HOUSING PRICE PREDICTION

1. Problem Formulation

The question I try to resolve using machine learning is: Given the basic information about an apartment (i.e., date of buying, postal code, number of rooms), what is the approximate price of that apartment?

The datapoints in the problem are the apartments. The apartments are characterized by several traits: the time when the apartment price was estimated, the postal code, and the number of rooms. Since all these three traits can be obtained easily, they will potentially be used as features for one datapoint. The label is the price of the apartments at the corresponding time.

To be more specific about the type of data, the time when the apartment price was estimated is in "string" format, which is a quarter of a year (for example, 2010Q1, which means the first quarter of 2010). The postal code is also "string" of number (for example, 00100). The number of rooms in the original dataset is also "string" (for example, Blocks of flats, one-room flat). The price of the apartment, which is the label, is integer.

2. Methods

2.1. Dataset

2.1.1. Data Preprocessing

For the data of this project, there is a record of the average prices of old dwellings in housing companies and numbers of transactions by postal code area from the first quarter of 2010 until the last quarter of 2021. Since my interest is solely in the price of the apartments in capital region (i.e., Helsinki, Espoo, Vantaa, Kauniainen), the number of transactions and information about other postal code areas are not included in my dataset. The original data can be downloaded from the database of Statistics Finland [1].

The original dataset has 23607 rows, and 16127 out of 23607 rows do not have information about the price. It is indeed a large portion of the whole dataset but if these rows are not used, there are still around 7000 rows with adequate information about features and labels. Eventually, the cleaned dataset contains 7480 datapoints with full information, which is sufficiently enough for the ML methods used.

The next step is converting the information of the features into some types of integer value that can be used for NumPy [2] arrays and for the ML model. For the time when the apartment price was estimated, the information is transformed into an integer which indicates the number of quarters counted from the beginning of 2010 (for example, "2010Q1" will be 1, "2011Q1" will be 5). Both the postal code and the number of rooms are converted into their corresponding integers (for example, "00100" changes to 100; "Blocks of flats, one-room flat" turns into 1). More details about data processing can be found in the "Data Processing" file [3].

2.1.2. Data Visualization and Exploration

Intuitively, a house price can be determined by various factors: size, location, number of bedrooms, construction year, etc. In the original and processed dataset, there are only three properties of the apartments: the time when the apartment price was estimated, the postal code, and the number of rooms. These properties can highly be potential features.

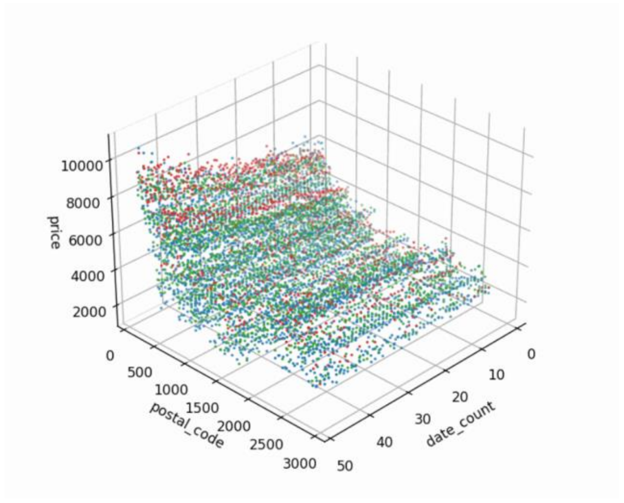


Figure 1 – Data Scatterplot
(Red: 1 room – Green: 2 rooms – Blue: 3 rooms)

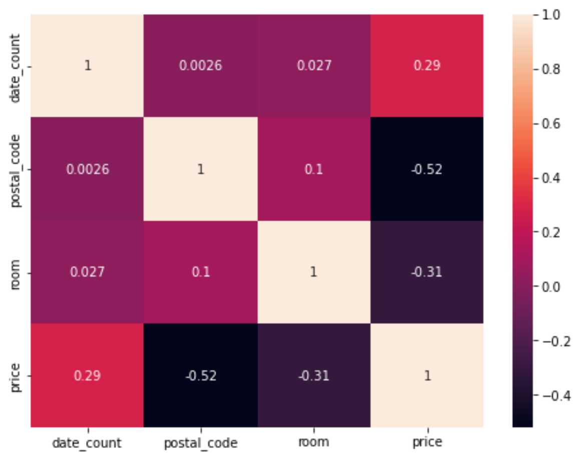


Figure 2 – Correlation Matrix between features and label

After visualizing the data with scatterplot (figure 1), as well as analyzing the correlation matrix (figure 2), all of three properties show some correlation with the house price. As a matter of fact, these properties are easy to obtain, so that it makes sense they are chosen as the features. Therefore, these three properties are chosen as the features for the ML model. More details about data visualization and exploration can be found in the corresponding file [4].

2.2. Polynomial Regression Model

2.2.1. Model Training + Validation

According to the data scatterplot, one can see that the house price follows increasing curved pattern. A chosen ML model is expected to show the relationship between the features and the labels in this pattern. Polynomial regression is a simple approach, and it can effectively show this curvilinear relationship between features and label. Therefore, polynomials with order from 2 to 10 are used to train and predict the price.

