HOUSING PRICE PREDICTION

Submitted by:

Rita Padghans-Karanjkar

• Business Problem Framing

- • Every person around the globe and therefore housing and real estate market is one of the markets which is one of the major contributors in the world's economy. It is a very large market and there are various companies working in the domain. Basically Price Prediction is technique to increase the companies that are in the domain of housing and real estate and are working to achieve the Business goals such as increasing overall revenue, profits, improving their marketing strategies and focusing on changing trends in house sales and purchases.
- Our problem is related to one such housing company. A US-based housing company named Surprise Housing has decided to enter the Australian market. The company uses data analytics to purchase houses at a price below their actual values and flip them at a higher price. For the same purpose, the company has collected a data set from the sale of houses in Australia. The data is provided in the CSV file below. The company is looking at prospective properties to buy houses to enter the market.I build a model using Machine Learning in order to predict the actual value of the prospective properties and decide whether to invest in them or not. For this company wants to know:
- • • Which variables are important to predict the price of variable?
- • How do these variables describe the price of the house? (This is our Problem statement.)

- Conceptual Background of the Domain Problem
- You are required to model the price of houses with the available independent variables. This model will then be used by the management to understand how exactly the prices vary with the variables. They can accordingly manipulate the strategy of the firm and concentrate on areas that will yield high returns. Further, the model will be a good way for the management to understand the pricing dynamics of a new market.
- • Review of Literature

  Data contains 1460 entries each having 81 variables.
- • • Data contains Null values. You need to treat them using the domain knowledge and your own understanding.
- • • Extensive EDA has to be performed to gain relationships of important variable and price.
- • • Data contains numerical as well as categorical variable. You need to handle them accordingly.
- • • You have to build Machine Learning models, apply regularization and determine the optimal values of Hyper
- • Parameters.
- • • You need to find important features which affect the price positively or negatively.
- • • Two datasets are being provided to you (test.csv, train.csv). You will train on train.csv dataset and predict on

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

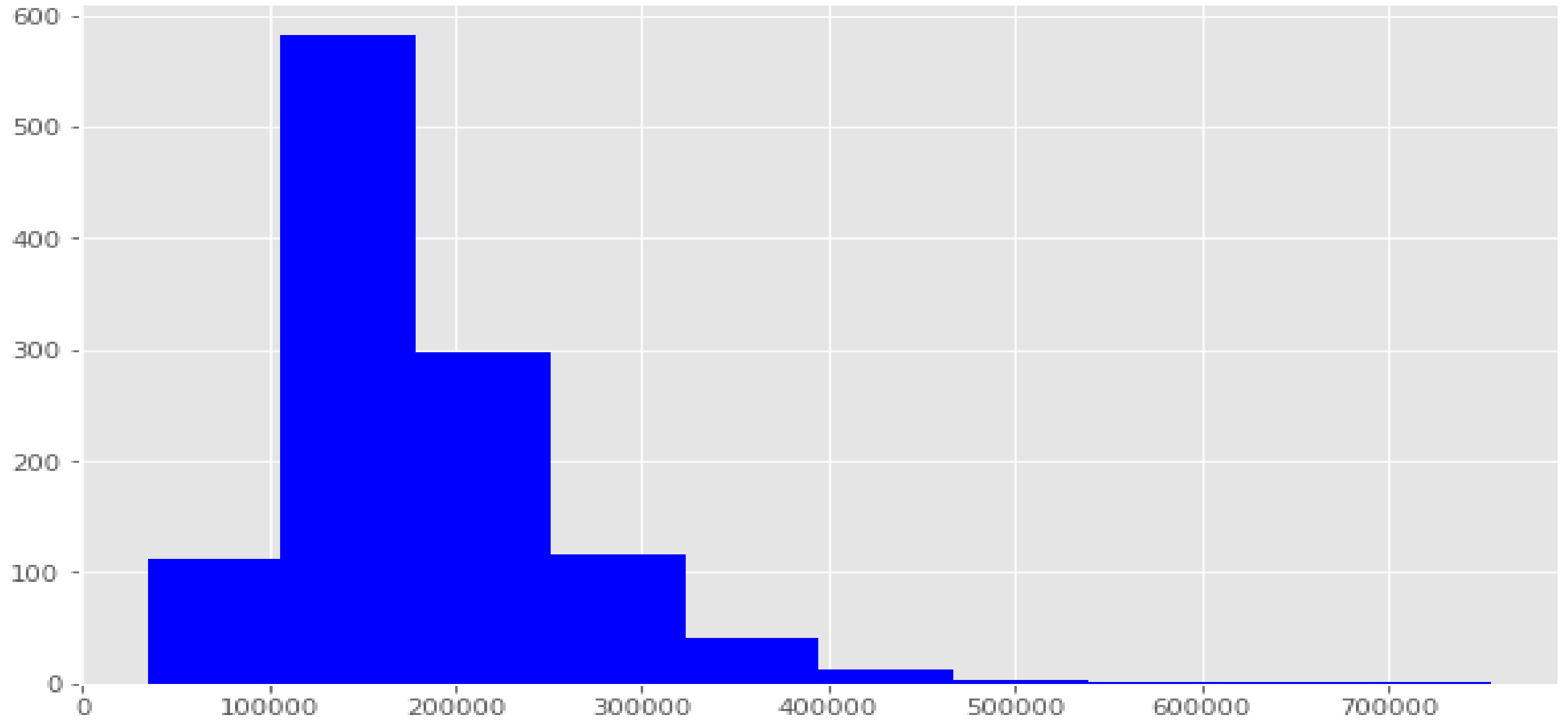
- I observe that there is anull values for integer variables that are replace by the mode of that column.
- also the feature variable columns some data is missing i replace the missing value with mode of that respective column.
- A US-based housing company named Surprise Housing has decided to enter the Australian market. The company uses
- • data analytics to purchase houses at a price below their actual values and flip them at a higher price. For the same
- • purpose, the company has collected a data set from the sale of houses in Australia. The data is provided in the CSV file below.
- • train.csv file is as follow:
- • test.csv

