ASSIGNMENT 1

ANDREW ID: parmenin

18-787: Data Analytics

1/26/23

Niyomwungeri Parmenide ISHIMWE

I, the undersigned, have read the entire contents of the syllabus for course 18-787 (Data Analytics) and agree with the terms and conditions of participating in this course, including adherence to CMU's AIV policy.

Signature: **Niyomwungeri Parmenide ISHIMWE**
Andrew ID: **parmenin**

Full Name: **Niyomwungeri Parmenide ISHIMWE**

LIBRARIES USED

- import numpy as np
- import pandas as pd
- import matplotlib.pyplot as plt
- import seaborn as sb
- import statsmodels.api as sm

QUESTION 1:

It was asked to download the historical daily weather daI for France and load them into the environment (Jupyter notebook on my side). After loading them, checking for empty and Nan fields is done, and filled fields with "-" with Nan using the fillna () function. and hence they are filled using linear interpolation with the help of the pandas interpolate () function[1]. After that, the "high Gust Wind (km/h)", and "Events" columns are dropped because they have many nan values and string objects that cannot be used for our analysis.

QUESTION 2:

The next step is to calculate the correlation matrix between all the weather variables, which was done after dropping the date column, calculated, and then depicted in the following heatmap graphic.

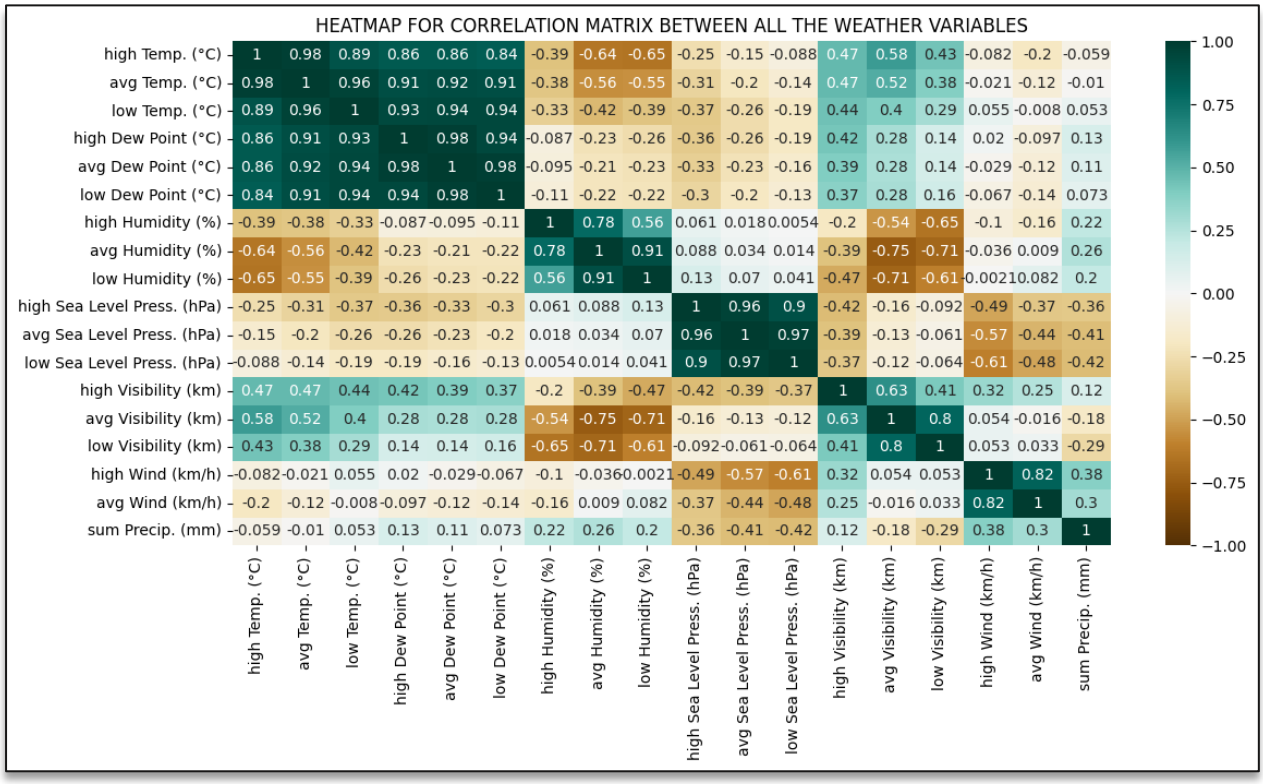

Figure 1: Weather variables correlation matrix heatmap

By looking at the correlation matrix heatmap, it can be inferred that hthereare six weather variables that are very correlated or have strong relationships of above 83% between them. Those are "high Temp. (°C)", "avg Temp. (°C)", "low Temp. (°C)", "high Dew Point (°C)",

