IE 360 Statistical Forecasting and Time Series

Spring 2024

Project - Solar Power Forecasting

Group Members:

Gizem Ergin - 2019402018

İdil Zeynep Ceylan - 2020402033

Emre Çağan Kanlı- 2020402072

1. Introduction

This project focuses on developing an hourly solar power forecasting model for the Edikli Güneş Enerjisi Santrali located in Niğde, Turkey which is situated at 38.29° North latitude and 34.97° East longitude. The forecasting model will generate 24 hourly predictions for day d+1, using data available up to day d-1. Weather measurements from 25 grid points around the power plant will be utilized to enhance the model's accuracy. These measurements include:

- DSWRF_surface: Downward shortwave radiation flux at the surface, which is highly related to solar power production levels.
- USWRF_top_of_atmosphere: Upward shortwave radiation flux at the top of the atmosphere.
- USWRF_surface: Upward shortwave radiation flux at the surface.
- DLWRF_surface: Downward longwave radiation flux at the surface.
- TCDC_low.cloud.layer: Total cloud cover percentage for the low cloud layer.
- TCDC_middle.cloud.layer: Total cloud cover percentage for the middle cloud layer.
- TCDC_high.cloud.layer: Total cloud cover percentage for the high cloud layer.
- TCDC_entire.atmosphere: Total cloud cover percentage for the entire atmosphere.
- CSNOW_surface: Categorical snow variable, indicating whether it snows (1) or not (0).
- TMP_surface: Temperature at the surface, which affects both seasonal patterns and the efficiency of solar panels.

By examining the paired correlations between these variables and the production data, we can identify the relationships that influence solar power output. This analysis will guide the selection of regressors for the prediction models.

2. Relevant Literature

The paper "Forecasting of Total Daily Solar Energy Generation using ARIMA: A Case Study" by Sharif Atique investigates the application of the ARIMA model for predicting daily solar energy generation from a solar panel system. The study focuses on transforming non-stationary, seasonal time series data into a stationary format suitable for ARIMA modeling. The model's parameters were selected and validated using statistical techniques like Akaike Information Criterion (AIC) and residual sum of squares (SSE). The authors found that the ARIMA model, despite its simplicity and linear assumptions, can serve as an effective benchmark for solar energy forecasting, although future improvements could enhance its accuracy.

Secondly, The Box–Jenkins time series approach was used to create an ARIMA model in the study "Forecasting of Demand Using ARIMA Model,[1]" which forecasted demand for a finished product in the setting of food manufacturing. Several models were developed using historical demand data, and the best one was chosen based on four performance criteria: maximum likelihood, standard error, AIC, and SBC.

In our project, we utilized the insights and methodologies from these papers to develop our own solar energy forecasting model. By applying similar data transformation and validation techniques, we ensured that our time series data were appropriately prepared for ARIMA modeling. This enabled us to predict solar energy generation more accurately. The paper provided a robust framework that guided our approach to handling and analyzing solar energy data.

3. Data Analysis

Before delving into the modeling process, a thorough analysis of the provided data is essential. This initial step involves examining the data for patterns, correlations, and any anomalies that could impact the forecasting model's accuracy. By understanding the relationships between the weather variables and solar power production, we can make informed decisions about which variables to include in our model. The following section will detail the data analysis steps undertaken to prepare the dataset for effective modeling:
First, the weather and production data are merged based on the 'date' and 'hour' columns to form a comprehensive dataset containing both weather and production information. The data is then grouped by 'date' and 'hour', and the mean value for each group is calculated to handle multiple measurements from different longitudes and latitudes. This method relies on averaging the measurements from different coordinates for the same date and hour to derive the average measurements.

[1] Fattah, Jamal, Latifa Ezzine, Zineb Aman, Haj El Moussami, and Abdeslam Lachhab. “Forecasting of demand using ARIMA model.” *International Journal of Engineering Business Management* 10 (January 1, 2018): 184797901880867. https://doi.org/10.1177/1847979018808673.

```
data_grouped = data.groupby(['date', 'hour']).mean()

# Optionally, you can drop the 'lat' and 'lon' columns
data_grouped = data_grouped.drop(columns=['lat', 'lon'])

# Reset index
data_grouped = data_grouped.reset_index()

data_grouped
```

Figure 1: *Code Snippet for Averaging*

As the first step, the autocorrelation function (ACF) of the solar power production data is analyzed to understand its temporal dependencies, considering hourly lags up to 7 days (168 hours). This plot reveals how production values are correlated with their past values, helping to identify daily and weekly patterns. Understanding these temporal dependencies is crucial for accurate forecasting, as it guides the selection of appropriate time series models, such as ARIMA, which can incorporate significant lags effectively.

```
import matplotlib.pyplot as plt
from statsmodels.graphics.tsaplots import plot_acf

# Plot ACF for 'production' column with hourly lags
plt.figure(figsize=(10, 6))
plot_acf(data_grouped['production'], lags=24*7)  # Plot for 7 days of hourly lags
plt.xlabel('Lag (hours)')
plt.ylabel('Autocorrelation')
plt.title('Autocorrelation Function (ACF) for Production (Hourly Lags)')
plt.show()
```

Figure 2: *Code Snippet for ACF Plot*

The output of this analysis is:
The ACF plot shows clear and repeating spikes at regular intervals, indicating strong periodic patterns in the solar power production data. These spikes occur approximately every 24 hours, reflecting a daily cycle in solar power generation. The plot shows negative autocorrelation at certain lags, which could indicate some level of inverse relationship at those points. This may happen due to the nighttime hours when solar production is zero.

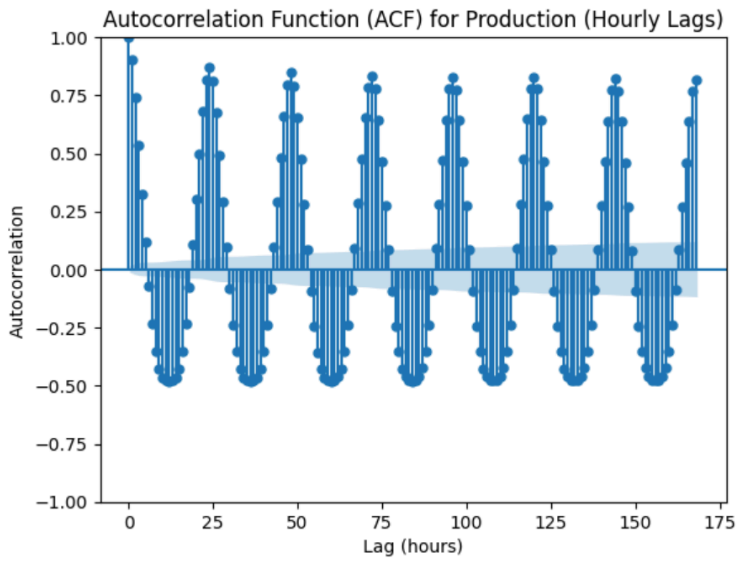


Figure 3: *ACF Plot for Production*

Secondly, a correlation analysis is performed on the weather variables to understand the relationships between them. High correlation coefficients (close to 1 or -1) indicate strong relationships, which can be useful for feature selection in the modeling process. Variables that are highly correlated with each other might provide redundant information. Identifying these can help in reducing multicollinearity in the model, leading to more stable and interpretable results.

```
import seaborn as sns

# Select the columns containing variables (columns 2-11)
variable_columns = data_grouped.iloc[:, 2:11]

# Calculate the correlation matrix
correlation_matrix = variable_columns.corr()

# Plot the correlation matrix as a heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix for Variable Pairs (data_grouped)')
plt.show()
```

Figure 4: *Code Snippet for Correlation Matrix*

The correlation matrix is printed as a heatmap as follows:

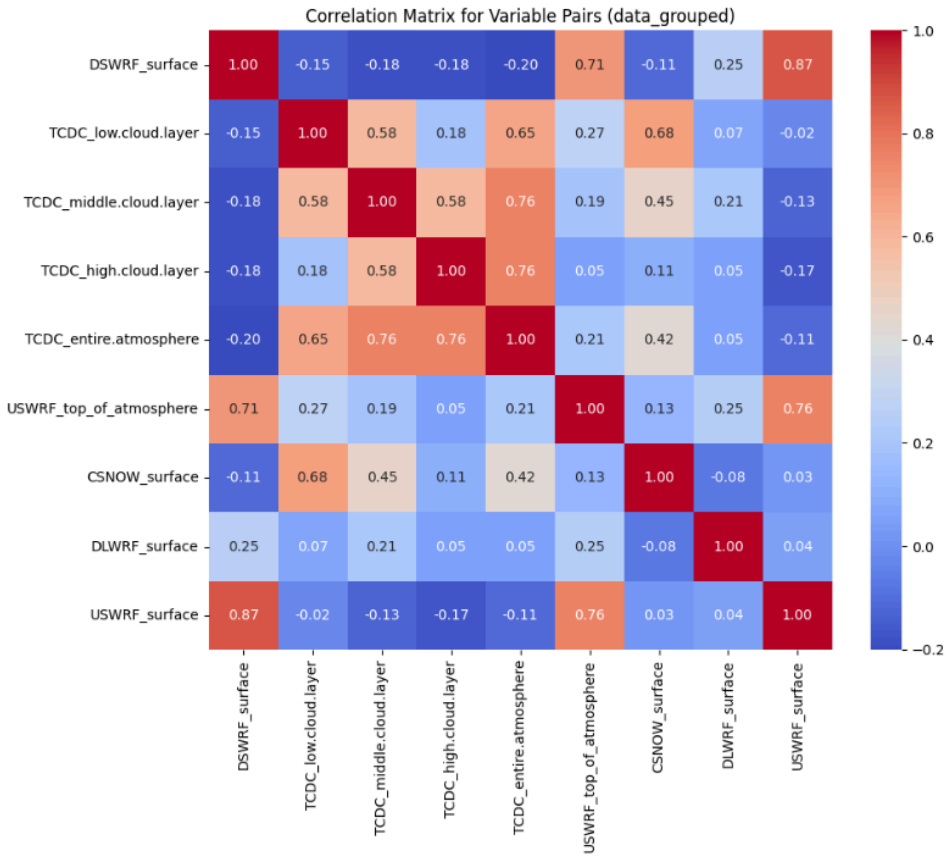


Figure 5:*Correlation Matrix for Variable Pairs*

The correlations matrix signals the following:

- Strong positive correlation between DSWRF_surface and USWRF_surface (0.87). This suggests that these two variables are closely related and tend to move together.
- Strong positive correlation between TCDC_entire.atmosphere and TCDC_high.cloud.layer (0.76)
- Similarly, strong positive correlation between TCDC_entire.atmosphere and TCDC_middle.cloud.layer (0.76)
- Strong positive correlation between USWRF_surface and USWRF_top_of_atmosphere (0.76)
- Strong positive correlation between DSWRF_surface and USWRF_top_of_atmosphere (0.71), indicating that radiation levels at different atmospheric levels are related.
- Moderate positive correlation between TCDC_low.cloud.layer and CSNOW_surface (0.68)
- Moderate positive correlation between TCDC_low.cloud.layer and TCDC_entire.atmosphere (0.65)

Next, the partial autocorrelation of the solar power production data is analyzed. The PACF plot helps identify the extent of correlation between the production at a given hour and its past values after removing the effects of intermediate lags. It helps to identify which lagged values of the target variable (production) have a significant partial autocorrelation with the current value. Significant lags are those where the PACF value is outside the confidence intervals.

```
from statsmodels.graphics.tsaplots import plot_pacf

# Plot PACF for 'production' column
plt.figure(figsize=(10, 6))
plot_pacf(data_grouped['production'], lags=24)  # Plot for 7 days of hourly lags
plt.xlabel('Lag (hours)')
plt.ylabel('Partial Autocorrelation')
plt.title('Partial Autocorrelation Function (PACF) for Production (Hourly Lags)')
plt.show()
```

Figure 6:*Code Snippet for PACF*

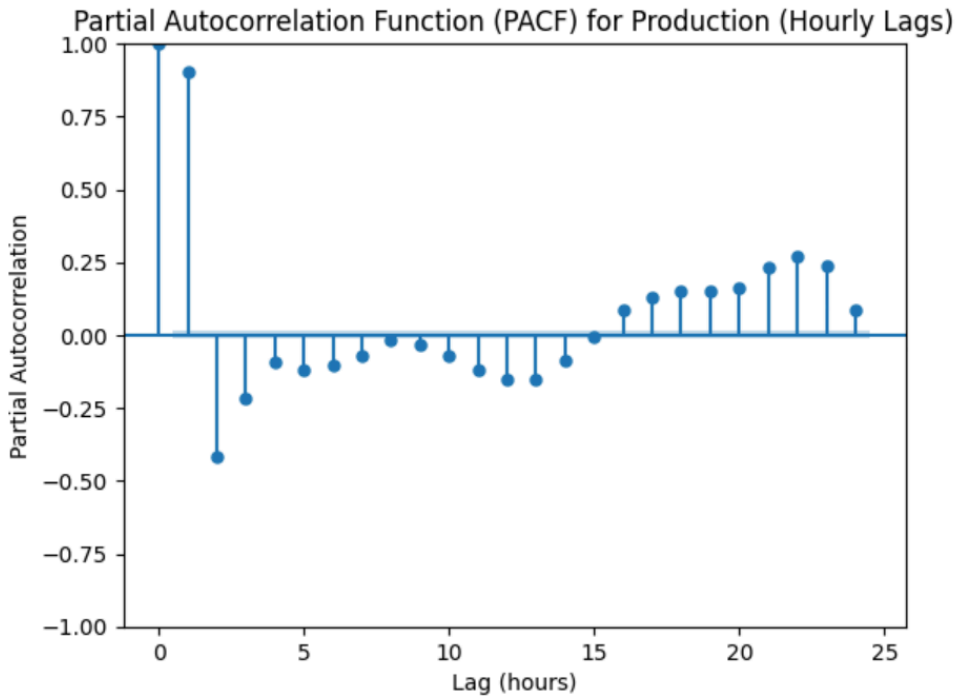


Figure 7: *PACF Plot*

The PACF plot shows a very high partial autocorrelation at lag 1, indicating that the production value at the current hour is strongly influenced by the production value 1 hour ago. After the initial significant lags, the partial autocorrelations at further lags (4 to 10) are not significant, indicating that the direct influence of past values on the current production decreases as the lag increases.

Next, the decomposition of time series data is performed to split the time series into components, each representing an underlying pattern (trends, seasonality, and residuals). Decomposing the time series provides insights into the nature of the data, such as whether production is influenced more by trend or seasonality, and how much variability is due to random factors.

```
import matplotlib.pyplot as plt
from statsmodels.tsa.seasonal import seasonal_decompose

ts_production = pd.Series(data['production'][:400])  # Select only the first 400 data points

# Perform decomposition
decomposition = seasonal_decompose(ts_production, model='additive', period=24)  # Assuming hourly data

# Plot the decomposition
plt.figure(figsize=(12, 8))

# Original time series
plt.subplot(411)
plt.plot(ts_production, label='Original', color='blue')
plt.title('Hourly Production Data (First 400 Points)')
plt.ylabel('Production')
plt.legend()

# Trend component
plt.subplot(412)
plt.plot(decomposition.trend, label='Trend', color='green')
plt.title('Trend')
plt.ylabel('Trend')
plt.legend()

# Seasonal component
plt.subplot(413)
plt.plot(decomposition.seasonal, label='Seasonal', color='red')
plt.title('Seasonal')
plt.ylabel('Seasonal')
plt.legend()

# Residual component
plt.subplot(414)
plt.plot(decomposition.resid, label='Residual', color='purple')
plt.title('Residual')
plt.ylabel('Residual')
plt.legend()

plt.tight_layout()
plt.show()
```

Figure 8: *Code Snippet for Decomposition*

Decomposition graphs are plotted for the first 400 data points only since individual variations and patterns in the trend, seasonal, and residual components are more visible with a smaller subset of data. The decomposition graph breaks down the original time series data into three components: trend, seasonal, and residuals. In this case, the trend starts at a low level, increases to a peak around the midpoint, and then gradually decreases. This indicates that there is a significant underlying trend in production that rises and falls over time. For the seasonal component, there is a clear repetitive pattern within each season of 24 hours. The production amount increases during the daytime and decreases to 0 towards the nighttime. The residual component represents the random fluctuations in the data after removing the trend and seasonality. The residuals do not appear completely random which signifies that trend and seasonality components are not adequately explaining all the systematic variations in the data.

