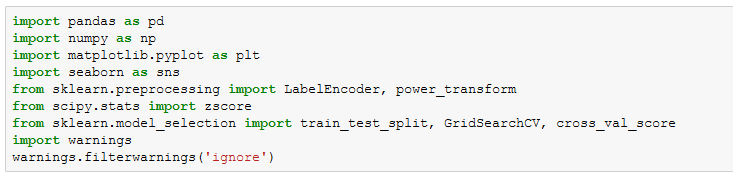
**Predicting the primary fuel and power plant capacity using the Global Power Plant Dataset**

In this post, I will be going through the whole process of building a Machine Learning model on the subset of Global power plant dataset. Here we are predicting two variables i.e., **capacity\_mw** and **primary\_fuel**

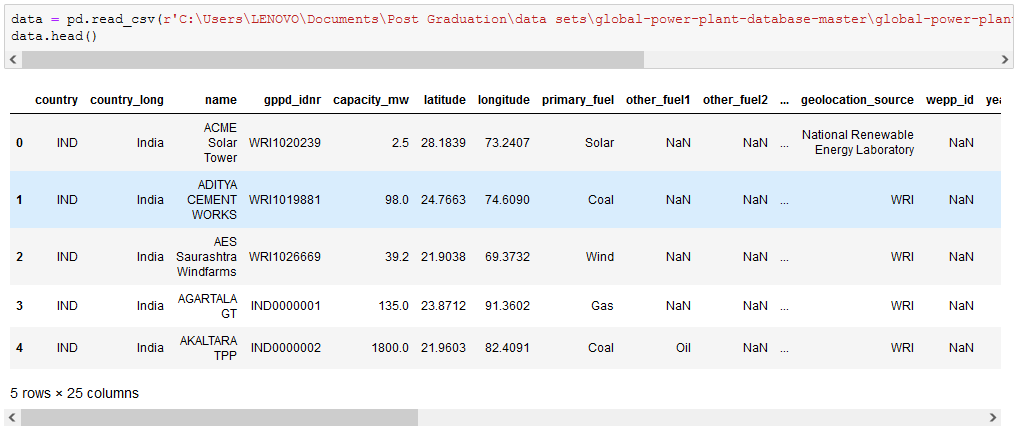
The dataset contains a details of the power plants in India, which is a subset of Global Power Plant Database. It is a comprehensive, open source database of power plants around the world. It centralizes power plant data to make it easier to navigate, compare and draw insights for one’s own analysis. The database covers approximately 35,000 power plants from 167 countries and includes thermal plants.

Since we are using the data to predict multiple dependent variables, we will be considering one dependent variable as independent while predicting the other and vice versa.

Before we can proceed, I’m importing necessary libraries

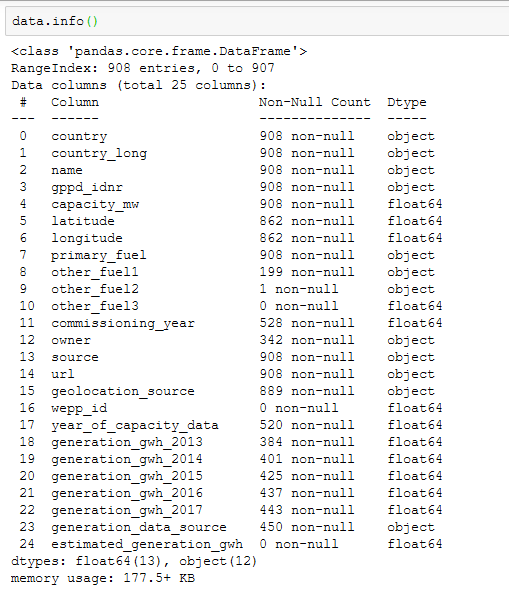


Importing the dataset and looking at a glimpse of few variables



Right away we can see that the data has null values and needs to be treated. Checking for the shape of the data using **.shape** function and I see that there are 908 rows and 25 columns

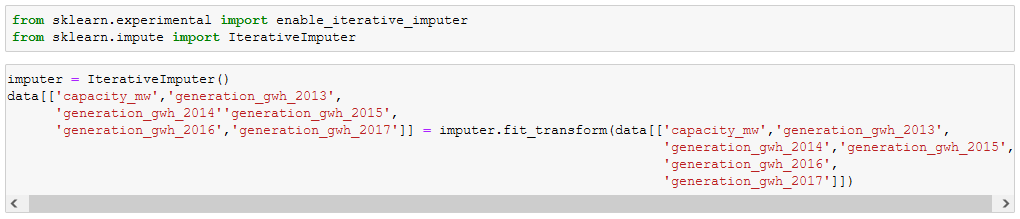
Checking for null values in the dataset and also verifying the data types of the variables.

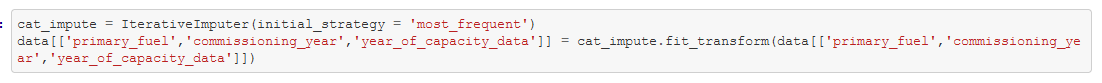


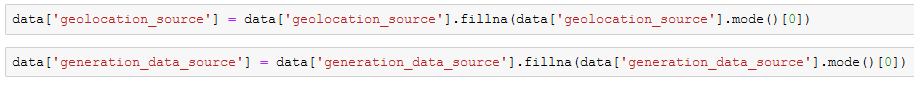
I can see from the above figure that some of the features like **'other\_fuel1', 'other\_fuel2', 'other\_fuel3', ‘wepp\_id', 'estimated\_generation\_gwh'** are missing more than 80% data, the column **'owner', 'name', 'url' and 'gppd\_idnr'** is unique for each row and **'country\_long', 'country'** is same for all rows. Therefore they will not help us in analysing the capacity or the fuel type. Therefore dropping the same.



