FACULTY OF INFORMATION TECHNOLOGY CTU IN PRAGUE

Assignment of bachelor's thesis

Title:	Application of Machine Learning in Real Estate
Student:	Jakub Nádraský
Supervisor:	Ing. Mgr. Ladislava Smítková Janků, Ph.D.
Study program:	Informatics
Branch / specialization:	Knowledge Engineering
Department:	Department of Applied Mathematics
Validity:	until the end of summer semester 2022/2023

Instructions

1. Map the state of the art in the field of application of machine learning in real estate sales and rentals.
2. Use publicly available data and create a data set containing information about the property, photo documentation and price (sale or lease).
3. Select the appropriate machine learning method to automatically determine the sale / lease price. Implement the method using existing libraries or tools.
4. Perform experiments and evaluate them.

Electronically approved by Ing. Karel Klouda, Ph.D. on 18 January 2022 in Prague.

Bachelor's thesis

APPLICATION OF MACHINE LEARNING IN REAL ESTATE

Jakub Nádraský

Faculty of Information Technology
Department of Applied Mathematics
Supervisor: Ing. Mgr. Ladislava Smítková Janků, Ph.D.
May 12, 2022

Contents

List of Figures

List of Tables

I would like to thank my supervisor, Ing. Mgr. Ladislava Smítková Janků, Ph.D., for her guidance, time, and valuable advice. I would also like to thank my family, who has supported me the last few years.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as a school work under the provisions of Article 60 (1) of the Act.

In Prague on May 12, 2022 .....

Abstract

The subject of this thesis is the application of machine learning for automatic real estate object rent price prediction, based on data from Czech real estate listing websites. A dataset containing technical parameters of real estate objects is compiled, visualized, and used to train several machine learning models, which are then compared.

Next, the thesis focuses on the usage of image data to improve real estate price prediction. The possibility of descriptive feature extraction from image data is explored. Methodologies for manual feature extraction are introduced, and several features, such as the level of natural light in the object, are experimentally extracted from image data. A dataset containing descriptive image features is compiled, visualized, and used with machine learning models. The inclusion of image data features was measured to decrease the error rate of the price predicion of the used models by approximately 10 to 20%.

Keywords real estate price prediction, real estate data visualization, image data features, machine learning, Python

Abstrakt

Tématem této práce je použití metod strojového učení pro automatické ohodnocení výše nájmu realitního objektu. Ohodnocení je založené na datech z českých serverů pro inzerci realit. Datová sada obsahující technické parametry jednotlivých realitních objektů je sestavena, vizualizována a použita pro trénování několika modelů strojového učení, které byly poté porovnány.

Dále se práce zabývá použitím obrazových dat pro vylepšení predikce výše nájmu realitních objektů. Byla prozkoumána možnost extrakce popisných příznaků z obrazových dat. Byly zavedeny metodiky pro manuální extrakci a několik příznaků, například úroveň přirozeného osvětlení v realitním objektu, bylo experimentálně extrahováno. Datová sada obsahující popisné příznaky extrahované z obrazových dat byla sestavena, vizualizována a použita pro trénování modelů strojového učení. Bylo naměřeno, že prezence příznaků extrahovaných z obrazových dat snížila chybu predikce výše nájmu použitých modelů strojového učení o zhruba 10 až 20 %.

Klíčová slova predikce cen realit, vizualizace realitních dat, příznaky z obrazových dat, strojové učení, Python

Abbreviations

AI	Artificial Intelligence
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
KNN	K Nearest Neighbors
MAE	Mean Average Error
ML	Machine Learning
MSE	Mean Squared Error
RE	Real Estate
RMSE	Root Mean Squared Error
SVM	Support Vector Machine
URL	Uniform Resource Locator

Chapter 1
Introduction

In this chapter, the reader is briefly introduced to the contents of this thesis. We talk about the general area of this thesis and why it is worthwhile to conduct research in this area. We mention what will and what will not be covered in this thesis. Lastly, we introduce the thesis structure and goals.

Everyone needs to live somewhere. Unfortunately, it is often quite expensive. That is why a correct and fair price evaluation of real estate objects is necessary. Today, most of the evaluation is done by people, and thus it is subject to human error. In this thesis, we focus on the application of machine learning in the area of automatic real estate price evaluation.

Machine learning models can be used to equitably predict the price of a real estate object based on existing listings on the real estate market. This is useful for anyone involved, as it minimizes the human error and subjectivity of the evaluation.

For example, real estate agents can use machine learning models to help improve the time efficiency of the financial evaluation of a real estate object. Either by using the machine learning models as tools to check the accuracy of their evaluation, a sort of a second opinion, or using the results of the models as a quick, baseline evaluation, that they can later elaborate on. Additionally, the agents can examine the evaluations of the models, to possibly gain new insights into the evaluation process, as not every agent is an expert in every region and real estate object type.

Machine learning models for price prediction are also useful for the average citizen entering the real estate market. Whether they want to buy, sell, or rent, a baseline, non-biased, readily available[1] evaluation will always be convenient. Currently, some Czech real estate listing websites even offer such baseline evaluation[2].

1.1 Thesis structure

Finally, all there is left to introduce is the thesis structure. In chapter one – *Introduction*, we present the motivation for writing this thesis, the thesis structure, and the thesis goals.

In chapter 2 – *Related works*, we map out the current usage of machine learning methods for price prediction, considering master theses, and journal articles.

In chapter 3 – *Data analysis*, we analyze the data available on Czech real estate listing websites and select a webserver as a source for a dataset. Next, we implement a web scraping script to download the data of individual real estate listings, including image data from the

[1] Without the need to talk to anyone.
[2] Albeit somewhat simple

chosen webserver. From this data, we compile a dataset. The dataset is explored, described, visualized, and processed to remove any errors or problems. Finally, we describe the process of manually extracting features from image data and introduce a feature extracting methodology for each extracted feature.

In chapter 4 – *Price prediction*, we select machine learning models to use for price prediction. We describe the chosen machine learning models and the error metrics that we use to measure the error of the models. We describe the process of training the machine learning models, as well as the used tools and software libraries.

In chapter 5 – *Image data experimentation*, we design, perform, and evaluate experiments using our trained machine learning models and the compiled dataset. The main experiment is to try and use extracted features from image data to try and improve the price prediction accuracy.

In the last chapter, chapter 6 – *Conclusion*, we conclude and reflect on the thesis, as well as summarize the results and introduce ideas for future work.

1.2 Thesis goals

The main goals of this thesis align with the four points of the thesis assignment. The first goal is to map out the current use of machine learning methods for real estate price prediction.

The second goal is to use publicly available data to compile, explore and visualize a dataset of features describing real estate objects, including image and point of interest data. If necessary, create a program to download data from the internet.

The third goal is to select machine learning models for real estate price evaluation and implement them using existing software tools and libraries. We will then train these models for real estate object price prediction using the compiled dataset and measure their performance.

The fourth goal of the thesis is to experiment with the compiled dataset. We will experiment with the use of features extracted from image data to try and improve the price prediction accuracy. Note that implementing a program for feature extraction from image data is not the goal of this thesis.

Chapter 2
Related works

In this chapter, we first mention the general applications of artificial intelligence methods in real estate. Then we map the contemporary usage of machine learning methods for real estate price prediction, considering master theses and journal articles.

2.1 Applications of artificial intelligence in real estate

In their master thesis *Artificial Intelligence and Machine Learning: Current Applications in Real Estate* [1], the author, Jennifer E. Conway, focuses on the area of AI (Artificial Intelligence) in the real estate sector, reviewing the current state of the art, as well as the possible future applications. The author recognizes several application areas for AI: data gathering distribution, analysis, automated valuation models, risk assessment, business processes, natural language processing, computer vision and image processing, 3D augmentation and space planning, geospatial analytics, and internet of things.

2.2 Real estate price prediction using machine learning methods

In this section, we mention various related works, regarding the applacation of machine learning methods for automatic real estate object price prediction.

2.2.1 Real estate price prediction in Hanoi – case study

In their research paper *House Price Estimation in Hanoi using Artificial Neural Network and Support Vector Machine: in Considering Effects of Status and House Quality* [2], the authors, Bui Quang-Thanh, Do Nhu-Hiep, and Phe Hoang set out to compare different machine learning methods for real estate price prediction on data gathered in Hanoi, the capital of Vietnam.

The authors worked with a dataset of 1000 records obtained in a survey conducted in 2014. The original dataset contained 244 features for each record. Using the stepwise regression method, the number of features was reduced to 30. Unlike in many other papers, the authors had the opportunity to work directly with survey data, which allowed them to consider both tangible and intangible real estate parameters. They considered the technical parameters of the real estate object as tangible, for example, the total area or the presence of air conditioning. The intangible parameters were considered to be, for example, the type of street, the distance to the city center, or the Hanoi neighborhood in which the real estate object is located.

To measure the model's prediction errors, the authors used the MAE (Mean Absolute Error) and RMSE (Root Mean Squared Error) metrics. The selected models were trained and evaluated using 10-fold cross-validation. The best performing models were SVM (Support Vector Machine) and ensemble models.

2.2.2 Price prediction using the K-NN model

In their paper *Implementation and Study of K-Nearest Neighbour and Regression Algorithm for Real-time Housing Market Recommendation Application* [3], the author, Shujia Zhao has set out to implement a real-time mobile application for recommending real estate objects for investment. The goal of the paper was to create a web-crawling program to collect real-time real estate data. Another goal was to implement a mobile application that suggests real estate suitable for investment based on the collected data with emphasis on the price to rent ratio of a given real estate.

The author obtained the data from Zoopla – a real estate listing website based in the United Kingdom. The data in the individual listings was only very technical: location, living area, number of bathrooms, etc. Since the listings often contain either only the price value or only the rent value, the author used a K-NN (K-Nearest Neighbors) machine learning model with weighted features to fill in the missing values. That is, they predicted the sale price or the rent price of an individual real estate object based on the other listings. The author computed the weights of the features using a linear regression model.

Considering the K-NN model, the author mentions the need to eliminate certain features created using one-hot encoding due to encountering issues related to the curse of dimensionality.

The author suggests working with a more detailed dataset as possible future work. The author mentions the possible use of image data or full-text descriptions of the listings.

