Proposed Work:

1. Dataset:

As mentioned above, according to the approach we decided, we have selected a dataset that is generated from one solar power plant for a specific period so that we can get data from all the seasons considering the climatic conditions, weather conditions, and many more. The weather data and energy produced is recorded every 3 hours a day. With the help of this dataset, we would be able to forecast and predict for a particular site along with keeping the weather conditions in check. Below is the schema of the dataset.

S. No.	Column	Datatype
1	Day_of_Year	Integer Type
2	Year	Integer Type
3	Month	Integer Type
4	Day	Integer Type
5	First_Hour_of_Period	Integer Type
6	Is_Daylight	String Type
7	Distance_to_Solar_Noon	Float Type
8	Average_Temperature_Day	Float Type
9	Average_Wind_Direction_Day	Float Type
10	Average_Wind_Speed_Day	Float Type
11	Sky_Cover	Integer Type
12	Visibility	Float Type
13	Relative_Humidity	Float Type
14	Average_Wind_Speed_Period	Integer Type
15	Average_Barometric_Pressure_Period	Float Type
16	Power_Generated	Float Type

2. Data Pre-processing:

	Day of Year	Year	Month	Day	First Hour of Period	Is Daylight	Distance to Solar Noon	Average Temperature (Day)	Average Wind Direction (Day)	Average Wind Speed (Day)	Sky Cover	Visibility	Relative Humidity	Average Wind Speed (Period)	Average Barometric Pressure (Period)	Power Generated
0	245	2008	9	1	1	False	0.859897	69	28	7.5	0	10.0	75	8.0	29.82	0
1	245	2008	9	1	4	False	0.628535	69	28	7.5	0	10.0	77	5.0	29.85	0
2	245	2008	9	1	7	True	0.397172	69	28	7.5	0	10.0	70	0.0	29.89	5418
3	245	2008	9	1	10	True	0.165810	69	28	7.5	0	10.0	33	0.0	29.91	25477
4	245	2008	9	1	13	True	0.065553	69	28	7.5	0	10.0	21	3.0	29.89	30069
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2915	243	2009	8	31	10	True	0.166453	63	27	13.9	4	10.0	75	10.0	29.93	6995
2916	243	2009	8	31	13	True	0.064020	63	27	13.9	1	10.0	66	15.0	29.91	29490
2917	243	2009	8	31	16	True	0.294494	63	27	13.9	2	10.0	68	21.0	29.88	17257
2918	243	2009	8	31	19	True	0.524968	63	27	13.9	2	10.0	81	17.0	29.87	677
2919	243	2009	8	31	22	False	0.755442	63	27	13.9	1	10.0	81	11.0	29.90	0

2920 rows × 16 columns

```
df2 = df.copy()
df2['Power Generated'] = df2['Power Generated'].where(df2['Power Generated'] == 0)
df2 = df2.dropna()
print(df2['First Hour of Period'].value_counts())
del df2

#Entire year power generated at times 1,4,22 is always zero. Hence, reducing time period from 7 to 19.
```

```
22     365
4      365
1      365
19     148
7       53
13       9
10       8
16       7
Name: First Hour of Period, dtype: int64
```

- As we can see power generated at times 1,4,22 is always zero. Hence, we shortened the time period from 7 to 19.
- After that we reset and dropped and index, because after shortening the time range, indexes were not updated.

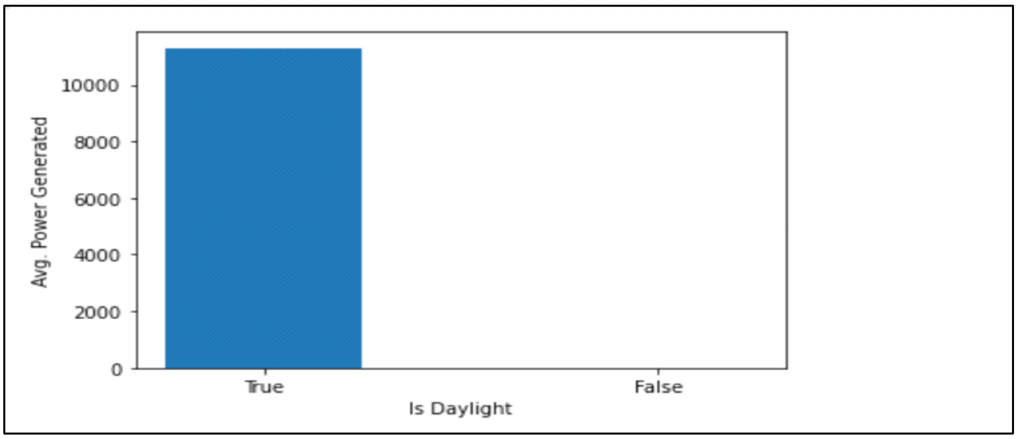
```
df['First Hour of Period'] = df['First Hour of Period'].where(df['First Hour of Period'] != 1)
df['First Hour of Period'] = df['First Hour of Period'].where(df['First Hour of Period'] != 4)
df['First Hour of Period'] = df['First Hour of Period'].where(df['First Hour of Period'] != 22)
df = df.dropna()
df.reset_index(inplace=True,drop=True)
df
```

