Electrical Power Output - EDA

September 14, 2020

1 Introduction

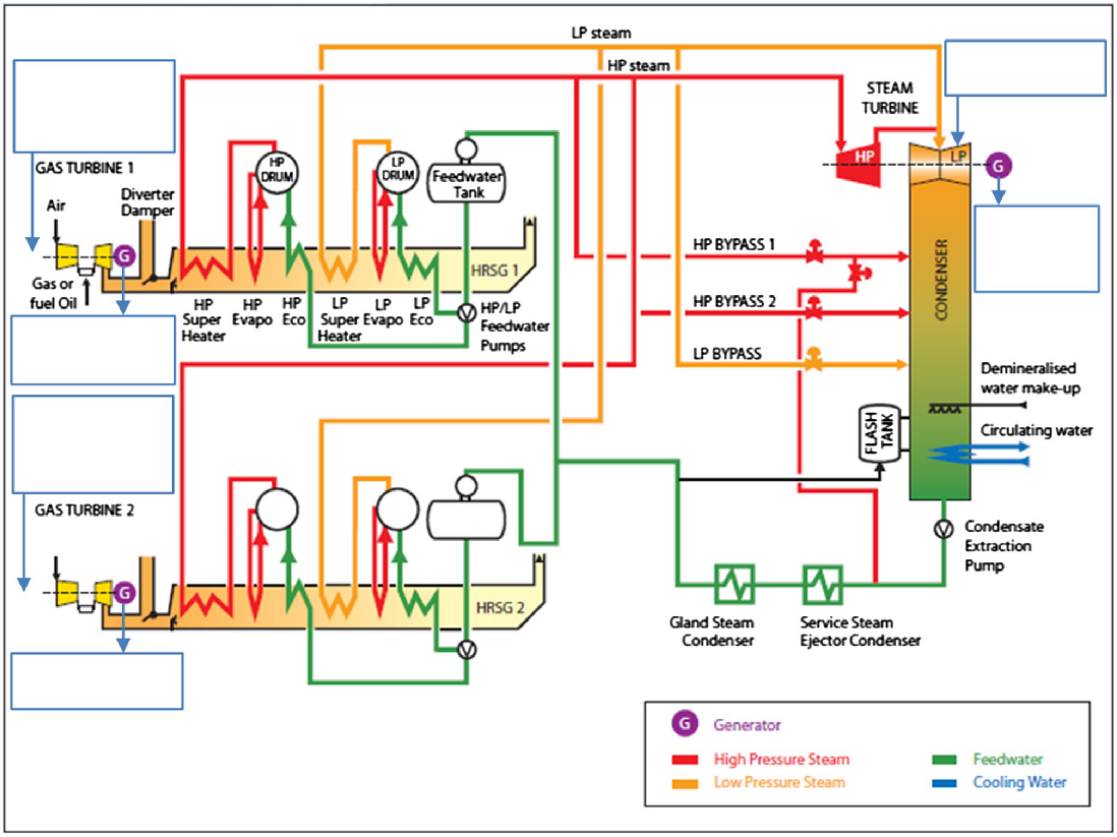
1.1 Description of Combined Cycle Power Plant

A combined cycle power plant is composed of gas turbines (GT), steam turbines (ST) and heat recovery steam generators (HRSG). In a CCPP, the electricity is generated by gas and steam turbines, which are combined in one cycle, and is transferred from one turbine to another. A gas turbine in a combined cycle system does not only generate the electrical power but also generates fairly hot exhaust. Routing these gases through a water-cooled heat exchanger produces steam, which can be turned into electric power with a coupled steam turbine and generator. Hence, a gas turbine generator generates electricity and waste heat of the exhaust gases is used to produce steam to generate additional electricity via a steam turbine. This type of power plant is being installed in increasing numbers around the world where there is access to substantial quantities of natural gas.

The CCPP, from which the dataset for this study was collected, is designed with a nominal generating capacity of 480 MW, made up of 2 X 160MW ABB 13E2 Gas Turbines, 2 X dual pressure Heat Recovery Steam Generators (HRSG) and 1 X 160MW ABB Steam Turbine as illustrated in the figure given below.

[1]:
```
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
plt.figure(figsize = (10,10))
fig, ax = plt.subplots(figsize = (20,15))
ax.imshow(mpimg.imread('ccppp.jpg'))
ax.spines['top'].set_visible(False)
ax.spines['left'].set_visible(False)
ax.spines['bottom'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.set_xticks([])
ax.set_yticks([])
plt.show()
```

```
<Figure size 720x720 with 0 Axes>
```



Gas turbine load is sensitive to the ambient conditions; mainly ambient temperature (AT), atmospheric pressure (AP), and relative humidity (RH).However, steam turbine load is sensitive to the exhaust steam pressure (or vacuum, V). These parameters of both gas and steam turbines, which are related with ambient conditions and exhaust steam pressure, are used as input variables in the dataset of this study. The electrical power generating by both gas and steam turbines is used as a target variable in the dataset. All the input variables and target variable, which are defined as below, correspond to average hourly data received from the measurement points by the sensors also denoted in Figure above.

1. Ambient Temperature (AT): This input variable is measured in whole degrees in Celsius as it varies between 1.81 degree C and 37.11 degree C.
2. Atmospheric Pressure (AP): This input variable is measured in units of minibars with the range of 992.89–1033.30 mbar.
3. Relative Humidity (RH): This variable is measured as a percentage from 25.56% to 100.16%.
4. Vacuum (Exhaust Steam Pressure, V): This variable is measured in cm Hg with the range of 25.36–81.56 cm Hg.
5. Full Load Electrical Power Output (PE): PE is used as a target variable in the dataset. It is measured in mega watt with the range of 420.26–495.76 MW.

1.2 Dataset Description

The dataset contains **9568 data points collected from a Combined Cycle Power Plant over 6 years (2006-2011)**, when the power plant was set to work with full load. Features consist of hourly average ambient variables **Temperature (T), Ambient Pressure (AP), Relative Humidity (RH) and Exhaust Vacuum (V) to predict the net hourly electrical energy output (EP) of the plant**.

A combined cycle power plant (CCPP) is composed of gas turbines (GT), steam turbines (ST) and heat recovery steam generators. In a CCPP, the electricity is generated by gas and steam turbines, which are combined in one cycle, and is transferred from one turbine to another. While the Vacuum is colected from and has effect on the Steam Turbine, the other three of the ambient variables effect the GT performance.

2 Exploring the Dataset : Exploratory Data Analysis

EDA is one of the crucial step in data science that allows us to achieve certain insights and statistical measures that are quite essential. EDA is generally cross-classified in two ways.

