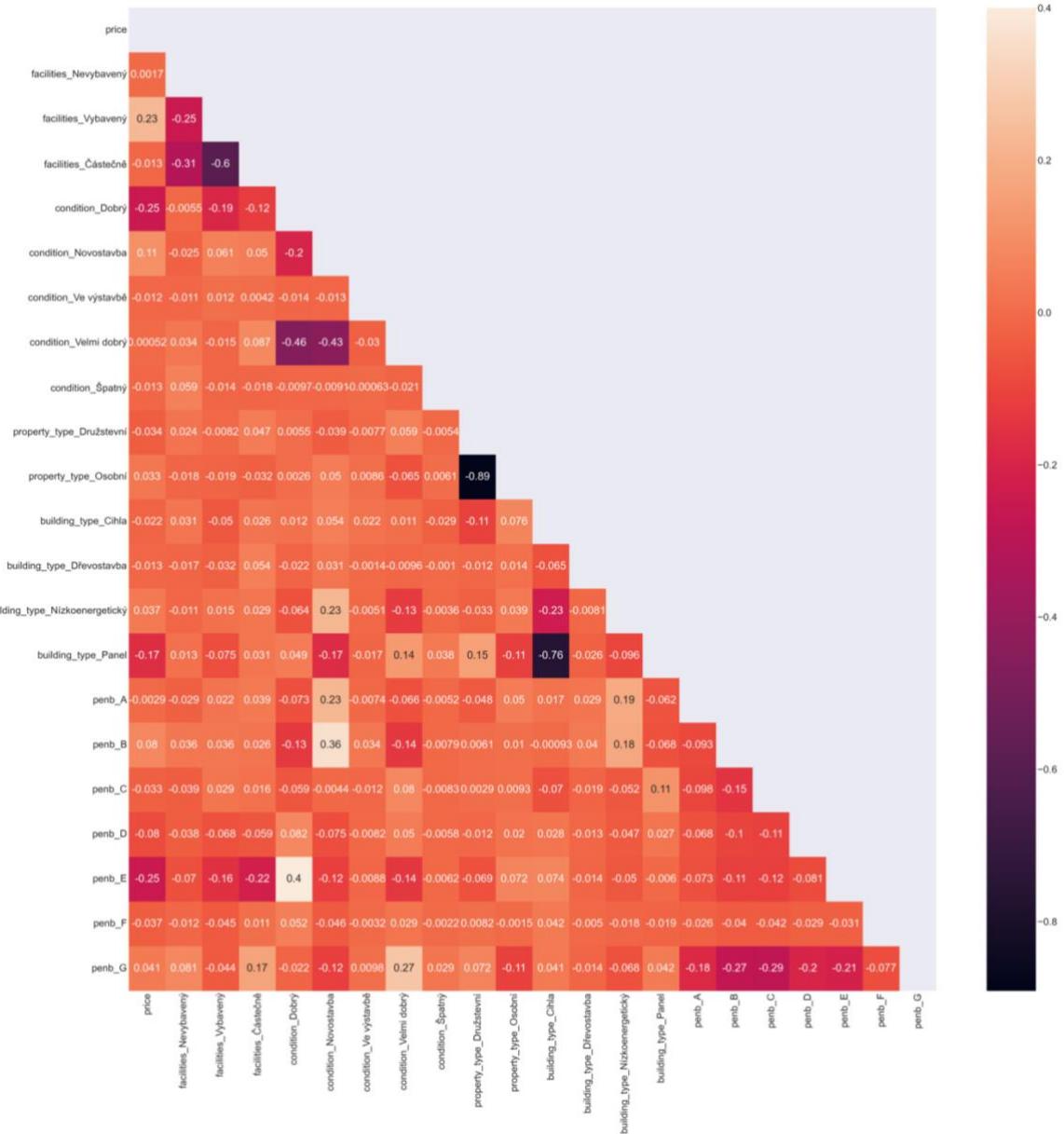
- lékárna a banka: **0,64** (byty na prodej)
- lékárna a škola: **0,61** (byty k pronájmu)
- restaurace a obchod: **0,68** (byty k pronájmu)
- PENB (E) a stav budovy (Dobrý): **0,4** (byty k pronájmu)
- PENB (B) a stav budovy (Novostavba): **0,37** (byty k pronájmu)



Obrázek 6o: Korelace vybraných atributů (autor)

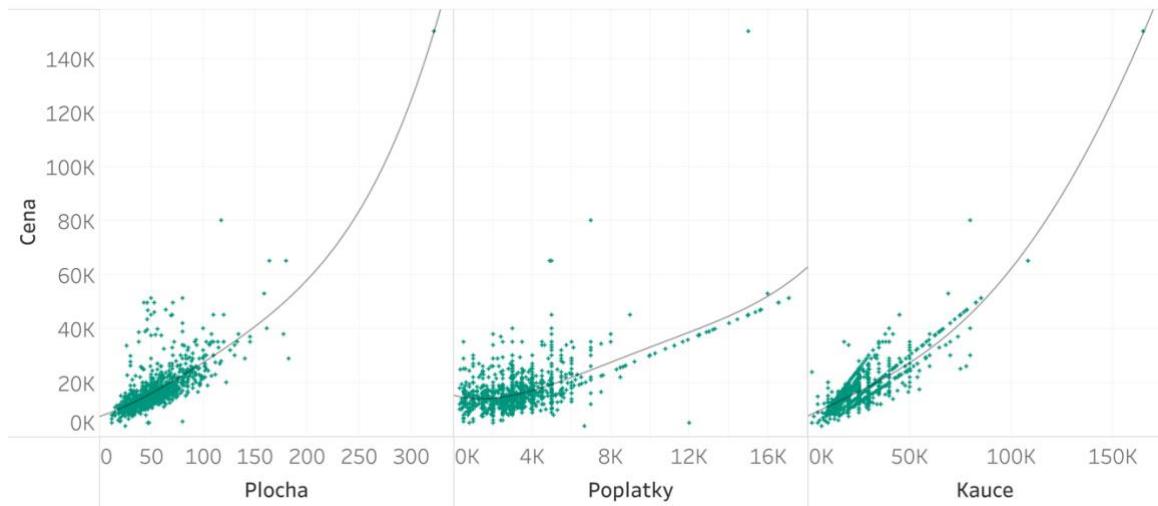


Obrázek 61: Korelace vybraných atributů u bytů na prodej (autor)



Obrázek 62: Korelace vybraných atributů u bytů k pronájmu (autor)

Následující obrázek pak znázorňuje korelaci ceny a plochy, poplatků a kauce v bytech, které se nacházejí v Praze. Tyto korelace jsou popsány polynomiálním trendem.



Obrázek 63: Korelace ceny, plochy, poplatků a kauce (autor)

3.7 Modelování

V rámci dané kapitoly se budu zabývat modelováním. Rozhodl jsem se soustředit na 2 typy úloh: regresní a klasifikační. Modelování zase probíhalo v prostředí Jupyter Notebook a ještě před samotnou tvorbou modelů jsem připravil pomocné funkce.

První funkce převádí kategoriální proměnné do dummies proměnných, aby se pak dalo tyto proměnné využívat v modelu.

```
def convert_to_dummies(df: pd.DataFrame, columns: list) -> pd.DataFrame:
    """
    Converts dataframe categorical columns to dummies
    """

    df_copy = df.copy()
    df_dummies = pd.get_dummies(df_copy[columns])
    df_copy.drop(columns, axis=1, inplace=True)
    df_copy = pd.concat([df_copy, df_dummies], axis=1)
    columns_to_drop = []
    for column in df_copy:
        # Remove meaningless columns
        if 'none' in column.lower() or 'ostatni' in column.lower():
            columns_to_drop.append(column)
    df_copy.drop(columns_to_drop, axis=1, inplace=True)
    df_dummies = df_copy
    return df_dummies
```

Obrázek 64: Funkce pro převod atributů do dummies proměnných (autor)

Další funkce vypisují evaluační metriky pro různé modely, a to buď pro jedno spuštění, anebo střední hodnotu z N-iterací.

```

def print_reg_metrics(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 mean_r2_score(X: pd.Series, Y: pd.Series, model, n_iter: int) -> None:
"""
Prints mean R2 score of regression model based on N interations.
"""

r2_scores = []

for _ in range(n_iter):
    x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size = 0.2)

    model.fit(x_train, y_train)

    y_prediction = model.predict(x_test)

    score = r2_score(y_test, y_prediction)

    if score > 0:
        r2_scores.append(score)

mean_r2_score = sum(r2_scores) / len(r2_scores)

print(f'Mean R2 score based on {n_iter} iterations ----- {round(mean_r2_score, 3)} ({int(round(mean_r2_score*100, 0))} %)')

```

Obrázek 65: Pomocné funkce pro modelování 1 (autor)

```

def eval_lasso_ridge(X: pd.Series, Y: pd.Series, model, plot_feature_importance=True) -> None:
"""
Prints the evaluation of Lasso / Ridgred regression model.
Also plots features importance barchart.
"""

model_name = 'LassoCV model' if 'LassoCV' in str(model) else 'RidgeCV model'

print(f'Best alpha using {model_name} ----- {round(model.alpha_, 2)}')
print(f'Best score using {model_name} ----- {round(model.score(X, Y), 3)} ({int(round(model.score(X, Y), 0))} %)')
coef = pd.Series(model.coef_, index = X.columns)

print(f'Number of choosen features ----- {str(sum(coef != 0))}')
print(f'Number of droped features ----- {str(sum(coef == 0))}')

imp_coef = coef.sort_values()

if plot_feature_importance:
    plt.rcParams['figure.figsize'] = (12, 20)
    plt.rc('font', size=15)
    imp_coef.plot(kind='barh', color="#00957d")
    plt.title(f'Feature importance using {model_name}')

```



```

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')

```

Obrázek 66: Pomocné funkce pro modelování 2 (autor)

3.7.1 Regresní úloha

Pro regresní úlohu jsem stanovil dva cílové atributy: *cena* a *poplatky*. Dataset byl opět rozdělen na dvě části: byty k pronájmu a byty na prodej. Pro daný typ úlohy jsem použil lineární regresi, lasso regresi a hřebenovou (ridge) regresi.