2.2.3 Real estate price prediction using image data

In their paper *Price Prediction and Deep Learning in Real Estate Valuation Models* [4], the author, Joseph Kintzel, explored the possibility of using deep learning for real estate price prediction using image data. The goal of the work was to explore methods for real estate price prediction and implement a model that would be competitive with existing price predictors. The author specifically mentions the Zestimate tool by Zillow – an American real estate online marketplace company. Another goal of the paper was to test the possibility of using image data to improve the prediction accuracy of machine learning models.

The author implemented and tested several machine learning models: linear regression, random forest, and neural network. As for the neural network model, the author had a problem balancing the prediction accuracy and the model's ability to generalize – predict previously unseen data. The author encountered the bias-variance trade-off problem. The author solved the problem by introducing dropout layers into the neural network model. Even though this improved the price prediction, the neural network model was outperformed by the other models.

Furthermore, unlike many other papers, the author focused on the use of image data to improve prediction accuracy. The author extracted features from the image data using existing feature extraction tools. Attempting to simply append these features to the descriptive data did not improve the prediction. So the author decided to implement a neural network with a special internal structure, tailored to the used data. This process yielded a small but measurable improvement in prediction. Despite the author's efforts, the neural network model was still outperformed by the other prediction models – specifically the ensemble models.

The author considers the use of image data to predict the price of real estate as untapped potential. Therefore, the author additionally implemented a binary classifier of a so-called "curb appeal" using already existing tools for feature extraction from image data. The classifier predicted whether the implemented price prediction model would over or under price a given real

estate based on image data in the form of a view from the street. The classifier yielded "better than random" results. In doing so, the author showcased another possible usage of image data in real estate price prediction.

2.2.4 Stacked model and comparison of methods for real estate price prediction

In their paper *Comparison of Data Mining Models to Predict House Prices* [5], the author, Stephen O'Farrell, sets out to compare several machine learning models, including a stacked model, for real estate price prediction. The goal of the paper was to test if stacking can measurably improve the prediction results.

The author used a dataset from the Kaggle.com website, randomly truncated to 3000 records, containing data about real estate sales in King County, USA, in 2014 and 2015. The dataset contained only basic technical features of a given real estate object.

The metrics used to evaluate the prediction were MAE, MSE, and RMSE. The selected methods were trained and measured by a 5 times repeated 10-fold cross-validation. The following models were tested: Linear Regression, Partial Least Squares, Gradient Boosting Machine, K-Nearest Neighbors, Support Vector Machine, and a Stacked model. All models performed comparably, except for K-Nearest Neighbors, which underperformed.

The prediction given by the stacked model was more accurate on the training set, but not on the test set. Therefore, the question of whether it is possible to achieve better prediction accuracy using stacked models could not be answered conclusively. The author points to the limited dataset and suggests further testing on a more complex dataset. Such dataset would take into account, for example, economic information about potential buyers.

2.3 Chapter summary

In this chapter, we mapped out the area of application of machine learning methods in automatic real estate object price prediction. We mainly focused on the specific used methods. Generally, ensemble models seemed to be suitable for real estate price prediction, while the K-NN model consistently underperformed.

We recognize a need for the usage of additional data in real estate price prediction, mainly image data. This serves as a motivation to experiment with image data in this thesis.

Chapter 3
Data analysis

In this chapter, we analyze the data available on Czech websites for real estate listing and select a website to use as a data source. Furthermore, we implement a web scraper to download data about listings, including image data. Finally, we will analyze and visualize the downloaded data and compile it into a dataset, that will be used to train machine learning models in later chapters of the thesis.

3.1 Data sources exploration

There are many detailed datasets of technical parameters compiled from real estate listings or completed sales already available on the internet. These datasets mostly lack other useful data, such as included images, or data on nearby points of interest. Since we intend to use more than just the technical parameters data, and additionally, we would like to focus specifically on data from the Czech Republic, we must compile our own dataset.

Since we aim to use image data in our experiments, we focus on real estate listing websites. So, in this section, we will take a look at the various Czech real estate listing websites. We observe what data they offer and if they allow us to download the data. The following websites were considered:

- **bezrealitky.cz**[1]
- **ceskereality.cz**[2]
- **eurobydleni.cz**[3]
- **mmreality.cz**[4]
- **reality.idnes.cz**[5]
- **sreality.cz**[6]

From these options, we chose the *bezrealitky.cz* webserver. The technical data about the real estate object included in the listings is quite basic, but it is comparable to the other webservers. The main advantage of this website is that the listings are consistent in terms of the filled-out technical parameters of the object. Furthermore, the technical parameters are also consistent in

[1] https://www.bezrealitky.cz/
[2] https://www.ceskereality.cz/
[3] https://www.eurobydleni.cz/
[4] https://www.mmreality.cz/
[5] https://reality.idnes.cz/
[6] https://www.sreality.cz/

terms of presentation. The data are presented in a table with predefined parameter names. In addition, the webserver includes data on nearby points of interest, which is fairly unique among the real estate websites. At the time of writing this thesis, the webserver contained approximately 1000 for sale flat listings and 2000 to rent flat listings. Although this is not the best in terms of amount, we consider it sufficient for the compilation of our dataset, as other works successfully used datasets of similar size. As a bonus, the HTML (Hypertext Markup Language) source of the real estate listing pages does not utilize a lot of JavaScript, which makes the pages static. This simplifies the web scraping process.

To summarize, the *bezrealitky.cz* webserver offers all the data that we require. The data is consistently presented and filled out. The HTML sources are simple to scrape, and the website does not prohibit web scraping as of the time of writing this thesis [6].

3.2 Data extraction

We chose the source of our dataset. Now, we extract the data. In this section, we describe the implementation of a web scraping program to extract the data from the source real estate listing website.

3.2.1 Used tools and software libraries for data extraction

The program was implemented using the Python 3 programming language.[7] This language was chosen due to the sheer amount of available tools and software libraries written for it. As well as the relative ease of use and the language's general popularity for the tasks that it will be used for in this thesis. The language contains quality software libraries for performing URL (Uniform Resource Locator) requests, HTML parsing, data manipulation, data analysis, data visualization, as well as libraries for machine learning. In the implementation of the web scraping program, the following Python libraries were utilized:

- **Jupyter Notebook** – Jupyter Notebook [7] is an open-source web platform that allows for interactive computing using the Python programming language. Jupyter Notebook allows the execution of individual cells containing blocks of code. This feature is useful for the development of a web scraping program since it provides a simple way to avoid running certain parts of the program more than necessary. We can also see the output of the individual cells, right under the cell itself, without running the whole program, which helps with visualization and debugging. Furthermore, Jupyter Notebook allows us to easily share the code in the form of so-called notebooks.

- **pandas** – Library [8] offering data structures and operations for data manipulation and analysis in Python. We mainly utilize the primary data structure of the library – the tablelike *DataFrame*, to store our extracted data and the compiled datasets.

- **Requests** – HTTP (Hypertext Transfer Protocol) library [9], built for streamlined sending of HTTP requests. We use it to download data (HTML sources and image data) from a given URL.

- **Beautiful Soup** – HTML parsing library [10], that allows for relatively simple extraction of data from HTML sources. We use it to locate and extract our desired data from downloaded HTML sources.

[7] Specifically version 3.8.10: https://www.python.org/

3.2.2 Web scraping

To scrape the data of each listing, we need a way to programmatically access the webpage of each listing and download its HTML source. On the website, we can view all real estate listings in the form of a paginated list of 10 listings per page. From these pages, we can scrape the links to the individual listings. Each of the paginated pages has a defined URL structure. Meaning, that in order to open all of the pages, we only need to know how many there are.[8] To get the links, we first scrape the number of paginated pages, generate all of the links to the paginated pages, and then download the HTML sources. Since downloading the HTML sources can be time-consuming, we periodically dump the HTML data to a disc in the form of a pickle file. This way, we only need to download each source once, instead of querying the website every time while developing and debugging the scraper. Once we have all the HTML sources, we locate and extract the links to the individual real estate listings using the Beautiful soup Python library.

Now that we have all the links to the real estate listing webpages, we can download the HTML source of every single listing. Again, we periodically dump the downloaded HTML sources to disc in the form of a pickle file. With the real estate listing HTML sources downloaded, all that is left to do is locate the desired data in the source using HTML tags and then extract it. From each real estate listing page, we extract the following data: address, coordinates, full-text description, a table containing technical parameters, and a table containing points of interest parameters. For some listings, we also save the included image data to be used later in our dataset containing features extracted from image data. The idea is to download as much data as we can so that we can reconstruct the listing later.[9]

The whole source code of the web scraper can be found on the included medium in the form of an interactive Jupyter notebook.

3.2.3 Overview of extracted data

In this subsection, we describe the structure of the extracted data. Since we downloaded more data than just the technical parameters of the real estate object, we split the data into multiple datasets. We describe each of those datasets below in the order that they appear on the source website. Note that all of the we download the data of listings of flats to rent. We use ”real estate object” to refer to the flats to rent. Similarly, we use ”price” to refer to the rent price.

3.2.3.1 Included images

Image data is one of the main reasons why we even have to compile our own dataset and will be used in chapter 5 – *Image data experimentation*. We will extract parameters further describing the real estate object from the image data and use these additional parameters to try and improve the price prediction accuracy. We save the image data of a reduced amount of listings on the disc. Each listing has its included images saved in a separate folder. The name of the folder is the listing identifier scraped from the website, this way, we can later easily reconnect the image data with the given listing.

3.2.3.2 Listing descriptions

Full-text descriptions of the listed real estate objects will not be used in this thesis. Nevertheless, they were still downloaded. The descriptions could prove to be useful later, allowing for some form of data extraction from full-text, or to serve a complementary role during the extraction of parameters from image data. Since we do not want the descriptions to be needlessly included

[8] Or simply keep opening until we encounter an error.
[9] e. g. if the listing is no longer active on the website

in the technical parameters dataset, we create another dataset just for the descriptions, indexed by the listing identifier scraped from the website.

3.2.3.3 Technical parameters

This is the most important of the downloaded data. Technical parameters of the given real estate objects will allow us to predict the price of a given object using machine learning models.

Since the technical paramaters are inputed by humans, there are some errors, missing data and other inconsistencies. We dedicated the next two sections to the technical paramaters. In section 3.3 we clean the data, and in section 3.4 we explore and visualize the data.

