QBUS2810 Assignment 2

zeemaBlue 2020-11-17

Task A House Price Prediction

We have dataset that aach row in the data set describes the characteristics of the house. Our goal is to predict the sales price based on characteristics of houses.

The evaluation model is based on the root mean square error (RMSE) between the sales price predicted by the model and the actual sales price. Convert RMSE to a logarithmic scale to ensure that the error in predicting expensive houses and cheap houses has the same score impact. Models: each cross-validation fits many models (including lasso, ridge, svr, ker, ela, bay, etc.) All trained models overfit the training data to varying degrees. Therefore, in order to make the final prediction, I mixed their predictions together to get a more reliable prediction, by using Stacking.

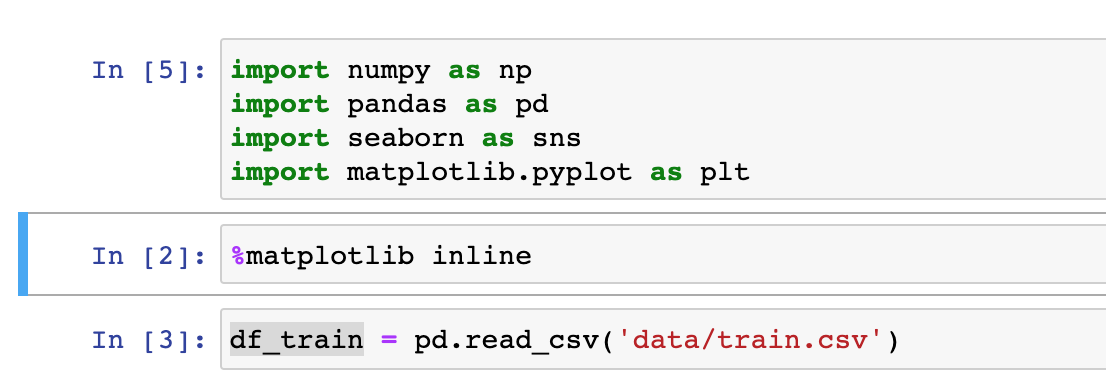
In order to improve RMSE, I use following methods:

Feature Engineering: Mainly assign values to discrete variables, feature combination and PCA

Model fusion: mainly weighted average and stacking

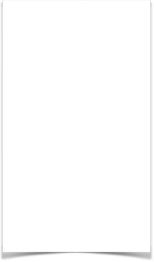
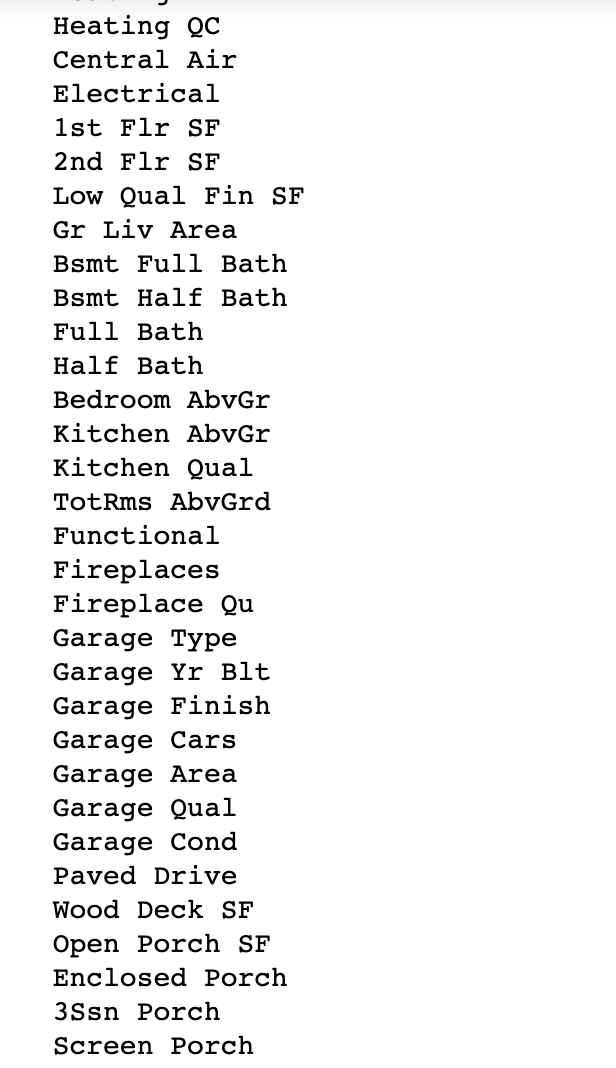
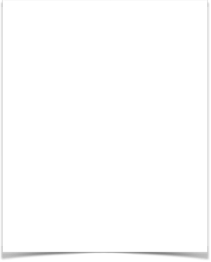
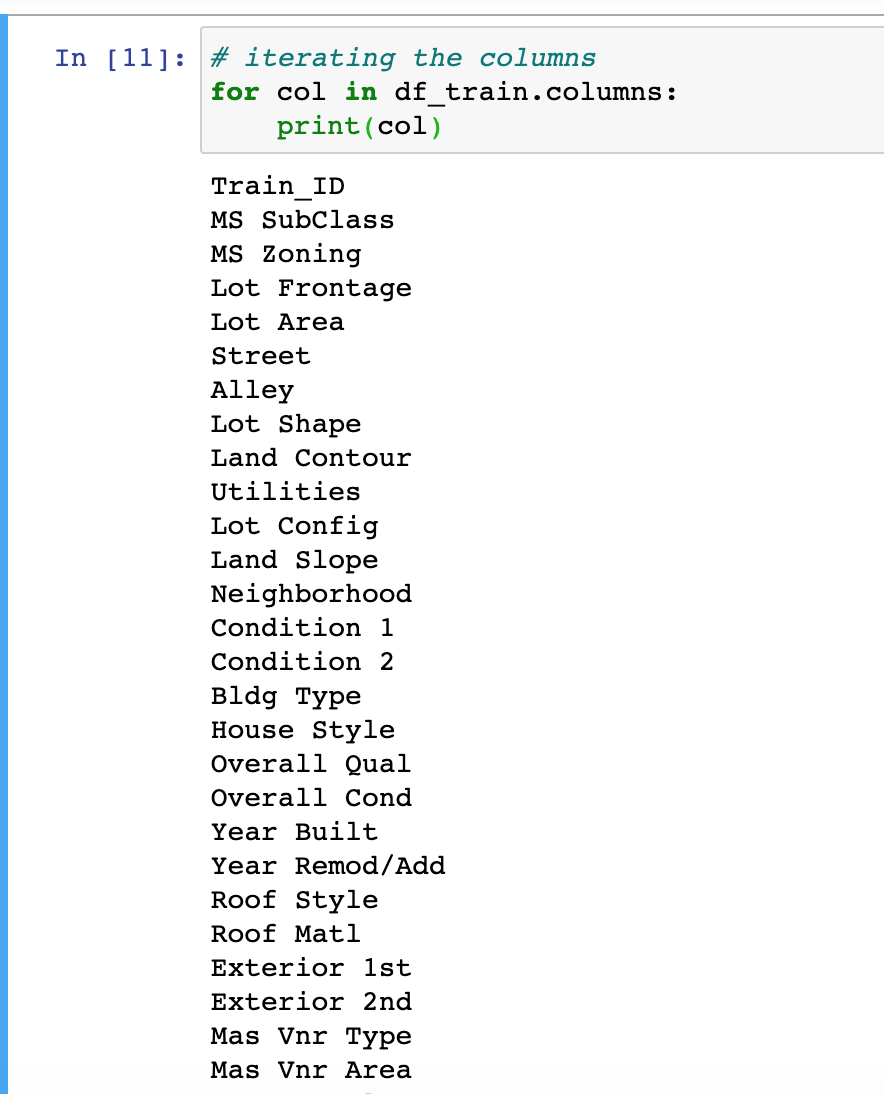
**Import necessary libraries**

We are using numpy,pandas, matplotlib.pyplot, seaborn… this libs to explore What attributes and data are needed for our predict.