	Day of Year	Year	Month	Day	First Hour of Period	Is Daylight	Distance to Solar Noon	Average Temperature (Day)	Average Wind Direction (Day)	Average Wind Speed (Day)	Sky Cover	Visibility	Relative Humidity	Average Wind Speed (Period)	Average Barometric Pressure (Period)	Power Generated
0	245	2008	9	1	7.0	True	0.397172	69	28	7.5	0	10.0	70	0.0	29.89	5418
1	245	2008	9	1	10.0	True	0.165810	69	28	7.5	0	10.0	33	0.0	29.91	25477
2	245	2008	9	1	13.0	True	0.065553	69	28	7.5	0	10.0	21	3.0	29.89	30069
3	245	2008	9	1	16.0	True	0.296915	69	28	7.5	0	10.0	20	23.0	29.85	16280
4	245	2008	9	1	19.0	True	0.528278	69	28	7.5	0	10.0	36	15.0	29.83	515
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1819	243	2009	8	31	7.0	True	0.396927	63	27	13.9	4	10.0	87	9.0	29.90	464
1820	243	2009	8	31	10.0	True	0.166453	63	27	13.9	4	10.0	75	10.0	29.93	6995
1821	243	2009	8	31	13.0	True	0.064020	63	27	13.9	1	10.0	66	15.0	29.91	29490
1822	243	2009	8	31	16.0	True	0.294494	63	27	13.9	2	10.0	68	21.0	29.88	17257
1823	243	2009	8	31	19.0	True	0.524968	63	27	13.9	2	10.0	81	17.0	29.87	677

1824 rows × 16 columns

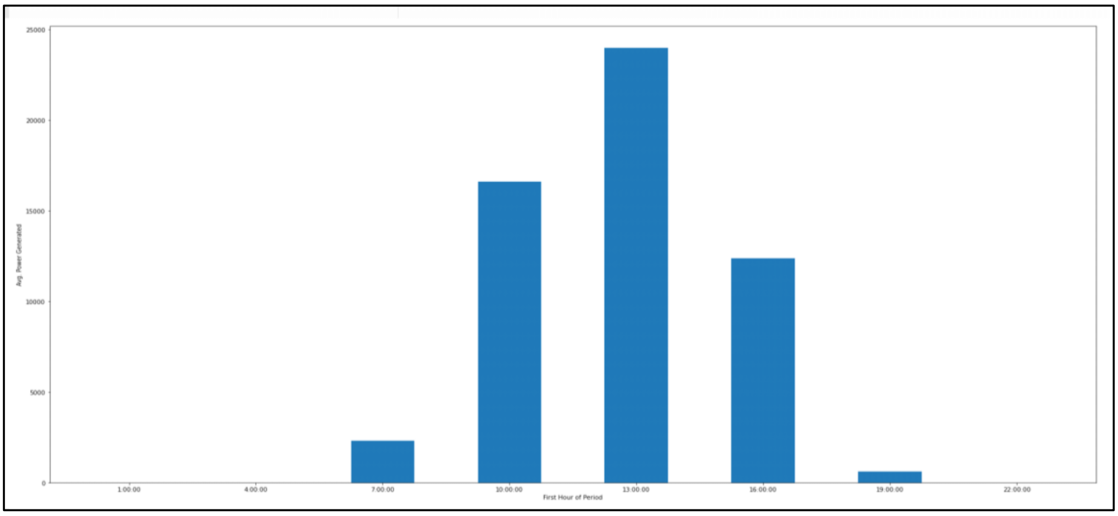
3. Analysis:

- *Avg. Power Generated vs Is Daylight*



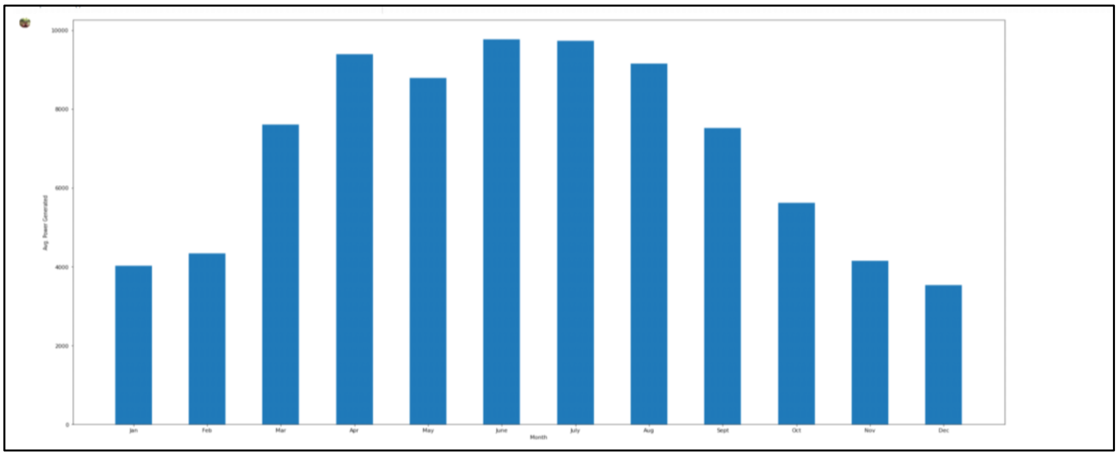
The graph plotted above is between Avg. Power Generated vs Is Daylight. This graph shows us that in the absence of daylight there is no power generation.