We include the final dataset compiled from the data on technical parameters in the attached medium. From this point, we use "dataset" to refer to the dataset compiled from the data on technical parameters.

3.2.3.4 Points of interest

Apart from the usual full-text description, images, and technical parameters, our data source website also contains precalculated data about the various points of interest near the given real estate object. Each listing contains the information about the following points of interest: listing.

- Public transport stop
- Post office
- Grocery store
- Bank
- Restaurant
- School
- Kindergarten
- Sports field
- Playground

The website provides the three parameters for each point of interest. The distance to the point of interest, either in meters or kilometers, the estimated time to reach in minutes, and an indactor of whether or not the point of interest can be reached by foot.

To include this data in our dataset, these parameters would have to be encoded as three columns for each point of interest respectively. This is a lot of extra columns. For now, we store the data on points of interest in its own dataset, indexed by the listing identifier scraped from the website.

3.3 Data cleaning

In this section, we clean up the dataset in order to provide better visualization. We translate the data to English, drop unnecessary columns, transform the data into proper formats, clean up poorly represented values of categorical features and handle outlier data.

3.3.1 Columns

The initial dataset compiled from the raw downloaded data describing the technical parameters of flats to rent contained 35 columns and 2707 rows. To get 2707 items, we scraped the source website twice, and then combined the results.

35 columns seems like a decent amount of columns, however, a great number of them were too specific to their individual listings. Some columns had as much as 95% of data missing.

Out of the 35 columns, 3 columns contained data that does not hold any information – so we removed them. Another 12 columns were missing more than 50% of data. For missing data, the decision is not so simple. The information that a record is missing a value in a certain column can still be very useful, in and of itself.

Ultimately we decided to remove these columns and use the more consistently filled-out technical parameters. As some of the columns with missing data were duplicates, some were too similar to already existing consistently filled-out parameters. Additionally, we remove the fees and security deposit columns as, from our observation, they are wildly inconsistent.

In contrast, we also add some columns. We transform the values of a column containing the address of the object. It is too specific, so we replace it with a new column, containig only the region that the object is situated in. This reduces the possible values to the 13 regions of the Czech Republic.

We also alter the column representing the real estate objects layout. It holds the information about the number of rooms and whether or not the kitchen in open plan. We split the columns into 2 new columns. One column representing the room count and one representing whether or not the kitchen is open plan.

Finally, we translate the names of the remaining colums to English. After the removal, we are left with a total of 19 columns. Some of the records in the dataset still have missing values in certain columns. This will be handled later in the thesis, during the data preprocessing part of machine learning model implementation. So, after some minor initial processing, the dataset of technical parameters of the individual real estate objects contains the following 19 columns.

Geographical columns

- *region*
- *longitude*
- *latitude*

Numerical columns

- *price*
- *room_count*
- *living_area*

Categorical columns

- *condition*
- *furnishing*
- *ownership_type*
- *building_type*
- *floor*

Boolean columns – Columns containing a simple *true* when the object has, for example, a balcony, or *false* when the object does not have a balcony.

- *open_kitchen*
- *balcony*
- *terrace*
- *cellar*

- *loggia*
- *parking*
- *lift*
- *garage*

3.3.2 Value formatting

All of the values in our data set are represented as a strings, and the categorical columns, contain values in the Czech language. Since we use English, we translate the values of the categorical features.

The numeric colums should be represented as numbers. The string representations are consistent and the values contain no floating points. Due to this, we can simply filter out all non-numeric characters from the string and then typecast the result.

3.3.3 Category merging

During initial exploration, we observed that some categorical columns contain values, that are not well represented in the dataset. If possible we merge the minority categories with a better represented category, or simply contain them in an *other* category. If the values are too rare, we remove the few irregular records from our dataset.

3.3.4 Outliers and possible errors

By errors, we mean data entry errors. We remove all items that are missing, what we conder to be, the core technical parameters that no listing should be missing. Specifically, the *price* – the targe variable, *living_area*, and *layout* features.

Since our dataset is mostly comprised of "average" real estate objects, we will now remove some of the outliers to make the price prediction more consistent. As for the numeric values, we will remove objects that are in the top and bottom 2 % of the *price* and *living_area* features. This change restricts our dataset to the price range of 5,000 to 39,000 Czech crowns, and the living area of 22 to 108 square meters. Albeit somewhat arbitrary, we believe these values to be reasonable. Futhermore the trimming removes possible data entry errors, or non standard listings. We include the effect the removal has on the statistical description of these features in a Table 3.1 and Table 3.2.

Table 3.1 The statistical description of the numeric features before outlier removal.

Function	Price (CZK)	Living area (m^2)
count	2690	2690
mean	14571	53
std	8868	23
min	1	13
25%	9500	38
50%	13000	50
75%	16827	63
max	150000	322

Table 3.2 The statistical description of the numeric features after outlier removal.

Function	Price (CZK)	Living area (m^2)
count	2482	2482
mean	13809	52
std	5838	17
min	5000	22
25%	9900	39
50%	13000	50
75%	16211	62
max	39000	108

3.4 Data exploration

At this point in the thesis, we still know very little about our dataset We know its source, what columns it contains, and how many records it has. To use this dataset as a training dataset for machine learning models, it is desirable to know as much as we possibly can about it. There still may be errors or outliers in the data that we have to handle. Additionally, there may be various patterns in the data that we may not be aware of.

In this section, we will explore and visualize our dataset with the goal of better understanding the structure and distribution of its data points, as well as uncovering patterns and correlationsthat are present in the dataset.

3.4.1 Numeric features

We start with the numeric features, specifically with the real estate object price – our target variable, and the living area of the real estate object.

From Figure 3.1 and Figure 3.2 we can see that the distribution of both of those features is of a similar shape. Both features are skewed to the right, which makes sense in this context, as there will hardly be objects that are outliers by being significantly cheaper or having a smaller living area.

The correlation between price and living area calculated using the standard Pearson correlation coefficient is 0.6.

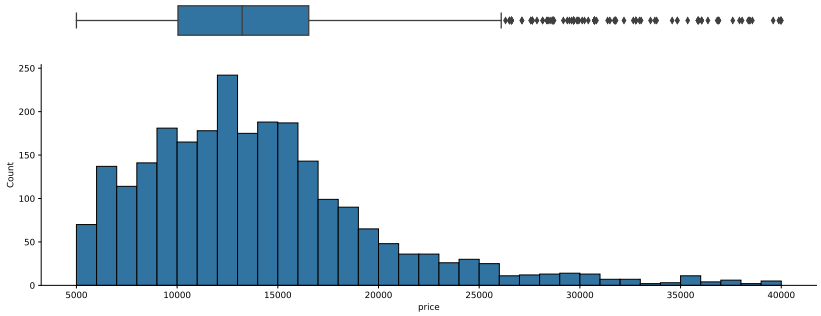

Figure 3.1 Distribution plot of the real estate object price

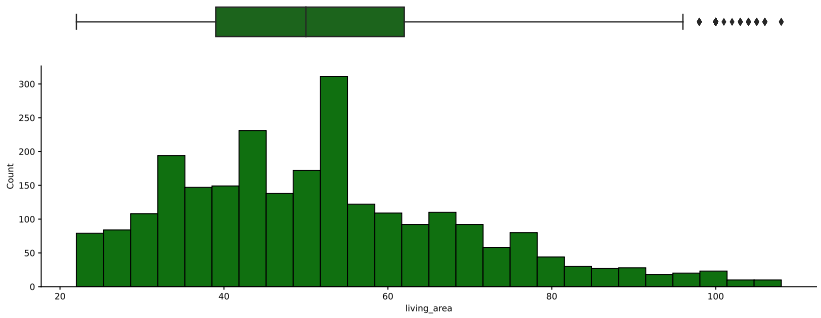

Figure 3.2 Distribution plot of the real estate object living area

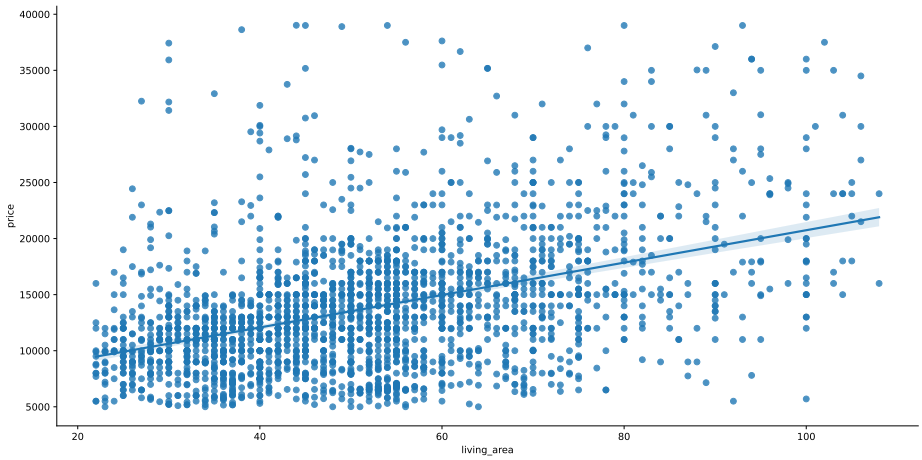

Figure 3.3 Regresion plot of price and living area.

3.4.2 Categorical features

Now, we will explore the categorical features of the dataset. We will visualize their distributions and their relation to the target variable – the price of the real estate object.

3.4.2.1 Layout

As we can see from the distribution plot in Figure 3.4, the average real estate object in this dataset has two rooms. This is aligned with the prevailing room count in multidwelling buildings, according to [11].

Additionally, we observe that real estate objects with open plan kitchen seem to have higher price, while also being the prevailing option.

3.4.2.2 Condition

The distribution of the condition feature in Figure 3.5 is a bit odd. Either the sellers are too optimistic, or it is worth the trouble to renovate or, at least, stage the real estate object before selling, as real estate agents suggest [12].

A human would probably gloss over this feature, look at the images and evaluate the condition for themselves. For a computer, it is not as simple. This situation is the motivation for the experimental chapter of this thesis, where we try to capture the features from the image data and provide them to the machine learning model to try and increase the price prediction accuracy.

3.4.2.3 Furnishing

The furnishing features relation the price of the real estate object behaves unexpectedly. Although the more furnished the object is, the higher the price, we expected a stronger relationship. In our dataset, it is hardly perceivable. One valuable piece of information visible on the box plot in Figure 3.6 is that as the real estate objects increase in price, they are more likely to be fully furnished.