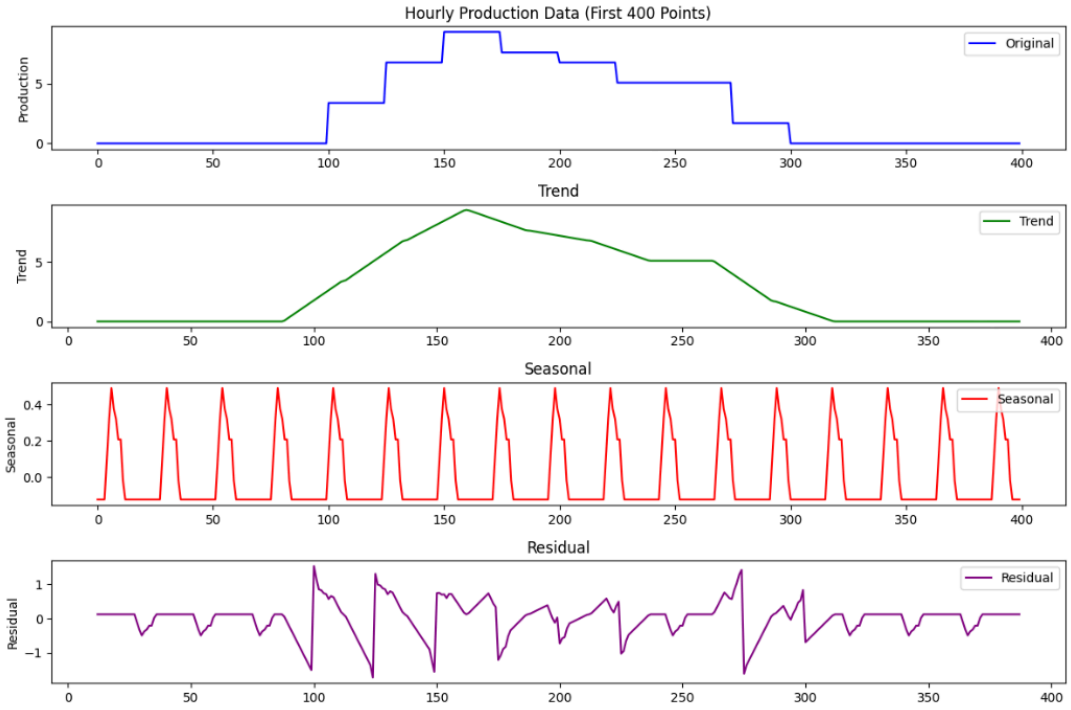


Figure 9: *Decomposition Graphs*

Next, the relationship between various weather variables and solar power production is analyzed. Variables with high positive or negative correlations are likely to be significant predictors in the forecasting model.

```
variable_columns = data_grouped[['DSWRF_surface', 'TCDC_low.cloud.layer', 'TCDC_middle.cloud.layer',
                                 'TCDC_high.cloud.layer', 'TCDC_entire.atmosphere', 'USWRF_top_of_atmosphere',
                                 'CSNOW_surface', 'DLWRF_surface', 'USWRF_surface', 'TMP_surface']]

# Calculate the correlation matrix
correlation_matrix = variable_columns.corrwith(data_grouped['production'])

# Plot the correlation matrix as a heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix.to_frame(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Between Variables and Production')
plt.xlabel('Variables')
plt.ylabel('Production')
plt.show()
```

Figure 10: *Code Snippet for Correlation Matrix for Variables and Production*

Results are visualized in a heatmap:

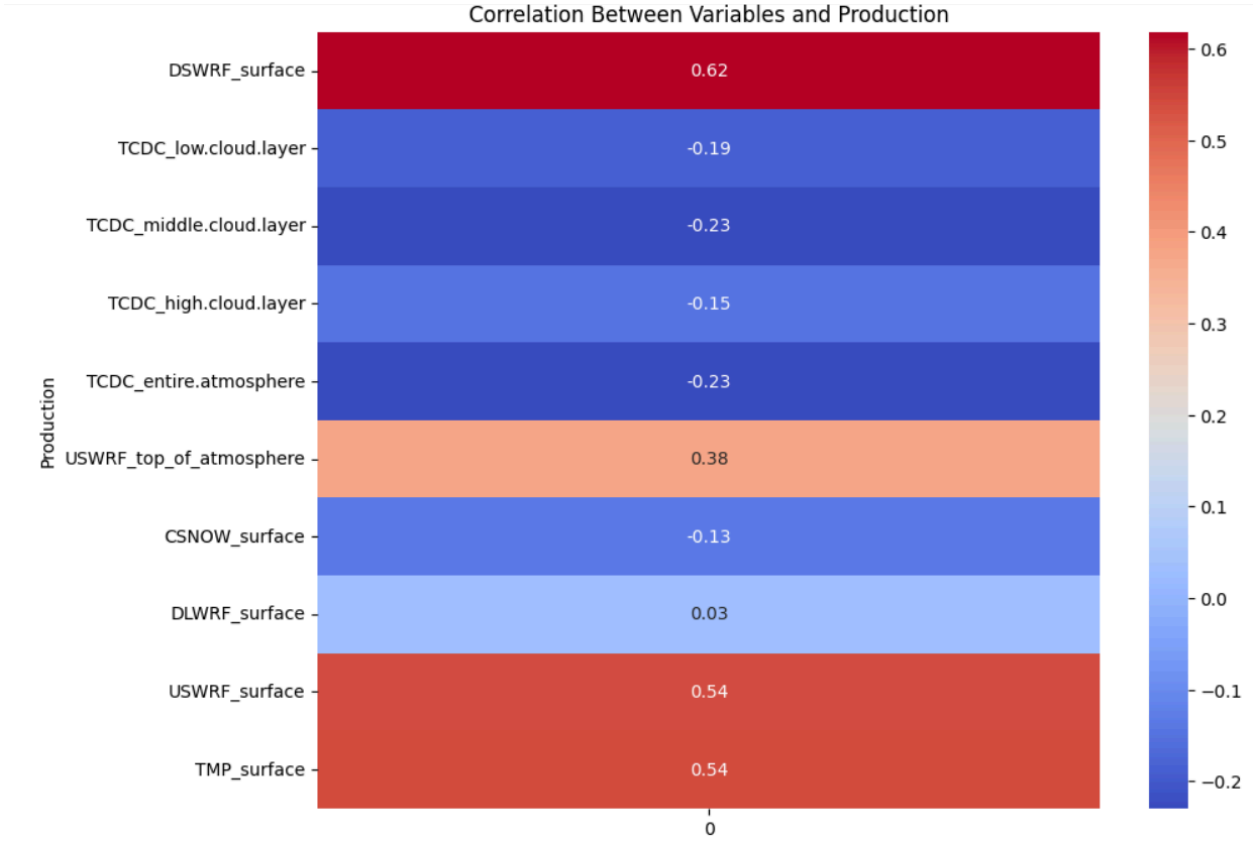


Figure 11: *Correlation Matrix for Variables and Production*

The interpretation of high correlation values are as follows:

- DSWRF_surface (Downward Shortwave Radiation Flux at the Surface) - Correlation: 0.62: Since solar power generation is directly dependent on the amount of sunlight hitting the solar panels, this high correlation is expected. It reinforces the importance of solar radiation in predicting solar power output.
- USWRF_surface (Upward Shortwave Radiation Flux at the Surface) - Correlation: 0.54: It suggests that higher values of reflected solar radiation are associated with higher solar power production, though the relationship is less strong than with downward radiation.
- TMP_surface (Temperature at the Surface) - Correlation: 0.54:Temperature affects the efficiency of solar panels. While higher temperatures can slightly reduce the efficiency, they are often associated with sunny conditions which lead to higher solar irradiance and thus higher overall production.

Due to their high correlation values, the variables DSWRF_surface, USWRF_surface, and TMP_surface will be the first ones incorporated into the models. This will allow us to evaluate their impact on the accuracy of the solar power forecasts.