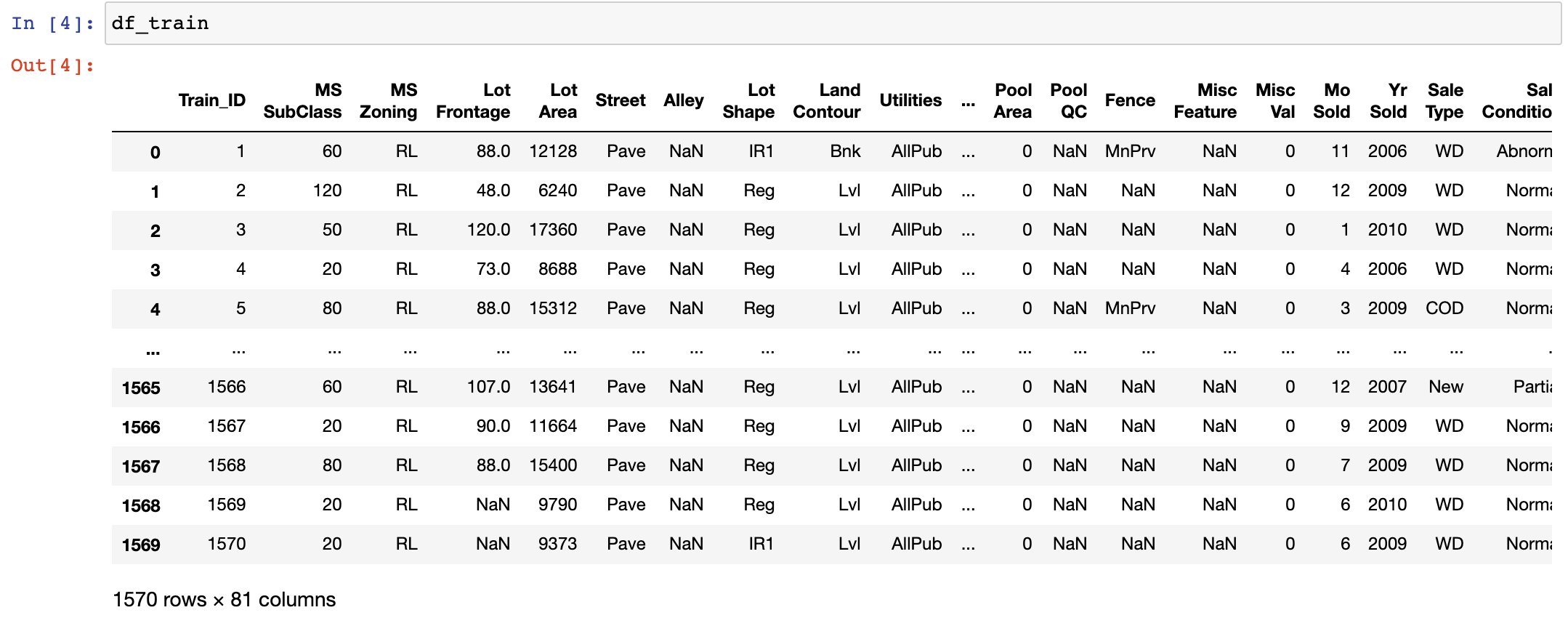


**Import Data**

We explore all column names , read Kaggle\_House\_Data\_Description’s details and explainations.

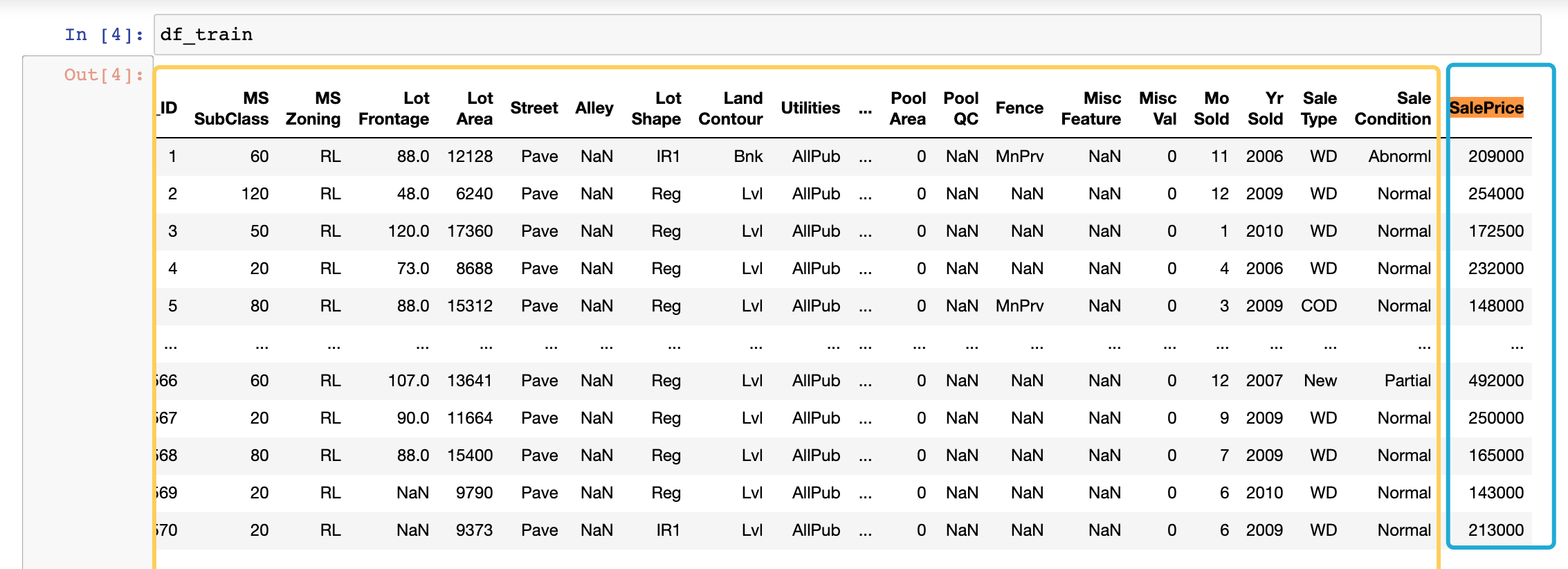


Get all column names



Get data shape and some records

A quick and simple look at the values and column names of the 81 columns, we can know that there are 80 features, including 1 predicted label, and 79 feature labels. The specific classification is as follows



Carefully read all the column names as features and the interpretation of the column data.

For example:

MSSubClass: Identifies the type of dwelling involved in the sale.

MSZoning: Identifies the general zoning classification of the sale.

LotFrontage: Linear feet of street connected to property

LotArea: Lot size in square feet

Street: Type of road access to property

Alley: Type of alley access to property

…

Fence: Fence quality

MiscFeature: Miscellaneous feature not covered in other categories

MiscVal: $Value of miscellaneous feature

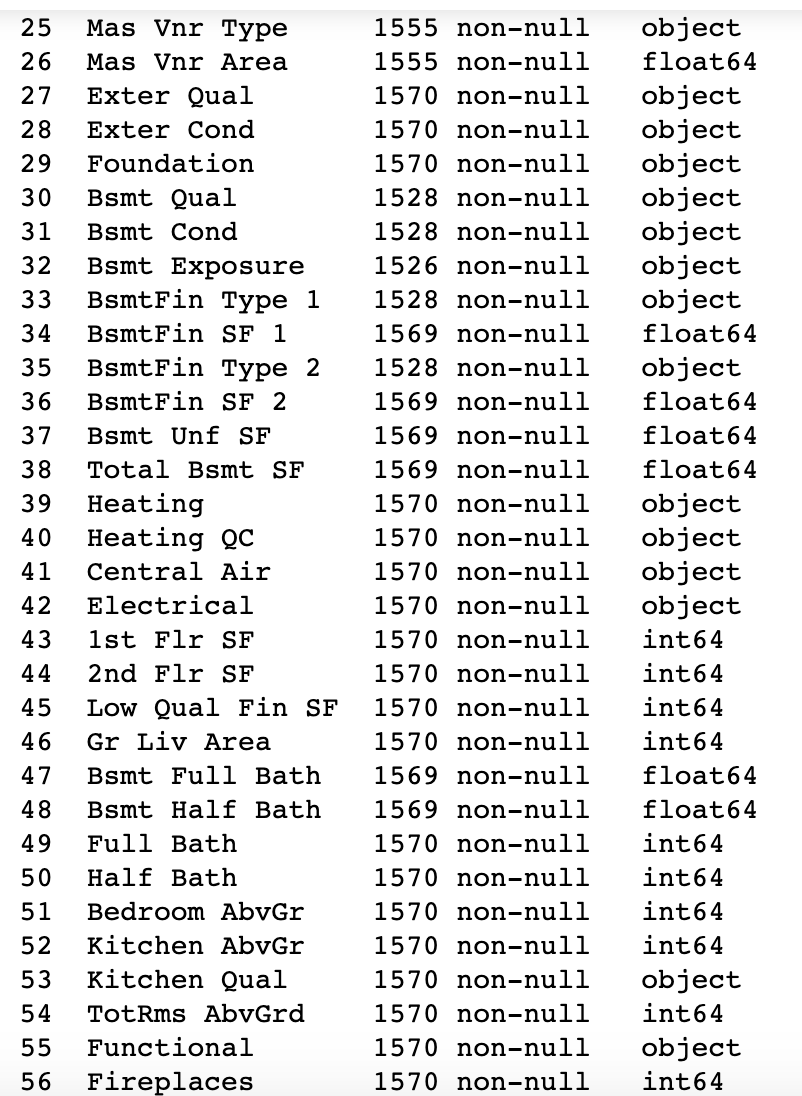
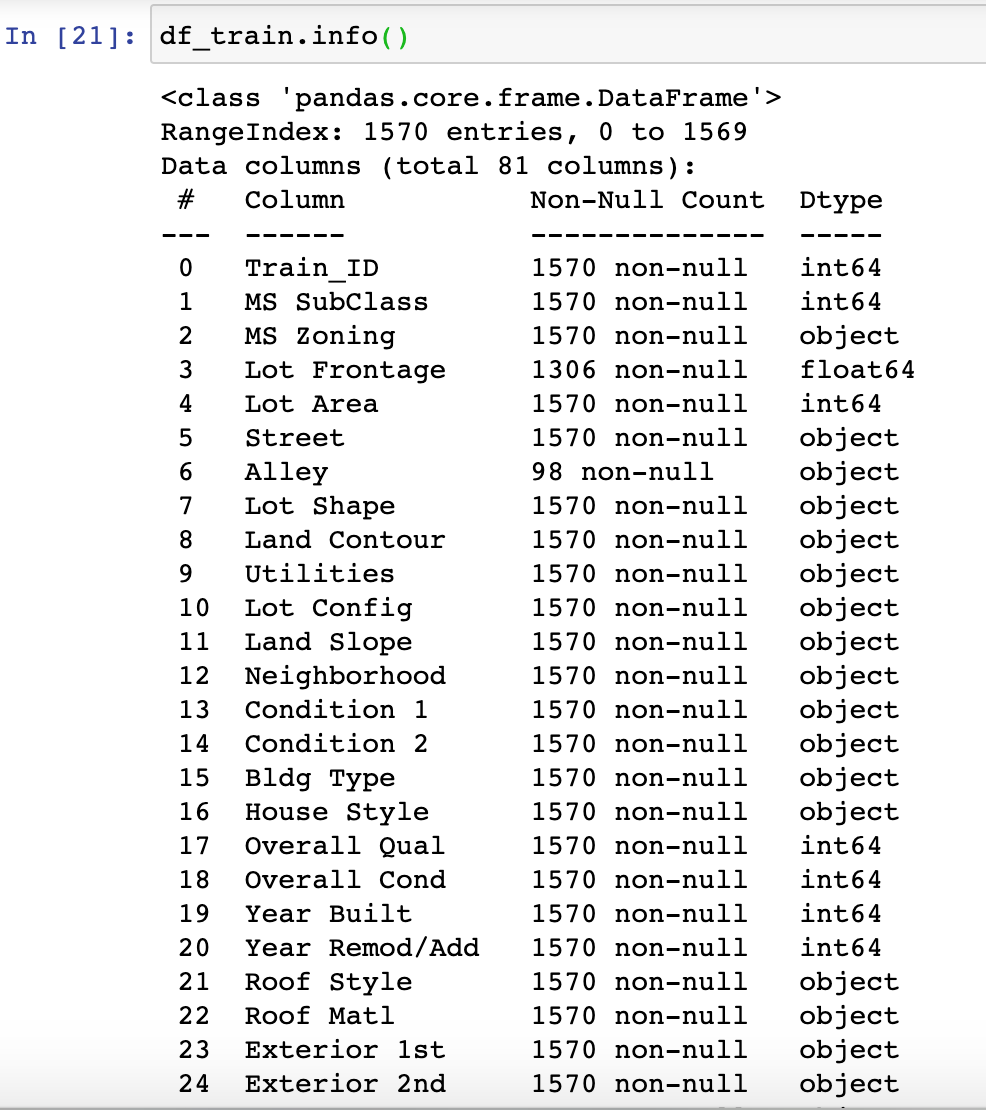
MoSold: Month Sold (MM)

