# Linear Regression Homework

Alireza Afzal Aghaei

March 4, 2021

**Abstract**

In this project we have addressed the linear regression model for an specific data-set. The data is collected from a real estate platform in Germany, named Immoscout24. The aim of this exercise is to predict the living space area of a house based on the other features, such as base rent or number of rooms. Here we analyse the data-set features, then remove the ineffective features. The learning process will be done using linear regression model. To increase the model accuracy, we transformed the data into a high-dimensional space using the Nystroem kernel approximation method. The results show a good prediction accuracy for both small and big houses. The mean absolute prediction error for the former is $3.24m^2$ and $4.1m^2$ for the latter.

## 1 data-set Description

The data was scraped from Immoscout24, the biggest real estate platform in Germany. Immoscout24 has listings for both rental properties and homes for sale, however, the data only contains offers for rental properties. The scraping process is described in this blog post and the corresponding code for scraping and minimal processing afterwards can be found in this GitHub repository. At a given time, all available offers were scraped from the site and saved. This process was repeated three times, so the data set contains offers from the dates 2018-09-22, 2019-05-10 and 2019-10-08.

The data set contains most of the important properties, such as living area size, the rent, both base rent as well as total rent (if applicable), the location (street and house number, if available, ZIP code and state), type of energy etc. for 268,850 different cases. It also has two variables containing longer free text descriptions: description with a text describing the offer and facilities describing all available facilities, newest renovation etc. The date column was added to give the time of scraping.

In the Table 1, we recall the full column details, provided in the data-set page in Kaggle.

| **Column Name** | **Missing** | **Column Description** |
|---|---|---|
| regio1 | 0 | Bundesland |
| serviceCharge | 6909 | aucilliary costs such as electricity or internet in € |
| heatingType | 44856 | Type of heating |
| telekomTvOffer | 32619 | Is payed TV included if so which offer |
| telekomHybridUploadSpeed | 223830 | how fast is the hybrid inter upload speed |
| newlyConst | 0 | is the building newly constructed |
| balcony | 0 | does the object have a balcony |

**Table 1 continued from previous page**

| Column Name | Missing | Column Description |
|---|---|---|
| picturecount | 0 | how many pictures were uploaded to the listing |
| pricetrend | 1832 | price trend as calculated by Immoscout |
| telekomUploadSpeed | 33358 | how fast is the internet upload speed |
| totalRent | 40517 | total rent (usually a sum of base rent, service charge and heating cost) |
| yearConstructed | 57045 | construction year |
| scoutId | 0 | immoscout Id |
| noParkSpaces | 175798 | number of parking spaces |
| firingTypes | 56964 | main energy sources, separated by colon |
| hasKitchen | 0 | has a kitchen |
| geo_bln | 0 | bundesland (state), same as regio1 |
| cellar | 0 | has a cellar |
| yearConstructedRange | 57045 | binned construction year, 1 to 9 |
| baseRent | 0 | base rent without electricity and heating |
| houseNumber | 71018 | house number |
| livingSpace | 0 | living space in sqm |
| geo_krs | 0 | district, above ZIP code |
| condition | 68489 | condition of the flat |
| interiorQual | 112665 | interior quality |
| petsAllowed | 114573 | are pets allowed, can be yes, no or negotiable |
| street | 0 | street name |
| streetPlain | 71013 | street name (plain, different formatting) |
| lift | 0 | is elevator available |
| baseRentRange | 0 | binned base rent, 1 to 9 |
| typeOfFlat | 36614 | type of flat |
| geo_plz | 0 | ZIP code |
| noRooms | 0 | number of rooms |
| thermalChar | 106506 | energy need in kWh/(m^2a), defines the energy efficiency class |
| floor | 51309 | which floor is the flat on |
| numberOfFloors | 97732 | number of floors in the building |
| noRoomsRange | 0 | binned number of rooms, 1 to 5 |
| garden | 0 | has a garden |
| livingSpaceRange | 0 | binned living space, 1 to 7 |
| regio2 | 0 | District or Kreis, same as geo krs |