Predikce ceny pronájmu

Jako první krok jsem převedl všechny atributy do dummies proměnných a následně jsem standardizoval data pomocí střední hodnoty a standardní odchylky. Výsledná data byla rozdělena na trénovači a testovací množiny v poměru 80 : 20.

```
df_rent = df[df['key_offer_type'] == 'pronajem']
df_rent.drop(['key_offer_type'], axis=1, inplace=True)

df_rent_dummies = convert_to_dummies(df_rent, ['facilities', 'condition', 'property_type', 'building_type', 'penb',
                                              'key_disposition', 'floor'])

X_rent = df_rent_dummies.drop('price', axis=1)
Y_rent = df_rent_dummies['price']

X_rent = (X_rent - X_rent.mean()) / X_rent.std()

x_rent_train, x_rent_test, y_rent_train, y_rent_test = train_test_split(X_rent, Y_rent, test_size = 0.2, random_state=42)
```

Obrázek 67: Příprava a rozdělení dat pro regresi (autor)

Dalším krokem byla vytvořena lineární regrese. Střední hodnota R² po 100 iteracích je 75 %.

```
lin_reg_rent = LinearRegression()
lin_reg_rent.fit(x_rent_train, y_rent_train)

y_rent_prediction = lin_reg_rent.predict(x_rent_test)

print_reg_metrics(y_rent_test, y_rent_prediction)

R2 ----- 0.834 (83 %)
Mean squared error ----- 9081387.4
Root mean squared error ---- 3013.5

mean_r2 = mean_r2_score(X_rent, Y_rent, lin_reg_rent, 100)
mean_r2

Mean R2 score based on 100 iterations ---- 0.754 (75 %)
```

Obrázek 68: Vytvoření a evaluace lineární regrese (autor)

U lasso regrese jsem nastavil parametr křížové validace na ten výchozí (5) s maximálním počtem iterací 10 000. Úspěšnost lasso regrese je skoro stejná – 76 %.

```

lasso_reg_rent = LassoCV(cv=5, random_state=17, max_iter=10000)
lasso_reg_rent.fit(x_rent_train, y_rent_train)

y_rent_prediction = lasso_reg_rent.predict(x_rent_test)
print_reg_metrics(y_rent_test, y_rent_prediction)

R2 ----- 0.84 (84 %)
Mean squared error ----- 8708558.5
Root mean squared error ----- 2951.0

mean_r2 = mean_r2_score(X_rent, Y_rent, lasso_reg_rent, 100)
mean_r2

Mean R2 score based on 100 iterations ----- 0.763 (76 %)

eval_lasso_ridge(X_rent, Y_rent, lasso_reg_rent, plot_feature_importance=False)

Best alpha using LassoCV model ----- 89.66
Best score using LassoCV model ----- 0.786 (79 %)
Number of choosen features ----- 36
Number of droped features ----- 24

```

Obrázek 69: Vytvoření a evaluace lasso regrese (autor)

U lasso regrese se navíc dá zjistit důležitost jednotlivých atributů. V případě predikce ceny byly za důležité označeny následující atributy: deposit, plocha, výtah, poplatky, vybavenost (Vybavený), terasa atd.



Obrázek 70: Určení důležitosti jednotlivých atributů pomocí lasso regrese (autor)

Hřebenová regrese měla také skoro stejnou hodnotu koeficientu determinace.

```

ridge_reg_rent = RidgeCV()
ridge_reg_rent.fit(x_rent_train, y_rent_train)

y_rent_prediction = ridge_reg_rent.predict(x_rent_test)

print_reg_metrics(y_rent_test, y_rent_prediction)

R2 ----- 0.834 (83 %)
Mean squared error ----- 9039289.9
Root mean squared error ---- 3006.5

mean_r2 = mean_r2_score(X_rent, Y_rent, ridge_reg_rent, 100)
mean_r2

Mean R2 score based on 100 iterations ---- 0.756 (76 %)

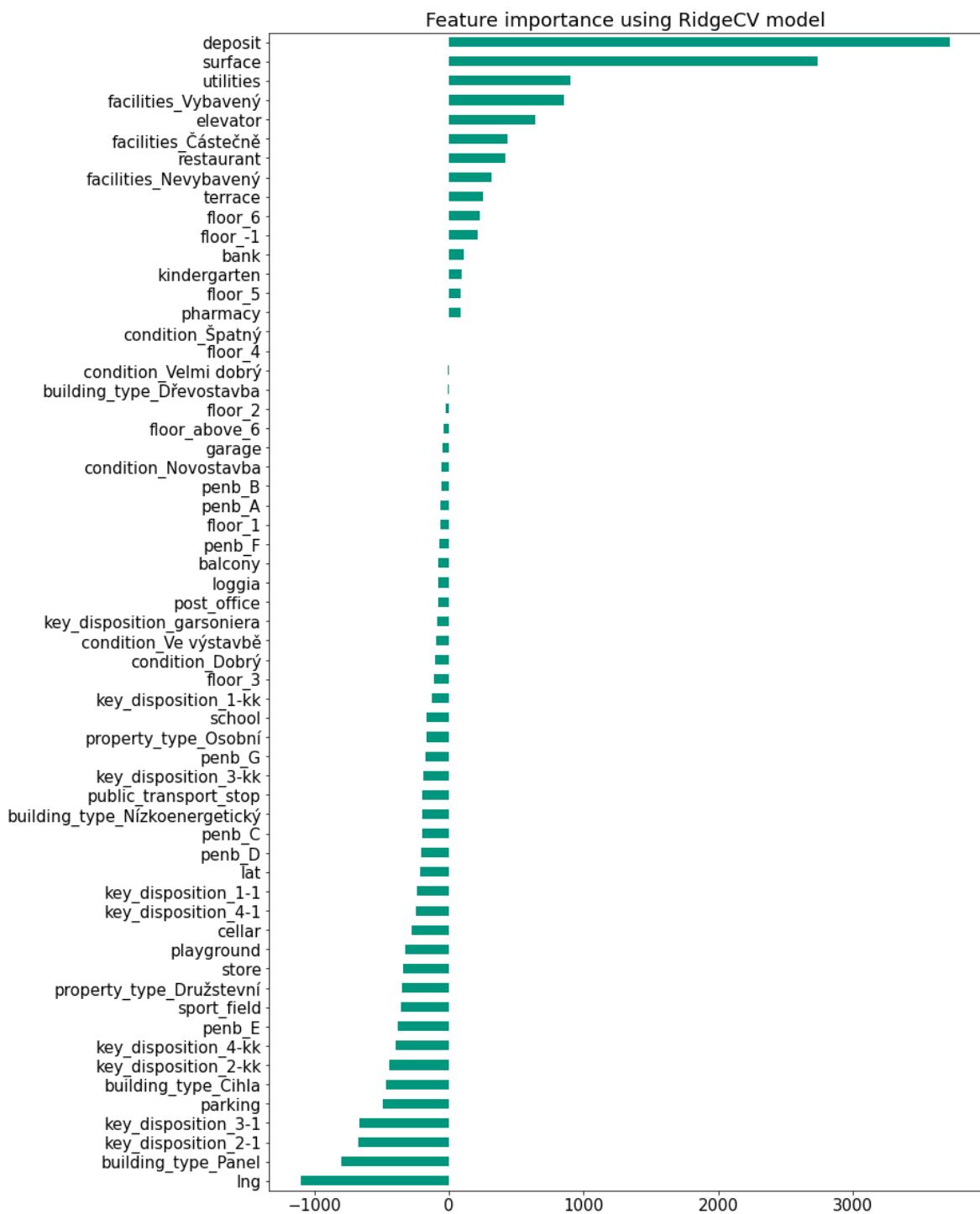
eval_lasso_ridge(X_rent, Y_rent, ridge_reg_rent, plot_feature_importance=True)

Best alpha using RidgeCV model ----- 10.0
Best score using RidgeCV model ----- 0.79 (79 %)
Number of choosen features ----- 60
Number of droped features ----- 0

```

Obrázek 71: Vytvoření a evaluace hřebenové regrese (autor)

Na druhou stranu označila za důležité trochu jiné atributy.



