METCS521 Final Project Submission

Option 1: House Price Predictor

Name: Dewei Tan

Introduction

In this project I am exploring a large house information dataset in order to do house price prediction, which contains just 2919 rows but 81 columns. It means I have to handle 81 attributes of each house. I will learn and apply accessible machine learning algorithms and models to predict the sale price of a house. To achieve this, I will create a Jupyter Notebook script that demonstrates various steps including:

1. Data preprocessing
2. Outlier analysis
3. Multi-collinearity exploration
4. Linear regression analysis
5. Model validation
6. Prediction

Data Preprocessing

To be clear, I will apply multiple linear regression analysis to do the house price prediction. After reading the dataset into script, I find that there are too many attributes and most of them are categorical variables. So, I have to convert them to dummy variables. Besides, I will drop all NA values as well. But after dummying, I find that the dataset become much larger. What I want to do is to find the correlation between dependent variable, which refers to sale price, and independent variables. I will remove those weak correlations after just in order to make the model simpler and readable.

Outlier Analysis

Before exploring correlations, I try to find if there are some abnormal sale prices. I will use z-score of 2.5 as standard and the result is shown below in Figure 1. I do this just to make sure most of data is in the similar level and I do not want abnormal items to influence model too much.

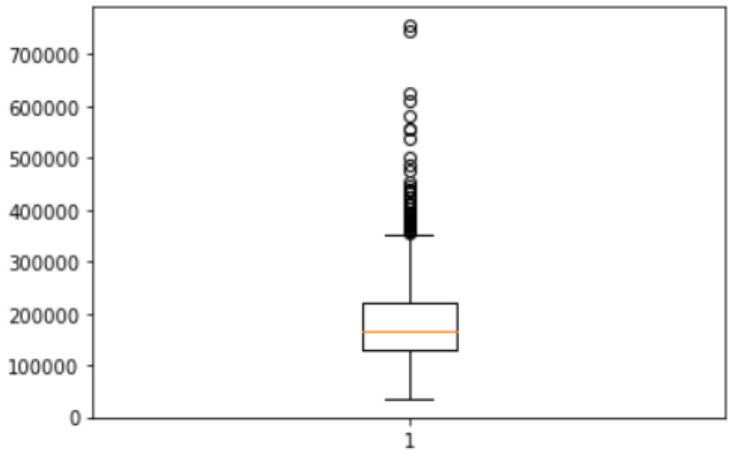

Figure 1

In Figure 1, those circles represent outliers. It shows that abnormal items just take a really small part of the whole dataset since they are almost countable.

Multi-Collinearity Exploration

Since after dummying, there are too many attributes in the dataset, I have to find the correlations between dependent variable and independent variables in order to remove weak correlations.

