Used libraries:

- numpy
- pandas
- matplotlib
- datetime
- scipy
- statsmodels
- arch
- Scikit-learn

This report encapsulates the ten tasks assigned in the second data analytics assignment, outlining the procedure, the achieved results, and my personal thoughts on each task.

Task1:

The first task was to import the data into my computer and create a visual representation of wind generation over time, as well as to determine whether there is evidence of annual seasonality and this was accomplished through the implementation of the following steps:

- I imported the given CSV file into my Jupyter workspace as a dataframe and verified the presence of any missing data.
- To effectively address the issue of missing data, I used linear interpolation and created a timestamp by combining the given date and time columns.
- I created a time series plot to visualize wind generation over time, as well as plots depicting intra-annual seasonality at daily, weekly, monthly, and quarterly intervals.

Results:

The task was successfully executed, as demonstrated in Figures 1 and 2. Figure 1 illustrates the hourly wind energy generation from January to December 2014. Figure 2 presents the variation of the generated electricity in relation to daily, weekly, monthly, and quarterly seasonality patterns. As shown in figure 2 there is the daily and weekly intra seasonality.

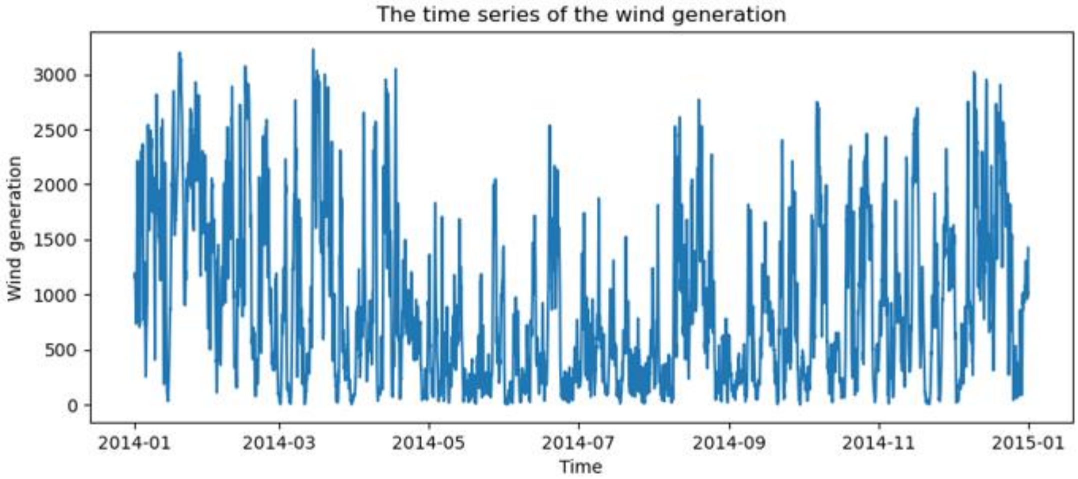


Figure 1: The time series of the generated electricity from a wind energy at each hour in 2014

