**Business Problem**

**Renting a house or an apartment is never easy. Whether you are a college student or a working professional, renting a place always seems like a daunting task that is often impulsive or risky. Rent is influenced by several factors.**

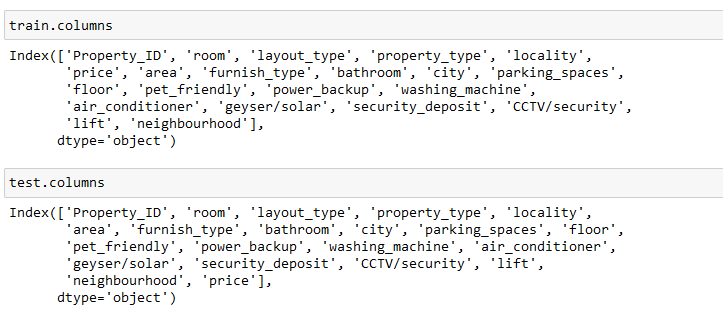
**In this challenge, participants will predict the house-rents using data science methods, machine learning, and hyper parameter tuning.**

**Approaching for the Solution**

1. **Data Understanding**
2. **Dimensions of the dataset**

Train: 134683 rows x 21 columns

Test: 57722 rows x 20 columns



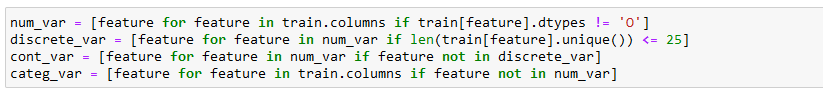
1. **Finding the type of variables**

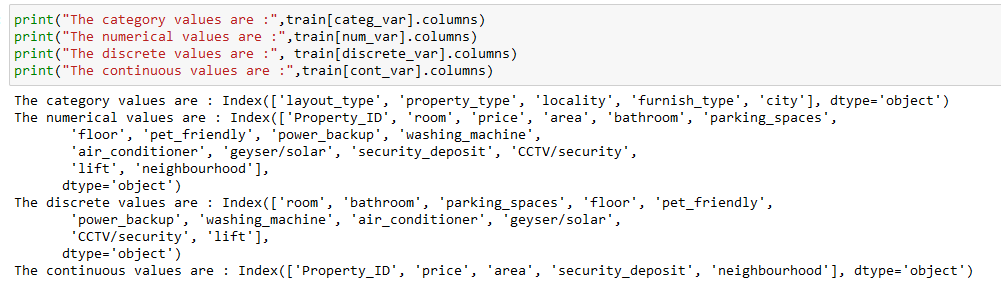
**The variables in the dataset can be classified as:**

1. **Numeric**
2. **Categorical**

**Or**

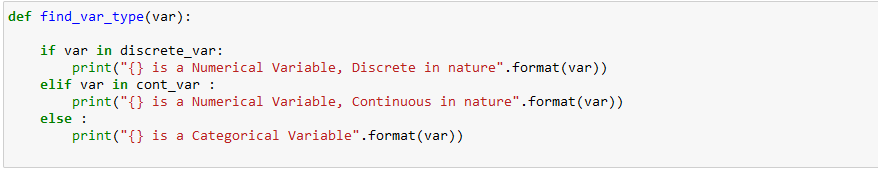
1. **Discrete**
2. **Continuous**





1. **Categorizing each variable to its type**

**Function Used:**



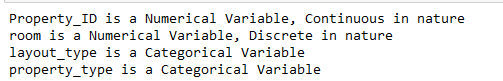
**Variable Types are : Example**

**find\_var\_type('Property\_ID')**

**find\_var\_type('room')**

**find\_var\_type('layout\_type')**

**find\_var\_type('property\_type')**



1. **Data Preparation**
2. **Handling Missing Values**

**Since the data are stored in the dataframe, the missing values can be identified as :**

1. **train.isnull().sum()**
2. **test.isnull().sum()**

**Here , in this problem , we don’t have any missing values.**

**Understanding the `missing` values :**

**Data That can be missing can be of two types :   
1) Continuous Data   
2) Discrete Or Categorical Data**

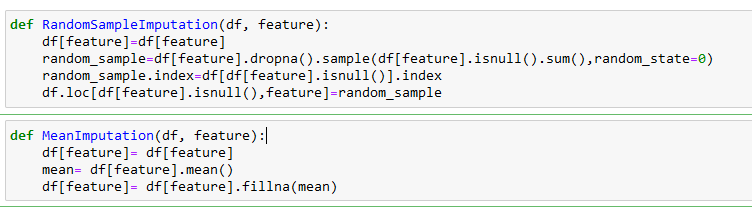
**The Types of missing can be of mentioned types:**

**1) MCAR - Missing Completely At Random   
If the probability of being missing is same for all the observations.  
2) MNAR - Missing Not At Random   
There is some relationship between the missing data   
3) MAR - Missing At Random**

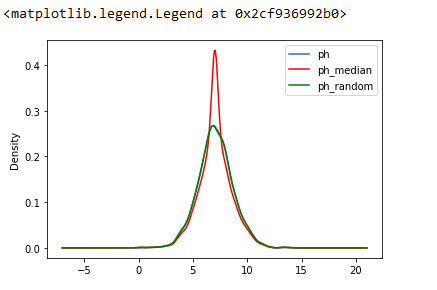
**Incase if there are missing values, we will follow the mentioned below function.**

1. **Random Sample Imputation**
2. **Mean Sample Imputation**

**Random Sample Imputation can be done when the missing values for that column will be large.**



**`KDE` graph example to validate the best imputation techniques if any:**



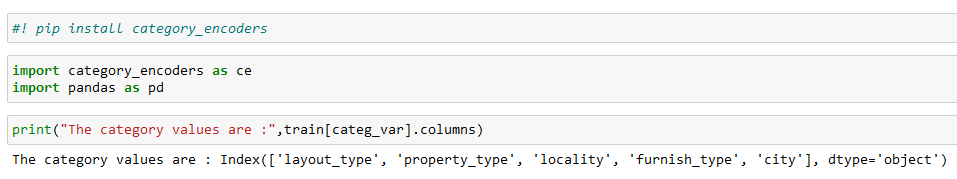
1. **Encoding**

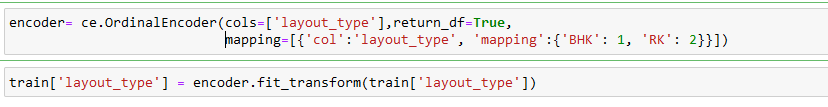
**The Text or String values is required to be converted in numerical features as a part of pre-processing steps of model building. This step is called Encoding.**

**The Category Data can be of two types:**

1. **Nominal Data :   
   Nominal data is data that can be labelled or classified into mutually exclusive categories within a variable. These categories cannot be ordered in a meaningful way.**
2. **Ordinal Data :   
   ordinal data is said to have been collected when a responder inputs his/her financial happiness level on a scale of 1-10.**