- *Avg. Power Generated vs First Hour of Period*



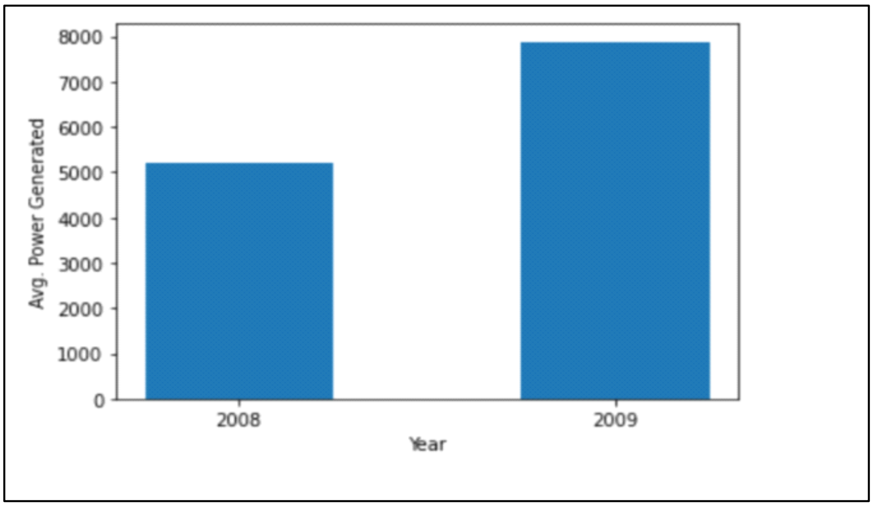
The graph plotted above is between Avg. Power Generated vs First Hour of Period. The graph shows us that during daytime power generation increases. As the day progresses, power generation is less in early morning times, gradually increases in the afternoon, and then decreases again at night. This can be seen by the abscissa which represents the time at which the avg. power generation varies.

- *Avg. Power Generated per Month*



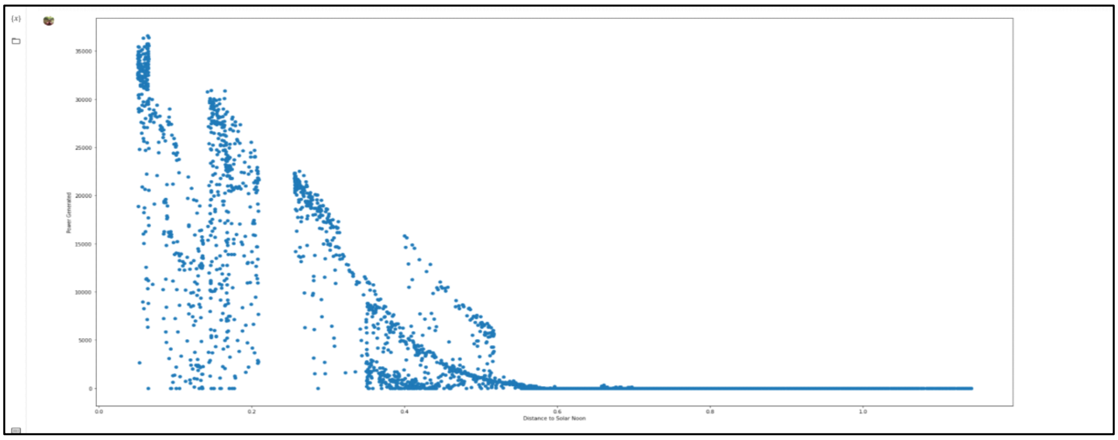
The graph plotted above is between Avg. Power Generated per Month. The graph shows that the average power generated in winter months from November-February is significantly lower as compared to the warmer months. This can be seen by the abscissa which represents the month at which the avg. power generation varies.