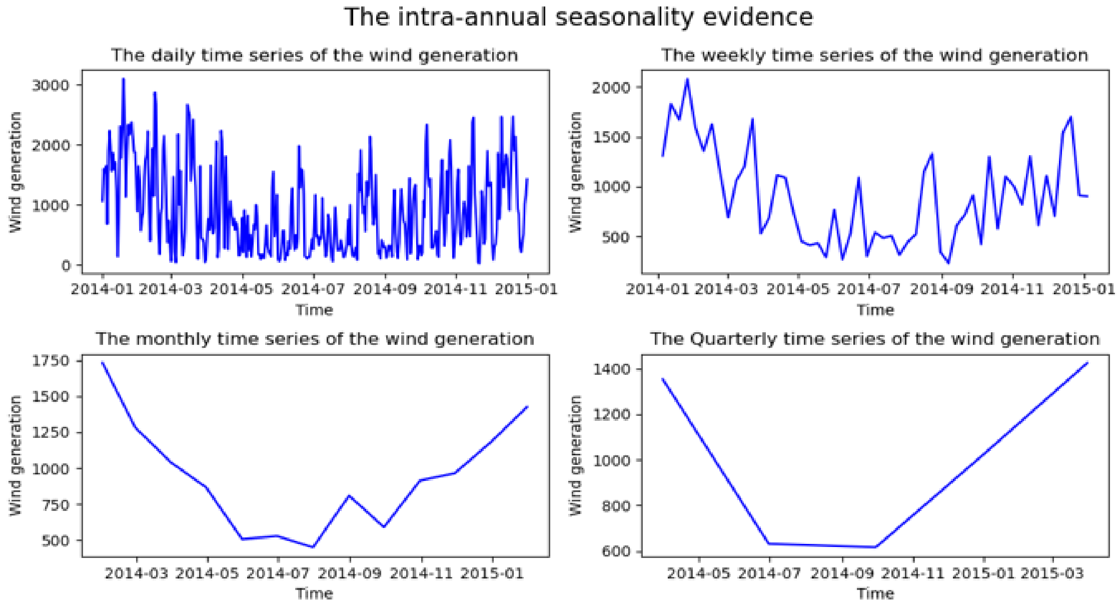


Figure 2: The daily, quarterly, weekly, and monthly intra seasonality evidence

Insight: Wind generation was highest from October to April and lowest from May to September, according to the findings. This implies a higher generation rate in the winter and a lower rate in the summer. The reduced demand for energy during the summer, when people do not need heating for their homes, could, in my opinion, be a contributing factor to the lower generation rate.

<u>Task 2:</u>

The objective of the second task was to create a plot of the evolution of wind generation over time as a percentage of the maximum generation, and to determine the presence of annual seasonality. To achieve this, the following steps were carried out:

- ❖ I transformed the wind generation values into an array and assigned those value to a declared variable
- ❖ I created a loop to calculate the change in wind generation as a percentage of the maximum generation and assigned the results to a declared variable
- ❖ I inserted the obtained change in wind generation into the data frame as a column and plotted the results with respect to the time

Results: X(t) - X(t-1)*100 /max_gen was the formula that was used to obtain the change in generation that was depicted on the figure 3 below with respect to time. As it is shown on the intra seasonality evidence figure, there is no evidence seasonality.

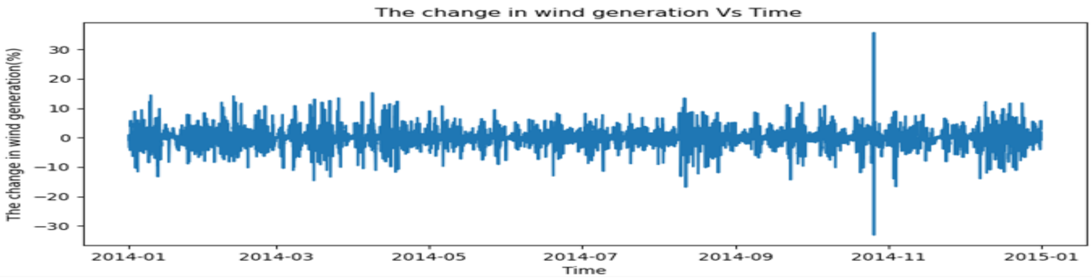
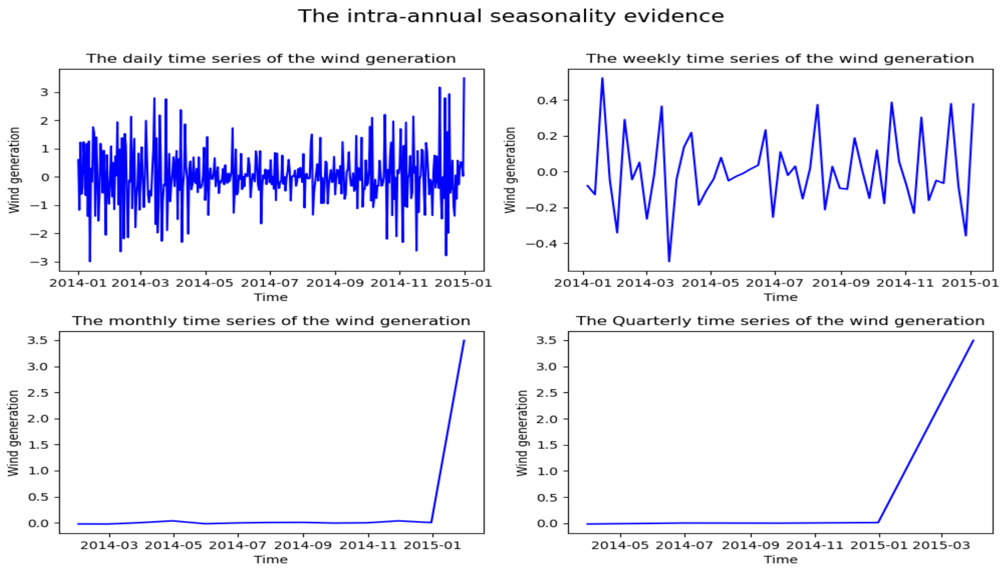