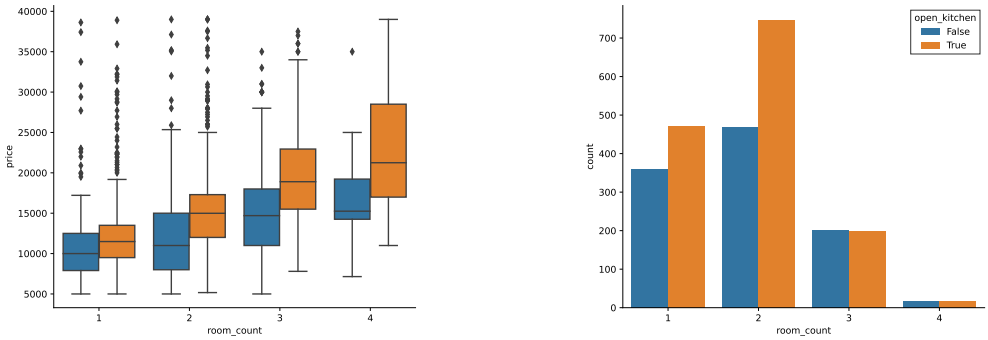

Figure 3.4 Box and distribution plot of the *layout* feature

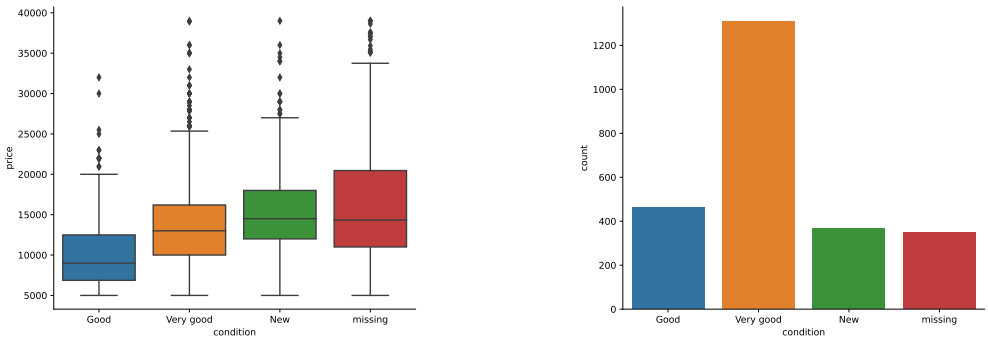

Figure 3.5 Box and distribution plot of the *condition* feature

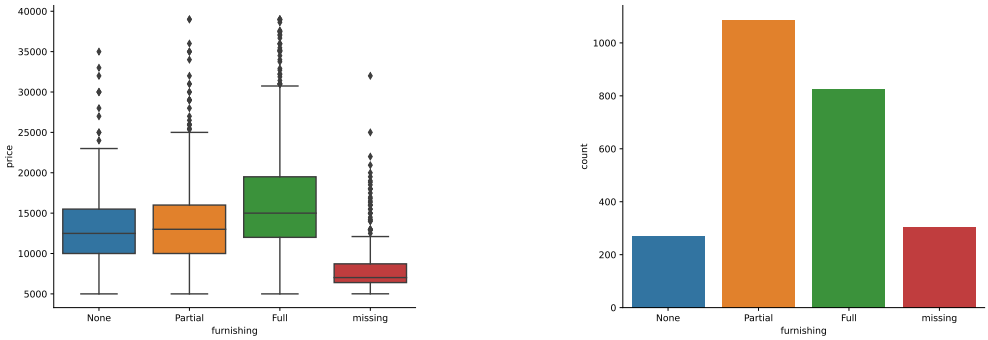

Figure 3.6 Box and distribution plot of the *furnishing* feature

3.4.2.4 Ownership type

The ownership types present in our dataset are personal, cooperative[10] and "other". In Figure 3.7, the cooperative category is merged into "other". We observe that personal ownership flats have a slightly higher median price but are also more varied. Based on the plots, it does not seem like ownership type affects the price, but we could also just be missing data.

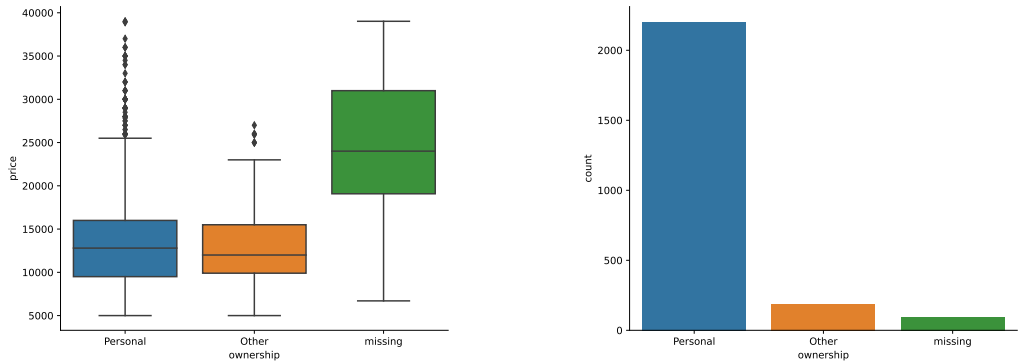

Figure 3.7 Box and distribution plot of the *ownership_type* feature

3.4.2.5 Building type

There are two prevalent building types – brick and panel. Brick has the highest median price but also considerably more outliers and variance. The rest of the types are not represented very well in our dataset, and we will merge them into the *other* type.

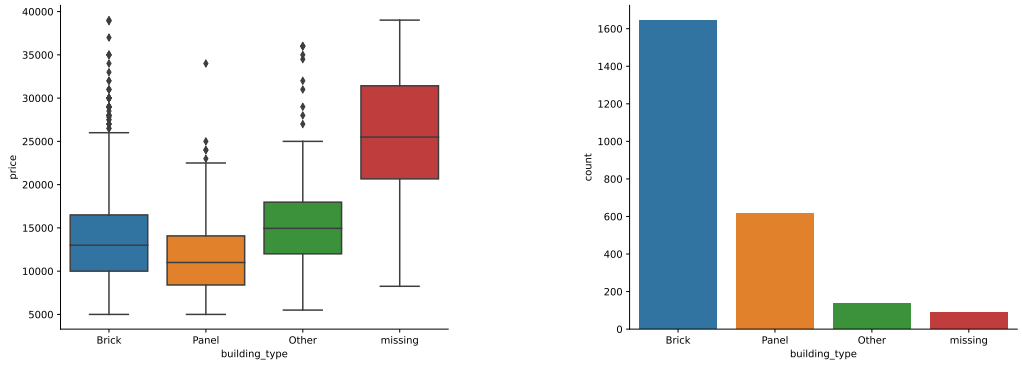

Figure 3.8 Box and distribution plot of the *building_type* feature

[10] In Czech: družstevní

3.4.2.6 Floor number

As we can see from the box plot in Figure 3.9, the floor feature seems to have no immediately noticeable effect on the real estate object price. However, there is interesting information in the distribution plot. Most of the real estate objects are on the lower floors of the building. Since the source of the data is active listings on a real estate listing webserver, the distribution could hint at several things. Either that higher floor is more desirable, which should increase the price. Another possible explanation is that mid to high-rise buildings are not well represented in our dataset. According to a 2008 analysis of the housing construction in the Czech Republic, a newly constructed multi-dwelling building in the Czech Republic has 4.6 floors on average [11].

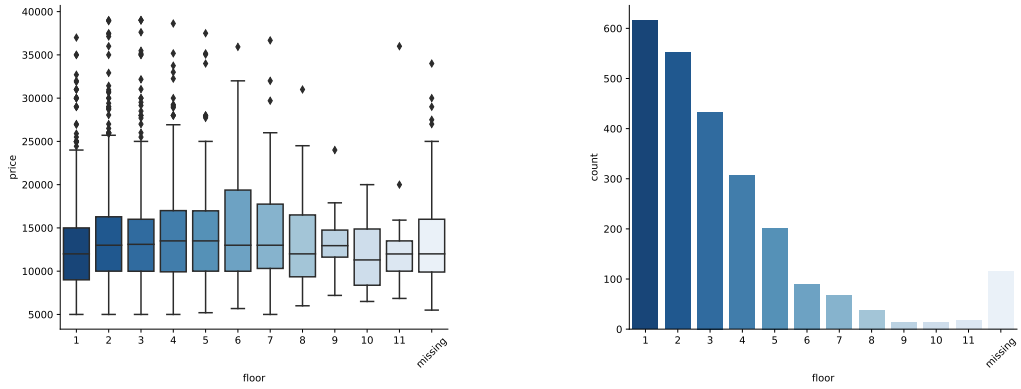

Figure 3.9 Box and distribution plot of the *floor* feature

3.4.2.7 Region

As we can see from the box plot in Figure 3.10, the real estate location plays a decent role in the price. Prague, the capital of the Czech Republic, has the highest prices, followed by the South Moravian region – mainly the city of Brno. Additionally, we notice that the Prague region has considerably more variance than the other regions and that, consequently, it might be worthwhile to split this into smaller level administration units.

Since our dataset contains the longitude and latitude features, let's plot the locations of our real estate objects on the map of the Czech Republic.

Because the real estate object covered in our dataset is a flat to rent, most of the listings are from bigger cities. Specifically, the listings are concentrated in Prague and the Moravian-Silesian, South Moravian, and Central Bohemian regions. On a city level, this corresponds to Prague, Ostrava, Brno, and finally, cities surrounding Prague, such as Kladno.