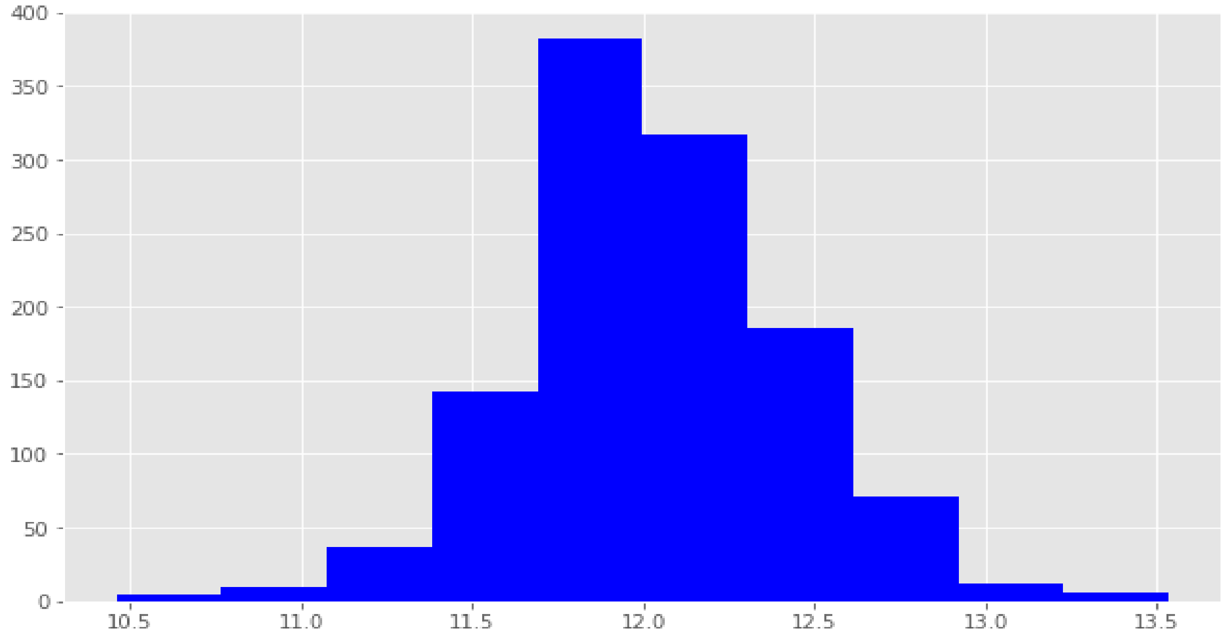
Data Pre-processing Done

- The challenge is to predict the final sale price of the homes. This information is stored in the SalePrice column. The value we are trying to predict is often called the target variable.

We use plt.hist() to plot a histogram of SalePrice. Notice that the distribution has a longer tail on the right. The distribution is positively skewed.

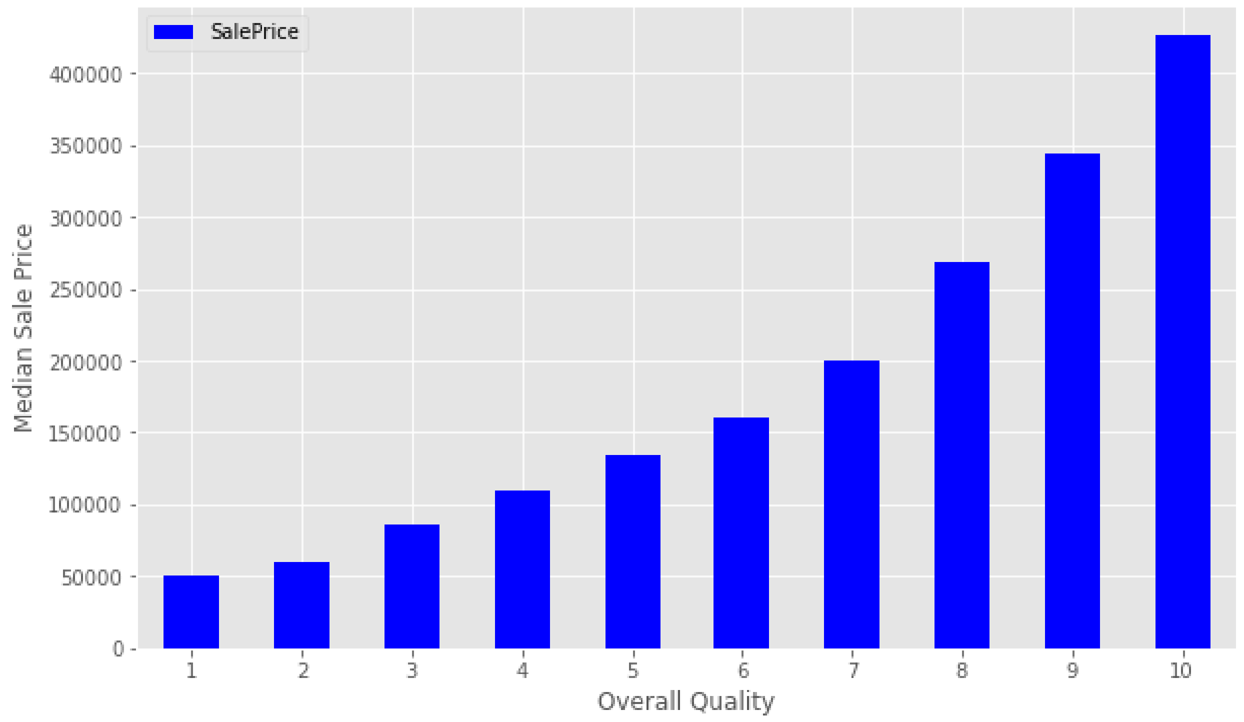
Skew is: 1.953877705368286





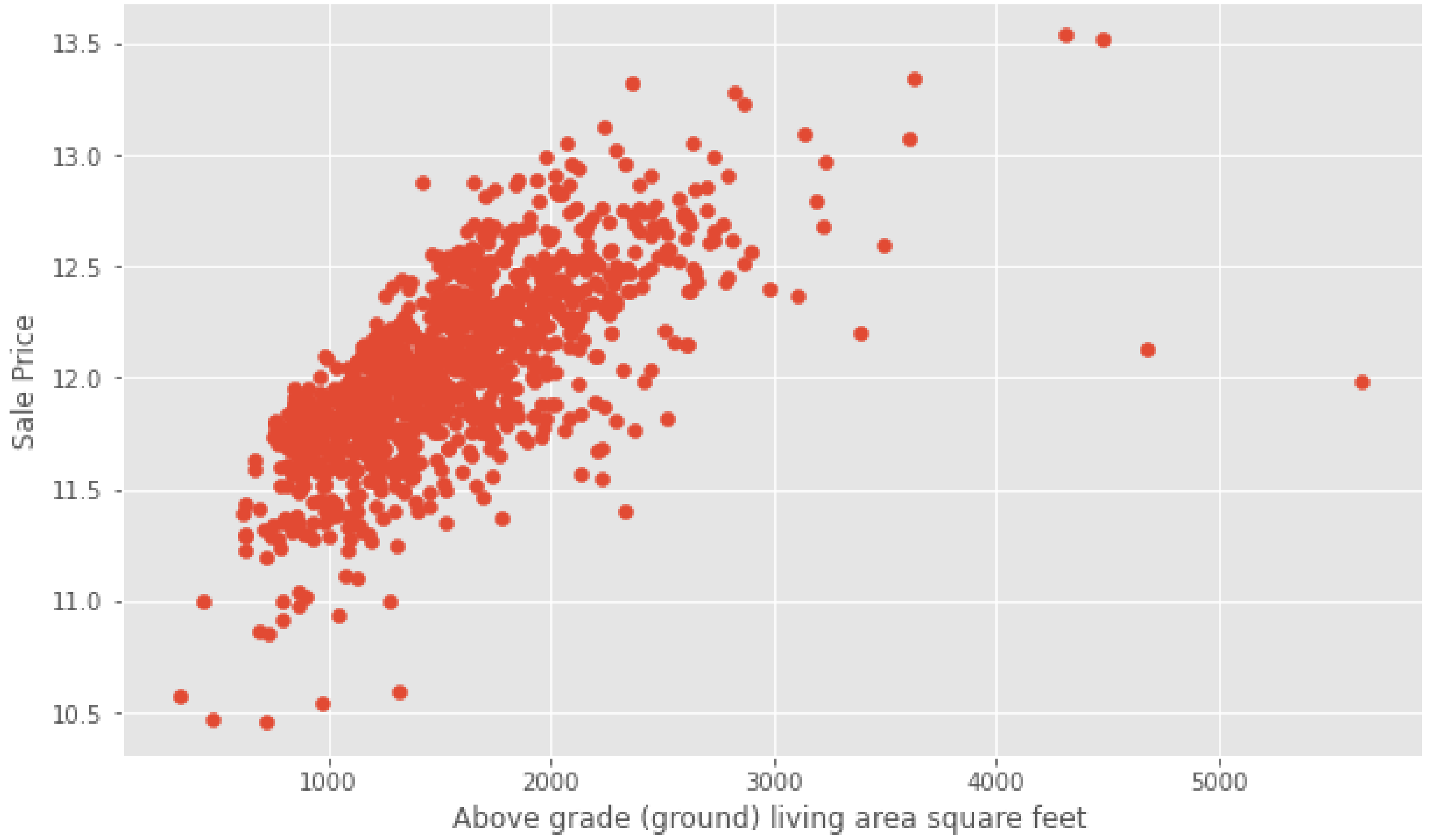
- Now we use np.log() to transform train.SalePric and calculate the skewness a second time, as well as re-plot the data. A value closer to 0 means that we have improved the skewness of the data. We can see visually that the data will more resembles a normal distribution.
- `Skew is: 0.07359740998337982`