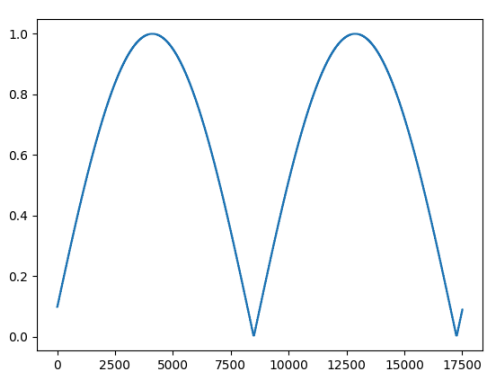
4. Data Modifications

4.1 Seasonality

To account for the seasonality and the trend, a column was added to the data for the hourly seasonality and the yearly seasonality. Looking at the decomposition, half sine (or cosine) for yearly and for time of day in the morning was used. These cosine functions peak at 11 for hourly and for 21 June for yearly.

```
weather["hourly_seasonality"] = np.cos(np.pi*(weather['datetime'].dt.hour - 11)/12)
weather['hourly_seasonality'] = np.where(weather['hourly_seasonality'] < 0, 0, weather['hourly_seasonality'])
weather["yearly_seasonality"] = abs(np.cos(np.pi*(weather['datetime'].dt.dayofyear - 172)/365))
```

Figure 12:*Code Snippet for Hourly Seasonality*



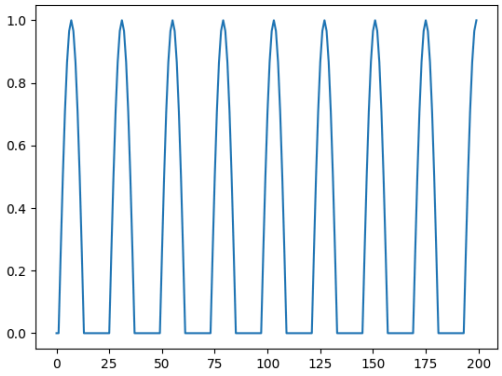


Figure 13&14: *Cosine Functions*

4.2 Feature Selections

As mentioned previously, the rows in the dataset include the hourly data for 25 locations around the powerplant. Therefore, some different approaches can be taken when choosing the input data. In this report, 3 ideas were tried.

Firstly, the average data for all 25 locations were used. For each hour and feature, all the values from all available locations corresponding to that hour and feature were added and divided by the

number of available locations. This way, a straightforward mean for each hour and feature was found.

```
data = pd.merge(df_weather, df_production, on=['date', 'hour'])
data = data.sort_values(by=['date', 'hour'], ascending=[True, True])
data = data.reset_index(drop=True)
data_grouped = data.groupby(['date', 'hour']).mean()
data_grouped = data_grouped.drop(columns=['lat', 'lon'])
data_grouped = data_grouped.reset_index()
y = data_grouped["production"]
X = data_grouped.drop(columns=["date", "hour", "production"])
```

Figure 15: *Code Snippet of Mean Calculations*

Secondly, only the closest location was taken into consideration. A distance metric was calculated for each observation and for every hour and feature, the value corresponding to the lowest distance was chosen.

```
weather["distance"] = ((38.29 - weather["lat"])**2 + (34.97 - weather["lon"])**2)
min_indices = weather.groupby("datetime")["distance"].idxmin()
weather = weather.loc[min_indices]
weather.reset_index(drop=True, inplace=True)
```

Figure 16: *Code Snippet of Distance Matrix*

Finally, the data was converted into a wide format to use every datapoint. This means that every feature was split into 25, for each location. For example, DSWRF_surface was replaced with DSWRF_surface_37.75_35.5, DSWRF_surface_38.5_34.5, DSWRF_surface_38.25_35.25 etc.

```
weather = weather.pivot_table(index="datetime", columns=["lat", "lon"])
new_columns = [f'{col[0]}_{col[1]}_{col[2]}' for col in weather.columns]
weather.columns = new_columns
weather.reset_index(inplace=True)
```

Figure 17: *Code Snippet of Wide Format Conversion*

5. Models Used

5.1 Linear Model

Linear Models are models where the results are predicted with a linear function on the features presented. The error terms are assumed to be iid.

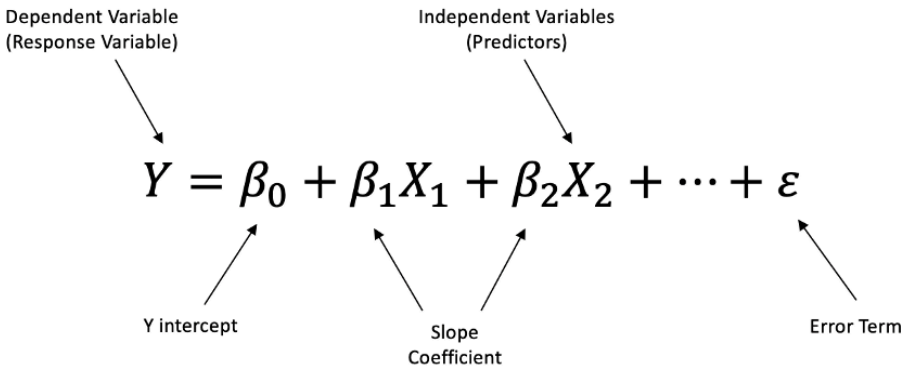


Figure 18: *Linear Model Equation*

Solving linear models is relatively easy. The code for the model, fitting and outputs are shown below.

```
y_train = y.iloc[:].copy()
y_test = y.iloc[:].copy()
X_train = X.iloc[:].copy()
X_test = X.iloc[:].copy()
X_train = X_train.fillna(X_train.mean())
y_train = y_train.fillna(y_train.mean())
X_test = X_test.fillna(X_train.mean())
y_test = y_test.fillna(y_train.mean())

# Linear model
model = sm.OLS(y_train, X_train)
results = model.fit()
print(results.summary())

# Forecast of the model
forecast = results.predict(X_test)
forecast = np.maximum(forecast, 0)
model_residuals = y_test.values - forecast.values

"""plt.plot(model_residuals, label='Residual Analysis')
plt.legend()
plt.show()"""

# Calculate Weighted MAPE
wmape = sum(abs(forecast-y_test))/sum(y_test)
print("Weighted MAPE:", wmape)
```

Figure 19: *Code Snippets for Linear Regression*

5.2 ARIMA with Exogenous Variable

ARIMA stands for Autoregressive Integrated Moving Average. It is again a regression model but the variables that the result is based on are not features. Autoregressive means that the independent variables used are the previous values of the result statistic or in other words lagged values, Moving Average means that the errors of the previous predictions are also used, and Integrated means that the data fed is differenced before putting it in a regression. Using exogenous variables simply adds those variables to the independent variables list.

$$y_t = \mu + \sum_{i=1}^{p} \phi_i y_{t-i} + \epsilon_t - \sum_{i=1}^{q} \theta_q \epsilon_{t-i} + \beta x_t$$

Figure 21: *ARIMA Equation*

```
y_train = y.iloc[:].copy()
y_test = y.iloc[:].copy()
X_train = X.iloc[:].copy()
X_test = X.iloc[:].copy()
X_train = X_train.fillna(X_train.mean())
y_train = y_train.fillna(y_train.mean())
X_test = X_test.fillna(X_train.mean())
y_test = y_test.fillna(y_train.mean())

# ARIMAX model
y_train.index = pd.Index(range(len(y_train)))
X_train.index = pd.Index(range(len(X_train)))
model = ARIMA(y_train, order=(2,1,2),exog=X_train)
results = model.fit()
print(results.summary())

# Forecast of the model
forecast = results.predict(start = 0, end=len(y_train) - 1)
forecast = np.maximum(forecast, 0)
model_residuals = y_test.values - forecast.values

"""plt.plot(model_residuals, label='Residual Analysis')
plt.legend()
plt.show()"""

# Calculate MAPE
wmape = sum(abs(forecast-y_test))/sum(y_test)
print("Weighted MAPE:", wmape)
```

Figure 19: *Code Snippets for ARIMA*

Thus, ARIMA requires 3 parameters; one for how many lagged values to add, one for how many times to differentiate and one for how many previous error terms to add. They are, in order, p, d and q. To choose the best p d q values, iterating through possible values is the common practice. Since there are many different ways that the data will be used and using exogenous variables makes the algorithm very slow, choosing the (p,d,q) values without exogenous variables and then using those for each was thought to be the best way to go.

Trying [0, 1, 2] for each of p, d and q and checking the AIC and BIC values, the best one can be chosen.

```
0 0 0  :  109825 109841
0 0 1  :  91223 91246
0 0 2  :  81144 81176
0 1 0  :  76670 76678
0 1 1  :  74691 74707
0 1 2  :  73911 73935
0 2 0  :  82037 82045
0 2 1  :  77322 77338
0 2 2  :  74699 74723
1 0 0  :  75685 75708
1 0 1  :  73255 73287
1 0 2  :  72018 72058
1 1 0  :  74087 74103
1 1 1  :  73971 73994
1 1 2  :  73851 73882
1 2 0  :  78746 78762
1 2 1  :  74096 74119
1 2 2  :  73979 74011
2 0 0  :  71876 71908
2 0 1  :  70057 70097
2 0 2  :  69933 69981
2 1 0  :  73918 73942
2 1 1  :  73875 73907
2 1 2  :  69322 69362
2 2 0  :  78191 78215
2 2 1  :  73926 73958
2 2 2  :  74021 74061
```

Figure 22: *p, d, q, AIC and BIC values*

The lowest values for AIC and BIC are for (p ,d ,q) = (2, 1, 2). These values will be used for ARIMA Models.

