3. Modeling:
Now that the data acquired is wrangled properly, we need to divide our dataset into training and test sets and then begin the part where the models will be trained. Since our goal is to predict the price, which is a continuous variable, we have used Regression models for this project.

A list called features_list was used to hold the column names that we would like to use to predict the target variable. Then, df.price.quantile(.95) was also used to subset the data frame to only keep the top 95% of prices. The target variable has a few outliers and this should get rid of them. The features_list was then used to subset the columns that we want to keep on the new data frame.

We have also used the 'One Hot encoding' technique where we have changed the categorical columns to numeric columns, since Machine Learning algorithms can only predict on numeric data. To accomplish this, we used Pandas .get_dummies() function. After transforming our categorical columns, we add MinMaxScaler() function to scale and transform the numerical columns. Many machine learning algorithms work better when features are on a relatively similar scale and close to normally distributed. Scale generally means to change the range of the values. The shape of the distribution doesn't change. The range is often set at 0 to 1.

In this problem, we decide to scale the quantitative features using MinMaxScaler. For each value in a feature, MinMaxScaler subtracts the minimum value in the feature and then divides by the range. The range is the difference between the original maximum and original minimum. MinMaxScaler preserves the shape of the original distribution. It doesn't meaningfully change the information embedded in the original data. Also, it doesn't reduce the importance of outliers. The default range for the feature returned by MinMaxScaler is 0 to 1.

There are a couple of models which we have used for the detection of the price. We have used Linear Regression as the baseline model to compare with the other models to see how it performs.

- Linear Regression
- Random Forest: consists of an ensemble of Decision Trees. It uses a method called 'Bagging' for sampling the data and aggregates all the Decision trees into a good performing model. The Random Forest Model without any hyperparameter tuning was performing better than the other models.
- RMSE: It measures the average error performed by the model in predicting the outcome for an observation.
- R^2 score: This corresponds to the squared correlation between the observed outcome values and the predicted values by the model. The higher the R-squared, the better the model.

The Training and Testing data have been split into 70% and 30% respectively using Machine Learning 'sci-kit learn' library. After splitting the data, we used our first model which was the Decision Tree Regressor. We set a few hyperparameters for this model but later on we will use the GridSearchCV to find the best_params_ for this model.

The R^2 score for our original Decision Tree model was 0.26. The coefficient R^2 is defined as (1 - u/v), where u is the residual sum of squares ((y_true - y_pred) ** 2).sum() and v is the total sum of squares ((y_true - y_true.mean()) ** 2).sum(). The best possible score is 1.0. Hyperparameter tuning is very important for the performance of the model which we use to train it. After hyperparameter tuning the Decision Tree model we get a R^2 score of 0.51.

Next, we use our Random Forest Regressor model.

- First, we have implemented a 'GridSearch Cross Validation' to select the best hyper-parameter for the Random Forest Regressor.
- Then those parameters were used for the model to train and predict the price. We get an R^2 score of 0.59.

The Random Forest Regressor has been the best model so far but we decided to use the XGBRegressor model as well. After hyperparameter tuning the XGBRegressor model, we get a R^2 score of 0.61.

From the XGBRegressor model, we can have the important features which are crucial for determining the price. Below you can see these features based on their importance level.