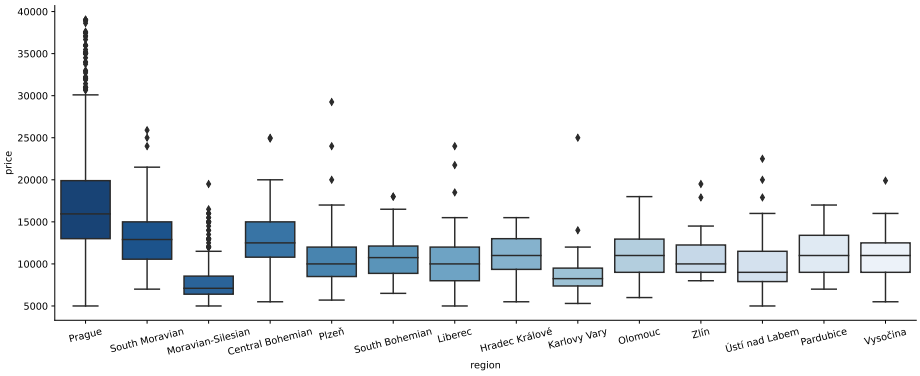

Figure 3.10 Box plot of the *region* feature

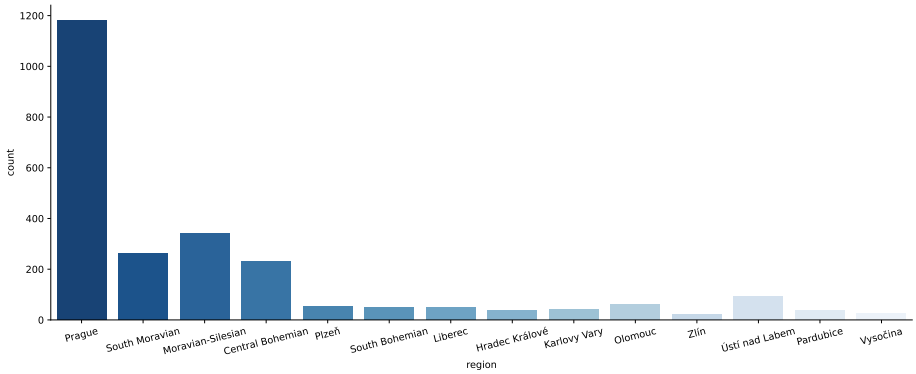

Figure 3.11 Distribution plot of the *region* feature

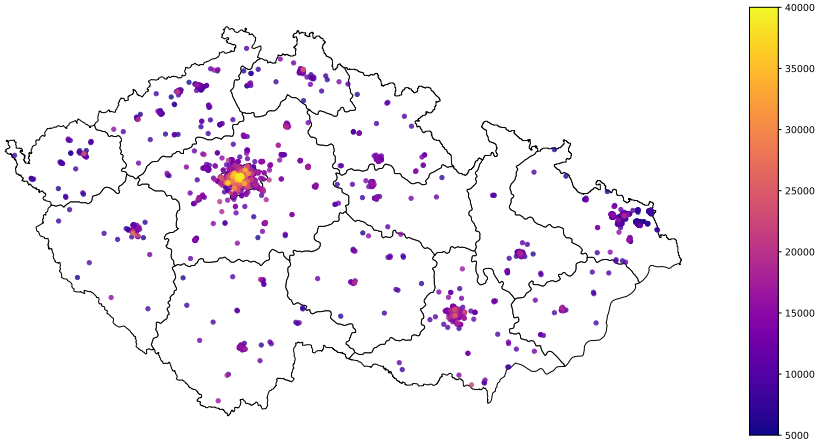

Figure 3.12 Individual datapoints location on a map, colored by the price of the real estate object

3.4.3 Boolean features

Finally, we look at the boolean features, describing the various amenities of the real estate object.

Interestingly, not a single amenity covered in the boolean data is predominantly present in the real estate objects in our dataset. Every boolean feature seems to have a small relation to the price. *cellar* and *parking*, being the only features that negatively influence the price.

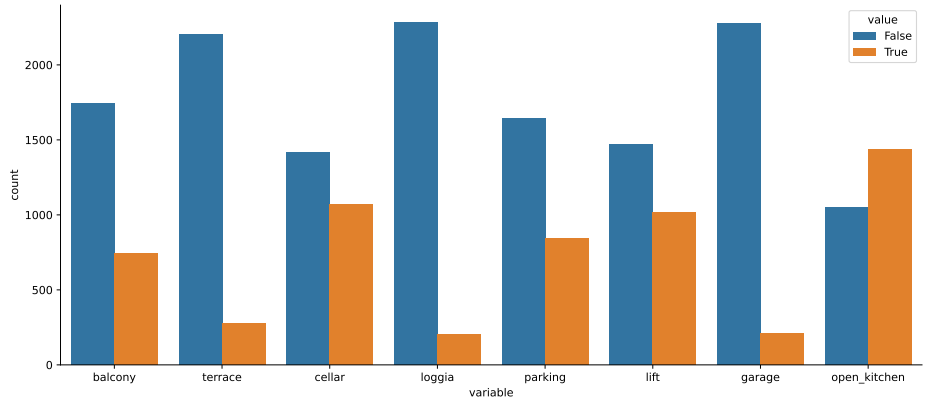

Figure 3.13 Distribution plot of boolean features

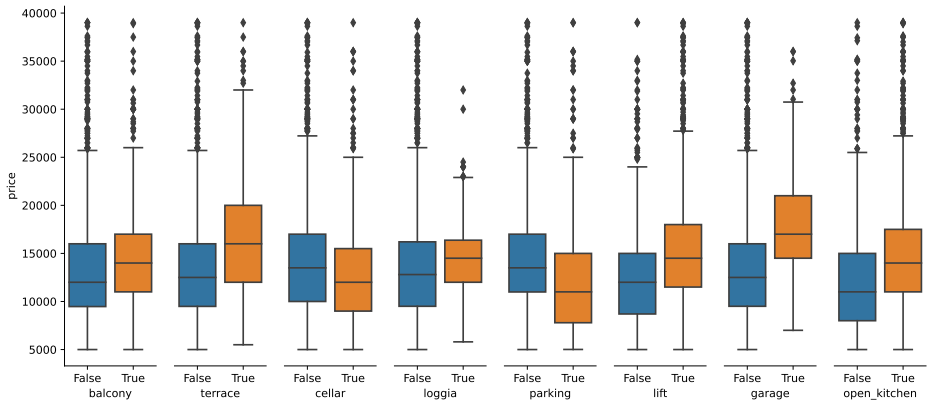

Figure 3.14 Box plot of boolean features

3.5 Chapter summary

In this section, we analyzed the data available on Czech real estate listing webservers and selected a webserver as a source for our dataset. We implemented a web scraping program to download the data of individual real estate listings, including image data from the source webserver. From this data, we compiled a dataset of technical paramaeters, which we cleaned, explored, and visualized.

Chapter 4

Price prediciton

In this chapter, we use machine learning models with our compiled dataset to predict the price of a real estate object based on its technical features. We describe the error metrics we use to measure the error rate of the price predictions produced by the machine models. We describe the used models, and finally, the process of training the models and the achieved results.

4.1 Used tools and software libraries for price prediction

Following the data chapter of this thesis, we use the Python 3 programming language and its libraries to implement our machine learning models. The central motivation for using Python is the *scikit-learn* [13] library for machine learning. During the implementation of our chosen machine learning models, we utilize the following Python libraries:

- **Jupyter Notebook** – *Jupyter Notebook* [7] is an open-source web platform that allows for interactive computing. We use *Jupyter Notebook* as a development environment that allows us to easily troubleshoot and visualize intermediate steps of the model training process.
- **pandas** – Library [8], offering data structures and operations for data manipulation and analysis. We mainly utilize the primary data structure of the library – the tablelike *DataFrame* while handling our datasets.
- **NumPy** – Library [14] for scientific computing and data manipulation. We mainly use *NumPy* to manipulate arrays of data.
- **scikit-learn** – Library [13], offering implementations of many machine learning methods and tools. We heavily use the *scikit-learn* library throughout the implementation chapter.
- **XGBoost** – Library [15], offering an efficient implementation of a gradient boosted machine learning model.

4.2 Data preprocessing

Before we begin training the machine learning models on our dataset, we must prepare the dataset. In this section, we handle data missingness and then encode categorical features to numeric values.

4.2.1 Missingness

Since the implementations of the machine learning models we use in the *Price prediction* chapter cannot handle missing data, we have to handle the data for them. We do not want to remove every item that is missing a column value and so we have to fill in the missing values.

One option to handle missing data is imputation. Due to the missing features in our dataset being categoric, and our usage of one-hot encoding, the imputation becomes non-trivial. Additionally, we believe that filling in the values with the most frequent category value would not be correct in the context of our dataset.

In this thesis, we resort to handling the missingness simply by using a *missing* value to represent it.

4.2.2 Categorical features encoding

The machine learning model implementations from the *scikit-learn* library cannot handle non numeric data. So, we have to encode our categorical features to numeric values. In our dataset, there are two types of categorical features. Nominal categorical features – features that have no intrinsic order to the categories. Ordinal categorical features – features that have an order to their categories. For nominal features, we use one-hot encoding – the categories are split into their own boolean so-called dummy features, and the original feature is removed. For ordinal features, normally, we would encode the ordinal features using label encoding. Because we made the decision to represent missing values as a value in the given category, the order is lost. Thus, we use one we use one-hot encoding for ordinal features as well.

4.3 Error metrics

In this section we introduce the error metrics that we use in this thesis to evaluate the price prediction of the used machine learning models. To provide these brief descriptions, we studied the *An introduction to statistical learning* book [16].

MAE – The Mean Absolute Error metric simply reports the average error across the predicted data points. To account for negative errors, the metric considers the absolute value of the error. MAE is in the same units as the target variable, which makes it easy to interpret. We value interpretability because our target variable is a monetary amount.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|Y_i^{\text{real}} - Y_i^{\text{predicted}}| \tag{4.1}$$

MSE – The Mean Squared Error considers the squared error instead of the absolute error. This results in greater errors being assigned exponentially worse values. The tradeoff is that MSE has worse interpretability, as the unit of the error is not the same as the unit of the target variable. We mainly use use this metric during hyperparameter tuning.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i^{\text{real}} - Y_i^{\text{predicted}})^2 \tag{4.2}$$

RMSE – The Root Mean Squared Error metric is an extension of the MSE metric. It is the square root of MSE. This makes the error unit aligned with the unit of the target variable so that it is easier to interpret.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i^{\text{real}} - Y_i^{\text{predicted}})^2} \tag{4.3}$$

4.4 Machine learning models

In this section, we list the machine learning models we use for price prediction. We briefly introduce each model, and then we describe the precise steps taken while training the model on our dataset, mentioning the specific tools we used. Before we start with the models, we introduce a generalized machine learning model training process that we used, so as not to repeat ourselves.