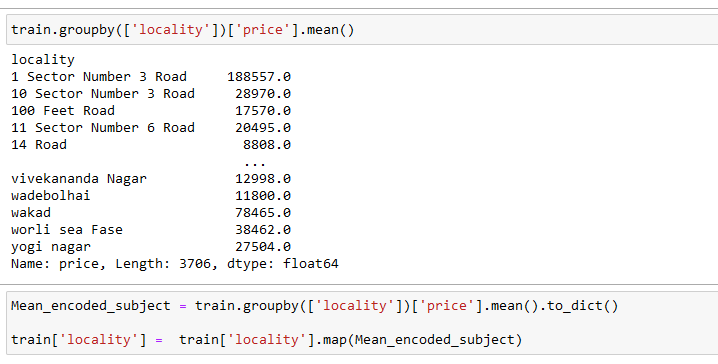
**The Category Encoding can be done by mentioned below ways:**



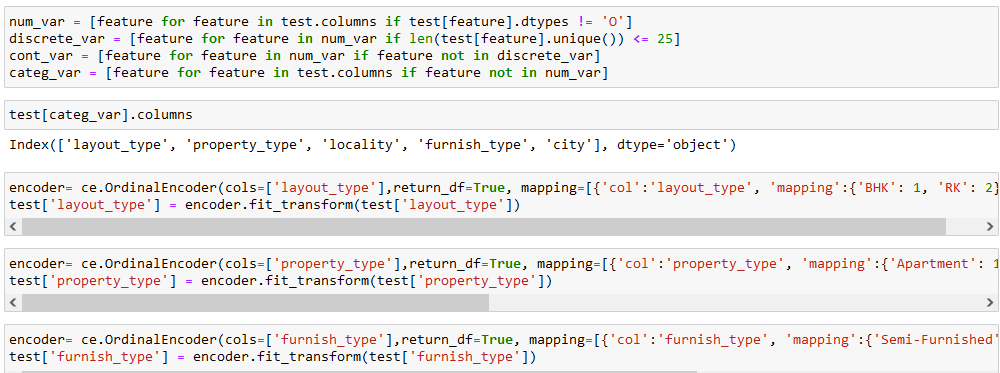


#### **For Locality and City, I can use **Label Encoding** or **Mean Encoding****

#### **Unlike **label encoding**, which gets the work done efficiently but in a random way, mean encoding tries to approach the problem more logically. In a nutshell, it uses the target variable as the basis to generate the new encoded feature.**



**Similarly, for the test dataset.**



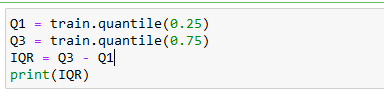
1. **Percentage of distribution of the dataset.**



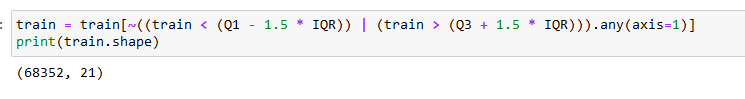
**The above step will help us to find the distribution of the columns or feature variables at various percentile levels. This also includes: mean, min, max, std, count etc. The result is displayed as shown below:**



**The Inter Quartile Range (IQR) is evaluated as:**



1. **Outliers Removal**



**Here from the above statement, we are removing the extreme values from the train data. We are preferably doing this step so that my data accuracy and majority is maintained when we will be training the model.**

1. **Data Visualization - Univariate And Multivariate Analysis**

**Univariate Analysis**

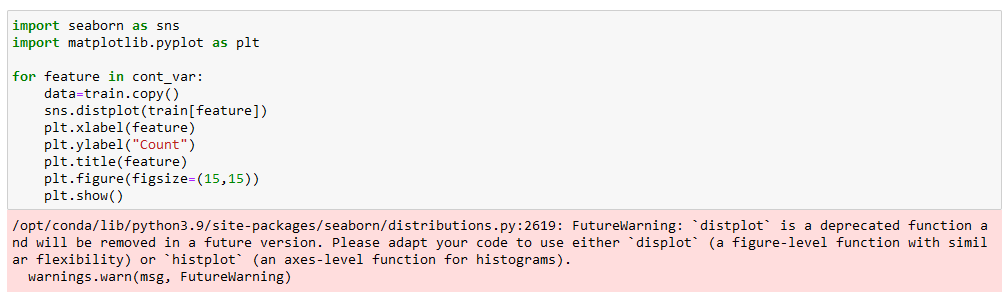
**Univariate analysis is checking the central tendency (mean, median and mode), the range, the maximum and minimum values, and standard deviation of a variable**

**For Example: Distribution of feature wrt. It’s central tendency.**

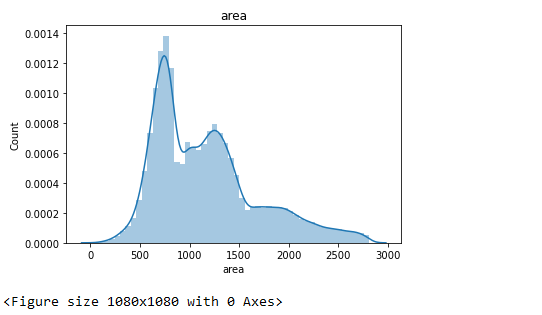
**Multivariate Analysis**

Multivariate analysis is similar to Bivariate analysis but you are comparing more than two variables.

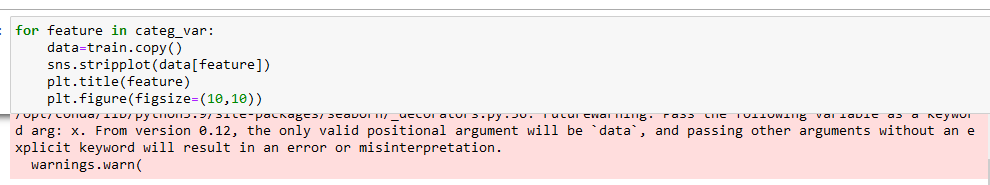
**Analysis Of the Continuous Variables:**



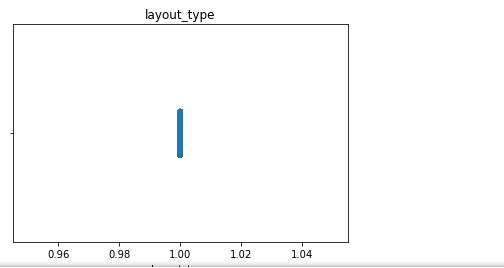
**The Univariate Distribution Graph will look like:**