6. Modeling with Averaged Data

6.1 Linear Regression Models

The purpose of the first linear regression model is to forecast the production variable using its lag value . Production is the dependent variable and lag_production is the independent variable in the model, along with a constant term. Based on the output summary, the model's R-squared score of 0.816 suggests that it is very significant, explaining about 81.6% of the production variability. With a p-value close to 0, the lagged_production coefficient of 0.9035 indicates a statistically significant positive correlation between the current and historical production numbers. With a p-value close to 0, the intercept term, which is 0.2484, is likewise statistically significant. Overall, the model offers a strong fit, as seen by the high F-statistic and low p-values for the coefficients.

```
import statsmodels.api as sm

data_grouped['lagged_production'] = data_grouped['production'].shift(1)

data_grouped.dropna(inplace=True)

X = data_grouped[['lagged_production']]
y = data_grouped['production']

X = sm.add_constant(X)

model = sm.OLS(y, X)

results = model.fit()
print(results.summary())
```

Figure 23: *The code snippet of the regression model with 1-lagged values*

As can be seen in *Figure 24*, additional model statistics that can be used to verify residual normality and autocorrelation are the Omnibus value of 2921.179 and the Durbin-Watson statistic of 1.247. The residuals show a deviation from normalcy according to both the Jarque-Bera (JB) test and the Prob(Omnibus) test. The residuals' distribution can be better understood by looking at the values of Skew and Kurtosis.

```
                            OLS Regression Results
==============================================================================
Dep. Variable:             production   R-squared:                       0.816
Model:                            OLS   Adj. R-squared:                  0.816
Method:                 Least Squares   F-statistic:                 8.909e+04
Date:                Sat, 11 May 2024   Prob (F-statistic):               0.00
Time:                        14:59:33   Log-Likelihood:                -37755.
No. Observations:               20075   AIC:                         7.551e+04
Df Residuals:                   20073   BIC:                         7.553e+04
Df Model:                           1
Covariance Type:            nonrobust
=====================================================================================
                        coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------------
const                 0.2484      0.014     18.212      0.000       0.222       0.275
lagged_production     0.9035      0.003    298.485      0.000       0.898       0.909
==============================================================================
Omnibus:                     2921.179   Durbin-Watson:                   1.247
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            16376.424
Skew:                           0.581   Prob(JB):                         0.00
Kurtosis:                       7.270   Cond. No.                         5.58
==============================================================================
```

Figure 24: *The output of the regression model with 1-lagged values*

Secondly, a lagged value from 24 periods prior is used to do a linear regression analysis. Missing values are removed after importing the required library and preparing the data by establishing a lagged_production column (shifted by 24 periods). The Ordinary Least Squares (OLS) approach is used to fit the regression in the model, which also contains a constant term.

```python
import statsmodels.api as sm

data_grouped['lagged_production'] = data_grouped['production'].shift(24)

data_grouped.dropna(inplace=True)

X = data_grouped[['lagged_production']]
y = data_grouped['production']

X = sm.add_constant(X)

model = sm.OLS(y, X)

results = model.fit()
print(results.summary())
```

Figure 25: *The code snippet of the regression model with 24-lagged values*

According to the regression analysis, the model (R-squared = 0.763) accounts for 76.3% of the production variability. The model is highly significant, as indicated by the p-value of 0.00 and the F-statistic of 6.440e+04. With low p-values, the intercept coefficient is 0.3268 and the lagged_production coefficient is 0.8737, both of which are statistically significant. Other statistics that point to possible autocorrelation in residuals and some deviations from normalcy include the Jarque-Bera test, the Durbin-Watson statistic, and the Omnibus value.

```
                            OLS Regression Results
==============================================================================
Dep. Variable:             production   R-squared:                       0.763
Model:                            OLS   Adj. R-squared:                  0.763
Method:                 Least Squares   F-statistic:                 6.440e+04
Date:                Sat, 11 May 2024   Prob (F-statistic):               0.00
Time:                        14:59:44   Log-Likelihood:                -40275.
No. Observations:               20051   AIC:                          8.055e+04
Df Residuals:                   20049   BIC:                          8.057e+04
Df Model:                           1
Covariance Type:            nonrobust
=======================================================================================
                          coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------
const                   0.3268      0.016     21.072      0.000       0.296       0.357
lagged_production       0.8737      0.003    253.780      0.000       0.867       0.880
==============================================================================
Omnibus:                     3807.273   Durbin-Watson:                   0.607
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            39656.869
Skew:                           0.610   Prob(JB):                         0.00
Kurtosis:                       9.781   Cond. No.                         5.58
==============================================================================
```

Figure 26: *The output of the regression model with 24-lagged values*

Subsequently, DSWRF_surface, USWRF_surface, and TMP_surface have been used as predictors. The Ordinary Least Squares (OLS) approach is used to fit the model once the data have been prepared and a constant term has been added.

```python
X = data_grouped[['DSWRF_surface', 'USWRF_surface', 'TMP_surface']]
X = sm.add_constant(X)

y = data_grouped['production']

model = sm.OLS(y, X)

results = model.fit()
print(results.summary())
```

Figure 27: *Code snippet of the regression model with 3 predictors*

According to the regression results in *Figure 28*, 41.2% of the variability in production can be explained by the model (R-squared = 0.412) which is considered low. With a p-value of 0.00(rounded) and an F-statistic of 4684, the model is highly significant. With p-values of 0.000 (rounded), the coefficients for TMP_surface (0.0865), USWRF_surface (0.0124), and DSWRF_surface (0.0032) are all statistically significant. With a p-value of 0.000, the intercept, which is -23.2999, is similarly significant.

```
                            OLS Regression Results
==============================================================================
Dep. Variable:             production   R-squared:                       0.412
Model:                            OLS   Adj. R-squared:                  0.412
Method:                 Least Squares   F-statistic:                     4684.
Date:                Sat, 11 May 2024   Prob (F-statistic):               0.00
Time:                        15:00:06   Log-Likelihood:                -49367.
No. Observations:               20051   AIC:                         9.874e+04
Df Residuals:                   20047   BIC:                         9.877e+04
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        -23.2999      0.752    -30.995      0.000     -24.773     -21.826
DSWRF_surface     0.0032      0.000     14.284      0.000       0.003       0.004
USWRF_surface     0.0124      0.001     16.975      0.000       0.011       0.014
TMP_surface       0.0865      0.003     31.994      0.000       0.081       0.092
==============================================================================
Omnibus:                     2068.093   Durbin-Watson:                   0.311
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             2788.845
Skew:                           0.855   Prob(JB):                         0.00
Kurtosis:                       3.641   Cond. No.                     1.55e+04
==============================================================================
```

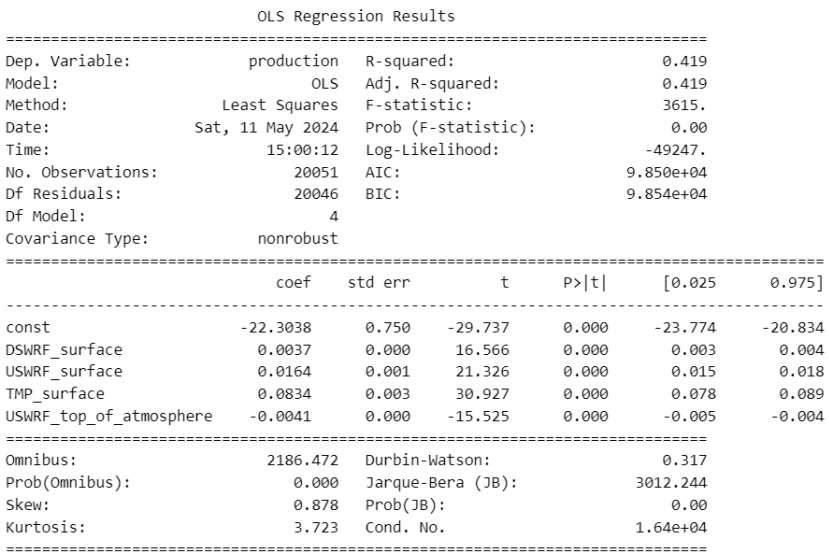
Figure 28: *The output of the regression model with 3 predictors*

This time, USWRF_top_of_atmosphere is added to the model.

```python
import statsmodels.api as sm

X = data_grouped[['DSWRF_surface', 'USWRF_surface', 'TMP_surface', 'USWRF_top_of_atmosphere']]
X = sm.add_constant(X)

y = data_grouped['production']

model = sm.OLS(y, X)

results = model.fit()
print(results.summary())
```

Figure 29: *Code snippet of the regression model with 4 predictors (USWRF_top_of_atmosphere added)*

The model explains only 41.9% of the variability in production (R-squared = 0.419) and is significant (F-statistic = 3615, p-value = 0.00). The coefficients for DSWRF_surface (0.0037), USWRF_surface (0.0164), TMP_surface (0.0834), and USWRF_top_of_atmosphere (-0.0041) are all statistically significant with p-values of 0.000 (rounded), indicating that surface radiation and temperature variables significantly impact production. The intercept is -22.3038, also statistically significant.