Obrázek 72: Určení důležitosti jednotlivých atributů pomocí hřebenové regrese (autor)

Ze tří představených modelů nelze jednoznačně určit nejlepší, ale zároveň lze říct, že každý z nich je relativně dobrý a lze je pokusit využít pro predikci ceny pronájmu.

Tabulka 6: Výsledky predikce ceny bytu k pronájmu pomocí regresí (autor)

Model	R² (100 iterací)
Lineární regrese	0,75
Lasso regrese	0,76
Ridge regrese	0,76

Predikce poplatků

Tabulka 7: Výsledky predikce poplatků bytu k pronájmu pomocí regresí (autor)

Model	R² (100 iterací)
Lineární regrese	0,59
Lasso regrese	0,57
Ridge regrese	0,57

K atributům důležitým pro predikci poplatků patří např. cena, penb (E), penb (D) a parkoviště.

Predikce ceny prodeje

Tabulka 8: Výsledky predikce ceny bytu na prodej pomocí regresí (autor)

Model	R² (100 iterací)
Lineární regrese	0,55
Lasso regrese	0,56
Ridge regrese	0,55

Atributy, které jsou důležité pro predikci ceny bytu na prodej, zahrnují plochu, výtah, penb (A), vybavenost (Vybavený), garáž a další.

3.7.2 Klasifikační úloha

Pro řešení klasifikační úlohy byly zvoleny následující metody:

- Support Vector;
- K-Nearest Neighbors;
- Gaussian Naive Bayes;
- Decision Tree;
- Random Forest.

Tentokrát jsem jako cílové atributy zvolil PENB, typ a stav budovy. Trénování a testování modelu probíhalo jak na datasetu, obsahujícím všechny typy inzerátů, tak i na datasetu, který obsahoval nabídky pouze jednoho typu.

Výsledky predikce energetické třídy dle PENB nejsou tak význačné – nejlepší model, kterým je Random Forest, dosáhl přesnosti 54 %, což není nic moc.

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

***** RENT *****

SVC ----- 0.419 (42 %)
KNeighborsClassifier ----- 0.354 (35 %)
GaussianNB ----- 0.286 (29 %)
DecisionTreeClassifier ----- 0.371 (37 %)
RandomForestClassifier ----- 0.505 (50 %)

***** SELL *****

SVC ----- 0.492 (49 %)
KNeighborsClassifier ----- 0.348 (35 %)
GaussianNB ----- 0.139 (14 %)
DecisionTreeClassifier ----- 0.426 (43 %)
RandomForestClassifier ----- 0.536 (54 %)

***** ALL *****

SVC ----- 0.379 (38 %)
KNeighborsClassifier ----- 0.344 (34 %)
GaussianNB ----- 0.141 (14 %)
DecisionTreeClassifier ----- 0.386 (39 %)
RandomForestClassifier ----- 0.532 (53 %)
```

Obrázek 73: Evaluace modelů pro predikci PENB (autor)

Predikce typu budovy se naopak dá považovat za více méně úspěšnou, neboť se v několika případech podařilo dosáhnout hodnoty 70–80 %.

```

eval_classification_by_offer_type(models, dfs, dummies_attr, 'building_type')

***** RENT *****

SVC ----- 0.673 (67 %)
KNeighborsClassifier ----- 0.703 (70 %)
GaussianNB ----- 0.273 (27 %)
DecisionTreeClassifier ----- 0.667 (67 %)
RandomForestClassifier ----- 0.81 (81 %)

***** SELL *****

SVC ----- 0.578 (58 %)
KNeighborsClassifier ----- 0.535 (53 %)
GaussianNB ----- 0.209 (21 %)
DecisionTreeClassifier ----- 0.639 (64 %)
RandomForestClassifier ----- 0.818 (82 %)

***** ALL *****

SVC ----- 0.637 (64 %)
KNeighborsClassifier ----- 0.642 (64 %)
GaussianNB ----- 0.212 (21 %)
DecisionTreeClassifier ----- 0.674 (67 %)
RandomForestClassifier ----- 0.798 (80 %)

```

Obrázek 74: Evaluace modelů pro predikci typu budovy (autor)

Co se týče predikce stavu budovy, tak ani zde není bohužel úspěšnost vysoká, v nejlepším případě je to 64 %.

```

eval_classification_by_offer_type(models, dfs, dummies_attr, 'condition')

***** RENT *****

SVC ----- 0.57 (57 %)
KNeighborsClassifier ----- 0.477 (48 %)
GaussianNB ----- 0.523 (52 %)
DecisionTreeClassifier ----- 0.507 (51 %)
RandomForestClassifier ----- 0.641 (64 %)

***** SELL *****

SVC ----- 0.497 (50 %)
KNeighborsClassifier ----- 0.444 (44 %)
GaussianNB ----- 0.257 (26 %)
DecisionTreeClassifier ----- 0.574 (57 %)
RandomForestClassifier ----- 0.61 (61 %)

***** ALL *****

SVC ----- 0.537 (54 %)
KNeighborsClassifier ----- 0.428 (43 %)
GaussianNB ----- 0.376 (38 %)
DecisionTreeClassifier ----- 0.515 (52 %)
RandomForestClassifier ----- 0.629 (63 %)

```

Obrázek 75: Evaluace modelů pro predikci stavu budovy (autor)

3.7.3 Shluková analýza

Předchozí modely byly součástí prediktivní analýzy, takže teď se soustředíme na deskriptivní modelování. Předně se pokusíme provést clusterovou analýzu z různých pohledů. V rámci dané analýzy byly použity 2 algoritmy: K-means a K-modes.

Pro shlukovou analýzu byly připraveny další pomocné funkce. První z nich pomáhá určit optimální hodnotu N (počet clusterů) pomocí loketní metody.

```

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 = KMeans(n_clusters=k, init="random", n_init=5, verbose=1)
            kmodes.fit_predict(data)
            ssd.append(kmodes.inertia_)

    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()

```

Obrázek 76: Funkce pro hledání optimálního počtu clusterů (autor)

Další funkce vypisuje úspěšnost K-means modelu s využitím Silhouette skóre, které vypočítá vzdálenost mezi jednotlivými shluky. Hodnota dané metriky se nachází v rozmezí od -1 do 1, přičemž čím blíž k 1, tím lépe byly shluky vytvořeny.

```

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)}')