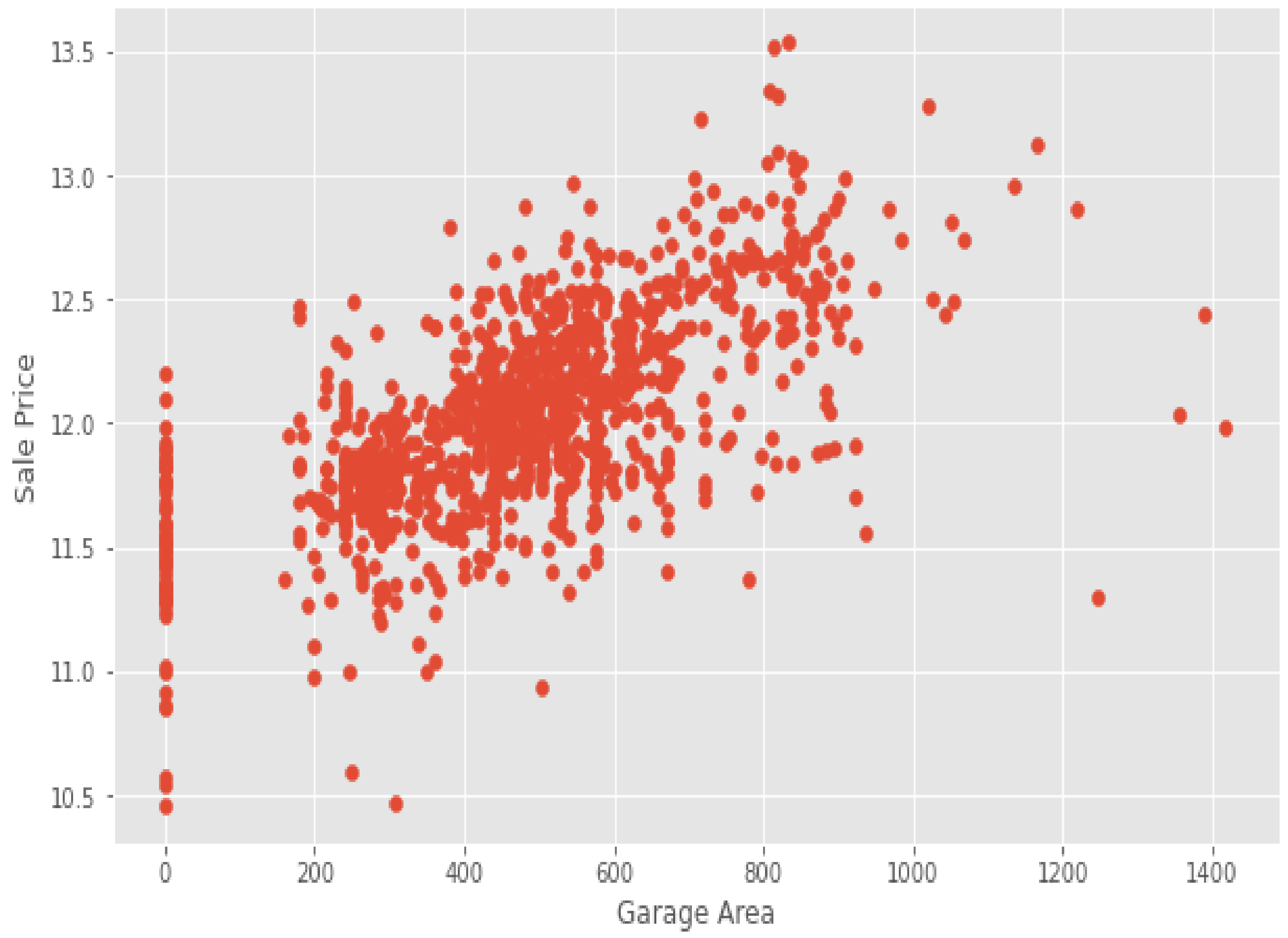
the median sales price strictly increases as Overall Quality increases.



Scatter plots and visualize the relationship between the Ground Living Area GrLivArea and SalePrice.

we see that increases in living area correspond to increases in price. We will do the same for GarageArea.





we notice that there are many homes with 0 for Garage Area, indicating that they don't have a garage. We'll transform other features later to reflect this assumption. There are a few outliers as well. Outliers can affect a regression model by pulling our estimated regression line further away from the true population regression line. So, we'll remove those observations from our data. Removing outliers is an art and a science. There are many techniques for dealing with outliers.

We will create a new dataframe with some outliers removed.

Handling Null Values

Next, we'll examine the null or missing values.

- We will create a DataFrame to view the top null columns. Chaining together the train.isnull().sum() methods, we return a Series of the counts of the null values in each column.
- The documentation can help us understand the missing values. In the case of PoolQC, the column refers to Pool Quality. Pool quality is NaN when PoolArea is 0, or there is no pool. We can find a similar relationship between many of the Garage-related columns.
- Let's take a look at one of the other columns, MiscFeature. We'll use the Series.unique() method to return a list of the unique values.
- `Unique values are: [nan 'Shed' 'Gar2' 'TenC' 'Othr']`
- We can use the documentation to find out what these values indicate:

Handling the Features Variables

Let’s now consider the non-numeric features.

