FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Assignment of bachelor's thesis

Title:	Scoring real estates based on customer personal preferences
Student:	Roman Zhuravskyi
Supervisor:	Ing. Jaroslav Kuchař, Ph.D.
Study program:	Informatics
Branch / specialization:	Knowledge Engineering
Department:	Department of Applied Mathematics
Validity:	until the end of summer semester 2022/2023

Instructions

The aim of this thesis is to get acquainted with residential real estate market, to design and evaluate a tool based on available data and interfaces (information about property, neighborhood, lifestyle, etc.) in order to compare them with personal preferences of renters and buyers.

- Familiarize yourself with the issue of residential real estate evaluation, study real estate portals in order to identify descriptive characteristics, conduct research on factors influencing the decision of buyers.
- Explore appropriate sources of information in the form of open data or interfaces to existing applications.
- Build a dataset including at least data related to property parameters and location.
- Propose several variants of algorithms for property scoring based on available characteristics. The proposed solution should take into account different personal preferences.
- Perform an evaluation of the quality of the proposed approaches.
- Make the resulting tool available as open source.

Electronically approved by Ing. Karel Klouda, Ph.D. on 8 December 2021 in Prague.

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Bachelor's thesis

Scoring real estates based on customer personal preferences

Roman Zhuravskyi

Department of Applied Mathematics
Supervisor: Ing. Jaroslav Kuchař, Ph.D.

May 11, 2022

Acknowledgements

First of all, I want to thank my supervisor Ing. Jaroslav Kuchař, Ph.D. I want to say thanks for his excellent pieces of advice and for the time spent with me. He showed me the right way to write a thesis and also showed me the right way to test a hypothesis. This knowledge will help me in the future to approach the problem-solving process in the right way.

I would also like to thank my family, who supported me during my education. Thanks to my mother and father, who guided me in the right direction during particularly difficult times. Without your help and support, I would never have the possibility to graduate from a European university.

Also, I want to thank my beloved partner Anhelina for her support and smile. And to my best friend Nilka Danilka for being the best roommate during my studies.

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 11, 2022 . . . . . .

Czech Technical University in Prague
Faculty of Information Technology

Citation of this thesis

Zhuravskyi, Roman. *Scoring real estates based on customer personal preferences*. Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2022.

Abstrakt

V této práci jsme se seznámili s problematikou hodnocení rezidenčních nemovitostí, studovali jsme realitní portály s cílem identifikovat popisné charakteristiky, prováděli jsme výzkum faktorů ovlivňujících rozhodování kupujících. Také byl zkoumán vhodný zdroj informací v podobě otevřených dat nebo rozhraní k existujícím aplikacím a byl sestaven datový soubor zahrnující alespoň údaje týkající se parametrů nemovitostí a lokality. Byla provedena rešerše nejnovějších článků o různých doporučovacích enginech a poté bylo navrženo několik variant algoritmů pro skórování nemovitostí na základě dostupných charakteristik pro český realitní trh. Byly vytvořeny návrhy a testovány dva základní typy doporučovacích motorů: kolaborativní filtrování a filtrování založené na obsahu. V závěru práce byl učiněn závěr a také byly navrženy kroky pro budoucí vývoj.

Klíčová slova doporučovací engine, body zájmu, analýza souboru dat o nemovitostech, kolaborativní filtrování, filtrování na základě obsahu

Abstract

In this thesis, we are going to get into the depths of the issue of residential real estate evaluation, study real estate portals to identify descriptive characteristics, and research factors influencing the decision of buyers. Also will explore appropriate sources of information in the form of open data or interfaces of existing applications and will build a dataset including data related to property parameters and location. Research of the state-of-art articles about different recommendation engines was concluded, which resulted in the proposition of several variants of algorithms for property scoring based on available characteristics of the Czech real estate market. There were created drafts and tested two basic types of recommendation engines: collaborative filtering and content-based filtering. In the conclusion of this thesis, several recommendations are made and also proposed steps for future development.

Keywords recommendation engine, point of interest, real estate dataset analysis, collaborative filtering, content-Based filtering

Contents

List of Figures

Chapter 1

Introduction

This thesis is divided into four main parts. **The first part, Real estate market definition**, will aim to familiarise you with the problems of the Czech real estate market. To achieve this objective, a comparison of the Czech real estate market with the USA real estate market will be presented. We will also take a look at why recommendation systems are so necessary for the real estate market. **The second part, Data part**, will be about data, we will describe the structure of datasets and data sources, after which will be provided data analysis and data enrichment. Also will be present graphical examples from data analysis for better understanding. **The third part, Recommendation engine**, will focus on understanding which type of recommendation system will be more suitable for the Czech real estate market. To answer this question, state-of-the-art real estate solutions will be explored. Also will be identified the main challenges that need to be resolved before a recommendation system can be set up. The research will be then followed up with the creation of two different draft systems and their comparison. And the last part - **the fours part, Conclusion**, will aim to define future ways of improvement.

The problem we tried to solve in our work: for most people, buying a property is one of the most important financial investments in their lives. That is why they want to make sure that they have made the best choice.
The motivation: to empower people with the tool, which will help to make one of the most important financial decisions of their lives.
The limitation: we would also like to point out that in this thesis we will be working with properties such as houses or flats, not lands.

First of all, we need to understand what real estate is and what exactly influences people's decisions in the context of buying real estate. Real estate is land and the structures on it, as well as its natural resources such as crops, minerals, and water; immovable property of this sort; an interest vested in this (also) an item of real property; buildings or houses in general [7].

For us this statement will mean that we have to look at this problem from the following sides:

1. From the side of analyzing the data about the property.
2. Understand how attractive a given location is from an infrastructure point of view for a given user.

Chapter 2

Real estate market definition

Before embarking on our project, firstly we need to define what the Czech real estate market is and why it would be interesting to work on this particular project. In this section, we will also review, analyze and compare the main representatives not only of the Czech real estate market but also the leaders of the USA real estate market. Before analyzing the real estate market we would like to define the key metric, which is used all over the world in the real estate field. During the research, such questions as how to compare the real estate market's income in previous years with the current one, the following metric was found, which is called the house price index (HPI). The house price index (HPI) measures the price changes of residential housing as a percentage change from some specific start date (the start day has HPI equal to 100) [8].

Also for analysis of Czech and USA real estate markets we define three main criteria:

1. **Location description.** The location is one of the most important component for customer while choosing of the property. So in this step we try to analyze how much information about the location was provided by the web page.

2. **Recommendation engine analysis.** This step we analyzed only from the graphical part. Because this thesis focuses on recommendation engines, we decided, that it will be interesting for us to look at the similar properties from different web pages.

3. **Cold Start problem.** One of the main challenges, which needs to be solved is the Cold Start problem. That's why we decided to research in which way the problem is solved into different web pages.

2.1 Czech real estate market

To begin with, I would like to point out the growth dynamics of the Czech market. The house price growth in the last quarter of 2021 is consecutively increasing and beating previous records. Land prices have risen by almost 6% in just three months, with year-on-year growth exceeding 23% [9], [10]. This is the largest increase in the history of the HB index [11]. This tells us that the interest in buying real estate has only increased and, as the result, the amount of different information that people have to process has increased as well, which makes navigation in the real estate market complicated. After analyzing the housing index [1] the following conclusion was made. From 2008 to 2021, the Czech Housing Index averages 113.27 points, reaching a record high of 189.60 points in the third quarter of 2021 and a record low of 93.60 points in the first quarter of 2013. For us, this means that interest in the property market is only increasing. This conclusion represents the following graph 2.1.

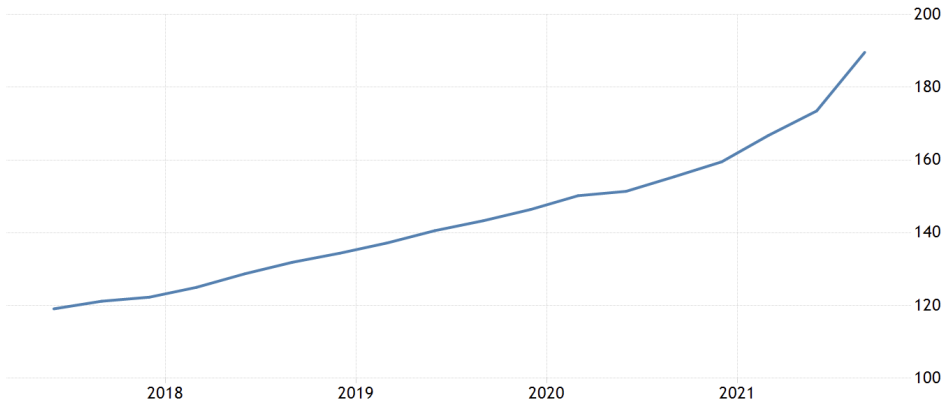

Figure 2.1: Housing index in Czech Republic for the last 5 years. Taken from [1]

The main portals for buying or renting property in the Czech Republic are: sreality.cz [12], bezrealitky.cz [13] and mmreality.cz [14].

After analyzing the above portals, we can make the following conclusions:

1. The detailed information about a property does not reveal the advantages of the location in which it is located. However, it is good for the user to understand how attractive a given location is compared to another location. By attractiveness, we mean infrastructure, parks, cafes, gyms, etc.

2. The recommendation system currently lacks a lot of useful features. On the website sreality.cz [12], for instance, during the research, a function

that would analyze the flats you selected and show what you would like to see was not available. The only substitute for the recommendation system was the use of filters, which the user would set before searching. But it would be better if the site would select the property not only based on the user's property information, but also by analyzing the interior/exterior and the location preferences of the user. In other words, the system should try to find as much information about the user as possible to present the most suitable property.

3. We would also like to notice that the portals mentioned above do not solve the Cold Start problem [15]. In other words, the portals do not know which properties to show to a new user. That is why on the main page they immediately ask to select filters and only then do they show the properties based on those filters, which can be interpreted as a variant of a Knowledge-Based recommendation system.

2.2 USA real estate market

After analyzing the housing index [2], the following conclusion was made - the index in the United States averaged 184.15 points from 1991 until 2022, reaching an all-time high of 373.35 points in January of 2022 and a record low of 100 points in January of 1991. Also worth mentioning that the housing index in the United States is expected to be 365.00 points by the end of this quarter, according to Trading Economics' global macro models and analysts' expectations. In the long-term, the United States FHFA House Price Index is projected to trend around 380.00 points in 2023, according to our econometric models [16]. For us, this means that interest in the property market is only increasing. This conclusion represents the following graph 2.2.

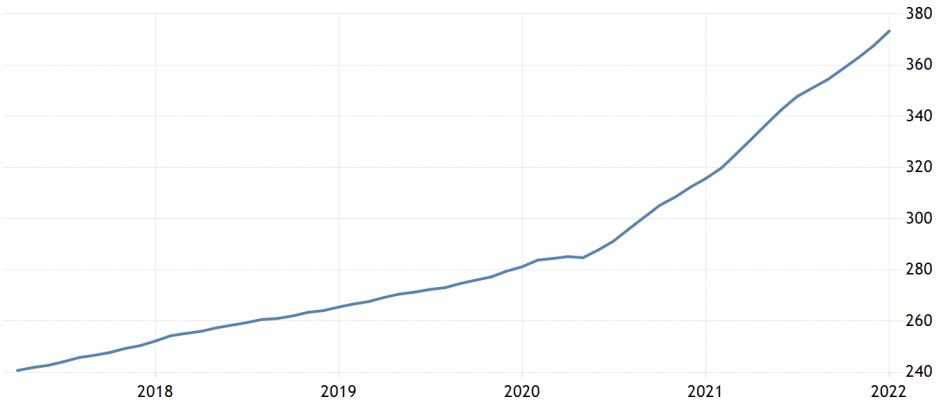

Figure 2.2: Housing index in USA for the last 5 years. Taken from [2]

The main portals for buying or renting property in the USA are gobii.com [17], zillow.com [18], trulia.com [19]. After analyzing the above portals, we can make the following conclusions:

1. On trulia.com [19], for example, the details of the property show the potential of the location in which it is located. The detailed information about property is divided into categories such as schools (elementary, middle, high), shop and eat (restaurants, groceries, nightlife, cafes, shopping, art and entertainment, fitness). The school category also contains a score for each school to give you an idea of how good the school is [20].

2. If we will take the website zillow.com [18] and analyze it for the presence of a recommendation system, we will find, that the page with detailed information on properties also shows offers with similar parameters, appearance, and price. In addition, the user is also offered flats that are in the same location.

3. If we will turn to the Cold Start problem, the three sites mentioned above solve this problem in some way. For the user to see the first real estate offers, they only need to specify the city that they prefer.

2.3 Comparison between Czech and USA real estate market

To compare the Czech and USA markets, we suggest looking at chart 2.3 which provides the comparison between the USA and the Czech Republic, from which we can conclude that both markets are very profitable. Also, we can see that as time goes on, the interest in the Czech and USA real estate markets only increases.

To conclude the comparison between the Czech and USA real estate markets, the technologies, we mean the sites analyzed above, from our point of view, used in the Czech market are clearly outdated compared to those used in the USA market. Improving the above-mentioned services or creating a completely new one, taking into account the previously listed remarks, could bring more value to the society than it had before. Also, we would like to mention, that at the time of reading the thesis, the situation may change in the future and new competitive products will already be on the market.

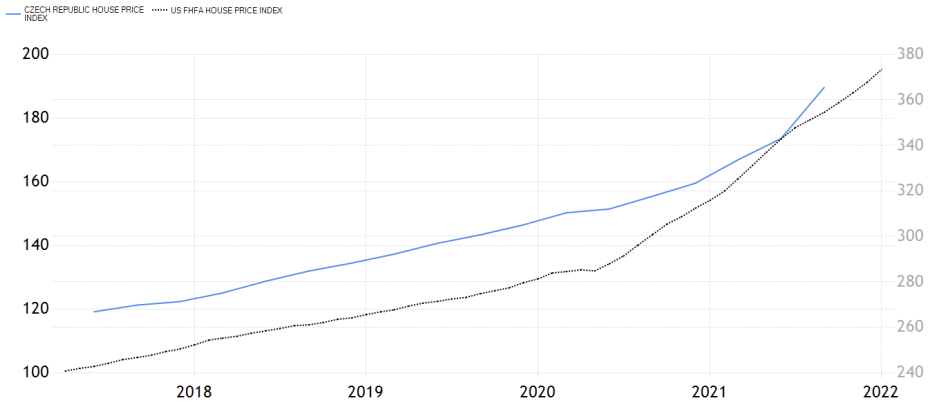

Figure 2.3: Comparison between housing index in USA and Czech Republic for the last 5 years. The housing index of Czech Republic represents the blue line and the housing index of USA represents the dotted line. To take this kind of graph you should click on Compare button. Taken from [1]

Chapter 3

Data part

3.1 Dataset info

For our project, we decided to choose three datasets:

1. The first dataset - describes properties
2. The second dataset - describes user preferences
3. The third dataset - points of interest information

Now let's take a closer look at the purpose and components of each dataset.

3.1.1 The first dataset - describes properties

Source: The data was taken from the API [21] from the real estate platform, which is called MM reality [14]. This API is not publicly available, so to gain access you will need to obtain access from the MM reality administrators.
Dataset description: The raw dataset consists of 93 columns. Also this dataset consists 6468 different apartments, lands and houses. The whole data dataset can be found in the attachment.