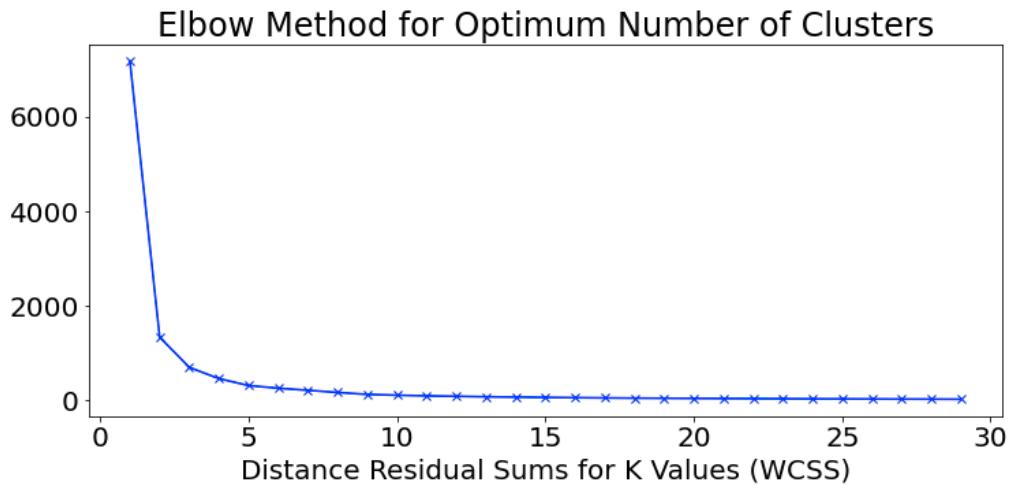
    return kmeans

```

Obrázek 77: Funkce pro evaluaci modelu K means (autor)

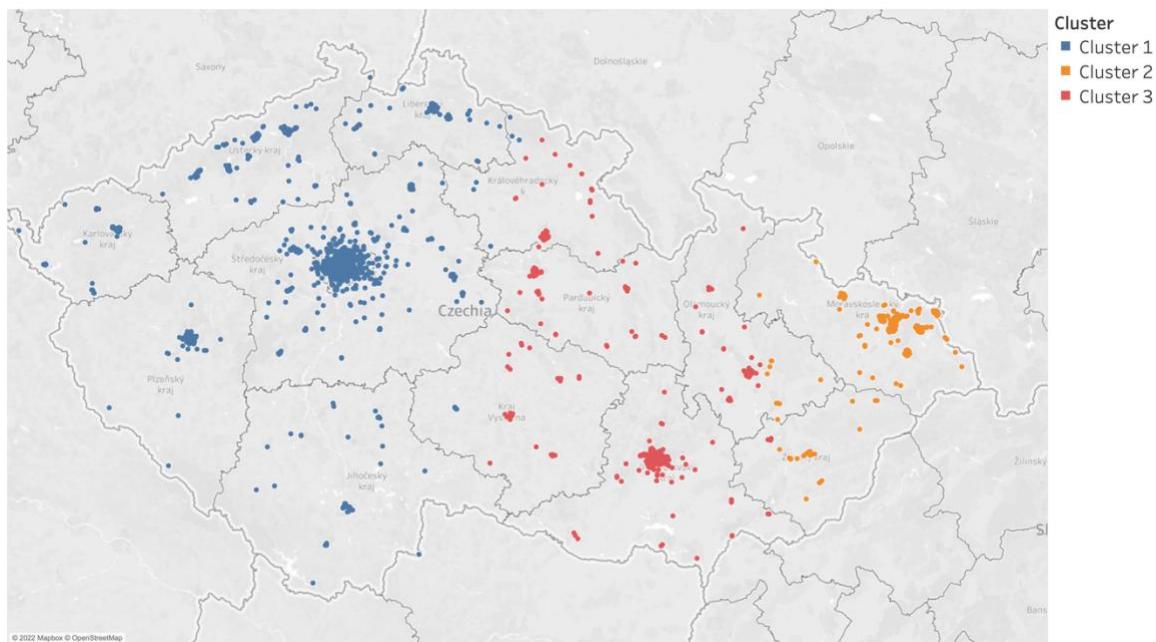
Poloha

Nejprve jsem zkusil použít algoritmus K-means pro 2 atributy, kterými jsou souřadnice polohy jednotlivých bytů. Loketní metodou bylo zjištěno, že optimální počet clusterů je 2.



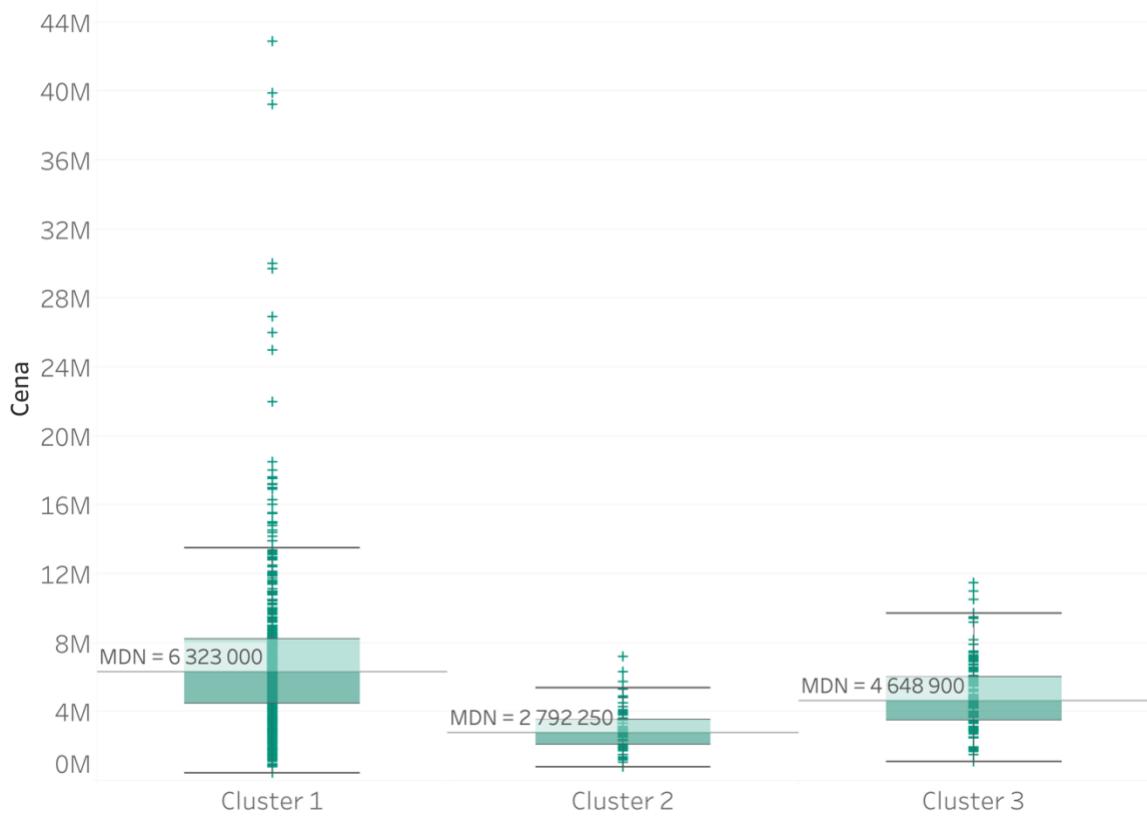
Obrázek 78: Loketní metoda pro určení optimálního počtu clusterů (autor)

Pro daný počet shluků se hodnota metriky úspěšnosti modelu rovná 0,782. Pokud zvýšíme počet shluků na 3, úspěšnost se sníží na 0,772, takže pro zajímavost jsem nastavil parametr K na 3 a vyexportoval data pro další vizualizační analýzu v Tableau. Vytvořené shluky jsem nejprve zobrazil na mapě. Je vidět, že do 1. shluku spadají s několika výjimkami kraje Karlovarský, Ústecký, Plzeňský, Středočeský, Liberecký a Jihočeský. Tento shluk je zároveň největší. Shluk č. 2 je pak představen pouze dvěma krajů – Moravskoslezským a Zlínským. Třetí shluk obsahuje nabídky z Královéhradeckého, Pardubického, Jihomoravského, Olomouckého a z Kraje Vysočina.



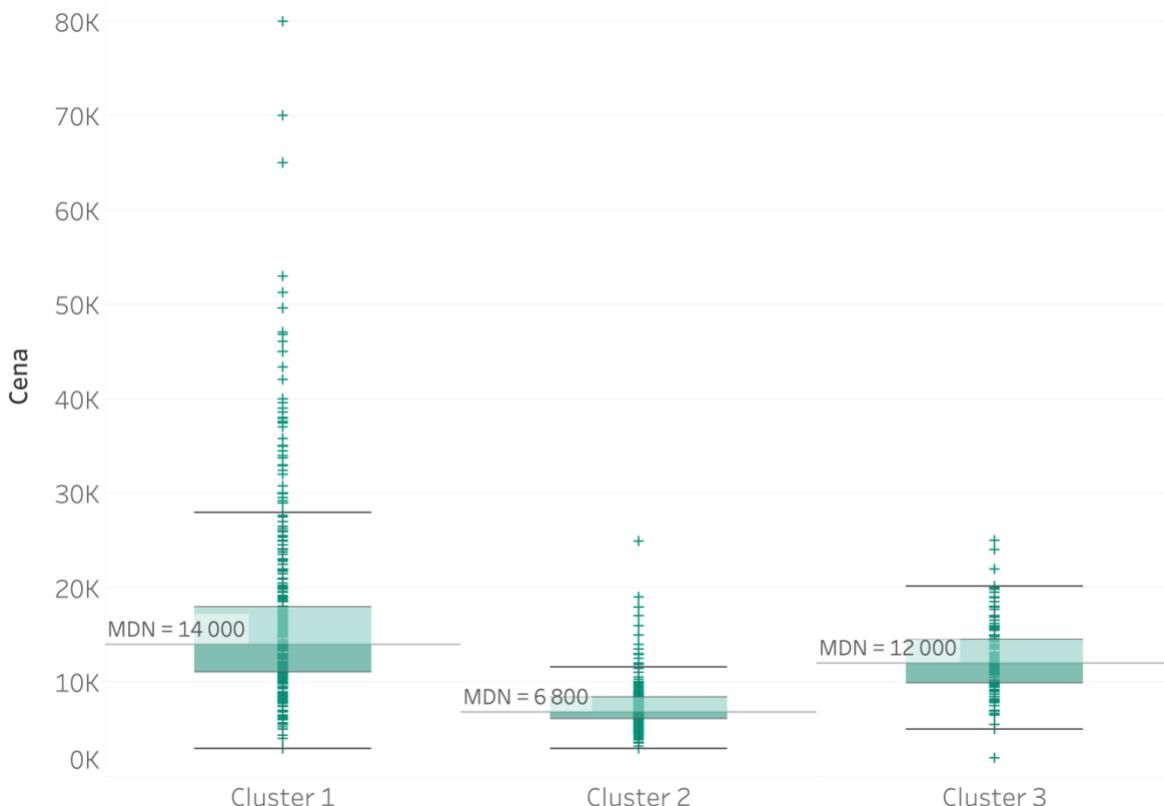
Obrázek 79: Clustery dle polohy (autor)

Podíváme-li se na rozložení cen bytů v každém clusteru, žádné zajímavosti tu asi nejsou – v prvním clusteru se nacházejí ty nejdražší nabídky, mediánní cena v clusteru je 6 300 000 Kč. Následuje cluster č. 3 s mediánní cenou přibližně 4 650 000 Kč a cluster č. 2, kde jsou nabídky s mediánní cenou 2 800 000 Kč.



Obrázek 80: Přehled cen bytů na prodej dle clusteru (autor)