However, I can still see there are variables with more than 50% null values. We can treat the same using imputers, I’m using Iterative Imputer for the variables in the dataset.

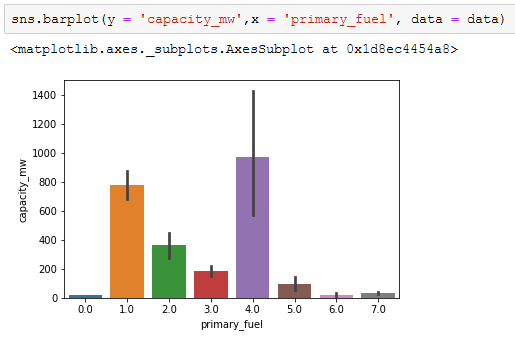




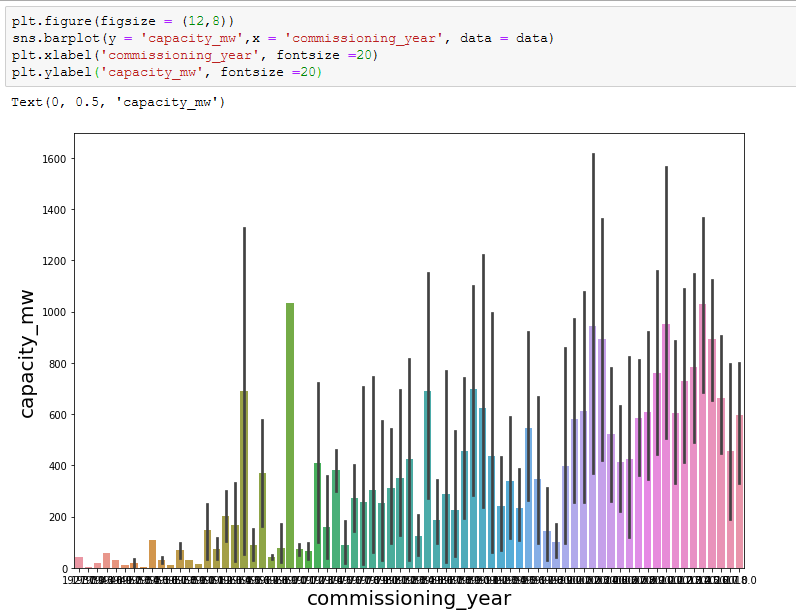


Here I’m imputing all the numerical type with Iterative Imputer and replacing the categorical variable with ‘mode’ of the dataset.

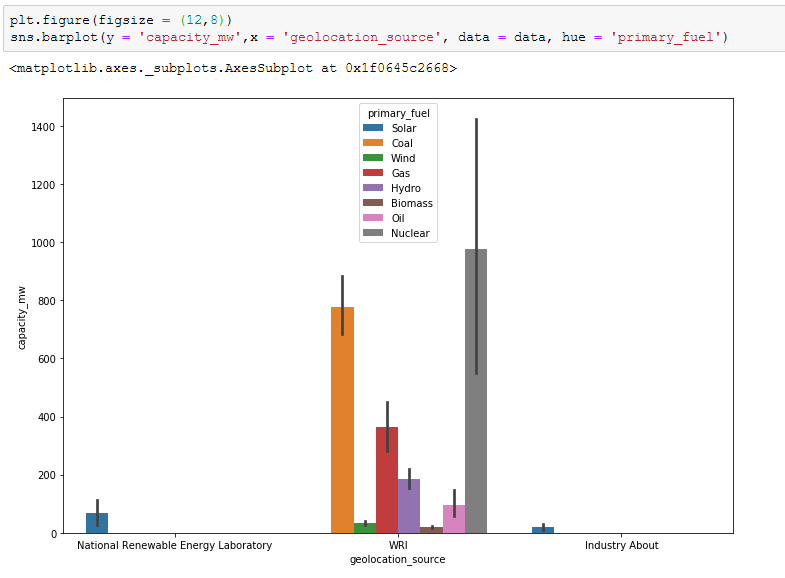
Once the null values are handled I’m proceeding with encoding the target variable **primary\_fuel** to analyse the relationship between some of the dependent and independent variables.