3.1.2 The second dataset - describes user preferences

Source: the data was obtained through a personal meeting with potential users. They were shown different types of properties and were asked whether they liked the above-mentioned properties or not.
Dataset description: after personal meetings with different people we got 12 responses 3.1. Each user was shown 10 random properties, it was presented in the table with the data and visualizations. Each of their users received the same set of properties. Each user should set up the rating between 0 (meaning that the user doesn't like it) and 10. The number will represent how much the user likes a certain flat. The row present the list of answers by one user and

each columns represents an apartment. We can see that to users were given different apartments 3.2. Because of the big amount of columns, only several of them will be presented, the main ones in our opinion. The whole list of columns can be found in the attachment.

	id	1	2	3	4	5	6	7	8	9	10
0	1	4	6	7	0	7	1	8	4	5	2
1	2	0	4	6	8	4	4	3	9	1	7
2	3	4	5	4	6	4	2	7	0	9	9
3	4	10	1	9	4	3	6	9	5	2	0
4	5	4	7	7	10	3	4	4	5	5	1
5	6	2	10	5	10	10	2	5	0	6	7
6	7	8	8	2	2	9	4	1	8	9	9
7	8	6	5	6	6	9	7	6	0	3	4
8	9	3	6	4	8	1	4	7	10	1	5
9	10	0	2	7	1	8	1	3	3	8	5
10	11	7	9	7	2	8	0	6	2	5	7
11	12	7	2	5	1	10	4	1	9	5	10

Figure 3.1: The dataset of received rating about the properties. The row is the user's rating and the column is a certain property id.

	propertySubType	listPrice	city	roomCount	coffee	pub	school_elementary	school_middle	school_high	fitness	restaurant
0	HOUSE_SINGLE_FAMILY_RESIDENCE	7900000.0	Mutějovice	6.0	0.5	2.6	7.9	0.0	0.0	2.6	8.5
1	HOUSE_SINGLE_FAMILY_RESIDENCE	5990000.0	Lukavice	8.0	2.3	2.3	2.1	0.6	0.0	2.3	1.9
2	HOUSE_SINGLE_FAMILY_RESIDENCE	nan	Vejprty	8.0	0.4	0.3	5.8	0.0	0.0	6.8	5.6
3	APPARTMENT_3_PLUS_1	2890000.0	Milevsko	4.0	6.6	2.7	4.9	0.0	0.0	7.5	6.5
5	COMMERCIAL_RESTAURANT	5995000.0	Bílovec	6.0	0.5	0.3	1.9	2.7	0.6	2.0	2.7
6	HOUSE_CHALET	nan	Andělská Hora	5.0	0.8	0.4	0.7	0.0	0.0	2.3	1.3
7	COMMERCIAL_RESTAURANT	8840000.0	Pernarec	7.0	0.5	0.4	6.8	0.9	0.0	1.8	0.4
8	COMMERCIAL_OFFICE	nan	Ostrava	1.0	5.3	4.4	5.6	6.0	7.3	5.3	4.9
9	HOUSE_SINGLE_FAMILY_RESIDENCE	3495000.0	Choustník	4.0	0.4	0.3	7.4	0.2	0.0	6.9	8.2
10	COMMERCIAL_PRODUCTION_FACILITY	4500000.0	Moravská Nová Ves	5.0	6.6	5.8	7.3	0.3	0.5	6.5	6.3
12	COMMERCIAL_HOTEL_MOTEL	nan	Mariánské Lázně	15.0	4.5	7.9	6.1	0.0	0.0	7.6	7.0

Figure 3.2: Some columns of rated properties

3.1.3 The third dataset - points of interest information

The Point of Interest (POI) is some specific location, which can be potentially interesting for somebody [22]. To understand how we can compare two locations with each other, we use a concept called Points of Interest (POI). This dataset will be used in chapter Internal enrichment. In the work, with a POI we will be able to understand the personal preferences of the user. Each location will contain the following points of interest: cafe, sport, parking, restaurant, and school.
Source: The data was obtained from the Open Street Map [23] then OSM more precisely, all information was obtained using Overpass turbo [3]. For example, to retrieve restaurants using Overpass turbo you have to perform the following query 3.3.

```
/*
This query looks for nodes, ways and relations
with the given key/value combination.
Choose your region and hit the Run button above!
*/
[out:json][timeout:25];
// gather results
(
  // query part for: “amenity=restaurant”
  node["amenity"="restaurant"]({{bbox}});
  way["amenity"="restaurant"]({{bbox}});
  relation["amenity"="restaurant"]({{bbox}});
);
// print results
out body;
>;
out skel qt;
```

Figure 3.3: Query to get restaurant from some location by Overpass Turbo API. Taken from [3]

After executing the request, we will receive a response in XML format from the server in the following form 3.4.

A script to retrieve data from Overpass turbo and process it and save it in a CSV file is provided in the attachment.

Dataset description: All information about the above-mentioned points of interest is stored in CSV files, one point of interest - one CSV file. Were have different CVS files, because of the way of running an algorithm. We decided to run an algorithm sequentially to check the quality of the saved data of each category. Each CSV file will contain such columns as geographical longitude, geographical latitude, establishment name, type, subtype, and additional information (e.g. opening hours, cuisine, telephone, etc.) if available. More detailed information about the dataset can be found in the attachment.

```
{
  "type": "node",
  "id": 252601050,
  "lat": 41.8957982,
  "lon": 12.4994754,
  "tags": {
    "addr:city": "Roma",
    "addr:housenumber": "33C",
    "addr:postcode": "00184",
    "addr:street": "Via di San Martino ai Monti",
    "amenity": "restaurant",
    "cuisine": "chinese",
    "diet:vegetarian": "yes",
    "name": "Sichuan Haozi",
    "opening_hours": "Mo-Su 11:00-15:00,17:30-23:00",
    "phone": "+39 06 4814425",
    "website": "https://www.facebook.com/Ristorante-Cinese-Sichuan-195910108024737/"
  }
},
```


Figure 3.4: The extract of the reply from Overpass Turbo API on the query from 3.3. Taken from [3]

3.2 Dataset enrichment

Once we have analyzed the textual data of the property, we need to analyze the images related to each property, which is described in this part External enrichment. Also, we will get detailed information about the environment/infrastructure where the property is located and you can find more information in part Internal enrichment.

3.2.1 Internal enrichment

As mentioned in the Introduction chapter, to understand which property is best suited to the user, we need not only to analyze their preferences in terms of technical features and affordability but also to understand their preferences in terms of location. For this purpose, we ask the user what type of area the user prefers (quiet, modern, suburban, etc.) and we also allow expressing interest in such POIs, more information about POI can be found in chapter Data part. The user may choose a number from 0 (not important) to 10 (very important) to express a personal interest in such POIs as coffee, pub, school elementary, school middle, school high, fitness, and restaurant.

3.2.1.1 Algorithm for using of POI

To understand how to use the dataset with the obtained POI and how to characterize the location, we need to figure out how we can convert the count of points of interest with different distances from the property into a numerical value (we further call this numerical value an index).
Obtaining a numerical value for each property can be divided into three parts:

1. Subsetting an index for a single POI - Algorithm 1: Single POI index.

2. Subsetting an index for a group of POIs - Algorithm 2: Single cluster index

3. Obtaining a final index for a property based on POI groups - Algorithm 3: Multiple cluster index

The following is a detailed breakdown of each part.
Basic information: The index is calculated based on a pre-existing category. The database contains POIs. Each of these records contains detailed information about the POI and also GPS coordinates of it, which are important for the calculation of Euclidean distance 3.5 from the user.

$$r_2(\mathrm{x,y}) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2} = \sqrt{\sum_{i=1}^{2}(x_i - y_i)^2} \quad (1)$$

Figure 3.5: The Euclidean distance formula for two item x and y

We would like to mention, that the Euclidean distance is the "perfect" distance and in the Real World we won't use this kind of distance. Another kind of metric was not used in this thesis, but this can be one of the open challenges, which can be solved in future improvements.

Example for the case of a cafe index:

- Index is close to 0 - few cafes in the area
- Index is close to 100 - many cafes in the area

The index is calculated based on three clusters, with each cluster weighted differently. Three clusters allow us to understand, what is the location of the property, is it downtown or not? The closer the cluster is, the higher its weight, and the more interesting the activity for a person (e.g. a cafe at a distance of 1 km is better than a cafe at a distance of 4 km):

- first_cluster - the closest, radius of 2 km. Weight of the cluster - 0.5
- second_cluster- middle, radius of 5 km. Weight of the cluster - 0.4
- third_cluster- farthest, radius 10 km. Weight of the cluster - 0.1

3.2.1.2 Algorithm 1: Single POI index

Input: gps_lat and gps_lng. In our case, these are the GPS coordinates of the properties which we want to index, for example the cafe.
Output: Number 0 to 100 given number of and proximity of nearby locations.

Steps:

1. Local distance = calculate the euclidean distance between the property and the point of interest.

2. Single POI index = 1 - (local distance * 0.8 / 1.99). 0.8 - weight for maximum possible distance and 1.99 - possible radius in the first_cluster(possible radius will be changing according to cluster).

To show, why we chose 0.8 as the optimum variable for the weight for maximum possible distance and also to show, why we choose 1.99 km as the optimum maximum possible distance, we will conduct testing of it in some examples.

Let's take the apartment in the center of the Prague (on Václavské náměstí) with the following coordinates: 50.081768, 14.426593. After running the algorithm we got the following result for the weights parameter, which is represented at 3.6 and for the maximum possible distance parameter at 3.7.

Testing of the weight for the distance for Václavské náměstí							
Weights	**Coffee**	**Pub**	**School elementary**	**School middle**	**School high**	**Fitness**	**Restaurant**
0.1	27.9	20.4	84.8	92.7	93.5	55.4	20.9
0.2	21.5	13.1	77.2	85.5	87.0	47.3	14.0
0.3	15.0	15.8	79.7	78.3	80.6	49.3	17.2
0.4	48.6	48.5	72.1	71.1	74.1	81.2	50.3
0.5	42.2	91.2	64.6	63.9	67.7	73.1	43.5
0.6	85.9	83.9	57.0	56.9	61.2	65.0	86.6
0.7	79.9	76.8	49.6	49.8	54.7	57.0	79.9
0.8	74.1	70.0	42.6	42.8	48.3	49.3	73.4
0.9	68.6	63.5	36.7	37.5	41.8	41.9	67.2

Figure 3.6: Testing of the weight for the distance in the center of Prague. The green line is the line which was selected for the current version of algorithm.

Also let's test with another location. Our chosen location is another district, the outskirts of Prague. The name of district is Kaceřov, the name of street is Na Brázdě and the gps coordinates are 50.045940, 14.462791. After running of the algorithm we got the following result for the weights parameter,

Testing of the maximum possible distance for Václavské náměstí

Maximum distance	Coffee	Pub	School elementary	School middle	School high	Fitness	Restaurant
0.1	29.4	36.7	28.1	56.9	58.3	61.6	26.8
0.5	40.7	56.4	33.6	83.4	60.7	38.1	45.5
0.99	43.9	41.9	39.2	42.8	32.5	35.6	42.4
1.99	74.1	70.0	42.6	42.8	48.3	49.3	73.4
2.99	40.4	88.9	62.3	61.6	65.6	70.6	91.3
3.99	49.1	48.7	72.4	71.2	74.2	81.5	50.5
4.99	14.3	14.5	78.5	77.0	79.3	88.0	16.0

Figure 3.7: Testing of the maximum possible distance in the center of Prague. The green line is the line which was selected for the current version of algorithm.

which is represented at 3.8 and for the maximum possible distance parameter at 3.9.

During analysis of the indexes from 3.6, 3.8, 3.7 and 3.9, we can say that our chosen parameters help to describe the location in the more suitable way. Logically we can expect that the number of places such as restaurants, coffee places, pubs, etc, will be bigger at the center than on the outskirts. The received indexes describe our logical statement. But at the same time, we are not saying that it's the only correct weight. In the future, this weight of index should be set up by some model, not by hand.

3.2.1.3 Algorithm 2: Single cluster index

As mentioned before, we use three clusters with different radii to estimate the location. Once we have calculated the Single POI index for each point, we can calculate the Single cluster index for one of the clusters (first_cluster, second_cluster, or third_cluster).

Input: All single POI indexes inside of some cluster. For example, if we want to calculate the single cluster index for the first_cluster, the input will be all single POI indexes for the activities inside the first_cluster.
Output: Number 0 to 100 given number of and proximity of nearby locations.
Steps: Take all indexes of POI inside the certain cluster and calculate the arithmetic average.

Testing of the weight for the distance for Kaceřov							
Weights	**Coffee**	**Pub**	**School elementary**	**School middle**	**School high**	**Fitness**	**Restaurant**
0.1	55.9	55.0	85.0	91.9	91.2	55.8	55.9
0.2	47.9	46.0	77.3	83.8	82.4	47.7	47.8
0.3	79.8	77.0	79.6	75.7	73.6	49.6	79.8
0.4	81.8	78.0	71.9	67.6	64.8	81.5	81.7
0.5	73.8	69.0	64.1	59.5	56.1	73.4	73.6
0.6	65.7	60.0	56.4	51.4	47.3	65.3	65.6
0.7	57.7	51.1	48.8	43.3	38.5	57.3	57.6
0.8	49.8	42.4	41.5	35.2	29.7	49.5	49.7
0.9	41.9	33.9	35.7	28.9	23.2	42.0	41.9

Figure 3.8: Testing of the weight for the distance in the outskirts of Prague. The green line is the line which was selected for the current version of algorithm.

Testing of the maximum possible distance for Kaceřov

Maximum distance	**Coffee**	**Pub**	**School elementary**	**School middle**	**School high**	**Fitness**	**Restaurant**
0.1	55.1	26.7	16.8	28.5	27.4	46.3	49.4
0.5	42.9	73.5	47.8	17.7	39.4	36.7	46.5
0.99	33.5	40.7	46.6	33.5	41.7	36.0	37.6
1.99	49.8	42.4	41.5	35.2	29.7	49.5	49.7
2.99	71.2	66.1	61.6	56.9	53.2	70.8	71.0
3.99	81.9	78.1	71.9	67.7	64.9	81.6	81.8
4.99	78.3	85.3	78.1	74.1	71.9	88.1	78.2

Figure 3.9: Testing of the maximum possible distance in the outskirts of Prague. The green line is the line which was selected for the current version of algorithm.

3.2.1.4 Algorithm 3: Multiple cluster index

Input: Three Single cluster index for the first_cluster, second_cluster and third_cluster.
Output: The final index for property, the number from 0 to 100.
Steps:

1. Repeat Algorithm 2: Single cluster index for three times to take single cluster index for first_cluster, second_cluster and third_cluster.
2. The final index for property = first_cluster_index * weight_first_cluster + second_cluster_index * weight_second_cluster + third_cluster_index * weight_third_cluster

3.2.2 External enrichment

3.2.2.1 Property condition

Images are scored on a scale of 1.0 - 6.0 (disrepair, poor, average, good, excellent, and luxury) based on each property's condition and quality. The overall property-level scores are calculated by weighting each individual image's relevance to the home's value [24]. To get more information how the property condition working inside restb.ai [25], we propose the check out demo [6].
Steps: we will use restb.ai trained model to get property condition for each image in each property and will take the following types: interior, exterior, bathroom, kitchen, and overall condition. These different types were taken according to restb.ai documentation [24].
Result: after analyzing of images and getting tags about the conditions of properties, we will connect information from the registration form or additional form about the user preference with these property condition information.
Testing: in Appendix it's possible to see the test result from restb.ai [25] functionality on two different properties: on the good one B.1 and on the poor one B.2. In the picture you can see, that returns not only the property condition but also returns the score from 1.0 to 6.0
Note: a subscription is required to use this solution, detailed pricing information can be found on the following page [26]

3.3 Dataset analysis

In this chapter, we will analyze our datasets so that we can understand the benefits that can be derived from them. We currently have three different datasets :

1. Describes properties
2. Points of interest information

3.3.1 Properties dataset analysis

The source code of the whole analysis can be found in the attachment. We can divide the whole analysis of this dataset into three logical parts:

1. Analysis of the dataset before enrichment.
2. Analysis of the dataset after enrichment.
    a) Enrichment of the location in which the property is located by means of indices.
    b) Gain more detailed information about the interior/exterior by analyzing images.