U bytů k pronájmu se situace moc neliší, ale malou zajímavostí jsou procentuální rozdíly mezi jednotlivými clustery. Rozdíl mezi cenami bytů k pronájmu v clusteru 1 a v clusteru 3 je 36 %, zatímco u bytů na prodej je rozdíl mezi stejnými clustery 16 %. Jinými slovy pronajmout byt v kraji z clusteru č. 2 stojí nemnohem menší, než v kraji č. 1. Kupování bytů je tam však význačně levnější.



Obrázek 81: Přehled cen bytů k pronájmu dle clusteru (autor)

Cena a plocha (prodej)

Další analýza byla provedena pouze na nabídkách typu prodej. Jako atributy byly tentokrát zvoleny cena a plocha bytu. Opět pomocí loketní metody byl určen optimální parametr $k = 2$. Úspěšnost modelu byla 0,667. Tabulka dole znázorňuje přehled clusterů včetně mediánních hodnot zvolených atributů.

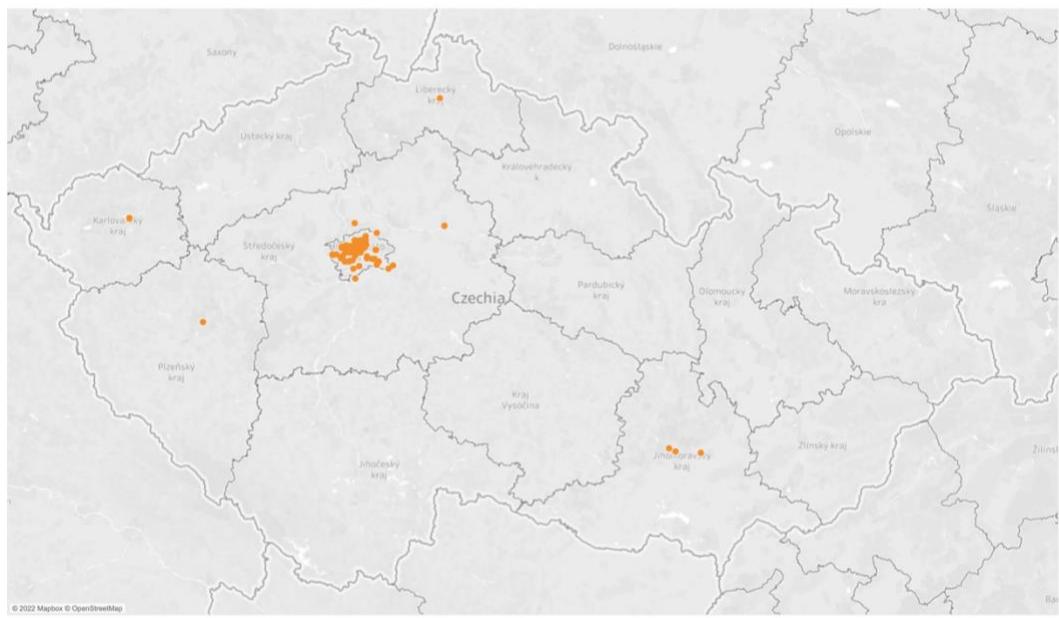
Tabulka 9: Přehled clusterů dle ceny a plochy (autor)

Cluster	Počet záznamů	Cena	Plocha
Cluster 1	818	5 299 450	57
Cluster 2	113	12 420 000	91

Model v podstatě rozdělil byty na 2 docela logické typy:

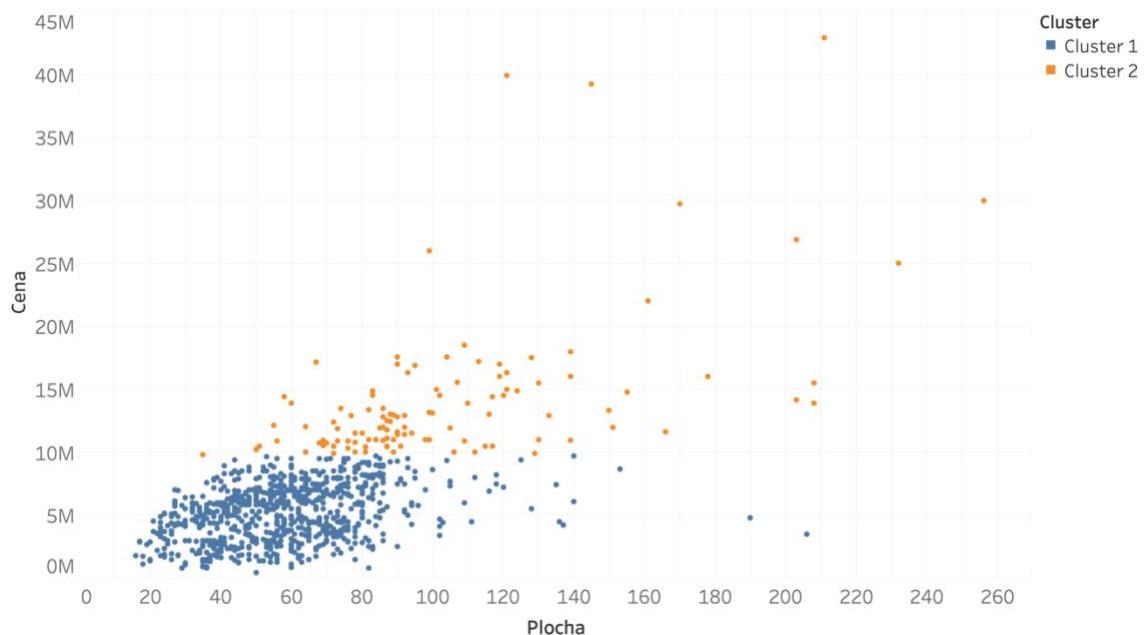
- větší a dražší,
- menší a levnější.

Nic zajímavého v podobném shlukování není, ale pokusíme se zobrazit tyto nabídky z druhého clusteru na mapě.



Obrázek 82: Cluster s dražšími a většími byty (autor)

Většina drahých nabídek se soustředila v Praze, což opět není žádné překvapení. Na bodovém grafu je zřetelně vidět, že lze nakreslit přímku na úrovni 10 000 000 Kč, která by byla hranicí mezi vytvořenými shluky.



Obrázek 83: Cena a plocha bytů dle clusteru (autor)

Cena, plocha, poplatky a kauce (pronájem)

Pro shlukování nabídek o pronájmu byly použity 4 atributy, a proto bylo potřeba je standardizovat a pak provést analýzu hlavních komponent (angl. zkrátka PCA). Ze tří atributů byly vytvořeny 2 ve stejném měřítku.