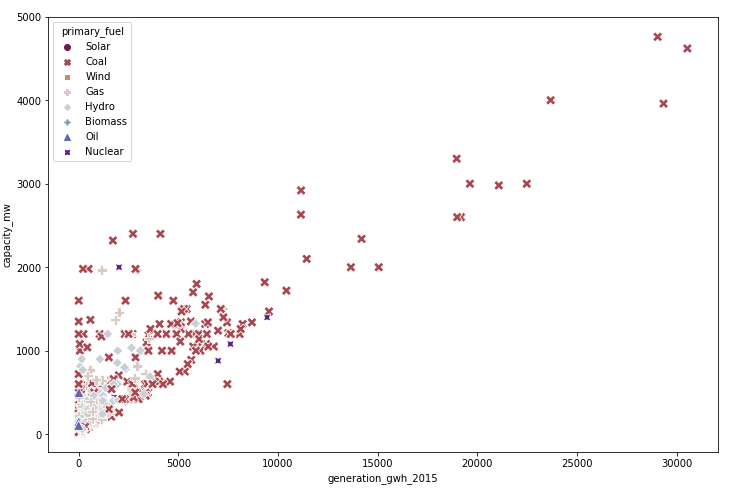
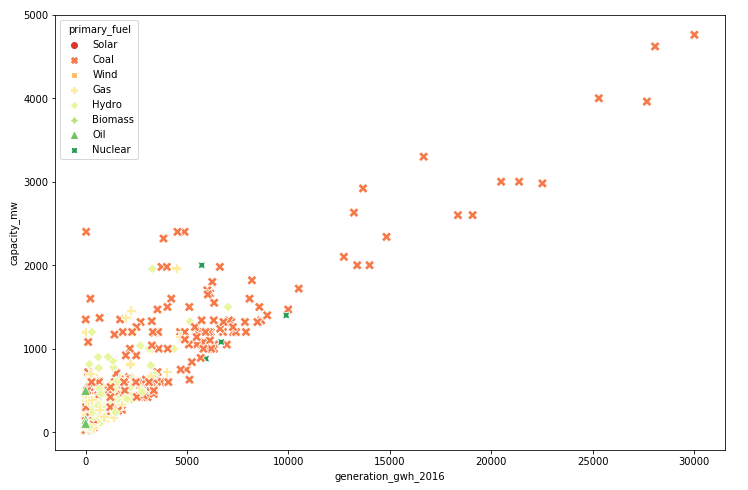
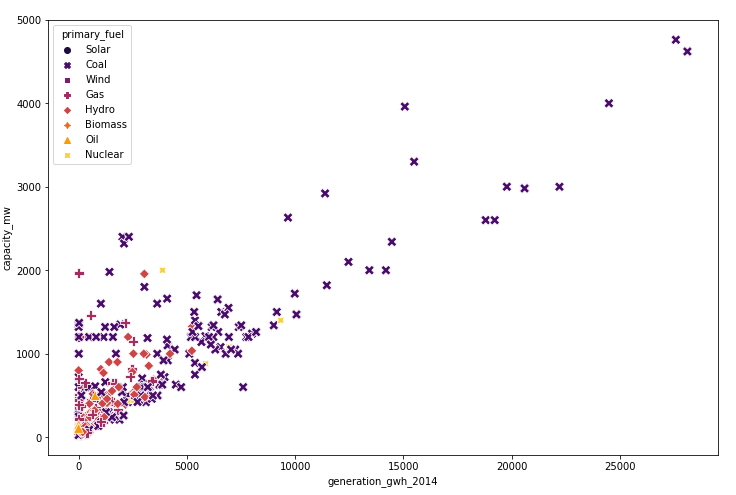
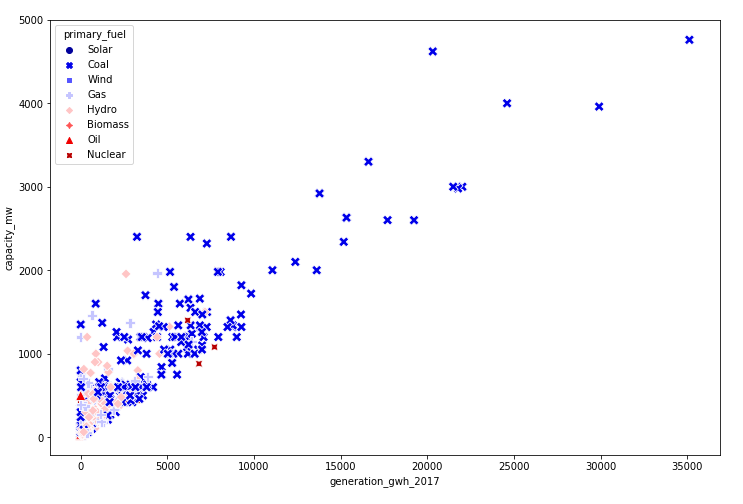
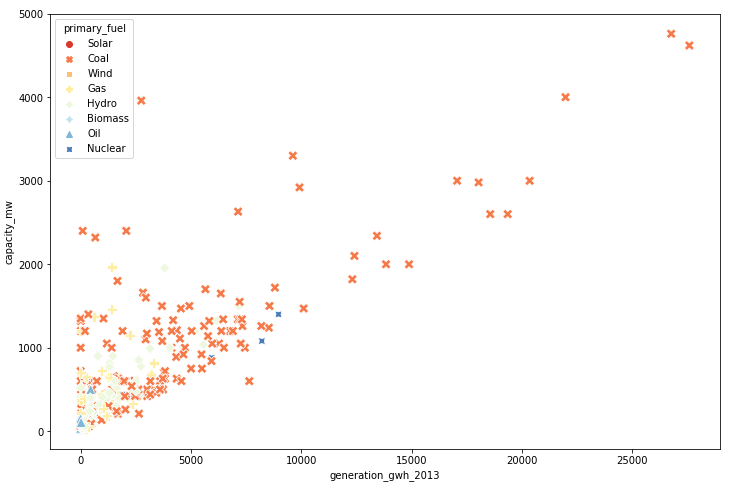
The fuel with label 4 has highest capacity and the fuel with label 0 and 6 being the lowest capacity plants for the fuel



I could see that as there is increase in commissioning year the capacity of the plant increased. This may imply that the power output demand increased with increase in population.