Again, same as with the error metrics, to provide the model descriptions, we studied *An introduction to statistical learning* book [16].

4.4.1 General training process

In this subsection, we describe the general training process that we use across all models to train them on our dataset and get prediction results. Note, that all of the mentioned software classes and methods are from the *scikit-learn* library.

First, we split the dataset into training and testing datasets using the *train_test_split* method. We use a 75% training to 25% testing ratio.

We instantiate a machine learning model using the given implementation class. And define tested hyperparameter values using a Python dictionary. The considered hyperparameters, as well as their values, are specific to each machine learning model.

Next, we define cross-validation using *RepeatedKFold* class. Specifically, we use 5-fold 3 times repeated cross-validation. We use 5 as our k value since it performs well regarding the bias-variance tradeoff [17].

Finally, we use the *GridSearchCV* class, along with our model and cross-validation, to tune the model hyperparameters.

Now that we know the optimal[1] hyperparameters, we can train the given model using the training dataset and then use the model to predict the real estate object price on our testing dataset. We use the *fit* and *predict* methods respectively. The methods are provided by the model implementation.

As the last step, we then measure the error of the model's prediction using methods from the *metrics* module implementing the error metrics mentioned in the section 4.3.

4.4.2 Multiple linear regression

Linear regression models assume that there is a somewhat linear relationship between the target variable and the features. The linear models represent the prediciton as a linear combination of features. Thus, all that is needed for prediction are the linear combination coefficients.

The simple regression model estimates these coefficients using the ordinary least squares method during during training of the model.

We chose to use Linear Regression to use as sort of a baseline model. Additionally, since we predict a price of a real estate object, we can use the coefficients as an estimation of feature importance.

[1] From the value ranges that we provided to the *GridSearchCV* class.

4.4.2.1 Implementation

We use the *LinearRegression* class from the *linear_model* module of the *scikit-learn* library as our linear regression implementation. We train the model following the general procedure mention above.

The multiple linear regression model achieved MAE: 2141, RMSE: 2990 on the training dataset, and MAE: 2341, RMSE: 3357 on the testing dataset.

4.4.3 Ridge Regression

The ridge regression model is an extension of the linear regression model. It aims to solve the inaccuracy of the ordinary least squares method when multicollinear features are present. The ridge regression model achieves this by using regularized least squares method to estimate the coefficients during the model training.

4.4.3.1 Implementation

We use the *Ridge* class from the *linear_model* module of the *scikit-learn* library as our Ridge Regression implementation. We train the model following the general procedure mention above.

The Ridge Regression model achieved MAE: 2146, RMSE: 2993 on the training dataset, and MAE: 2341, RMSE: 3358 on the testing dataset. We used the following hyperparameter value:

- `alpha = 0.5`

Additionally, we list features with the highest coefficients in Table 4.1. We observa that the location of the real estate objects heavily influences the price prediction in this ridge regression model.

Table 4.1 Features with the highest coefficients in the ridge regression model trained on the dataset of technical parameters

Feature	Coefficient
d_building_type_missing	7692
d_region_Prague	4587
d_region_South_Moravian	2126
d_condition_New	1278
lift	1068

4.4.4 Random Forest

Random forest is an ensemseble bagging based model. Ensemble meaning it utilizes multiple machine learning models to produce one result. Bagging meaning the models are fitted in parallel and the result is constructed by voting, or averaging the results of the individual models. Specifically, the random forest model trains several decision trees using random samples of the dataset in parallel and then uses averaging to produce a prediction.

We chose to use the random forest model because during our research, the *Related works* chapter, ensemble models proved to perform well when it comes to real estate price prediction.

4.4.4.1 Implementation

We use the *RandomForestRegressor* class from the *ensemble* module of the *scikit-learn* library as our random forest implementation. We train the model following the general procedure mention above.

The random forest model achieved MAE: 2097, RMSE: 3055 on the training dataset, and MAE: 2160, RMSE: 3214 on the testing dataset. We used the following hyperparameter values:

- `n_estimators = 300`
- `max_depth = 10`
- `min_samples_leaf = 4`
- `min_samples_split = 3`

4.4.5 XGBoost

XGBoost is a specific implementation of an ensensemle (extreme) gradient boosting based model. Same as random forest, it utilizes multiple machine learning models to produce one result. Boosting, the alternative to bagging, means that the individual models are fitted in series, and are taking the results of the previous model into account – usually by utilizing data weighting. And finally, gradient boosting, meaning the method by which the erros are minimized during the boosting.

We chose to use XGBoost model because like random forest, it is an ensemble model – but, unlike the bagging based random forest, it is boosting based. Additionaly, XGBoost has a reputation for performing well at data science competitions [18].

4.4.5.1 Implementation

XGBoost is the only model we use that is not implemented by the *scikit-learn* library. We us the implementation from the *xgboost* library – specifically the *XGBRegressor* class. We train the model following the general procedure mention above.

The XGBoost model achieved MAE: 1986, RMSE: 2886 on the training dataset, and MAE: 2185, RMSE: 3188 on the testing dataset. We used the following hyperparameter values:

- `n_estimators = 120`
- `max_depth = 3`
- `learning_rate = 0.075`
- `scale_pos_weight = 0`

4.5 Chapter summary

In this chapter, we trained several machine learning models for price prediction using our compiled dataset of technical parameters of real estate objects – specifically flats to rent. We introduced error metrics that we used to evaluate the price prediction results. We described a generalized model training process we used to train each model. Additionally, we briefly described each model, mentioned implementation details specific to the model, and presented reached results.

Overall, the ensemble models outperformed the linear regression based models. The best performing model was the random forest with MAE: 2160, and RMSE: 3124 on the testing dataset. To summarize, we provide a full comparison in the Table 4.2 and Table 4.3.

Table 4.2 Results of the machine learning models price prediction on the training dataset of technical parameters

Model	Train MAE	Train RMSE
Linear Regression	2141	2990
Ridge Regression	2146	2993
Random Forest	2097	3055
XGBoost	1986	2886

Table 4.3 Results of the machine learning models price prediction on the testing dataset of technical parameters

Model	Train MAE	Train RMSE
Linear Regression	2341	3357
Ridge Regression	2341	3358
Random Forest	2160	3214
XGBoost	2185	3188

Chapter 5

Image data experimentation

In this chapter, we design, perform and evaluate experiments using our trained machine learning models and the downloaded data about real estate objects.

At this point in the thesis, we compiled a dataset from data on technical parameters of real estate objects and used it to train several machine learning models. Apart from the data on technical parameters, we also downloaded image data for a reduced amount of real estate listings. In this chapter, we use this data to try and improve the real estate price prediction of our chosen machine learning models.

5.1 Motivation

One of the goals of this thesis is to experiment with using image data to improve the price prediction accuracy. In this section, we describe the process of extracting features from image data.

When a human is looking at real estate listings, they rarely start with the technical parameters. Instead, they look at the images, or possibly the full-text description as that is often enough to extract all of the information about the real estate object, that could potentially be included in the technical parameters, as well as a general feel of the real estate object. Additionally, the images can tell the rest of the story, that is not captured within the full-text description or the technical parameters.

For a computer, extracting features from image data and correctly interpreting them is not so intuitive. The approach chosen in this thesis is to analyze the interior images and note what we focus on during this process. We then create descriptive parameters that can be treated the same as the technical parameters and thus be included in the dataset of technical parameters. We can then use this enriched dataset to train our machine learning models without any significant changes. In this thesis, we will extract these features manually – using a human that is provided with a grading methodology for each parameter. We believe, that extracting such parameters from image data using a computer is possible,[1] but the implementation would be beyond the scope of this thesis. Moreover, the features we will be extracting will be very specific to our dataset. Later in this section, we introduce a concrete methodology for extracting each of the mentioned features from the image data. Implementing a methodology for the feature instruction will help us stay as objective[2] as possible, even though the nature of some of the features extracted is inherently subjective.

[1] Although in some cases it is much more complicated than in others.
[2] Or rather, for some of the features, subjective in a consistent matter.

5.2 Feature extraction

In this section, we describe the process of extracting features from image data. We mention problems that we encountered, then we list and comment on the features we extracted.

5.2.1 Encountered problems

In this subsection, we briefly go over the reasons that we did not extract certain initially considered features. Generally, we encountered one of the problems described in the following paragraphs.

Low-quality image data – With low-quality image data, it becomes difficult to recognize different types of material and spot any signs of wear on furniture or appliances. One such feature affected by this is floor type. In a considerable amount of listings, we were not able to distinguish between wood, laminate, and linoleum floor.

Feature homogenity – We cannot use a feature if it is homogenous in our dataset since we gain no information. An example of such feature is a stovetop type. The idea was to judge the age of the kitchen based on the stovetop. Unfortunately, almost every single object had a glass induction stovetop.

Feature subjectivity – A lot of the features that we intuitively presume influence the price are inherently subjective. Or, it is too difficult to create a simple to follow extracting methodology for them. Such features include the perceived quality of material, modernity of the object, or lighting quality. In this thesis, we still want to include such features[3]. Instead of a guide or a concrete grading system, we use example image data and assign values based on visual similarity. Even though the nature of the features may be subjective, the values of the extracted features will be consistent.

5.2.2 Extracted features

In this subsection, we focus on the features that we extracted from the image data. We mention the idea and motivation behind extracting these images. We describe the features, and finally, we introduce an extracting methodology for each feature.

5.2.2.1 Perceived quality and modernity

The perceived features are problematic, as they are difficult to create an extracting methodology for. We still include them, as we believe they can tell us a lot about the real estate object. Intuitively, if an object has more quality materials, more money has been invested into it, and the price will be higher. As for the modernity of the space – furniture, appliances, or even windows and floors. If the object feels modern, we presume that it is either newly built or recently renovated.

To capture the quality and the modernity of the object we introduce *quality* and *modernity* features. To represent these features, we use a scale from one to three, indicating the level of quality and modernity, with three being the best. Three levels are not enough to capture these features, but in this thesis, we expect to have a relatively small dataset, thus we keep the grading coarse.

In the context of this thesis, by quality, we mean the general state of the object. Observed low quality or cheap materials lower the score, as well as visible wear anywhere in the real estate objects.

[3]To potentially mark an area for future work, if we discover some sort of a measurable relationship with the real estate object price.

