Project D6: Real estate price prediction

Team members: Kaarel Vesilind, Tarmo Pungas, Marten Vainult
Link to Github repository

Business understanding

Identifying your business goals

Putting your real estate up for rent can be troublesome. Mainly because the price can fluctuate to a great extent. There's a lot of factors to consider: when was it built, how much does the infrastructure add in, how many rooms does it have etc. Even the smallest things can either pump or decrease the price. And for the exact same reasons it's difficult to know if the property is a good deal or not.

Most people in the renter's scenario go to an advertisement portal, where they can compare their property with others. And it gives a close estimate on the price, because the price is chosen mostly by market value. As for the interested tenant, there might be hidden gold up for rent, but it's hard to come by and takes a lot of effort to find.

Business goals

Our goal is to make life easier for both ends, by creating a handy tool which evaluates the real estate based on its properties. Renter would acquire a rightful price and an interested tenant would find a great deal.

Business success criteria

We will measure our results by testing the tool with average and professionally priced properties to see how closely the price is predicted. Succesful model would predict with the accuracy of at least ±10%

Assessing the situation

Inventory of resources

To get the most accurate pricing we need a lot of advertisements. Because the real estate properties may differ, we will only mine data from the biggest resource, which is KV.EE, Estonia's first and leading real estate portal. Each advertisement has some properties defined, but we'll try to read in extra features from the description (furnitured or not).

Our team consists of the finest diamonds in the upcoming data science world ;). For our work, we'll be using Google Colaboratory to work simultaneously and not worry about software problems, which might occur when trying out new packages.

Requirements, assumptions, and constraints

As this project is not handling personal information, we don't have any legal or security obligation. However we try to do our best not to make the data open to the public. We require that our finished work would predict with the accuracy of at least ±10%. This means that if a property's monthly rent is 500€, then our prediction would be in the range of 450 – 550€.
Our current schedule is:

- 26th November
  - Acquire the scraped data
  - Prepare the data
    - Remove irrelevant features
    - Remove irrelevant offers
    - Clean the data
    - Add Nulls or 0-s where necessary

- 3rd December
  - First model is trained

- 10th December
  - Prepare for the project showcase

- 17th December
  - Game day

Terminology

- Machine learning (ML) - Machine learning is the study of computer algorithms that improve automatically through experience.

- Dataset - A collection of data

- Plotting - Data set's graphical representation

- Model: The definition of a model, according to Merriam-Webster, is a "system of postulates, data, and inferences presented as a mathematical description of an entity or state of affairs."

- Feature engineering - Preparing the proper input dataset, compatible with the ML algorithm requirements to improve the performance of models

Costs and benefits

If done properly, the usage of this tool can be sold as a service for either putting up an advertisement or negotiating with the renter for a more suitable rent.

Defining data-mining goals

Data-mining goals

Our models' development is based on the effects of certain features to its final price. Also we are putting an effort on informative graphs. We will showcase the distribution of advertisements on a map and also the correlation of some real estate properties to its price.

Data-mining success criteria

To support our business success criteria we need to find the most accurate models with our datasets.

Data understanding

Gathering Data

Outline data requirements

- Fresh data(Advertisements used are maximum 1 month old, so the prediction model can stay up to date)
- Property location (geo coordinates)
- Property administrative location (Tartu, Tallinn, Paide,..)
- Property renting price (in euros)
- Property Type (Apartment, House, Garage, Commercial, …)
- Number of rooms (integer)
- Size of the property(area m^2) (integer
- Property state(renovated, new, bad etc)

optional

- Year constructed
- Floor number
- Energy class (A,B,S,D,F,G)
- Address (string)
- Property description (String) (Can be used to find some keywords. For example how is the apartment heatead.)

Verify data availability

The data is scraped weekly from kv.ee, so we have fresh data every week. Also, all the needed properties about a property are scraped from kv.ee, so we have all the required data.

Define selection criteria

We will use data, which is scraped weekly by Tarmo's brother Taivo Pungas from kv.ee. Because he scrapes all the advertisements(including sales) from kv.ee, then we have to do a lot of selection. First of all we only look at the properties which have transactionType = RENTAL. Then we choose only the properties where category = APARTMENT, because apartments are the most common rented real-estate. Also with all the other categories, there are a lot more things which affect the price. For example what exactly is the function of that property. That's why we are not using these. Also we only select the advertisements which are not passive, because they increase the inaccuracy of our model. We also drop images from our data, because we are not going to get any info from them. Right now we are not going to do any predictions from the images.
Additionally, we also only look at apartments which monthly price is over 70, because there were some advertisements, which had a daily rental price. All that's left now is the data we can use to do our predictions. The same data and columns which were listed in “Outline data requirements”.

Describing data

The data includes all the fields that we expect. We have the data which is suitable for our data-mining goals. There are 12 fields right now.