**Table 1 continued from previous page**

| Column Name | Missing | Column Description |
|---|---|---|
| regio3 | 0 | City/town |
| description | 19747 | free text description of the object |
| facilities | 52924 | free text description about available facilities |
| heatingCosts | 183332 | monthly heating costs in € |
| energyEfficiencyClass | 191063 | energy efficiency class (based on binned thermalChar, deprecated since Feb 2020) |
| lastRefurbish | 188139 | year of last renovation |
| electricityBasePrice | 222004 | monthly base price for electricity in € (deprecated since Feb 2020) |
| electricityKwhPrice | 222004 | electricity price per kwh (deprecated since Feb 2020) |
| date | 0 | time of scraping |

Table 1: Column descriptions

As you can see, some of the scraped features are completely unrelated to our regression task. For example, `scoutId`, `condition`,`street`, `facilities`, `description` and `date` are among the redundant columns.

The next important point is the missing values of the data-set. The Table 1 also report the count of the number of missing values for each feature. There are different methods to handle the missing values in the data-set. dropping rows or columns, imputing them with a constant value or the most frequent value or even filling them with the most similar cases having the desired column. In this exercise we use a more efficient method based on the data-set properties. Before doing it, we should done some preprocessing steps on the data-set. Here we show the distribution of the numeric variables using a box plot. Figure 1 shows this box plot for `floor`, `noRooms`, `livingSpace`, `heatingCosts`, `totalRent` and`baseRent`. You can see there are many outliers in the data. Removing this cases from the data reduces the data-set size to approximately 255000 rows.

The next important information which can help the regression model, is the **livingSpaceRange** feature. This feature summarizes the `livingSpace` feature into 7 different categories. The first category contains the smallest houses while the category 7 includes the biggest living space houses.

From now we preprocess the data based on this feature. Here we explain the outlier detection for the first living space range. The process is same for other ranges, except the bound constraints. Since there are many missing values in columns, we fill them based on their living space range. In other words, we fill the missing values in `heatingCosts` feature, using the values corresponding to the first living space range. This will help filling them with a more appropriate values.

As described in 1, the `totalRent` feature is sum of the base rent, heating cost and service charge. Also,