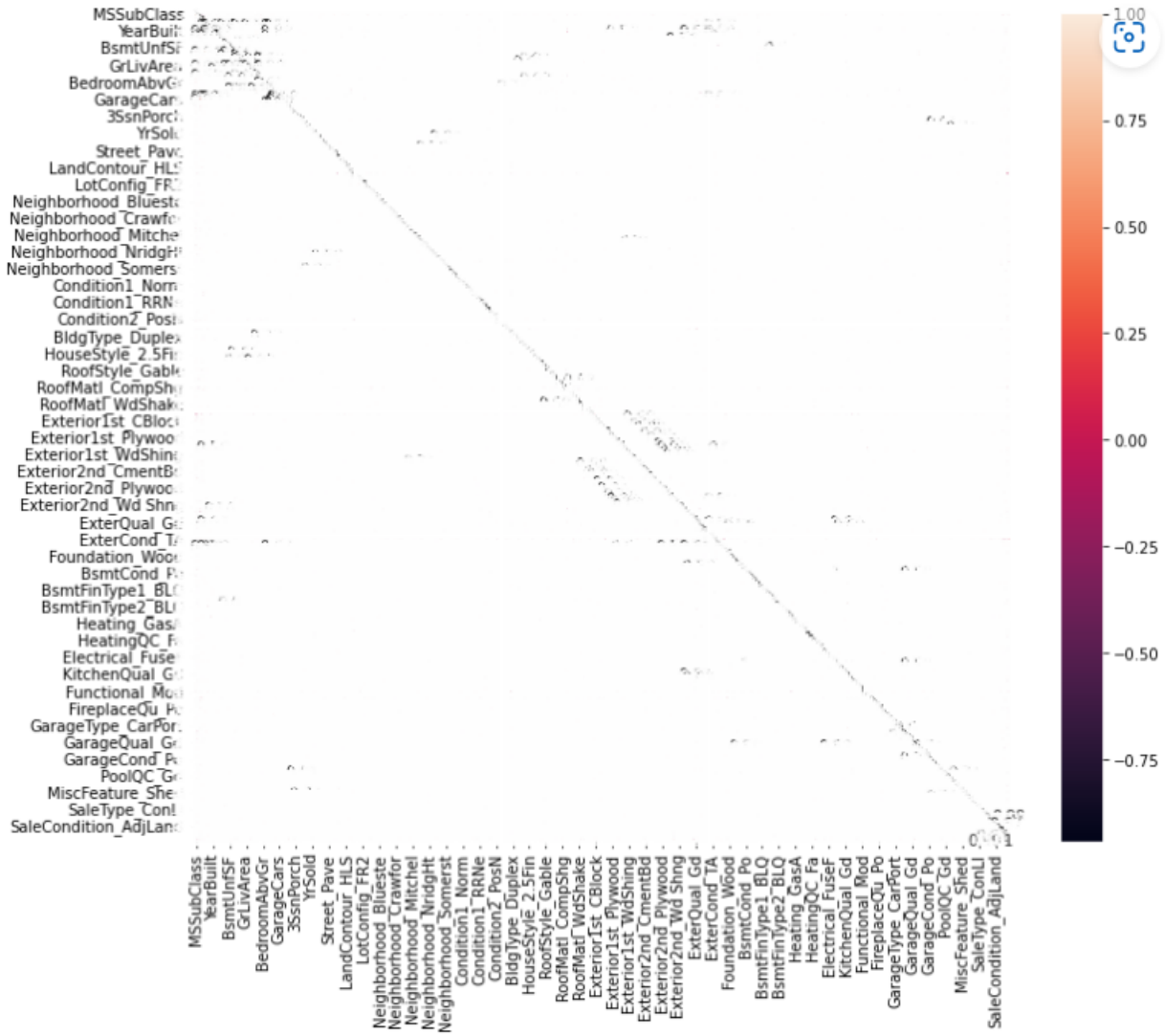

Figure 2

In Figure 2, the heatmap is kind of vague. I will take correlation number > 0.7 as high-correlated. Some color points can still be seen. Some of them are low correlated but they are hard to be figured out in this heatmap. But at least I know they exist now.

Linear Regression Analysis

Before creating the model, we split dataset into training set and testing set. I would like to use 0.2 as size of testing set. In my understanding, linear regression model aims to find the coefficient of each independent so that a linear regression equation can be formed. We can use this formula to predict what we want as long as we set independent variables. In this project, I will apply OLS (Ordinary Least Squares) regression to estimate coefficients of linear regression because there are too many independent variables that affect sale prices and normal linear regression models do not work well.

```
Model_1 = sm.OLS(train_y, train_X).fit()
```

```
Model_1.summary2()
```

Model:	OLS	Adj. R-squared:	0.917
Dependent Variable:	SalePrice	AIC:	20760.9481
Date:	2022-08-12 05:56	BIC:	21854.8785
No. Observations:	896	Log-Likelihood:	-10152.
Df Model:	227	F-statistic:	44.53
Df Residuals:	668	Prob (F-statistic):	3.98e-299
R-squared:	0.938	Scale:	5.4568e+08

	Coef.	Std.Err.	t	P>|t|	[0.025	0.975]
const	-692256.7647	1013012.9625	-0.6834	0.4946	-2681329.6207	1296816.0912
MSSubClass	-46.3062	147.7924	-0.3133	0.7541	-336.4997	243.8874
LotFrontage	47.1805	65.9125	0.7158	0.4744	-82.2402	176.6012
LotArea	0.5729	0.1726	3.3192	0.0010	0.2340	0.9117
OverallQual	6686.6188	1532.8380	4.3622	0.0000	3676.8583	9696.3793