### **Analysis Of Categorical Variables**

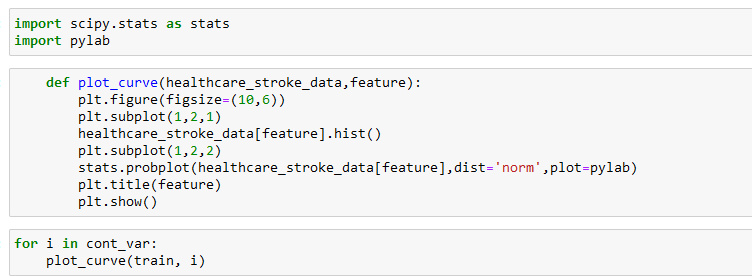


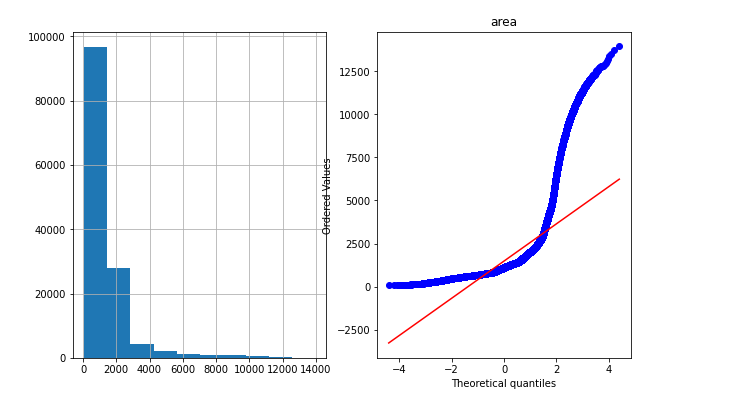
**The Univariate Distribution Graph for categorical variable will look like:**



**Vi) Skewness Vs Normalized Data :**

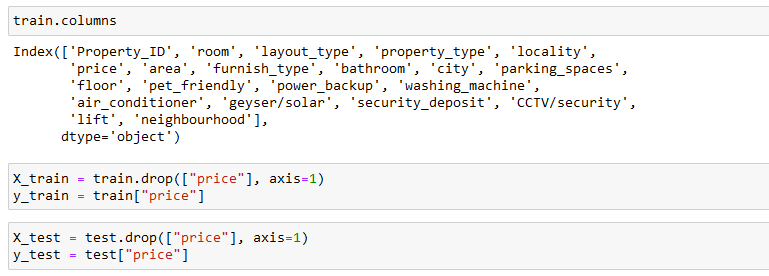
The skewness and normalization of data can be determined with the help of the Q-Q plot.





**Vii) Splitting the data with respect to it’s target/label variable :**

**This step will help us to determine the label and the feature variable for both train and test dataset.**



**Viii) Scaling and Normalization :**

Normalization is a scaling technique in which values are shifted and rescaled so that they end up ranging between (0 and 1) or (-1 and 1). The recommended techniques are:

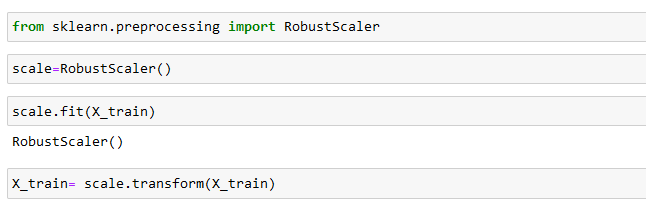
1. Standard Scaling (0 to 1)
2. Min-Max Scaling (-1 to 1)

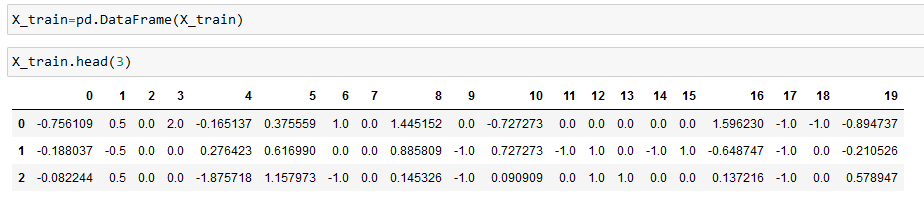
We can also use the mentioned below scaling technique as well,

**Robust Scaler**

* It is used to scale the feature to median and quantiles Scaling using median and quantiles consists of subtracting the median to all the observations, and then dividing by the interquartile difference. The interquartile difference is the difference between the 75th and 25th quantile:
* IQR = 75th quantile - 25th quantile
* Scaled = (X - X.median) / IQR

By using **Robust Scaler, we can remove the outliers during pre-processing as it works on quartiles which is not possible with Standard or Mix Max Scaling.**







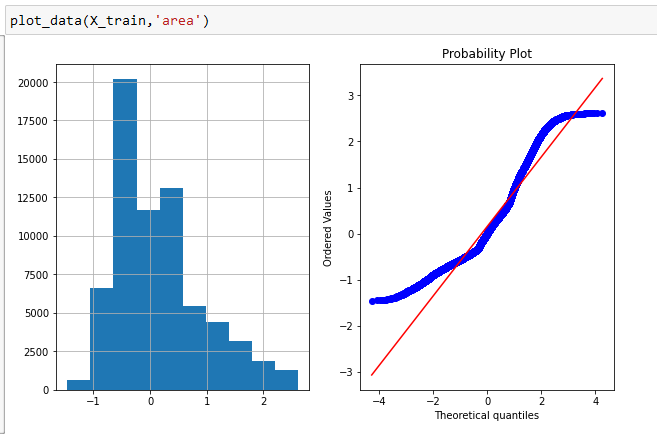
**Similar steps will be followed for the test dataset as well.**

**IX) Gaussian Transformation:**

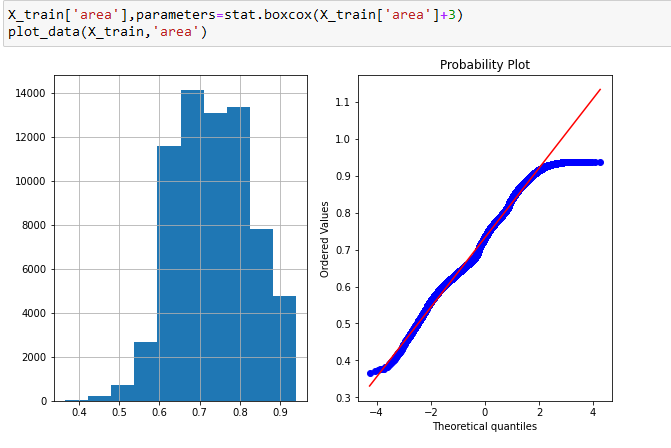
Some machine learning algorithms like linear and logistic assume that the features are normally distributed. The normal distribution helps to improve accuracy. There are some transformation techniques that can be used to achieve the required result. This includes:

* **logarithmic transformation**
* **reciprocal transformation**
* **square root transformation**
* **exponential transformation (more general, you can use any exponent)**
* **boxcox transformation**

**Original Q-Q Distribution of the feature variables – Eg : Area**



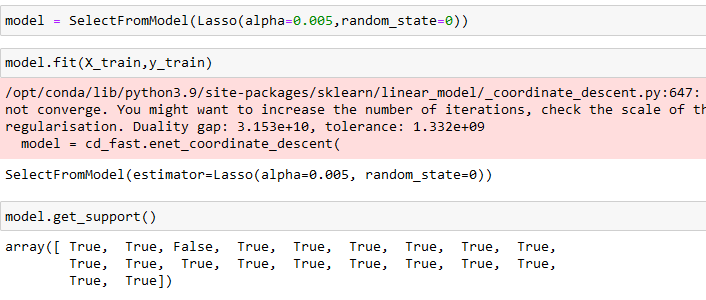
**Gaussian Transformed Q-Q Distribution of the feature variables – Eg:Area**

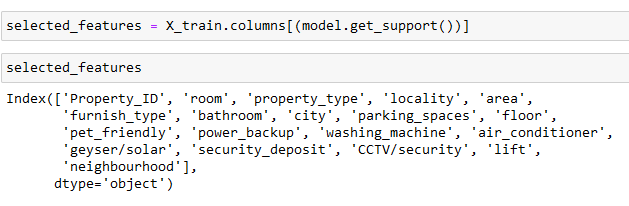


**X) Feature Selection:**

The feature selection or feature importance is the step where we try to find the importance of a particular feature with respect to the target columns.

**Lasso Techniques:**



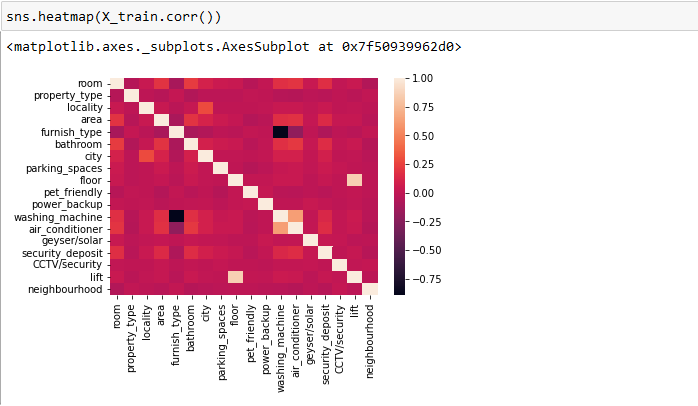


**As per the Lasso Techniques of Feature Selection, It’s discarding the feature variable `Layout Type` but if we go by the business problem, Layout Type is important in determining the rent of the apartment. (Target Variable). Hence , it is to be discussed with business before dropping it.**

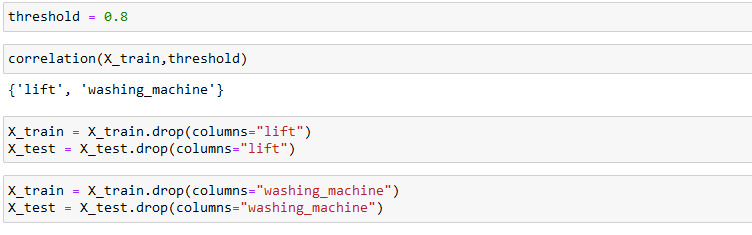
**Xi) Checking the Multi-collinearity**

Multicollinearity is a statistical concept where several independent variables in a model are correlated. Two variables are considered to be perfectly collinear if their correlation coefficient is +/- 1.0. Multicollinearity among independent variables will result in less reliable statistical inferences

The **multicollinearity** can be determined using **Heat Map.**



**Here We considered, the Threshold value as 0.8**



**Important Note :**

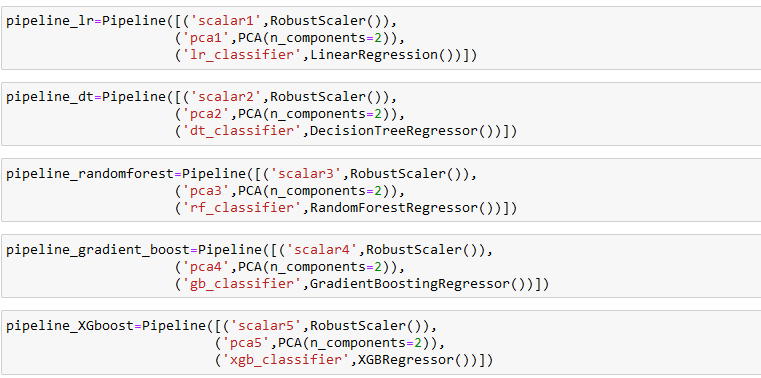
**It’s true that the `multi-collinearity` relationship among the feature variables will result in less statistical inference. So ideally we should drop the variables but at the same time the independent variables like {‘lift’ , ‘washing machine’ } are very important in determining the ‘rent’ of the apartment. So, it is a business take or decisions to be considered before dropping the variables.**

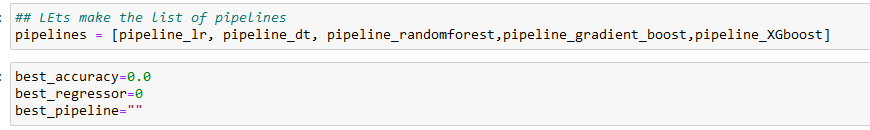
**In this case, I tried using both by dropping and without dropping the variables.**

1. **Model Building & Evaluation**
2. Cross-validation is **a resampling procedure used to evaluate machine learning models on a limited data sample**. It is mainly used in settings where the goal is prediction, and one wants to estimate how accurately a predictive model will perform in practice.

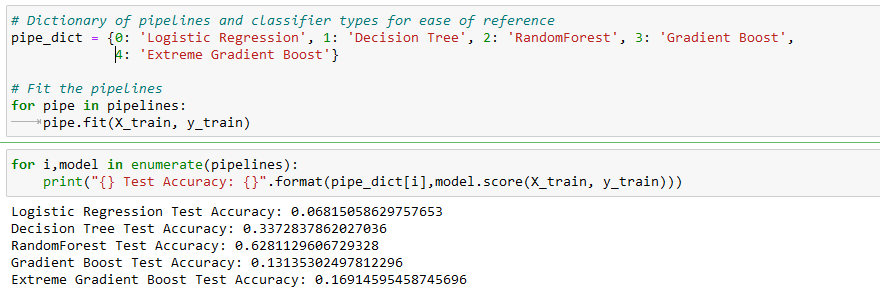
This technique will help you to estimate the best model can be used for the dataset.