3.3.1.1 Analysis of the dataset before enrichment

We will need to clean the data before we can start analyzing it. The list of all available columns is available in Appendix B. All data cleaning and preprocessing for this dataset we divide into 5 steps:

1. Clean the NaN values. The following columns in dataset were deleted because of the low count of not NaN values: *closePrice, listPriceLow, originalListPrice, previousListPrice, placement, characterOfVillage, countyOrParish, postalCity, stateOrProvince, township, unparsedAddress, developmentStatus, areaTotal, areaFloor, areaCellar, areaBuilding, numberOfBuildings, numberOfUnitsTotal, view, toilet, coreOfApartment.* If we analyze the above-mentioned columns, we will see that most of the columns that were removed from the dataset describe street names, city names, etc. This is a small loss for us, because latitude and longitude columns do not contain empty values. And in case we need some additional information, such as street name, city name, etc, we can use the Python library Nominatim [27] to take that kind of information.
2. As we said before, we will be working only with primary residences. That means that all rows which do not have property_type = "HOUSE" or "APPARTMENT" should be deleted. After cleaning of dataset from 6468 rows we got 4355 rows.

After cleaning the listing data, we decided to show all properties in a chart. For creating the picture 3.10 were taken GPS coordinates from the dataset.

3.3.1.2 Analysis of the dataset after enrichment

As described earlier, we use two ways to enrich the dataset as a part of this research work: location enrichment using our algorithm and image analysis and interior/exterior information collection.

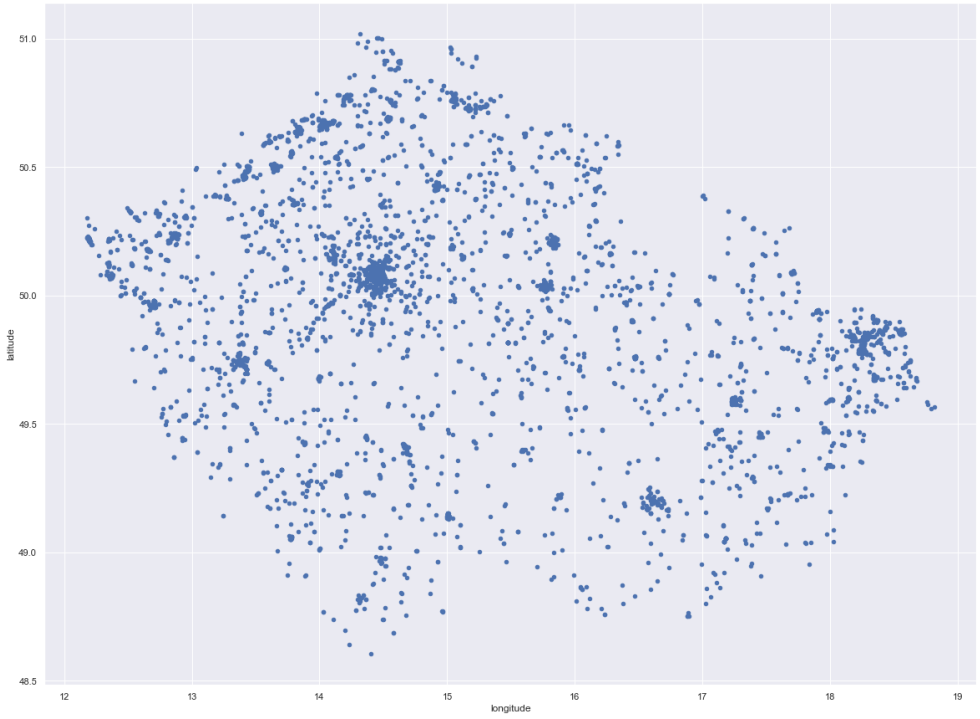

Figure 3.10: Here we can see, that points created the shape of the Czech Republic. And also we can see, that the largest number of listings are in Prague, Brno and Ostrava.

The algorithm that was used to obtain a location index from 0 to 10 based on the defined POI and GPS coordinates can be found in the attachment. A description of the principle of this algorithm was provided in chapter Internal enrichment.

At the moment we have 7 different indices which have been added based on the dataset with the POIs. These are the following index names that describe the location: coffee, pub, school_elementary, school_middle, school_high, fitness, and restaurant. Each index column has been normalized and converted to a value between 0 and 10.

Two different types of plots were constructed to analyze the dataset after location enrichment:

1. The first type shows the distribution of the index value over the entire Prague area.

2. The second type shows the most popular index as well as the percentage of all index values.

We will analyze the restaurant index and school_middle index as an example.

Restaurant index analysis

By analyzing 3.11 we can see that the main city with the largest number of restaurants is Prague. We can also see that the map is dominated by higher index values.

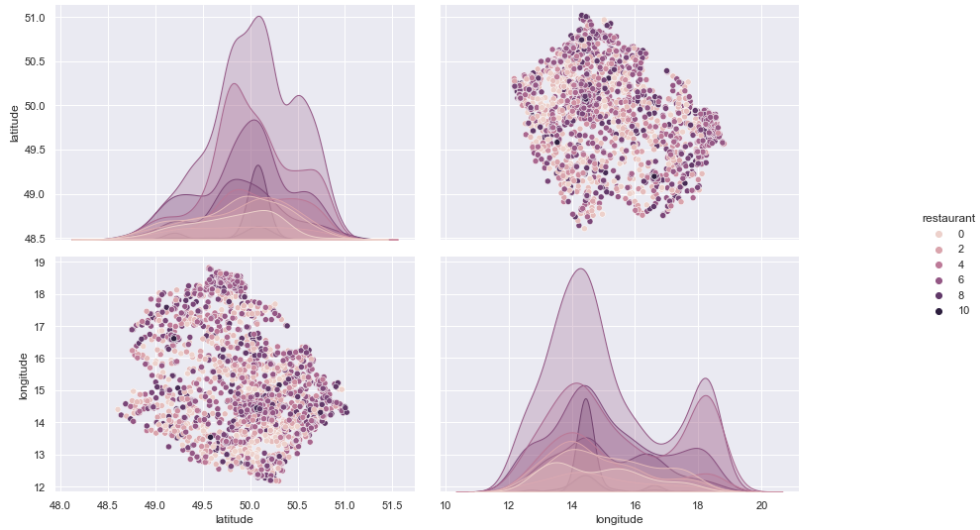

Figure 3.11: Displays the distribution of values for the restaurant index

By analyzing 3.12, we see that the most popular restaurant index is 6 (29.6 per cent is equal to 1,287 listings), 5 (18.1 per cent is equal to 788 listings) and the third most popular restaurant index is 7 (14.6 per cent is equal to 636 listings). This tells us that the Czech Republic has a large number of restaurants, which are evenly distributed throughout the whole territory.

Also at the following charts you can see the distribution of restaurant index for our properties on the map of Prague 3.13 and on the map of Czech Republic 3.14. The blue point is the lowest index which is equal to 0 and the yellow one is the highest index which equals to 10. As we can see in the graphs the bigger is the city the higher is the index and also it works with the different city district. The closer the property is to the center of the city, the bigger its the index 3.13.

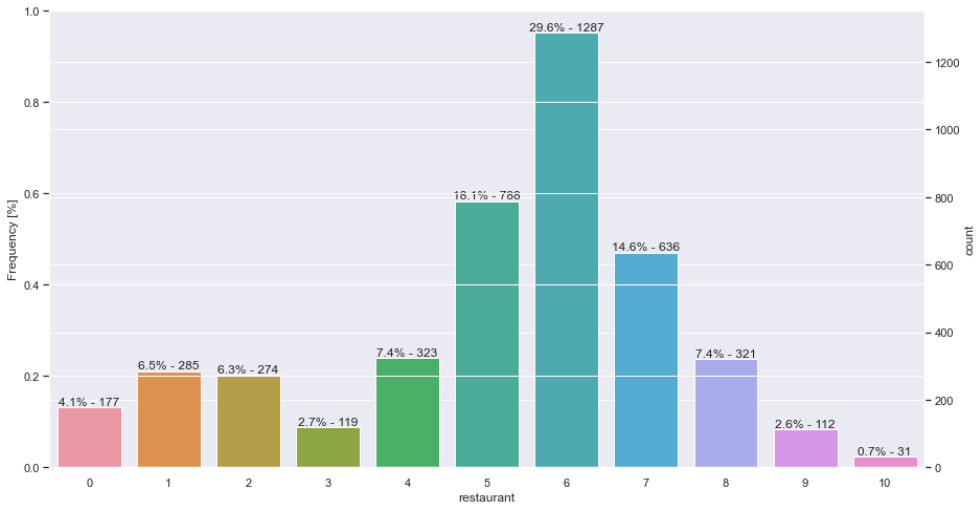

Figure 3.12: Displays the percentage split of the normalised indexes between listings.

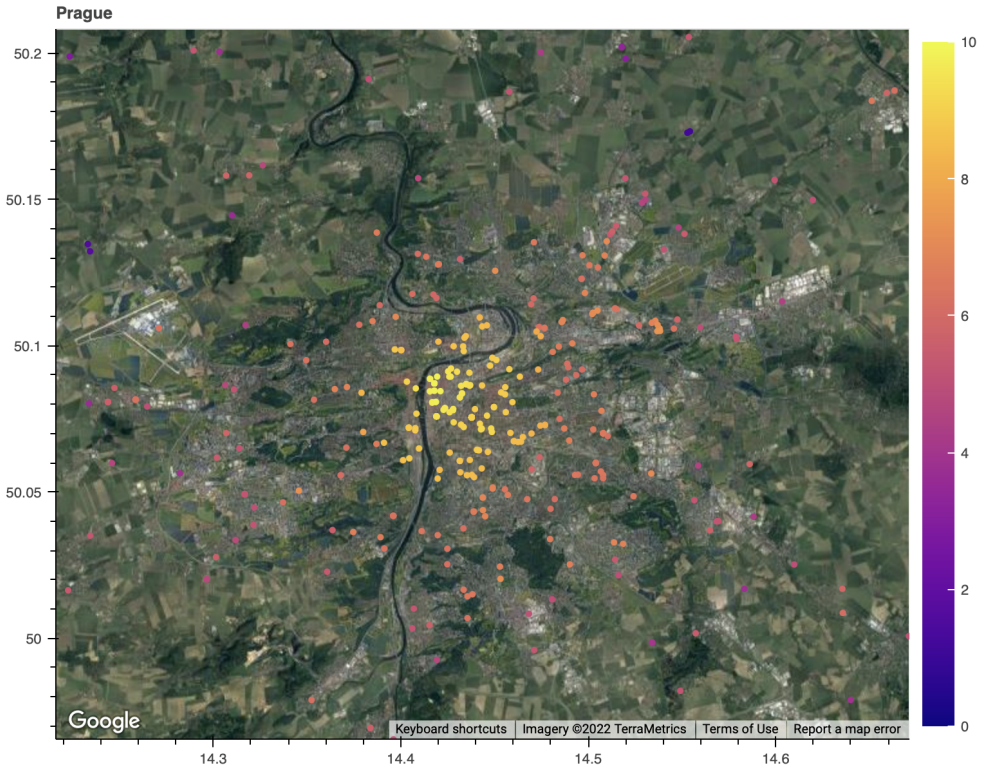

Figure 3.13: Distribution of restaurant index in Prague

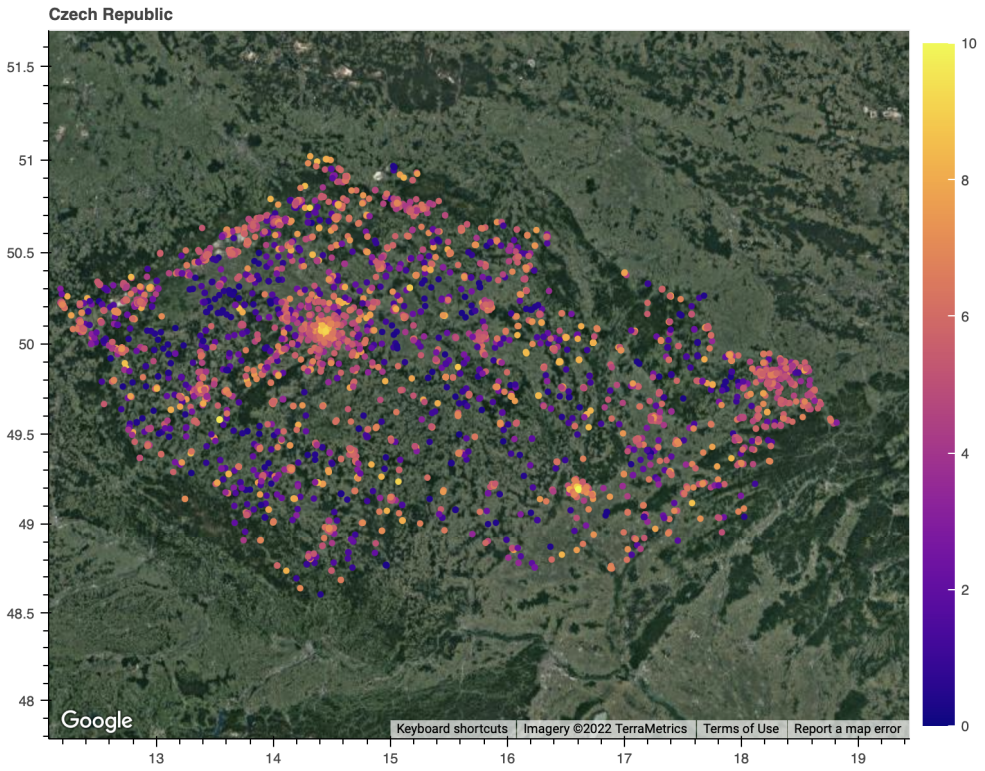

Figure 3.14: Distribution of restaurant index in Czech Republic

School_middle index analysis

By analyzing 3.15 we can see that the main city with the highest number of school middle is Prague. We can also see that the map is dominated by lower index values, such as 0 or 1.

By analyzing 3.16, we see that the main values of the school_middle index are 0 (42.6% is equal to 1854 listings) and 1 (20.0% is equal to 870 listings). This tells us that school middle schools are not evenly distributed throughout the Czech Republic and the main concentration of middle schools is in Prague.