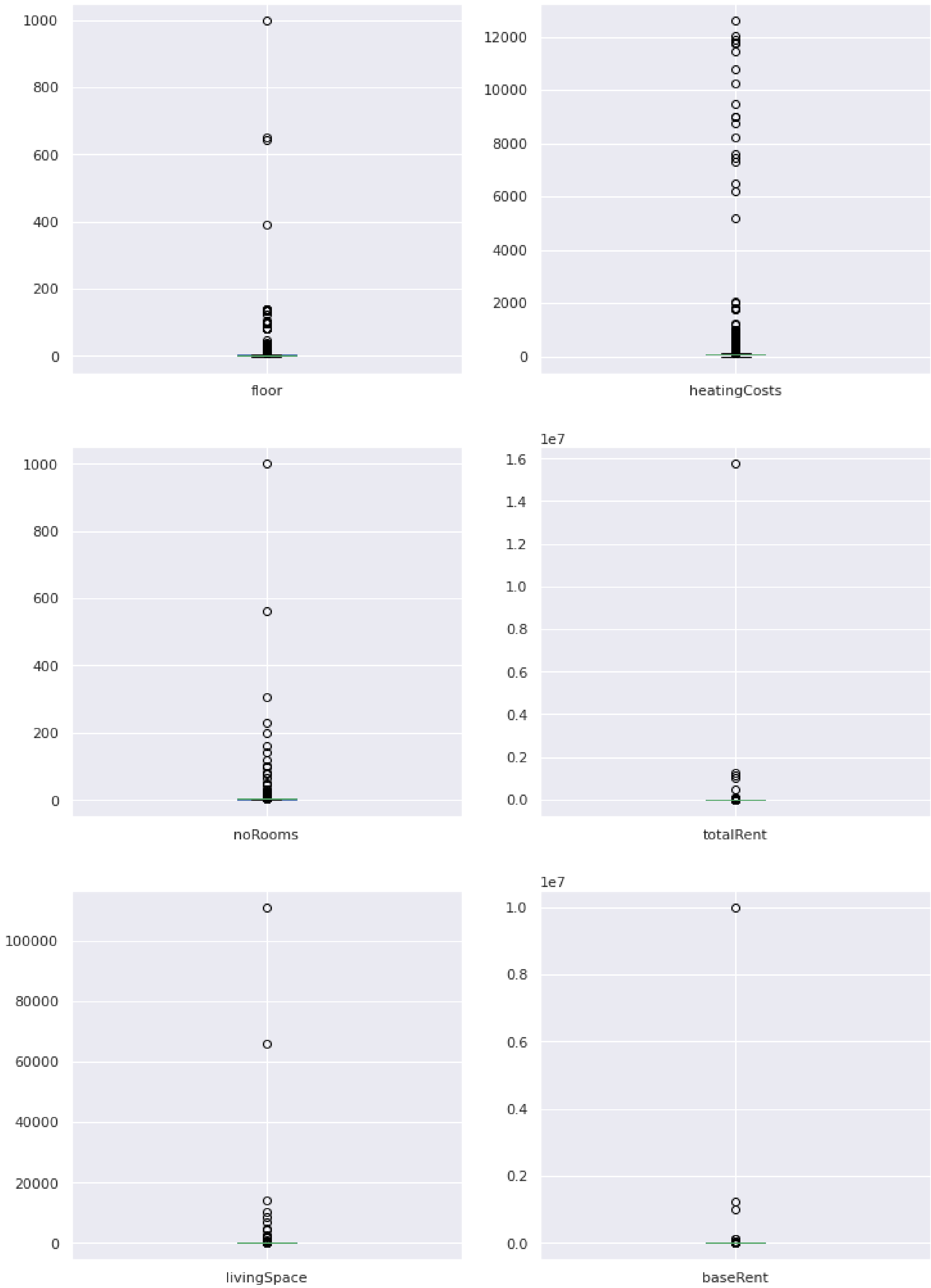


Figure 1: box plot for some of the data-set features

| regio1 | | heatingType | | petsAllowed | |
|---|---|---|---|---|---|
| Category | Count | Category | Count | Category | Count |
| Nordrhein_Westfalen | 31480 | central_heating | 77802 | negotiable | 94758 |
| Sachsen | 28490 | district_heating | 13262 | no | 20242 |
| Sachsen_Anhalt | 11583 | gas_heating | 9498 | yes | 5080 |
| Niedersachsen | 7188 | self_contained_central_heating | 7900 | | |
| Bayern | 6229 | floor_heating | 5656 | | |
| Hessen | 5138 | oil_heating | 2450 | | |
| Thüringen | 4883 | heat_pump | 1064 | | |
| Baden_Württemberg | 4669 | combined_heat_and_power_plant | 838 | | |
| Brandenburg | 4107 | night_storage_heater | 678 | | |
| Mecklenburg_Vorpommern | 4068 | wood_pellet_heating | 399 | | |
| Schleswig_Holstein | 3506 | electric_heating | 336 | | |
| Rheinland_Pfalz | 3278 | stove_heating | 125 | | |
| Berlin | 2672 | solar_heating | 72 | | |
| Bremen | 1336 | | | | |
| Hamburg | 840 | | | | |
| Saarland | 613 | | | | |

Table 2: The number of rows in each category

we fill the missing values in `totalRent` feature by the summation of this three feature, based on the living space range.

Now, it's time to remove outliers in this living space range. Again, we draw the box plot for some of the numerical features. Figure 2 shows these plots.

Applying the outlier removal constraints, discards approximately 15000 rows of data. We do the same process for other ranges 2-7. Our analysis showed that the last house range are very difficult to approximation, i.e. they are luxury houses, so we removed the last range. After doing the purity process, we lost about 140000 rows.

The next preprocessing step is to remove the less frequent categories in categorical features. To find them we count unique values in `regio1`, `heatingType`, `petsAllowed` and reported in 2.

Here, we remove 3 less often categories of `regio1`. For the `heatingType` feature we only use the 5 most used categories. Negotiable option in `petsAllowed` feature is mapped to yes, while the missing values considered as no.