**Pipeline Creation:**





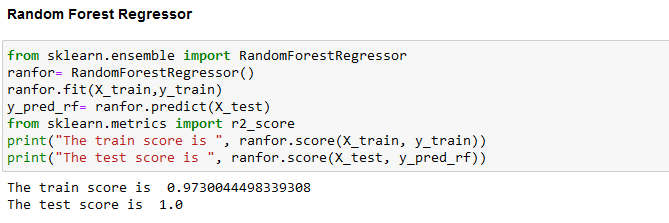
**Test Accuracy on CV data:**



1. **Evaluation Metrics for the regression problem**

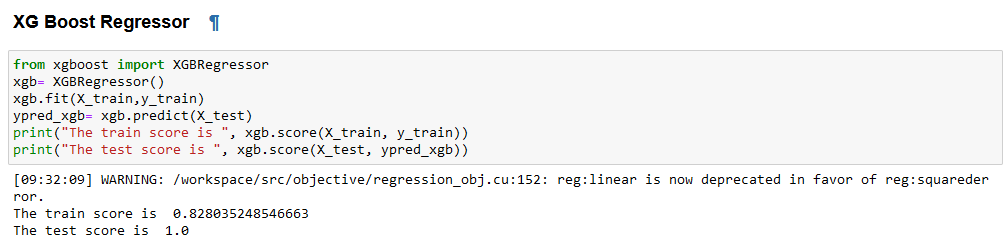
Mean Squared Error represents the average of the squared difference between the original and predicted values in the data set. It measures the variance of the residuals.





The mean squared error is 872153646.5421617

The rmse is 29532.2475701082



The mean squared error is 840575135.7427963

The rmse is 28992.673828793308

1. **Hyper Parameter Tuning**

**#! pip install scikit\_optimize**

Hyperparameter tuning (or hyperparameter optimization) is **the process of determining the right combination of hyperparameters that maximizes the model performance**. It works by running multiple trials in a single training process.

There are two types of hyper parameter tuning :

1. Grid Search

2. Random Search

When my dataset is large, this can lead to time complexity while performing the hyperparameter tuning. In such case, Bayes Search CV should be used.

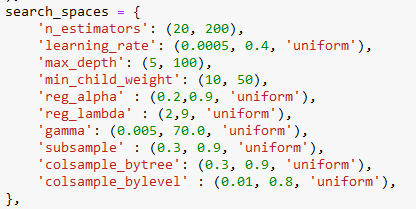
**BayesSearchCV** implements a “fit” and a “score” method. It also implements “predict”, “predict\_proba”, “decision\_function”, “transform” and “inverse\_transform” if they are implemented in the estimator used.

The parameters of the estimator used to apply these methods are optimized by cross-validated search over parameter settings.

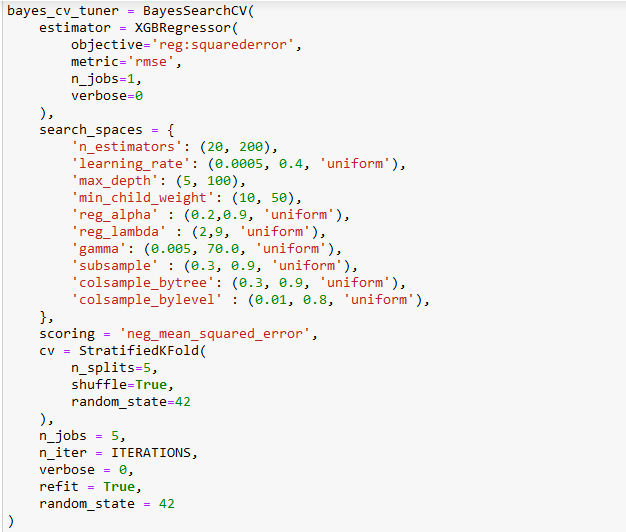
In contrast to GridSearchCV, not all parameter values are tried out, but rather a fixed number of parameter settings is sampled from the specified distributions. The number of parameter settings that are tried is given by n\_iter.

**ITERATIONS = 4 # 1000**

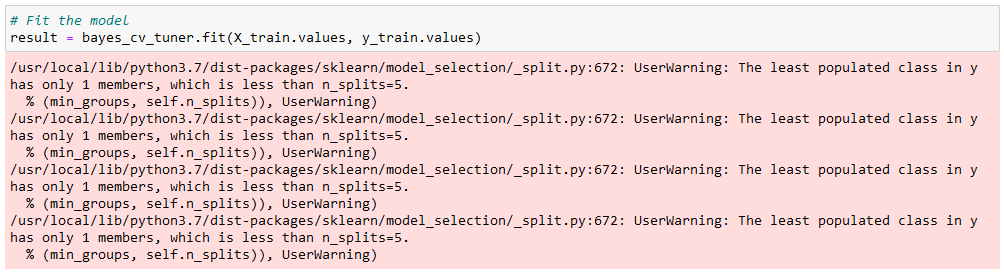
**SEARCH SPACE USED:**



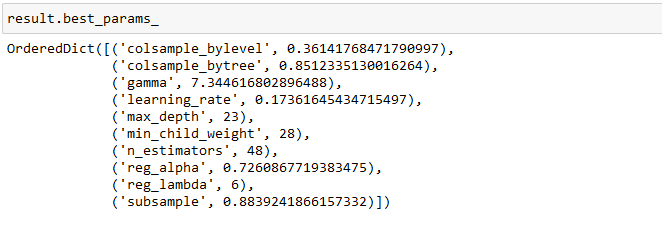
While using the Bayes Search CV – n\_jobs and the n\_iters should be mentioned.



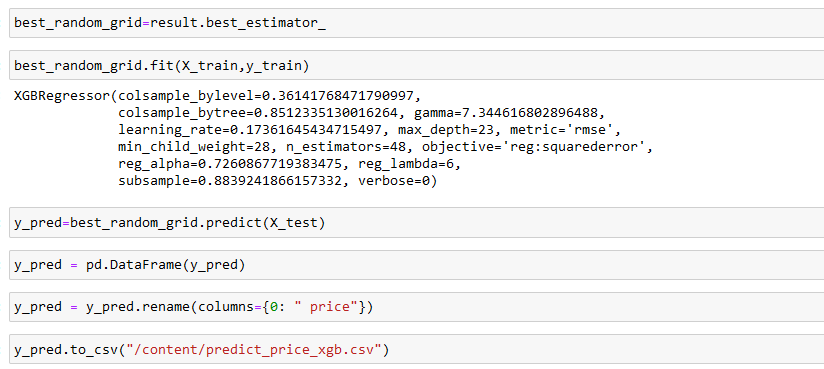
Fitting the Bayes Search CV :



The Best params selected are:



The Final prediction can be done with the help of the params selected.

