Power Generation - Climate Connect

DATA

1) According to the problem statement states, there is a power plant whose solar generation from October 1st, 2017 to September 30th, 2019 is given.

2) 3 files are given which contain the data:

File1: power_actual

1. This file contains the solar generation of a certain plant from October 1st, 2017 to September 30th, 2019.

2. You'll find the following columns: 'power', 'gti' and 'ghi'. Power is the actual power generated while GHI (Global Horizontal Irradiance) and GTI (Global Tilt Irradiance) are the parameters relevant to the that define the radiation received from the sun.

File2: weather_actuals

1. This file contains the weather data of the same plant from October 1st, 2017 to September 30th, 2019.

2. The columns' names are self-explanatory.

File3: weather forecast

1. This file contains the weather data from October 1st, 2019 to October 27th, 2019.

We need to predict the generation of power of the given plan in this duration: October 1st, 2019 to October 27th, 2019.

Analysis process

1) Imported and read the file 'power actual' in jupyter notebook using pd.read_csv('file path')

2) Did the usual checks for the data.

3) Its shape was 70080 x 4 .

4) It contains the column 'power' which must be predicted for the period

from October 1st, 2019 to October 27th, 2019.

5) As the column 'ghi' and 'gti' were having only 1 unique value and these had 50% null values, so dropped these columns as they won't help in analysis.

6) Imported and read the file 'weather actual' in jupyter notebook using pd.read_csv('file path')

7) Did the usual checks for the data.

8) Its shape was 13619 x 31.

9) As both the files ‘power actual’ and ‘weather actual’ combinedly formed our training data as the target column’s ‘power’ value was given for this period, so merged both the files into 1 data frame i.e. ‘train_df’ using df.merge
train_df = pd.merge(power , weat_act, how='inner', left_on=['datetime'], right_on = ['datetime_local'])

10) Now we have our training data to build our model.

11) Its shape was 13619 x 34.

12) Did a sanity check on our training data and calculated null values of the columns.

13) Dropped all those columns which had a very high percentage of null values, namely
'precip_type','precip_accumulation','heat_index','qpf','snow','pop','fctcode','wind_chill'

14) Now we had our data frame in full structured form having no null value in any column.

15) Dropped all the column which were irrelevant to our model, namely
'Unnamed:0_x','Unnamed:0_y','datetime','datetime_utc','updated_at','summary'.

16) Dropped all such variables which had only 1 unique value, as they won’t add any significance to our model.

17) Now our training data is in good structure free from any column containing any null value and any irrelevant column.

18) Did Feature engineering on some columns to help in our analysis.

19) As the column 'apperent_temperature' is more relevant than 'temperature' as it the real temperature felt in the day, so dropped the column ‘temperature’.

20) Converted all the datetime columns such as 'datetime_local' ,'sunrise','sunset' in required datatype of timestamp to help in analysis.

21) As the hour is the most important feature of the timestamp, so created new columns namely datetime_hour, sunrise_hour, sunset_hour by extracting the hour from the timestamp from columns datetime, sunrise and sunset to help in analysis.

22) Made column to show duration of how much the plant was exposed to the sunlight at that time:
train_df['exposure_duration'] = train_df['datetime_hour'] -train_df['sunrise_hour']

23) Made column to show how much the sun was there in the sky i.e. how much was the daylight: train_df['daylight_duration']= train_df['sunset_hour'] - train_df['sunrise_hour']

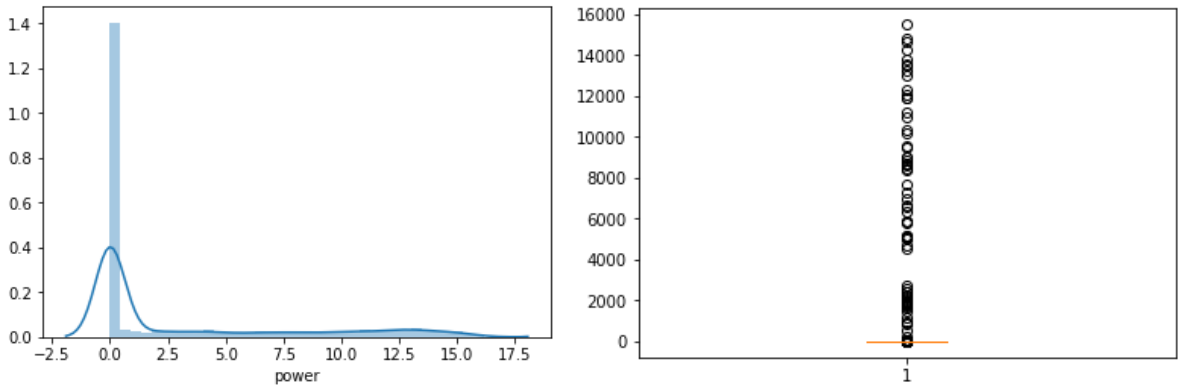
24) Now dropped the date time columns because we extracted the required information out of these columns, i.e the hour factor