```

df_all_area_info = df[AREA_INFO_ATTRIBUTES]

pca = PCA(2)
scaler = StandardScaler()

df_all_area_info = pd.DataFrame(scaler.fit_transform(df_all_area_info), columns=df_all_area_info.columns)
df_all_area_info_pca = pd.DataFrame(pca.fit_transform(df_all_area_info))

find_optimum_n_clusters_elbow(df_all_area_info_pca)

```

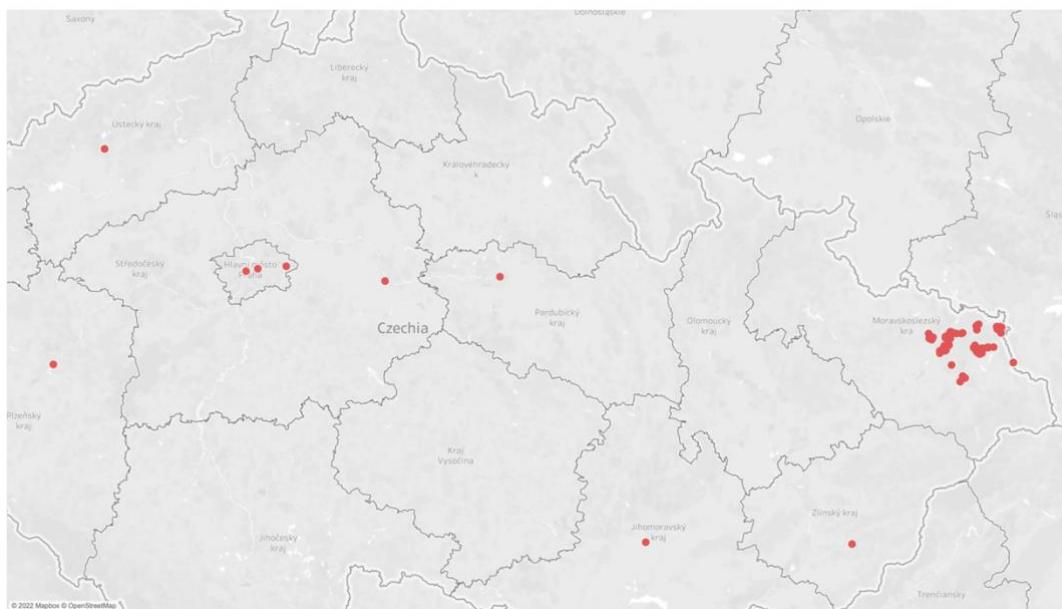
Obrázek 84: Příprava dat pro modelování (autor)

Výsledkem modelu (úspěšnost 0,564) byly 3 clustery.

Tabulka 10: Přehled clusterů dle ceny, plochy, poplatků a kauce (autor)

Cluster	Počet záznamů	Cena	Plocha	Poplatky	Kauce
Cluster 1	279	24 000	83	5 000	40 000
Cluster 2	1694	12 500	47	3 000	19 000
Cluster 3	247	6 550	51	10 917	13 100

Cluster č. 1 obsahuje nejdražší a největší byty s velkou hodnotou kauce. Největší cluster č. 2 má nejmenší plochu a poplatky ze všech clusterů. Poslední cluster je zajímavý tím, že se v něm soustředily nabídky s nejmenšími cenami, ale zároveň i s největšími poplatky. Tento shluk lze charakterizovat jako nabídky, u kterých jsou poplatky vyšší než cena bytu. Zobrazíme poslední shluk na mapě.



Obrázek 85: Cluster s byty s velkými poplatky (autor)

Okamžitě upoutá, že většina inzerátů z tohoto shluku se nachází v Moravskoslezském kraji. Stejné pozorování bylo nalezeno i během explorační analýzy.

Atributy o okolí (prodej + pronájem)

Další shluková analýza byla provedena s využitím všech atributů, týkajících se okolí. I když je model docela úspěšný (hodnota Silhouette skóre 0,836), výsledné shluky nejsou překvapivé. V jednom shluku jsou byty, které se nacházejí nadprůměrně daleko od různých míst. V druhém shluku jsou tedy byty, vzdálenost do míst od kterých není velká.

Tabulka 11: Přehled clusterů dle atributů o okolí (autor)

	Cluster 1	Cluster 2
Počet záznamů	3002	149
Zastávka MHD	219	395
Pošta	647	1700
Obchod	268	1430
Banka	689	6500
Restaurace	289	1397
Lékárna	466	4700
Škola	444	3300
Mateřská škola	580	4300
Sportoviště	734	4200
Hřiště	340	2300

Kategoriální atributy

Dosud všechny modely předpokládaly využití numerických atributů, takže jako poslední model jsem se rozhodl vytvořit model, který by používal kategoriální proměnné. Algoritmus K-means neumí pracovat s kategoriálními proměnnými, takže tentokrát jsem použil K-modes s počtem iterací 5, ze kterých byl vybrán nejlepší výsledek.

Před vytvořením modelu jsem diskretizoval vybrané numerické atributy (cena, plocha, poplatky a kauce).

```

df_cat = df_rent.copy()

df_cat['price_bin'] = pd.cut(df_cat['price'], [0, 5000, 10_000, 15_000, 20_000, 25_000, 30_000,
                                              35_000, 40_000, 45_000, 50_000, 150_000],
                               labels=['0K-5K', '5K-10K', '10K-15K', '15K-20K', '20K-25K', '25K-30K',
                               '30K-35K', '35K-40K', '40K-45K', '45K-50K', '>50K'],
                               right=True,
                               ordered=True)

df_cat['surface_bin'] = pd.cut(df_cat['surface'], [0, 20, 30, 40, 50, 60,
                                                    70, 80, 90, 100, 110, 350],
                                 labels=['0-20', '20-30', '30-40', '40-50', '50-60', '60-70',
                                 '70-80', '80-90', '90-100', '100-110', '>110'],
                                 right=True,
                                 ordered=True)

df_cat['utilities_bin'] = pd.cut(df_cat['utilities'], [0, 1000, 2000, 3000, 4000, 5000, 6000,
                                                       7000, 8000, 9000, 10000, 20000],
                                  labels=['0-1K', '1K-2K', '2K-3K', '3K-4K', '4K-5K', '5K-6K',
                                  '6K-7K', '7K-8K', '8K-9K', '9K-10K', '>10K'],
                                  right=True,
                                  ordered=True)

df_cat['deposit_bin'] = pd.cut(df_cat['deposit'], [0, 5000, 10_000, 15_000, 20_000, 25_000, 30_000,
                                                   35_000, 40_000, 200_000],
                                labels=['0K-5K', '5K-10K', '10K-15K', '15K-20K', '20K-25K', '25K-30K',
                                '30K-35K', '35K-40K', '>40K'],
                                right=True,
                                ordered=True)

df_cat['floor'] = df_cat['floor'].astype('str')

df_cat.drop(['price', 'surface', 'utilities', 'deposit', 'lat', 'lng'] + AREA_INFO_ATTRIBUTES,
            axis=1, inplace=True)

for attribute in df_cat.columns:
    df_cat[attribute] = df_cat[attribute].astype('category')

```

Obrázek 86: Příprava dat pro použití algoritmu K-modes (autor)

Výsledné clustery jsou představeny v následující tabulce.

Tabulka 12: Přehled clusterů dle kategoriálních atributů (autor)

	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
Počet záznamů	1085	361	246	238	290
Dispozice	2-kk	1-kk	2-kk	2-1	2-1
Vybavenost	Částečně	Vybavený	Částečně	Částečně	None
Podlaží	2	4	1	3	1
Balkón	Ne	Ne	Ano	Ano	Ne
Terasa	Ne	Ne	Ne	Ne	Ne
Sklep	Ne	Ano	Ne	Ano	Ano
Lodžie	Ne	Ne	Ne	Ne	Ne
Parkoviště	Ne	Ne	Ano	Ne	Ano
Výtah	Ne	Ano	Ano	Ano	Ne
Garáž	Ne	Ne	Ne	Ne	Ne
Stav budovy	Velmi dobrý	Velmi dobrý	Novostavba	Velmi dobrý	Dobrý
Typ vlastnictví	Osobní	Osobní	Osobní	Osobní	Osobní
Typ budovy	Cihla	Panel	Cihla	Cihla	Cihla
PENB	G	G	B	G	E
Cena	10K–15K	5K–10K	15K–20K	15K–20K	5K–10K
Plocha	40–50	30–40	50–60	60–70	50–60
Poplatky	2K–3K	1K–2K	3K–4K	3K–4K	>10K
Kauce	15K–20K	10K–15K	20K–25K	25K–30K	10K–15K

3.7.4 Detekce anomálií

Poslední úlohou je detekce anomálií. Pro danou úlohu byl použit algoritmus Isolation Forest. Jako atributy byly zvoleny cena, plocha, poplatky a všechny atributy o okolí. Pomocí GridSearchCV byly vybrány optimální parametry pro model.

```
model = IsolationForest(random_state=47)

param_grid = {'n_estimators': [50, 200],
              'max_samples': [10],
              'contamination': ['auto', 0.0001, 0.0002, 0.01, 0.02],
              'max_features': [2, 15],
              'bootstrap': [True],
              'n_jobs': [-1]}

grid_search = GridSearchCV(model, param_grid, scoring="neg_mean_squared_error", refit=True, cv=10,
                           return_train_score=True)

best_model = grid_search.fit(df_c)
print('Optimum parameters', best_model.best_params_)

Optimum parameters {'bootstrap': True, 'contamination': 'auto', 'max_features': 2, 'max_samples': 10, 'n_estimators': 50, 'n_jobs': -1}
```

Obrázek 87: Určení optimálních parametrů modelu Isolation Forest (autor)

Získané optimální parametry byly pak dosazeny do modelu.

```
model = IsolationForest(n_estimators=50, max_samples=10,
                        contamination='auto', max_features=5,
                        n_jobs=-1, random_state=17)

model.fit(df_c)