OLS Regression Results

Dep. Variable:	production	R-squared:	0.419
Model:	OLS	Adj. R-squared:	0.419
Method:	Least Squares	F-statistic:	3615.
Date:	Sat, 11 May 2024	Prob (F-statistic):	0.00
Time:	15:00:12	Log-Likelihood:	-49247.
No. Observations:	20051	AIC:	9.850e+04
Df Residuals:	20046	BIC:	9.854e+04
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-22.3038	0.750	-29.737	0.000	-23.774	-20.834
DSWRF_surface	0.0037	0.000	16.566	0.000	0.003	0.004
USWRF_surface	0.0164	0.001	21.326	0.000	0.015	0.018
TMP_surface	0.0834	0.003	30.927	0.000	0.078	0.089
USWRF_top_of_atmosphere	-0.0041	0.000	-15.525	0.000	-0.005	-0.004

Omnibus:	2186.472	Durbin-Watson:	0.317
Prob(Omnibus):	0.000	Jarque-Bera (JB):	3012.244
Skew:	0.878	Prob(JB):	0.00
Kurtosis:	3.723	Cond. No.	1.64e+04

Figure 30: *The output of the regression model with 4 predictors (USWRF_top_of_atmosphere added)*

6.2 ARIMA Models

For the purpose of further research, an ARIMA model with the configuration (2, 1, 2) was developed using average values. However, this model was not selected for the final solution. The corresponding code and output for this model are illustrated in *Figure 31*. The model includes two autoregressive terms, one differencing operation, and two moving average terms.

```
data_grouped = data.groupby(['date', 'hour']).mean()
data_grouped = data_grouped.drop(columns=['lat', 'lon'])
data_grouped = data_grouped.reset_index()

y = data_grouped["production"]
X = data_grouped.drop(columns=["date", "hour", "production"])

# ARIMA modeling
y_train = y.iloc[:].copy()
y_test = y.iloc[:].copy()
X_train = X.iloc[:].copy()
X_test = X.iloc[:].copy()
X_train = X_train.fillna(X_train.mean())
y_train = y_train.fillna(y_train.mean())
X_test = X_test.fillna(X_train.mean())
y_test = y_test.fillna(y_train.mean())

y_train.index = pd.Index(range(len(y_train)))
X_train.index = pd.Index(range(len(X_train)))
model = ARIMA(y_train, order=(2,1,2),exog=X_train)
results = model.fit()
print(results.summary())

# Forecast
forecast = results.predict(start = 0, end=len(y_train) - 1)
forecast = np.maximum(forecast, 0)
model_residuals = y_test.values - forecast.values

"""plt.plot(model_residuals, label='Residual Analysis')
plt.legend()
plt.show()"""

# WMAPE
wmape = sum(abs(forecast-y_test))/sum(y_test)
print("Weighted MAPE:", wmape)

plt.plot(y.index[-300:-1], y[-300:-1], label='Actual')
plt.plot(y.index[-300:-1], forecast[-300:-1], label='Forecast')
plt.legend()
plt.show()
```

Figure 31: *The code for the ARIMA model for average values*

The output of the ARIMA code is shown below in Figure 32.

```
                               SARIMAX Results
==============================================================================
Dep. Variable:             production   No. Observations:                20084
Model:                 ARIMA(2, 1, 2)   Log Likelihood              -34726.927
Date:                Thu, 09 May 2024   AIC                          69483.853
Time:                        15:27:53   BIC                          69602.467
Sample:                             0   HQIC                         69522.652
                                - 20084
Covariance Type:                  opg
==============================================================================
                    coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
DSWRF_surface     -0.0010      0.000     -4.791      0.000      -0.001      -0.001
TCDC_low.cloud.layer  0.0308      0.002     14.902      0.000       0.027       0.035
TCDC_middle.cloud.layer  0.0106      0.001      8.229      0.000       0.008       0.013
TCDC_high.cloud.layer  -0.0067      0.001     -4.902      0.000      -0.009      -0.004
TCDC_entire.atmosphere  0.0158      0.002      8.193      0.000       0.012       0.020
USWRF_top_of_atmosphere  0.0032      0.000     10.565      0.000       0.003       0.004
CSNOW_surface     0.9334      0.225      4.151      0.000       0.493       1.374
DLWRF_surface     -0.0938      0.002    -42.914      0.000      -0.098      -0.090
USWRF_surface     -0.0040      0.001     -4.613      0.000      -0.006      -0.002
TMP_surface       0.4537      0.005     91.124      0.000       0.444       0.463
ar.L1             0.8318      0.025     32.719      0.000       0.782       0.882
ar.L2            -0.2046      0.023     -8.796      0.000      -0.250      -0.159
ma.L1            -0.8165      0.025    -32.766      0.000      -0.865      -0.768
ma.L2            -0.0588      0.025     -2.356      0.018      -0.108      -0.010
sigma2            1.8507      0.011    169.786      0.000       1.829       1.872
===================================================================================
Ljung-Box (L1) (Q):                   3.26   Jarque-Bera (JB):             32740.10
Prob(Q):                              0.07   Prob(JB):                         0.00
Heteroskedasticity (H):               0.74   Skew:                             0.17
Prob(H) (two-sided):                  0.00   Kurtosis:                         9.25
===================================================================================
```

Figure 32: *The output of the ARIMA model for average values*

The model has a reasonable Weighted MAPE 28.65%. lower WMAPE values indicate better model performance.

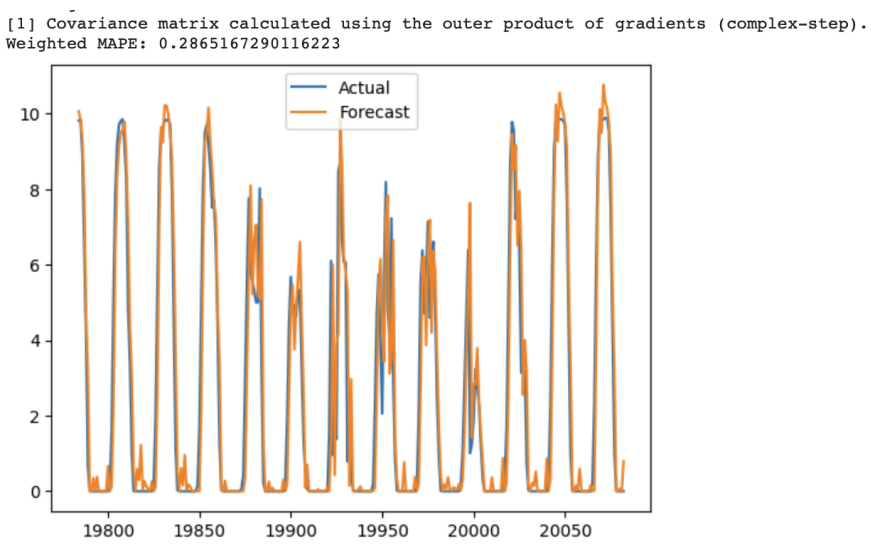


Figure 33: *Actual vs Forecasted Values of the ARIMA (2,1,2) Model*

7. Modeling with Closest Data

7.1 Linear Regression Models

Using every feature in the data found by choosing the closest values for each feature, a linear fit was computed. The summary of this fit can be found below.