Figure 3

The created model and its summary are shown in Figure 3. I first look at R-squared. R-squared value means that the percentage of variation in selling price in this case that is explained by the model. The higher it is, the better is the model. R-squared is 0.938. So far, the model is good. Then I look at p values. P value less than 0.05 means the variable is significant to the model. Now we can remove those insignificant features.

```
sig_var = ['LotArea', 'OverallQual', 'OverallCond', 'YearBuilt', 'BsmtFinSF1',
           'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF', 'Fireplaces',
           'GarageCars', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MoSold',
            'MSZoning_FV', 'MSZoning_RL', 'MSZoning_RM', 'Street_Pave',
            'LandContour_HLS', 'Neighborhood_Edwards', 'Neighborhood_NoRidge',
           'Neighborhood_NridgHt', 'Neighborhood_StoneBr', 'Condition1_Norm',
           'Condition2_PosN', 'Condition2_RRAe', 'RoofMatl_Metal', 'ExterQual_Gd',
           'ExterQual_TA', 'ExterCond_Po', 'Foundation_Wood', 'BsmtExposure_Gd',
           'KitchenQual_Gd', 'KitchenQual_TA', 'Functional_Sev', 'FireplaceQu_TA',
           'GarageQual_Fa', 'GarageQual_Gd', 'GarageQual_Po', 'GarageQual_TA',
           'GarageCond_Fa', 'GarageCond_Gd', 'GarageCond_Po', 'GarageCond_TA',
           'PoolQC_Fa', 'PoolQC_Gd', 'SaleCondition_Normal']
```

```
train_X = train_X[sig_var]
```

Figure 4

In Figure 4, I collect the significant variables and iterate the model repeatedly till there are no insignificant variables in the model. In the script, I iterate 4 times.

Now we can get all the coefficients we need from the model.

Model Validation

Since multiple linear regression are based on some statistical assumptions. If assumptions are not satisfied, the model is not reliable even it behaves well. One widely used method is to check homoscedasticity. This is an important assumption of parametric statistical tests because they are sensitive to any dissimilarities.

```
def get_standard_values(vals):
    return (vals - vals.mean()) / vals.std()
```

```
plt.scatter(get_standard_values(Model_4.fittedvalues),
            get_standard_values(Model_4.resid));
```

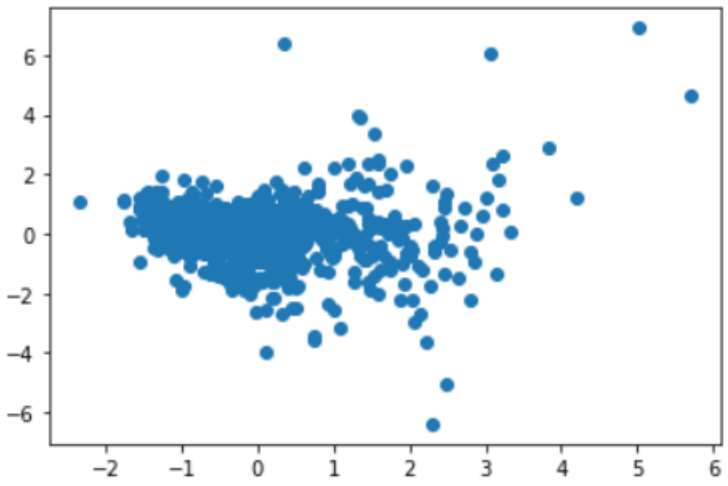

Figure 5

The result is shown in Figure 5. It should be no presence of pattern in this scatterplot. Fitted values and residuals should not have the same scatter. This scatterplot is concentrated at a particular area and points are further scattered apart suggesting no pattern in this plot. We can conclude that the model is valid.

Prediction

Now we can make predictions with our model. We also can use coefficients to set the equation.

```
pred = Model_4.predict(test_X[sig_var3])
pred
```

```
183     191219.708339
1046    381175.153905
1142    325404.065880
349     430141.117040
29       52609.222113
              ...
653     130843.055820
571     126182.042716
25      262528.224376
1051    183942.222071
644     353483.275709
Length: 225, dtype: float64
```

Figure 6

We also can measure RMSE and R-squared score.

```
# Measuring RMSE
from sklearn import metrics
np.sqrt(metrics.mean_squared_error(pred, test_y))
```

```
50441.70871721007
```

```
# Measuring R2
np.round(metrics.r2_score(pred, test_y))
```

```
1.0
```

Figure 7

RMSE refers to Root Mean Square Error. It is a standard way to measure the error of a model in predicting. From that we can know the deviation of prediction generally.