- *Avg. Power Generated per Year*



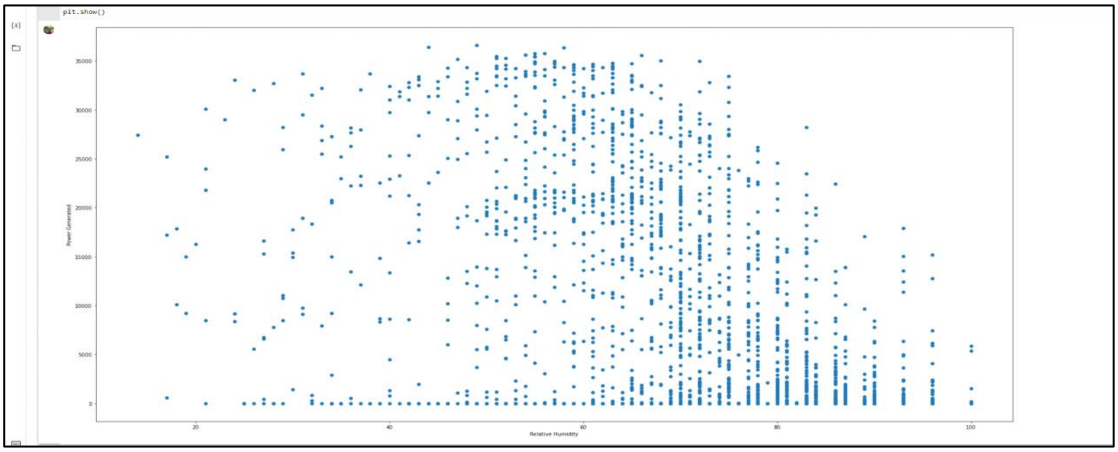
The graph plotted above is between Avg. Power Generated per Year. The graph shows that as we progress year by year, avg. power generation is increasing (there can be multiple reasons behind it but the prominent reason is that as global warming is increasing ozone layer is depleting and thus, sun radiation increases). This can be seen by the abscissa which represents the year at which the avg. power generation varies.

- *Avg. Power Generated vs Distance to Solar Noon*



The graph plotted above is between Avg. Power Generated vs Distance to Solar Noon. The graph shows that as the distance of the sun decreases the avg. power generation increases. This can be seen by the dots which represents the distance to solar noon at which the avg. power generation varies.

- *Avg. Power Generated vs Relative Humidity*



The graph plotted above is between Avg. Power Generated vs Relative Humidity. The avg. relative humidity of Berkeley is around 60% and thus most of the recordings are of this range, making it a prominent parameter to be considered. The graph shows that in between 50% to 70% the power generation is maximum. This can be seen by the dots which represent the relative humidity at which the avg. power generation varies.

4. Implementation Details:

In order to implement various time series forecasting models and make a comparative analysis among them, few pre-analyses were done.

- ***<u>Implementing Augmented Dickey-Fuller Method</u>* -**

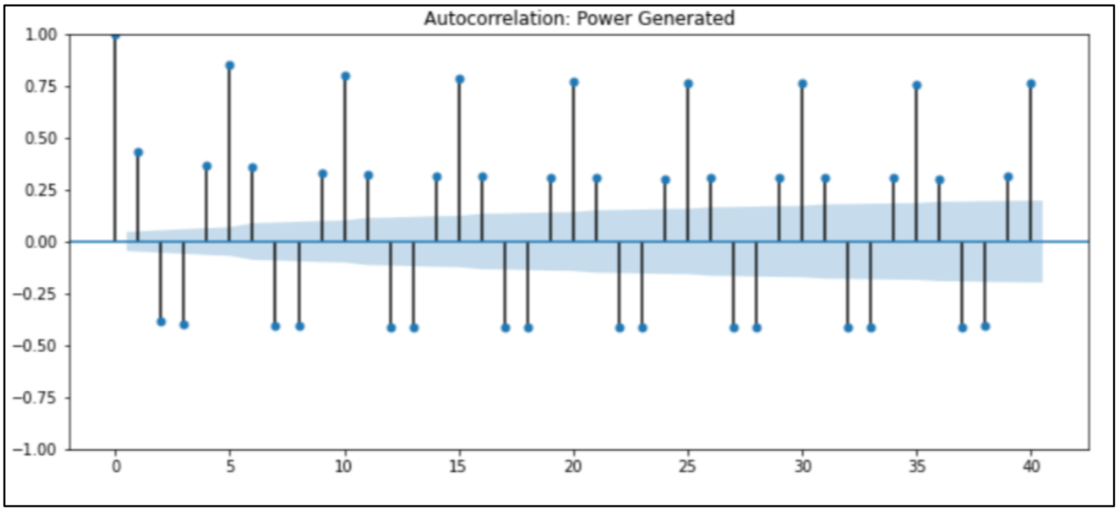
After implementing this, we came to know that our dataset is a stationary dataset i.e. it does not exhibit seasonality and trend. From this we made an initial assumption that ARMA should work best, because there is no need of differencing.