Dep. Variable:	production	R-squared (uncentered):	0.847
Model:	OLS	Adj. R-squared (uncentered):	0.847
Method:	Least Squares	F-statistic:	9300.
Date:	Wed, 05 Jun 2024	Prob (F-statistic):	0.00
Time:	12:50:11	Log-Likelihood:	-39959.
No. Observations:	20132	AIC:	7.994e+04
Df Residuals:	20120	BIC:	8.004e+04
Df Model:	12		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
DSWRF_surface	-1.546e-05	0.000	-0.155	0.877	-0.000	0.000
TCDC_low.cloud.layer	-0.0085	0.001	-13.152	0.000	-0.010	-0.007
TCDC_middle.cloud.layer	-0.0044	0.001	-8.683	0.000	-0.005	-0.003
TCDC_high.cloud.layer	-0.0012	0.001	-2.060	0.039	-0.002	-5.88e-05
TCDC_entire.atmosphere	-0.0035	0.001	-4.960	0.000	-0.005	-0.002
USWRF_top_of_atmosphere	-0.0005	0.000	-2.784	0.005	-0.001	-0.000
CSNOW_surface	-0.2543	0.071	-3.606	0.000	-0.392	-0.116
DLWRF_surface	0.0016	0.000	3.674	0.000	0.001	0.003
USWRF_surface	0.0015	0.000	3.691	0.000	0.001	0.002
TMP_surface	-0.0032	0.000	-7.899	0.000	-0.004	-0.002
hourly_seasonality	8.1564	0.032	254.273	0.000	8.094	8.219
yearly_seasonality	1.4219	0.053	26.924	0.000	1.318	1.525

Omnibus:	2142.274	Durbin-Watson:	0.600
Prob(Omnibus):	0.000	Jarque-Bera (JB):	4654.559
Skew:	-0.665	Prob(JB):	0.00
Kurtosis:	4.945	Cond. No.	2.86e+03

Figure 34: *Output of the linear fit with the closest values for each feature*

The Weighted MAPE value of the model is 43.00%.

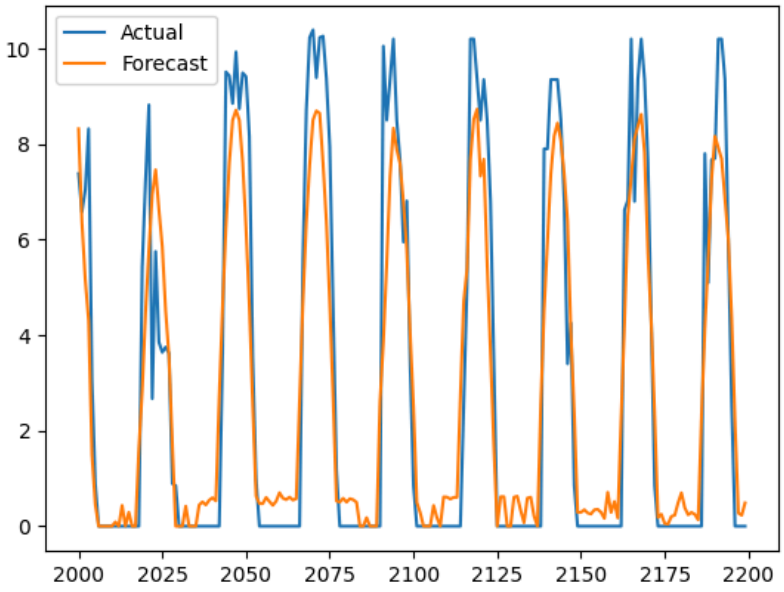


Figure 35: *Actual vs Forecasted Values Plotted*

7.2 ARIMA Model

Using every feature in the data found by choosing the closest values for each feature, an ARIMA Model with parameters (2,1,2) was fitted. The summary of this fit can be found below.

```
==============================================================================
Dep. Variable:             production   No. Observations:                20132
Model:                 ARIMA(2, 1, 2)   Log Likelihood              -33042.049
Date:                Wed, 05 Jun 2024   AIC                          66118.098
Time:                        12:51:44   BIC                          66252.568
Sample:                             0   HQIC                         66162.078
                                - 20132
Covariance Type:                  opg
==============================================================================
                             coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
DSWRF_surface          -3.234e-05   8.99e-05     -0.360      0.719      -0.000       0.000
TCDC_low.cloud.layer   -8.604e-05      0.000     -0.174      0.862      -0.001       0.001
TCDC_middle.cloud.layer   -0.0003      0.000     -0.827      0.408      -0.001       0.000
TCDC_high.cloud.layer     -0.0007      0.000     -1.864      0.062      -0.001    3.66e-05
TCDC_entire.atmosphere     0.0010      0.000      2.091      0.036     6.5e-05       0.002
USWRF_top_of_atmosphere 1.912e-05    9.6e-05      0.199      0.842      -0.000       0.000
CSNOW_surface              0.0262      0.050      0.526      0.599      -0.071       0.124
DLWRF_surface             -0.0007      0.001     -1.118      0.264      -0.002       0.001
USWRF_surface              0.0003      0.000      1.215      0.224      -0.000       0.001
TMP_surface               -0.0012      0.002     -0.549      0.583      -0.005       0.003
hourly_seasonality         7.5788      0.051    148.017      0.000       7.478       7.679
yearly_seasonality        -0.1585     33.246     -0.005      0.996     -65.320      65.003
ar.L1                      0.5565      0.108      5.155      0.000       0.345       0.768
ar.L2                      0.1163      0.092      1.267      0.205      -0.064       0.296
ma.L1                     -0.6377      0.108     -5.918      0.000      -0.849      -0.426
ma.L2                     -0.1965      0.102     -1.931      0.053      -0.396       0.003
sigma2                     1.6061      0.011    144.118      0.000       1.584       1.628
===================================================================================
Ljung-Box (L1) (Q):                  18.70   Jarque-Bera (JB):             31965.16
Prob(Q):                              0.00   Prob(JB):                         0.00
Heteroskedasticity (H):               0.55   Skew:                             0.13
Prob(H) (two-sided):                  0.00   Kurtosis:                         9.17
===================================================================================
```

Figure 35: *Output of the ARIMA(2,1,2) Model*

The Weighted MAPE value of the model is 57.76%.

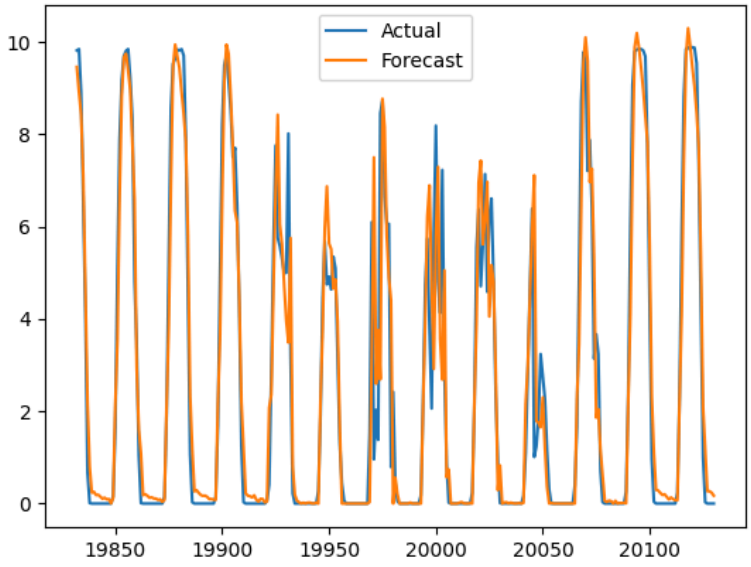


Figure 36: *Actual vs Forecasted Values of the ARIMA Model*

8. Modeling with Wide Data

8.1 Linear Model

Using every feature in the wide data, a linear fit was computed. The summary of this fit can be found below.

Dep. Variable:	production	R-squared (uncentered):	0.857
Model:	OLS	Adj. R-squared (uncentered):	0.855
Method:	Least Squares	F-statistic:	473.8
Date:	Wed, 05 Jun 2024	Prob (F-statistic):	0.00
Time:	12:54:51	Log-Likelihood:	-39276.
No. Observations:	20132	AIC:	7.906e+04
Df Residuals:	19880	BIC:	8.105e+04
Df Model:	252		
Covariance Type:	nonrobust		

Omnibus:	1773.534	Durbin-Watson:	0.616
Prob(Omnibus):	0.000	Jarque-Bera (JB):	3598.232
Skew:	-0.586	Prob(JB):	0.00
Kurtosis:	4.708	Cond. No.	2.72e+04

Figure 37&38: *Output of the linear model with wide data*

The Weighted MAPE value of the model is 41.49%.