- price - monthly rent in euros
- time_scraped - time and date when this advertisement was scraped from kv.ee
- numRooms - number of rooms the apartment has
- area - area of the apartment in m^2
- yearConstructed - the year when the apartment/building was constructed
- state - state of the apartment(['Heas korras' 'Uus' 'Renoveeritud' 'Valmis' nan 'San. remont tehtud' 'Keskmine' 'Vajab renoveerimist' 'Vajab san. remonti']
- energyClass - Energyclass of the apartment A,B,C,D,E,F,G
- floor - in which floor the apartment is
- address - the address of the apartment
- municipality - municipality where the apartment is
- geo - latitude and longitude of the property
- cluster - cluster calculated from all the used apartment coordinates using (epsilon distance). Right now we have used OPTICS clustering.
  OPTICS works with geo-coordinates and it also deals with noise. Also it has the ability to have clusters with varying densities. We used right now min_samples=5, because it made decent amount of clusters(not too much and not too less) and also max_eps=0.1,

to eliminate noise in clusters. max_eps=0.1 is around 10km(so if an apartment is more than 10km away from the other, it is not in the same cluster anymore). One bad thing here with using (epsilon distance) is that in Estonia one degree longitude is 111km and one degree latitude is ~57km, so longitude values have pretty much double the weight than latitude. Might take it into account later, but right now we haven't taken it into account yet.

Exploring data

Distributions

Price

```
count    2710.000000
mean      465.808812
std       346.755800
min         8.000000
25%       280.000000
50%       390.000000
75%       550.000000
max      5500.000000
```

numRooms

```
count    2690.000000
mean        2.013755
std         1.202837
min         1.000000
25%         1.000000
50%         2.000000
75%         2.000000
max        29.000000
```

Area

```
count    2698.000000
mean       48.830504
std        29.588289
min         8.000000
25%        32.600000
50%        44.200000
75%        58.875000
max       660.000000
```

yearConstructed

```
count    1704.000000
```

```
mean     1977.535798
std        82.372875
min         2.000000
25%      1963.000000
50%      1985.000000
75%      2008.000000
max      2020.000000
```

State

```
count             2432
unique               8
top       Renoveeritud
freq               804
```

EnergyClass

```
count     1073
unique       7
top          D
freq       279
```

Floor

```
count    2541.000000
mean        3.145218
std         2.287024
min        -1.000000
25%         2.000000
50%         3.000000
75%         4.000000
max        29.000000
```

Municipality

```
count        2710
unique         54
top       Tallinn
freq         1816
```

Geo

```
count                                       2710
unique                                      2095
top       {'lat': 59.4342046, 'lng': 24.8096691}
freq                                          12
```

Cluster

```
Cluster_nr frequency
5      23
89     22
169    21
```

```
55     21
174    20
NaN   974   (unclustered noise)
```

Data quality problems

year_Constructed has 1006 NaN values and energyClass has 1637 NaN values, which is quite a lot, since the size of the dataframe is 2710.
Also there seems to be some faulty values in year_Constructed. The minimal value in year_constructed is 2, which is probably a mistake.
Besides these it looks like there aren't really big quality issues.

Summaries

Overall the data in all columns is good. The variability of course differences a bit in different columns but the values look true to what they are supposed to be. Also the dataset is big enough to make fair conclusions.

Verifying data quality

The data is good enough for our project goals. There are very little faulty values in the columns. Of course if we take new fresh data after one month, we have to recheck that it's still good. We are going to use only the values which stay inside 3 std-s in a column when we are training our model, so we still use 99.7% of the data and the extreme values will not make our model inaccurate.

Planning the project

- Data preparation | Tarmo Pungas, approx 3 hours
    - Drop “images” column
    - Do some research into “isPassive” column: is it necessary? Might be smart to drop all rows where isPassive == TRUE
    - Extract rows with “propertyType” == “apartment”, then drop the column
    - Extract rows with “transactionType” == “rental”, then drop the column
    - Clean up “category” – check that only apartments are left. Extract rows with “rental”.
    - Remove duplicates by “id”
    - Extract important data from “extra” as separate columns
    - Replace missing values with NaN
    - Group “time_scraped” by year/season?
    - Group “yearConstructed”?

    - Create dummy values for appropriate columns
- Geo clustering | Kaarel Vesilind, approx 2 hours
    - Find the best clustering method for columns “geo”(coordinates), to find clusters where the price higher than normal or lower.
    - Extract municipal area from “address”
- Initialize Google Colab notebook | Marten Vainult, approx 2 hours
    - Add code that’s already been written
    - Add team members as collaborators
- Visualizing
    - Plot all prices to get an idea of the shape of the data (scatter plot, perhaps) | Tarmo Pungas, approx 30 min
    - Plot the geo coordinates on a map | Marten Vainult, approx 30 min
    - Make GIS maps for visualization | Kaarel Vesilind, approx 1 hour
- Modeling and evaluating
    - Split into training, validation ja testing dataset (also remove price from validation and testing) | Kaarel Vesilind, approx 20 min
    - Decide on some questions. Are we going to use cross validation? Regularization? Model ensembles (bagging, boosting, stacking)? Combining different learners? | Everyone, approx 1 hour
    - Do some research into which models could/should be used: naïve Bayes, logistic regression, random forest, SVM… | Tarmo Pungas, approx 1 hour
    - Feature engineering: Train a bunch of models with different hyperparameters and check them on the validation dataset to see which perform best. | Everyone, approx 2 hours each
    - Fine-tuning opportunity: search for important info from “description” and improve model with that info (e.g. is the apartment furnished). | Tarmo Pungas, approx 3 hours