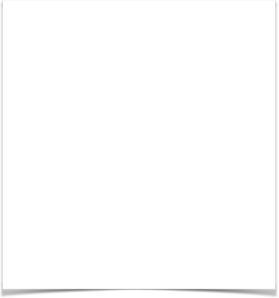
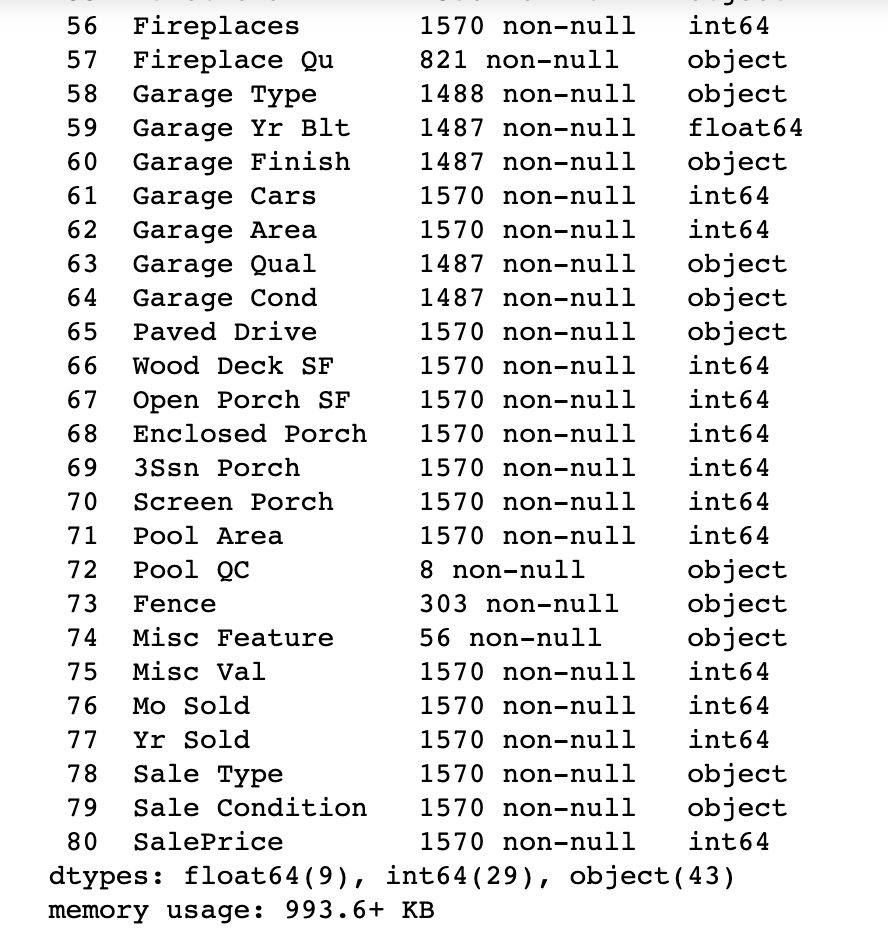
YrSold: Year Sold (YYYY)

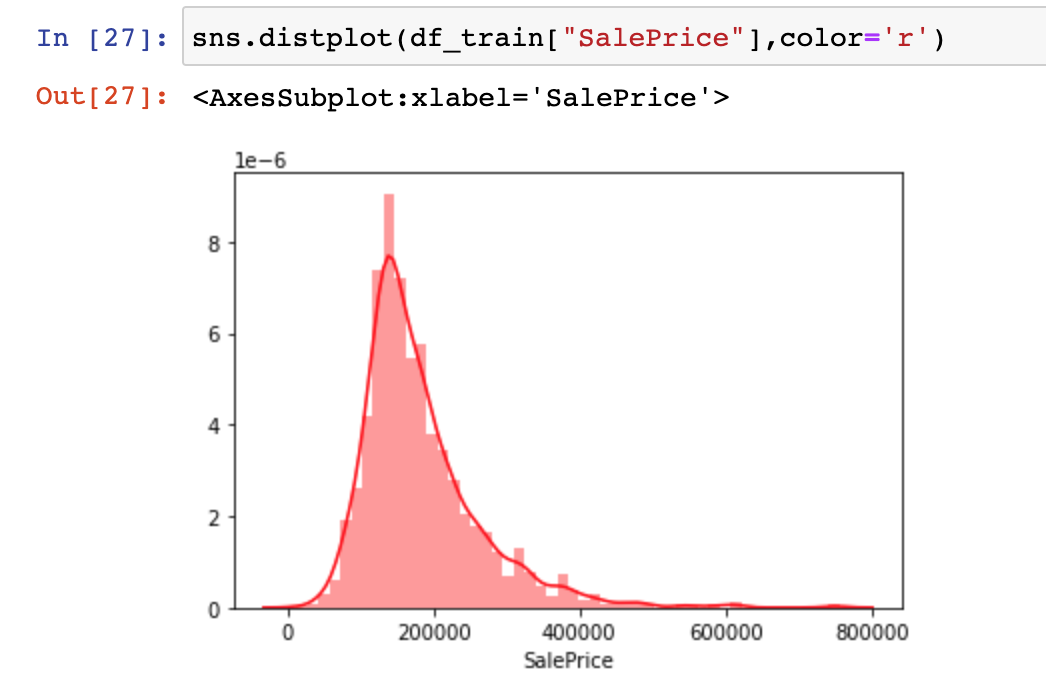
SaleType: Type of sale

SaleCondition: Condition of sale

**Look at the statistics**

View the default data of each data column in the data set, and see which columns have too few data and need to be removed 



Through visualization, analyze the statistical law of the SalePrice itself

We need to find the most relevant columns to salesprice, we do test, first find top 20, then top 10