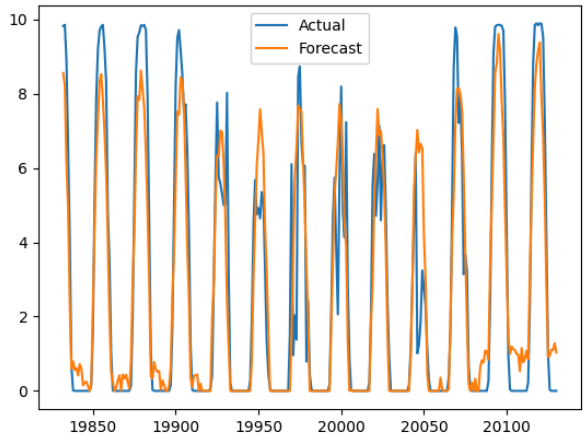


Figure 39: *Actual vs Forecasted Values of the model with wide data*

8.2 ARIMA Model

Using every feature in the wide data, an ARIMA Model with parameters (2,1,2) was fitted. The summary of this fit can be found below.

Dep. Variable:	production	No. Observations:	20132
Model:	ARIMA(2, 1, 2)	Log Likelihood	-32933.546
Date:	Wed, 05 Jun 2024	AIC	66381.093
Time:	13:04:52	BIC	68413.967
Sample:	0	HQIC	67045.976
	- 20132		
Covariance Type:	opg		

Ljung-Box (L1) (Q):	16.53	Jarque-Bera (JB):	30004.45
Prob(Q):	0.00	Prob(JB):	0.00
Heteroskedasticity (H):	0.56	Skew:	0.13
Prob(H) (two-sided):	0.00	Kurtosis:	8.98

Figure 40&41: *Output of the wide data with an ARIMA(2,1,2) Model*

The Weighted MAPE value of the model is 26.90%.

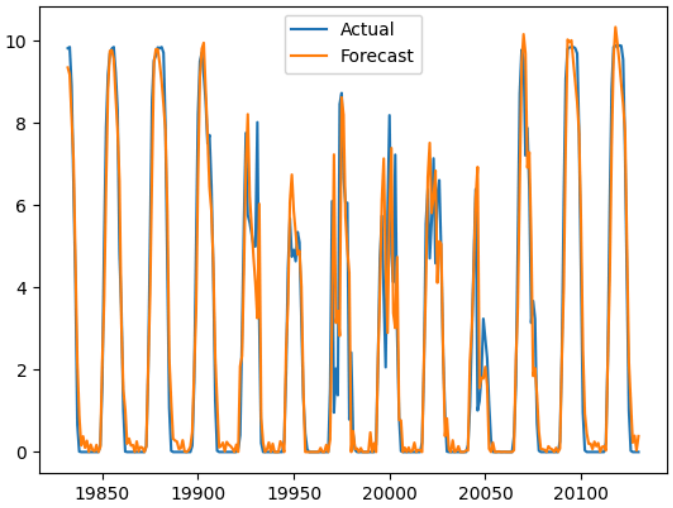


Figure 42: *Actual vs Forecasted Values of the wide data with an ARIMA(2,1,2) Model*

9. Different Features with the Best Model

Looking at the AIC, BIC and Weighted MAPE values, ARIMAX(2,1,2) has the best results. However, this model uses every feature. In the data analysis part, the features DSWRF, USWRF_surface and TMP_surface were found to be correlated with the production values. Using only them, with the ARIMA(2,1,2) model, the resulting summary is shown in *Figures 43 and 44.*

Dep. Variable:	production	No. Observations:	20132
Model:	ARIMA(2, 1, 2)	Log Likelihood	-33024.158
Date:	Wed, 05 Jun 2024	AIC	66212.315
Time:	15:35:43	BIC	66860.937
Sample:	0	HQIC	66424.457
	- 20132		
Covariance Type:	opg		

Ljung-Box (L1) (Q):	17.86	Jarque-Bera (JB):	31731.88
Prob(Q):	0.00	Prob(JB):	0.00
Heteroskedasticity (H):	0.56	Skew:	0.14
Prob(H) (two-sided):	0.00	Kurtosis:	9.14

Figure 43&44: *Output of the Updated Model*

The Weighted MAPE for this Model is 26.72%.

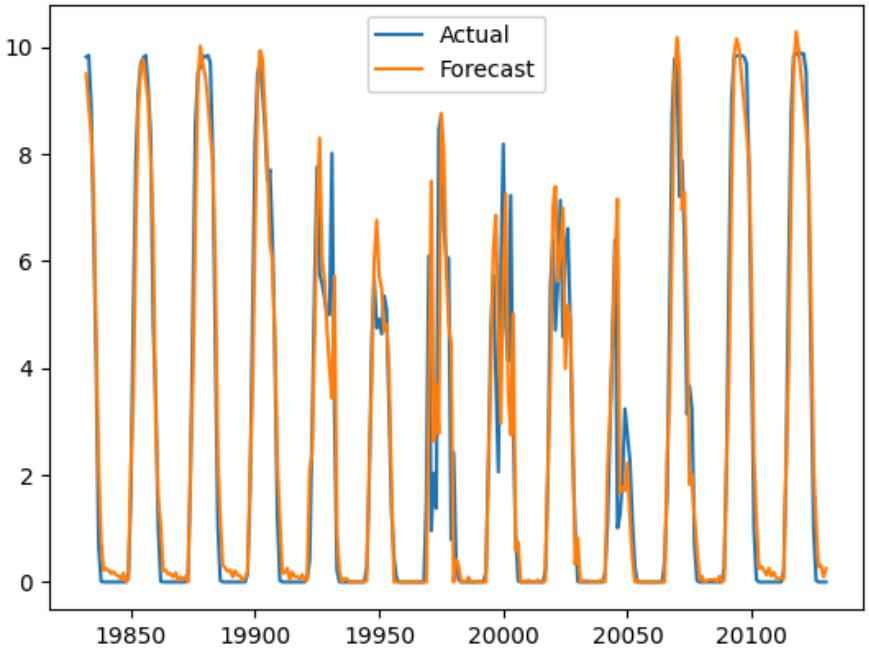


Figure 45: *Actual vs Forecasted Values of the Updated Model*

10. Conclusion

In conclusion, our forecasting project systematically evaluated multiple linear and ARIMA models to predict production accurately. Initially, three linear models were tested: the wide format, incorporating all measurements from all coordinates and time points; the average format, which averages measurements from different coordinates for the same date and hour; and the closest format, utilizing measurements from the nearest location to the power plant. The performance of these models laid the foundation for our advanced analysis.

Subsequently, we explored ARIMA models to capture the temporal dynamics of the data. By determining the optimal parameters p,d,q to be 2, 1, and 2, respectively, we applied the ARIMA(2,1,2) model across the wide, closest, and average formats. Among these, the ARIMA model applied to the wide format yielded the best results, indicating the advantage of using comprehensive data.

To further refine our forecasts, we integrated the three features most highly correlated with the production data into the ARIMA model using the wide format. This approach significantly improved the forecasting accuracy, demonstrating the effectiveness of combining feature selection with a robust time series model. Overall, the ARIMA(2,1,2) model with the wide format and selected features provided the most accurate and reliable forecasts, highlighting the importance of both comprehensive data utilization and targeted feature selection in production forecasting.

11. References

1. https://otexts.com/fpp2/
2. https://www.sfu.ca/~dsignori/buec333/lecture%2016.pdf
3. https://www.econstor.eu/bitstream/10419/49919/1/668828234.pdf
4. S. Atique, S. Noureen, V. Roy, V. Subburaj, S. Bayne and J. Macfie, "Forecasting of total daily solar energy generation using ARIMA: A case study," 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 2019, pp. 0114-0119, doi: 10.1109/CCWC.2019.8666481.
5. Fattah, Jamal, Latifa Ezzine, Zineb Aman, Haj El Moussami, and Abdeslam Lachhab. "Forecasting of demand using ARIMA model." International Journal of Engineering Business Management 10 (January 1, 2018): 184797901880867. https://doi.org/10.1177/1847979018808673.

12. Appendix

Github Link for all the codes used:
https://github.com/BU-IE-360/spring24-EmreCaganKanli/blob/main/360_project_codes.ipynb