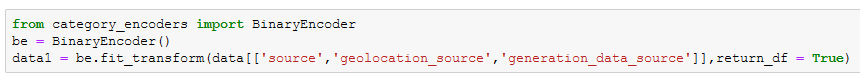


The **geolocation\_source (WRI)** has all type of fuel sources from which the plants generated electricity.

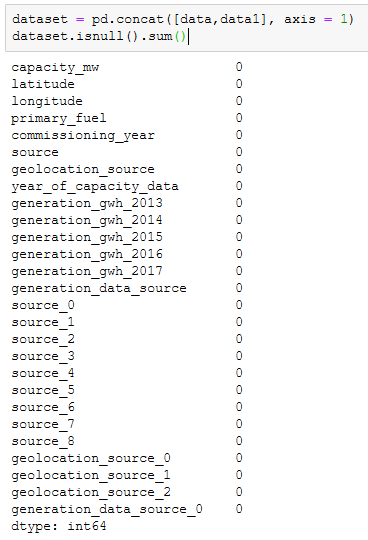


The above scatter plots shows that the electricity (gigawatt-hours) has a positive linear relationship with the capacity of a power plant every year from 2013 to 2017. I can also say that every year power plants using coal as the primary fuel source, generated higher electricity when compared to other primary fuels.

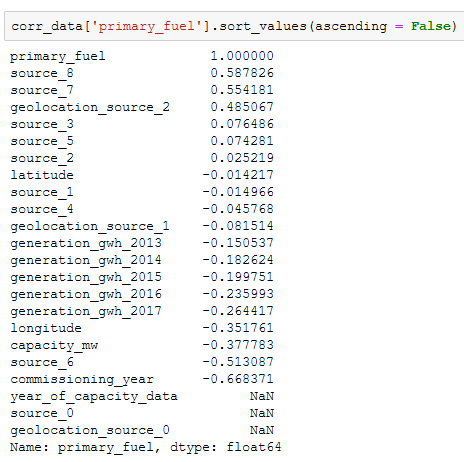
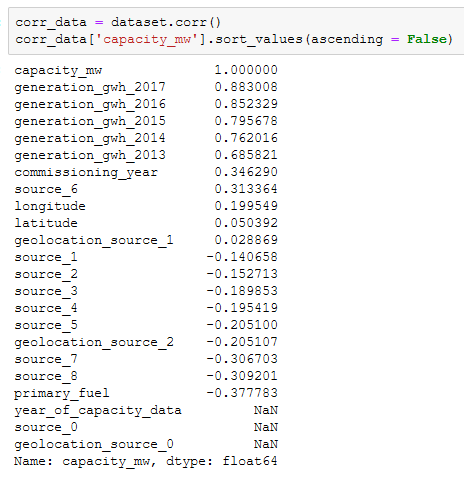
Before I can proceed with finding the actual correlation numbers, I’m encoding the categorical variables using Binary Encoder.



Once the encoding is done I’m using concat operation to join the encoded data with the actual dataset and I’m checking for null values again



I’m removing the variables **'source', 'geolocation\_source', 'generation\_data\_source'** and **'generation\_data\_source\_0'** which were already encoded and joined previously.

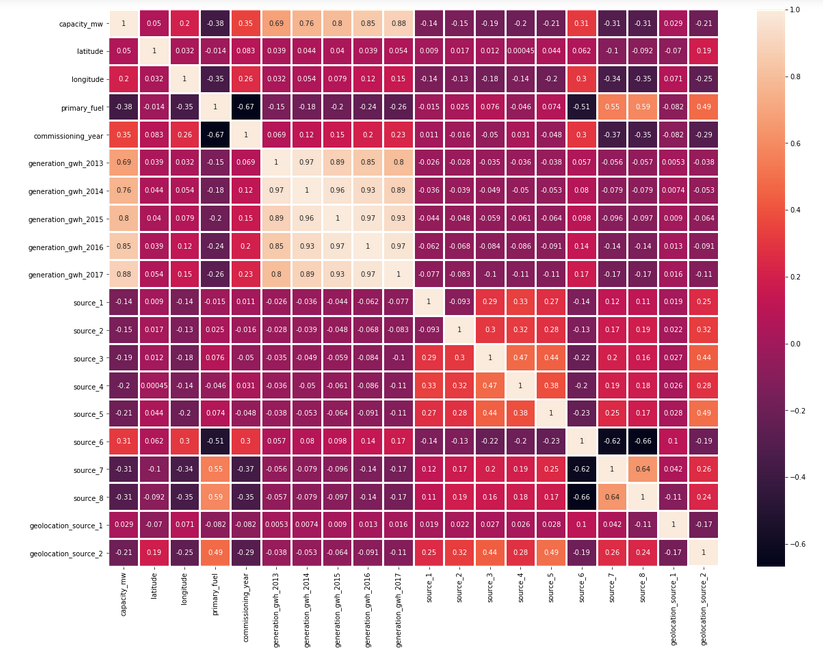
Below are the actual correlation of the independent variables with the dependent variable (there are two **capacity\_mw** and **primary\_fuel**)

I see that the variables **generation\_gwh (2013 to 2017)** are highly correlated with the **capacity\_mw** and the variables **year\_of\_capacity\_data, source\_0** and **geolocation\_source\_0** have no correlation with the target variable **capacity\_mw.** This is because when we encoded data with Binary Encoder some of the columns turned to 0 in all rows (this is an expected behaviour). Likewise the **source (8 and 7)** and **geolocation\_source\_2** is highly correlated with the **primary\_fuel** and the same columns **year\_of\_capacity\_data, source\_0** and **geolocation\_source\_0** has no correlation with the primary fuel.