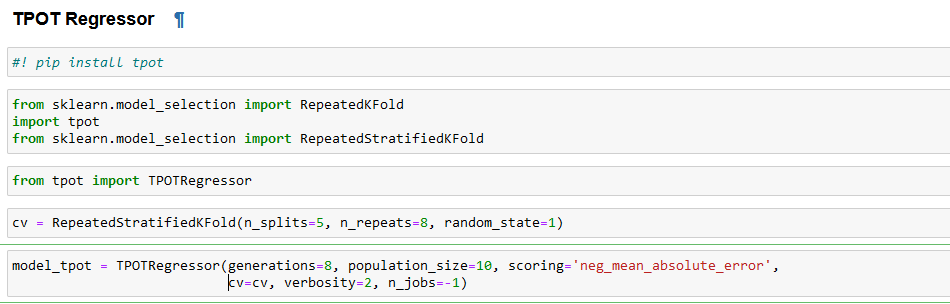


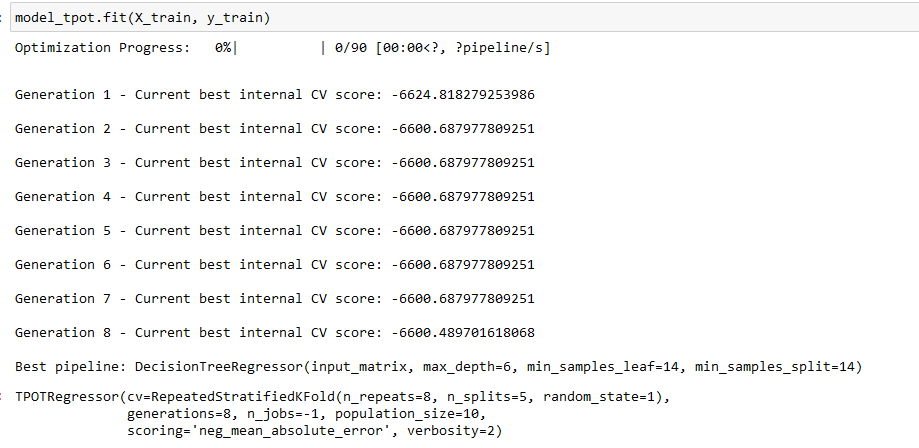
**IV) Automated Hyper Parameter Tuning:**

### **TPOT Regressor**

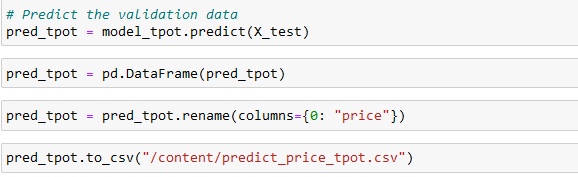
TPOT uses a tree-based structure to represent a model pipeline for a predictive modeling problem, including data preparation and modeling algorithms and model hyper parameters.

An optimization procedure is then performed to find a tree structure that performs best for a given dataset. Specifically, a genetic programming algorithm, designed to perform a stochastic global optimization on programs represented as trees.



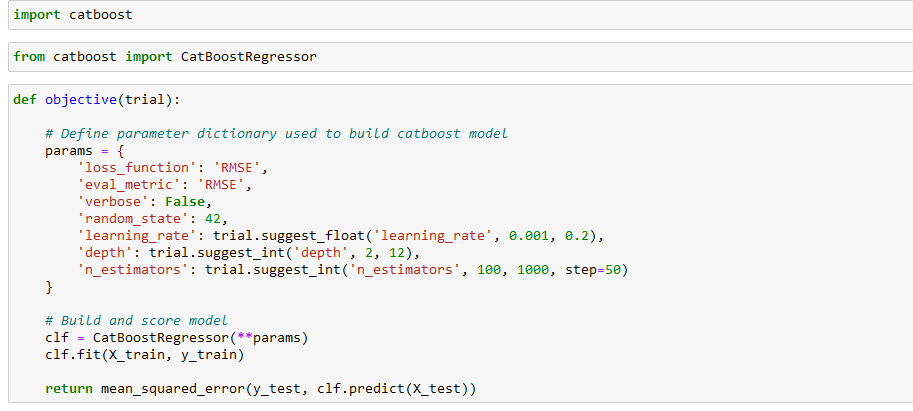


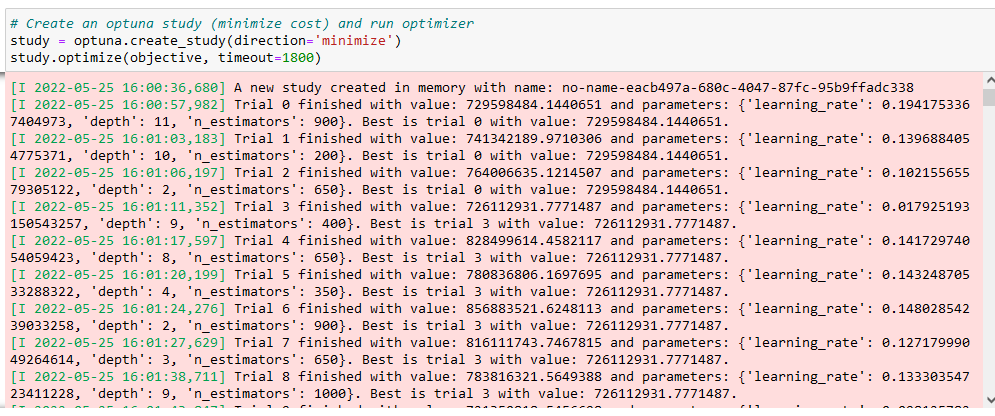
**Predicted dataset after using TPOT :**



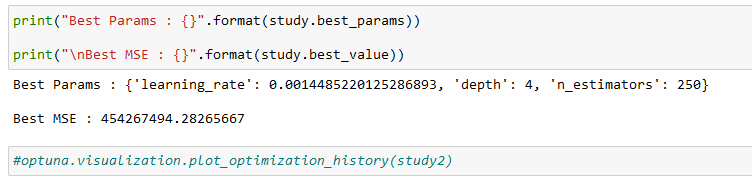
**OPTUNA :**

Optuna is a software framework for automating the **optimization process of these hyperparameters**. It automatically finds optimal hyperparameter values by making use of different samplers such as grid search, random, bayesian, and evolutionary algorithms.