Also at the followings charts you can see the distribution of school_middle index for our properties on the map of Prague 3.17 and on the map of Czech Republic 3.18. The blue point is the lowest index equals to 0 and the yellow one is the highest index equals to 10. As we can see on the graphs, the bigger is the city the higher is the index and also it works with the different city district. The close is the property to the center of the city, is bigger is the index 3.17. Also, at the same time, we should remember, that it can be caused by the low count of properties. On the map there are some places without points, that means that in this region we don't have the property. But the index we calculate now only for gps coordinates of properties. In the

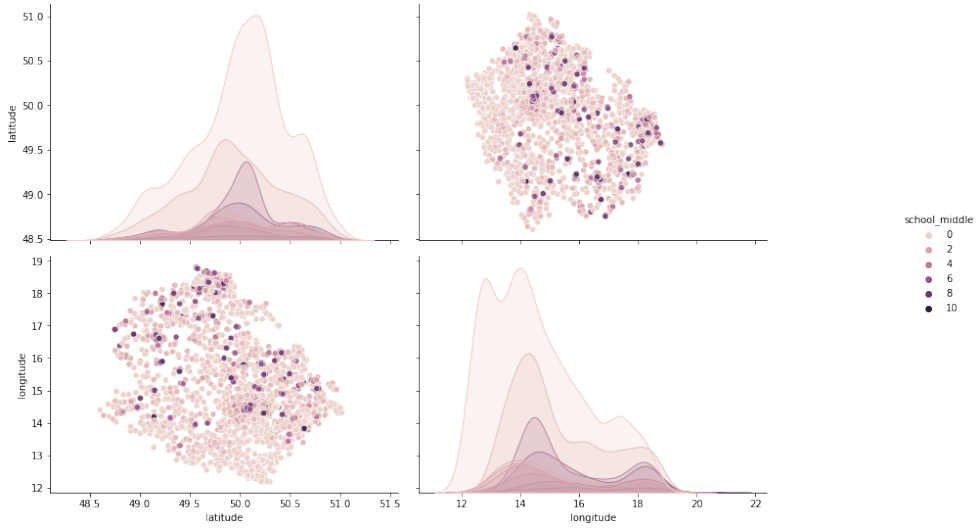

Figure 3.15: Displays the distribution of values for school_middle index

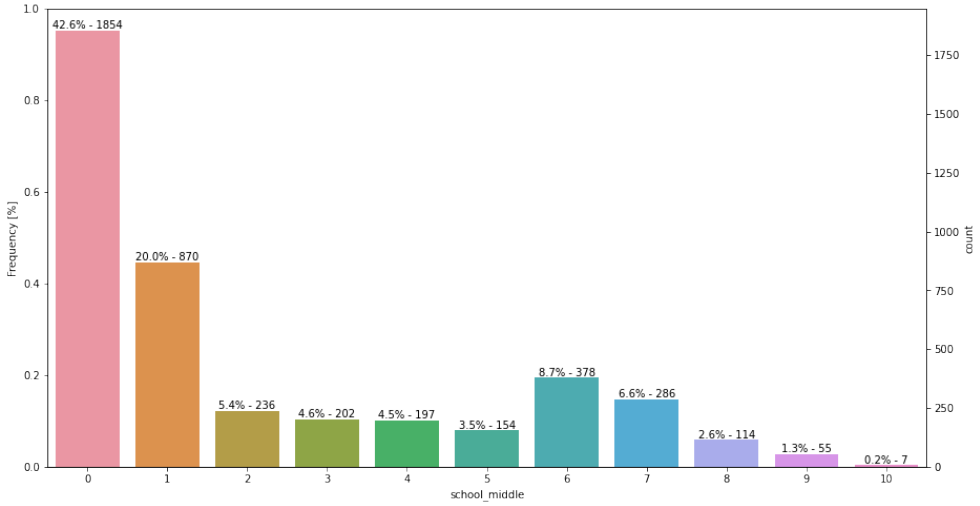

Figure 3.16: Displays the percentage split of normalised indices between listings

future implementation it can be changed in such a way that we will calculate the index, for example, each 100 meters (the step on the map should also be tested). So the index analysis won't be so dependent on properties' placement.

Graphs of all our indices can be found in the attachment. We have decided to point out and break down in detail only the example with the prevailing high value and the low value.s

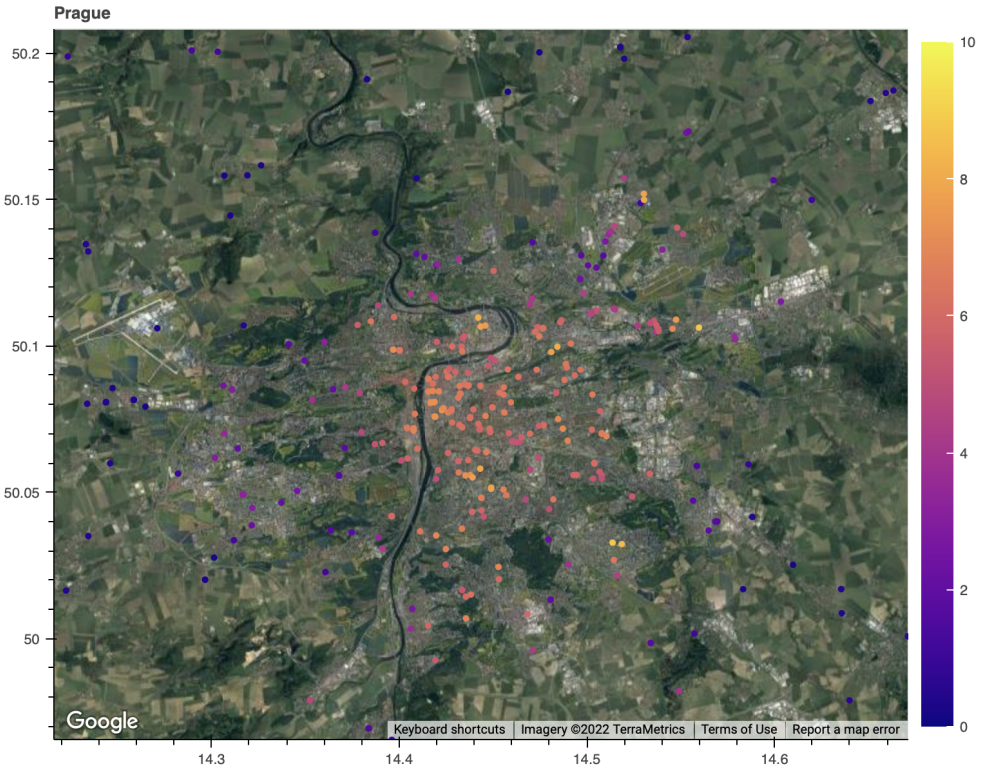

Figure 3.17: Distribution of school_middle index in Prague

3.3.2 Points of interest dataset analysis

To begin with, I suggest understanding the structure of the dataset and looking at the percentage distribution of the main categories and subcategories. This will show us based on which subcategories the above 7 indices were created.

The POI dataset consists of 70889 lines, each line representing some POI. By analyzing 3.19 for the percentage distribution of the main categories, we obtained the following percentage:

1. 56.8% (40298 of the different POI) are categorised as ”shop and eat”
2. 21.0% (14919 of the different POI) are categorised as ”livewell”
3. 17.2% (12163 of the different POI) are categorised as ”school”
4. 4.9% (3509 of the different POI) are categorised as “commute”. This category will not be analyzed further as it is not used further in this thesis.

By analyzing the sub-category *shop and eat* 3.20 we obtained this percentage of sub-categories:

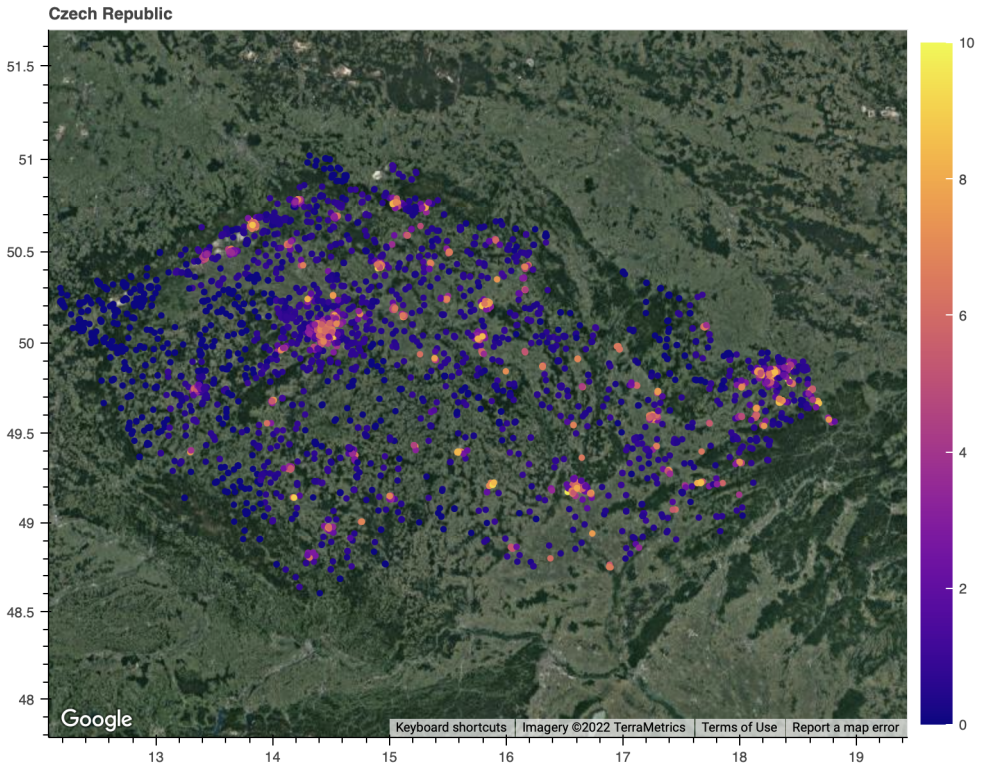

Figure 3.18: Distribution of school_middle index in Czech Republic

1. Fitness (48.6% - 19567 amenities) - inside of this category we have the following amenities: stadium, gold source, pitch, sports center, swimming pool, fitness station, track
2. Restaurant (27.7% - 11162 amenities).
3. Pub (12.4% - 4984 amenities) - inside of this category we have the following amenities: bar, pub.
4. Cafe (6.0% - 2406 amenities).
5. Grocery (2.7% - 1087 amenities) - inside of this category we have the following amenities: supermarket, grocery.
6. Art and entertainment (2.0% - 806 amenities) - inside of this category we have the following amenities: theatre, cinema.
7. Night life (0.6% - 241 amenities) - inside of this category we have the following amenities: nightclub.
8. Shopping (0.1% - 45 amenities) - inside of this category we have the following amenities: mall, general.

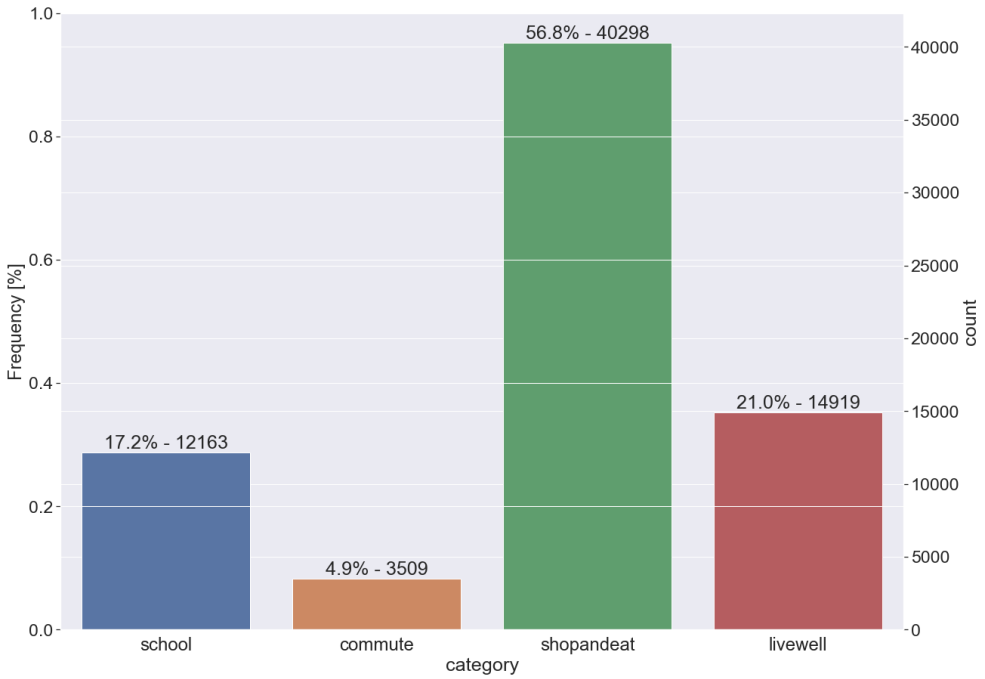

Figure 3.19: Percentage distribution of the main categories *school, commute, shop and eat* and *livewell*

By analyzing the sub-category *live well* 3.21 we obtained this percentage of subcategories:

1. Care and essential (57.8% - 8620 amenities) - inside of this category we have the following amenities:
   a) Daycare (68.7% - 5920 amenities) - here we have these kind of amenities: tanning salon, sauna, swimming pool, public bat
   b) Pharmacy (25.5% - 2197 amenities)
   c) Hospital (5.8% - 496 amenities)
   d) Pediatrician (0.1% - 7 amenities)
2. Great place to play (42.2% - 6299 amenities) - inside of this category we have only playgrounds