“avg Dew Point (°C)”, and “low Dew Point (°C)”. It can also be seen that “low Humidity (%)” and “high Wind (km/h)” are the least correlated or are likely to be not correlated, with the smallest correlation coefficient of 0.21%.

QUESTION 3:

It was asked to download the historical daily electricity consumption data for France, save them as CSV, load them into the environment (Jupyter notebook on my side), and extract useful columns. After that, empty rows are dropped using the dropna() function and resetting the index to avoid the gaps between them.

QUESTION 4:

It was then asked to synchronize the dates for both time series i.e., for electricity (energy) consumption and weather (temperature) data, and then a scatter plot against each other. First, both dates are changed from string to timestamps using the pandas’ to_datetime() function. Then, the dates are used to synchronize both datasets using the merge () function on the ‘Date’ column. Finally, energy consumption (Energie journalière (MWh)) and mean temperature (avg Temp. (°C)) are plotted on the below scatter plot.

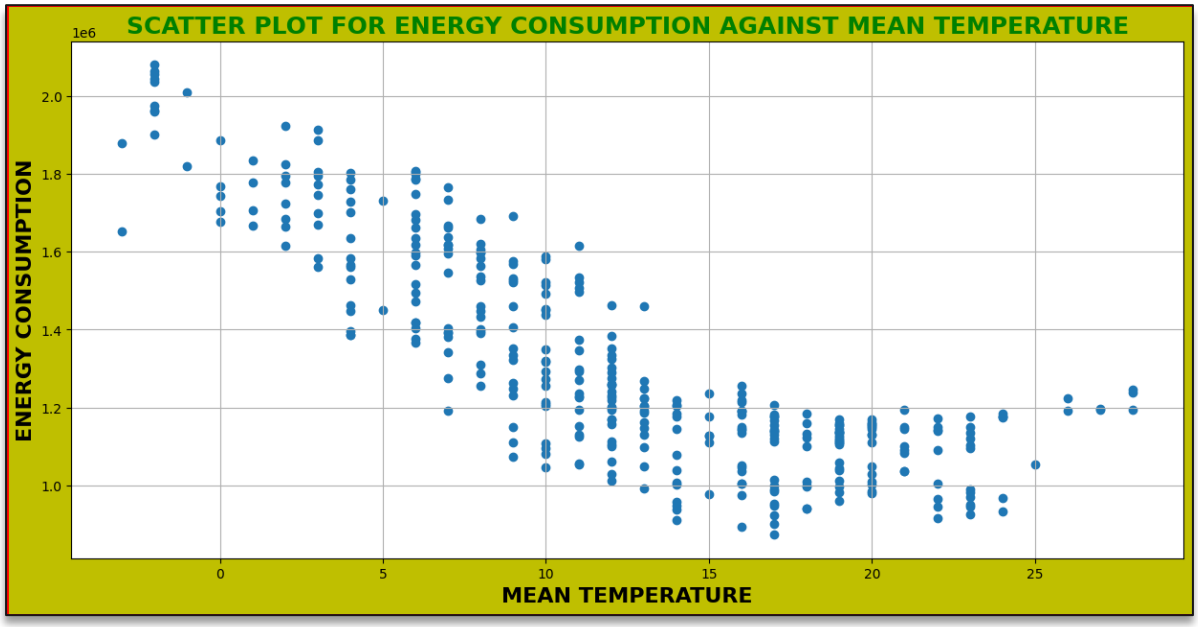

Figure 2: Weather variables correlation matrix heatmap

From the previous graph, we can infer that as the mean temperature increases, causes the energy consumption to drop in general. In addition, the minimum and the maximum mean temperature in the dataset are -3°C and 28°C respectively. Moreover, the minimum and the maximum mean energy consumption in the dataset are 873166.0 MWh and 2080679.0 MWh respectively.

