Regression of Solar Irradiance Using a Neural Network

CAPSTONE PROJECT

FINAL REPORT

Hikmat Sultan

Contents

1 Introduction **2**
1.1 Machine Learning Literature Review 2
1.2 Machine Learning 2
1.3 Supervised Learning 2
1.4 Regression 3
1.5 Gradient Descent 3
1.6 Overfitting and Underfitting 4
1.7 Neural Networks 5
1.7.1 Activation Functions 7
1.7.2 Metrics 8
1.7.3 Backpropagation for Neural Networks 9
1.8 Validation set 10
1.9 Previous Work 11
1.10 Objectives 11

2 The Data Set **12**
2.1 Background 12
2.2 Features Description 13
2.3 Relevance and Correlation Analysis using EDA 13

3 Prediction Results **20**

4 Results Analysis and Comparisons **23**
4.1 Effects of normalization on results 23
4.2 Accuracy and Loss 24

5 Limitations **25**

6 Concluding Remarks **25**

1 Introduction

1.1 Machine Learning Literature Review

This project revolves around the idea of clearsky prediction which is a phenomenon that occurs when no clouds are visible throughout the entire sky dome. During this condition, clear sky solar irradiance begins to occur. This model will predict three attributes of solar irradiance to be able to theoretically predict how clear the sky is at a given moment in time. The attributes of solar irradiance that will be discussed in this project are Direct normal irradiance-DNI, Diffuse horizontal irradiance-DHI, and Global horizontal irradiance-GHI. Therefore, the current model will determine the degree of clearance in the sky by predicting the aforementioned attributes. Several important features will also be utilized in order to predict the attributes of solar irradiance. The dataset that was used in this project was extracted from Kaggle which is website that hosts countless datasets for machine learning scientists to try out. Additionally, Google Colab will be used as an environment to execute python code, and implement machine learning. Google Colab is a product created by Google to serve this purpose. It is widely used because it is a cloud-based free notebook environment that is free to use and most importantly, it offers its users free use of GPUs. The dataset that was used in this project is large. Thus, a GPU is necessary in order to properly train and test it.

1.2 Machine Learning

Machine Learning is a sub-field of Artificial Intelligence which is simply defined as training a machine to enable it to solve tasks across various fields. There are several methods of teaching machines such as Supervised Learning, Unsupervised Learning, and Reinforcement learning. Artificial Intelligence and Machine Learning are ever-expanding fields that are experiencing regular breakthroughs which allow for advanced models to be developed. Such models are being implemented into machines to enable them to tackle countless problems around the world.

1.3 Supervised Learning

The model developed in this project used supervised learning to perform its predictions. Thus, only supervised learning will be discussed about in this report. Supervised Learning is a method used in machine learning to train a model using well-labeled data which means that certain data has already been tagged with the

appropriate outputs. As a result, when new data is introduced into the dataset supervised learning algorithms will examine the data and attempt to predict correct output by using the knowledge already learnt from the labeled data. Supervised Learning algorithms are extremely important because eventually they allow models to predict highly accurate outputs for new data. Such models can then be used to perform daily repetitive tasks such as factory jobs and eventually harder tasks such as surgery. However, they require constant human intervention which that humans have to correctly label the data in order for the model to predict accurate results.

1.4 Regression

The model discussed later in this report implemented regression to achieve its results. Regression is a technique used in supervised learning when the output we want to predict is continuous. Generally, a regression algorithm plots a line through the data points. To create the best fit line, the distance between each point and the line is minimized. This is done in order to establish a relation between independent variables (value used to predict) and dependent variables (value we want to predict). Once a correlation has been established, predictions can begin to occur. In general, supervised learning algorithms use regression by understanding the relationship between predicted outputs and actual outputs. This understanding comes from the previously mentioned labelled data which is used by models to perform a prediction. The distance between the prediction and the actual value is then calculated to estimate the accuracy of the model. Models that use regression normally attempt to predict future trends from unseen data or based on previous patterns. Popular implementations of regression are found in machine learning in finance where models are tasked to predict continuous values such as stock costs and house prices. This report will discuss a technique implemented in regression models called Multi-output regression or Multiple linear regression. This method is used when the model is attempting to predict 2 or more values. Models that implement multiple regression aim to establish a linear correlation between a dependent variable and two or more independent variables. In some cases the correlation can result in a non-linear relation.