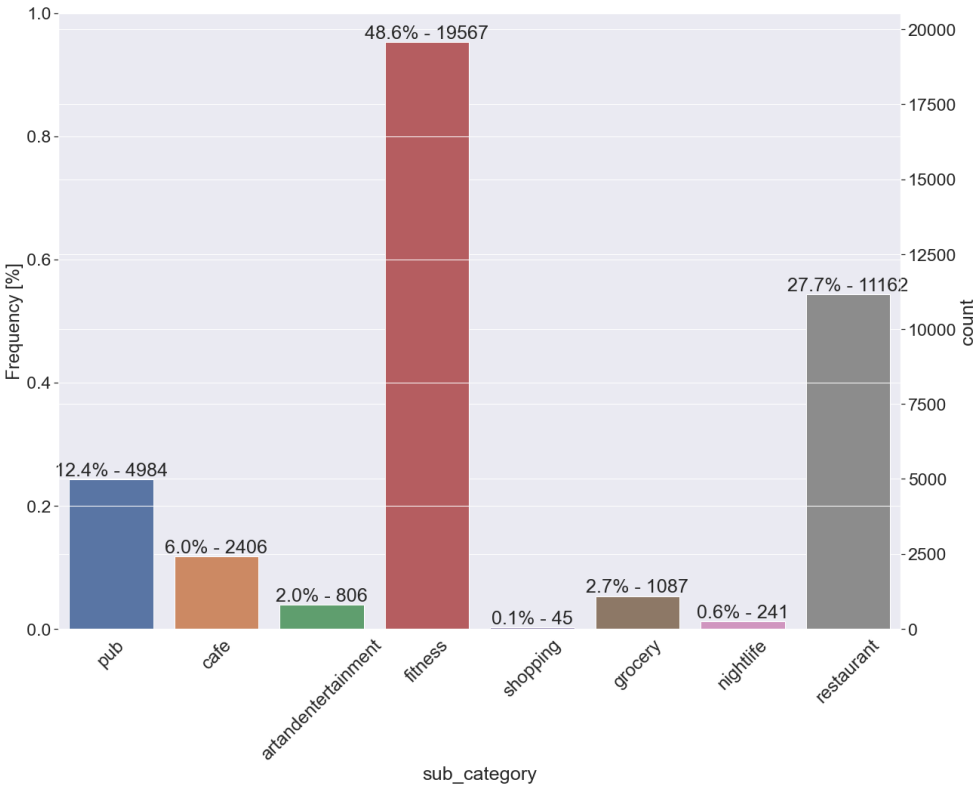

Figure 3.20: Percentage distribution of *shop and eat* subcategories

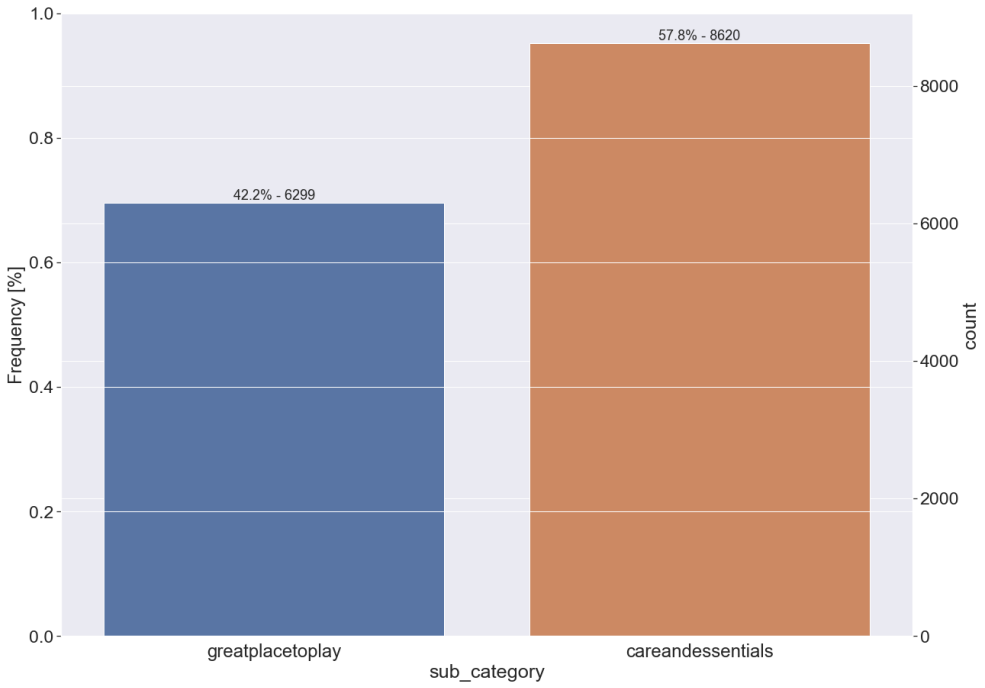

Figure 3.21: Percentage distribution of *live well* subcategories

Chapter 4

Recommendation engine

Many aspects of business are being transformed by the shift to e-commerce. This surge of modern technology has had an impact on the real estate industry. A recommendation system is an intelligence algorithm that assists real estate platform customers in finding the finest properties to meet their requirements. However, due to a number of sector-specific limitations that impede ordinary recommendation systems, the task of suggestion in the real estate market is substantially more complex. Real estate recommendation systems, for example frequently confronts the cold start problem, in which there is no prior experience of new customers or projects, so the recommendation system must make recommendations for these new entities. As a result, unlike other fields, real estate recommendation systems have not been thoroughly investigated. This chapter's goal is to give a thorough and systematic assessment of the literature on the application of recommendation systems in the real estate market. The majority of research and commercial solutions in the field of property recommendation systems were evaluated in a series of research publications. In this chapter we will review papers and will analyze information about their methodological approach, categorizations of recommendation engines and the primary difficulties and the future research directions [28].
The structure of this chapter is as follows:

1. We will define, what is the Recommendation engine.

2. We will overview The main types of recommendation systems.

3. We will study Issues connected with real estate recommendation engines.

4. Applying of the recommendation engine into our case.

Firstly, we will define what exactly the recommendation system is. It is the system (or engine), which is trying to predict the preference or the rating [29]. They can be used for solving of different tasks in different spheres, for

example, product recommendation for online shopping, recommendations on media platforms, online dating, financial services [30] and many more.

4.1 The main types of recommendation systems

Before diving into details of each type of recommendation engine, we will overview them from different perspectives and will define the main differences between them.
We will define three main categories of recommendation engines[28]:

1. Content-Based Filtering
2. Collaborative Filtering
3. Hybrid Recommender Systems

4.1.1 Content-Based Filtering

This type of recommendation system is based on the customers visits (some history) and it becomes more accurate the more inputs the customers makes. This category of recommendation systems has the following sub categories [31] [28]:

1. Content Similarity - this is the basic type of Content-Based Filtering. It is good when we have the rich dataset with historical data. This stately recommends the new content that is close on its metadata.
2. Latent Factor Modeling - this is the way, when is trying to predict the future by choosing is based on the historical one. This method is more complex, that the previous one (The Content Similarity), because the strategy is trying to discover implicit relationships.
3. Topic Modeling - it is used when you have rich and unstructured textual information (for example, it can be the news articles). It is the variation of the Latent Factor Modeling. Here is using the strategy, when we can make the conclusion by analyze of the instructed text to detect the customer's topic of interest, instead of considering customer's larger inputs.
4. Popular Content Promotion is used in case when the new content is the majority. It's useful to use it when you have defined features which are considered interesting for majority of people. It can be price, popularity, characteristics etc. Also the age and the freshness are important and can help show the most trended content.

Lops et al. [32] defined the trends in the available sources of data for Content-Based recommendation systems as follows: linked open data, user-generated content, multimedia features, and heterogeneous information.

4.1.2 Collaborative Filtering

It's the type of recommendation engine which makes preferences based on the other users preferences. The main idea is that the customer with the similar history has the same preferences. The main difference between the Collaborative Filtering and the Content-Based Filtering is that the Content-Based Filtering provides over-specified recommendations, but the Collaborative Filtering provides more surprising result (when trying to discover implicit relationships).
Example of using: If we have two similar users (the user A and the user B) and the user A watched the movie, then we can recommend this movie to the user B.
This category of recommendation systems has the following subcategories [28]:

1. Based on the model - we had found such model-based method: FunkSVD [31], BPR [33]. These methods are using two low-rank matrices for customer and items, and then representing them in the dense latent feature.

2. Based on the memory - here we are talking about the user's feedback, it can be: explicit (for example, rating) and implicit (for example, user's clicks).

4.1.3 Hybrid Recommender Systems

If we would like to combine Content-Based and Collaborative Filtering, we can apply the Hybrid Recommender Systems. For example, Batet et al. [34] proposed an agent-based Hybrid System, which was a combination of Content-Based and Collaborative Recommender Systems. It was used for personalized recommendations of tourist activities.
This category of recommendation systems has the following methods [35] [28]:

1. Weighted - we can combine Content - Based and Collaborative Filtering and each will take a weight of 50% at the final predication. The benefit in using of weighted Hybrid Recommender System is that we can join a lot of model to support the dataset on the recommendation process in a linear way.

2. Switching - this is the kind of system, which is based on the situation. It should choose which recommendation system should be used.

3. Mixed - firstly it takes the customer profile and features, and based on this information it generates the set of candidates datasets. Then using the set of candidates and models it combines results.

4. Feature Combination - we have the main model, but at the same time we are going to use virtual contributing recommendation mode, which will

work as a feature engineering. For example, we can inject Collaborative Filtering into the main Content-Based Recommendation System.

5. Feature Augmentation - by contributing recommendation model will generate a rating for the customer or item, then we will be able to enhance the customer profile dataset. With the extended dataset, the performance of the recommendation model will be improved.

6. Cascade - is using the hierarchical structure. The first recommendation system produce the primary result, then we use the second recommendation system for solving of several issues, such as line breaking tie in scoring. Also the second recommendation model can be useful for solving of the missing data issue or equal scoring issue.

7. Meta-Level - is the same as Feature Augmentation, the only difference is that Meta-Level replaces the original dataset with a learned model from the contributing model as the input to the main recommendation model.

4.2 Issues connected with real estate recommendation engines

In this chapter we will define the main challenges in housing recommendation tasks and we will introduce the papers, which are specified on this issues. Also we would like to add, that we found one of many possibilities which can be used to solve some certain problem, that means, that the proposed solutions are not the only right ones. The main challenges are the following:

1. Cold-Start problem
    a) Cold-Start problem for the new customer
    b) Cold-Start problem for the new property
2. Sparsity of the data
3. Feature specification

4.2.1 Cold-Start problem

The Cold-Start problem in Recommendation Systems is the problem, when the new customers or the new item appears on the platform. When visiting a website for the first time, the user has little to no interaction on the website, and because of that lack of information from the user the system does not know what to reccomend to him. And when the new item appears for which we didn't collect enough feedback to understand how interesting can the property be for different groups of people.

4.2.1.1 Cold-Start problem for the new customer

While research of this challenge, we found the paper from Rehman et al. [36]. The typical Collaborative Filtering is not applicable, because it's hard to create the long history for the user in real estate platforms. It was suggested to use Session-Based Recommendation Method, Knowledge-Based or Dialog-Based Recommender System, even with the poor history it will predict the next property, so it is considered inefficient to use User-Based Approach.

4.2.1.2 Cold-Start problem for the new property

We found the paper from Zhang et al. [37] it was recommended to use Content-Based to solve the Cold-Start problem. The main ideas was that users and properties were represented in the same feature space based on property metadata, that is why it's possible to make recommendations for new properties even if we don't have a lot of interactions with this objects.

4.2.2 Sparsity of the data

Here we tried to solve the matrix of interaction between properties and customers and is sparse. In this case Oh and Tan [38] said that the property search space is highly sparse. They solved the sparsity problem by providing initial knowledge to the agents, it will help to limit the searching space and, as the result, it will improve the final result of the recommendation engine. Also, we would like to mention that this challenge is possible to solve by embedding.

4.2.3 Feature specification

Here we will define the main features, which we can meet at the real estate portals. After analyzing of the real estate portals such as sreality.cz [12], mm-reality [14], zillow.com [18] and trulia.com [19], we can mention the following features: price, number of rooms/parking, living/land areas, address information, longitude, latitude, neighborhood description, furnished, type of the apartment, property condition. Also while researching this topic some additional information about the location was found. In [39], the authors said that the location is the most import feature for the user when trying to find the property, and in that case, the recommendation engine should be able to consider the information about the location and find out more suitable locations based on the user preferences. Also we would like to mention that Daly et al. [40] claimx that the distance between the property and the school, work, and other users' locations is also important in the decision-making process. These points of interest were defined here as POI.

4.3 Applying of the recommendation engine into our case.

After learning some state-of-art concepts we will try to define the steps which will be applicable for our suggestion of our application on our data within Czech real estate market. Also we will create the draft of the recommended system and we will make some conclusion.
Before building of each recommendation engine we need to solve three problems:

1. To collect, to analyze and to prepare the data. This task was described in the Data part chapter, so in this chapter we won't discuss this step one more time.
2. To choose which type of the recommendation system will be the most convenient for our case
3. To solve described challenges which are connected with recommendation engine.
4. To create the draft of recommendation system
5. To test the results obtained

4.3.1 Choosing of the recommendation system type

Firstly, we would like to mention, that as the real estate market is the hard case to apply recommendation system, because usually we will not get a lot of historical data from users. Also we propose to get as much data as possible during the user interaction with the platform. This can be such data as: gender, age, occupation, monthly income etc. After getting of this this kind of data, you can use them in such recommendation systems as: Knowledge Based or Interactive Iterative approach, but in the content of this thesis, we are not going to make any research or implementation of above systems. It's can be the possible field for the future improvement.

We would like to assume that in the real world the most suitable type of recommendation engine is the hybrid one. But we can't be sure about that 100%, because it needs to be tested on the bigger amount of data that we have now. Also it's possible, that for different regions, cities or countries, you will need to have different types of recommendation systems. It very much depends on what data you managed to get. As a part of this research work, we would like to test two basic recommender system types:

1. User-Based Collaborative Filtering
2. Content-Based Filtering

4.3.2 Solving of challenges

In this part, we will present the possible solution for problems which were mentioned above. We will propose the solving of the following problems:

1. Cold-Start problem for the new customer.
2. Cold-Start problem for the new property.
3. Sparsity of the data.
4. Feature specification.

Cold-Start problem for the new customer. For solving of this problem you need to take as much data from user as possible. The data you can get from registration process, from analyzing of the user flow, while he is using the platform or you can propose some benefits after the user will agree to answer on your questions. This can be such data as: gender, age, occupation, monthly income etc. After you will get this kind of data the Content-Based Filtering can be applied. We will try to find the similar user and to show properties which were shown to the similar user before.

Cold-Start problem for the new property. For solving this problem the same method can be used which was described above, while solving of the Cold-Start problem for the new customer. However the only difference is that here we will strive to find the similar properties, not users.

Sparsity of the data. If we will be talking about the user data collection, it's hard to solve this problem in real estate world, since users are not visiting real estate portal as often as some websites like Netflix or Youtube. But the sparse dataset with properties can be enriched in a way which is described at the chapter about Dataset enrichment 3.2. Also feature selection or embeddings can be used to solve the sparsity of data.

Feature specification. Here we would like to mention which property parameters are the most imporant for the user when he is looking for apartment. As were mentioned above, the most preferred information about property is: price, number of rooms/parking, living/land areas, address information, longitude, latitude, furnished, type of the apartment, property condition and neighborhood description. The way in which it is possible to get more information about the neighborhood was described in the chapter about Data internal enrichment 3.2.1. It will help describe and to compare the neighborhood in seven areas, such as: coffee, pub, school_elementary, school_middle, school_high, fitness, restaurant, be the grade from 0 to 10. Also analyzing of images of properties can be useful for understanding of the user preferences. Which data and in which way can be obtained the information from images

is described in chapter about the Data external enrichment 3.2.2.

Also we would like to mention that this is just one way in which the above problems can be solved. The way of dealing with problems may vary depending on what information you have been able to obtain.

4.3.3 Drafts of recommendation engine

We would like to mention, that in this chapter, we won't analyze each row of the code. However, we will concentrate more on basic concepts and understanding how good or bad the solution was and what can be improved in the future steps. The full code of the following recommendation engines is available in the attachment. In this chapter, we would like to create drafts of two basic recommendation systems:

1. User-Based Collaborative Filtering
2. Content-Based Filtering

4.3.3.1 User-Based Collaborative Filtering

The Basic concept of the method is the work the dataset of users gives the rating to properties. Firstly, let's describe how it is working. This part we can divide into the following steps:

1. Data part. The data, which will be using in this part is described in part 3.1.2. Because of the data part was analyzed above, we would not repeat steps.
2. Find the most similar apartment to the apartment we want to make prediction.
3. Calculate the weighted average of the user's ratings for the properties that are the most comparable.

Find the most similar apartment to the apartment we want to make prediction. There are a lot of different ways how we can solve this task. In this part we are using cosine similarity. For better understanding what cosine similarity is, we suggest you to look and to analyze the following figure 4.1. The User 0's ratings are on the x-axis, and the User 1's are on the y-axis. Then we may discover points in the space for each property. The property_0 corresponds to the point (2,3) in the space, for example. Cos(θ) is used to calculate the distance between two vectors in the cosine similarity. Cos(θ) declines with increasing (cos(θ) = 1 when = 0 and cos(θ) = 0 when = 90). As a result, the two vectors are regarded closer because the value of is smaller (the similarity gets greater). Movie_3 is the closest to Movie_1, and

Movie_2 is the farthest, because 1 is the smallest and 3 is the largest. The interesting thing is that similarity is looking between all users [4]. In the code we are using the *NearestNeighbors()* in the *sklearn.neighbors* to calculate the cosine similarity.

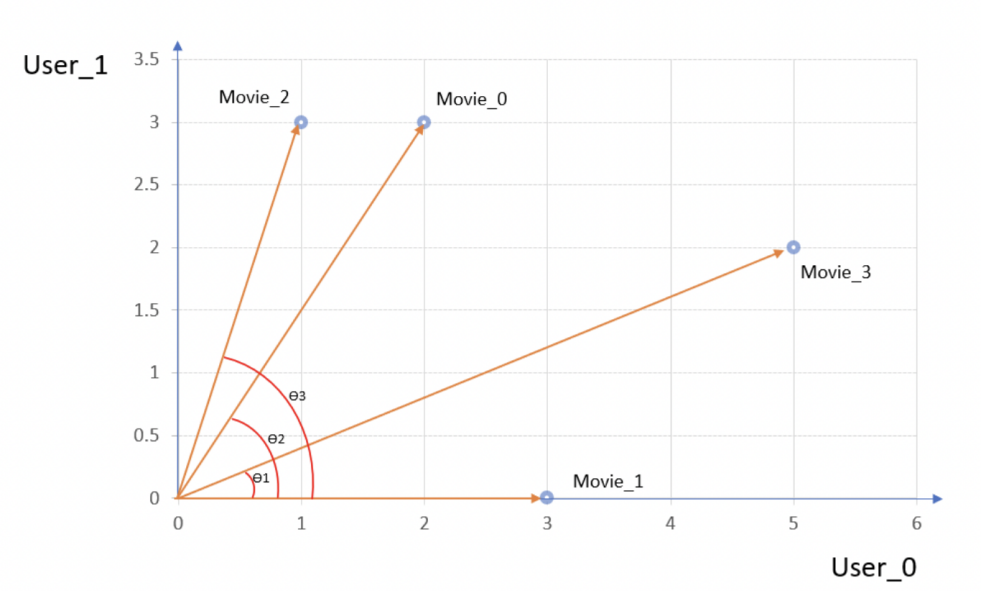

Figure 4.1: The graphical representation of the cosine similarity. In our case, points will be properties, not movies. Taken from [4]

For better understanding of cosine similarity, we suggest you to look at the figure which explains when two vectors are similar, orthogonal or opposite 4.2.