- The count column indicates the count of non-null observations, while unique counts the number of unique values. top is the most commonly occurring value, with the frequency of the top value shown by freq.

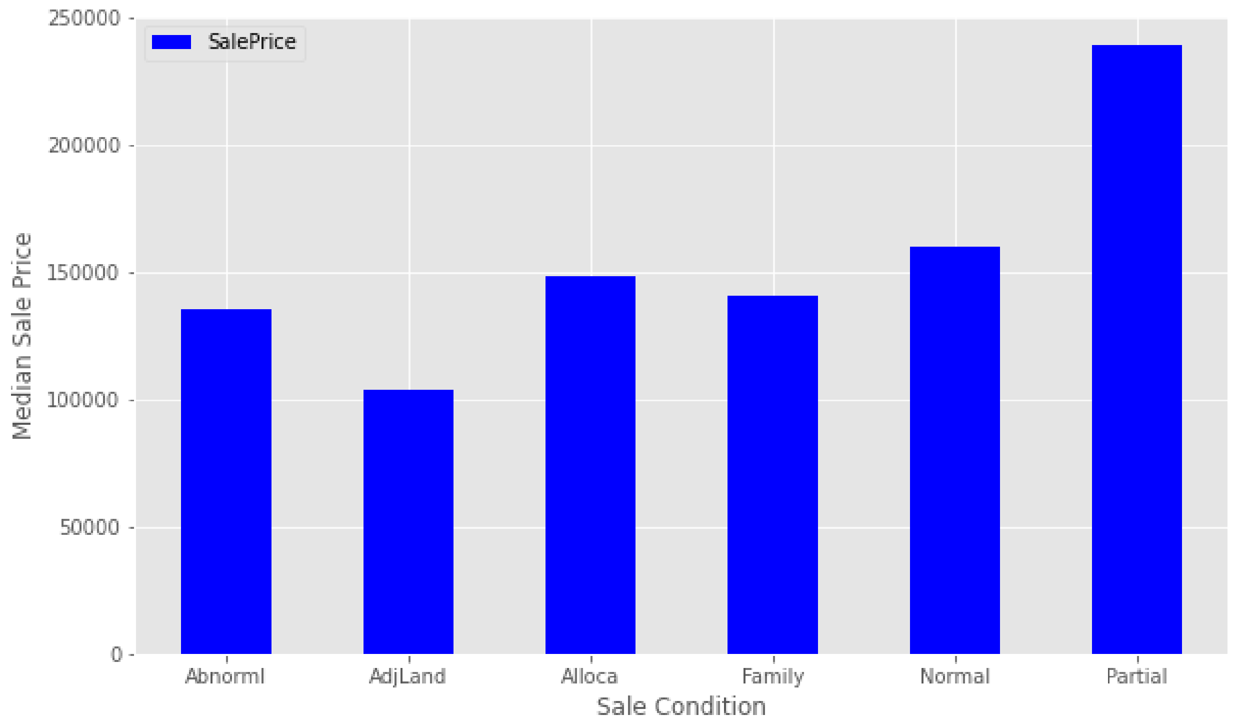
- For many of these features, we might want to use one-hot encoding to make use of the information for modeling.
- One-hot encoding is a technique which will transform categorical data into numbers so the model can understand whether or not a particular observation falls into one category or another.hen transforming features, it’s important to remember that any transformations that you’ve applied to the training data before fitting the model must be applied to the test data.

- Our model expects that the shape of the features from the train set match those from the test set. This means that any feature engineering that occurred while working on the train data should be applied again on the test set.

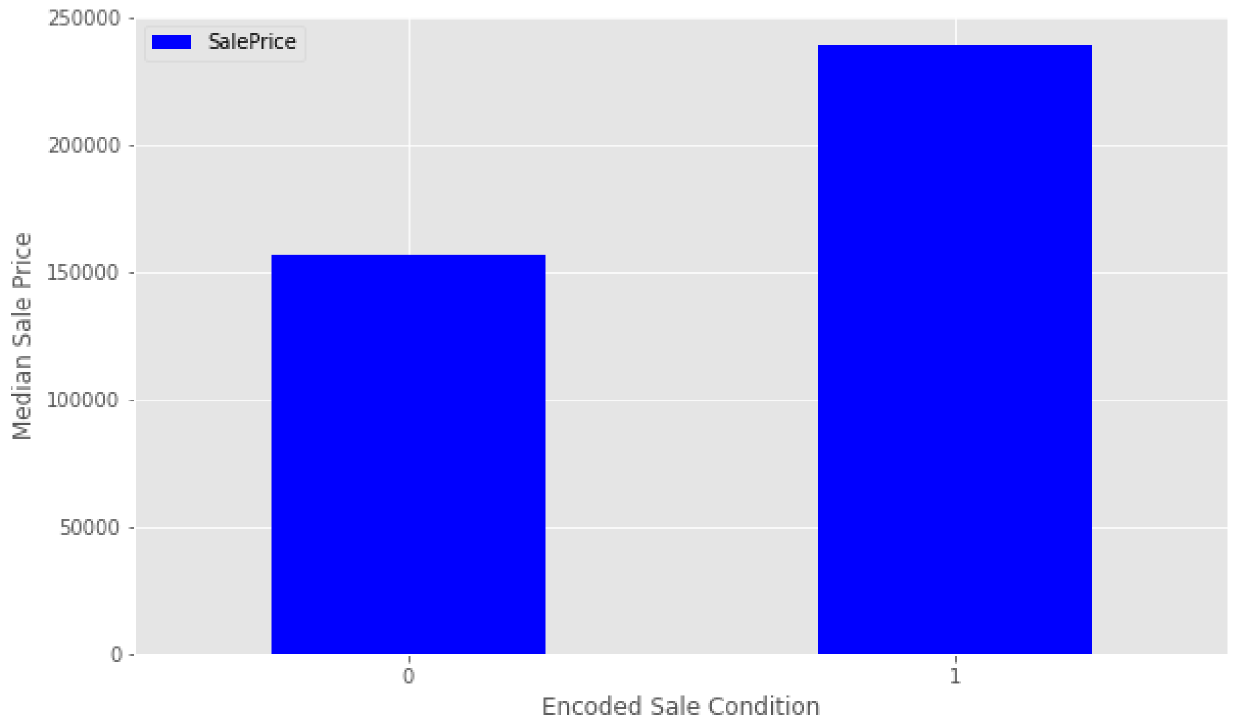
- To demonstrate how this works, consider the Street data, which indicates whether there is Gravel or Paved road access to the property.

- `Original:`
-
- `Pave    1160`
- `Grvl       3`
- `Name: Street, dtype: int64`
-
-
- In the Street column, the unique values are Pave and Grvl, which describe the type of road access to the property. In the training set, only 5 homes have gravel access. Our model needs numerical data, so we will use one-hot encoding to transform the data into a Boolean column.
- We create a new column called enc_street. The pd.get_dummies() method will handle this for us.
- As mentioned earlier, we need to do this on both the train and test data.`Encoded:`
-
- `1    1160`
- `0       3`
- `Name: enc_street, dtype: int64`
-
- The values agree. We've engineered our first feature! Feature Engineering is the process of making features of the data suitable for use in machine learning and modelling. When we encoded the Street feature into a column of Boolean values, we engineered a feature.