By modernity, we mean the overall modernity of the items in the object, or recency of renovation in the object. We look at the modernity in all rooms and grade the feature according to the combined modernity.

Additionally, for both features, a series of example images is included to help guide the human through the feature extraction process. We evaluate a given real estate object based on the grading described above and the similarity to the provided example images.

5.2.2.2 Natural light

Personally, when looking through the listings, the ones that stood out from the rest were mainly due to the lighting. This effect can be observed in the listing descriptions, as well as in literature [19]. The author of the listing description never fails to mention that the real estate object is sunny or sunlit. The natural conclusion is that people would then prefer to live in spaces with a lot of natural light.

To represent this feature, we use a scale from one to three, indicating the levels of natural light, with three being the best.

While extracting this feature, we look at things like the total window count, window to wall ratio, window per room ratio, window orientation, and windows in unexpected places such as a bathroom or a kitchen. We also consider what is outside the window. Big windows are of no use if they are looking straight into another building or a big tree. Generally, we tend to grade down. Each object starts at 3 points, which get reduced as we observe negative attributes.

Additionally, a series of example images is included to help guide the human through the feature extraction process. We evaluate a given real estate object based on the grading described above and the similarity to the provided example images.

5.2.2.3 Color palette

While analyzing the real estate listings, we observe that a color palette seems to be related to the perceived modernity of the object. Which we presume influences the real estate object price. Additionally, interior design trends change every few years, and so do the color palettes. Thus, the color palette feature could help the timeframe in which the interior was renovated, beyond the capability of the *modernity* feature.

Specifically, we noticed that modern objects usually had muted color palettes, with a lot of white, beiges, and gray. The more average objects usually contained white, yellow, and rarely atypical colors.

Atypical colors seem to lower the value of the real estate object, as these colors are very personalized, which means a considerable amount of potential buyers will not like them, which lowers the interest in the real estate object.

These observations align with what real estate agents suggest to do before selling a house. They often advise repainting the walls using neutral colors so that the potential buyer is not distracted and can imagine themselves living in the space [12].

We will try to capture these observations in the *color* feature. Finally, *color* will be a categorical feature with the following values:

- White
- Light
- Dark
- Atypical

By color, we generally mean the predominant color of the walls, which is usually also the predominant color in the space. Although there is an exception – if we observe an unusual color in places like the bathroom and the kitchen cabinets or appliances, we label this real estate object as unusual even if the color is not predominant. For the *color* feature, no example images are included.

5.2.2.4 Split bathroom

The *split_bathroom* parameter indicates whether or not the bathroom is split. By split, we mean that the toilet is in a separate room from the bath. The value of the *split_bathroom* parameter is *True*, if the bathroom contains a toilet. Otherwise, if the toilet has a dedicated room, it is *False*.

In the data exploration subsection, we observed that flats with open plan kitchens are more desired than flats with a separate kitchen room. The motivation behind this feature is to try and measure if this is also the case for the bathroom.

It is theoretically one of the simpler parameters to extract. Although, from the included images, it can sometimes be hard to recognize whether or not the bathroom is split. The exception is if there is a floor plan among the included images. Usually, if the toilet is in a separate room, the room is very small, and we can see the toilet in the picture enclosed by walls. There are no example images or further instructions for this parameter.

5.2.2.5 Shower vs. bath

Since we are in the bathroom already, let's extract another feature. The boolean *shower* parameter, which indicates whether or not the bathroom contains a shower. The value of the *shower* parameter is *True*, if the bathroom contains a shower. Otherwise, if the bathroom contains a bath, it is *False*.

This parameter is the easiest one to extract. Thus, there are no example images or further instructions for this parameter.

5.3 Exploration of extracted features

Using the methodologies mentioned in the above subsection, we manually extracted image data features from 100 real estate listings. The data is not entirely random – we work with flats to rent, situated only in Prague. We filtered the data like this to omit the influence of the location on the price.

5.3.1 Quality, modernity and lightning

By looking at Figure 5.1, Figure 5.2, and Figure 5.3, it might be difficult to tell these features apart. We observe, that the price increases with the value of the features. This is the desired outcome. Although, the features are correlated. We calculate the correlation using the standard Pearson correlation coefficient. For *quality* and *modernity*, the coefficient is 0.72. For *lighting*, 0.48 and 0.42 with *quality* and *modernity* respectively.

Additionally, we observe that the distributions of the values are skewed towards higher values. Either we were too optimistic while grading the real estate objects, or our dataset comprises mostly of objects, that are newly built, or have undergone some form of renovation. If we return to the technical parameters of the objects that we explored in the *Data* chapter – one of the parameters was *condition*, where we observed a similar trend. The condition of the objects ranged from good to newly built – there were almost no objects in bad condition.

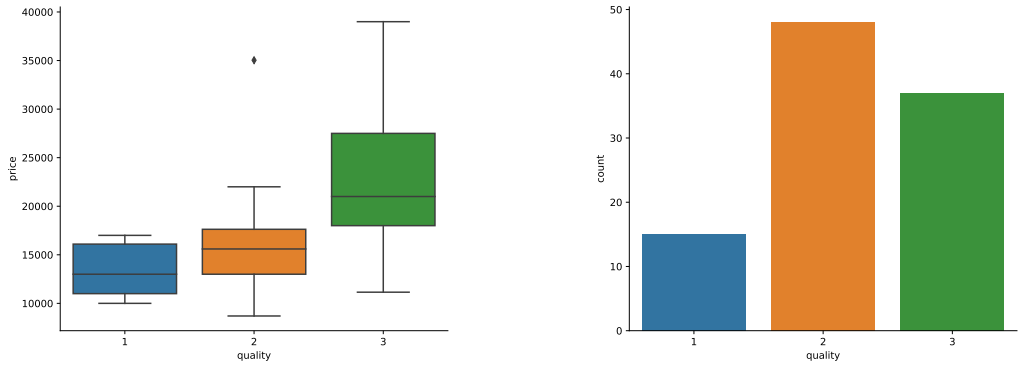

Figure 5.1 Box and distribution plot of the *quality* feature

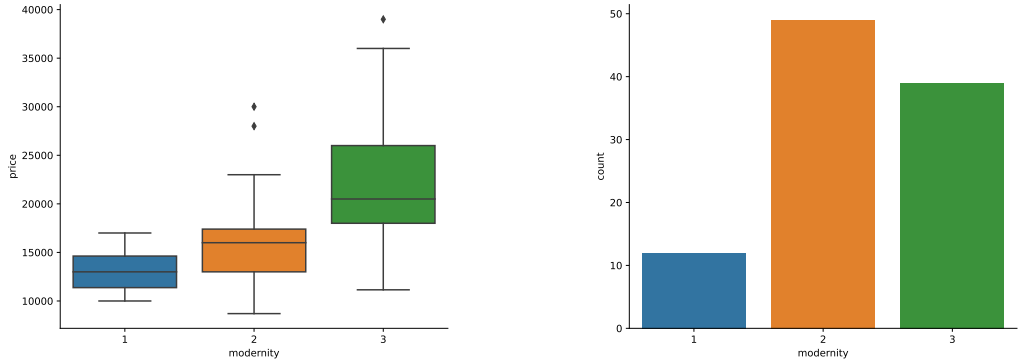

Figure 5.2 Box and distribution plot of the *modernity* feature

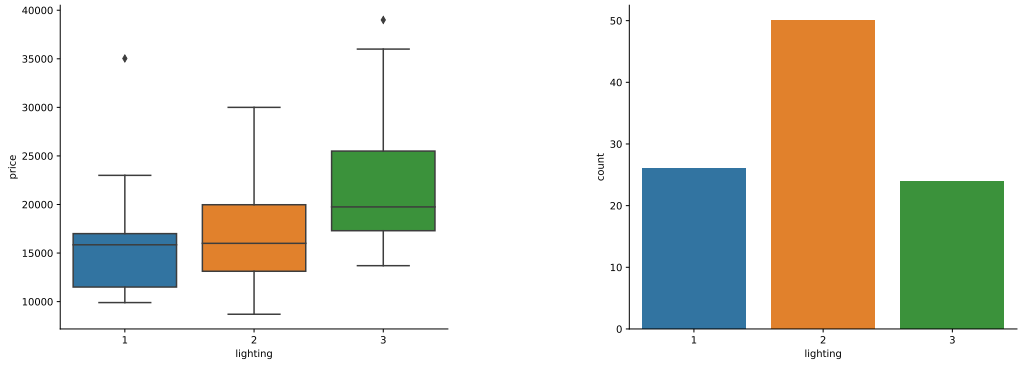

Figure 5.3 Box and distribution plot of the *lighting* feature

5.3.2 Color

The distribution of this feature is disappointing but expected – most of the objects have white walls with no immediately noticeable atypical colors in the object. From the box plot in Figure 5.4, it seems like the atypical colors are associated with lower price objects, which we expected. Unfortunately, due to the unbalanced distribution and our relatively small dataset, we cannot draw too many conclusions from this feature.

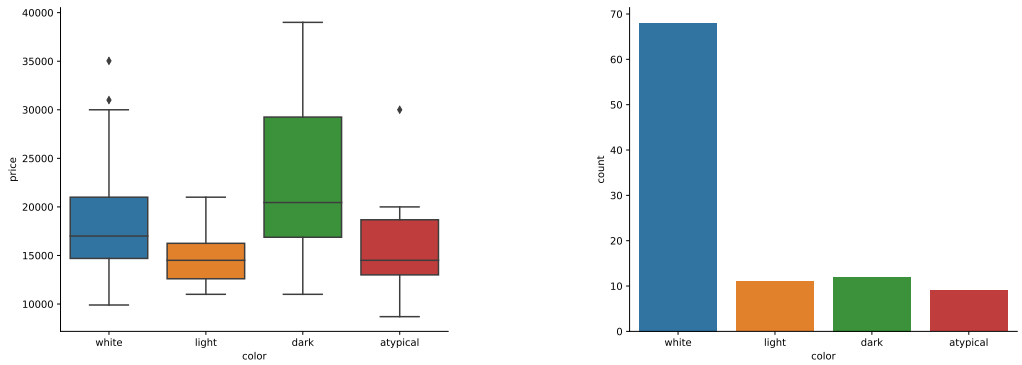

Figure 5.4 Box and distribution plot of the *color* feature

5.3.3 The bathroom features

As we can see from Figure 5.5, the *has_shower* feature seems to have no relation to the real estate object price. Although, the distribution is interesting – we expected showers to be rarer.