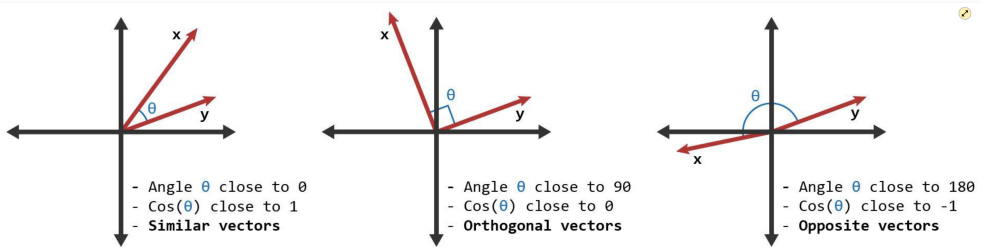

Figure 4.2: The figure is showing two vectors with similarities close to 1 (similar), close to 0 (orthogonal), and close to -1 (opposite). Taken from [5]

Calculate the weighted average of the user's ratings for the properties that are the most comparable. A user offers similar ratings to properties that are considered similar. All the following numbers are taken

from the dataset about User preferences 3.1. As a result, it is appropriate to consider the average of the user's ratings for similar properties when predicting a rating for a property. After that, we utilize *property_3* and *property_0* to forecast *user_1*'s rating of *property_0*. *user_1* has given *property_3* a rating of 7, whereas *user_3* has given *property_1* a rating of 4. We may estimate the rating for *property_2* by *user_1* as 5.5 if *property_3* and *property_1* are comparable to *property_2* at the same distance. If property_3 is regarded to be more similar to *property_2*, its weight should be higher than that of *property_1*. As a result, as shown in the diagram below, the anticipated rating for *property_2* will be closer to the rating for *property_3*. And then we can calculate the predicted rating by cosine similarity as the weight [4].

The above steps are the most important while creating the User-Based Collaborative Filtering recommendation engine. The code with comments can be found in the Jupyter Notebook.

4.3.3.2 Content-Based Filtering

The basic concept which was used to create this part is to transform each row of the dataset as the single vector. And then we tried to work with the single vectors and find similarities between them. The method which was used to compare vector's similarity is called cosine similarity. For implementing of the cosine similarity, we are using *cosine_similarity* function [41] form sklearn.

Also in this part was made the following data transformation before using of cosine similarity:

1. Was dropped the following columns: *city, country, streetName, streetNumber, postalCode, streetNumberNumeric and unitNumber*. It was dropped because of duplicates, gps coordinates are enough to describe the location.

2. Was used such methods as *get_dummies* [42] and *OneHotEncoding* [43] to convert categorical variable into indicator variables. By other words, we were trying to convert as much data as possible to the vector of 0 and 1.

3. Now each row will be presented as the single vector. And we can use the cosine similarity for each of these vectors.

The whole dataset before and after using of cosine similarity and the described methods above are available on Jupyter Notebook.

4.3.4 Testing of the obtained results

In this part we will analyze obtained result from our Content-Based Filtering model, which is based on the cosine similarity.

4.3.4.1 User-Based Collaborative Filtering

While the creating of the User-Based Collaborative Filtering with the way, which is described above, we didn't got good results, because of the small amount of data. So we decided to focus on the Content-Based Filtering model. The dataset with rating consists only from 12 rows, which is now enough for creating of this types of recommendation engine, however the code is available in attachment. This part can be done in the future improvements.

4.3.4.2 Content-Based Filtering

After comparing vectors by cosine similarity, we got the sparse matrix with scores between each pair of vectors in the listing 4.3.4.2

Listing 4.1: The sparse matrix

```
[[1.      0.85365854 0.79674797 ... 0.72899729 0.83468835]
 [0.85365854 1.      0.80487805 ... 0.76693767 0.80216802]
 [0.79674797 0.80487805 1.      ... 0.79132791 0.79403794]
 ...
 [0.92682927 0.81842818 0.75609756 ... 0.72086721 0.84281843]
 [0.72899729 0.76693767 0.79132791 ... 1.      0.75067751]
 [0.83468835 0.80216802 0.79403794 ... 0.75067751 1.    ]]
```

Now the next step required is to create the function which will take an index on input, find the same index in our matrix and take the biggest numbers, which will show the most similar properties for our input property. We will return top 10 properties. For input equals to *1*, it's the index of the property in dataset, we got the following properties and scores 4.3. The index of the property is the first parameter and the score is the second parameter.

As we can see on the figure 4.3 the first row with the index of property: *1* and the score equals to *1.0000000000000033*, we can make the conclusion, that the similarity matrix was done in the correct way. Because of property with index *1* is similar to itself on 100%.

Now lets look at the dataset and compare our input property from the Appendix B and other properties, which were returned as the most similar properties 4.4. For comparing output properties with the input property we will use the following columns: *propertyType, propertySubType, propertyCondition, transportAvailability, surroundingArea, service, roomCount, floorCount-Aboveground, parkingLotCount, lift, accessibility, parkingFeatures, furnished,*

```
[[(1, 1.0000000000000033),
  (945, 0.8780487804878077),
  (2823, 0.8726287262872656),
  (3565, 0.8726287262872656),
  (1329, 0.8672086720867236),
  (1564, 0.8672086720867236),
  (1690, 0.8672086720867236),
  (2835, 0.8672086720867236),
  (363, 0.8644986449864526),
  (2300, 0.8644986449864526)]]
```

Figure 4.3: The scores returned from the Content-Based Filtering model, which was based on cosine similarity, for the input with the property by index equals to 1.

bathroomFeatures, constructionMaterials, school_middle, restaurant. For example, let's compare the columns *propertySubType* and *propertyType* from the Appendix B. The first row with index *1* is representing the input property. For the column *propertySubType* we wee, that 6 properties from 9 properties proposed properties have the same values, as input property, and for the column *propertyType* 8 properties form 9 properties proposed properties have the same values as input. The comparison of the above columns can be found in Appendix B.

	propertyType	propertySubType	ownershipType	state	currency	type	priceType
1	HOUSE	HOUSE_SINGLE_FAMILY_RESIDENCE	PERSONAL	ACTIVE	CZK	SELL	TOTAL
1385	HOUSE	HOUSE_SINGLE_FAMILY_RESIDENCE	PERSONAL	ACTIVE	CZK	SELL	TOTAL
4174	HOUSE	HOUSE_CHALET	PERSONAL	ACTIVE	CZK	SELL	TOTAL
5289	HOUSE	HOUSE_CHALET	PERSONAL	ACTIVE	CZK	SELL	TOTAL
1945	HOUSE	HOUSE_SINGLE_FAMILY_RESIDENCE	PERSONAL	ACTIVE	CZK	SELL	TOTAL
2303	HOUSE	HOUSE_SINGLE_FAMILY_RESIDENCE	PERSONAL	ACTIVE	CZK	SELL	TOTAL
2497	HOUSE	HOUSE_SINGLE_FAMILY_RESIDENCE	PERSONAL	ACTIVE	CZK	SELL	TOTAL
4191	HOUSE	HOUSE_SINGLE_FAMILY_RESIDENCE	PERSONAL	ACTIVE	CZK	SELL	TOTAL
538	APPARTMENT	APPARTMENT_3_PLUS_KK	PERSONAL	ACTIVE	CZK	SELL	TOTAL
3406	HOUSE	HOUSE_SINGLE_FAMILY_RESIDENCE	PERSONAL	ACTIVE	CZK	SELL	TOTAL

Figure 4.4: The output properties from Content-Based Filtering model on the input index equals to *1*.

Now let's test out model another input property with the index *67*, in the

same way which were described above. Parameters of another input property also can be found in the Appendix B. Let's look at the figure 4.5, we can see the top 10 scores of the most similar properties. The first row is our input row, so the score is equals to *1.0000000000000033* and it's correct. The list of recommended properties is presented in the figure 4.6

Now we will compare the input property with the recommended similar properties. We will use the columns above for comparing. The full comparison of the above columns can be found in the Appendix B.

```
[[(67, 1.0000000000000033),
  (3102, 0.93766937669377),
  (2609, 0.934959349593499),
  (3767, 0.9186991869918729),
  (4273, 0.9186991869918729),
  (1195, 0.9159891598916019),
  (1934, 0.9159891598916019),
  (3259, 0.9159891598916019),
  (3579, 0.9159891598916019),
  (3742, 0.9159891598916019)]]
```

Figure 4.5: The scores returned from the Content-Based Filtering model, which was based on cosine similarity, for the input with the property by index equals to 67.

Based on the fact that most of the values in the columns from Appendix B have similar values to the input property, we can say that the cosine similarity for vectors comparison works good for our dataset. Also to prove our conclusion we propose to look at the figure 4.7. The figure is showing how many percent of output values are the same as the input one.

	propertyType	propertySubType	ownershipType	state	currency	type	priceType
96	APPARTMENT	APPARTMENT_2_PLUS_1	PERSONAL	ACTIVE	CZK	SELL	TOTAL
4610	APPARTMENT	APPARTMENT_2_PLUS_1	COOPERATIVE	ACTIVE	CZK	SELL	TOTAL
3868	APPARTMENT	APPARTMENT_2_PLUS_1	PERSONAL	ACTIVE	CZK	SELL	TOTAL
5585	APPARTMENT	APPARTMENT_2_PLUS_1	PERSONAL	ACTIVE	CZK	SELL	TOTAL
6346	APPARTMENT	APPARTMENT_2_PLUS_1	PERSONAL	ACTIVE	CZK	SELL	TOTAL
1754	APPARTMENT	APPARTMENT_1_PLUS_1	PERSONAL	ACTIVE	CZK	SELL	TOTAL
2834	APPARTMENT	APPARTMENT_2_PLUS_1	COOPERATIVE	ACTIVE	CZK	SELL	TOTAL
4827	APPARTMENT	APPARTMENT_3_PLUS_1	PERSONAL	ACTIVE	CZK	SELL	TOTAL
5309	APPARTMENT	APPARTMENT_2_PLUS_1	PERSONAL	ACTIVE	CZK	SELL	TOTAL
5542	APPARTMENT	APPARTMENT_2_PLUS_1	PERSONAL	ACTIVE	CZK	SELL	TOTAL

Figure 4.6: The output properties from Content-Based Filtering model on the input index equals to *67*.

Percent distribution of the correct values

Input property index	propertyType	propertySubType	propertyCondition	transportAvailability	surrounding Area	service	lift	accessibility	parking Features	furnished	bathroomFeatures	constructionMaterials
1	88	66	22	22	33	33	100	100	66	55	66	44
67	100	11	44	100	55	11	88	11	100	100	77	100
456	44	44	11	88	66	88	22	11	33	55	100	11
1234	88	77	66	33	22	44	100	100	22	55	66	11
952	100	22	44	11	33	77	11	100	77	11	77	88

Figure 4.7: The comparison of the values from input property with the values of recommended properties. Each value represents the percent of the same values, which the input property had.

Conclusion

In this thesis, we familiarized ourselves with the issue of real estate evaluation for the Czech market and understand, what influences on user's choice.

In Chapter 2 we have defined what the real estate is, which helped us to understand and structure work into different topics. While researching the index called HPI was found, which we used to analyze and to compare Czech real estate market and USA real estate market. At the end of this research we concluded that the interest on the real estate is intensively growing and, because of that reason, we made the conclusion that for the regular customers it is also getting more complicated to find the suitable property. That is why the topic of recommendation engines can be considered be interesting.

In Chapter 3 we defined the main types of datasets which we will use: the dataset which describes the properties, the dataset which describes user preferences and consist the rating from users on the certain apartments and the dataset, which describes the points of interests. For each of this dataset an analyze was made. Also in this chapter two ways of data enrichment were proposed: the internal one and the external one. As a part of an internal enrichment, the algorithm was proposed, which can describe the location using seven main categories, that were defined by dataset with the point of interests: coffee, pub, school elementary, school middle, school high, fitness and restaurant. And as a part of external enrichment, we proposed the way and the platform which can be used to analyze images of properties on property condition. At the end of this chapter the mentioned algorithm was used for the location enrichment and for calculating of indexes for each property in our dataset.

In Chapter 4 we defined what recommendation system or engine is and researched the stat-of-art articles about the current topic. Then we identified the main types of recommendation systems: the Content-Based Filtering, the

Collaborating Filtering and the Hybrid Recommendation System. Also the subcategories were defined and the main difference between them. The next part was primarily focused on understanding of the main issues connected with the creating of real estate recommendation engines: cold-start problem, sparsity of data and the feature specification. And the last part of this chapter is about applying of gathered knowledge into our case. Here we proposed which type of the recommendation engine will be suitable in our case and also the ways of solving the problems above. Next task was to create and test two draft of different recommendation engines: the User-Based Collaborative Filtering and the Content-Based Filtering. After testing, we realised that the good results can only be provided with the quality and big amount of data concerning the user preferences, which, unfortunately, we currently can not achieve. Also we would like to mention that there is no certain answer about recommendation system suitability, it's always needed to combine different systems.

Future Work

We would like to mention that this thesis may be perceived as the starting platform for improving this topic. As we understand after the testing of our drafts, there are a lot of different topics, which can be improved:

1. Also in order to get the most out of the real estate dataset [44], we would like to pay attention to the images, because by analyzing images we can understand things like interior/exterior style, real estate condition, find similar images in other possible listings. The API that we will use for image analysis is called restb.ai [25]. More details can be found in the part 3.2.2.

2. The possibility of changing the radius of the clusters, for index calculation in the part 3.2.1, could be added. This is due to the fact, that for some people walking 500 meters to a restaurant is a normal distance, but for others it is a long distance. In the future version of the algorithm the default radius can be set depending on the city population. Also we would like to point out that it will be imported to create the mode, which will set up the correct weight for the distance according to the user preferences or the location peculiarities.

3. Try to use not the Euclidean distance between two POI.

4. To create the system for automatically weights set up for the index algorithm.

5. Apply different scenarios of index creating for people, who have a car, us public transport a lot or mostly walk.

6. To test created drafts on the bigger amount of data.
7. To apply different recommendation models such as the Hybrids.

Each of this topics can be used as an extension for the thesis.

Open source

All source code on github, algorithm which was applied and the data, which has been received during the research, are open for use and improvements.