Before proceeding we can remove this columns from the dataset as they have no correlation with both the target variables.



With the remaining variables I can plot the correlation table in a heat-map for better analysis of correlation and multi-collinearity.

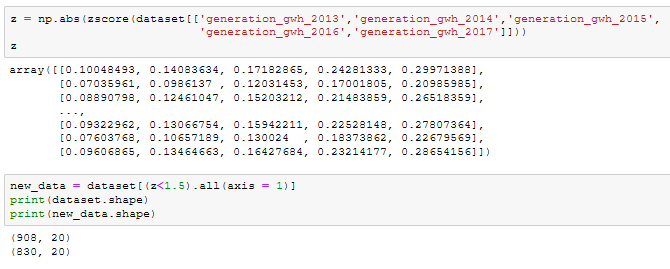


I can see that some of the independent variables are correlated with each other, since it doesn’t affect the prediction of both the variables, I’m not removing any multi-collinearity from the data.

Let’s proceed with further analysis of data…

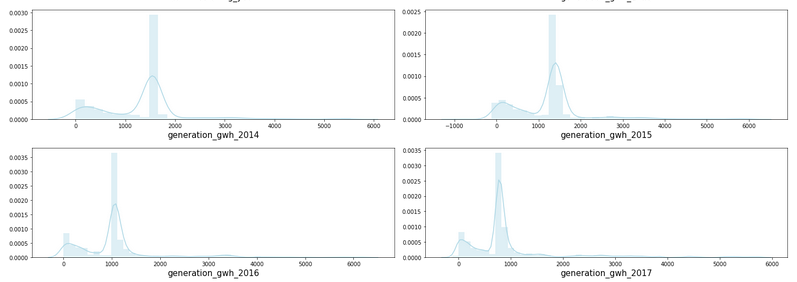
In this step I’m checking for outliers using boxplot. In order achieve that I’ll be writing a small ‘for’ loop which will only display continuous variables. (Note: Outliers cannot be detected for categorical variables)



I can see lot of outliers in the data and in order to treat them I’ll be using statistics. I will only be using the data values which has a z score of less than 1.5.

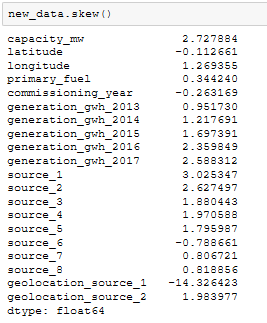
There is an approximate 9% data loss, considering the outliers present, I’m proceeding with the outlier removal.

Now, I’ll be using the outlier treated dataset (**new\_data**) to check the data distribution for any skewness in the continuous data variable. (Note: It’s impossible to detect skewness for categorical variables)

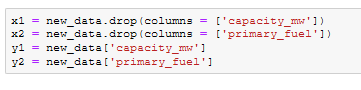


There is skewness present in the data. In order to treat the skewness I’ll be first separating the dataset to dependent and independent variables for both predictions (**capacity\_mw** and **primary\_fuel**) and treat the skewness using power transformation technique. Which proves to be effective.

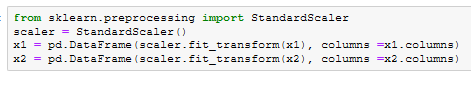
I want to make sure the skewness coefficient of the continuous data variables should be in the range of -0.5 to +0.5. However, I can see that most of it are out of range.



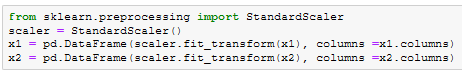
As discussed before, we are splitting dataset into x1 and y1 for capacity prediction model, similarly x2 and y2 for primary fuel prediction model and apply any further pre-processing techniques separately for independent variables related to both the predictions.

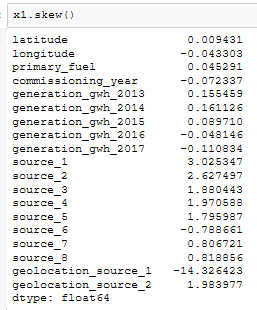
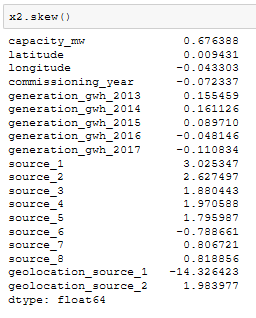


I’m scaling the data using Standard Scaler in order to keep the data values within the range or -3 to +3 and just like we split the data for 2 models. I’m scaling the independent variables x1 and x2 separately.



Applying power transformation on scaled x1 and x2 and checking for skewness in the dataset after transformation.