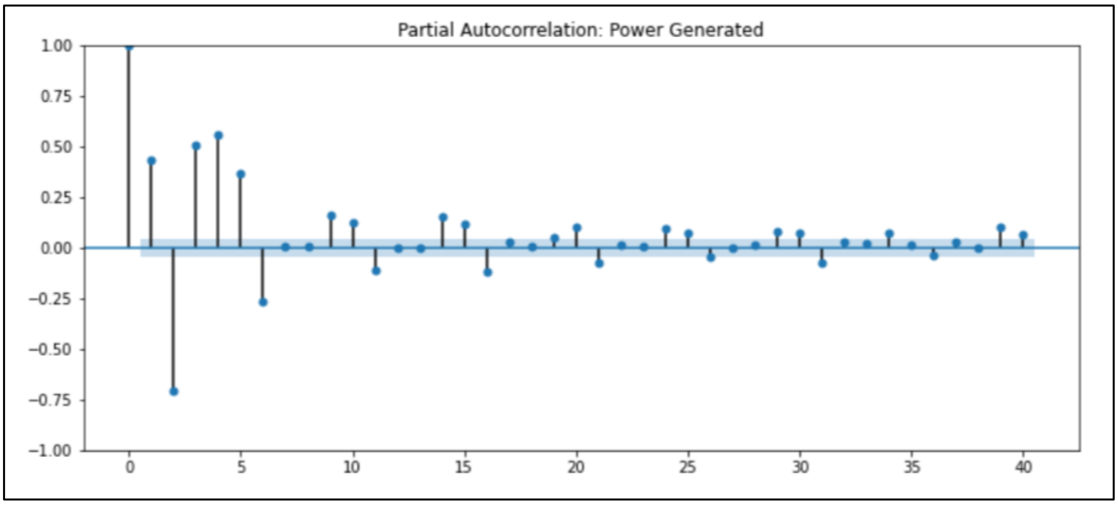
```
from statsmodels.tsa.stattools import adfuller

#creating a report
dftest = adfuller(df['Power Generated'])
dfout = pd.Series(dftest[0:4],index=['ADF test statistic','p-value','# lags used','# observations'])

for key,val in dftest[4].items():          #for critical value 1%, 5%, 10%
    dfout[f'critical value ({key})']=val
print(dfout)

#Our data is stationary, so seasonal component is false
```

```
ADF test statistic        -3.741219
p-value                    0.003569
# lags used               25.000000
# observations          1798.000000
critical value (1%)       -3.433992
critical value (5%)       -2.863149
critical value (10%)      -2.567626
dtype: float64
```

Since, the p-value < 0.05, we can safely say that the data is stationary.

Plotting the ACF and PACF graphs, we can make similar conclusion.



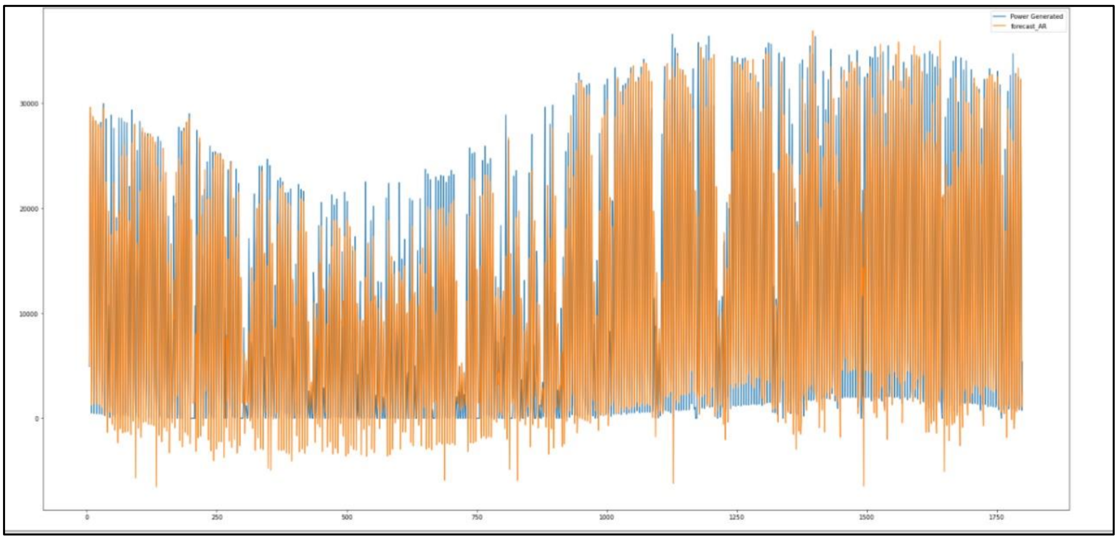


Analyzing ACF and PACF plots also helped us to select order of models -
p: 5
q: 1

- ***<u>Implementing AR model</u>* –**

We selected the lag=5, based on the ACF and PACF plot and got following results.

```
from statsmodels.tsa.ar_model import AutoReg, ARResults

model = AutoReg(df['Power Generated'],lags=5)
ARfit = model.fit()
ARfit.params

const                  1467.317522
Power Generated.L1        0.615297
Power Generated.L2       -0.430651
Power Generated.L3        0.054091
Power Generated.L4        0.263170
Power Generated.L5        0.365969
dtype: float64
```



Data Forecast using AR(5) model

- ***Implementing ARMA model*** **–**

We trained our model using ARMA (5,1) model and got following results.

```python
from statsmodels.tsa.arima_model import ARMA,ARMAResults,ARIMA,ARIMAResults
import statsmodels.api as sm

model = sm.tsa.ARIMA(df['Power Generated'],order=(5,0,1))
ARMAfit = model.fit()
ARMAfit.summary()
```

