Predicting Prices in German Real Estate Market

Hanna Martens (7266768)
Konrad Wartke (6481209)

June 25, 2020

1 Original Data Sources

We started with two datasets from Kaggle, one with rent price and one with house prices for the German Real Estate market. The Rent data was scraped from Immoscout24 on three different dates (22.09.2018, 10.05.2019, 08.10.2019). The house prices were also taken from Immoscout24. While no date is given for the creation of this dataset, it is most likely from novembre 2017. The name of the house price dataset is misleading, it features houses from all over Germany instead of Hesse only.

2 Cleanup and Merging

Since both datasets are compiled from the same original source, they share most of the features. For several features, the names are in German in one dataset, but English the other, so we had to translate the names.
The most important feature is the price, which is influenced by most features. For example, all other things equal, having a garden increases the price. To eliminate the influence of several features, we trained a linear regression model. The feature weight is then used to calculate a clean price as if the real estate had no extras. This price can be negative, which means money has to be invested to bring the house into a standard state.
At the end, all rent and buying prices where grouped by the zip code and the average rent and buying price per square metre is calculated. Also included are the percentages of real estate units for which a binary feature is true.

3 Prediction

3.1 Simple Linear Regression

The implemented function predicts the target values for any given features, where the column_y is predicted based on column_x. To implement the regression model we used

Scikit-learn a free software machine learning library for python. The datasets that were used for the linear regression model are cleaned_rent and cleaned_house_prices.

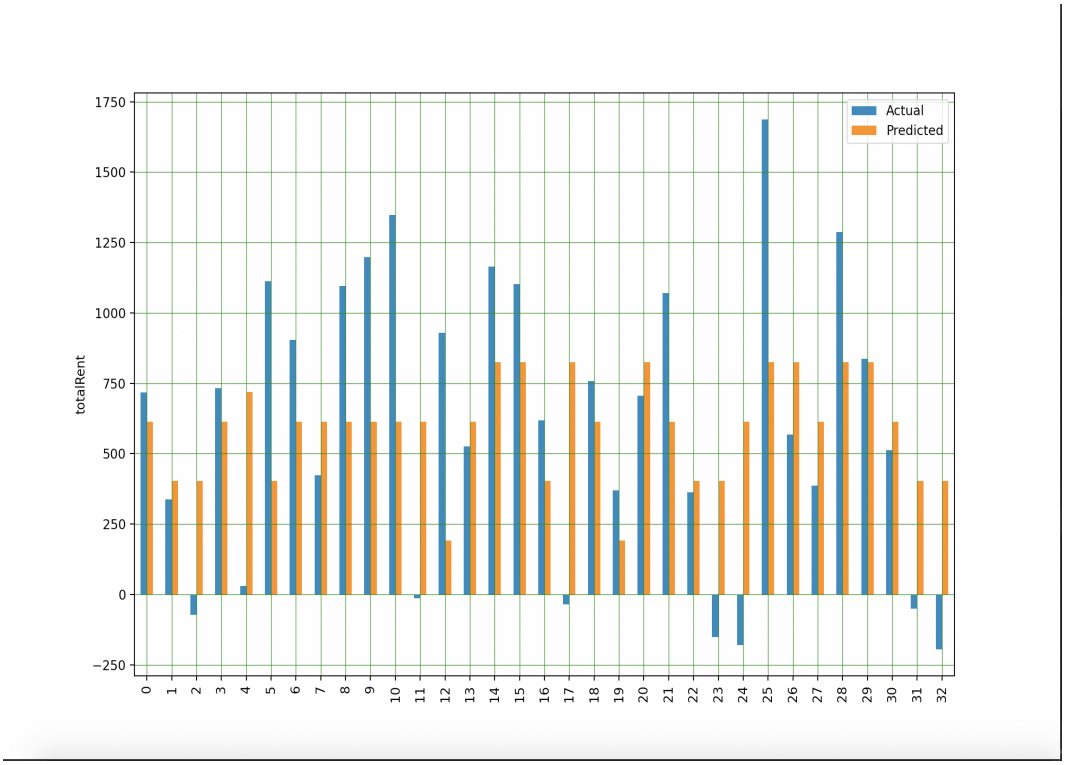

Figure 1: Number of rooms and total rent

Evaluation

The evaluation observed here by way of example is based on the prediction of the rental price based on the number of rooms. As seen in figure 1) the algorithm performs quite poorly. While a few predicted values closely match the actual values, other values are quite off. Exceptions like negative prices for houses which are in need of financial investment or generally missing values may have caused problems here. These outliers in the values could haven been detected and dealt with even more carefully. Rows with incorrect data, such as NAN or infinity values, were deleted for this project, which reduced the overall size of the data set. One approach for improvement would be to either replace the faulty data e.g. with average values or to integrate more data to extend the data set. A series of experiments with possible combination pairs could be another approach to find two features that have a stronger connection to improve the result.