Figure 3: This graph illustrates the evolution of wind generation as a percentage of the maximum generation over time, represented by hourly increments from January to December 2014



Insight: The figure three above illustrates two outliers in the data. The sudden increase in wind generation, resulting in a noticeable change in percentage on that day, in my opinion was caused by a change in weather patterns. As shown in the figure, the impact of the weather change was short-lived and lasted for approximately an hour before returning to normal conditions.

Task 3:

The aim of this task was to study the fluctuations of wind power generation, both positive and negative, relative to the maximum generation capacity, over a one-hour interval. This fluctuation was defined as $r(t,d)=100*\left[x(t+d)-x(t)\right]/max(x)$, where d=1 for an hourly time period. The objective was to create empirical cumulative distribution functions (CDFs) for both positive and negative fluctuations, and to plot these CDFs with probability on a vertical logarithmic axis. Furthermore, a normal distribution was plotted with a mean of zero and standard deviation based on the data, and it was determined whether this normal distribution was a suitable model for the extremes of wind power generation as shown in the following steps:

- ❖ I created a loop to calculate ramps using the formula that was given a and the results that I obtained were appended to the declared variable
- ❖ I then separated the positive and negative ramps through another created loop and also sorted the calculated ramps but I considered the absolute value the negative ramps
- ❖ I plotted the cumulative distribution function of the normal distribution with respect to the calculated standard deviation and the mean xero
- ❖ I used empirical cumulative distribution functions to create visual representations of the sorted positive and negative ramps in wind power generation, and I plotted the probabilities on a vertical logarithmic axis for easy comparison and analysis

Results: This task was successfully completed, and I managed to plot the positive and negative ramps as shown in the figure 4 below. The blue curve shows the cumulative distribution function of a the normal distribution, the red scatters shows the empirical cumulative distribution function of the positive ramps and the green scatters shoes the empirical cumulative distribution function of the negative ramps.