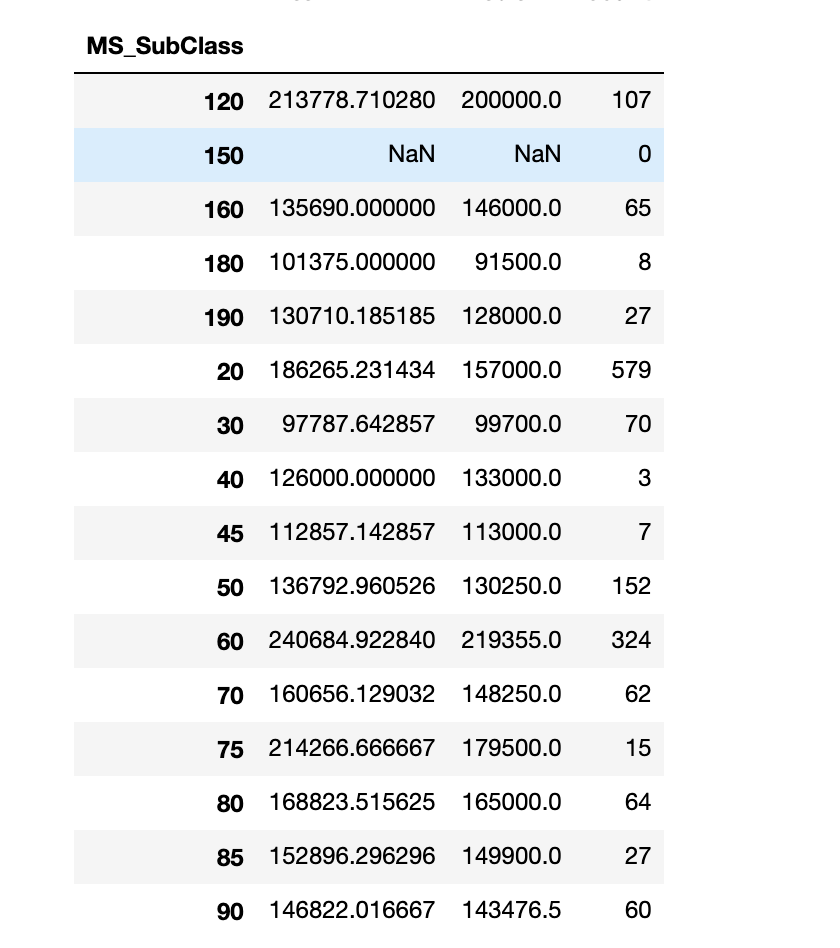
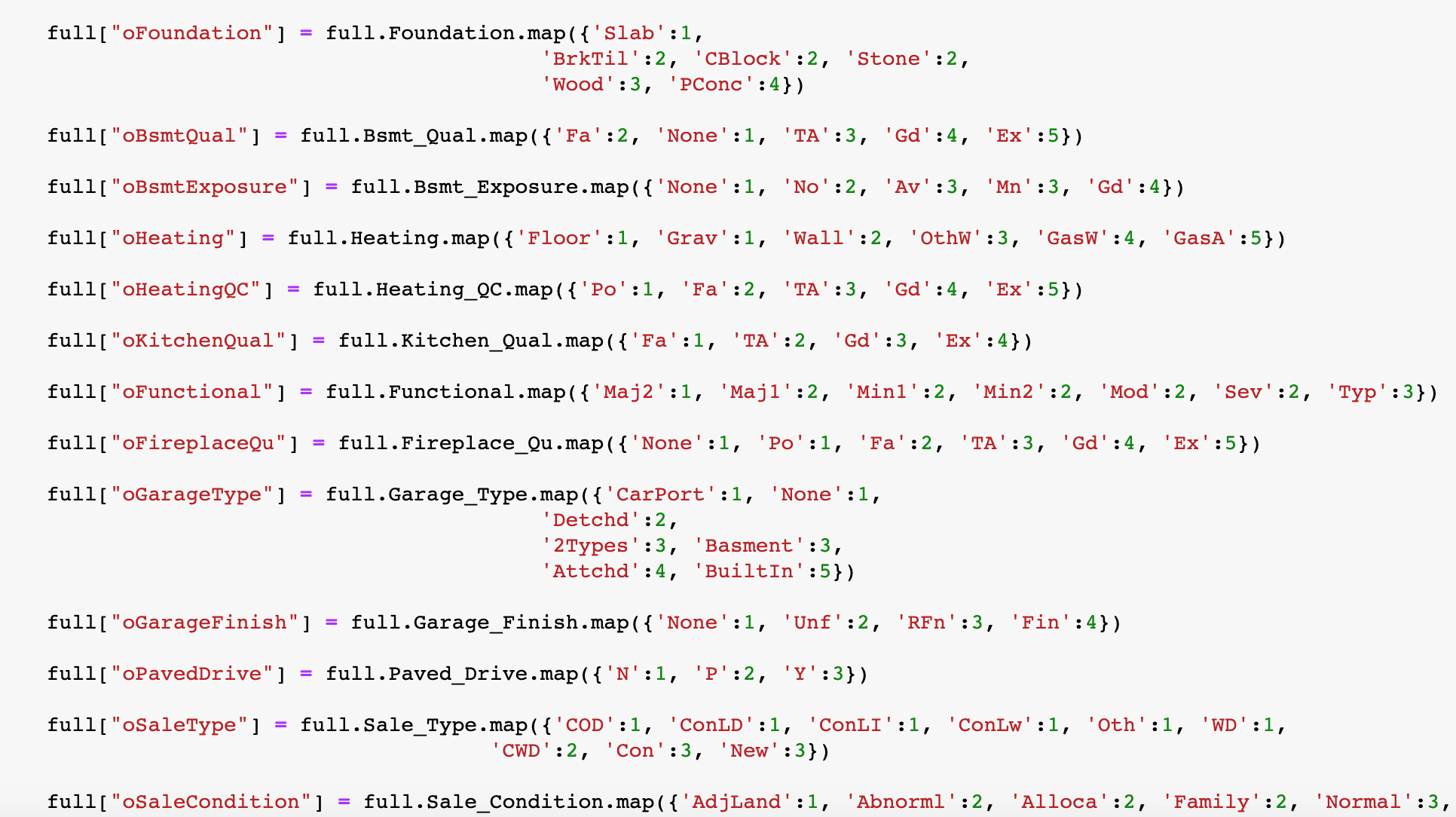
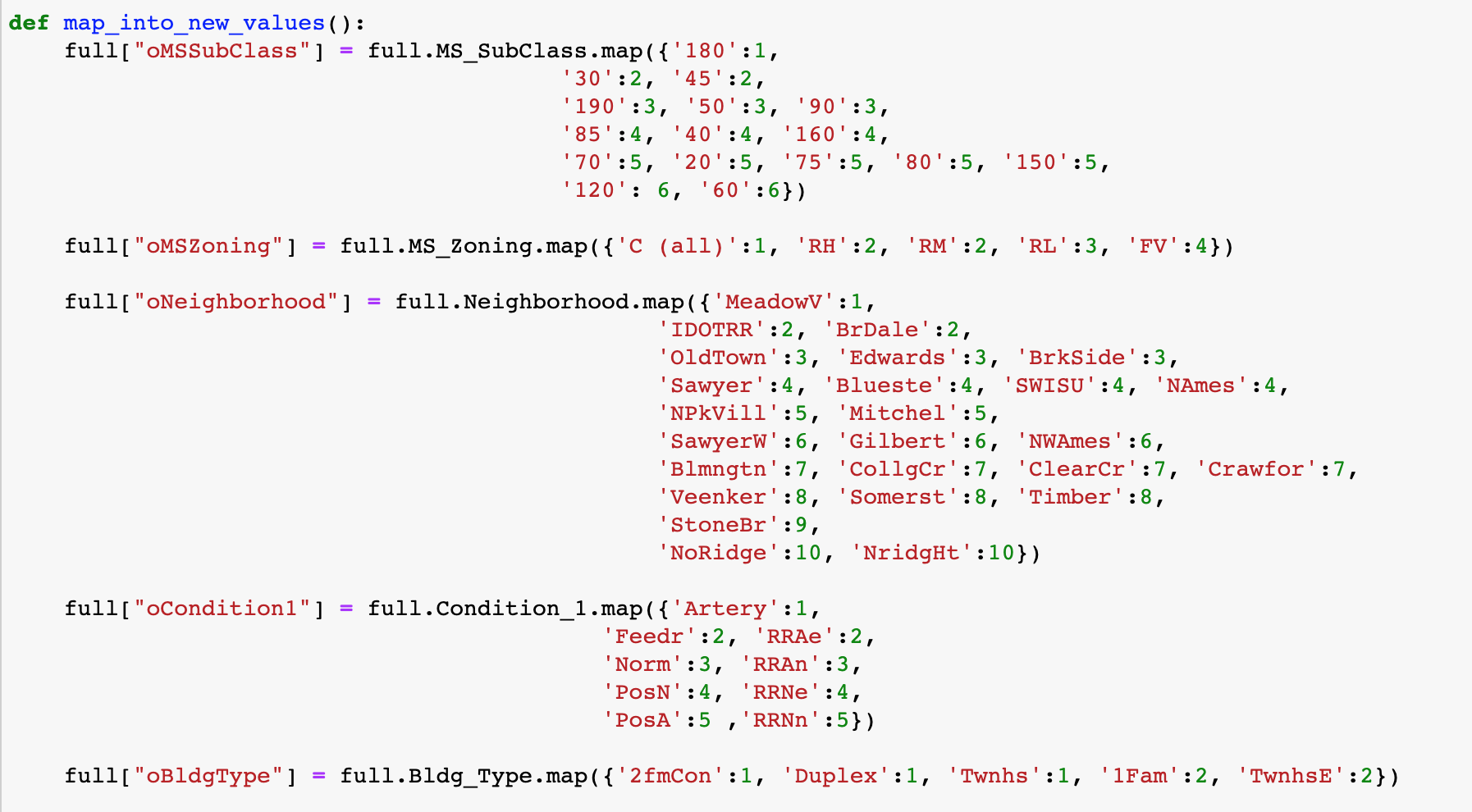


**Feature Engineering**

For discrete features, get\_dummies in pandas is generally used for digitization, but this may not be enough in this competition, so the method I use below is to group by feature and calculate the average of SalePrice for each value of the feature And the median, and then sort and assign values based on this, as an example:

The feature of MSSubClass represents the type of house, and the data is grouped by it.