The *split_bathroom* feature is a bit more interesting. We observe that the real estate objects that do not have a split bathroom are more varied in terms of price. In contrast, the objects with a split bathroom concentrate around the median price, which is roughly 18000 Czech crowns in the reduced dataset.

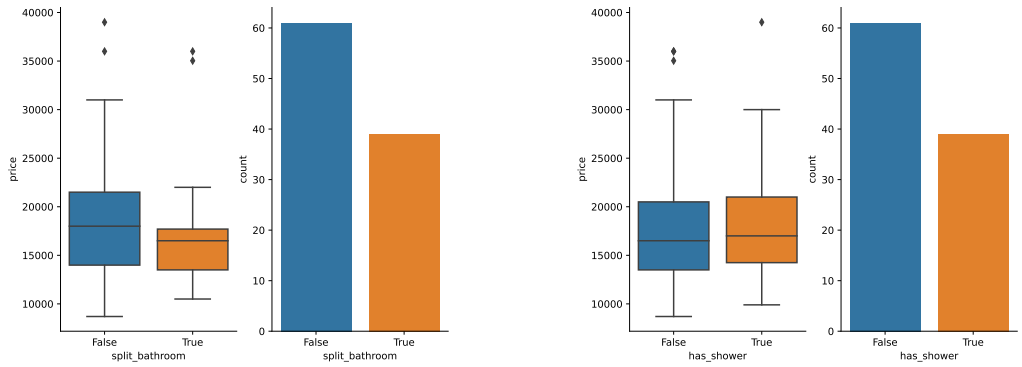

Figure 5.5 Box and distribution plot of the *split_bathroom* and *has_shower* features

5.4 Effect on price prediction

We connect the extracted image features with the technical parameters of the given real estate object using its listing identifier. We filter out the listings missing the image features to get our dataset of 100 items containing both technical and image features.

Then, we use the models mentioned in the *Price prediction* chapter and train them twice on this reduced dataset. First, we omit the image features, and second, we include them. During the training process, we take the same steps already mentioned in the *Price prediction* chapter.

In our experiments, the inclusion of image features provided measurable improvement across all models. We decreased the average error using the test dataset by roughly 10 to 20%. We list the exact results using the test dataset in Table 5.1 and Table 5.2

Table 5.1 Price prediction error values on the image features dataset without image features

Model	MAE	RMSE
Linear regression	3049	3893
Ridge regression	2720	3531
Random forest	3153	3968
XGBoost	3244	4012

Table 5.2 Price prediction error values on the image features dataset with image features

Model	MAE	RMSE
Linear regression	2237	2948
Ridge regression	2228	2880
Random forest	2868	3887
XGBoost	2477	3475

5.5 Chapter summary

In this chapter, we introduced an experiment area of including image data in the price prediction process and the motivation behind it. We identified possible features to extract, wrote up extracting methodologies for these features, and described issues encountered along the way. We then extracted the features from the image data, explored them, and compiled a reduced dataset containing technical and image data features. The extracted features were: *quality*, *modernity*, *natural_light*, *color*, *split_bathroom*, and *has_shower*. We trained several machine learning models using the reduced dataset and observed an increase in price prediction accuracy. We consider image data features worthy of further exploration, mainly in the direction of increasing the dataset size and automatizing the feature extraction process.

Chapter 6
Conclusion

In this chapter, we summarize the goals and the results of the thesis, as well as introduce ideas for future work.

The goals of this thesis were to map out the contemporary use of machine learning methods for real estate price prediction and compile a dataset from publicly available data. Then, to choose and implement a machine learning model for automatic real estate price prediction. And lastly, to experiment with the compiled dataset – mainly using image data.

In the *Related works* chapter, we mapped out the current usage of machine learning methods for price prediction, considering master theses, and journal articles. We recognized a need to use image data in automatic real estate object price prediction, which served as a motivation for this thesis.

In the *Data analysis* chapter, we focused on compiling a dataset. Since we set out to use image data, we had to use a real estate listing website as our data source. So, we analyzed the data available on Czech real estate listing websites and selected a website as a source for a dataset – we used *bezrealitky.cz*. We implemented a web scraping script to download the data of individual real estate objects from the listings, including image data, from the chosen website. Using this data, we compiled a dataset of technical parameters describing individual real estate objects. The dataset was processed to remove any errors and outlier data. Then we thoroughly described and visualized the dataset to gain a better understanding of the data.

In the *Price prediciton* chapter, we described the process of using machine learning models for real estate price prediction. We used the dataset of technical parameters compiled in the *Data analysis* chapter.

Specifically, we described the error metrics we used to evaluate the models. We listed the general steps taken during training the models, including final preprocessing. For each used model, we briefly introduced it and presented the motivation behind using it. Then, we described the implementation specificities and used hyperparameters of each model.

Finally, we compared the model results. The best performing models were XGBoost, and the randon forest model.

In the *Image data experimentation* chapter, we introduced, performed, and evaluated an experiment using our chosen machine learning models and the compiled dataset. Specifically, we experimented with using features extracted from image data to try and improve price prediction accuracy.

We described the process of manually extracting features from image data and introduced a feature extracting methodology for each extracted feature. The extracted features were: *quality*, *modernity*, *natural_light*, *color*, *split_bathroom*, and *has_shower*. We then manually extracted these features from 100 real estate listings. We appended the image data features to the already

processed dataset of technical features. We then trained the models mentioned in the *Price prediction* chapter on this data – with and without the inclusion of the image data. With the image features included, we observed an approximately 10 to 20% decrease in the average price prediction error – measured by the MAE and RMSE metrics. Thus, we demonstrated that features extracted from image data can be used to improve the real estate price prediction accuracy. We consider the image data features experimentation a success and worthy of further exploration.

We consider all the goals of the thesis met. As for future work, we should focus on substantially increasing the dataset size. It would also be interesting to explore the options of extracting the features from image data programmatically, using computer vision methods. In another direction, we could focus on including more unusual parameters, such as more geographical or points of interest data.

Bibliography

1. CONWAY, Jennifer. *Artificial intelligence and machine learning : current applications in real estate*. 2018. Available also from: http://hdl.handle.net/1721.1/120609. MA thesis. Massachusetts Institute of Technology.
2. BUI, Quang-Thanh; DO, Nhu-Hiep; PHE, Hoang. House Price Estimation in Hanoi using Artificial Neural Network and Support Vector Machine: in Considering Effects of Status and House Quality. In: 2017. Available also from: https://www.researchgate.net/publication/317328847.
3. ZHAO, Shujia. *Implementation and Study of K-Nearest Neighbour and Regression Algorithm for Real-time Housing Market Recommendation Application*. 2018. Available also from: https://www.academia.edu/37195157. MA thesis. University of Nottingham.
4. KINTZEL, Joseph. *Price Prediction and Computer Vision in the Real Estate Marketplace*. 2019. Available also from: https://nrs.harvard.edu/URN-3:HUL.INSTREPOS:37365260. MA thesis. Harvard Extension School.
5. O'FARRELL, Stephen. Comparison of Data Mining Models to Predict House Prices. In: 2018. Available also from: https://www.academia.edu/38358601.
6. *Smluvní podmínky serveru bezrealitky.cz* [online]. 2022. Available also from: https://www.bezrealitky.cz/informace/smluvni-podminky. [2022-05-08].
7. *Project jupyter* [online]. 2022. Available also from: https://jupyter.org/. [2022-05-07].
8. TEAM, The pandas development. *pandas-dev/pandas: Pandas*. Zenodo, 2022. Latest. Available from DOI: 10.5281/zenodo.6408044.
9. *HTTP for humans* [online]. 2022. Available also from: https://docs.python-requests.org/en/latest/. [2022-05-07].
10. RICHARDSON, Leonard. *Beautiful Soup: We called him Tortoise because he taught us* [online]. 2022. Available also from: https://www.crummy.com/software/BeautifulSoup/. [2022-05-08].
11. *Analysis of the housing construction in the Czech Republic - in 2008* [online]. Czech Statistical Office, 2009. Available also from: https://www.czso.cz/csu/czso/ari/-analysis-of-the-housing-construction-in-the-czech-republic-in-2008-9m9azc35ol. [2022-05-07].
12. *How to stage your home for a quick sale* [online]. Investopedia, 2022. Available also from: https://www.investopedia.com/articles/mortgages-real-estate/08/staging-home.asp. [2022-05-07].

13. PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*. 2011, vol. 12, pp. 2825–2830. Available also from: `https://scikit-learn.org/stable/`.

14. *NumPy* [online]. 2022. Available also from: `https://numpy.org/`. [2022-05-08].

15. *XGBoost documentation* [online]. 2022. Available also from: `https://xgboost.readthedocs.io/en/stable/`. [2022-05-08].

16. JAMES, Gareth; WITTEN, Daniela; HASTIE, Trevor; TIBSHIRANI, Robert. In: *An introduction to statistical learning: With applications in R*. 2nd ed. Springer, 2021. ISBN 978-1-0716-1418-1. Available also from: `https://www.statlearning.com/`.

17. JAMES, Gareth; WITTEN, Daniela; HASTIE, Trevor; TIBSHIRANI, Robert. Bias-Variance Trade-Off for k-Fold Cross-Validation. In: *An introduction to statistical learning: With applications in R*. 2nd ed. Springer, 2021, p. 206. ISBN 978-1-0716-1418-1. Available also from: `https://www.statlearning.com/`.

18. BEKKERMAN, Ron. *The present and the future of the KDD Cup Competition: An Outsider's perspective* [online]. LinkedIn, 2018. Available also from: `https://www.linkedin.com/pulse/present-future-kdd-cup-competition-outsiders-ron-bekkerman`. [2022-05-07].

19. EDWARDS, L.; TORCELLINI, P. *Literature Review of the Effects of Natural Light on Building Occupants*. 2002. Available from DOI: `10.2172/15000841`.

Contents of the attached medium

```
README.txt ..... a brief description of medium contents
data ..... included datasets
src
    impl ..... source codes of the implementation
    thesis ..... source of the thesis in LaTeX format
text ..... text of the thesis
    thesis.pdf ..... text of the thesis in PDF format
```