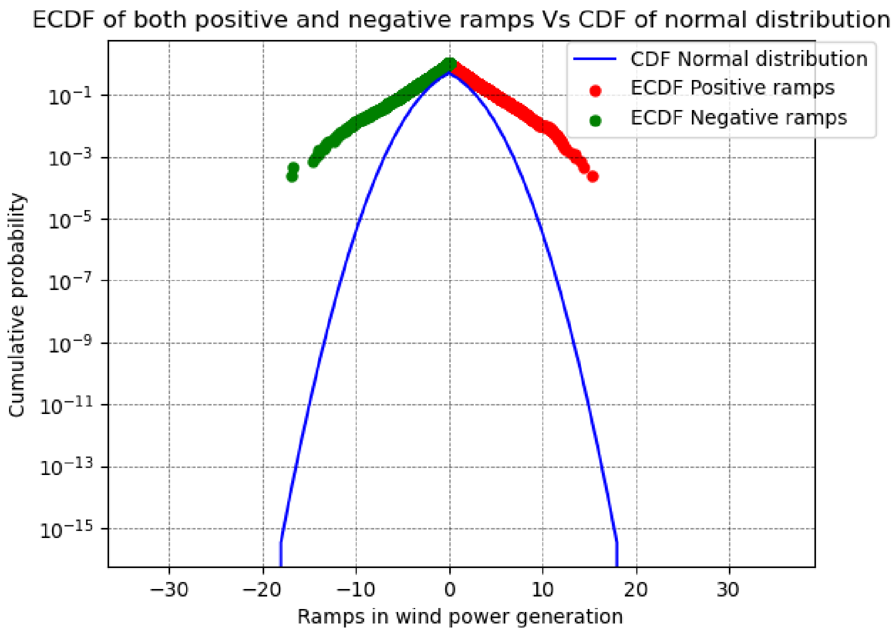


Figure 4: the graph displaying the empirical cumulative distribution function (CDF) of positive and negative ramps, as well as the normal cumulative distribution

Insight: It is crucial to consider positive and negative ramps separately in the analysis of how wind power changes affect system operation to have a more nuanced understanding of the ramps. As shown in figure 4 above it is rare that the wind power changes exceed 20%.

Task 4:

The task required examining fluctuations over time frames ranging from one hour to one day by plotting the 1%, 5%, 95%, and 99% percentiles. This was accomplished by employing ramp distributions r(t,d) with d values ranging from 1 to 24.

- I created a function that finds the ramps by inserting a preferred timescale and appended the obtained values to a declared carriable
- I then declared four independent variables because we aimed to calculate for given four percentile and I also created four independent loop where each loop over every hour of the day and to calculate the percentile and the obtained values were appended in their corresponding variables
- Then I plotted the obtained results with respect to the hourly timescale

Results: Due to the 24-hour timescale, I obtained 24 values for each percentile as a result of the analysis. This means I was able to calculate a value for each timescale and percentile. As shown in Figure 5, the values obtained for the 1st and 99th percentiles have the same magnitude but differ in sign. Similarly, the 5th and 95th percentiles have the same magnitude but different signs.