1. First, each method is either non-graphical or graphical.
2. And second, each method is either univariate or multivariate (usually just bivariate).

Non-graphical methods generally involve calculation of summary statistics, while graphical methods obviously summarize the data in a diagrammatic or pictorial way.

Let us look at each types one by one.

2.1 Descriptive Statistics (Non-Graphical)

In this section we will look at the **Measures of Central Tendency** (Mean, Median, Mode) and **Measures of Dispersion** (Standard Deviation, Range and Quartiles).

[2]:
```
# importing necessary libraries
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
plt.style.use('ggplot')
import math
```

- We will first import the dataset will be taking a look at the type of the variables we have in the dataset.

[3]:
```
df = pd.read_excel('Folds5x2_pp.xlsx') # saving the data in the dataframe
print("First Five Observations are: \n")
df.info()
```

```
First Five Observations are:

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9568 entries, 0 to 9567
```

```
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   AT      9568 non-null   float64
 1   V       9568 non-null   float64
 2   AP      9568 non-null   float64
 3   RH      9568 non-null   float64
 4   PE      9568 non-null   float64
dtypes: float64(5)
memory usage: 373.9 KB
```

It can be observed that the dataset has all the variables in floating point integer

- Now we will look the first five obseravations in the data set to get an idea of magnitudes of each variable

[4]:
```
df.head()
```

[4]:

	AT	V	AP	RH	PE
0	14.96	41.76	1024.07	73.17	463.26
1	25.18	62.96	1020.04	59.08	444.37
2	5.11	39.40	1012.16	92.14	488.56
3	20.86	57.32	1010.24	76.64	446.48
4	10.82	37.50	1009.23	96.62	473.90

- Now, let us dive into some **Descriptive Statistis**

[5]:
```
print("\nThe  summary of descriptive statistics of the data is: \n")
df.describe()
```

```
The  summary of descriptive statistics of the data is:
```

[5]:

	AT	V	AP	RH	PE
count	9568.000000	9568.000000	9568.000000	9568.000000	9568.000000
mean	19.651231	54.305804	1013.259078	73.308978	454.365009
std	7.452473	12.707893	5.938784	14.600269	17.066995
min	1.810000	25.360000	992.890000	25.560000	420.260000
25%	13.510000	41.740000	1009.100000	63.327500	439.750000
50%	20.345000	52.080000	1012.940000	74.975000	451.550000
75%	25.720000	66.540000	1017.260000	84.830000	468.430000
max	37.110000	81.560000	1033.300000	100.160000	495.760000

Though the Descriptive Statistics gives us all the numerical information, they are not enough to get the knowledge of actual distribution of variables. This is where graphs and plots come in role.

2.2 Adding some extra information in the Dataset

Here, we will be dividing each of the variable **Here we will be adding new columns to the dataset which will classify whether the data falls in Low, Medium or High Category.**

This Low, Medium and High Category is calculated by divinding the Range of Data Equally into 3 Categories

The Code for the same is as below:

[6]:
```
lab = ["Low", "Medium", "High"]
amb_Temp_bins = np.linspace(math.floor(min(df['AT'])), math.
 ↪ceil(max(df['AT'])),4)
exht_Vac_bins = np.linspace(math.floor(min(df['V'])), math.ceil(max(df['V'])),4)
amb_pres_bins = np.linspace(math.floor(min(df['AP'])), math.
 ↪ceil(max(df['AP'])),4)
rh_bins = np.linspace(math.floor(min(df['RH'])), math.ceil(max(df['RH'])),4)
pe_bins  = np.linspace(math.floor(min(df['PE'])), math.ceil(max(df['PE'])),4)

df['Amb_Temp'] = pd.cut(df['AT'], bins = amb_Temp_bins, labels = lab )
df['Exht_vac'] = pd.cut(df['V'], bins = exht_Vac_bins, labels = lab )
df['Amb_Pres'] = pd.cut(df['AP'], bins = amb_pres_bins, labels = lab )
df['Rel_Humid'] = pd.cut(df['RH'], bins = rh_bins, labels = lab )
df['Pwr'] = pd.cut(df['PE'], bins = pe_bins, labels = lab )
```

Lets look at the top 5 columns of our mutated Dataset

[7]:
```
df.head()
```

[7]:

	AT	V	AP	RH	PE	Amb_Temp	Exht_vac	Amb_Pres	Rel_Humid	\
0	14.96	41.76	1024.07	73.17	463.26	Medium	Low	High	Medium	
1	25.18	62.96	1020.04	59.08	444.37	Medium	Medium	High	Medium	
2	5.11	39.40	1012.16	92.14	488.56	Low	Low	Medium	High	
3	20.86	57.32	1010.24	76.64	446.48	Medium	Medium	Medium	High	
4	10.82	37.50	1009.23	96.62	473.90	Low	Low	Medium	High	

	Pwr
0	Medium
1	Low
2	High
3	Medium
4	High

Now, we will look at the descriptive Statistics of the Categorical Values in the Dataset that we have just defined

[8]:
```
df.iloc[:,5:10].describe(include='all')
```

[8]:

	Amb_Temp	Exht_vac	Amb_Pres	Rel_Humid	Pwr
count	9568	9568	9568	9568	9568
unique	3	3	3	3	3
top	Medium	Low	Medium	High	Medium
freq	4833	3268	7338	4590	3798

2.3 Visualising the Data (Graphical)

In this section we will be exploring the data through some visualitions such as Histograms, Box and Whisker Plot, Pie chart, Bar chart and scatter plots. Every chart has their own set of advantages and disadvantages, therefore we use multiple chats/graphs to get the full idea about the dataset.

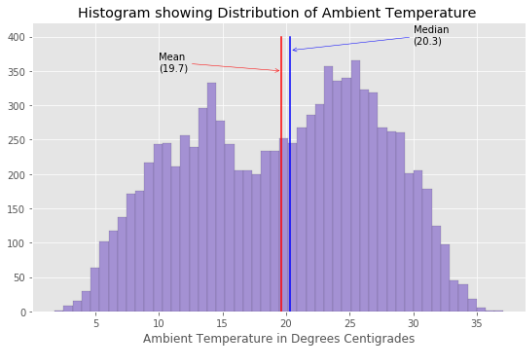
Now we will be visualizing the graphs for each variable one by one.