We'll look at SaleCondition by constructing and plotting a pivot table, as we did above for OverallQual.

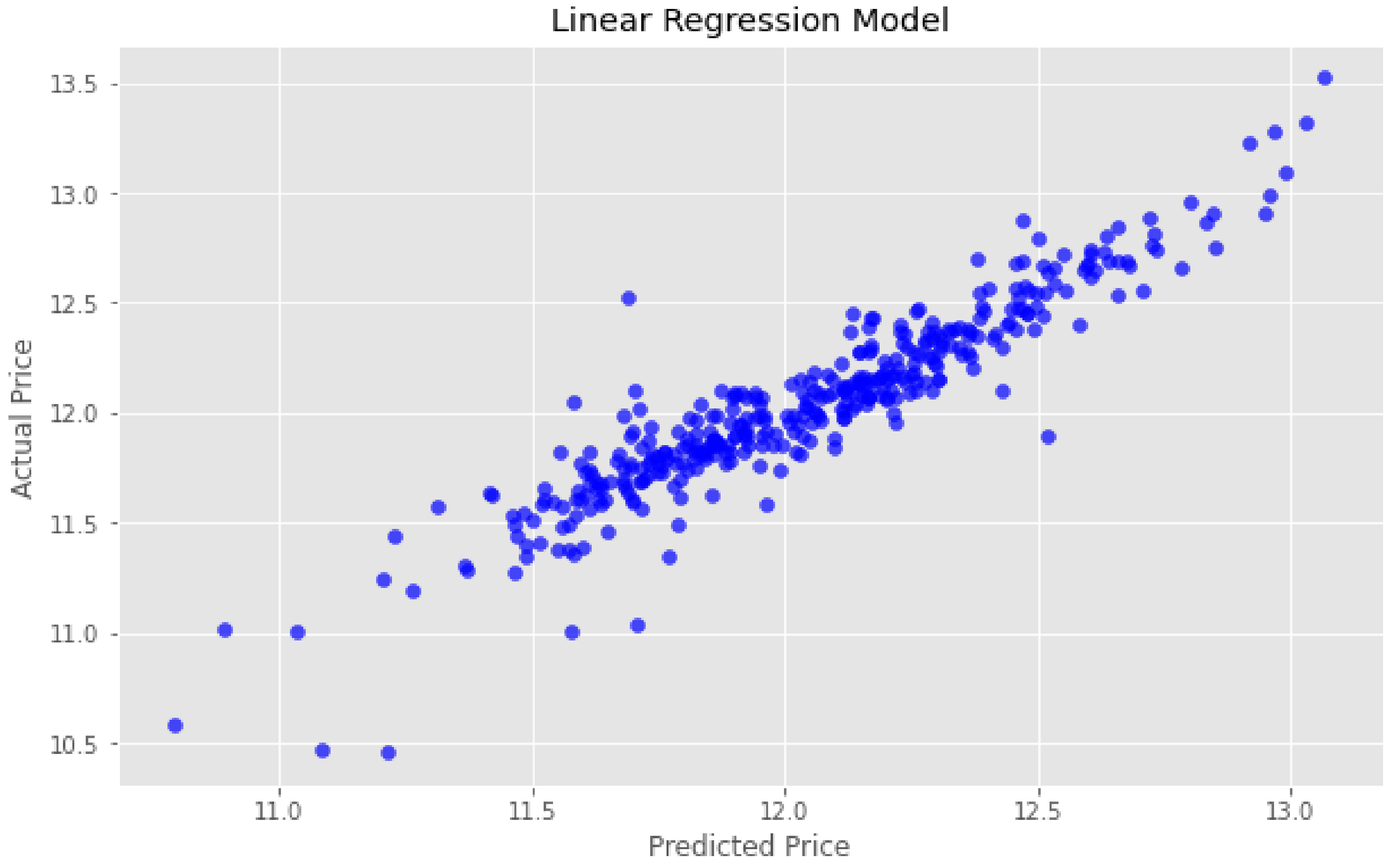


explore this new feature as a plot.