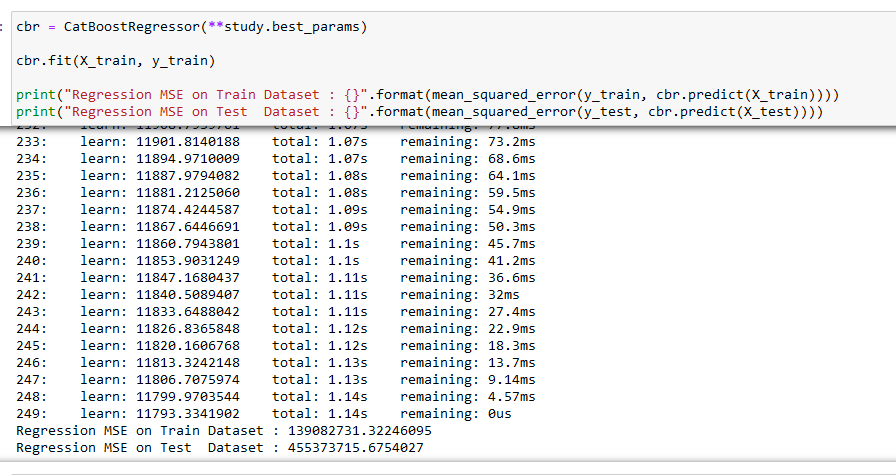


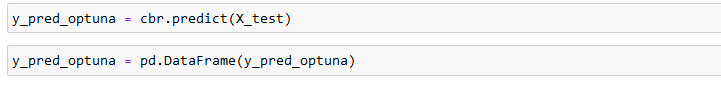


**Best Params and MSE Selected :**



**CATBOOST Regressor Prediction Using Optuna :**



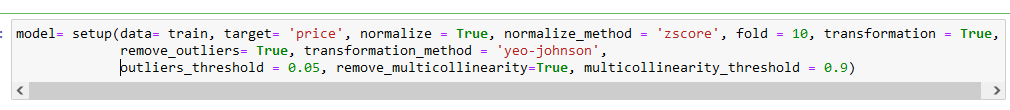


1. **Using Auto ML – PyCaret And Explainable AI**

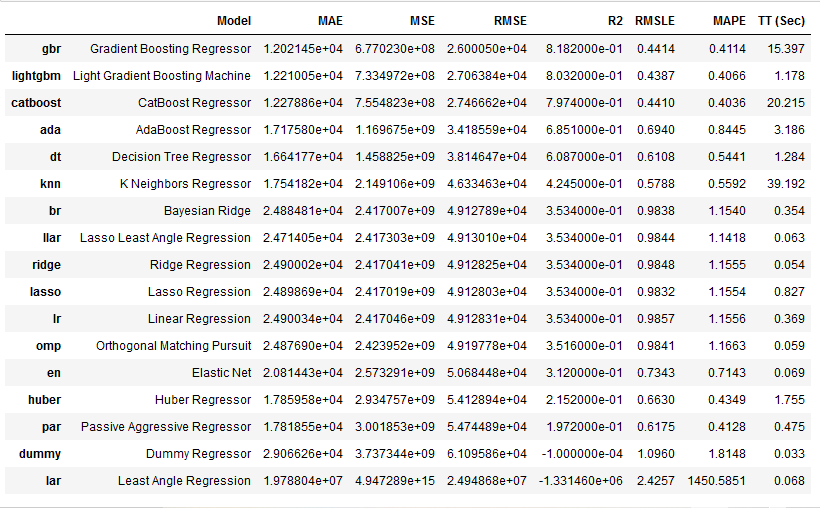
### PyCaret is an open source, low-code machine learning library in Python that allows you to go from preparing your data to deploying your model within minutes in your choice of notebook environment.



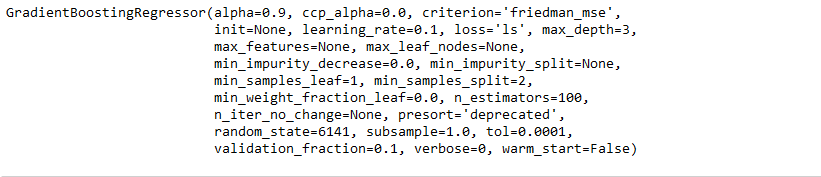
Model Setup:



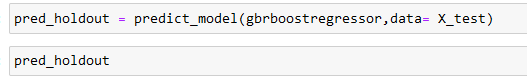
Comparison of Various Models:



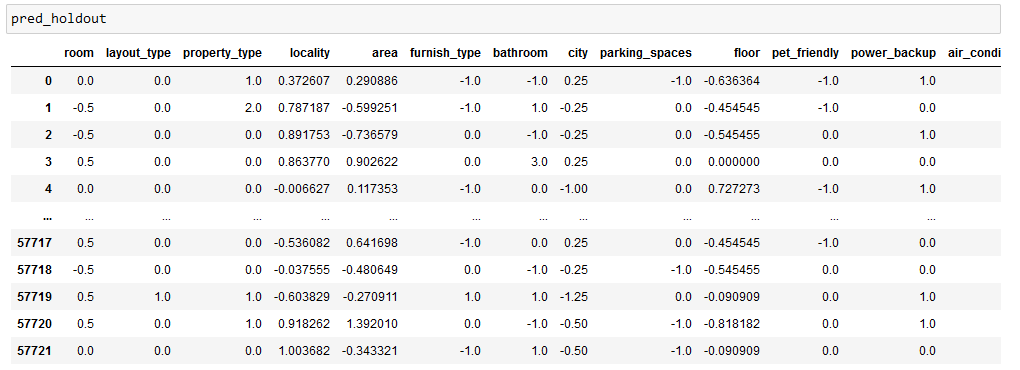
Final Selection of the Model performing best:



Final Prediction:



**The DataFrame looks like :**



**Explainable AI**

**Explainable AI – xAI** can perform all the required steps in correct fashion by not only building the model but also explaining the model and it’s feature performances even before the result is produced.

For all the independent columns that we are passing in the model, xAI can successfully identify the features influence over the desired result and correcting the parameters to be predicted as inappropriate for the dataset that we are using.

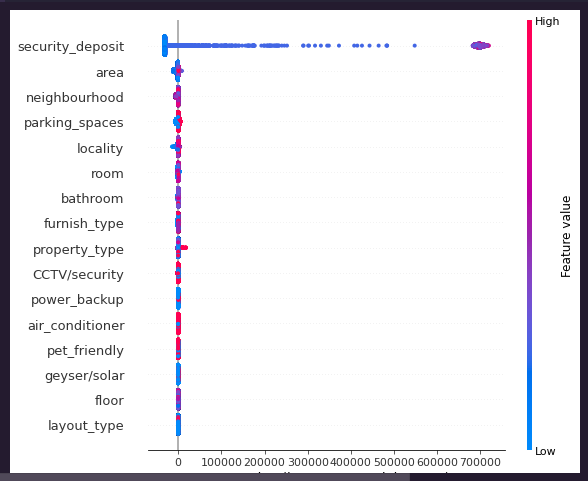
Explainable AI is a set of tools and frameworks to help you understand and interpret predictions made by your machine learning models, natively integrated with a number of Google's products and services. With it, you can debug and improve model performance, and help others understand your models' behavior.