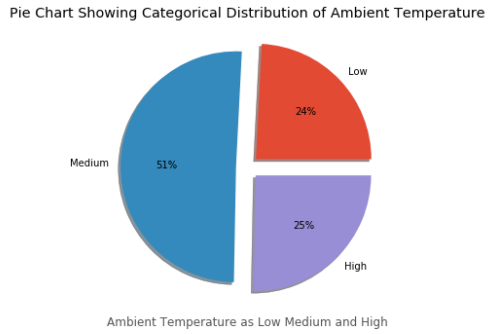
[9]:
```
#Hexacodes for Colors to use
#c=['#904C77','#E49AB0','#ECB8A5','#ECCFC3','#78A1BB']
c = ['#A491D3','#A63D40','#E9B872','#90A959','#818AA3']
```

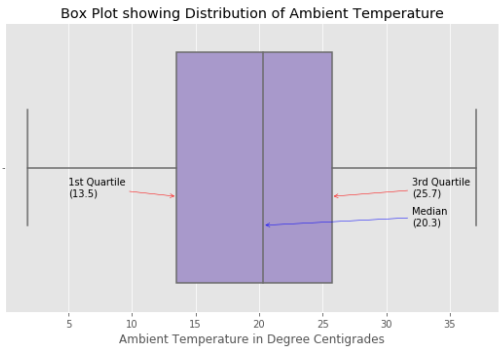
2.3.1 Visualizations of Ambient Temperature

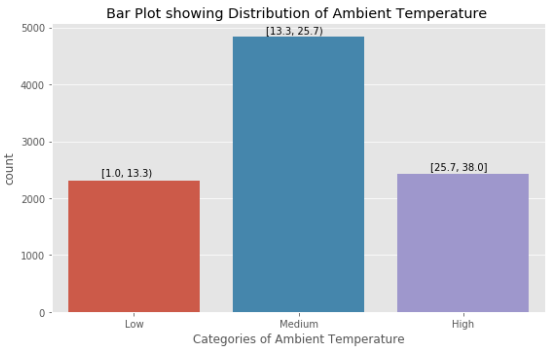
[10]:
```
mean_AT = df['AT'].mean()
median_AT = df['AT'].median()
mode_AT = df['AT'].mode()
qnt_1_AT = df['AT'].quantile(0.25)
qnt_3_AT = df['AT'].quantile(0.75)

fig, ax = plt.subplots(2,2, figsize=(20,12))
ax[0,0].hist(df['AT'], bins = 50, color = c[0], linewidth = 0.1, edgecolor =␣
↪'black')

#Histogram

ax[0,0].plot([mean_AT, mean_AT, mean_AT], [0, 200, 400], color = 'red' )
ax[0,0].plot([median_AT, median_AT], [0, 400] , color = 'blue')
ax[0,0].annotate(f"Mean\n({round(mean_AT,1)})",xy=(mean_AT, 350),␣
↪xytext=(10,350), arrowprops={"arrowstyle":"->", "color":"red"})
ax[0,0].annotate(f"Median\n({round(median_AT,1)})",xy=(median_AT, 380),␣
↪xytext=(30,390), arrowprops={"arrowstyle":"->", "color":"blue"})
ax[0,0].set_title("Histogram showing Distribution of Ambient Temperature ")
ax[0,0].set_xlabel("Ambient Temperature in Degrees Centigrades")

# Pie Chart
t_count = df['Amb_Temp'].value_counts()
ax[0,1].pie([t_count[0], t_count[1],t_count[2]], labels=['Low', 'Medium',␣
↪'High'], shadow= True, explode = [0.1 , 0.1 , 0.1], autopct = '%1.0f%%')
ax[0,1].set_title("Pie Chart Showing Categorical Distribution of Ambient␣
↪Temperature")
```

```
ax[0,1].set_xlabel("Ambient Temperature as Low Medium and High")

# Box Plot
sns.boxplot(df['AT'],ax= ax[1,0], color = c[0] )
ax[1,0].annotate(f"1st Quartile\n({round(qnt_1_AT,1)})",xy=(qnt_1_AT, 0.1),␣
 ↪xytext=(5,0.1), arrowprops={"arrowstyle":"->", "color":"red"})
ax[1,0].annotate(f"3rd Quartile\n({round(qnt_3_AT,1)})",xy=(qnt_3_AT, 0.1),␣
 ↪xytext=(32,0.1), arrowprops={"arrowstyle":"->", "color":"red"})
ax[1,0].annotate(f"Median\n({round(median_AT,1)})",xy=(median_AT, 0.2),␣
 ↪xytext=(32,0.2), arrowprops={"arrowstyle":"->", "color":"blue"})
ax[1,0].set_title("Box Plot showing Distribution of Ambient Temperature")
ax[1,0].set_xlabel("Ambient Temperature in Degree Centigrades")

# Bar Chart
sns.countplot(df['Amb_Temp'], ax = ax[1,1])
ax[1,1].set_title("Bar Plot showing Distribution of Ambient Temperature ")
ax[1,1].set_xlabel("Categories of Ambient Temperature")
ax[1,1].annotate(f"[{round(amb_Temp_bins[0],1)}, {round(amb_Temp_bins[1],1)})",␣
 ↪xy=(-0.2,2400))
ax[1,1].annotate(f"[{round(amb_Temp_bins[1],1)}, {round(amb_Temp_bins[2],1)})",␣
 ↪xy=(0.8,4900))
ax[1,1].annotate(f"[{round(amb_Temp_bins[2],1)}, {round(amb_Temp_bins[3],1)}]",␣
 ↪xy=(1.8,2500))
plt.show()
```