For the final preprocessing steps, we round the target feature living space with one digits of accuracy and encode the categorical features using one hot encoding. The data is now ready to be used for our

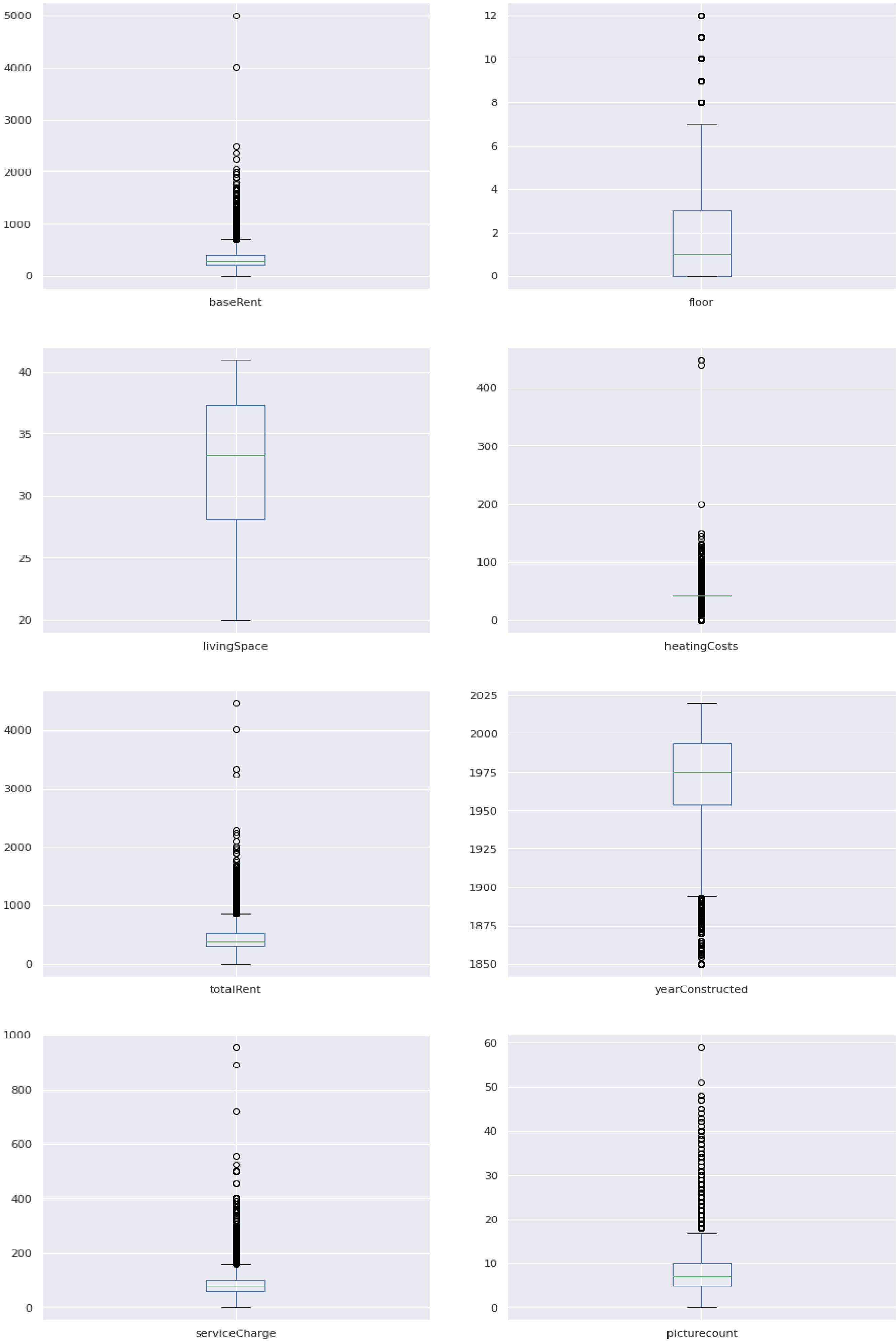


Figure 2: box plot for some of the data-set features

regression task. In the next section we analyse the features using some statistical tests.

# 2 Data analysis

In the previous section we done some preprocessing tasks to remove outliers from the given data-set. In this section, we analyse the final data-set by some statistical measures and tests.

## 2.1 Correlation analysis

For the first test, we compute the correlation matrix of the numberical features of the data-set. The features are `livingSpaceRange`, `heatingCosts`, `totalRent`, `baseRentRange`, `baseRent`, `noRooms`, `noRoomsRange`, `serviceCharge`, `picturecount`, `balcony`, `regio1`, `heatingType`, `petsAllowed` and `livingSpace`. The figure 3 shows this matrix.