1.5 Gradient Descent

Gradient descent is a popular optimization method used in machine learning. It is an optimization method used for finding a function's local minimum which is

done by iteratively tuning the parameters of a function until the cost is minimized a much as possible. The function is then used to minimize a cost function. A cost function is used to determine how inaccurate the model is by establishing a relationship between the input and output. It is crucial for training a model because it informs us how poorly our model is performing and also it helps us to avoid underfitting and overfitting. Thus, gradient descent is an optimization technique used to optimize the cost function by tweaking the parameters of that function until a local minimum is reached (the least amount of error possible). Gradient descent can be visualized by imagining a person descending a hill (convex function). As the person descends, the error will continue to decrease until a global minimum is reached; where the error is the least and the algorithm becomes optimized. A very important attribute of gradient descent is the learning rate. It determines the rate at which the model will learn. A high learning rate allows the model to learn faster while a low learning rate will enable it to learn slower. At first, it seems that a high learning rate should always be implemented however, this might lead to the model to overshoot the global or local minimum because it will keep on bouncing back and forth. Moreover, if the learning rate is set to a low vales, the global or local minimum will eventually be reached however, it may take some time.

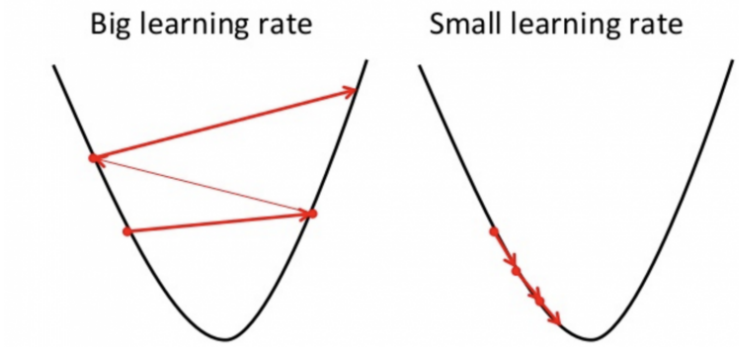

Figure 1: The difference between a high learning rate and a low learning rate

1.6 Overfitting and Underfitting

During training, all model experience some degree of bias and variance. Bias is a type of systematic inaccuracy that arises in machine learning models as a result of false assumptions made during the learning process. Variance is the variation in the model prediction-how much the machine learning function can change based

on the data set.A correct ratio of bias and variance can enable the model to predict more accurate results and less errors. Machine learning algorithms experience two types of errors: reducible and irreducible errors. Irreducible errors are out of our control. They result from the nature of the dataset and its features. On the other hand, we can control reducible error. It can be controlled by tuning the bias and variance which govern flexibility and complexity respectfully until an accurate output is achieved. Figure 1 displays an overfit model and an underfit model. A high variance will result in an overfit model which is too complex and cannot generalize well to new data. A high bias will result in an underfit model that will compute inaccurate results due to its naiveness. Thus, an appropriate trade-off between bias and variance must be reached in order to obtain accurate results.

1.7 Neural Networks

Multi-output regression is supported by many machine learning algorithms. However, the current model used a neural network in order to achieve its results. A neural network is a very popular machine learning algorithm that is currently being used frequently. Their overall structure is heavily inspired by the human brain, simulating the way actual neurons communicate with one another. Additionally, a feedforward neural network is a type of neural network in which the nodes do not form a cycle. It performs this process after every iteration based on the error achieved in the previous iteration Neural networks are made from several layers; an input layer, a hidden layer and, an output layer. Each layer contains a set of nodes that operate together to form an output. Moreover, every node is associated with a weight and a threshold. If a node's output exceeds a certain threshold value, the node is activated, and data is sent to the next layer of the network. Otherwise, no data is sent. Figure 3 displays the mathematical formula of feedforward propagation. Each node contains a value 'z' which is the weighted sum of inputs of the input units. The value 'b' is a bias unit which adds a constant to allow us to shift the activation function either to the right or to the left, this can be helpful in various scenarios. Applying an activation function on 'z' will result in an activation value 'a' which will be outputted to the next layer. Neural networks become extremely powerful algorithms once they are trained properly. That is the reason why a lot of care and effort has to be put into training them; the result of which can become outstandingly reliable models. Various real-world tasks such as speech recognition or image classification can take up to several minutes to give a result while manual identification by humans can take