QUESTION 5:

For this, it was asked to fit a quadratic model with the energy against the temperature, and then graph the quadratic fit as a line on top of their scatter plot. The steps used are first, to find the coefficients of a polynomial equation using the NumPy's polyfit () function. Next, the polynomial object (model) is created using the poly1d () function from NumPy.

The next step is to generate the points for the line i.e., x-axis points and y-axis points. To find the x-axis points, the NumPy's linspace () function is used to generate evenly spaced numbers between the minimum and maximum value of the mean temperature. Moreover, to find the y-axis points, the created polynomial model object is used along with the generated x-axis points to give y-axis points corresponding to the x-axis points.

Finally, a scatter plot is plotted using the mean temperature and the energy consumption, and a quadratic line is drawn using the x-axis and y-axis points generated in the above steps.

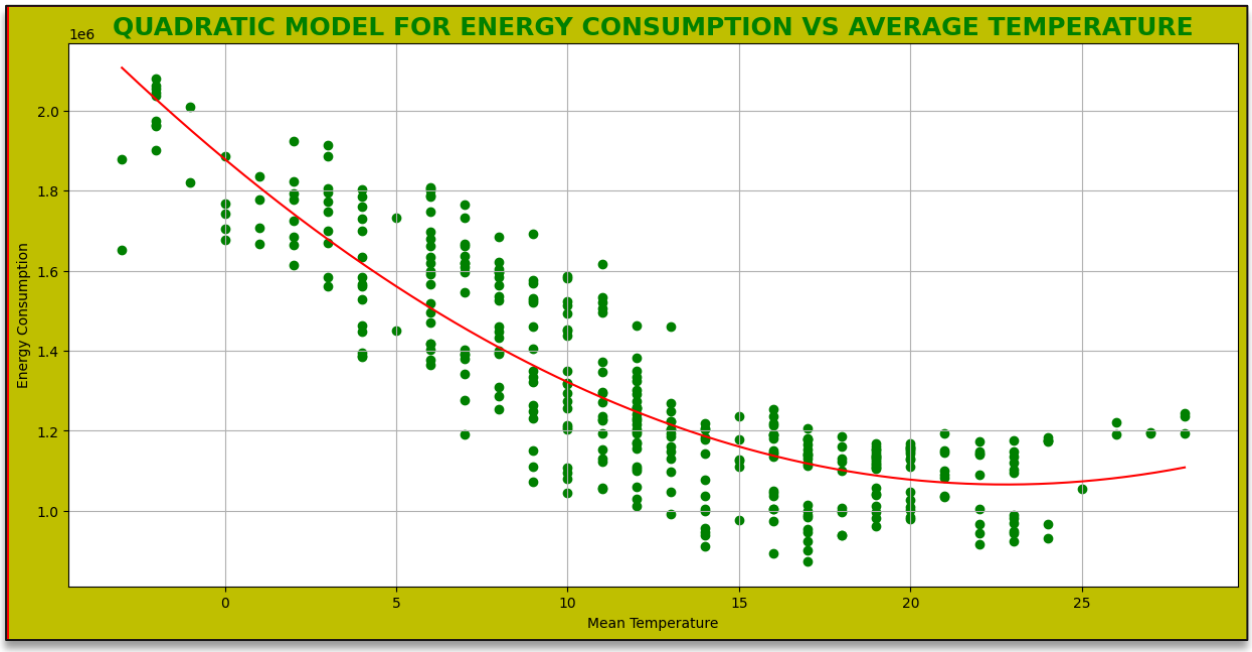

Figure 3: Quadratic fit as a line over the scatter plot of temperature vs energy

QUESTION 6:

Using the empirical analysis, it is asked to find the optimal temperature coinciding with minimal energy consumption using the quadratic fit to verify visually. This was done by first, identifying the minimum energy consumption using the min () function. Then using the NumPy's argmin () function, an index corresponding to the minimum value for the points on the y-axis is found and this index is then used to identify the optimal temperature.