As you can see in the plot, the features `totalRent`, `baseRentRange` and `baseRent` are highly correlated. This is not a surprising fact because the base rent range is computed directly based on base rent and the total rent always includes the base rent features. `noRooms` and `noRoomsRange` are highly correlated which is obvious. Finally, the living space and the living space range feature have a high correlation which is directly consequence of categorizing it to 7 bins.

## 2.2 Hypothesis testing

Here we use the ANOVA test to checking whether the two features have the same distribution or not.The ANOVA, tests whether the means of two or more independent samples are significantly different. It has the following assumptions on the data

- Observations in each sample are independent and identically distributed (iid).
- Observations in each sample are normally distributed.
- Observations in each sample have the same variance.

The interpretation of the method is as following:

- H0: the means of the samples are equal.
- H1: one or more of the means of the samples are unequal.

In the table 3 we reported this test on all combinations of the columns of length 2.

# 3 Building the model

The purpose of the exercise is to build a linear regression model to predict the living space area. Given the training data $X$ and $y$, this model defines it's predictor function as

$$y(x) = w^T x + b, \tag{1}$$

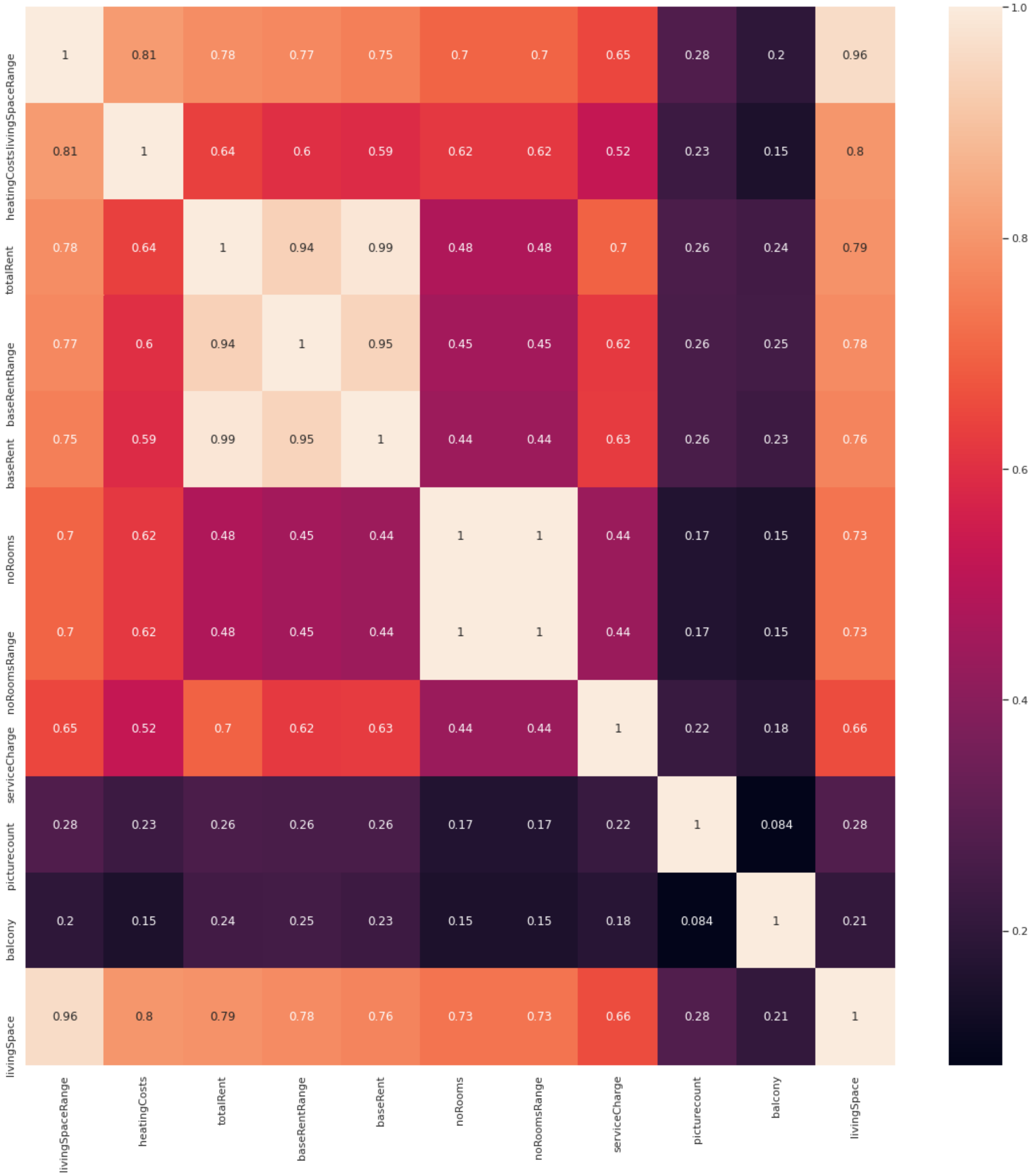


Figure 3: box plot for some of the data-set features

| Feature 1 | Feature 2 | ANOVA test |
|---|---|---|
| livingSpaceRange | noRooms | + |
| livingSpaceRange | totalRent | + |
| livingSpaceRange | serviceCharge | + |
| livingSpaceRange | livingSpace | + |
| livingSpaceRange | heatingCosts | + |
| livingSpaceRange | balcony | - |
| livingSpaceRange | baseRent | + |
| noRooms | totalRent | + |
| noRooms | serviceCharge | + |
| noRooms | livingSpace | + |
| noRooms | heatingCosts | + |
| noRooms | balcony | + |
| noRooms | baseRent | - |
| totalRent | serviceCharge | + |
| totalRent | livingSpace | + |
| totalRent | heatingCosts | + |
| totalRent | balcony | + |
| totalRent | baseRent | + |
| serviceCharge | livingSpace | + |
| serviceCharge | heatingCosts | + |
| serviceCharge | balcony | + |
| serviceCharge | baseRent | + |
| livingSpace | heatingCosts | + |