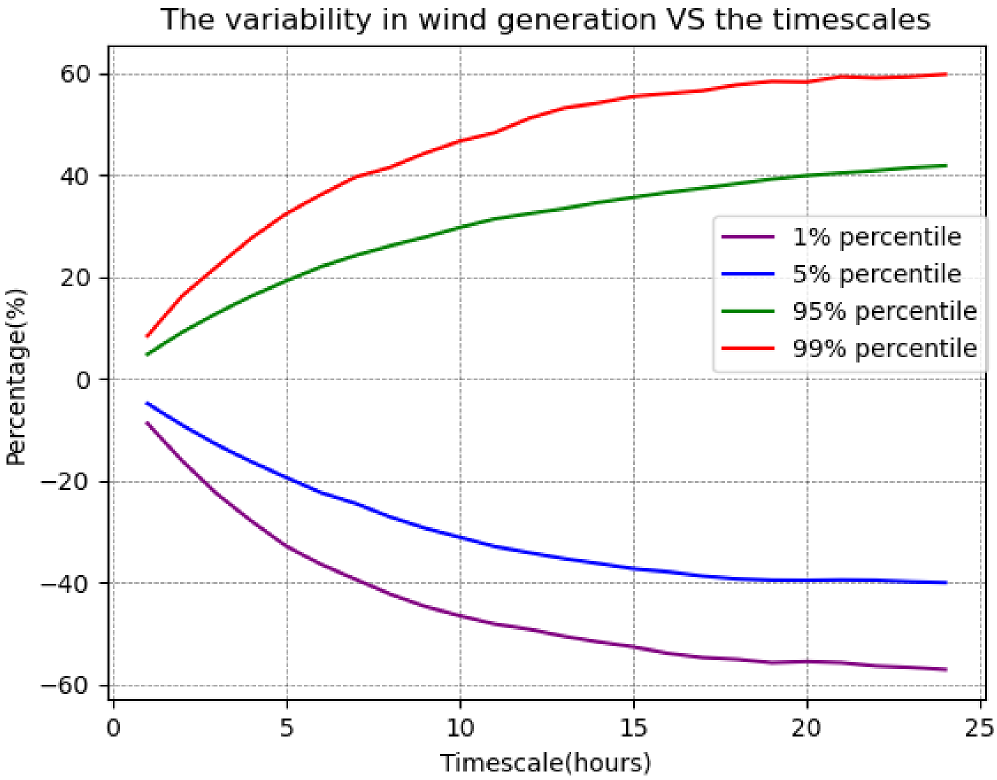


Figure 5: The graph that shows the variability in wind generation due to different timescales

Insight: The results shown in the graph above indicate that our data set is symmetrical, indicating that the variability in wind generation was distributed uniformly this year. In reality, however, wind generation variability is not always uniformly distributed. This is due to the fact that wind generation is influenced by a variety of factors, including weather events, geographical location, and other environmental conditions. These factors can vary non-uniformly, resulting in non-uniform wind generation distributions over time.

Task 5:

The goal of this task was to determine the autocorrelation of wind generation at lags greater than 10 days and to analyse the pattern of the autocorrelation by providing insightful comments and the following steps helped to bring this through completion:

- I calculated the autocorrelation of the wind generation using a pandas function' autocorr' and I considered 10 days times steps which correspond to 240 I used as the lag of the calculation
- I then used the function 'plot_acf' that belongs to the python library 'statsmodel' to plot the autocorrelation of the wind generation

Results: The calculated autocorrelation for wind generation with a lag of 10 days is 0.157. The autocorrelation decreases as the time steps increase, eventually leveling off to a nearly constant value, as shown in the following figure 6.

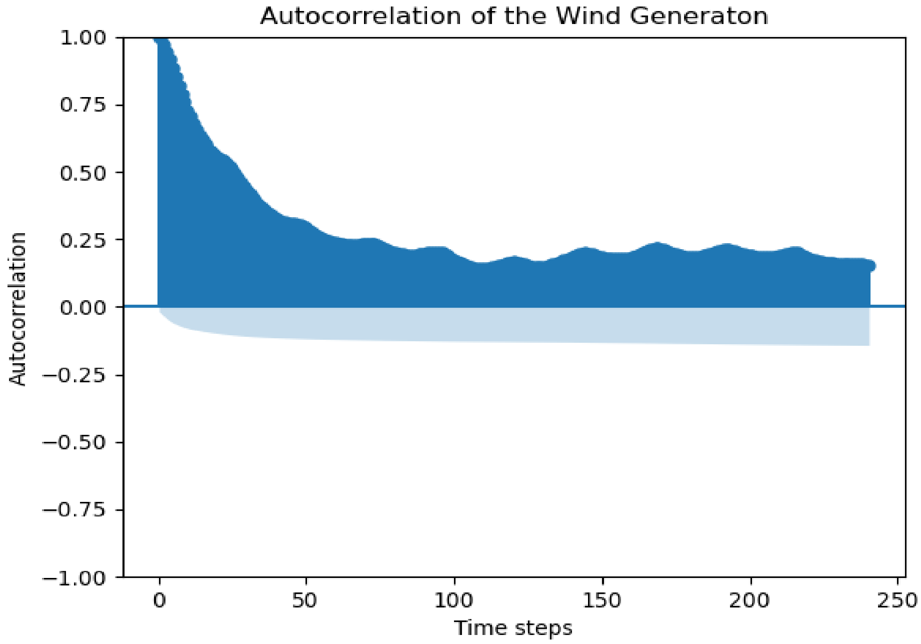


Figure 6:The graph that demonstrate the autocorrelation of the wind generation for 10 days lags

Insight: The autocorrelation of values separated by a 24-hour, or 1-day lag is substantial, indicated by a value greater than 0.50. This suggests a strong positive association between these values in the time series. However, as the lag increases to more than one day, the autocorrelation decreases dramatically, with values falling below 0.50 for lags between one and ten days. This weaker association could indicate either a weak negative relationship, or potentially no significant relationship, between the values in the time series.