SARIMAX Results

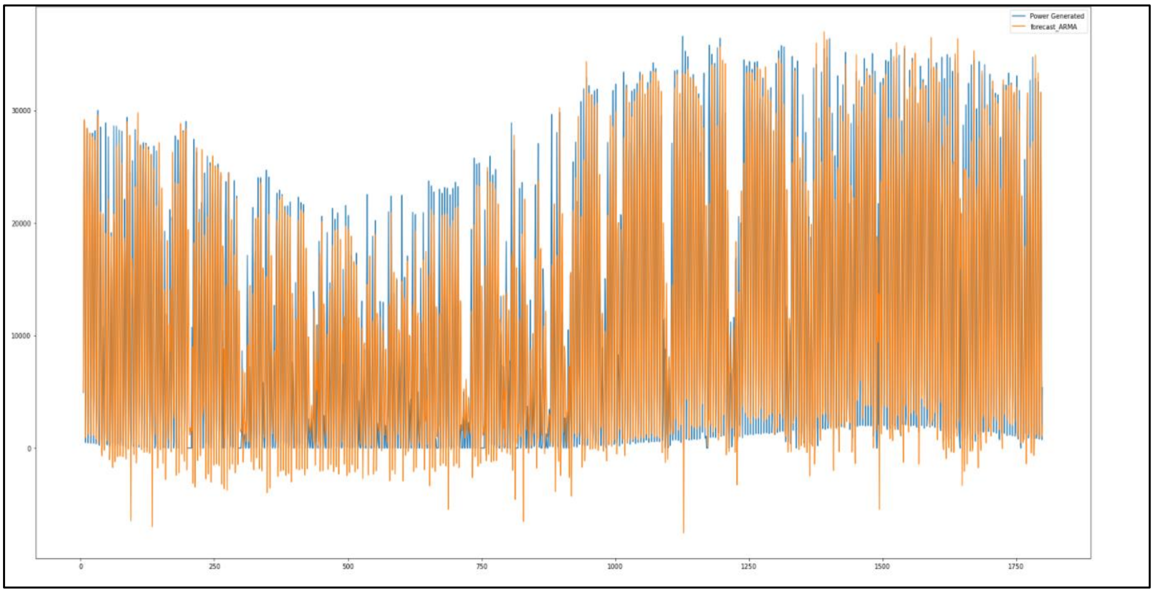
Dep. Variable:	Power Generated	**No. Observations:**	1824
Model:	ARIMA(5, 0, 1)	**Log Likelihood**	-17944.612
Date:	Sun, 28 Nov 2021	**AIC**	35905.224
Time:	16:20:07	**BIC**	35949.295
Sample:	0 - 1824	**HQIC**	35921.482
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	1.117e+04	1081.607	10.331	0.000	9053.879	1.33e+04
ar.L1	0.2150	0.040	5.402	0.000	0.137	0.293
ar.L2	-0.1091	0.037	-2.972	0.003	-0.181	-0.037
ar.L3	-0.1205	0.028	-4.264	0.000	-0.176	-0.065
ar.L4	0.1994	0.017	11.892	0.000	0.167	0.232
ar.L5	0.6060	0.023	25.968	0.000	0.560	0.652
ma.L1	0.4761	0.042	11.465	0.000	0.395	0.558
sigma2	2.172e+07	0.007	3.25e+09	0.000	2.17e+07	2.17e+07

Ljung-Box (L1) (Q):	1.83	**Jarque-Bera (JB):**	822.02
Prob(Q):	0.18	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.92	**Skew:**	0.07
Prob(H) (two-sided):	0.27	**Kurtosis:**	6.29

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
[2] Covariance matrix is singular or near-singular, with condition number 1.83e+26. Standard errors may be unstable.

(Since Integrating component (d) is absent, we put d=0)



Data Forecast using ARMA (5,1) model

- ***Implementing ARIMA model*** –

We trained our model using ARIMA (5,1,1) model and got following results.

```
from statsmodels.tsa.arima_model import ARMA,ARMAResults,ARIMA,ARIMAResults
import statsmodels.api as sm

model = sm.tsa.ARIMA(df['Power Generated'],order=(5,1,1))
ARIMAfit = model.fit()
ARIMAfit.summary()
```