| livingSpace | balcony | + |
| livingSpace | baseRent | + |
| heatingCosts | balcony | + |
| heatingCosts | baseRent | + |
| balcony | baseRent | + |

Table 3: ANOVA test results. plus means the columns have same distributions, minus means different distributions

where $w$ and $b$ are the unknown coefficients. If we define the loss function as

$$loss(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2, \tag{2}$$

the solution of the model can be found by solving the following linear system of equations:

$$\begin{bmatrix} \mathbf{1} & X \end{bmatrix} \begin{bmatrix} b \\ w \end{bmatrix} = y \tag{3}$$

# 4 Fitting the model

Before fitting the model, we split the data-set into two categories: small and big houses. Small cases are those that their living space range are 1 or 2. every house with higher living space range will be considered as a big house. This process helps us to find a more accurate model for each case. It's worth to note that we can use 6 different models for each living space range but it has more computational cost which may not be efficient in some cases.

Now, it's time to fit the model on these sub-data-sets. Before doing this, we transform the raw data into a high dimensional space using the Nystroem kernel approximation method. This method approximates a kernel map using a subset of the data. Here we used a radial basis function as the kernel of the method. The RBFs hyper-parameter $\gamma$ for both model is found by trial and test. Using this method, we create 1500 new features replacing the original 30 features. After the feature expansion step, we change the scale of data using sklearn's StandardScaler class.

# 5 Results

To show the accuracy of the method we split each sub-data-set into two train and test sets with the proportion of 75%-25%.In the table 4, we compared the approximation error with different loss functions. The results show that the proposed method can approximated the small house living spaces' with mean error of $3.2m^2$ and $4.1m^2$ for big houses. To show the accuracy of the model, we compute the following criteria:

$$accuracy = \frac{\text{no instances with true labels}}{\text{no total instances}} \tag{4}$$

in which a prediction label is true if it falls in the interval $[0.85 * y_{true}, 1.15 * y_{true}]$. The last column of table 4 reports the accuracy of the model.