IsolationForest(max_features=5, max_samples=10, n_estimators=50, n_jobs=-1,
                 random_state=17)
```

Obrázek 88: Použití modelu Isolation Forest (autor)

Výsledkem modelu z předchozího kroku je 760 řádků. Jsou tam byty, které se nacházejí blízko (< 100 m) některých míst, nebo naopak daleko od nich. Kromě toho se ve výsledku vyskytují velké byty (120 m²) za nízkou cenu (16 000 Kč).

	price	surface	utilities	public_transport_stop	post_office	store	bank	restaurant	pharmacy	school	kindergarten	sport_field	playground	...
3	37500	60	12500.000000		96	290	15	494	73	336	519	216	542	280
4	9850	78	16416.666667		262	551	567	967	555	723	170	329	734	120
10	16000	55	7000.000000		480	635	523	2400	361	189	770	1076	795	676
11	11439	40	4032.000000		389	995	739	5100	926	941	1126	315	774	222
13	13000	58	4500.000000		553	1010	1222	3900	785	1000	970	1368	1900	391
...
2188	8700	34	1800.000000		852	1223	138	618	1020	670	498	2200	1060	502
2190	7000	55	5000.000000		646	1003	657	1096	526	510	405	646	517	759
2191	16000	120	3500.000000		602	2900	2800	2900	2900	2700	819	565	3000	288
2208	8900	38	2900.000000		149	1394	438	1407	523	858	1163	1052	488	483
2215	7900	27	1400.000000		273	1188	239	779	680	825	278	446	1277	483

Obrázek 89: Nalezené anomálie 1 (autor)

Pokud změníme parametr „contamination“ (podíl outlierů v datasetu) na 0,005, tj. tak, aby výsledné outliersy představovaly 0,5 % z datasetu a parametr „max_features“ zvýšíme o 5, tak získáme 11 inzerátů, které jsou na obrázku dole.

	price	surface	utilities	public_transport_stop	post_office	store	bank	restaurant	pharmacy	school	kindergarten	sport_field	playground	ano
175	9000	60	2500.000000	5000	2200	7200	8400	6900	8400	4400	7900	7300	4700	
184	12000	86	4000.000000	1488	1500	2700	9500	2900	1700	4000	2800	10200	1900	
341	6000	30	2800.000000	653	2300	2300	8900	3200	3300	3500	3500	3900	2400	
443	5500	20	9166.666667	578	1900	1900	6600	1102	2000	1700	2600	3200	3100	
968	18000	75	3500.000000	1325	1600	1322	6000	1600	4700	4600	2000	5000	1283	
1288	12000	42	2000.000000	1161	4800	2600	5800	8100	6000	5700	6600	6200	7000	
1356	7000	33	3500.000000	4400	3700	4500	10500	4800	5300	4800	8900	4700	4200	
1406	21000	79	3500.000000	90	3200	3100	9100	3100	6200	3000	3100	3700	1800	
1527	8000	30	3000.000000	3700	3900	7700	7700	7100	7400	8000	8600	3200	228	
1610	11000	80	4000.000000	2200	3500	710	3600	2800	4300	2500	2600	2900	4000	
2063	9500	36	2500.000000	874	1308	36100	6200	37200	6200	5400	5000	6400	1009	

Obrázek 90: Nalezené anomálie 2 (autor)

Lze předpokládat, že model označuje za outliery byty, které se nacházejí více než 2, 3, někdy i 7 km daleko od různých míst.

3.8 Diskuze výsledků

Předposledním krokem metodiky CRISP-DM je evaluace výsledků. Tato část shrnuje všechno, co bylo zjištěno během analýzy.

Pomocí web scrapingu se podařilo vyextrahovat více než 3500 inzerátů, po zpracování dat jich však zůstalo 3131: z toho 2220 nabídek pronájmu a 931 nabídek prodeje bytů.

Z explorační analýzy bylo zjištěno, že v Moravskoslezském kraji jsou 2 města (Havířov a Poruba), kde poplatky za byt jsou obzvlášť vysoké. Je možné, že jde o staré budovy nebo o budovy s nízkou energetickou třídou. Ovšem není to tak: všechny byty v těchto městech jsou buď v budovách v dobrém, nebo ve velmi dobrém stavu. Energetická třída budov se nachází v rozmezí C–F.

Navíc bylo také zjištěno, že energetická náročnost budovy PENB nesouvisí s poplatky, tj. budovy s energetickou třídou F nebo G (nejhorší) mají nižší, nebo stejné poplatky jako budovy s třídou A–C, což je dost zajímavé, ale na druhou stranu by bylo vhodné to prokonzultovat s doménovým expertem ohledně relevantnosti daného pozorování. Kromě výše uvedených zajímavostí bylo také zjištěno, že 40 % bytů z datasetu se nachází v budovách s nejhorší energetickou třídou (G).

Byly rovněž určeny typické byty k pronájmu a na prodej v ČR a Praze.

Tabulka 13: Přehled typických bytů v ČR a Praze (autor)

	Cena	Plocha	Poplatky	Cena za m²
Pronájem				
ČR	13 800	52	4100	265
Praha	17 100 (+ 24 %)	53	3500 (- 17 %)	322 (+ 22 %)
Prodej				
ČR	6 300 000	63	-	100 200
Praha	8 260 000 (+ 31 %)	63	-	131 200 (+ 31 %)

Další nalezenou zajímavostí byla cena za byt podle dispozice. Zjistilo se, že v některých případech jsou větší byty levnější. Tak například pronajmout si byt 1–1 stojí méně než byt 1–kk. To samé platí i pro byty 2–1 a 2–kk, 3–1 a 3–kk. U bytů na prodej je to stejně. Například byty 3–1 stojí méně než byty 3–kk. To by mohlo být způsobeno různou likviditou bytů, ale tak to nebude v souladu s dalším nálezem o tom, že nejpopulárnější byty v datasetu jsou 2–kk (byty 2–kk by měly stát více než byty 1–kk, což z datasetu nevychází).

Další nález se týká vybavenosti bytů. Z nějakého důvodu se v datasetu vyskytují byty v budovách ve stavu „ve výstavbě“ a s plnou či částečnou vybaveností, ale je možné, že jde jen o chybu při zadávání inzerátu.

Poté byla provedena rychlá korelační analýza, ale nebyly zjištěny žádné zajímavé ani překvapivé výsledky. Z nejvíce korelujících proměnných lze označit vzdálenost do lékárny a do banky u bytů na prodej a vzdálenost do restaurace a do obchodu u bytů k pronájmu.

V části modelování bylo vytvořeno několik modelů. Lasso regresí se podařilo predikovat cenu pronájmu s úspěšností 76 % a byl to nejlepší regresní model. Predikce poplatků a ceny prodeje byly na úrovni úspěšnosti 56–59 %.

Z klasifikační úlohy byl nejúspěšnější model (82 %) RandomForest pro predikci typu budovy, ale s ohledem na rozložení četnosti kategorií daného atributu bych tohle za úspěch nepovažoval.

Následovala deskriptivní analýza datasetu, během níž bylo zjištěno, že se mezi některými kraji České republiky liší procentuální rozdíl mezi cenami bytů k pronájmu a na prodej. Tak například někde je rozdíl v cenách bytů k pronájmu 36 %, ale u bytů na prodej je to pouze 16 %.

Závěr

Cílem práce bylo analyzovat realitní trh ČR s využitím kombinace web scrapingu a data miningu. Práce se dělí na teoretickou a praktickou část. Teoretická část obsahuje uvedení do oblasti data miningu, včetně popisu metodik a modelů. Pak obsahuje představení technologie web scrapingu.

Praktická část začíná představením nástrojů použitých pro analýzu, další kapitoly byly v souladu s kroky z metodiky CRISP-DM. Nejprve byla krátce představena doménová oblast. Důležitým krokem byl pokus o hledání API webové stránky. Nalezené API umožnilo zrychlit proces sběru dat. Pomocí web scrapingu se podařilo vyexportovat více než 3500 nabídek o prodeji / pronájmu bytů. Během získání dat jsem narazil na několik menších problémů, jako například popup okénko, které neumožňovalo vyextrahovat informaci o bytu. Nejprve jsem vyzkoušel vytvořený skript na malém počtu nabídek. Celkem proces sběru dat trval přes 15 hodin kvůli používání proxy, ale naštěstí stačilo jedno spuštění skriptu.

Následující kapitola měla za cíl představit získaná data. Kapitola Předzpracování dat byla časově nejnáročnější a zahrnovala vytvoření nových atributů, transformace dat, vyplnění chybějících hodnot atd. V kapitole o explorační analýze dat byly odhaleny první zajímavosti. Po kapitole s EDA byla kapitola s modelováním, kde byly vytvořeny modely pro různé typy data mining úloh – regresní, klasifikační, shluková analýza a detekce anomalií. Jako poslední byla kapitola s diskusí výsledků.

Za přínos práce lze považovat některé zjištěné zajímavosti v datech. Jako osobní přínos práce vidím seznámení s různými data mining metodami, získanou praxi vizualizace dat v Tableau a hlubší seznámení s doménovou oblastí realitních dat celkem.

Jako možnost rozšíření práce bych uvažoval o použití více zdrojů dat s regulárním obnovením dat, aby byly ve výsledku co nejvíce aktuální inzeráty. Také z mého pohledu nejsou výsledky samotného modelování význačné a v této podobě se asi nedají moc použít v praxi. Jako další možnost rozšíření práce bych viděl lepší předzpracování dat s hlubším pohledem na doménovou oblast, aby se pak daly vybrat a použít optimální data mining modely.

Použitá literatura

1. FAYYAD, Usama M., Gregory PIATETSKY-SHAPIRO a Padhraic SMYTH. *Advances in knowledge discovery and data mining*. Menlo Park (Calif.): AAAI press MIT press, 1996. ISBN 978-0-262-56097-9.