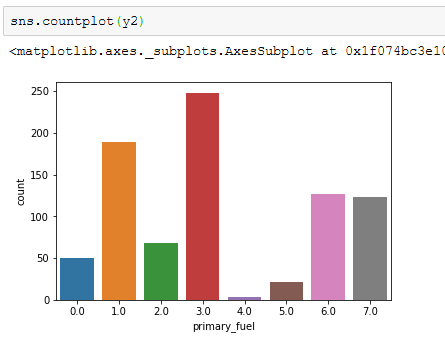


From both of the above analysis, I can see that after applying the power transformation technique, the skewness is under control for almost all the continuous variables.

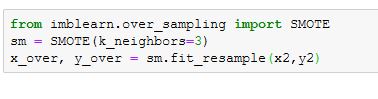
Now we have performed all the necessary data cleaning steps, we can proceed with Model Building and we are predicting 2 variables here (**primary\_fuel** and **capacity\_mw**)

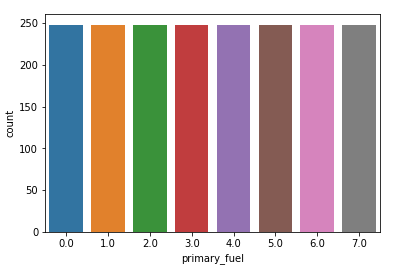
1. Predicting **primary\_fuel** (x2 and y2)

This is a multi-class prediction and I’m checking and treating class imbalance issue before proceeding with model building.

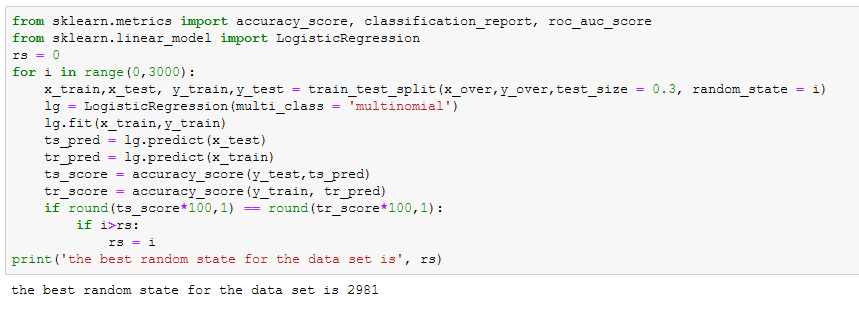


I can see that the classes are imbalanced and I’m treating the same with SMOTE over-sampling technique.



The class has been balanced. We can verify the same from the below count plot.

Before, I can split the data and build Machine Learning models, I’m writing a small for loop to control the overfitting to certain extent. This for loop will find best random state for split where the accuracy scores of the training data set will be equal to the testing dataset for that random state (split).

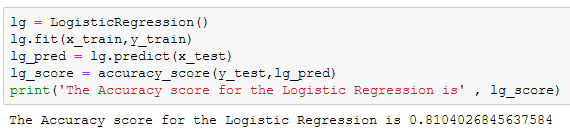


Now I’m splitting the dataset with the best random state in 70:30 ratio

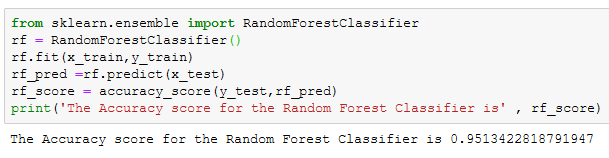


**Machine Learning Models**

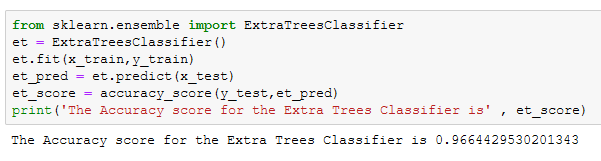
Model 1: Logistic Regression



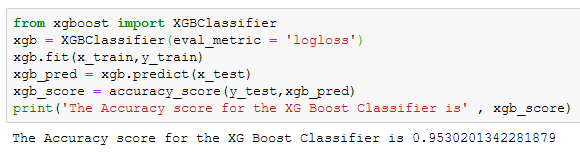
Model 2: Random Forest Classifier



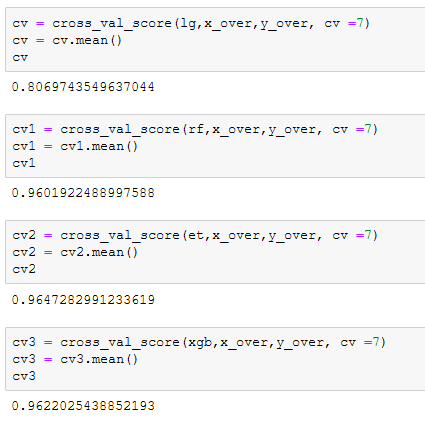
Model 3: Extra Trees Classifier



Model 4: XG Boost Classifier



To verify the model’s fit, I’m using ‘cross\_val\_score’ on all the models with 7 fold cross validation.



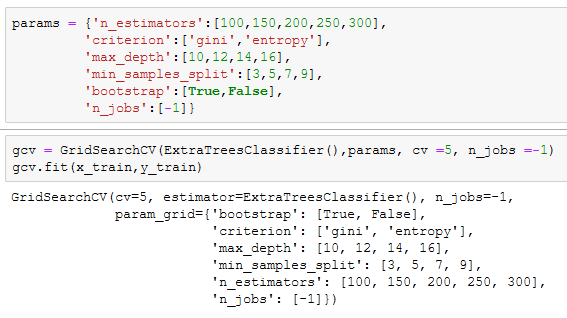
Let’s compare them both to finalize our best model.