Also, to show the error distribution, we drew box plot of residual errors for both small and big houses. The plots are drawn in the figure 4.

| House Type | Mean Absolute Error | Mean Squared Error | Mean Epsilon Insensitive Error | Accuracy |
|---|---|---|---|---|
| Small | 3.24 | 17.38 | 2.35 | 88.48% |
| Big | 4.1 | 24.83 | 3.17 | 99.03% |
| Mean | 3.67 | 21.11 | 2.76 | 93.76% |
| All | 3.81 | 22.28 | 2.89 | 95.01% |

Table 4: Approximation Error of living space feature

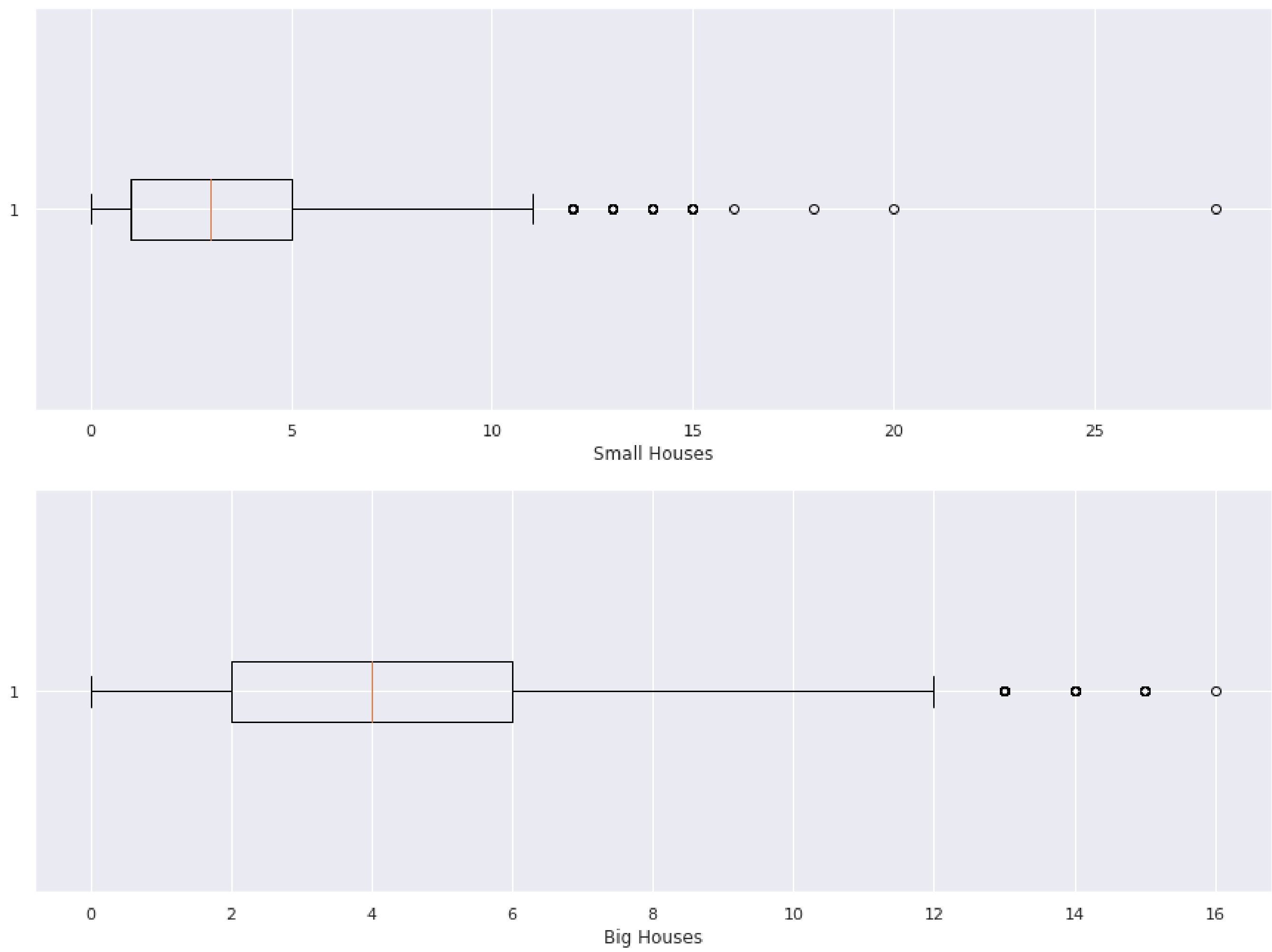


Figure 4: box plot for residual error of prediction for small and big houses

# 6 Conclusion

In this exercise we proposed a linear regression model equipped with a nonlinear feature expander, for predicting the living space area of different houses. Since our implementation of LinearRegression uses scipy's `lstsq` function to solve a linear least squares problem, the time complexity of our model and the sklearn's LinearRegression class are approximately equal.