<u>Task 6:</u>

This section evaluated the autocorrelation of the variations in wind generation over lags of more than 10 days with the intention of plotting the results. Horizontal lines were added to the plot to help identify any values that were statistically significant (p <0.05).

- ❖ I calculated the change in wind generation using the formula $X(\text{t})-X(\text{t}-1)$ when $X(\text{t})$ was the current wind generation value and $X(\text{t}-1)$ was the previous wind generation value. This was computed into a loop and the results were added to an empty declared variable
- ❖ In order to identify statistically significant values, I calculated the critical value from the standard normal distribution and represented it visually as a horizontal line.

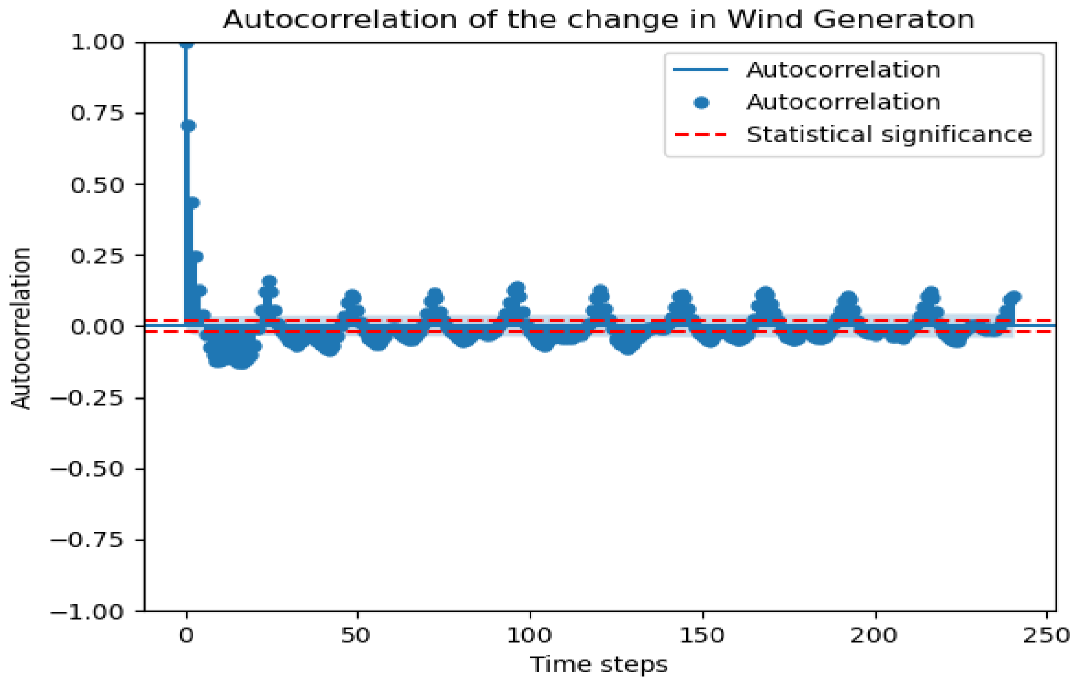


Figure 7: This is visualisation of the autocorrelation of the change in wind generation

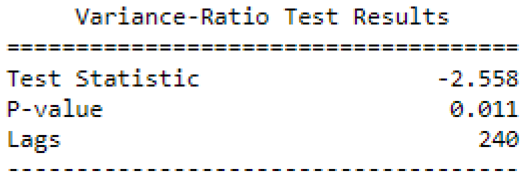
Insight: The above graph was visualized using 10-day time steps, and the results show a 10-unit fluctuation, which I believe indicates diurnal seasonality. In addition, it would be more appropriate to model the actual value of wind generation because according to figure 7 above the change in wind generation add seasonality.