SARIMAX Results

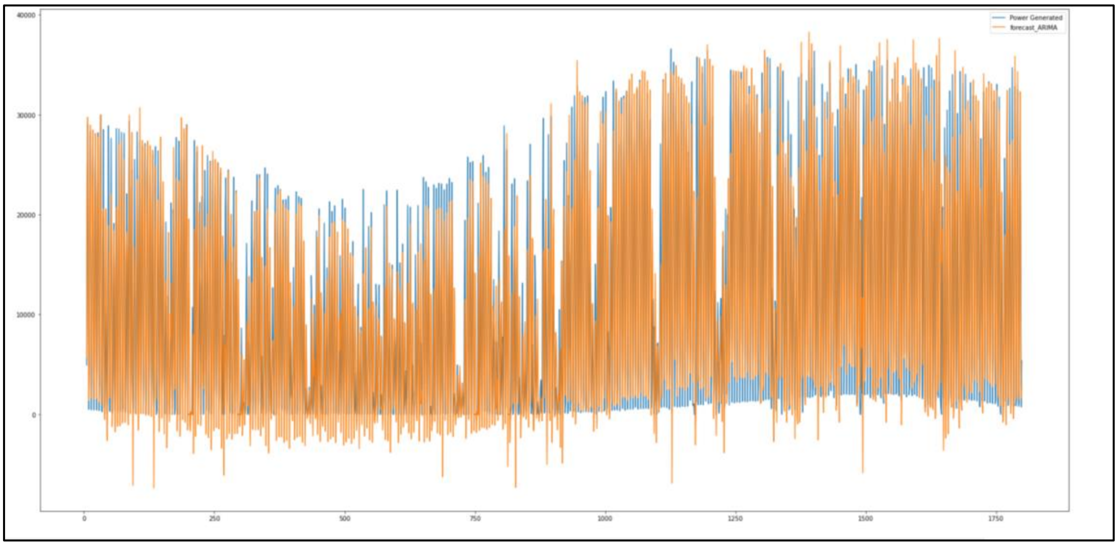
Dep. Variable:	Power Generated	**No. Observations:**	1824
Model:	ARIMA(5, 1, 1)	**Log Likelihood**	-17949.366
Date:	Sun, 28 Nov 2021	**AIC**	35912.732
Time:	16:20:12	**BIC**	35951.290
Sample:	0	**HQIC**	35926.956
	- 1824		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-0.4103	0.122	-3.354	0.001	-0.650	-0.171
ar.L2	-0.6562	0.051	-12.771	0.000	-0.757	-0.555
ar.L3	-0.6107	0.097	-6.289	0.000	-0.801	-0.420
ar.L4	-0.4118	0.085	-4.827	0.000	-0.579	-0.245
ar.L5	0.1811	0.057	3.171	0.002	0.069	0.293
ma.L1	0.1588	0.124	1.279	0.201	-0.085	0.402
sigma2	2.091e+07	9.82e-09	2.13e+15	0.000	2.09e+07	2.09e+07

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	1104.37
Prob(Q):	0.99	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.91	**Skew:**	0.25
Prob(H) (two-sided):	0.26	**Kurtosis:**	6.78

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
[2] Covariance matrix is singular or near-singular, with condition number 7.76e+30. Standard errors may be unstable.

(Since, our data is stationary and doesn't require differencing we selected d = 1, as selecting large 'd' will result in reduced accuracy)



Data Forecast using ARIMA (5,1,1) model

- ***Implementing SARIMA model*** –

We implemented SARIMA with order (1,1,1) and seasonal order (5,1,1,5) and got following results.

```python
from statsmodels.tsa.arima_model import ARIMA
import statsmodels.api as sm

model = sm.tsa.SARIMAX(df['Power Generated'],order = (1,1,1),seasonal_order = (5,1,1,5))
SARIMAfit = model.fit()
SARIMAfit.summary()
```