up to hours if not days. As previously mentioned, a neural network is a structure or net of layers consisting of neurons. Every neuron has a weight attached to it which is used to calculate a weighted sum of the inputs. A weight is a numerical value used to represent the strength of a relationship between two neurons. Note that the input layer only contains the inputs, it does not contain any neurons. As the model propagates through its net of layers, the weights of the neurons are adjusted depending on how much they contributed to the overall error, until they reach an error that is below the acceptable threshold. Thus, the output of a neuron results from applying an activation function on a set of weighted inputs. A neuron's inputs might either be features from a training set or outputs from neurons in a previous layer. The activation function is applied on a sum of weighted inputs in order to output a value that propagates to the next layer. This process continues as long as there are layers to propagate to. The output layer displays the final predictions of the model.

$$z = Wx + b \tag{1}$$

$$a = ReLU(z) \tag{2}$$

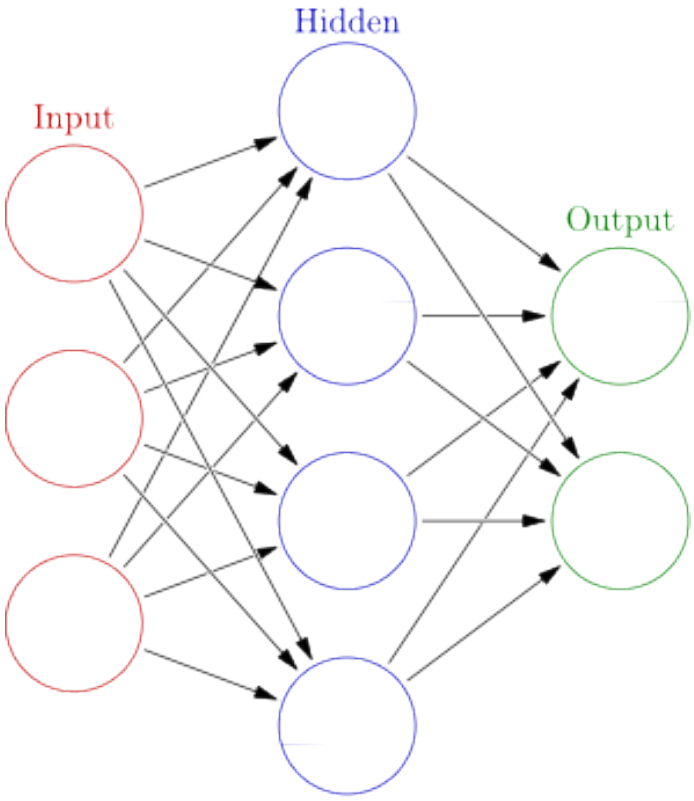

Figure 2: General formula for a Feedforward propagation along with the structure of a Neural Network.

1.7.1 Activation Functions

An activation function specifies how the weighted sum of the input is turned into an output by modifying the values they receive from previous layers and then sending the updated value to the next layer. The modified value depends on the activation function in use. In the current project, the Rectified linear unit or ReLU activation function was used. ReLU is a linear activation function that outputs the same input if the input was greater than zero, otherwise it will output zero. The output layer has no activation function due to this task being a regression problem. ReLU has recently become one of the most popular activation functions to use due to its computational simplicity and how well it performs on deep neural networks. These advantages made ReLU a compelling option to use in this project.