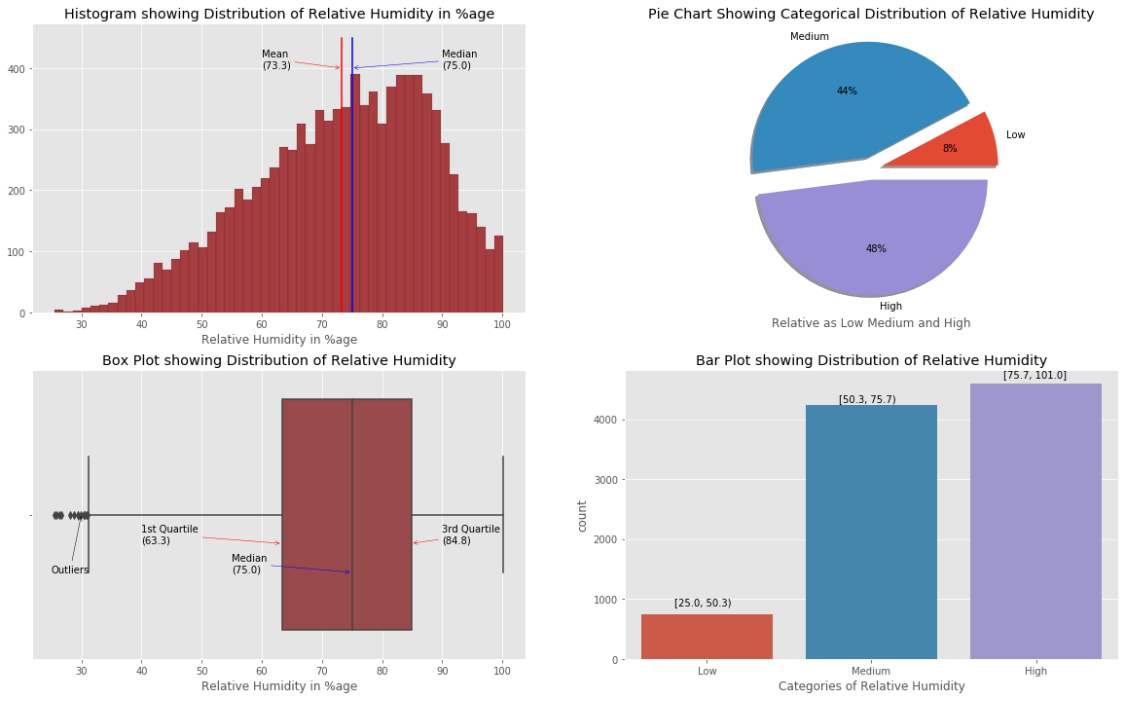


- **It can be observed from the Histogram that the Distribution of the Ambient Temperature is more or less bimodal.**
- **Mean and Median lie very close to each other.**
- **However, the box and wiskers graph fails to capture this bimodal trend. It just depicts that the data is almost eqaully spread of values on the both side of median.**
- **The bar plot and pie chart just show the distribution of values in each category. It shows that most of the values fall in the Medium Category which is described by the interval [13.3, 25.7). However, they fail to capture any other information.**

2.3.2 Visualizations of Relative Humidity

[11]:
```
mean_RH = df['RH'].mean()
median_RH = df['RH'].median()
mode_RH = df['RH'].mode()
qnt_1_RH = df['RH'].quantile(0.25)
qnt_3_RH = df['RH'].quantile(0.75)

fig, ax = plt.subplots(2,2, figsize=(20,12))
ax[0,0].hist(df['RH'], bins = 50, color = c[1], linewidth = 0.1, edgecolor =␣
↪'black')

#Histogram
ax[0,0].plot([mean_RH, mean_RH], [0, 450], color = 'red' )
ax[0,0].plot([median_RH, median_RH], [0, 450] , color = 'blue')
ax[0,0].annotate(f"Mean\n({round(mean_RH,1)})",
                 xy=(mean_RH, 400), xytext=(60,400), arrowprops={"arrowstyle":
↪"->", "color":"red"})
ax[0,0].annotate(f"Median\n({round(median_RH,1)})",
                 xy=(median_RH, 400), xytext=(90,400), arrowprops={"arrowstyle":
↪"->", "color":"blue"})
ax[0,0].set_title("Histogram showing Distribution of Relative Humidity in %age")
ax[0,0].set_xlabel("Relative Humidity in %age")

#Pie Chart
rh_count = df['Rel_Humid'].value_counts()
ax[0,1].pie([rh_count[0], rh_count[1],rh_count[2]],
            labels=['Low', 'Medium', 'High'], shadow= True, explode = [0.1 , 0.
↪1 , 0.1], autopct = '%1.0f%%')
ax[0,1].set_title("Pie Chart Showing Categorical Distribution of Relative␣
↪Humidity")
ax[0,1].set_xlabel("Relative as Low Medium and High")

#Box Plot
sns.boxplot(df['RH'],ax= ax[1,0], color  = c[1]  )
ax[1,0].annotate(f"1st Quartile\n({round(qnt_1_RH,1)})",
                 xy=(qnt_1_RH, 0.1), xytext=(40,0.1), arrowprops={"arrowstyle":
↪"->", "color":"red"})
```

```
ax[1,0].annotate(f"3rd Quartile\n({round(qnt_3_RH,1)})",
                 xy=(qnt_3_RH, 0.1), xytext=(90,0.1), arrowprops={"arrowstyle":
 ↪"->", "color":"red"})
ax[1,0].annotate(f"Median\n({round(median_RH,1)})",
                 xy=(median_RH, 0.2), xytext=(55,0.2), arrowprops={"arrowstyle":
 ↪"->", "color":"blue"})
ax[1,0].annotate("Outliers",xy=(30, 0), xytext=(25,0.2),␣
 ↪arrowprops={"arrowstyle":"->", "color":"black"})
ax[1,0].set_title("Box Plot showing Distribution of Relative Humidity")
ax[1,0].set_xlabel("Relative Humidity in %age")

#Bar Chart
sns.countplot(df['Rel_Humid'], ax = ax[1,1])
ax[1,1].set_title("Bar Plot showing Distribution of Relative Humidity")
ax[1,1].set_xlabel("Categories of Relative Humidity")
ax[1,1].annotate(f"[{round(rh_bins[0],1)}, {round(rh_bins[1],1)})", xy=(-0.
 ↪2,900))
ax[1,1].annotate(f"[{round(rh_bins[1],1)}, {round(rh_bins[2],1)})", xy=(0.
 ↪8,4300))
ax[1,1].annotate(f"[{round(rh_bins[2],1)}, {round(rh_bins[3],1)}]", xy=(1.
 ↪8,4700))
plt.show()
```



- **It can be observed from the above given Histogram and the box plot that the Distribution of Relative Humidity is negatively skewed. This can also be verified**