Bibliography

[1] Czech Republic House Price index2022 data - 2023 forecast - 2008-2021 historical. [online], [cit. 2022-04-12]. Available from: https://tradingeconomics.com/czech-republic/housing-index

[2] United States FHFA house price IndexMarch 2022 data - 1991-2021 historical. [online], [cit. 2022-04-12]. Available from: https://tradingeconomics.com/united-states/housing-index

[3] Overpass Turbo. [online], [cit. 2022-04-12]. Available from: https://overpass-turbo.eu/

[4] Jeong, Y. Item-based collaborative filtering in Python. [online], [cit. 2022-04-15]. Available from: https://towardsdatascience.com/item-based-collaborative-filtering-in-python-91f747200fab

[5] Cosine similarity. [online], [cit. 2022-03-17]. Available from: https://www.learndatasci.com/glossary/cosine-similarity/

[6] Property condition demo. [online], [cit. 2022-04-12]. Available from: https://demo.restb.ai/property/

[7] real-estate noun - Definition, pictures, pronunciation and usage notes — Oxford Advanced Learner's Dictionary at OxfordLearnersDictionaries.com. [online], [cit. 2022-04-12]. Available from: https://www.oxfordlearnersdictionaries.com/definition/english/real-estate?q=real%2Bestate

[8] House price index. [online], [cit. 2022-04-12]. Available from: https://en.wikipedia.org/wiki/House_price_index

[9] HB index: Zdražování Nemovitostí Neustává. Ceny Pozemků Raketově Rostou a trhají Rekordy. [online], [cit. 2022-04-12]. Available from: https://www.hypotecnibanka.cz/o-bance/pro-media/hb-index/hb-index-zdrazovani-nemovitosti-neustava-/

[10] Ceny Nemovitosti. [online], [cit. 2022-04-12]. Available from: https://www.czso.cz/csu/czso/indexy_cen_nemovitosti

[11] HB index. [online], [cit. 2022-04-12]. Available from: https://www.hypotecnibanka.cz/o-bance/pro-media/hb-index/page\protect\protect\leavevmode@ifvmode\kern+.2222em\relax1/

[12] sreality.cz • reality a nemovitosti z celé čr. [online], [cit. 2022-04-12]. Available from: https://www.sreality.cz/

[13] Bezrealitky. [online], [cit. 2022-04-12]. Available from: https://www.bezrealitky.cz/

[14] Prodáme I Vaši NEMOVITOST: M&M reality. [online], [cit. 2022-04-12]. Available from: https://www.mmreality.cz/

[15] Cold start (recommender systems). [online], [cit. 2022-04-12]. Available from: https://en.wikipedia.org/wiki/Cold_start_(recommender_systems)

[16] Home — Federal Housing Finance Agency. [online], [cit. 2022-04-12]. Available from: https://www.fhfa.gov/

[17] New York City's largest source of verified home listings. [online], [cit. 2022-04-12]. Available from: https://www.gobii.com/landing

[18] Real estate, apartments, Mortgages & Home values. [online], [cit. 2022-04-12]. Available from: https://www.zillow.com/

[19] Discover a place you'll love to live. [online], [cit. 2022-04-12]. Available from: https://www.trulia.com/

[20] Trulia: Disclaimers. [online], [cit. 2022-04-12]. Available from: https://www.trulia.com/disclaimers/%23schools/

[21] STORMM exporty - administrace. [online], [cit. 2022-04-12]. Available from: https://export.stormm.cz/

[22] Yuan, Q.; Cong, G.; et al. Time-aware point-of-interest recommendation. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, 2013, pp. 363–372.

[23] OpenStreetMap. [online], [cit. 2022-04-12]. Available from: https://www.openstreetmap.org/#map=8/49.817/15.478

[24] Property condition API: Restb.ai Computer Vision. [online], [cit. 2022-04-12]. Available from: https://restb.ai/solutions/property-condition/

[25] Computer vision with an eye for real estate. [online], [cit. 2022-04-12]. Available from: `https://restb.ai/`

[26] Pricing: Restb.ai's Specialized Computer Vision Solutions. [online], [cit. 2022-04-12]. Available from: `https://restb.ai/pricing/`

[27] Open-source geocoding. [online], [cit. 2022-04-12]. Available from: `https://nominatim.org`

[28] Gharahighehi, A.; Pliakos, K.; et al. Recommender Systems in the Real Estate Market—A Survey. *Applied Sciences*, volume 11, no. 16, 2021: p. 7502.

[29] Ricci, F.; Rokach, L.; et al. Introduction to recommender systems handbook. In *Recommender systems handbook*, Springer, 2011, pp. 1–35.

[30] Felfernig, A.; Isak, K.; et al. The VITA financial services sales support environment. In *Proceedings of the national conference on artificial intelligence*, volume 22, Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007, p. 1692.

[31] Paterek, A. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD cup and workshop*, volume 2007, 2007, pp. 5–8.

[32] Lops, P.; Jannach, D.; et al. Trends in content-based recommendation. *User Modeling and User-Adapted Interaction*, volume 29, no. 2, 2019: pp. 239–249.

[33] Rendle, S.; Freudenthaler, C.; et al. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618*, 2012.

[34] Batet, M.; Moreno, A.; et al. Turist@: Agent-based personalised recommendation of tourist activities. *Expert systems with applications*, volume 39, no. 8, 2012: pp. 7319–7329.

[35] Burke, R. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, volume 12, no. 4, 2002: pp. 331–370.

[36] Rehman, F.; Masood, H.; et al. An Intelligent Context Aware Recommender System for Real-Estate. In *Mediterranean Conference on Pattern Recognition and Artificial Intelligence*, Springer, 2019, pp. 177–191.

[37] Zhang, Q.; Zhang, D.; et al. A Recommender System for Cold-start Items: A Case Study in the Real Estate Industry. In *2019 IEEE 14th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*, IEEE, 2019, pp. 1185–1192.

[38] Oh, D.; Tan, C. L. Making better recommendations with online profiling agents. *AI Magazine*, volume 26, no. 3, 2005: pp. 29–29.

[39] Yuan, X.; Lee, J.-H.; et al. Toward a user-oriented recommendation system for real estate websites. *Information Systems*, volume 38, no. 2, 2013: pp. 231–243.

[40] Daly, E. M.; Botea, A.; et al. Multi-criteria journey aware housing recommender system. In *Proceedings of the 8th ACM Conference on Recommender systems*, 2014, pp. 325–328.

[41] Sklearn.metrics.pairwise.cosine_similarity. [online], [cit. 2022-04-17]. Available from: `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.pairwise.cosine_similarity.html`

[42] Pandas.get_dummies. [online], [cit. 2022-04-17]. Available from: `https://pandas.pydata.org/docs/reference/api/pandas.get_dummies.html`

[43] Sklearn.preprocessing.onehotencoder. [online], [cit. 2022-04-17]. Available from: `https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html`

[44] Implementation/Data/mm.csv · master · Roman Zhuravskyi / bachelor work. [online], [cit. 2022-04-12]. Available from: `https://gitlab.fit.cvut.cz/zhurarom/bachelor-work/blob/master/Implementation/Data/MM.csv`

Appendix A

Acronyms

POI Point of Interest

OSM Open Street Map

XML Extensible Markup Language

CSV Comma-Separated Values

API Application Programming Interface

HPI House Price Index

FHFA Federal Housing Finance Agency Frequency

Appendix B

Appendix

Listing B.1: The column *school middle* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: school_middle
{
  1:  '0.6',
  538: '0.2',
  1385: '0.5',
  1945: '0.6',
  2303: '0.0',
  2497: '0.8',
  3406: '0.0',
  4174: '0.0',
  4191: '0.1',
  5289: '0.0'
}
```

Listing B.2: The column *school middle* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: school_middle
{
  96:  '0.2',
  1754: '7.2',
  2834: '0.5',
  3868: '0.7',
  4610: '7.5',
  4827: '3.0',
  5309: '0.0',
  5542: '0.2',
  5585: '5.1',
  6346: '3.6'
}
```

Listing B.3: The column *restaurant* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: restaurant
```

```
{
  1:  '1.9',
  538: '6.1',
  1385: '0.4',
  1945: '5.6',
  2303: '1.2',
  2497: '5.3',
  3406: '0.4',
  4174: '0.9',
  4191: '1.7',
  5289: '0.9'
}
```

Listing B.4: The column *restaurant* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: restaurant
{
  96:  '4.3',
  1754: '6.6',
  2834: '5.7',
  3868: '5.4',
  4610: '5.3',
  4827: '5.3',
  5309: '6.9',
  5542: '5.5',
  5585: '5.0',
  6346: '5.8'
}
```

Listing B.5: The column *propertyType* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: propertyType
{
  1:  'HOUSE',
  538: 'APPARTMENT',
  1385: 'HOUSE',
  1945: 'HOUSE',
  2303: 'HOUSE',
  2497: 'HOUSE',
  3406: 'HOUSE',
  4174: 'HOUSE',
  4191: 'HOUSE',
  5289: 'HOUSE'
}
```

Listing B.6: The column *propertyType* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: propertyType
{
  96:  'APPARTMENT',
  1754: 'APPARTMENT',
```

```
  2834: 'APPARTMENT',
  3868: 'APPARTMENT',
  4610: 'APPARTMENT',
  4827: 'APPARTMENT',
  5309: 'APPARTMENT',
  5542: 'APPARTMENT',
  5585: 'APPARTMENT',
  6346: 'APPARTMENT'
}
```

Listing B.7: The column *propertySubType* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: propertySubType
{
  1:  'HOUSE_SINGLE_FAMILY_RESIDENCE',
  538: 'APPARTMENT_3_PLUS_KK',
  1385: 'HOUSE_SINGLE_FAMILY_RESIDENCE',
  1945: 'HOUSE_SINGLE_FAMILY_RESIDENCE',
  2303: 'HOUSE_SINGLE_FAMILY_RESIDENCE',
  2497: 'HOUSE_SINGLE_FAMILY_RESIDENCE',
  3406: 'HOUSE_SINGLE_FAMILY_RESIDENCE',
  4174: 'HOUSE_CHALET',
  4191: 'HOUSE_SINGLE_FAMILY_RESIDENCE',
  5289: 'HOUSE_CHALET'
}
```

Listing B.8: The column *propertySubType* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: propertySubType
{
  96:  'APPARTMENT_2_PLUS_1',
  1754: 'APPARTMENT_1_PLUS_1',
  2834: 'APPARTMENT_2_PLUS_1',
  3868: 'APPARTMENT_2_PLUS_1',
  4610: 'APPARTMENT_2_PLUS_1',
  4827: 'APPARTMENT_3_PLUS_1',
  5309: 'APPARTMENT_2_PLUS_1',
  5542: 'APPARTMENT_2_PLUS_1',
  5585: 'APPARTMENT_2_PLUS_1',
  6346: 'APPARTMENT_2_PLUS_1'
}
```

Listing B.9: The column *propertyCondition* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: propertyCondition
{
  1:  'GOOD',
  538: 'GOOD',
  1385: 'GOOD',
  1945: 'EXCELLENT',
  2303: 'GOOD',
```

```
  2497: 'BEFORE_RECONSTRUCTION',
  3406: 'GOOD',
  4174: 'UNDER_CONSTRUCTION',
  4191: 'EXCELLENT',
  5289: 'UNDER_CONSTRUCTION'
}
```

Listing B.10: The column *propertyCondition* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: propertyCondition
{
  96:  'EXCELLENT',
  1754: 'EXCELLENT',
  2834: 'EXCELLENT',
  3868: 'EXCELLENT',
  4610: 'GOOD',
  4827: 'EXCELLENT',
  5309: 'GOOD',
  5542: 'GOOD',
  5585: 'GOOD',
  6346: 'EXCELLENT'
}
```

Listing B.11: The column *transportAvailability* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: transportAvailability
{
  1:  'BUS,TRUCK,OWN_CAR',
  538: 'BUS,OWN_CAR',
  1385: 'BUS,OWN_CAR',
  1945: '',
  2303: 'BUS,TRAIN,OWN_CAR',
  2497: 'BUS,TRUCK,OWN_CAR',
  3406: '',
  4174: 'OWN_CAR',
  4191: 'TRUCK,OWN_CAR',
  5289: 'OWN_CAR'
}
```

Listing B.12: The column *transportAvailability* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: transportAvailability
{
  96:  'BUS,TRAIN,CITY_TRANSPORT,OWN_CAR',
  1754: 'BUS,TRAIN,CITY_TRANSPORT,OWN_CAR',
  2834: 'BUS,TRAIN,CITY_TRANSPORT,OWN_CAR',
  3868: 'BUS,TRAIN,CITY_TRANSPORT,OWN_CAR',
  4610: 'BUS,TRAIN,CITY_TRANSPORT,OWN_CAR',
  4827: 'BUS,TRAIN,CITY_TRANSPORT,OWN_CAR',
  5309: 'BUS,TRAIN,OWN_CAR',
  5542: 'BUS,TRAIN,CITY_TRANSPORT,OWN_CAR',
```

```
  5585: 'BUS,TRAIN,CITY_TRANSPORT,OWN_CAR',
  6346: 'BUS,TRAIN,CITY_TRANSPORT,OWN_CAR'
}
```

Listing B.13: The column *surroundingArea* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: surroundingArea
{
  1:  'MOUNTAIN,FOREST,MEADOW,RECREATIONAL_DEVELOPMENT,RIVER',
  538: '',
  1385: '',
  1945: '',
  2303: '',
  2497: '',
  3406: '',
  4174: 'MOUNTAIN,FOREST,MEADOW,FIELD,RECREATIONAL_DEVELOPMENT',
  4191: 'FOREST,MEADOW,FIELD,RECREATIONAL_DEVELOPMENT,GARDENS',
  5289: 'MOUNTAIN,FOREST,MEADOW,FIELD,RECREATIONAL_DEVELOPMENT'
}
```

Listing B.14: The column *surroundingArea* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: surroundingArea
{
  96:  'URBAN_DEVELOPMENT',
  1754: 'URBAN_DEVELOPMENT,HOUSING_DEVELOPMENT',
  2834: 'URBAN_DEVELOPMENT',
  3868: 'RIVER,HOUSING_DEVELOPMENT',
  4610: 'URBAN_DEVELOPMENT',
  4827: 'HOUSING_DEVELOPMENT',
  5309: 'URBAN_DEVELOPMENT',
  5542: 'URBAN_DEVELOPMENT',
  5585: '',
  6346: 'HOUSING_DEVELOPMENT'
}
```

Listing B.15: The column *service* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: service
{
  1:  'DOCTOR,POST,ELEMENTARY_SCHOOL',
  538: 'DOCTOR,KINDERGARTEN,SHOPS,POST,ELEMENTARY_SCHOOL',
  1385: '',
  1945: '',
  2303: '',
  2497: '',
  3406: '',
  4174: 'BANK,DOCTOR,KINDERGARTEN,SHOPS,POST,ELEMENTARY_SCHOOL',
  4191: '',
  5289: 'BANK,DOCTOR,KINDERGARTEN,SHOPS,POST,ELEMENTARY_SCHOOL'
}
```

Listing B.16: The column *service* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: service
{
  96:  'BANK,DOCTOR,KINDERGARTEN,SHOPS,POST,ELEMENTARY_SCHOOL',
  1754: 'BANK,DOCTOR,KINDERGARTEN,SHOPS,POST,ELEMENTARY_SCHOOL',
  2834: 'BANK,DOCTOR,KINDERGARTEN,SHOPS,POST,ELEMENTARY_SCHOOL',
  3868: 'BANK,DOCTOR,KINDERGARTEN,SHOPS,POST,ELEMENTARY_SCHOOL',
  4610: 'BANK,DOCTOR,KINDERGARTEN,SHOPS,POST,ELEMENTARY_SCHOOL',
  4827: 'BANK,DOCTOR,KINDERGARTEN,SHOPS,POST,ELEMENTARY_SCHOOL',
  5309: 'BANK,DOCTOR,KINDERGARTEN,SHOPS,POST,ELEMENTARY_SCHOOL',
  5542: 'BANK,DOCTOR,KINDERGARTEN,SHOPS,POST,ELEMENTARY_SCHOOL',
  5585: 'BANK,CULTURE,DOCTOR,KINDERGARTEN,MALL,SHOPS,POST,
      SPORTS_GROUND,ELEMENTARY_SCHOOL',
  6346: 'BANK,DOCTOR,KINDERGARTEN,SHOPS,POST,ELEMENTARY_SCHOOL'
}
```