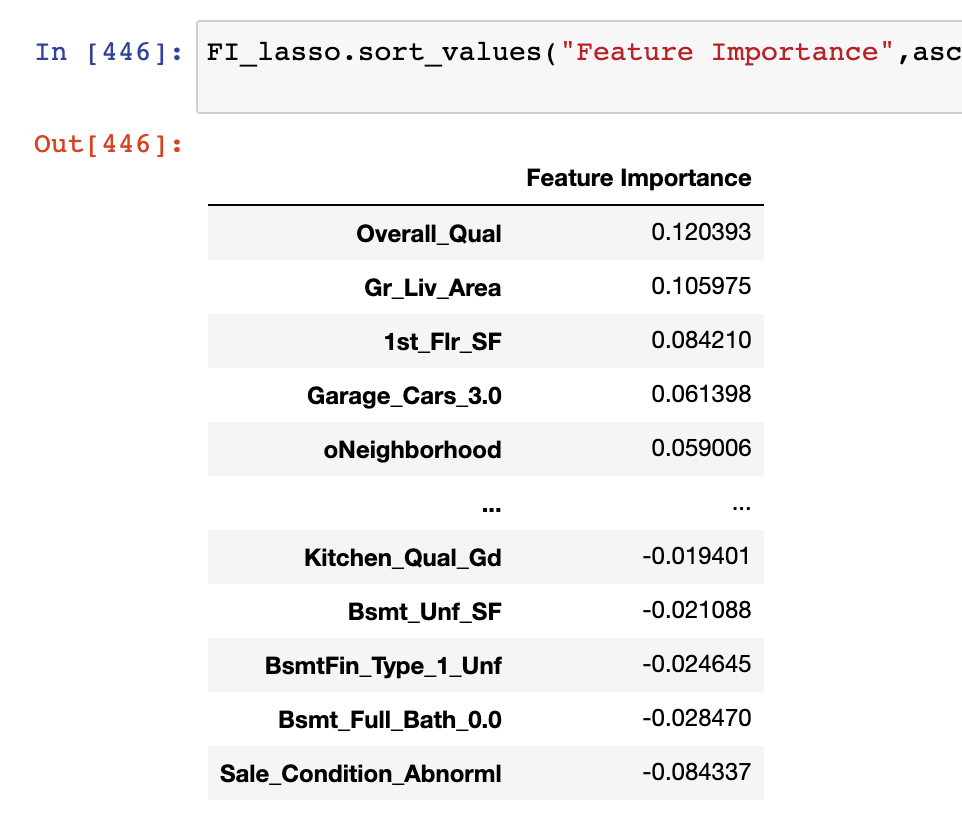
Combining the original features can usually produce unexpected results. However, there are many original features in this data set, and it is impossible to combine all of them one by one, so here we use Lasso for feature screening first, and select some of the more important features for combination

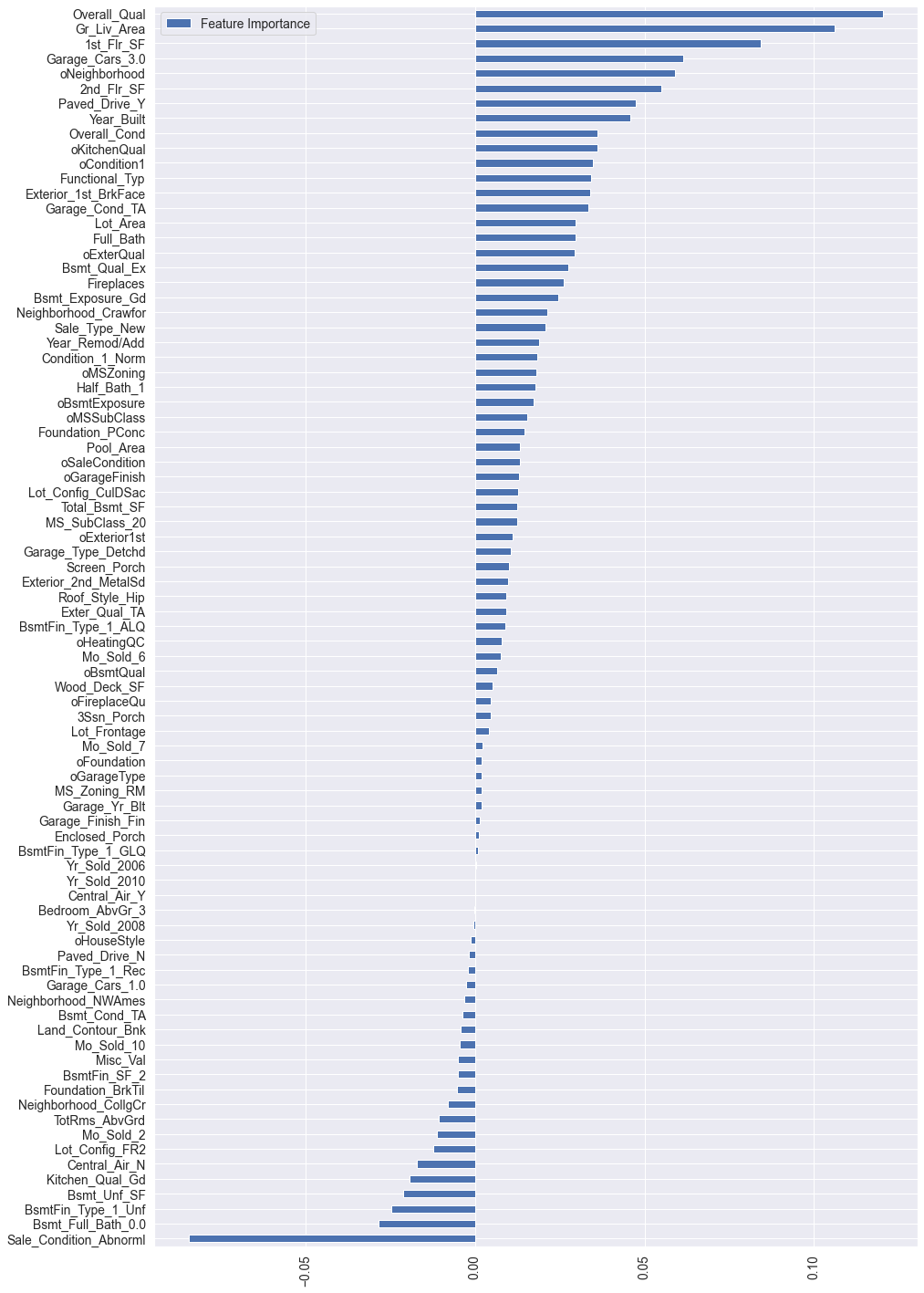


I have converted the values of dozens of columns in total, in order to facilitate classification, you can refer to the code for details

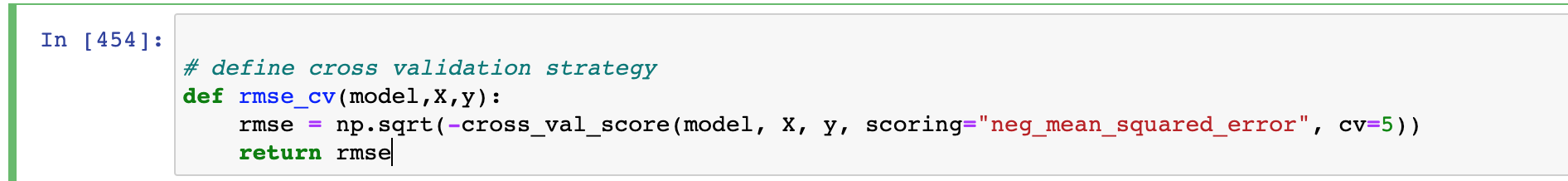
**Methodology**

Combining the original features can usually produce unexpected results. However, there are many original features in this data set, and it is impossible to combine all of them one by one, so here we use Lasso for feature screening first, and select some of the more important features for combination





PCA is a very important part, which greatly improves the final score. Because these new features I added are highly correlated with the original features, this may lead to strong multicollinearity (Multicollinearity), and PCA can just decorate the correlation. Because the purpose of using PCA here is not to reduce dimensionality, n\_components uses dimensions that are similar to the original. This is the result of my multi-party experiment, that is, add XX features in the front, and then reduce to XX dimensions.

First define the cross-validation evaluation index of RMSE

12 algorithms and 5-fold cross-validation are used to evaluate the baseline effect:

LinearRegression

Ridge

Lasso

Random Forrest

Gradient Boosting Tree

Support Vector Regression

Linear Support Vector Regression

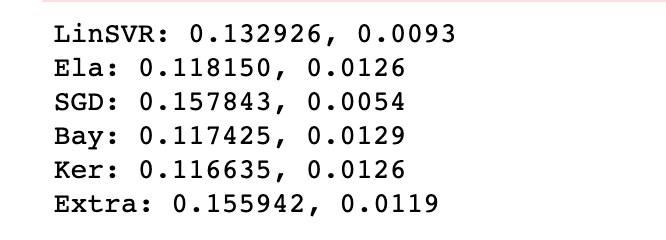
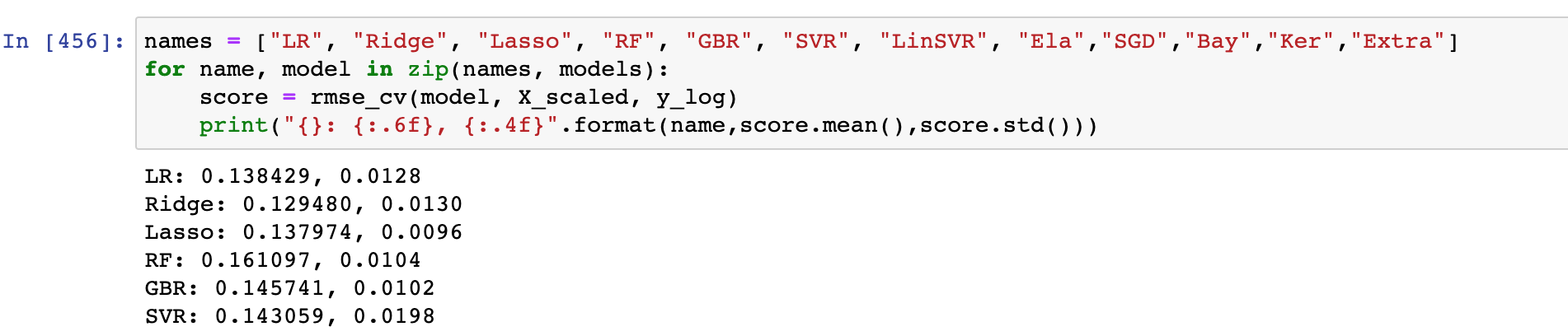
ElasticNet