.

25) Outlier treatment for the target variable 'power’.

26) As it had normal increment in values from 0-16 till 99th percentile but increased abruptly and had the maximum value of 15504.750.

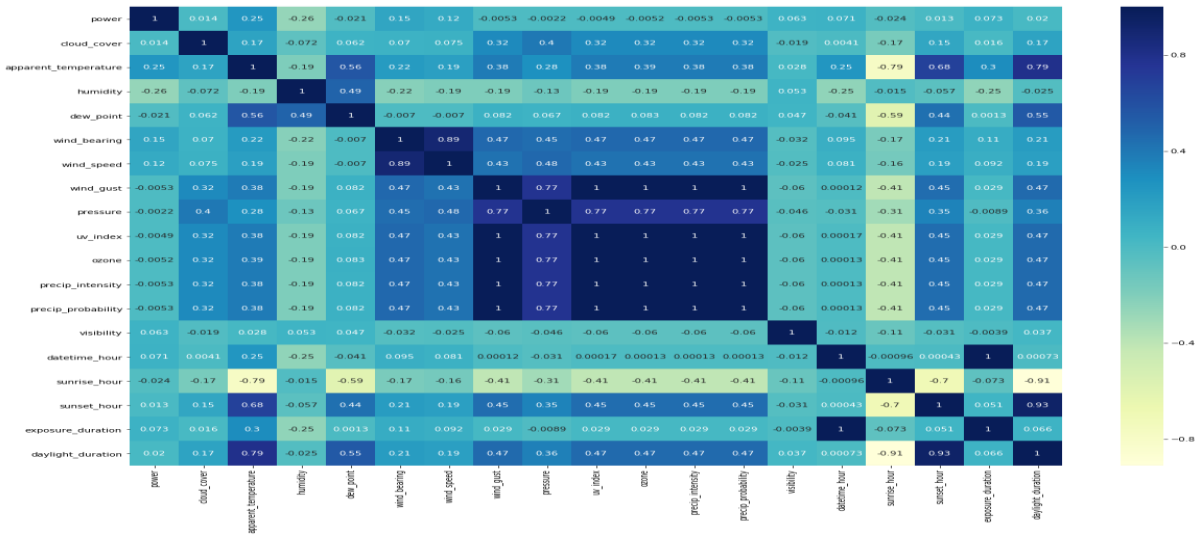
27) Plotted a box plot and and distplot to show the same.



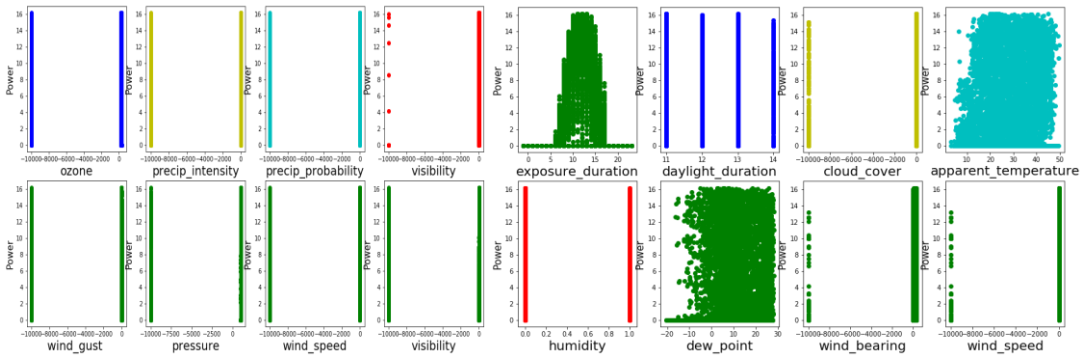
28) So filtered training data with values of column ‘power’ in 0-99th percentile, removed values above than 99th percentile.