The datapoints in the dataset are randomly divided into training, validation, and test sets with the ratio of 60%, 20%, 20% respectively. The motivation of using a single split is that the dataset has a large amount of data (7480), and this method will not increase the training time or require heavy computation resource like some other methods (for example, k-fold). Ratio 60-20-20 is used as it is a fairly common ratio in ML problems [5], and 60% of the dataset provides enough data for training.

The hypothesis space of a polynomial regression model is constituted by polynomial maps.

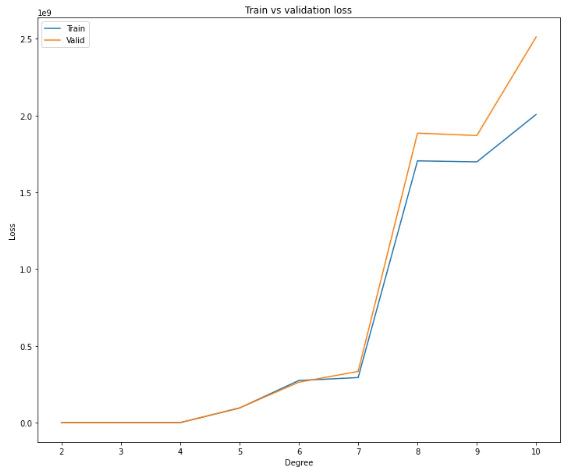
$$\mathcal{H}_{\text{poly}}^{(n)} = \{h^{(\mathrm{w})} : \mathrm{R} \rightarrow \mathrm{R} : h^{(\mathrm{w})}(x) = \sum_{r=1}^{n} w_r\, x^{r-1} \text{ , with some } w = (w_1, \ldots, w_n)^T \in \mathrm{R}^n \}.$$ [6]

Different orders of polynomials (degree = 2, . . ., 10) are fitted to the training set by minimizing the mean squared loss [6]. The mean squared loss is calculated by examining all datapoints and obtaining the average squared difference between the predicted labels and the real labels.

$$\mathrm{MSE} = \frac{1}{m} \sum_{i=1}^{m} (y^{(i)} - h^{(w)}(x^{(i)}))^2 .$$ [6]

This loss function is used for the regression fitting because it is normally used in polynomial regression [6], and it is also the default option in the used scikit-learn package [7].

After training the model, the mean squared loss calculated for training and validation data sets with different models is shown below in figure 3.



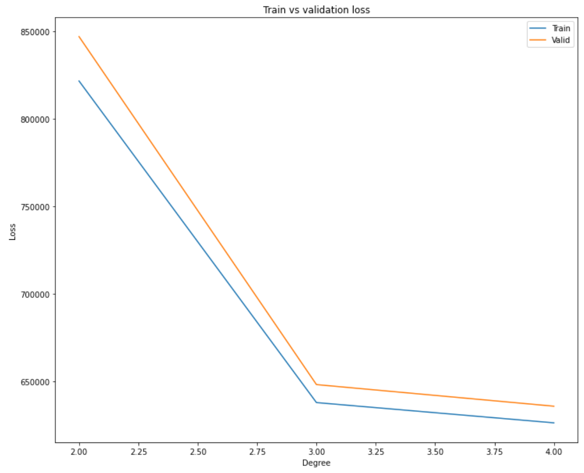


Figure 3: Train vs validation loss plot

The MSE of the models are high (hundreds of thousands), but it is not a huge problem since the features and the labels take large values (thousands). One can observe that 4th order polynomial has lowest loss, and for higher order polynomials, the loss starts to increase. Based on the result, polynomial regression model with degree 4 is the best model, and thus being chosen as the final model. The model shows a stable performance, and no overfitting or underfitting is visible.

To justify the features choice again, other polynomial regression models with only one or two features out of three are also analyzed. The training and validation loss of these models are much higher than the final model (4th order polynomial with three features). This is another concrete reason to support the features choice at the beginning. More information about the comparisons can be found in “Polynomial Regression” file [8].

2.2.2. Final Model Test

The mean squared error for the final model and the test data is 648508.76675.

3. Code file

All the code used to produce the results in the report can be found in this GitHub repository:

https://github.com/tamdnguyen/House-Price-Prediction

- The raw data and the processed data can be found directly through the link.
- The data preparation related files can be found in the folder “Data Preparation”.
- The data visualization and the code of the Polynomial Regression method can be found in “Model 1_Polynomial Regression” folder

4. References

[1] https://pxnet2.stat.fi/PXWeb/pxweb/en/StatFin/StatFin__asu__ashi__nj/statfin_ashi_pxt_112p.px/

[2] NumPy: https://numpy.org/

[3] Data Processing file: https://github.com/tamdnguyen/House-Price-Prediction/blob/main/Data%20Preparation/Data%20Processing.ipynb

[4] Data Visualization and Exploration file: https://github.com/tamdnguyen/House-Price-Prediction/blob/main/Model%201_Polynomial%20Regression/Data%20Visulization%20and%20Exploration.ipynb

[5] Train-Val-Test split ratio: https://glassboxmedicine.com/2019/09/15/best-use-of-train-val-test-splits-with-tips-for-medical-data/

[6] Course book, chapter 3.2

[7] scikit-learn: https://scikit-learn.org/stable/

[8] Polynomial Regression file: https://github.com/tamdnguyen/House-Price-Prediction/blob/main/Model%201_Polynomial%20Regression/Polynomial%20Regression.ipynb