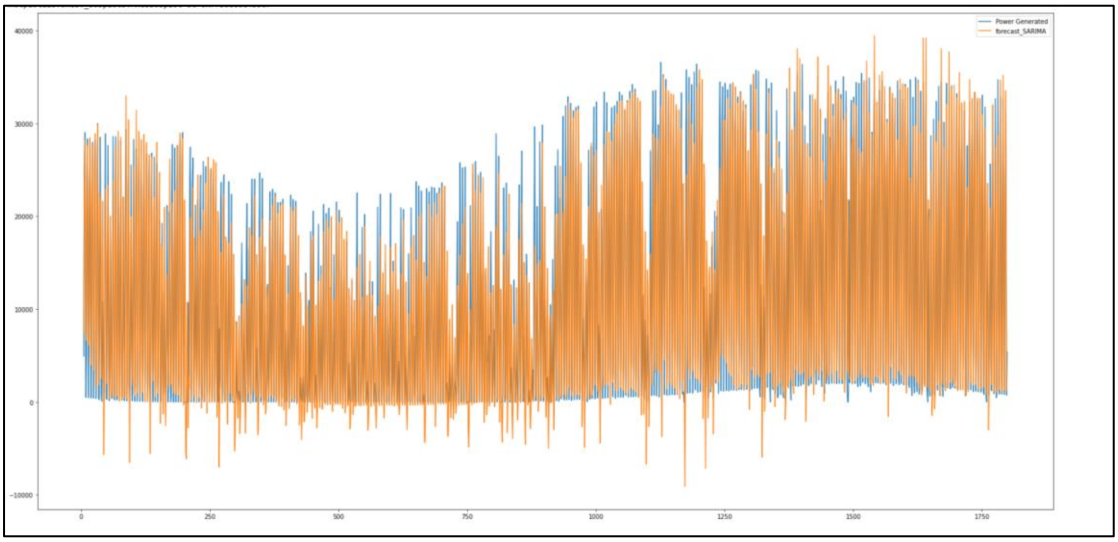
SARIMAX Results

Dep. Variable:	Power Generated	**No. Observations:**	1824
Model:	SARIMAX(1, 1, 1)x(5, 1, 1, 5)	**Log Likelihood**	-17864.353
Date:	Sun, 28 Nov 2021	**AIC**	35746.705
Time:	16:20:47	**BIC**	35796.255
Sample:	0	**HQIC**	35764.987
	- 1824		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	0.5654	0.017	33.958	0.000	0.533	0.598
ma.L1	-0.9993	0.024	-42.395	0.000	-1.046	-0.953
ar.S.L5	0.4469	0.025	18.209	0.000	0.399	0.495
ar.S.L10	0.0866	0.028	3.134	0.002	0.032	0.141
ar.S.L15	0.1695	0.028	6.124	0.000	0.115	0.224
ar.S.L20	0.0742	0.028	2.661	0.008	0.020	0.129
ar.S.L25	0.1447	0.024	6.072	0.000	0.098	0.191
ma.S.L5	-0.9962	0.009	-106.065	0.000	-1.015	-0.978
sigma2	2.522e+07	3.93e-10	6.42e+16	0.000	2.52e+07	2.52e+07

Ljung-Box (L1) (Q):	18.81	**Jarque-Bera (JB):**	832.62
Prob(Q):	0.00	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.97	**Skew:**	-0.46
Prob(H) (two-sided):	0.69	**Kurtosis:**	6.18

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
[2] Covariance matrix is singular or near-singular, with condition number 6.33e+30. Standard errors may be unstable.

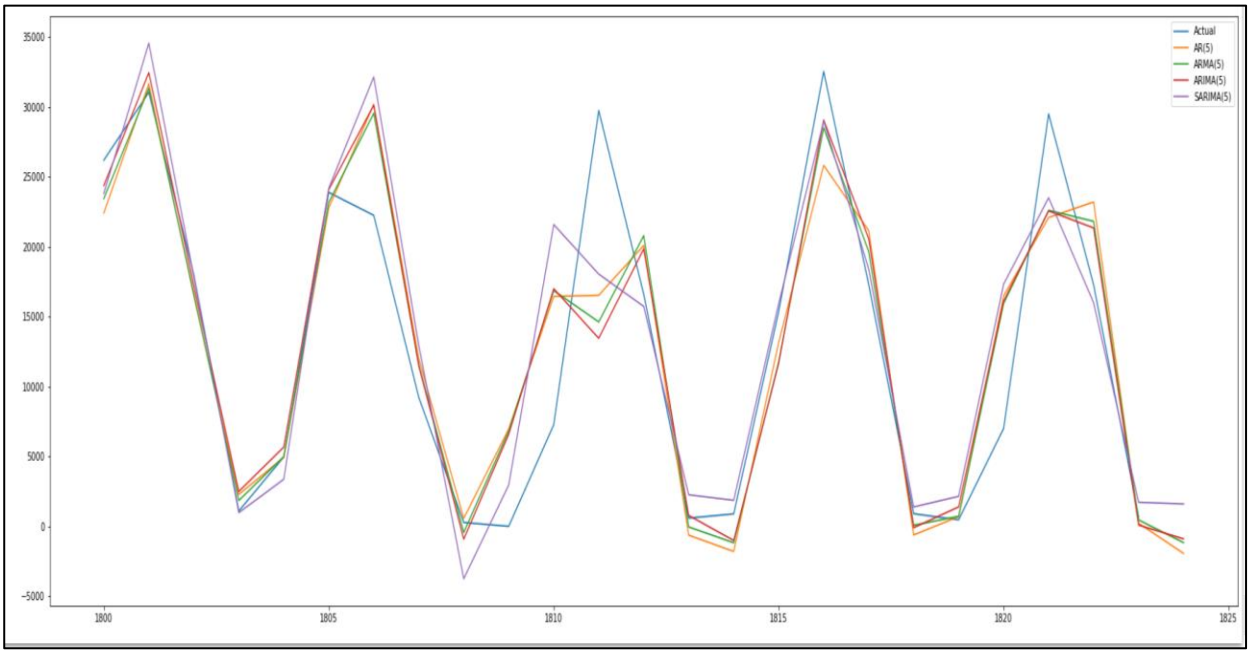


Data Forecast using SARIMA (1,1,1)x(5,1,1,5) model

- ***Results*** **–**

Model	RMSE	R2_Score	Time Taken
AR (5)	4649.9906	0.8244	0.011s
ARMA (5,1)	4514.9320	0.8344	0.684s
ARIMA (5,1,1)	4556.1534	0.8314	2.041s
SARIMA (1,1,1)x(5,1,1,5)	4391.1514	0.8434	35.449s

Based on upon table, we can choose ARMA model over SARIMA model because of difference in execution time. Hence, the assumption made earlier is verified.



➔ If we take a closer look at some of the values of dataset, we can see that although all the four models are predicting near to each other, ARMA is fitting better than others in most cases.

5. Future Scope:

- We can implement SARIMAX model and include exogenous variables which affected the Power Generated and achieve higher accuracy. We can also implement more complex models like LSTM, VAR, VARMA etc.
- Using these models, we can make a system for each region within a state and then predict power generation at future time points and calculate the cumulative generation that a state can generate each day.