29) Calculated correlation of the columns of training data and plotted a heatmap for it.

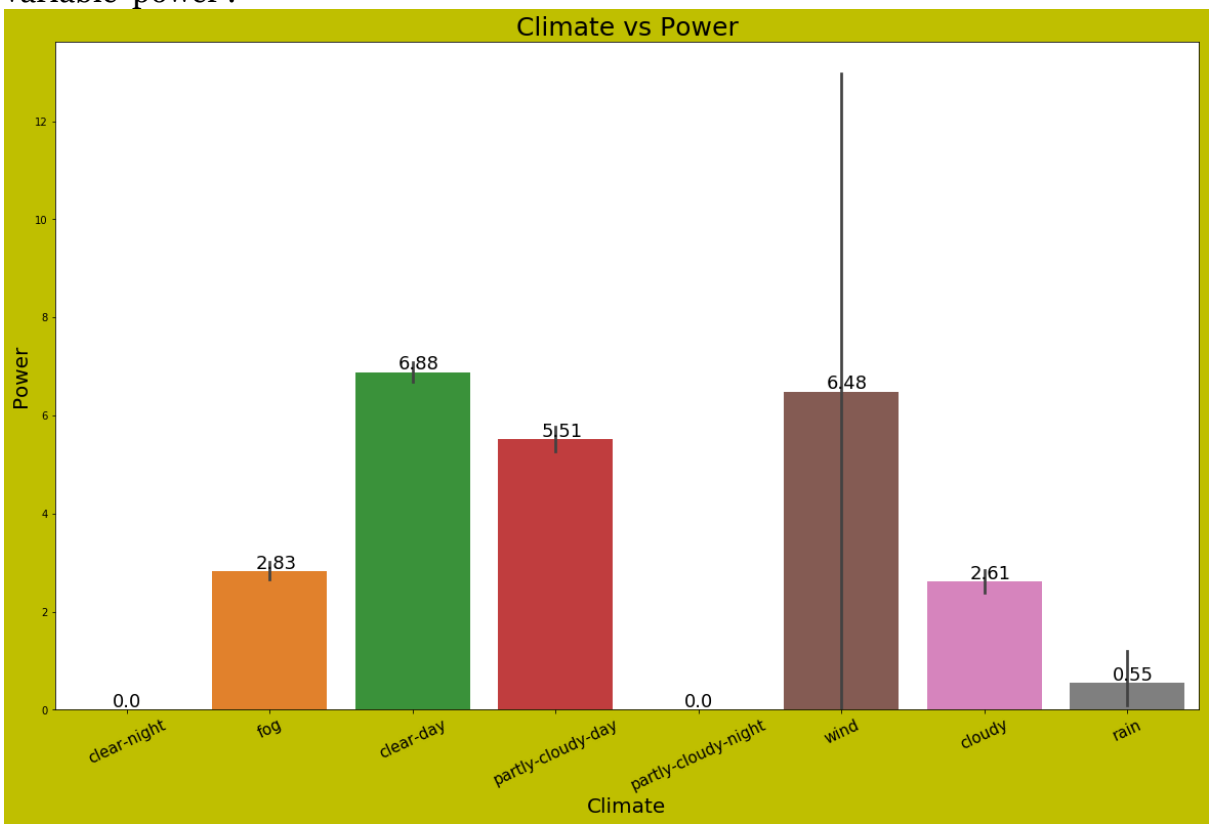
30) This showed. that our target variable ‘power’ somewhat strongly positivel correlates with ‘apparent_temperature’.



31) Plotted scatter plots of the independent features with dependent features to get some intuition about the data.



32) Plotted a barplot of independent categorical column ‘icon’ with dependent variable ‘power’.



33) Did one hot encoding of the categorical column to convert the values into numeric datatype.

34) Scaled the numerical columns to get their values in same magnitude.

35) Divided the data into X and Y variable, X containing all the independent features and Y containing the target variables.