Task 7:

The task is to examine the pattern of the wind generation time series using a variance ratio test to see if the assumption of a random walk (null hypothesis) can be rejected. In fact, the goal is to determine whether the series exhibits mean-reversion or mean-aversion and this was computed thanks to the subsequent procedures:

- ❖ I used the function “VarianceRatio” from a python library called”arch” but I also used the “adfuller” function from statsmodel library to compare the results. In fact, both functions do the same work.

Results: The variance ratio test was computed, and the process was found to not be a random walk which means that the null hypothesis of a random walk has been reject and this result in a stationary time series



```
     Variance-Ratio Test Results
=====================================
Test Statistic                 -2.558
P-value                         0.011
Lags                              240
-------------------------------------
```

Insight: The results of two statistical tests approaches that was conducted such as the Augmented Dickey-Fuller (ADF) test and the Variance Ratio (VR) test, suggest that this wind generation time series may have mean reversion. This is because the test statistic from the ADF test was found to be below all critical values, and the VR test resulted in a value less than one.

Task 8:

Finding the ideal window size for a simple moving average and figuring out whether there is a straightforward benchmark that performs better than the persistence benchmark are the tasks that was done in this part and the following steps helped to achieve it:

- I created a python function that calculate the simple moving average using the python function called "rolling()" for every given test window size
- I also calculated the mean absolute error between the simple moving average calculated in step 1 and the actual power generated from the wind that was given in the data set then I identified the test window size that was giving me the minimum mean absolute error

Results: The section was successfully completed, and the simple moving average was computed for each test window size. The existing columns were expanded by 24 columns, with each column representing the results of each test window consideration. The column produced for a test window size of one (n=1) was identical to the original wind generation data, yielding a minimum mean absolute error of zero. However, it would make more sense to consider the results obtained at n=2 because that's where the optimal prediction has taken place.

Insight:

In our situation, a Simple Moving Average (SMA) with a larger window size will produce a smoother depiction of the data's trend. But, using a smaller number of window size will show the smaller changes in the data available. In a timeseries data set, future values can be predicted using the Simple Moving Average (SMA). However, if a small window size was used, this method might not produce reliable predictions on its own [1]. It is preferable to combine SMA with other techniques like ARIMA in order to obtain more accurate results.

Task 9:

The goal was to evaluate the persistence benchmark forecast's mean-absolute-error (MAE) performance for forecast horizons ranging from one hour to one day (1-24 hours), and to visualize the MAE as a portion of the maximum generation for the persistence benchmark.

- Firstly, I calculated the persistence through a python function using the formula $X\_predicted(\mathrm{t}) = \mathrm{X(t-n)}$ and removed the missing values for future analysis
- Secondly, I calculated the mean absolute error between predicted values obtained from formula $X\_predicted(\mathrm{t}) = \mathrm{X(t-n)}$ and the actual wind generation using a python function "mea" from Scikit-Learn library
- The mean absolute error expressed as a percentage of the lowest value generated for the prediction was then plotted.

Results: As the results shows in the following figure the percentage of the mean absolute error increases from 1% at 1 hour timescale to 17% at 24 hours timescale.