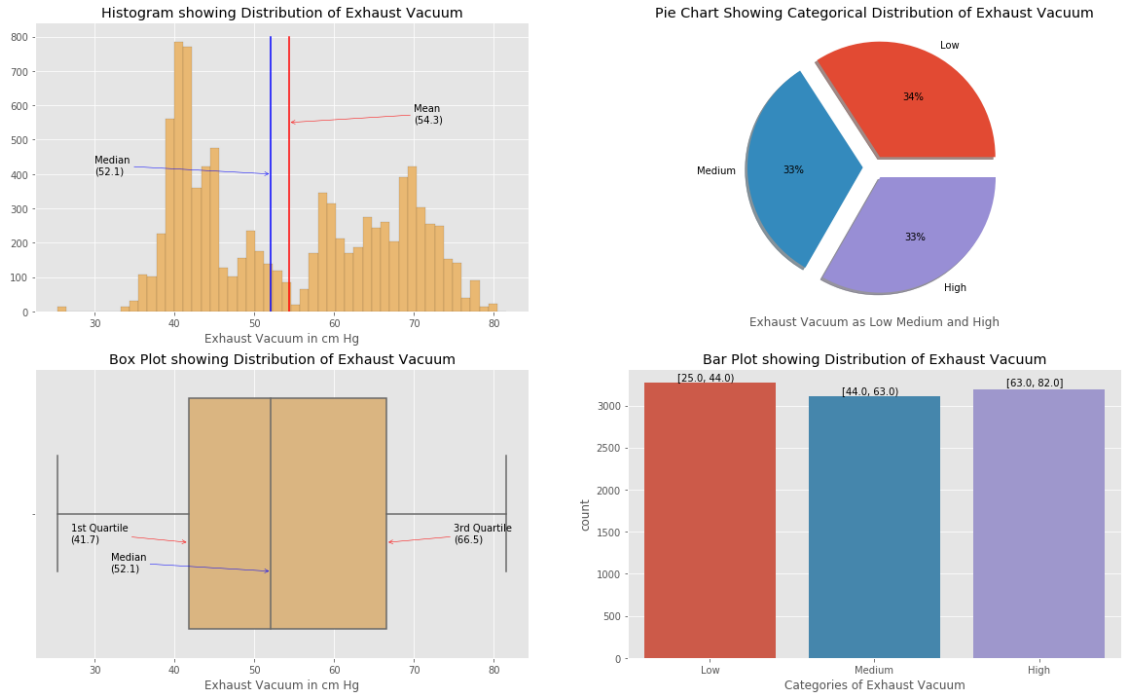
from the fact that the mean is less than the median.

- **The boxplot is also able to capture some of the outliers that fall below the value 31**
- **The bar graph and the pie plot on the left hand side of the above figure shows that the values are almost equally ditributed in Medium and High Category while very less value lies Low Relative Humidity band**

2.3.3 Visualizations of Exhaust Vacuum

[12]:
```
mean_V = df['V'].mean()
median_V = df['V'].median()
mode_V = df['V'].mode()
qnt_1_V = df['V'].quantile(0.25)
qnt_3_V = df['V'].quantile(0.75)

fig, ax = plt.subplots(2,2, figsize=(20,12))
ax[0,0].hist(df['V'], bins = 50, color = c[2], linewidth = 0.1, edgecolor =␣
↪'black')

#Histogram
ax[0,0].plot([mean_V, mean_V], [0, 800], color = 'red' )
ax[0,0].plot([median_V, median_V], [0, 800] , color = 'blue')
ax[0,0].annotate(f"Mean\n({round(mean_V,1)})",xy=(mean_V, 550),␣
↪xytext=(70,550), arrowprops={"arrowstyle":"->", "color":"red"})
ax[0,0].annotate(f"Median\n({round(median_V,1)})",xy=(median_V, 400),␣
↪xytext=(30,400), arrowprops={"arrowstyle":"->", "color":"blue"})
ax[0,0].set_title("Histogram showing Distribution of Exhaust Vacuum")
ax[0,0].set_xlabel("Exhaust Vacuum in cm Hg")

#PieChart
v_count = df['Exht_vac'].value_counts()
ax[0,1].pie([v_count[0], v_count[1],v_count[2]], labels=['Low', 'Medium',␣
↪'High'], shadow= True, explode = [0.1 , 0.1 , 0.1], autopct = '%1.0f%%')
ax[0,1].set_title("Pie Chart Showing Categorical Distribution of Exhaust␣
↪Vacuum")
ax[0,1].set_xlabel("Exhaust Vacuum as Low Medium and High")

#Box Plot
sns.boxplot(df['V'],ax= ax[1,0] , color  = c[2] )
ax[1,0].annotate(f"1st Quartile\n({round(qnt_1_V,1)})",xy=(qnt_1_V, 0.1),␣
↪xytext=(27,0.1), arrowprops={"arrowstyle":"->", "color":"red"})
ax[1,0].annotate(f"3rd Quartile\n({round(qnt_3_V,1)})",xy=(qnt_3_V, 0.1),␣
↪xytext=(75,0.1), arrowprops={"arrowstyle":"->", "color":"red"})
ax[1,0].annotate(f"Median\n({round(median_V,1)})",xy=(median_V, 0.2),␣
↪xytext=(32,0.2), arrowprops={"arrowstyle":"->", "color":"blue"})
ax[1,0].set_title("Box Plot showing Distribution of Exhaust Vacuum")
```

```
ax[1,0].set_xlabel("Exhaust Vacuum in cm Hg")
#BarChart
sns.countplot(df['Exht_vac'], ax = ax[1,1])
ax[1,1].set_title("Bar Plot showing Distribution of Exhaust Vacuum")
ax[1,1].set_xlabel("Categories of Exhaust Vacuum")
ax[1,1].annotate(f"[{round(exht_Vac_bins[0],1)}, {round(exht_Vac_bins[1],1)})",␣
 ↪xy=(-0.2,3300))
ax[1,1].annotate(f"[{round(exht_Vac_bins[1],1)}, {round(exht_Vac_bins[2],1)})",␣
 ↪xy=(0.8,3140))
ax[1,1].annotate(f"[{round(exht_Vac_bins[2],1)}, {round(exht_Vac_bins[3],1)}]",␣
 ↪xy=(1.8,3250))
plt.show()
```