2. Cambridge International AS & A Level Information Technology. *Cambridge International Examinations* [online]. 2015. Dostupné z: <https://www.cambridgeinternational.org/images/285017-data-information-and-knowledge.pdf>
3. *methodology noun - Definition, pictures, pronunciation and usage notes | Oxford Advanced American Dictionary at OxfordLearnersDictionaries.com* [online]. [vid. 2022-04-25]. Dostupné z: https://www.oxfordlearnersdictionaries.com/definition/american_english/methodology
4. MARTINEZ-PLUMED, Fernando, Lidia CONTRERAS-OCHANDO, Cesar FERRI, Jose HERNANDEZ-ORALLO, Meelis KULL, Nicolas LACHICHE, Maria Jose RAMIREZ-QUINTANA a Peter FLACH. CRISP-DM Twenty Years Later: From Data Mining Processes to Data Science Trajectories. *IEEE Transactions on Knowledge and Data Engineering* [online]. 2021, **33**(8), 3048–3061. ISSN 1041-4347, 1558-2191, 2326-3865. Dostupné z: doi:10.1109/TKDE.2019.2962680
5. *SAS Help Center: Introduction to SEMMA* [online]. [vid. 2022-04-25]. Dostupné z: <https://documentation.sas.com/doc/en/emref/14.3/n061bzurmej4j3n1jnj8bbjjm1a2.htm>
6. *CRISP-DM Help Overview* [online]. 17. srpen 2021 [vid. 2022-04-25]. Dostupné z: <https://prod.ibmdocs-production-dal-6099123ce774e592a519d7c33db8265e-0000.us-south.containers.appdomain.cloud/docs/en/spss-modeler/SaaS?topic=dm-crisp-help-overview>
7. RAUCH, Jan a Milan SIMUNEK. *Dobývání znalostí z databází, LISp-Miner a GUHA*. Praha: Oeconomica, 2014. ISBN 978-80-245-2033-9.
8. OLSON, David L. *Data Mining Models, Second Edition*. [online]. New York: Business Expert Press, 2018 [vid. 2022-05-09]. ISBN 978-1-948580-50-2. Dostupné z: <https://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&db=nlabk&AN=1797956>
9. LAROSE, Daniel T. a Chantal D. LAROSE. *Discovering knowledge in data: an introduction to data mining*. Second edition. Hoboken: Wiley, 2014. ISBN 978-1-118-87357-1.
10. ROGALEWICZ, Michał a Robert SIKA. Methodologies of Knowledge Discovery from

Data and Data Mining Methods in Mechanical Engineering. *Management and Production Engineering Review* [online]. 2016, 7(4), 97–108. ISSN 2082-1344. Dostupné z: doi:10.1515/mper-2016-0040

11. *Different Types of Regression Models - Analytics Vidhya* [online]. [vid. 2022-04-25]. Dostupné z: <https://www.analyticsvidhya.com/blog/2022/01/different-types-of-regression-models/>
12. FAYYAD, Usama M., Gregory PIATETSKY-SHAPIRO a Padhraic SMYTH. Knowledge discovery and data mining: towards a unifying framework. In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96)*. B.m.: AAAI Press, 1996, s. 82–88.
13. DANGETI, Pratap. *Statistics for machine learning: techniques for exploring supervised, unsupervised, and reinforcement learning models with Python and R*. Birmingham, UK: Packt Publishing, 2017. ISBN 978-1-78829-575-8.
14. AGRAWAL, Shikha a Jitendra AGRAWAL. Survey on Anomaly Detection using Data Mining Techniques. *Procedia Computer Science* [online]. 2015, 60, 708–713. ISSN 18770509. Dostupné z: doi:10.1016/j.procs.2015.08.220
15. KOHAVI, Ron. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. *International Joint Conference on Artificial Intelligence*. 2001.
16. SAYAD, Saed. *Model Evaluation - Classification* [online]. [vid. 2022-05-09]. Dostupné z: https://www.saedsayad.com/model_evaluation_c.htm
17. HAN, Jiawei, Micheline KAMBER a Jian PEI. *Data mining: concepts and techniques*. 3rd ed. Burlington, MA: Elsevier, 2012. ISBN 978-0-12-381479-1.
18. GU, Qiong, Li ZHU a Zhihua CAI. Evaluation Measures of the Classification Performance of Imbalanced Data Sets. In: Zhihua CAI, Zhenhua LI, Zhuo KANG a Yong LIU, ed. *Computational Intelligence and Intelligent Systems* [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009 [vid. 2022-05-09], Communications in Computer and Information Science, s. 461–471. ISBN 978-3-642-04961-3. Dostupné z: doi:10.1007/978-3-642-04962-0_53
19. BHALLA, Deepanshu. Understand Gain and Lift Charts. *ListenData* [online]. [vid. 2022-04-25]. Dostupné z: <https://www.listendata.com/2014/08/excel-template-gain-and-lift-charts.html>
20. BRADLEY, Andrew P. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition* [online]. 1997, 30(7), 1145–1159. ISSN 00313203. Dostupné z: doi:10.1016/S0031-3203(96)00142-2
21. BUI, Huy. ROC Curve Transforms the Way We Look at a Classification Problem. *Medium* [online]. 3. březen 2020 [vid. 2022-04-25]. Dostupné z: <https://towardsdatascience.com/a-simple-explanation-of-the-roc-curve-and-auc-64db32d75541>

22. SAYAD, Saed. *Model Evaluation - Regression* [online]. [vid. 2022-05-09]. Dostupné z: https://www.saedsayad.com/model_evaluation_r.htm
23. PARVEZ, Momin Saniya, Khan Shaista Agah TASNEEM, Shivankar Sneha RAJENDRA a Kalpana R. BODKE. Analysis Of Different Web Data Extraction Techniques. In: *2018 International Conference on Smart City and Emerging Technology (ICSCET): 2018 International Conference on Smart City and Emerging Technology (ICSCET)* [online]. Mumbai: IEEE, 2018, s. 1–7 [vid. 2022-05-09]. ISBN 978-1-5386-1185-2. Dostupné z: doi:10.1109/ICSCET.2018.8537333
24. *XML Essentials - W3C* [online]. 1996 [vid. 2022-05-09]. Dostupné z: <https://www.w3.org/standards/xml/core>
25. HAROLD, Elliotte Rusty a W. Scott MEANS. *XML in a nutshell*. 3rd ed. Beijing ; Sebastopol, CA: O'Reilly, 2004. In a nutshell. ISBN 978-0-596-00764-5.
26. *XML Path Language (XPath)* [online]. 1993 [vid. 2022-05-09]. Dostupné z: <https://www.w3.org/TR/1999/REC-xpath-19991116/>
27. IBM CLOUD EDUCATION. *What is an Application Programming Interface (API)* [online]. 19. srpen 2020 [vid. 2022-05-09]. Dostupné z: <https://www.ibm.com/cloud/learn/api>
28. JUVILER, Jamie. *4 Types of APIs All Marketers Should Know* [online]. 26. srpen 2021 [vid. 2022-05-09]. Dostupné z: <https://blog.hubspot.com/website/types-of-apis>
29. JOUMAN HAJJAR, Alamira. *The Ultimate Guide to Web Scraping Challenges & Best Practices* [online]. 16. srpen 2021 [vid. 2022-05-09]. Dostupné z: <https://research.aimultiple.com/web-scraping-challenges/>
30. Co je GDPR? *GDPR.cz* [online]. [vid. 2022-05-09]. Dostupné z: <https://www.gdpr.cz/gdpr/>
31. NARÍZENÍ EVROPSKÉHO PARLAMENTU A RADY (EU) o ochraně fyzických osob v souvislosti se zpracováním osobních údajů a o volném pohybu těchto údajů a o zrušení směrnice 95/46/ES (obecné nařízení o ochraně osobních údajů). 27. duben 2016
32. *XMLHttpRequest - Web APIs | MDN* [online]. [vid. 2022-04-27]. Dostupné z: <https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest>
33. What is an API Endpoint? | API Endpoint Definition | RapidAPI. *The Last Call - RapidAPI Blog* [online]. [vid. 2022-04-25]. Dostupné z: <https://rapidapi.com/blog/api-glossary/endpoint/>
34. *Průkaz energetické náročnosti budovy - vše co o PENB potřebujete vědět - V-systém elektro, s.r.o.* [online]. [vid. 2022-04-27]. Dostupné z: <https://www.v-system.cz/blog/prukaz-energeticke-narocnosti-budovy-vse-co-o-penb-potrebujete-vedet/>

Přílohy

Příloha A: Zdrojový kód web scraperu

Příloha obsahuje okomentovaný zdrojový kód web scraperu v jazyce Python. Skládá se z 2 složek (scraper a data).

1. scraper/main.py – spuštění skriptu
2. scraper/web_scraper.py – základní logika web scraperu
3. scraper/web_scraper.log – záznam logů o běhu programu
4. scraper/geckodriver.log – záznam logů o web driveru
5. scraper/requirements.txt – přehled potřebných ke spuštění skriptu knihoven
6. data/input.xlsx – ukázka vstupního souboru s inzeráty
7. data/proxies.txt – ukázka seznamu proxy
8. data/used_ids.xlsx – ukázka souboru se seznamem již použitých inzerátů

Příloha byla zároveň nahrána na GitHub: <https://github.com/tsaioo/bc-thesis>

Příloha B: Zdrojový kód Jupyter Notebooks

Příloha obsahuje okomentované soubory Jupyter Notebooks.

1. API.ipynb – získání dat z API
2. merging.ipynb – sloučení souborů s inzeráty
3. preprocessing.ipynb – předzpracování dat
4. modelling.ipynb – vytvoření modelů

Příloha byla zároveň nahrána na GitHub: <https://github.com/tsaioo/bc-thesis>

Příloha C: Data

Příloha obsahuje příklady výsledných dat.