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Model Accuracy | Cross Validated Accuracy | Difference |
| Logistic Regression | 0.810403 | 0.806974 | 0.003428 |
| Random Forest Classifier | 0.951342 | 0.960192 | -0.00885 |
| Extra Trees Classifier | 0.966443 | 0.964728 | 0.001715 |
| XG Boost Classifier | 0.95302 | 0.962203 | -0.009182 |

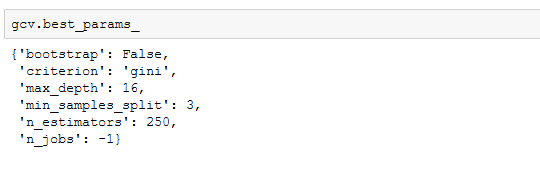
From the above score comparisons between the Model’s Accuracy and Cross Validated Accuracy, I’m considering Extra Trees Classifier as the best model for the primary fuel prediction.

Therefore performing Hyper Parameter Tuning on the same.

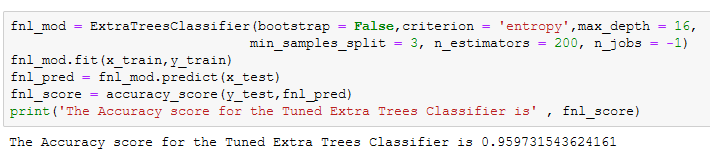
Hyper Parameter Tuning



Best Parameters are…

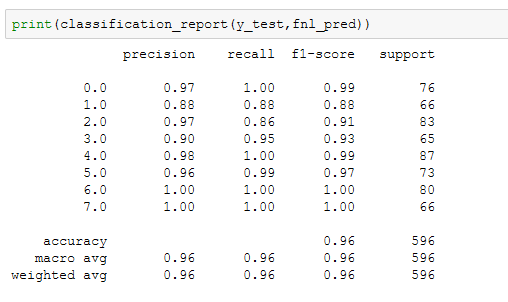


Training the dataset with the best parameters



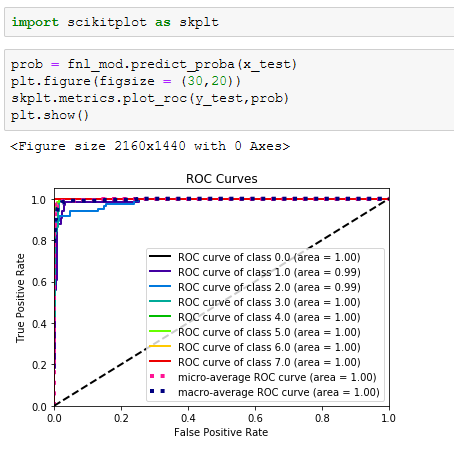
The accuracy score is 0.96 and it’s the best fit for this prediction. Further we can use some evaluation metric to validate the model

Classification Report



We can see that the Average balanced F-1 score, precision and recall is 0.96.

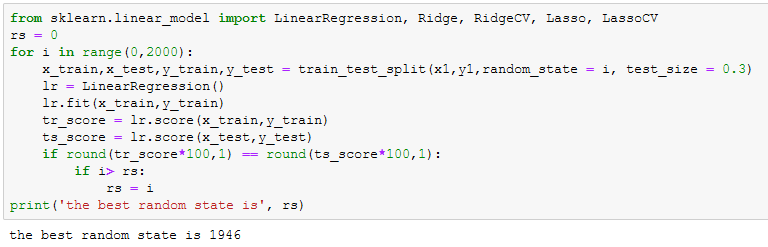
Plotting the AUC ROC Curve for the analysis.



We are able to predict the Fuel type with 96% accuracy and the Extra Trees Model is able to distinguish between classes very accurately.

1. Predicting **capacity\_mw**

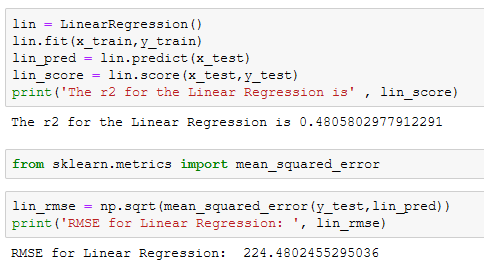
Just like previous prediction, we are first finding the best random state to split the data to train and test.



Using the best random state, I’m splitting the dataset in 70:30 ratio.

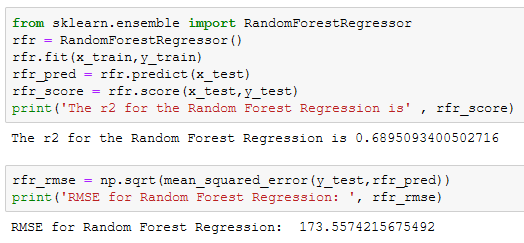


Model 1: Linear Regression



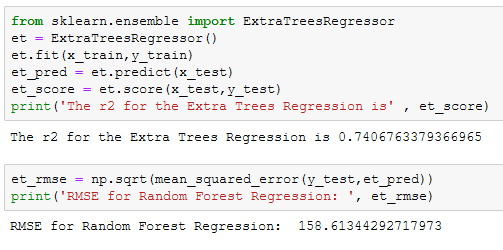
The Linear Regression model is not a good fit for the dataset in predicting the power plant’s capacity as the r2 value is 0.48 and the RMSE (Root Mean Squared Error) is 224.48. Therefore, I’m proceeding to use other models.