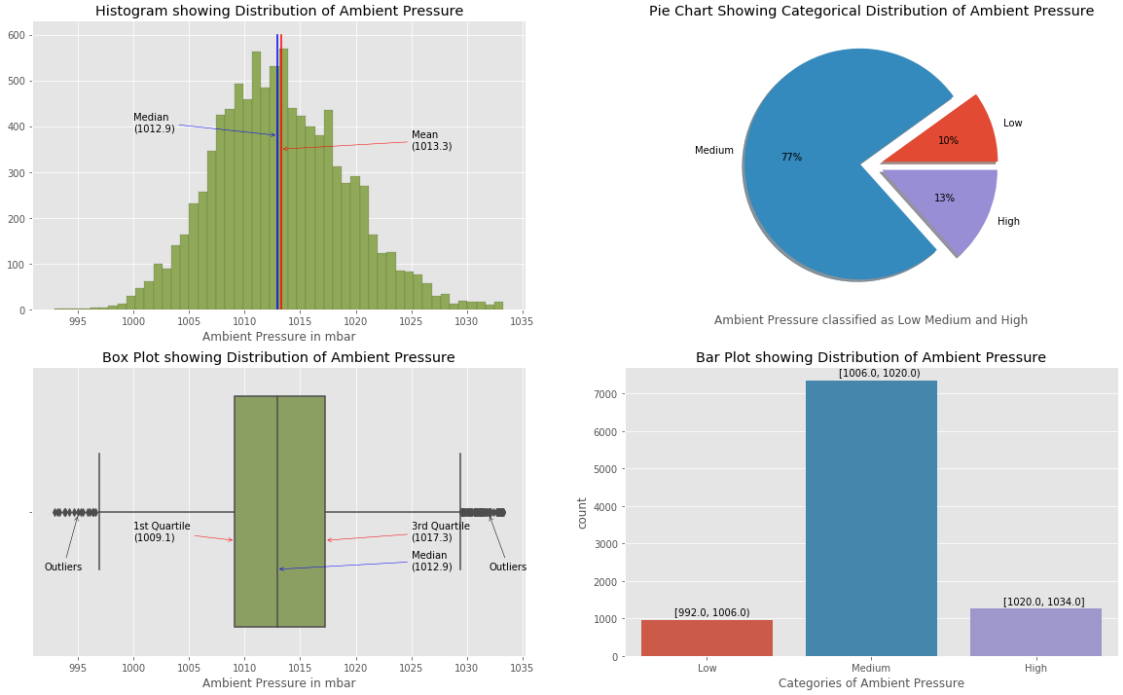


- **From the above Histogram, it can be oberved that the Distribution in kind of *Bimodal*.**
- **However the boxplot fails to capture such this *Bimodal distribution*. The box plot also shows that there are no outliers in the dataset.**
- **An intersting picture has been represented by the Pie-chart and the Bar Plots. The categorical distribution of values are almost equal in each of the categories**

2.3.4 Visualizations of Ambient Pressure

[13]:
```
mean_AP = df['AP'].mean()
median_AP = df['AP'].median()
mode_AP = df['AP'].mode()
qnt_1_AP = df['AP'].quantile(0.25)
qnt_3_AP = df['AP'].quantile(0.75)

fig, ax = plt.subplots(2,2, figsize=(20,12))
ax[0,0].hist(df['AP'], bins = 50, color  = c[3] , linewidth = 0.1, edgecolor =␣
↪'black')

#Histogram
ax[0,0].plot([mean_AP, mean_AP, mean_AP], [0, 200, 600], color = 'red' )
ax[0,0].plot([median_AP, median_AP], [0, 600] , color = 'blue')
ax[0,0].annotate(f"Mean\n({round(mean_AP,1)})",xy=(mean_AP, 350),␣
↪xytext=(1025,350), arrowprops={"arrowstyle":"->", "color":"red"})
ax[0,0].annotate(f"Median\n({round(median_AP,1)})",xy=(median_AP, 380),␣
↪xytext=(1000,390), arrowprops={"arrowstyle":"->", "color":"blue"})
ax[0,0].set_title("Histogram showing Distribution of Ambient Pressure")
ax[0,0].set_xlabel("Ambient Pressure in mbar")
#Pie chart
ap_count = df['Amb_Pres'].value_counts()
ax[0,1].pie([ap_count[0], ap_count[1],ap_count[2]], labels=['Low', 'Medium',␣
↪'High'], shadow= True, explode = [0.1 , 0.1 , 0.1], autopct = '%1.0f%%')
ax[0,1].set_title("Pie Chart Showing Categorical Distribution of Ambient␣
↪Pressure")
ax[0,1].set_xlabel("Ambient Pressure classified as Low Medium and High")
#BoxPlot
sns.boxplot(df['AP'],ax= ax[1,0], color  = c[3]  )
ax[1,0].annotate(f"1st Quartile\n({round(qnt_1_AP,1)})",xy=(qnt_1_AP, 0.1),␣
↪xytext=(1000,0.1), arrowprops={"arrowstyle":"->", "color":"red"})
ax[1,0].annotate(f"3rd Quartile\n({round(qnt_3_AP,1)})",xy=(qnt_3_AP, 0.1),␣
↪xytext=(1025,0.1), arrowprops={"arrowstyle":"->", "color":"red"})
ax[1,0].annotate(f"Median\n({round(median_AP,1)})",xy=(median_AP, 0.2),␣
↪xytext=(1025,0.2), arrowprops={"arrowstyle":"->", "color":"blue"})
ax[1,0].annotate("Outliers",xy=(995, 0.01), xytext=(992,0.2),␣
↪arrowprops={"arrowstyle":"->", "color":"black"})
ax[1,0].annotate("Outliers",xy=(1032, 0.01), xytext=(1032,0.2),␣
↪arrowprops={"arrowstyle":"->", "color":"black"})
ax[1,0].set_title("Box Plot showing Distribution of Ambient Pressure")
ax[1,0].set_xlabel("Ambient Pressure in mbar")
#Bar Chart
sns.countplot(df['Amb_Pres'], ax = ax[1,1])
ax[1,1].set_title("Bar Plot showing Distribution of Ambient Pressure")
ax[1,1].set_xlabel("Categories of Ambient Pressure")
```

```
ax[1,1].annotate(f"[{round(amb_pres_bins[0],1)}, {round(amb_pres_bins[1],1)})",␣
↪xy=(-0.2,1100))
ax[1,1].annotate(f"[{round(amb_pres_bins[1],1)}, {round(amb_pres_bins[2],1)})",␣
↪xy=(0.8,7500))
ax[1,1].annotate(f"[{round(amb_pres_bins[2],1)}, {round(amb_pres_bins[3],1)}]",␣
↪xy=(1.8,1400))
plt.show()
```

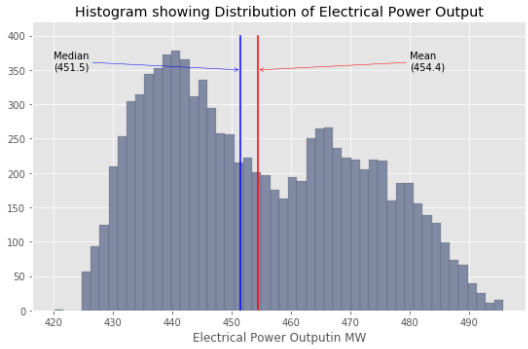


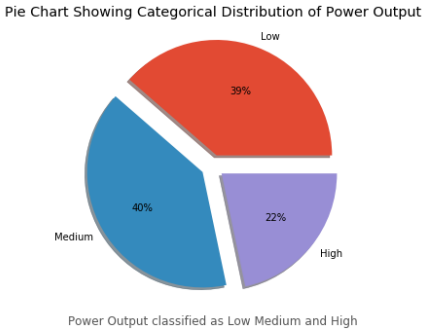
- **It can be observed from Histogram, that the values of Ambient Pressure are almost Normally distributed.**
- **The box plot on the lower left corner, shows that their are some outlier on the both side of the distribution**
- **The pie chart and the bar graph show that most of the values lie in the Medium Category and very less amount of Data is distributed in Low and High category, moreover the distribution in these respective categories are almost equal**