R-squared score increase to 1. It means this model cannot be improved further.

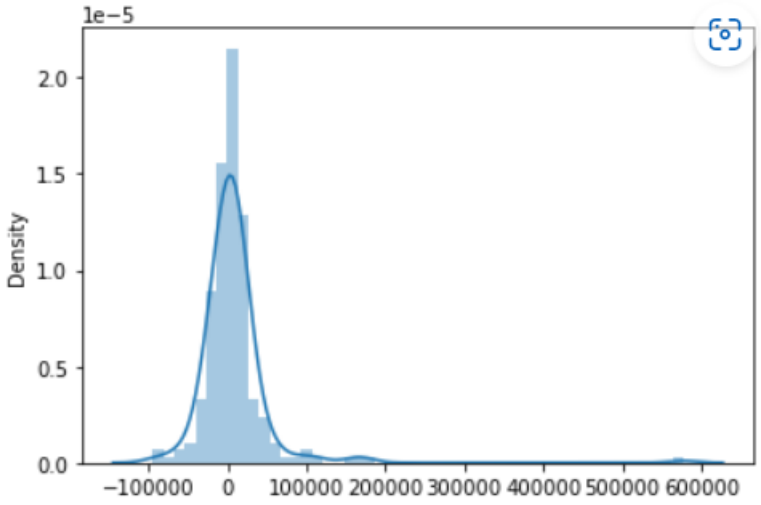

Figure 8

In Figure 8, the residuals normal distribution means the assumption is valid since normality of residuals is an assumption of running a linear regression model.

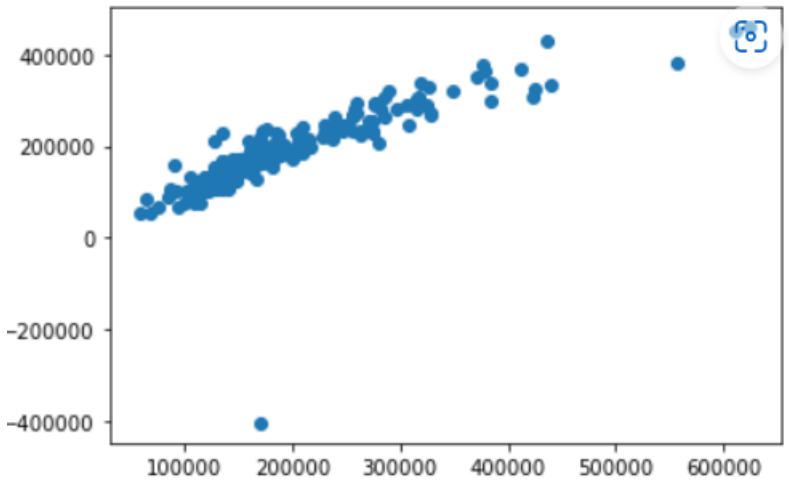

Figure 9

I am plotting a scatter plot in Figure 9 between the actual value and predicted value. The plot is linear but as values increase, prediction accuracy goes down

Conclusion

I have built a model to do house price prediction above. There are two ways to predict. For this project, I think the most difficult step is to get started. This dataset is the most complex one I have ever managed since it has too many attributes. The categorical variables have to be converted to dummy variables. But it put a lot of burden on model. I have tried Linear Regression model from sklearn, but it is hard to get coefficients and set equations. And it is hard to remove unnecessary attributes in that model. The data is a mess. Maybe my way of using the model is wrong. But I think it does not fit the dataset. I also have tried to use RandomForest method to find feature importance and remove low ones. But the dataset is too big after dummying and the Jupyter Notebook can not allocate. The method and model I apply in this project

is newly learned for me. But it is very useful and deepen my understanding of some parameters in models and add one new model to my model list. It also improves my ability to interpret data analysis since we can apply it only when we totally understand it. Every time before we dive into exploring data, we should first consider the correct way to explore it, or you might spend a lot of time on trial like me. Every time I search a new method which seems work, I will have a try. This process improves my search capabilities and I get a bunch of good websites for data analysis. From their codes, I have met many new libraries that I did not use before. Some of them are pretty practical to use. New parameters and concepts within the method and models are impressive as well. So, from this final project, I improve various aspects of myself.