To show this optimal temperature, the same plot as the previous question is plotted below and then the optimal temperature is shown using an orange scatter dot and annotated with an arrow using the matplotlib's annotate function.

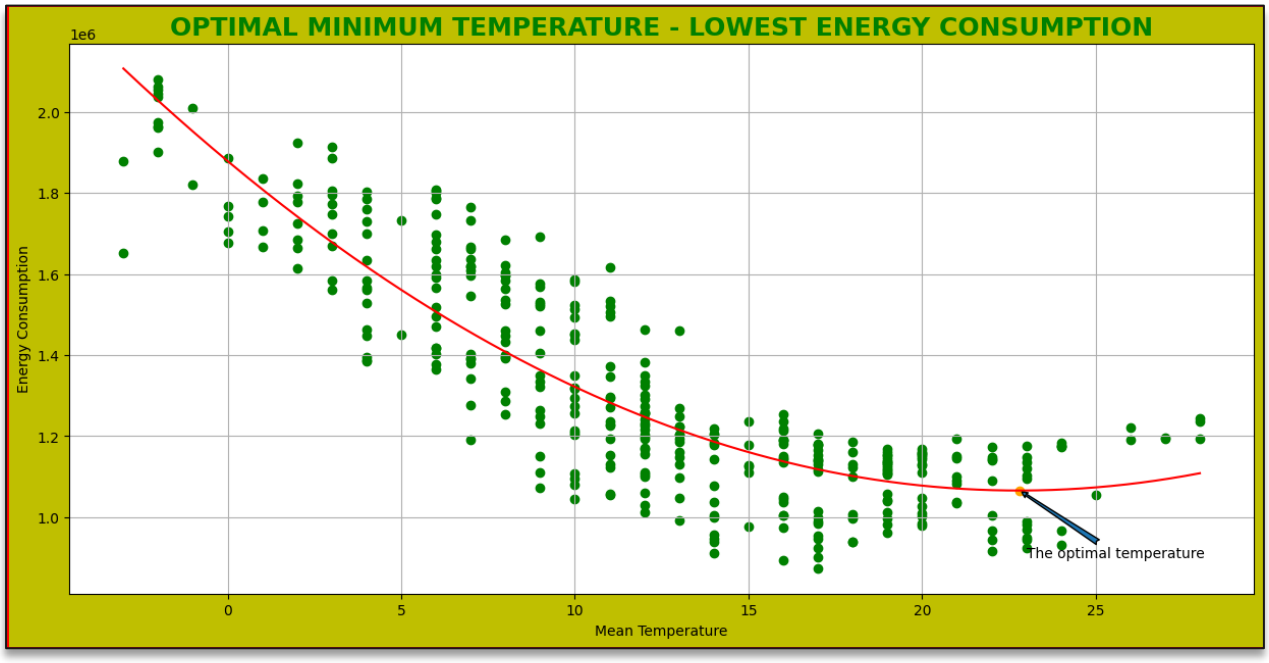

Figure 4: The optimal temperature coinciding with minimal consumption

As it is depicted above, the minimum energy consumption is equal to 1065867.1469665153 MWh and the optimal temperature is equivalent to 22.804945054945055 °C.

QUESTION 7:

The task here was to use a stepwise approach to find an optimal multivariate linear regression model using the weather variables to forecast energy consumption. For this, is used forward regression, which starts with an empty model and iteratively adds variables to the model until no further improvement is made to the model[2].

Based on the p-value from the statsmodels, this forward selection is carried out. The OLS function of the api is used to create a model. All candidate features are contained in the X data frame, while the target variable's values are listed in the Y data frame. When a feature is included, its p-value must be lower than the threshold for inclusion (thresholdToBeIn=0.05). The inclusion sequence is printed verbosely to accomplish this. Last but not least, this method produces a list of chosen features [3]. Using this method, 7 features were selected which are:

['high Temp. (°C)', 'high Visibility (km)', 'high Humidity (%)', 'avg Temp. (°C)', 'low Humidity (%)', 'avg Dew Point (°C)', 'low Sea Level Press. (hPa)'].