Stochastic Gradient Descent

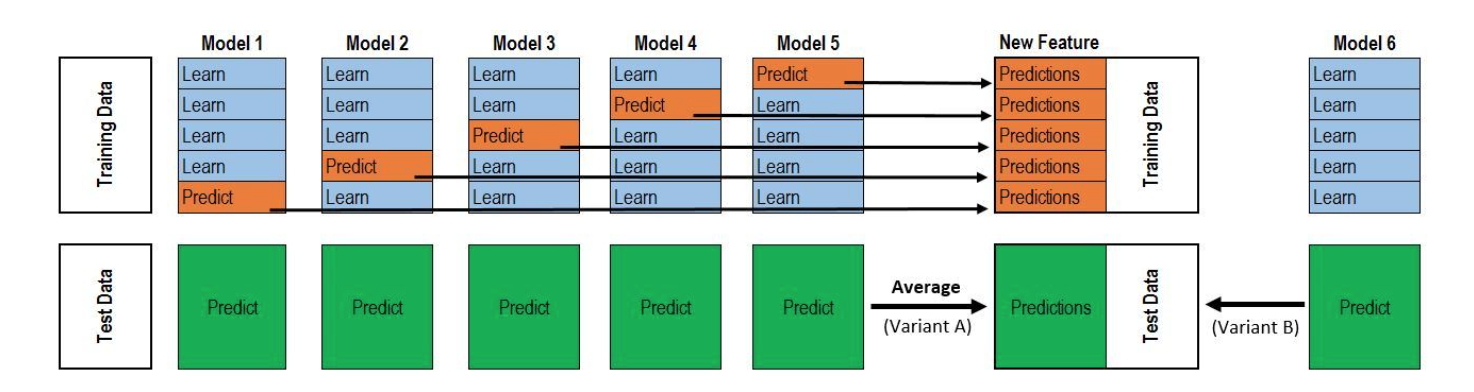
BayesianRidge

KernelRidge

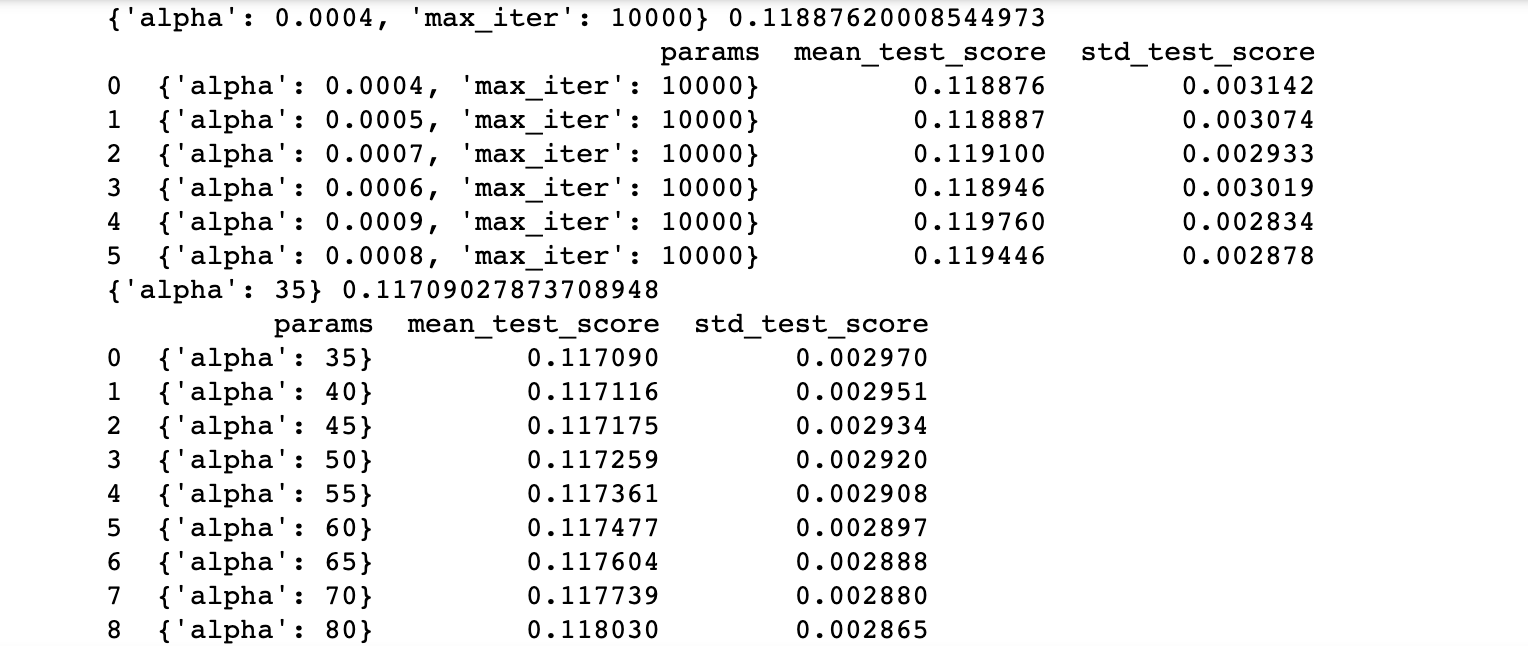
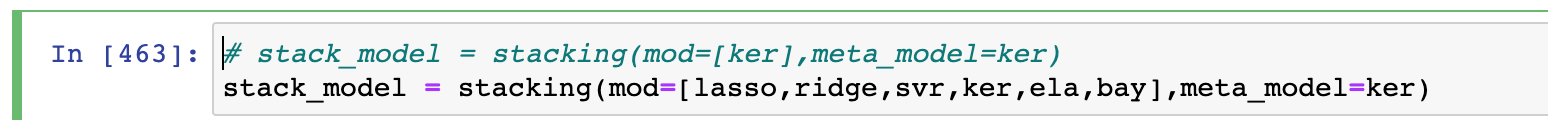
ExtraTreesRegressor

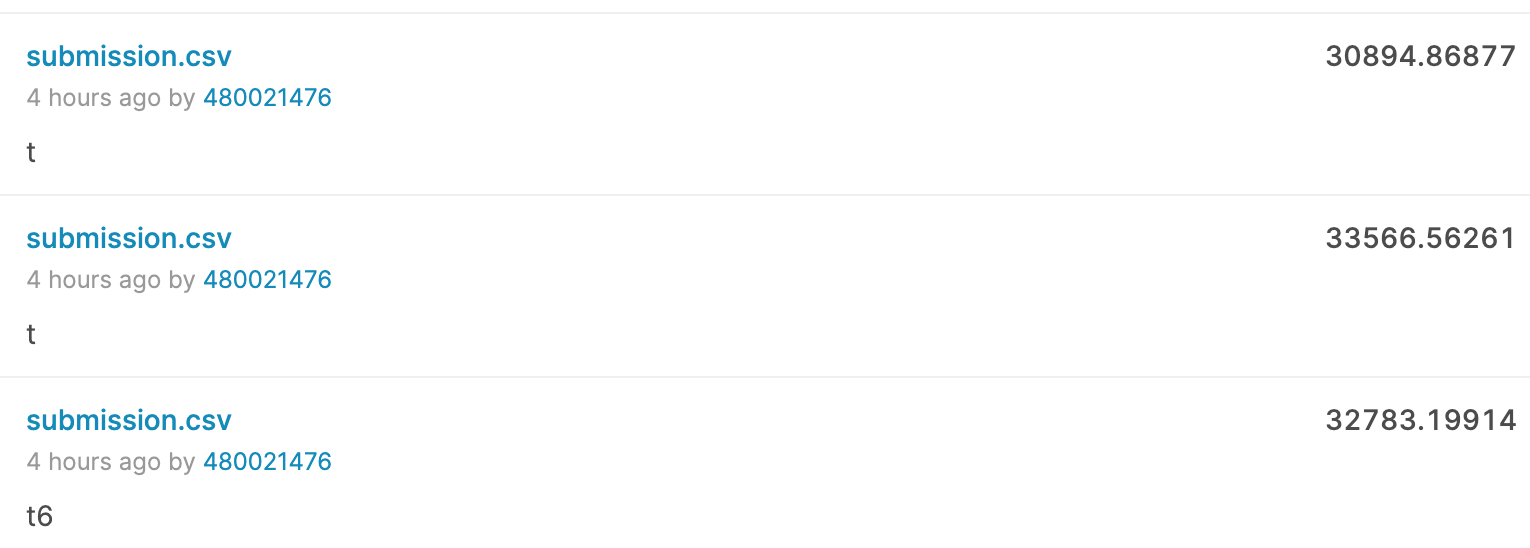
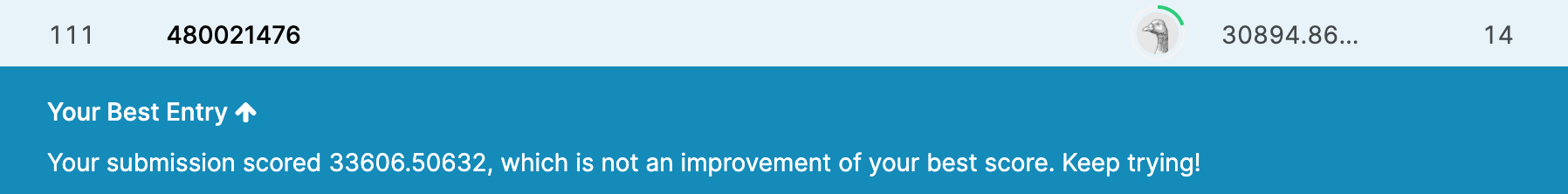


After many long rounds of testing, **svr,ker** are my best 2 models. If I use single model, I will choose from SVR and Kernel Ridge two models.