$$ReLU(x) = max(0, x) \tag{3}$$

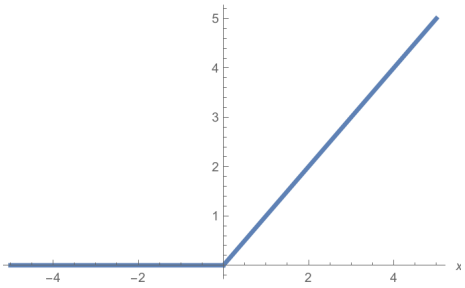

Figure 3: ReLU activation function

1.7.2 Metrics

Metrics are used to track and measure a model's performance during training and testing. They are similar to loss functions however, the results obtained from evaluating a model using a certain metric, are not used when training the model. Since we want to build a model to solve a regression problem, the chosen metric should understand continuous output in order to monitor the performance appropriately. There are several metrics that can be used:

- Mean Squared Error

$$MSE = 1/N \sum_{j=1}^{N} (y_j - y_j^2) \tag{4}$$

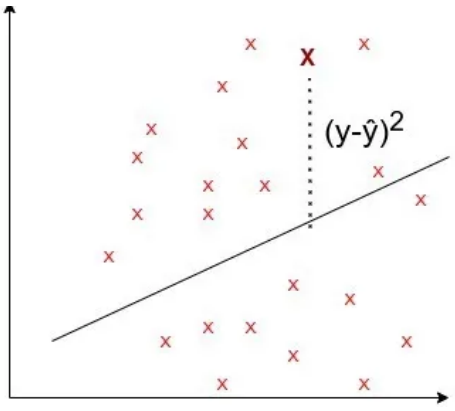

Figure 4: Mean Squared Error

Mean squared error or MSE is a regression metric that computes the average of the squared difference between the actual value and the predicted value. The squaring is done to remove all negative signs. Although, this might also lead to the model to exaggerate how bad the model actually is. The The lower the MSE, the more accurate the results are. Depending on the nature of the data, some datasets will never reach the line of best fit. That is, as close as possible to the actual values.

- Mean Absolute Error

$$MSE = 1/N \sum_{j=1}^{N} |y_j - y_j| \tag{5}$$

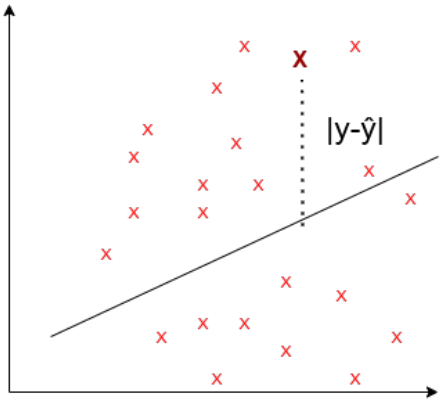

Figure 5: Mean Absolute Error

Mean absolute error or MAE is a regression metric that computes the average of the difference between the actual value and the predicted value. Unlike MSE, there is no squaring here. As a result, the model will not overestimate the results. However, because of the absolute value, we do not know whether we are underpredicting or overpredicting, i.e. the direction of the error.

1.7.3 Backpropagation for Neural Networks

Backpropagation is a technique used when training feedforward networks to fine-tune the weights of a neural network in order to lower the error rates. It minimizes

the cost function, making the model more dependable. The process begins by applying forward propagation which is used to compute the activation values of the layers. After computing all of the activation values of all the layers, backpropagation begins. It works by computing a value delta for each node in a particular layer; delta will represent the error of that specific node. Note that the input layer has no delta value because it only consists of the features of our dataset. Hence, it has no error. Backpropagation is widely used because of its fast and simplistic nature, and aside from the input, it has no other parameters to adjust. Furthermore, it is a versatile technique because it does not demand prior network understanding. However, backpropagation has its limitations. Its performance on a given task depends on the input data. Some input data are naturally problematic and require other techniques to fix.