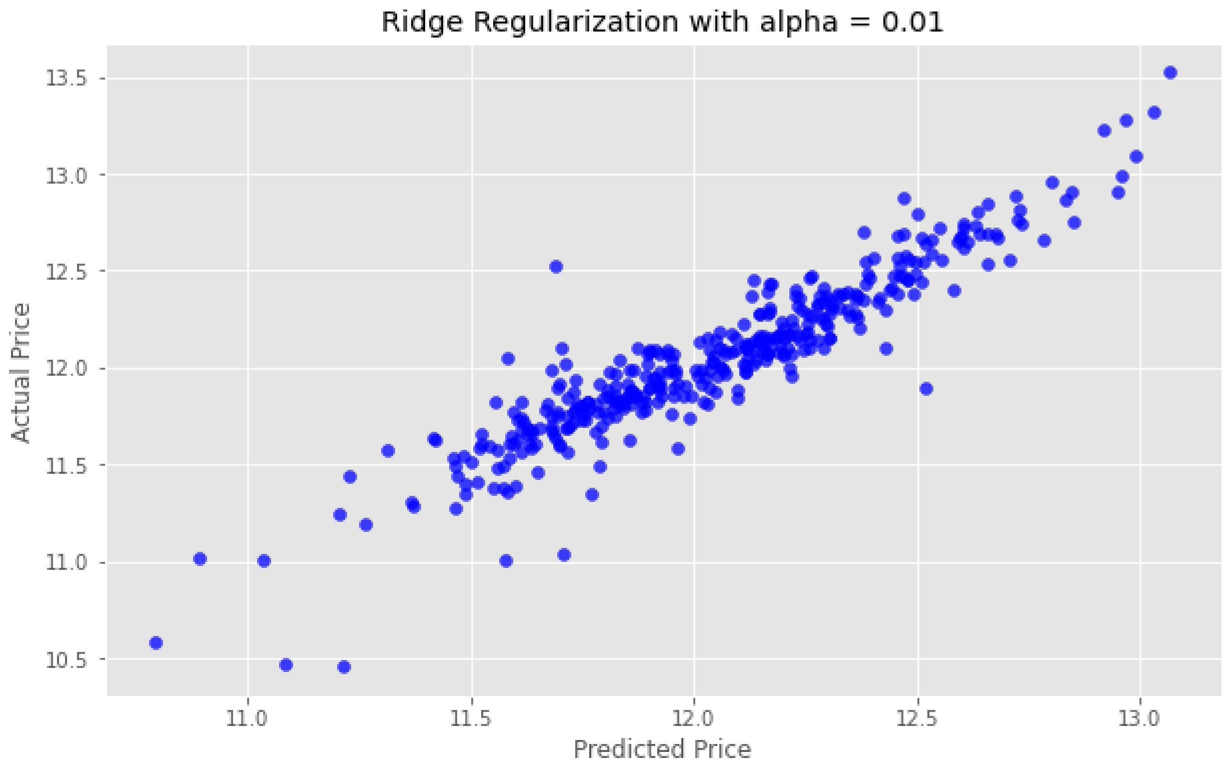


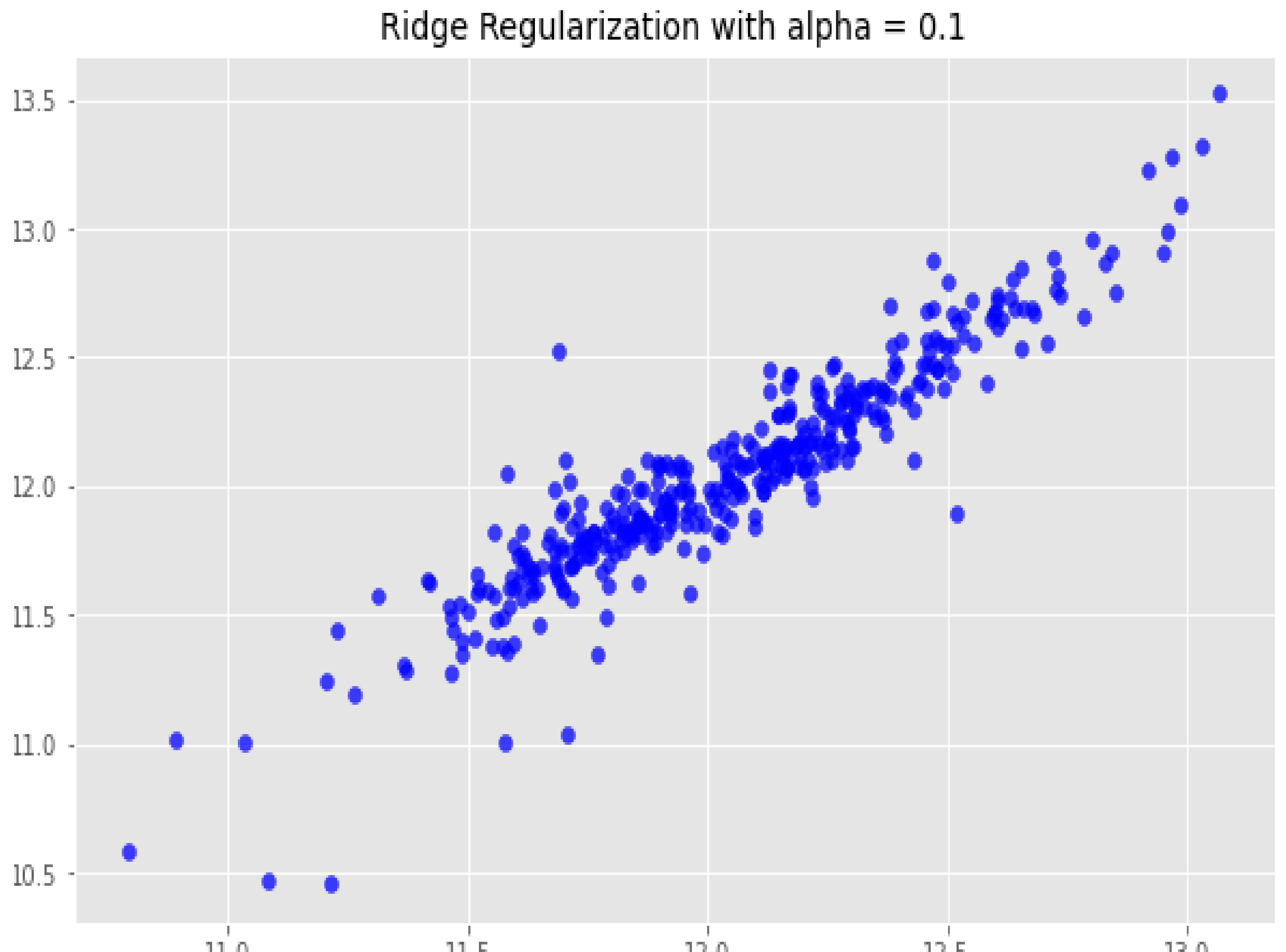
Interpreting this value is somewhat more intuitive that the r-squared value. The RMSE measures the distance between our predicted values and actual values.

We can view this relationship graphically with a scatter plot.