Model 2: Random Forest Regressor

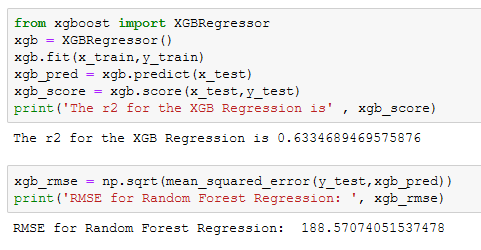


The score is improved a bit with the Random Forest Regressor and so is the RMSE.

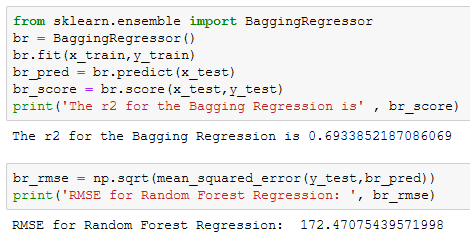
Model 3: Extra Trees Regressor



Model 4: XG Boost Regressor



Model 5: Bagging Regressor



Now that I have fit the dataset to the models, I am verifying the scores with the cross validation score to verify the model’s fit to check whether they are over fitting or not.

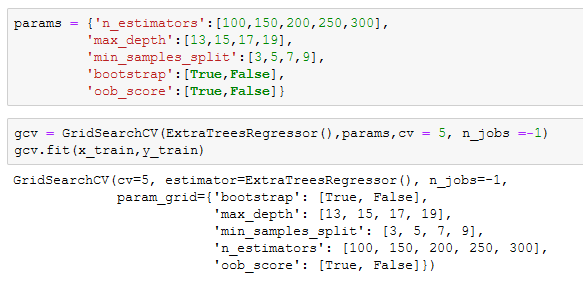


Now I’m selecting the best model by comparing the accuracy score and the cross validation scores along with the RMSE, to get the lowest difference and RMSE

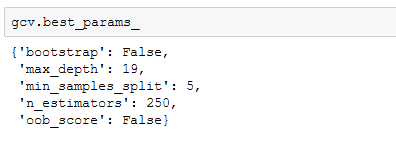
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **R2** | **Cross Validated R2** | **Difference** | **RMSE** |
| Linear Regression | 0.48058 | 0.373005 | 0.107575 | 224.48 |
| Random Forest Regressor | 0.689509 | 0.702554 | -0.013045 | 173.56 |
| Extra Trees Regressor | 0.740676 | 0.733396 | 0.00728 | 158.61 |
| XG Boost Regressor | 0.633469 | 0.671525 | -0.038056 | 188.57 |
| Bagging Regressor | 0.693385 | 0.651699 | 0.041686 | 172.47 |

Based on the above comparison, I can consider that the Extra Trees Regressor is the best model, because it is giving me better Accuracy and lower RMSE

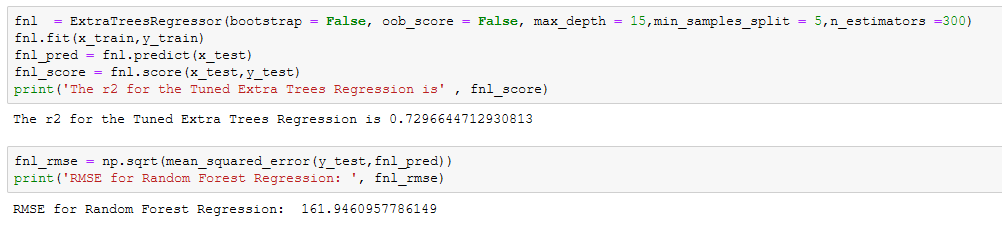
Hyper Parameter Tuning for the Extra Trees Regressor



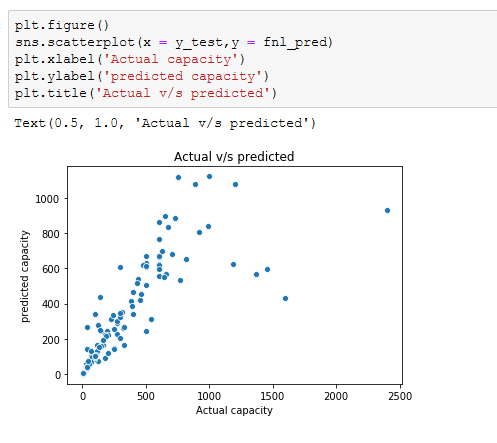
Best Parameter for post tuned model



Let’s fit the best parameters to the Extra Trees Regressor



This is the best fit model in predicting the capacity of a power plant with the given dataset. Let’s visualize the actual and predicted value using a scatterplot



Although we are able to get r2 score of 0.73 at the maximum, we were able to predict the primary fuel attribute with the accuracy of 0.96. This model can still be improved with more data and more number of parameters in the hyper parameter tuning (especially for the regression problem - power plant capacity).

Further, for both the analysis Extra Trees was the best model which provided better results with lesser error.