But I want to combine Weighted average and stacking all 5 models.

In the two-layer stacking as shown in the figure, there are 5 models in the first layer and 1 meta-model in the second layer. The function of the first layer model is to train to obtain a [formula] feature matrix for input to the second layer model training, where n is the number of training data rows and m is the number of the first layer model.

After many long rounds of testing, these six models were finally selected

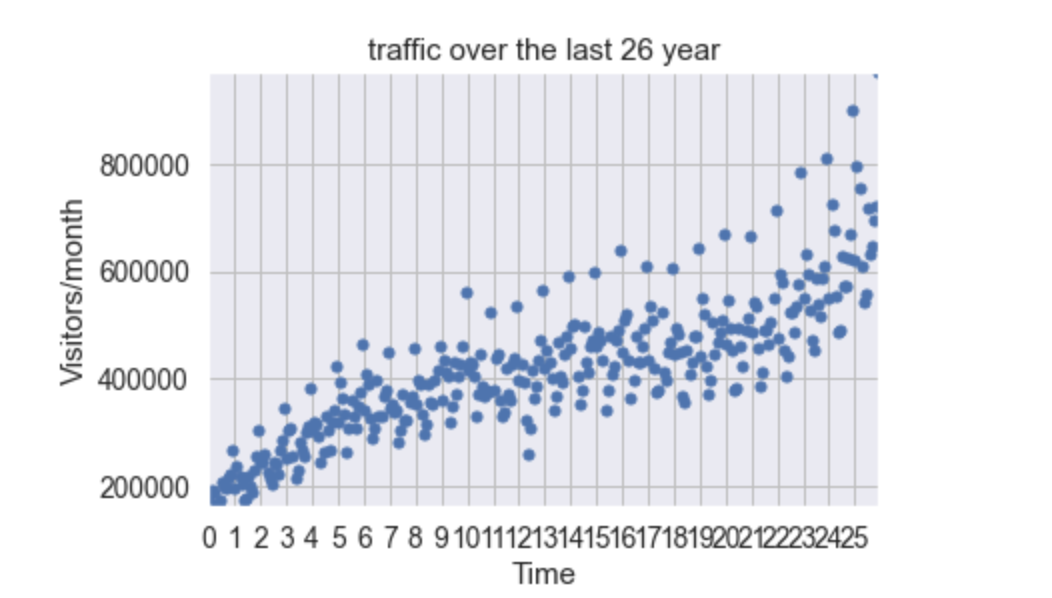
**Validation set Kaggle results   
**

**Final analysis, conclusion, limitations**

It is important to understand the data, and it is important to clean and transform the data. There are both discrete and continuous features, and there are a lot of missing values. Fortunately, the contestant provided the file data\_description.txt, which describes the meaning of each feature. After understanding the content, most of the missing values can be smoothly interpolated.

If I have more time and more data, I want to try the neural network method. In the training process, as long as there are enough input x and output y, a better neural network model can be trained. This model is In similar housing price prediction problems, more accurate results can be obtained.

Task B forecast visitors

First transform the data, turn the month into a number, and then visualize the effective data after the conversion to find the law. First transform the data, turn the month into a number, and then visualize the effective data after the conversion to find the law.

We have to choose an algorithm to predict future visits, which is obviously supervised learning.

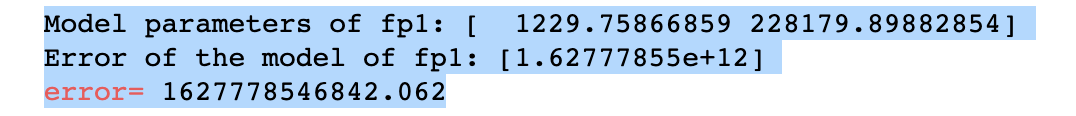
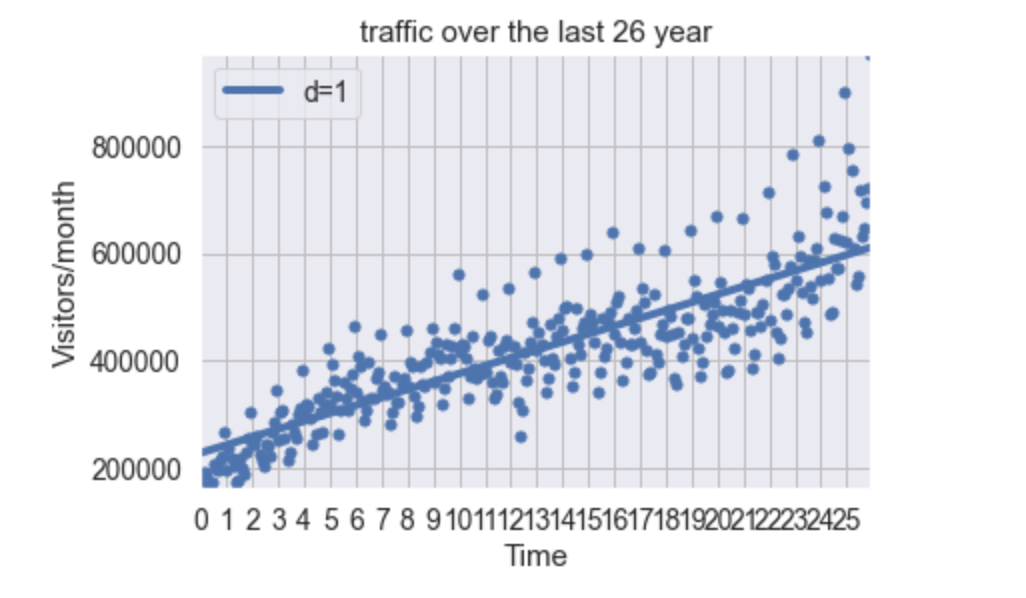
Before building our first model, we need to design an evaluation function to judge what kind of model is good. That is, the error function, which can be calculated like this,

def error(f, x,y):

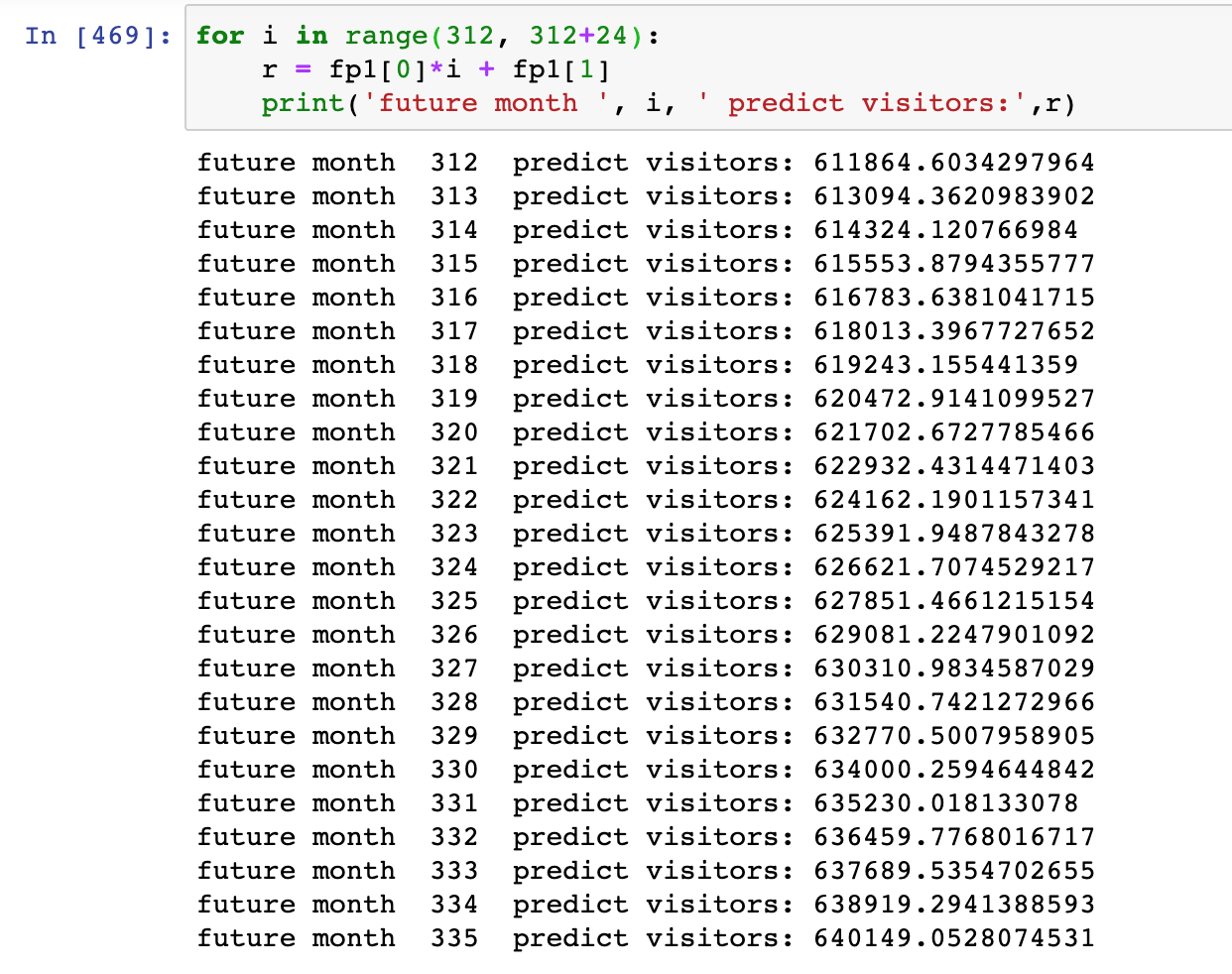
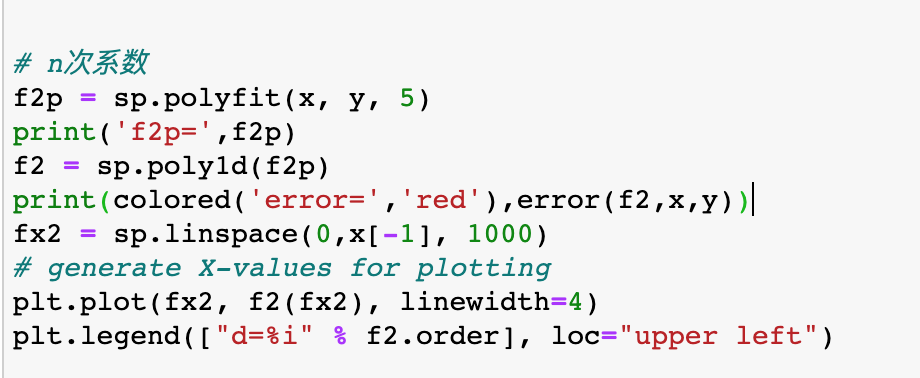
return sp.sum( (f(x) - y)\*\*2 )