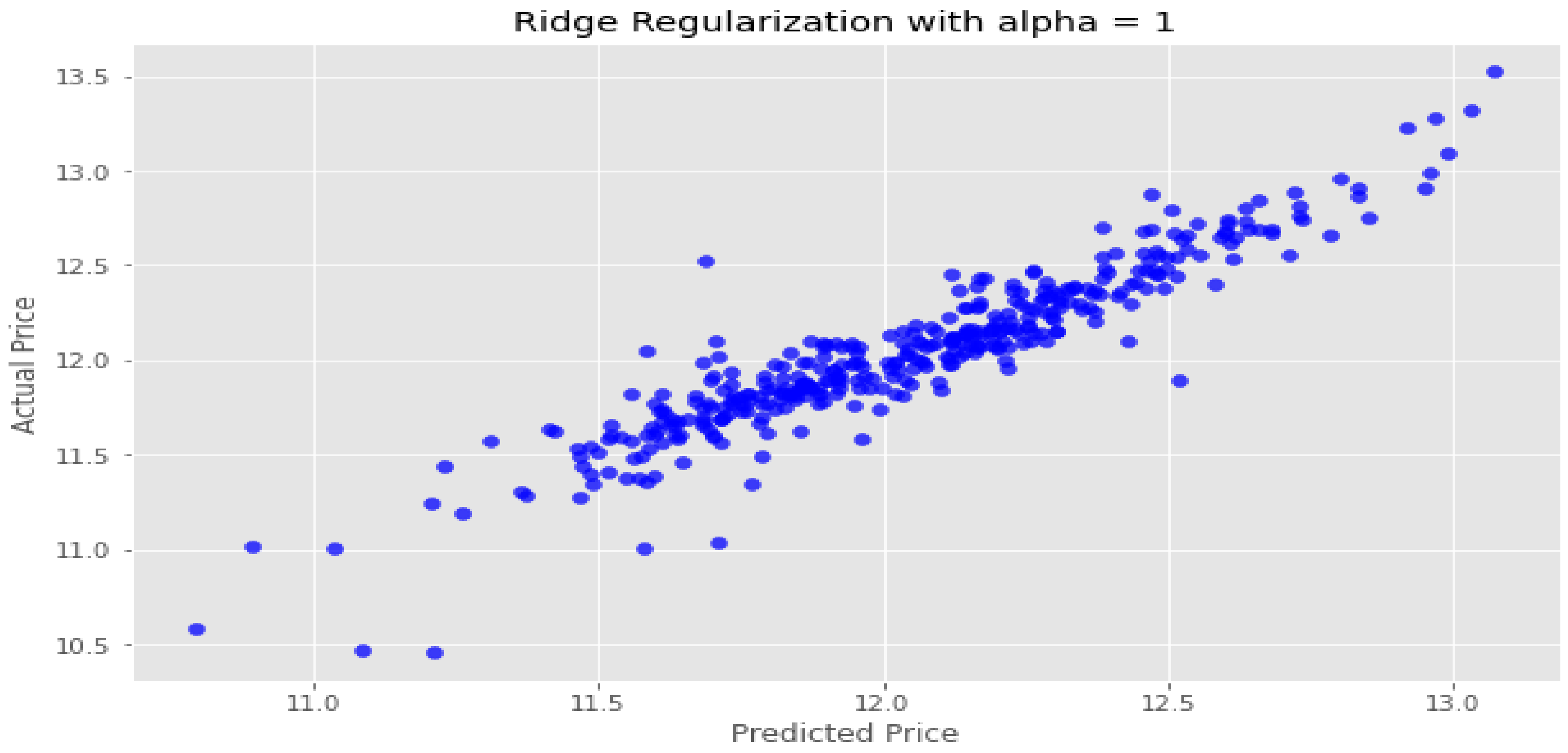


We'll once again instantiate the model. The Ridge Regularization model takes a parameter, alpha , which controls the strength of the regularization.
We'll experiment by looping through a few different values of alpha, and see how this changes our results.

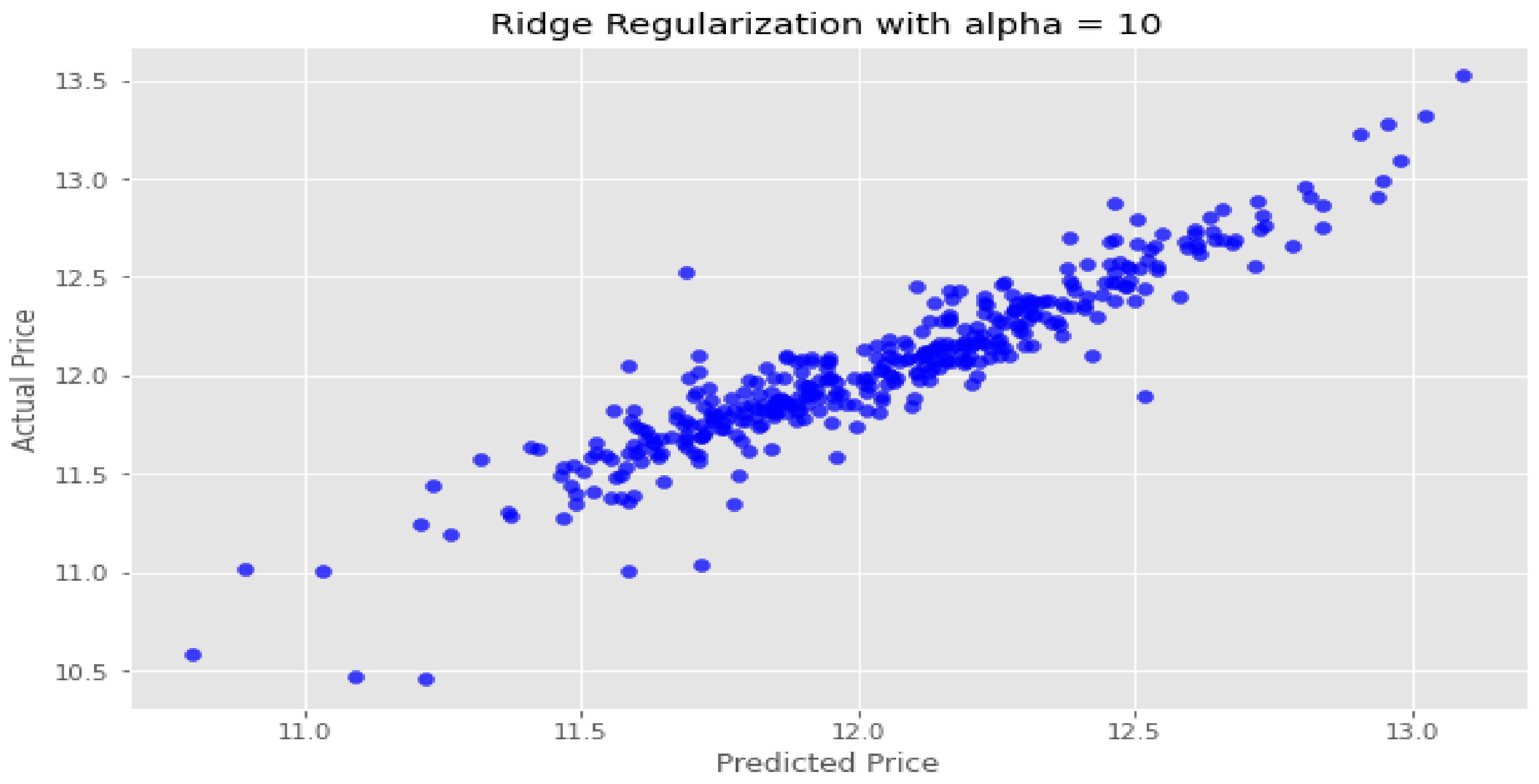



Ridge Regularization with alpha = 0.1
13.5
13.0
12.5
12.0
11.5
11.0
10.5
11.0
11.5
12.0
12.5
13.0