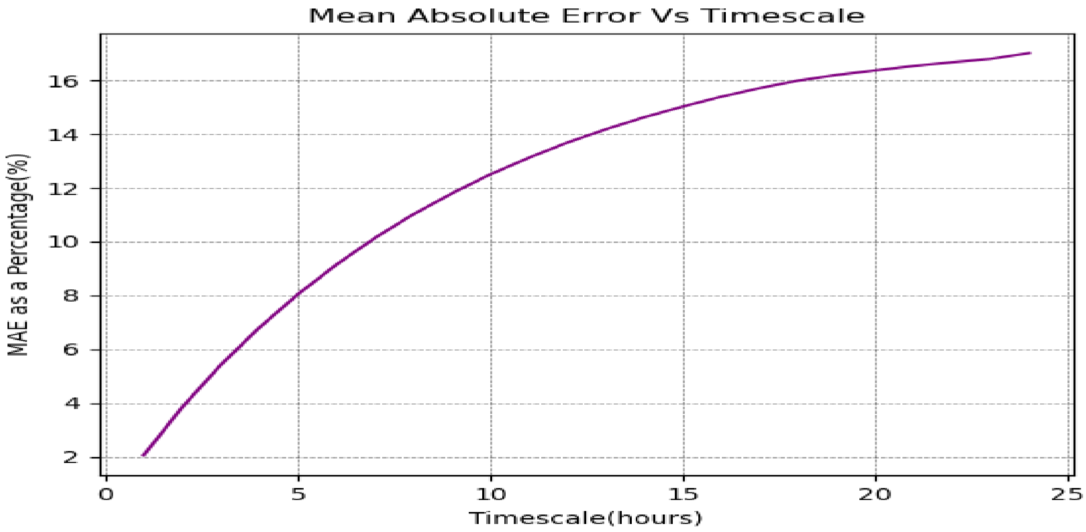


Figure 8:A visualisation of a mean absolute error as a percentage of maximum generation for the persistence benchmark

Insight: Based on the results shown in the graph, the current wind generation data is more reliable for predicting the next hour, but if we try to use today's data to predict further into the future, the predictions may not be accurate. This is because the weather conditions can change and affect wind energy generation.

Task 10:

The task on this section was to iterate through various parameter values for an ARIMA model that will model wind generation, and then use information criteria like AIC and BIC to determine the best ARIMA model. To bring this task through completion the following steps were followed:

- I established an ARIMA model by specifying the values of parameters d and q to range from 1 to 4, and then I applied the model to fit the data.
- I extracted the AIC and BIC values after fitting the model established in step one and added them separately to a newly created empty variable.
- Additionally, I retrieved and recorded the parameters for each iteration and stored them in a designated variable.

Results:

The outcome indicates an improvement in the performance of the model as it has the lowest values of AIC (Akaike Information Criterion) and BIC (Bayesian Information Criterion). The obtained minimum AIC value was 98781.65 while the minimum BIC value was 98817.04. The optimal model with the best performance would have the parameters (3,1,1), where the p value is set to 3, d parameter is set to 1, and q parameter is also set to 1.

Insight: There are many advantages to using ARIMA models to analyse data on wind generation, particularly for forecasting. Because they can produce incredibly accurate forecasts, these models are crucial for making wise choices in energy management [2]. Many researchers have found that ARIMA models are a good way to make predictions about wind power. For example, some scientists in China used ARIMA to guess how much wind energy a wind farm would produce, and they said it was better than other ways of predicting wind power. [3].

References

[1] Anjali, "A Practical Introduction to Moving Average Time Series Model," Project Pro, 02 February 2023. [Online]. Available: https://www.projectpro.io/article/moving-average-time-series-model/716. [Accessed 13 February 2023].

[2] X. Wang, P. Guoc and X. Huang, "A Review of Wind Power Forecasting Models," *Energy Procedia,* no. Chengdu, China, 2011.

[3] M. Lei, L. Shiyan, J. Chuanwen, L. Hongling and Z. Yan, "A review on the forecasting of wind speed and generated power," *ELSEVIER,* vol. 13, no. Issue 4, pp. 915-920, May 2009.

[4] G. James, D. Witten, T. Hastie and •. R. Tibshirani, An Introduction to Statistical Learning, London: Springer, 2021.

[5] S. S. Skiena, The Data Science Design Manual, New York: Springer, 2017.

[6] A. Nagpal, "L1 and L2 Regularization Methods," Towards Data Science, 13 October 2013. [Online]. Available: https://towardsdatascience.com/l1-and-l2-regularization-methods-ce25e7fc831c. [Accessed 20 January 2023].

[7] A. Nagpal, "L1 and L2 Regularization Methods, Explained," Buit in, 05 January 2022. [Online]. Available: https://builtin.com/data-science/l2-regularization. [Accessed 20 January 2023].

[8] B. Giba, "Elastic Net Regression Explained, Step by Step," Machine learning compass, 26 June 2021. [Online]. Available: https://machinelearningcompass.com/machine_learning_models/elastic_net_regression/. [Accessed 20 January 2023].