To calculate the coefficient of determination or the r squared, a constant is added to each feature, and an OLS model is created using energy consumption and features with the constant added. This model helps to get the coefficient of determination which is equal to **0.7506437341112864** or approximately 75%.

QUESTION 8:

By considering squared terms for each weather variable (squaring their values), the number of explanatory variables is doubled. The forward selection method is also used to select the best features to be used for the new model.

The result is a list of 4 useful features selected as: **['high Temp. (°C)', 'high Temp. (°C) _squared', 'high Visibility (km)_squared', 'high Visibility (km)']**

Furthermore, the new coefficient of determination after adding the squared terms is calculated to be **0.8068265031072407** which is 80% approximately. This implies an improvement of around 5% from the previous model.

QUESTION 9:

By also considering the days of the week using dummy variables for the day of the week in the multivariate regression, the three days i.e., **['Sunday', 'Saturday', 'Monday']** are selected among other 11 features that were selected which are: ['high Temp. (°C)', 'high Temp. (°C)_squared', 'Sunday', 'Saturday', 'avg Temp. (°C)', 'low Humidity (%)', 'high Wind (km/h)_squared', 'Monday', 'sum Precip. (mm)', 'avg Temp. (°C)_squared', 'high Dew Point (°C)_squared'].

Moreover, the final coefficient of determination after adding dummy variables for the days of the week is equal to **0.8945057843312311**, or almost 89 percent. This suggests a 14% improvement over the first model and a 4% improvement over the model with squared terms. Therefore, this last model is the best model that can be more useful than the previous models.

QUESTION 10:

No, it is not possible to be sure that this modeling strategy is not overfitting, as overfitting can be caused by several factors, including high model complexity, low size of training data, and noisy or erroneous data[4]. By using forward regression, the final model has a big number of features selected, eleven (11), hence much complexity compared to others; and a high coefficient of determination (89%).

Two approaches that can be used to prevent overfitting and hence improve the model performance are:

Regularization: Regularization methods such as Lasso, Ridge, and ElasticNet add a penalty term to the cost function that discourages large coefficients and helps to make the model less complex. This can lessen overfitting by bringing the coefficients of less significant variables closer to zero[5].

Cross-validation: Cross-validation is a technique that enables you to assess a model's performance by training it on a subset of the data and then testing it on a different subset. This may lessen the risk of overfitting by lowering the risk of memorizing the noise in the training data instead of generalizing it to new data. Some methods used for Cross-Validation are K-fold cross-validation, Stratified k-fold cross-validation, Leave one out cross-validation, Leave-P-out cross-validation, and Validation Set Approach[6]

REFERENCES

[1] ‘pandas.DataFrame.interpolate — pandas 1.5.3 documentation’. https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.interpolate.html (accessed Jan. 23, 2023).

[2] ‘Stepwise Regression in Python’, *GeeksforGeeks*, Dec. 28, 2022. https://www.geeksforgeeks.org/stepwise-regression-in-python/ (accessed Jan. 23, 2023).

[3] ‘stepwise-regression/step_reg.py at master · AakkashVijayakumar/stepwise-regression’, *GitHub*. https://github.com/AakkashVijayakumar/stepwise-regression (accessed Jan. 24, 2023).

[4] T. A. Team, ‘Overfitting: Causes and Remedies – Towards AI’. https://towardsai.net/p/l/overfitting-causes-and-remedies, https://towardsai.net/p/l/overfitting-causes-and-remedies (accessed Jan. 24, 2023).

[5] P. Gupta, ‘Regularization in Machine Learning’, *Medium*, Nov. 16, 2017. https://towardsdatascience.com/regularization-in-machine-learning-76441ddcf99a (accessed Jan. 24, 2023).

[6] ‘Cross-Validation in Machine Learning - Javatpoint’. https://www.javatpoint.com/cross-validation-in-machine-learning (accessed Jan. 24, 2023).