Few of the Libraries that can be used includes,

1. LIME
2. YELLOWBRICKS
3. SHAP
4. Eli5
5. Interpret ML

**SHAP Summary Plot**

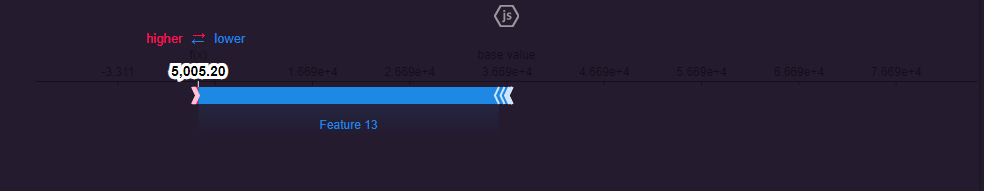
Summary plots are easy-to-read visualizations which bring the whole data to a single plot. All of the features are listed in y-axis in the rank order, the top one being the most contributor to the predictions and the bottom one being the least or zero-contributor. Shap values are provided in the x-axis. As we discussed already, a value of zero represents no contribution whereas contributions increase as the shap value moves away from zero. Each circular dot in the plot represents a single data point. Color of the dot denotes the value of that corresponding feature. It can be observed that the feature ‘worst perimeter’ contributes greatly to the model’s prediction with low values deciding one class and higher values deciding the other.



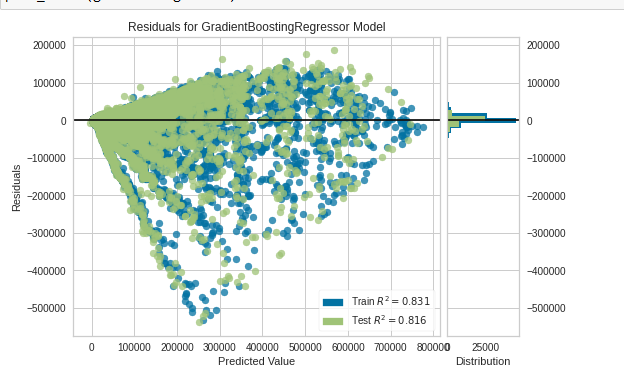
**SHAP Force Plot**

Develop a tree-based SHAP explainer and calculate the shap values. Shap values are arrays of a length corresponding to the number of classes in target. Here the problem is binary classification, and thus shap values have two arrays corresponding to either class.

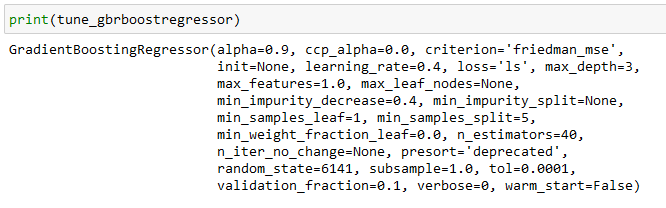
Shap values are floating-point numbers corresponding to data in each row corresponding to each feature. Shap value represents the contribution of that particular data point in predicting the outputs. If the shap value is much closer to zero, we can say that the data point contributes very little to predictions. If the shap value is a strong positive or strong negative value, we can say that the data point greatly contributes to predicting the positive or negative class.   
Force plots are suitable for row-wise SHAP analysis. It takes in a single row and shows in a rank order how each of the features contributed to the prediction. Wider a feature’s block, more the contribution.



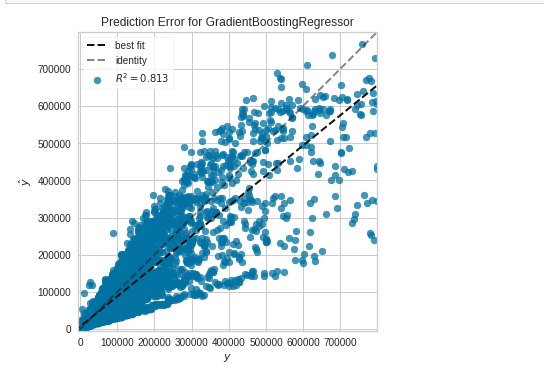
**Explainable AI using PyCaret**



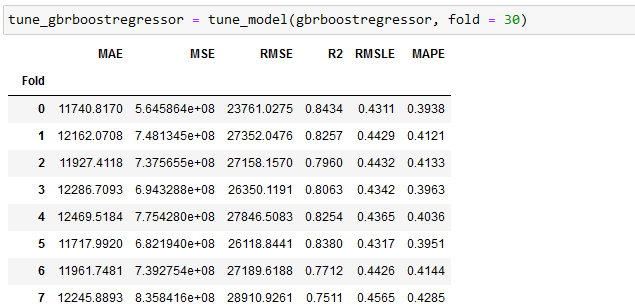
Tuning The GBR Model:



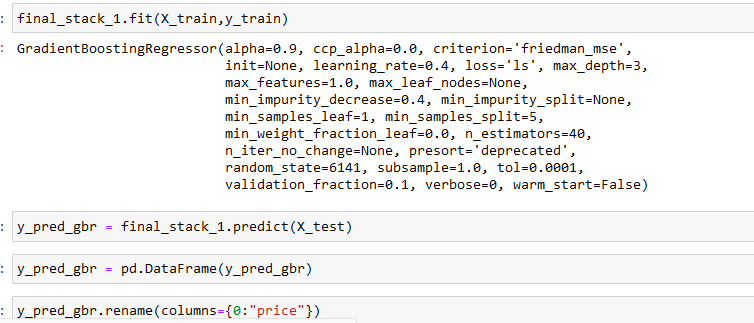
**Plotting The Error Model :**



**Model Tuning:**



The Final Stack Data:

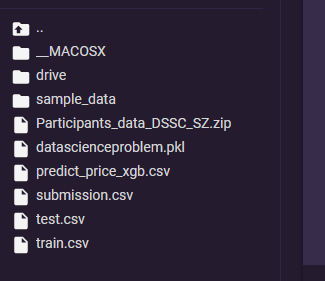


1. **Results And Recommendations**

Pickle file in the binary file that will be used for the model deployment. Please find the steps below to create this file.



The Downloaded Pickle File is :



**Steps Required to be performed for Model Deployment are:**

Creating GUI or **Web Interactive Form** for considering the input variables and evaluating it for the predicted output.

**This will include:**

* **1. Web UI – In HTML or CSS**
* **2. Procfile – Explaining where it is to be deployed**
* **3. Requirement.txt – it should consist of the python libraries name and its versions used**
* **4. app.py – that can be used to extract the values from the web form and the calculate it by using the pickle file. It uses the GET and POST method.**
* **5. Pickle File (.pkl) – The binary file of the model created. This file is required to be present in the web server.**
* **6. Platform - Web Server or Cloud Server – like Heroku, Render**
* **7. DNS - .com or .edu for which the Web ML is used.**