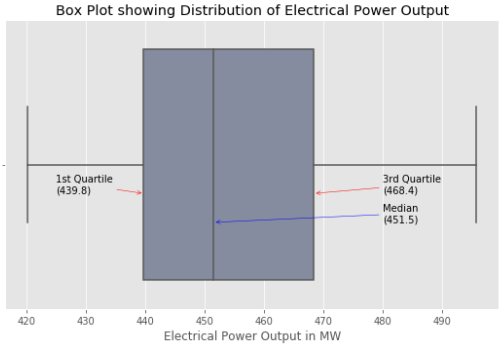
2.4 Visualizations of Power output

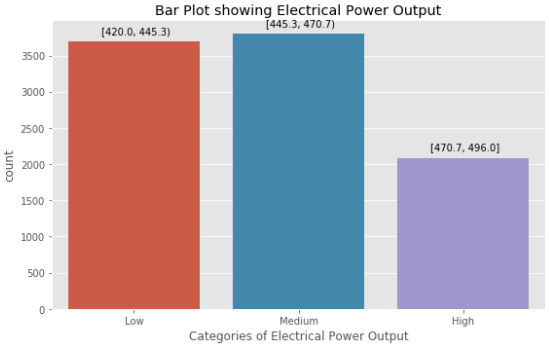
[14]:
```
mean_PE = df['PE'].mean()
median_PE = df['PE'].median()
mode_PE = df['PE'].mode()
qnt_1_PE = df['PE'].quantile(0.25)
qnt_3_PE = df['PE'].quantile(0.75)

fig, ax = plt.subplots(2,2, figsize=(20,12))
```

```
ax[0,0].hist(df['PE'], bins = 50, color  = c[4] , linewidth = 0.1, edgecolor =␣
 ↪'black')

ax[0,0].plot([mean_PE, mean_PE, mean_PE], [0, 200, 400], color = 'red' )
ax[0,0].plot([median_PE, median_PE], [0, 400] , color = 'blue')
ax[0,0].annotate(f"Mean\n({round(mean_PE,1)})",xy=(mean_PE, 350),␣
 ↪xytext=(480,350), arrowprops={"arrowstyle":"->", "color":"red"})
ax[0,0].annotate(f"Median\n({round(median_PE,1)})",xy=(median_PE, 350),␣
 ↪xytext=(420,350), arrowprops={"arrowstyle":"->", "color":"blue"})
ax[0,0].set_title("Histogram showing Distribution of Electrical Power Output")
ax[0,0].set_xlabel("Electrical Power Outputin MW")

pe_count = df['Pwr'].value_counts()
ax[0,1].pie([pe_count[0], pe_count[1],pe_count[2]], labels=['Low', 'Medium',␣
 ↪'High'], shadow= True, explode = [0.1 , 0.1 , 0.1], autopct = '%1.0f%%')
ax[0,1].set_title("Pie Chart Showing Categorical Distribution of Power Output")
ax[0,1].set_xlabel("Power Output classified as Low Medium and High")

sns.boxplot(df['PE'],ax= ax[1,0], color  = c[4]  )
ax[1,0].annotate(f"1st Quartile\n({round(qnt_1_PE,1)})",xy=(qnt_1_PE, 0.1),␣
 ↪xytext=(425,0.1), arrowprops={"arrowstyle":"->", "color":"red"})
ax[1,0].annotate(f"3rd Quartile\n({round(qnt_3_PE,1)})",xy=(qnt_3_PE, 0.1),␣
 ↪xytext=(480,0.1), arrowprops={"arrowstyle":"->", "color":"red"})
ax[1,0].annotate(f"Median\n({round(median_PE,1)})",xy=(median_PE, 0.2),␣
 ↪xytext=(480,0.2), arrowprops={"arrowstyle":"->", "color":"blue"})
ax[1,0].set_title("Box Plot showing Distribution of Electrical Power Output")
ax[1,0].set_xlabel("Electrical Power Output in MW")


sns.countplot(df['Pwr'], ax = ax[1,1])
ax[1,1].set_title("Bar Plot showing Electrical Power Output")
ax[1,1].set_xlabel("Categories of Electrical Power Output")
ax[1,1].annotate(f"[{round(pe_bins[0],1)}, {round(pe_bins[1],1)})", xy=(-0.
 ↪2,3800))
ax[1,1].annotate(f"[{round(pe_bins[1],1)}, {round(pe_bins[2],1)})", xy=(0.
 ↪8,3900))
ax[1,1].annotate(f"[{round(pe_bins[2],1)}, {round(pe_bins[3],1)}]", xy=(1.
 ↪8,2200))
plt.show()
```









- **From the above shown Histogram it can be observed that the Values of the Electrical Power Output is somewhat positively distributed. The same can also be inferred from thr Box-plot as well.**
- **In the Bar-chart and Pie-chart we can observe that the values of Electrical Power Output are equally distributed in the Low and Medium Categories, however, there are very less values that fall in the High Electrical Output Category**

2.5 Exploring Relationship between two variables

In this section we will be looking at the graphs that are mostly used for identifying the relationship between the two variable. The two variables in study may have Positve, Negative or correlation. These informations can be shown by using: 1. Scatter Plot 2. Heatmap

Lets us look at each of these plots one by one.

One thing to note here is that correlation does not show causality. Let us explore what this actually means.