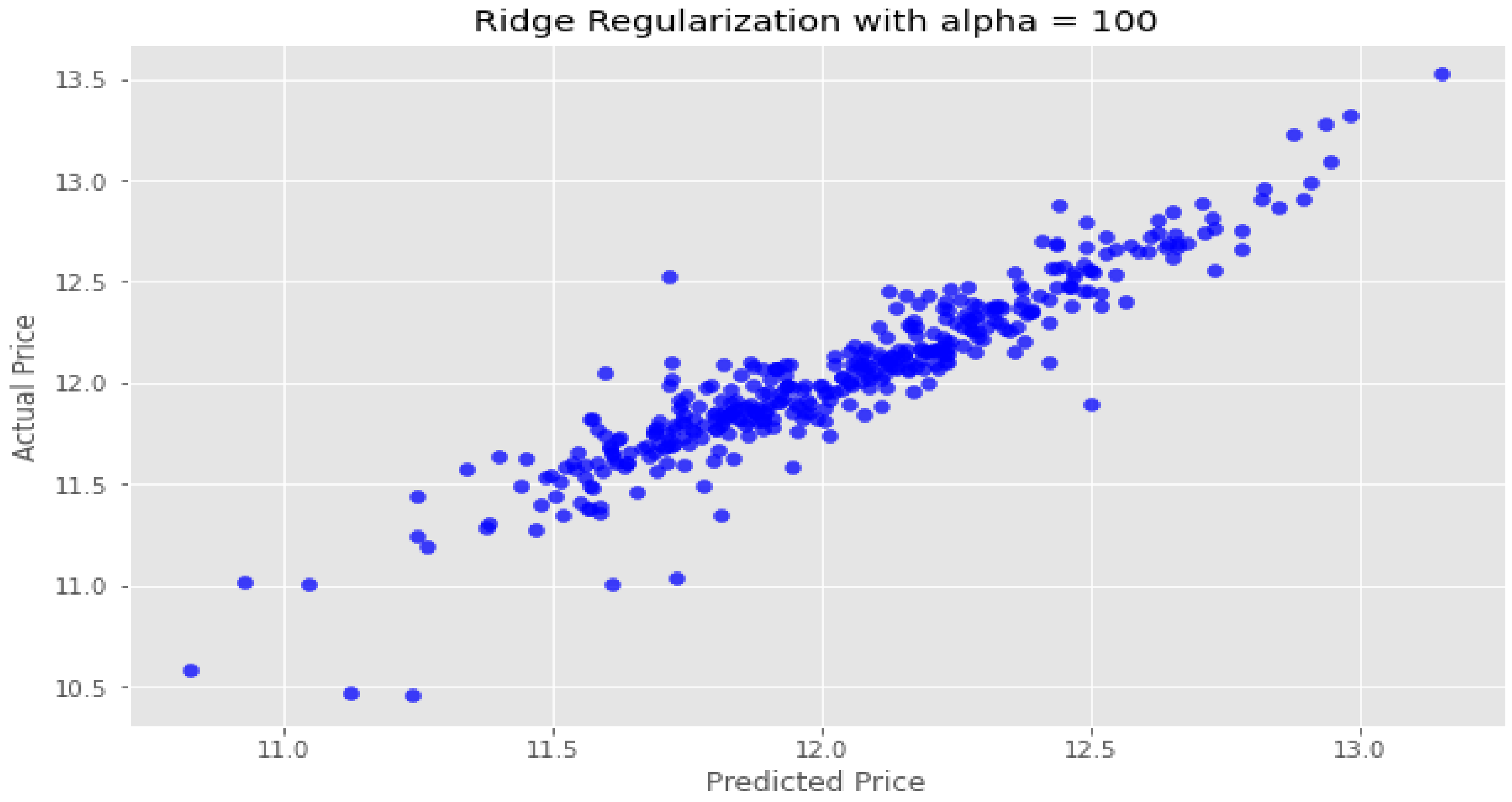
alpha = 1


Ridge Regularization with alpha = 1
Actual Price
Predicted Price
13.5
13.0
12.5
12.0
11.5
11.0
10.5
11.0
11.5
12.0
12.5
13.0

alpha = 10

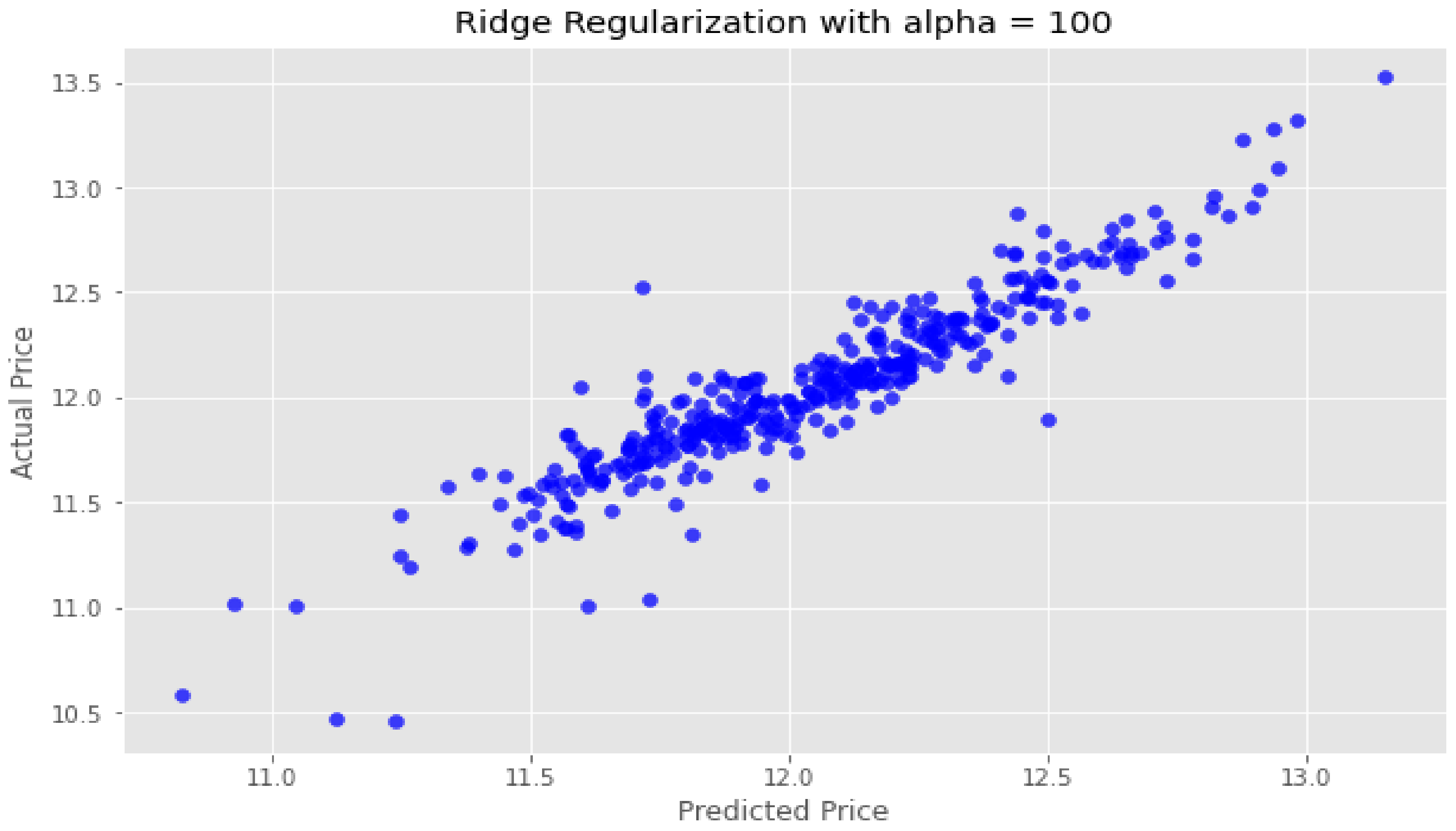

Ridge Regularization with alpha = 10
Actual Price
Predicted Price
13.5
13.0
12.5
12.0
11.5
11.0
10.5
11.0
11.5
12.0
12.5
13.0

alpha = 100


Ridge Regularization with alpha = 100
Actual Price
Predicted Price
13.5
13.0
12.5
12.0
11.5
11.0
10.5
11.0
11.5
12.0
12.5
13.0

sublesson1.csv

- `we've got the data arranged in the proper format, we can export to a .csv file. We pass index=False because Pandas otherwise would create a new index for us.`
- We've created a file called sublesson1.csv in our working directory that conforms to the correct format. Go to the sublesson page to make a sublesson.
- We placed 1602 out of about 2400 competitors. Almost middle of the pack, not bad! Notice that our score here is .15097, which is better than the score we observed on the test data. That's a good result, but will not always be the case.


Ridge Regularization with alpha = 100
Actual Price
Predicted Price
13.5
13.0
12.5
12.0
11.5
11.0
10.5
11.0
11.5
12.0
12.5
13.0