Listing B.17: The column *roomCount* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: roomCount
{
  1:  '8.0',
  538: '',
  1385: '6.0',
  1945: '6.0',
  2303: '7.0',
  2497: '8.0',
  3406: '4.0',
  4174: '2.0',
  4191: '3.0',
  5289: '2.0'
}
```

Listing B.18: The column *roomCount* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: roomCount
{
  96:  '3.0',
  1754: '2.0',
  2834: '3.0',
  3868: '4.0',
  4610: '3.0',
  4827: '4.0',
  5309: '3.0',
  5542: '4.0',
  5585: '3.0',
  6346: '4.0'
}
```

Listing B.19: The column *floorCountAboveground* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: floorCountAboveground
```

```
{
  1:  '1.0',
  538: '2.0',
  1385: '2.0',
  1945: '2.0',
  2303: '2.0',
  2497: '2.0',
  3406: '1.0',
  4174: '2.0',
  4191: '1.0',
  5289: '2.0'
}
```

Listing B.20: The column *floorCountAboveground* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: floorCountAboveground
{
  96:  '8.0',
  1754: '8.0',
  2834: '9.0',
  3868: '8.0',
  4610: '9.0',
  4827: '8.0',
  5309: '4.0',
  5542: '3.0',
  5585: '8.0',
  6346: '7.0'
}
```

Listing B.21: The column *parkingLotCount* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: parkingLotCount
{
  1:  '2.0',
  538: '2.0',
  1385: '2.0',
  1945: '2.0',
  2303: '2.0',
  2497: '',
  3406: '',
  4174: '1.0',
  4191: '1.0',
  5289: '1.0'
}
```

Listing B.22: The column *parkingLotCount* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: parkingLotCount
{
  96:  '20.0',
  1754: '1.0',
```

```
  2834: '1.0',
  3868: '20.0',
  4610: '50.0',
  4827: '20.0',
  5309: 'nan',
  5542: 'nan',
  5585: '30.0',
  6346: '20.0'
}
```

Listing B.23: The column *lift* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: lift
{
  1:  'False',
  538: 'False',
  1385: 'False',
  1945: 'False',
  2303: 'False',
  2497: 'False',
  3406: 'False',
  4174: 'False',
  4191: 'False',
  5289: 'False'
}
```

Listing B.24: The column *lift* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: lift
{
  96:  'True',
  1754: 'True',
  2834: 'True',
  3868: 'True',
  4610: 'True',
  4827: 'True',
  5309: 'True',
  5542: 'False',
  5585: 'True',
  6346: 'True'
}
```

Listing B.25: The column *accessibility* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: accessibility
{
  1:  'False',
  538: 'False',
  1385: 'False',
  1945: 'False',
  2303: 'False',
```

```
  2497: 'False',
  3406: 'False',
  4174: 'False',
  4191: 'False',
  5289: 'False'
}
```

Listing B.26: The column *accessibility* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: accessibility
{
  96:  'False',
  1754: 'False',
  2834: 'False',
  3868: 'False',
  4610: 'False',
  4827: 'True',
  5309: 'False',
  5542: 'False',
  5585: 'False',
  6346: 'False'
}
```

Listing B.27: The column *parkingFeatures* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: parkingFeatures
{
  1:  'OUTSIDE',
  538: 'ON_STREET',
  1385: 'GARAGE,OUTSIDE,ON_STREET',
  1945: '',
  2303: 'OUTSIDE,ON_STREET',
  2497: 'OUTSIDE,ON_STREET',
  3406: '',
  4174: 'OUTSIDE',
  4191: 'OUTSIDE,ON_STREET',
  5289: 'OUTSIDE'
}
```

Listing B.28: The column *parkingFeatures* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: parkingFeatures
{
  96:  'ON_STREET',
  1754: 'ON_STREET',
  2834: 'ON_STREET',
  3868: 'ON_STREET',
  4610: 'ON_STREET',
  4827: 'ON_STREET',
  5309: 'ON_STREET',
  5542: 'ON_STREET',
```

```
  5585: 'ON_STREET',
  6346: 'ON_STREET'
}
```

Listing B.29: The column *furnished* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: furnished
{
  1:  'PARTIALLY',
  538: 'PARTIALLY',
  1385: 'UNFURNISHED',
  1945: 'UNFURNISHED',
  2303: 'PARTIALLY',
  2497: 'UNFURNISHED',
  3406: 'PARTIALLY',
  4174: 'UNFURNISHED',
  4191: 'PARTIALLY',
  5289: 'UNFURNISHED'
}
```

Listing B.30: The column *furnished* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: furnished
{
  96:  'PARTIALLY',
  1754: 'PARTIALLY',
  2834: 'PARTIALLY',
  3868: 'PARTIALLY',
  4610: 'PARTIALLY',
  4827: 'PARTIALLY',
  5309: 'PARTIALLY',
  5542: 'PARTIALLY',
  5585: 'PARTIALLY',
  6346: 'PARTIALLY'
}
```

Listing B.31: The column *bathroomFeatures* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: bathroomFeatures
{
  1:  'CERAMIC_TILES,SHOWER_CABIN,TOILET',
  538: 'CERAMIC_TILES,WASHBASIN,BATHTUB',
  1385: 'CERAMIC_TILES,BATHTUB,TOILET',
  1945: 'SHOWER_CABIN,WASHBASIN,BATHTUB',
  2303: '',
  2497: 'SHOWER_CABIN,WASHBASIN,TOILET',
  3406: '',
  4174: 'CERAMIC_TILES,SHOWER_CABIN,WASHBASIN',
  4191: 'SHOWER_CABIN,WASHBASIN,TOILET',
  5289: 'CERAMIC_TILES,SHOWER_CABIN,WASHBASIN'
}
```

Listing B.32: The column *bathroomFeatures* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: bathroomFeatures
{
  96:  'WASHBASIN,BATHTUB,TOILET',
  1754: 'SHOWER_CABIN,WASHBASIN,TOILET',
  2834: 'SHOWER_CABIN,TOILET',
  3868: 'WASHBASIN,BATHTUB',
  4610: 'WASHBASIN,BATHTUB,TOILET',
  4827: 'SHOWER_CABIN,WASHBASIN,BATHTUB,TOILET',
  5309: 'WASHBASIN,BATHTUB,TOILET',
  5542: 'CERAMIC_TILES,WASHBASIN,BATHTUB,TOILET',
  5585: 'BATHTUB,TOILET',
  6346: 'WASHBASIN,BATHTUB,TOILET'
}
```

Listing B.33: The column *constructionMaterials* from output of Content-Based Filtering. The row with the key 1 is the input property.

```
Current column: constructionMaterials
{
  1:  'COMBINED,BRICK',
  538: 'BRICK',
  1385: '',
  1945: 'BRICK',
  2303: '',
  2497: 'BRICK',
  3406: '',
  4174: '',
  4191: 'BRICK',
  5289: ''
}
```

Listing B.34: The column *constructionMaterials* from output of Content-Based Filtering. The row with the key 67 is the input property.

```
Current column: constructionMaterials
{
  96:  'PANEL',
  1754: 'PANEL',
  2834: 'PANEL',
  3868: 'PANEL',
  4610: 'PANEL',
  4827: 'PANEL',
  5309: 'PANEL',
  5542: 'PANEL',
  5585: 'PANEL',
  6346: 'PANEL'
}
```

Listing B.35: The list of available columns in dataset of properties.

```
['propertyType', 'propertySubType', 'ownershipType', 'state',
'currency', 'type', 'priceType', 'pricePeriod', 'listPrice',
```

```
'transportAvailability', 'surroundingArea', 'service', 'city',
'country', 'postalCode', 'streetName', 'streetNumber',
'streetNumberNumeric', 'unitNumber', 'latitude', 'longitude',
'roomCount', 'floorNumber', 'floorCountAboveground',
'floorCountUnderground', 'flatCountTotal', 'parkingLotCount',
'builtupYear', 'reconstructionYear', 'lift', 'accessibility',
'maisonnette', 'communityFeatures', 'parkingFeatures', 'furnished',
'bathroomFeatures', 'toiletFeatures', 'accessToLot', 'fence',
'landType', 'garden', 'gardenCondition', 'areaUsable', 'areaLot',
'waterSource', 'electric', 'waterQuality', 'waterHotSource', 'sewer',
'gasDistribution', 'engineeringNetworks', 'fuelUsed', 'heating',
'heatingYN', 'roofShape', 'roofMaterial', 'plasterType',
'moistureInsulation', 'moistureStatus', 'constructionMaterials',
'flooring', 'ceilings', 'wallMaterial', 'directionFaces',
'propertyCondition', 'balcony', 'balconyCounter', 'coffee', 'pub',
'school_elementary', 'school_middle', 'school_high', 'fitness',
'restaurant']
```

Listing B.36: The input property with the index 1

```
{
  'propertyType': 'HOUSE',
  'propertySubType': 'HOUSE_SINGLE_FAMILY_RESIDENCE',
  'ownershipType': 'PERSONAL',
  'state': 'ACTIVE',
  'currency': 'CZK',
  'type': 'SELL',
  'priceType': 'TOTAL',
  'listPrice': '5990000.0',
  'transportAvailability': 'BUS,TRUCK,OWN_CAR',
  'surroundingArea': 'MOUNTAIN,FOREST,MEADOW,RECREATIONAL_DEVELOPMENT,RIVER',
  'service': 'DOCTOR,POST,ELEMENTARY_SCHOOL',
  'latitude': 50.19071899999999},
  'longitude': 16.2955007},
  'roomCount': '8.0',
  'floorCountAboveground': '1.0',
  'parkingLotCount': '2.0',
  'reconstructionYear': '2017.0',
  'lift': 'False',
  'accessibility': 'False',
  'maisonnette': 'False',
  'communityFeatures': 'CYCLING',
  'parkingFeatures': 'OUTSIDE',
  'furnished': 'PARTIALLY',
  'bathroomFeatures': 'CERAMIC_TILES,SHOWER_CABIN,TOILET',
  'toiletFeatures': 'THREE_MORE',
  'areaUsable': '220.0',
  'areaLot': '759.0',
  'electric': 'VOLT_220,VOLT_380',
  'waterQuality': 'DRINKING',
  'sewer': 'CESSPOOL',
  'engineeringNetworks': 'WATER',
  'fuelUsed': 'ELECTRIC',
  'heating': 'DUCTLESS',
```

```
  'heatingYN': 'True',
  'roofShape': 'GABLE',
  'moistureInsulation': 'ISOLATED',
  'moistureStatus': 'DRY',
  'constructionMaterials': 'COMBINED,BRICK',
  'flooring': 'CERAMIC_TILE,LAMINATE',
  'propertyCondition': 'GOOD',
  'balcony': 'NONE',
  'balconyCounter': '0',
  'coffee': '2.3',
  'pub': '2.3',
  'school_elementary': '2.1',
  'school_middle': '0.6',
  'school_high': '0.0',
  'fitness': '2.3',
  'restaurant': '1.9'
}
```

Listing B.37: The input property with the index 67

```
{
  'propertyType': 'APPARTMENT',
  'propertySubType': 'APPARTMENT_2_PLUS_1',
  'ownershipType': 'PERSONAL',
  'state': 'ACTIVE',
  'currency': 'CZK',
  'type': 'SELL',
  'priceType': 'TOTAL',
  'transportAvailability': 'BUS,TRAIN,CITY_TRANSPORT,OWN_CAR',
  'surroundingArea': 'URBAN_DEVELOPMENT',
  'service': 'BANK,DOCTOR,KINDERGARTEN,SHOPS,POST,ELEMENTARY_SCHOOL',
  'latitude': 50.3880633},
  'longitude': 13.2709175},
  'roomCount': '3.0',
  'floorNumber': '5.0',
  'floorCountAboveground': '8.0',
  'parkingLotCount': '20.0',
  'lift': 'True',
  'accessibility': 'False',
  'maisonnette': 'False',
  'communityFeatures': 'CYCLING,HORSEBACK_RIDING,CULTURE,CULTURAL_MONUMENTS,NATURE,TE
  'parkingFeatures': 'ON_STREET',
  'furnished': 'PARTIALLY',
  'bathroomFeatures': 'WASHBASIN,BATHTUB,TOILET',
  'toiletFeatures': 'ONE',
  'areaUsable': '51.0',
  'waterSource': 'PUBLIC',
  'electric': 'VOLT_220',
  'waterQuality': 'DRINKING',
  'waterHotSource': 'DISTRICT_HEATING',
  'sewer': 'PUBLIC',
  'gasDistribution': 'NATURAL_GAS',
  'engineeringNetworks': 'SEWERAGE,GAS,WATER',
  'heating': 'HUMIDITY_CONTROL',
```

```
  'heatingYN': 'True',
  'moistureInsulation': 'NONE',
  'moistureStatus': 'DRY',
  'constructionMaterials': 'PANEL',
  'flooring': 'PARQUET,VINYL',
  'ceilings': 'PANEL',
  'wallMaterial': 'STUCCO',
  'directionFaces': 'SOUTH,NORTH',
  'propertyCondition': 'EXCELLENT',
  'balcony': 'LOGGIA',
  'balconyCounter': '1',
  'coffee': '3.6',
  'pub': '6.6',
  'school_elementary': '5.1',
  'school_middle': '0.2',
  'school_high': '0.0',
  'fitness': '7.8',
  'restaurant': '4.3'
}
```

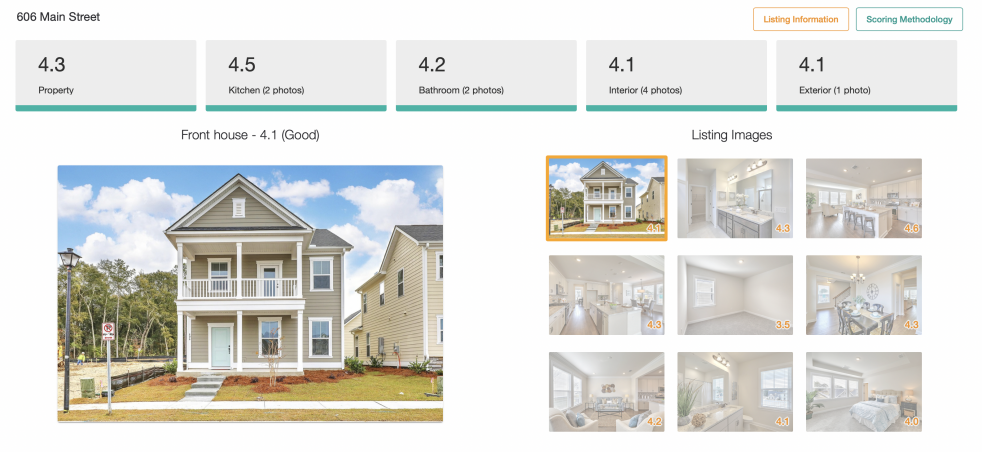

Figure B.1: The example of property condition evaluation for the good one house. Taken from [6].

Figure B.2: The example of property condition evaluation for the poor one house. Taken from [6].

Appendix C

Contents of enclosed CD

- `implementation` ..... the main directory
  - `src` ..... the directory of source codes
  - `notebooks` ..... the directory with Jupyter notebooks
  - `data` ..... the directory with datasets
- `text` ..... the thesis text directory
  - `thesis` ..... the directory of LaTeX source codes of the thesis
  - `thesis.pdf` ..... the thesis text in PDF format