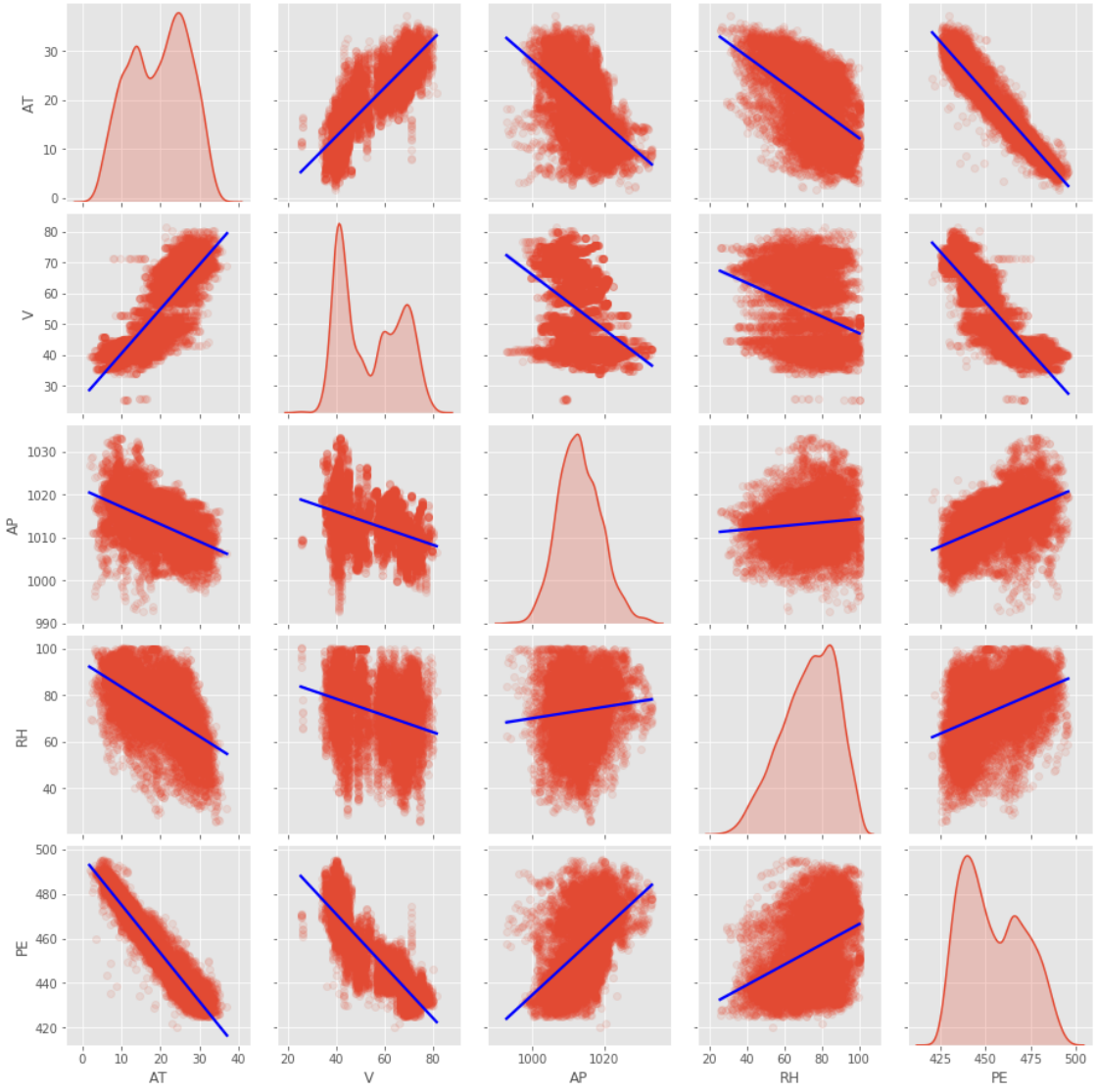
2.5.1 Scatter-plot (Graphical)

In the Scatter-plot, the two variables are plotted against each other, one each axis, and the relationship between the two variables can be inferred by observing the trend. Similarly, in the figure given below each variable is plotted against each other pair-wise and for ease of clarity trend line has also been plotted.

The positive slope can be interpreted as Positive Correlation and the Negative Slope can interpreted as Negative Correlation. The steepness of the slope of the trend line shows the magnitude of the

Correlation. Steeper the slope more is the correlation between each variable.

[15]:
```
sns.pairplot(df, diag_kind='kde',kind='reg', plot_kws={'line_kws':{'color':
 ↪'blue'}, 'scatter_kws': {'alpha': 0.1}})
plt.show()
```



2.5.2 Heat-map (Graphical & Non-Graphical Combined)

The Lighter Shades of color shows Positive Correlation and the Darker Shades of color shows the Negative Correlation. This can also be inferred by the color bar present on the right side of the graph as well as the values of correlation labelled in each box.

[16]:
```
fig, ax = plt.subplots(figsize = (12,10))
sns.heatmap(df[['AT', 'V', 'AP', 'RH' ,'PE']].corr(), annot = True, ax=ax,␣
 ↪linewidth = 0.1)
plt.show()
```

It can be seen from the Scatter-plot and Heat-map, that there is a Positive correlation between the Ambient Temperature and Exhaust Vacuum, a negative correlation between Ambient Temperature and Relative, and similar cases with other pairs of variables (Ambient Pressure, Ambient Temperature, Relative Humidity and Ehaust Vacuum). However, it must be mentioned that such variables are all independent variables, which implies that even if is a correlation between these variables, their variation does not impact or induce change in the other.

The study in hand, has Power Output (PE) as the dependent variable, and therefore it is more important to see as relationship between Independent variables and Dependent Variable, per the scope of our study.

It may also be mentioned that having a knowledge of correlation between independent

Variables is also very important and should not be overlooked. These correlations can help us in making more robust Machine Learning Models by utilizing the notions of Synergy Effects/ Iteraction Effects and Feature Engineering.

3 Conclusions

It can be seen that Exploratory Data Analysis is very essential for getting the basic understanding of the data. In this project we have looked at each variable and their distributions. We also looked at the relationship each variable shares with the other in the Scatter-plots & Heat-maps. The EDA gives us notions and information on how we could build our Machine Learning Model more robust, i.e. it has less variance as well as less bias. The scope of this study was to cover limited part of Exploratory Data Analysis. EDA also includedes Dimensionality Reduction and Cluster Analysis. However, these techiques are outside the scope of present project.

-Project Prepared by Arpit Malhotra

(PGPDS'July2020)

GitHub link to view the Jupyter Notebook is:

https://github.com/arpitmalhotra009/DaysOfDataVisualization/tree/master/Power%20Output%20EDA

4 Bibliograpghy

The data has been downloaded from "The UC Irvine Machine Learning Repository"

4.1 Citations:

1. Pınar Tüfekci, Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods, International Journal of Electrical Power & Energy Systems, Volume 60, September 2014, Pages 126-140, ISSN 0142-0615, http://dx.doi.org/10.1016/j.ijepes.2014.02.027. (http://www.sciencedirect.com/science/article/pii/S0142061514000908)
2. Heysem Kaya, Pınar Tüfekci , Sadık Fikret Gürgen: Local and Global Learning Methods for Predicting Power of a Combined Gas & Steam Turbine, Proceedings of the International Conference on Emerging Trends in Computer and Electronics Engineering ICETCEE 2012, pp. 13-18 (Mar. 2012, Dubai)