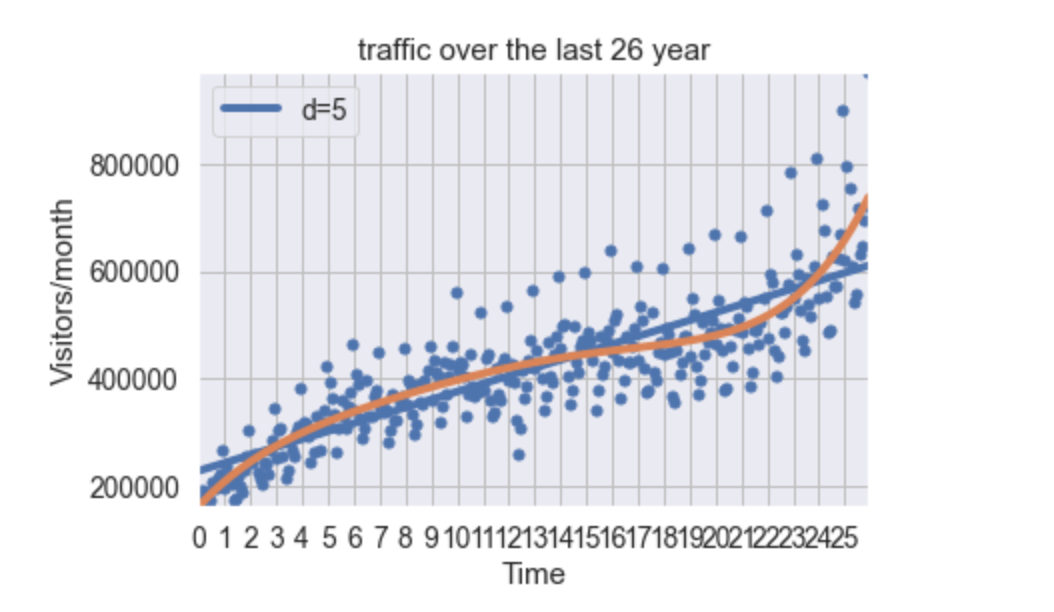
and evaluated by the square of the difference between the predicted value of the model and the true value (the training sample has been provided)

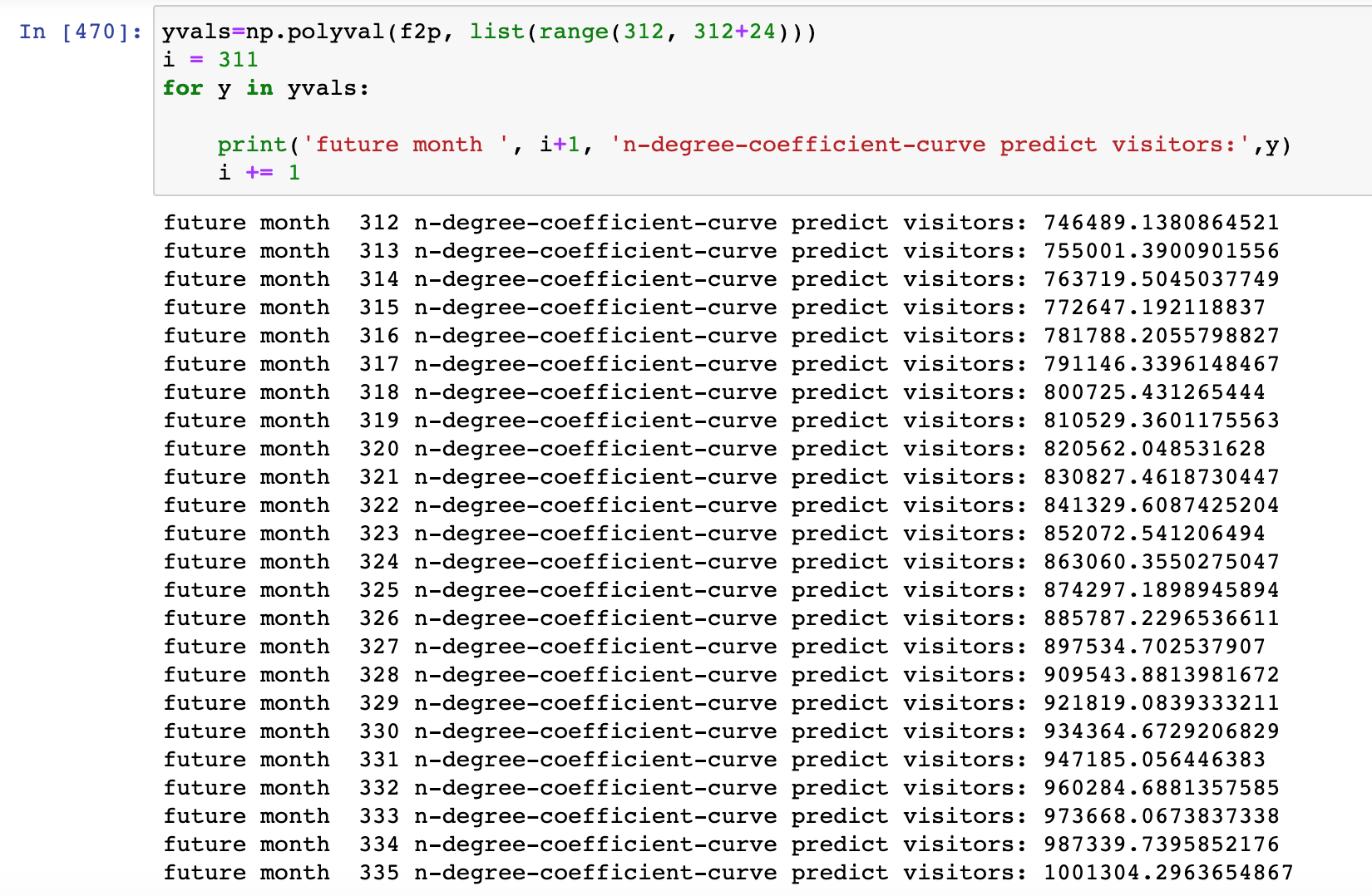
Linear regression

**Linear regression**：It is easy to know that this is actually a fitting problem. Fit these data to the best model (ie a function, and then use this function to predict new data). Starting from the simplest case, we first go to a straight line to fit the data. SciPy provides the function polyfit(), as long as the data x and y and the order of the polynomial are given (a straight line is a function of 1st order), it can find the function of the model so that the previously defined error function is minimized (only the smallest error is the surface The best model). Type in: fp1,residuals,rank,sv,rcond=sp.polyfit(x,y,1,full=True)

forecasting results for 24 months of monthly number of visitors following   
the last period in the dataset:

**Polynomial regression**：A straight line is a first-order function, obviously not optimized enough, and then we start to consider the second-order curve, the third-order curve... Up to the fifth-order curve, it can find the function of the model, so that the previously defined error function is minimized

the error function value is lower than line-function model.

forecasting results for 24 months of monthly number of visitors following   
the last period in the dataset:

To visually compare the pros and cons of algorithms. This time Use simple linear regression and polynomial regression to solve the problem. In simple situation like this: dataset is small, we can use this way.

In the future, you can also try to deal with the number of visitors in a time series method, if the dataset is more related with time and If the trend after visualization is still not obvious enough。