3.2 Multiple Linear Regression

Another way to improve the results from the last section is multiple linear regression. We have again set the rental price as prediciton target and this time we include several different features from the data set to train the model.

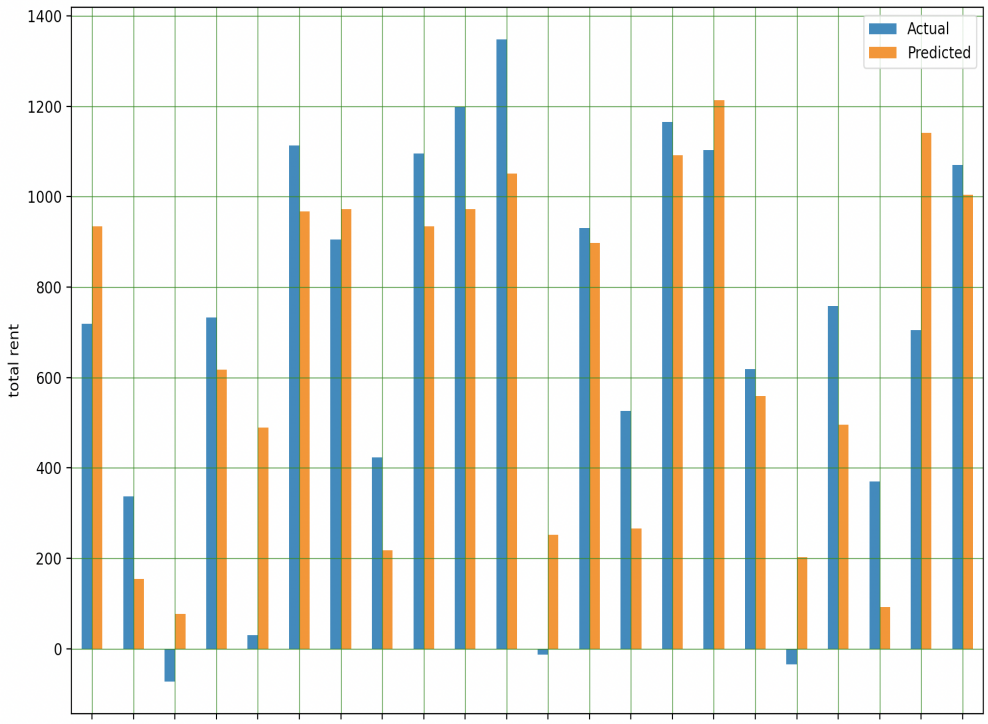

Figure 2: Multiple features and total rent

Evaluation

The quality of the algorithms can also be assessed directly by the Scikit-learn metrics library.

Metric	simple regression	multiple regression
Mean Absolute Error	389.5304	195.7756
Mean Squared Error	219942.1611	50782.9660
Root Mean Squared Error	468.9799	225.3507

Mean squared error (MSE) - takes the mean squared difference between the target and predicted values Mean absolute error (MAE) - calculates prediction errors (actual value minus predicted value) for each row of data and find the mean of all absolute prediction errors. Root Mean Squared Error (RMSD) - is the standard deviation of the prediction errors.

	Coefficient
Wohnflaeche in m^2	6.376861
Zimmer	65.611050
baseRentRange	147.183205
livingSpaceRange	-35.609006
yearConstructed	-2.467776
numberOfFloors	28.994673.

The regression model finds the best fitting coefficients for all attributes and weights them accordingly. Here we see that, for example, the year of construction has no influence on the rent, while the rooms, number of square meters and baseRentRage are decisive. Overall it seems like including several features instead of just looking at two features at a time improved the performance of our model.

3.3 KNN

The k nearest neighbor (KNN) algorithm is a type of supervised machine learning and is used for both classification and regression problems. We used the algorithm to predict the average rent per square meter based on all other values contained in the merged dataset. For our regression problem we used the Scikit-learn knn model to implement the algorithm for our dataset. Since there is no optimal number of neighbors that suits all kind of data sets fitting the model was mostly trial and error. For our dataset the optimal number of neighbors is five, while the train set contains 90% of the data and the test set accordingly contains 10%.

Metric	KNN Model
Mean Absolute Error	3.160
Mean Squared Error	89.093
Root Mean Squared Error	9.438

4 Results

If we compare the algorithms with each other we can see a steady improvement of the values. While multiple linear regression is much better than single linear regression, the knn algorithm finds the best prediction. Obviously the lineare regression models and the knn model used different sets of data thus are not fully comparable. With more time, the knn model could be trained with the data used in linear regression and a final statement could be made about which algorithm makes the best prediction for our data set.

5 Discussion