36) Then splitted the data obtained into training and validation data to check the performance on validation data.

37) Used **Random Forest** algorithm to build the model.

38) With hyperparameters of n_estimators=100 and random_state = 0.

39) Fitted the random forest regressor function on x_train and y_train.

39) the variance captured in train data, R^2: 0.9769807105361356

the variance captured in validation data,R^2: 0.8437971879038821

40.A) Mean Absolute Error train: 0.06660021528576558

Mean Squared Error train: 0.02307360531732711

Root Mean Squared Error train: 0.1518999845863294

40.B) Mean Absolute Error valid: 0.17458857083731136

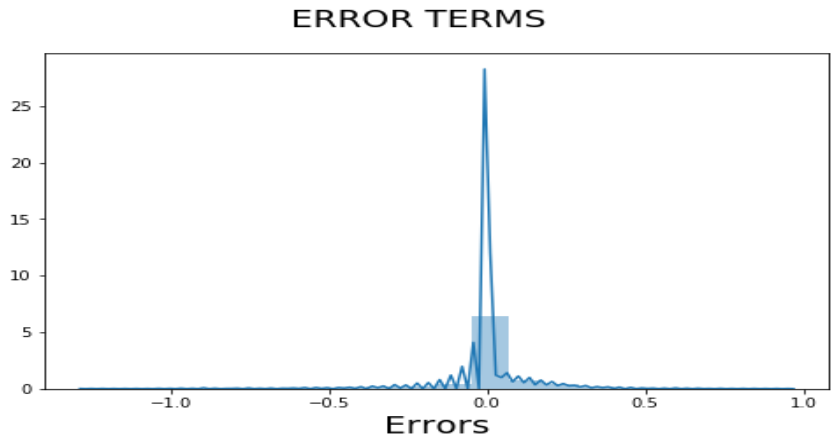
Mean Squared Error valid: 0.15533619624849496

Root Mean Squared Error valid: 0.3941271320887398

41) Top 5 most significant features are:

0	4.344371e-01	uv_index
1	2.106441e-01	datetime_hour
2	7.669690e-02	exposure_duration
3	4.576836e-02	dew_point
4	3.522482e-02	apparent_temperature

42) Plotted the error in predictions on training data, it followed a normal distribution and symmetric around mean i.e. 0



43) Read the file 'weather forecast' and is our test data because we have to predict the generation of power in the interval given.

44) Carried out all the similar steps to preprocess the test data as those which were used in processing training data, so that the model encounters same features in test data as it encountered in training data so that it performs well.

45) Scaled the numerical features, did one hot encoding of categorical column.

46) Did a check on those features that were present in training data but absent in test data.

47) Created such features and imputed the values to be 0 so that they don't create any disturbance in the predictions.

48) Did Y_pred =regressor .predict(test_new), to predict the target variable for test data.

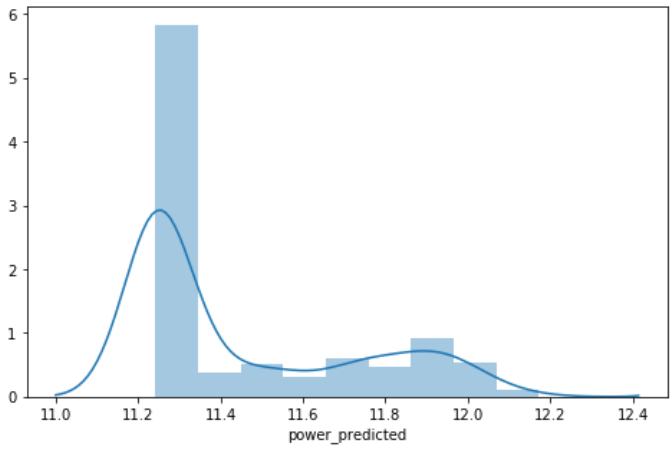
49) Created a new column 'power_predicted' in the test data using test_new['power_predicted'] = Y_pred.

50) But it was the scaled value and we want values on original scale to show our predictions.

51) So inverse transformed the scaled feature to get the values in original scale.

52) Did data.describe() to see the variations in the value.

53) It came to be in accordance to the values of training data.



The Plot for distribution of predicted feature