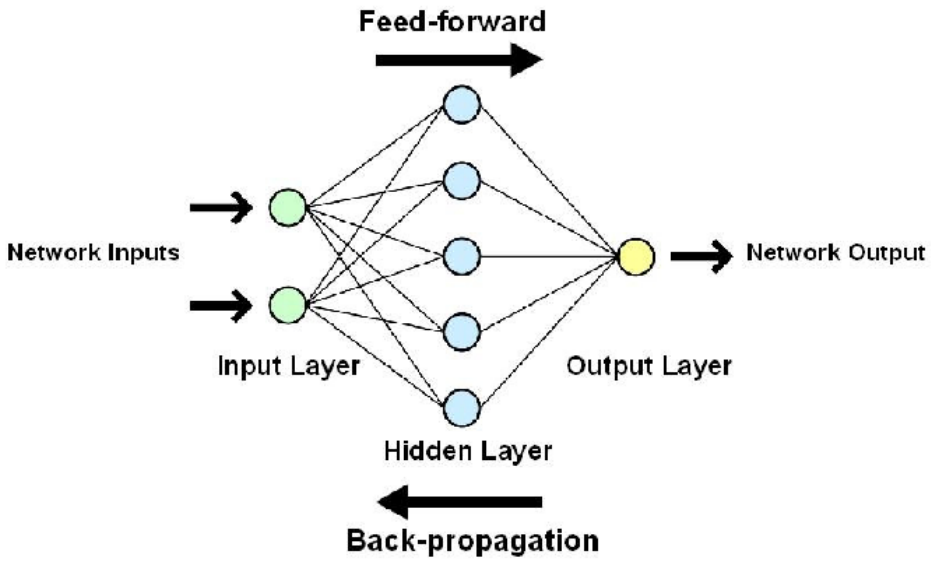

Figure 6: Backpropagation in feedforward networks

1.8 Validation set

Using a validation set when training a model has become a staple in the machine learning industry. A validation set is a sample of the data that is used to tune the hyperparameters on. Usually the dataset is split into training, validation, and

test sets. The test set differs from the validation in that the model is uses the validation set to tweak the hyperparameters throughout the training process. On the other hand, the model does not see the sample of data in the test set until the the entire training process is done. This is done to observe whether or not the model generalizes well to new data.

1.9 Previous Work

At the time this model was designed using the aforementioned dataset, there was only one notebook published using the same dataset on Kaggle. However, the author of that dataset only predicted the direct horizontal irradiance (DHI). Furthermore, the author also used a package called automl from the library flaml. Automl is a package that assists non-machine learning experts to train, test, and implement models into their choice of application. Automl is extremely helpful because usually the user has to understand most of the underlying math of machine learning algorithms in order to implement them. Thus, people from various fields of study such as medicine, agriculture, or business can easily implement a machine learning model using automl into their own field of choice. Furthermore, flaml is a fast library for autoML and tuning created by Microsoft. The author of the previously mentioned notebook expertly utilized automl by training the model using several algorithms including lightGBM, randomforest, catboost, xgboost, extratree, and xgblimitdepth. Automl automatically tunes the hyperparameters and performs various tasks that would normally require the author to manually configure. The author only specified the training set for the model to train on, the desired task which was regression in this case, the preferred metric, and the total amount of training time. Additionally, automl has a function called 'automl best estimator' which is normally used after training is complete. It is a very helpful tool which outputs the algorithm which resulted in the most accurate results. As you can see, automl is an exceptionally beneficial and convenient tool that can assist countless people in their respective fields.

1.10 Objectives

There are two main objectives set out for this project:

- To build a model that can efficiently utilize the given features in order to accurately predict the labels.
- To achieve predictions as accurate as the only published notebook using the

same dataset on Kaggle.

2 The Data Set

2.1 Background

The dataset consists of a variety of attributes that revolve around the weather such as temperature and pressure. It contains 175,297 rows and 18 columns which is fairly large. The dataset was separated into features and labels; 15 features which were used to predict 3 labels. The labels being DHI, DNI, and GHI. This dataset also contains data from 2009 which means that a lot of observed scenarios and instances have been recorded and placed into it. Furthermore, it includes samples collected throughout the year. This proved to be useful because it considered how the weather changes throughout the year due to the varying seasons.

2.2 Features Description

Direct normal irradiance	the amount of solar radiation received per unit area by a surface that is always held perpendicular (or normal) to the rays that come in a straight line from the direction of the sun at its current position in the sky.
Global horizontal indicator	the maximum solar radiation at a specific time and place on the earth's surface when no cloud is present.
Diffuse horizontal irradiance	the radiation at the Earth's surface from light scattered by the atmosphere.
Solar zenith angle	is the angle between the sun's rays and the vertical direction.
Year	the year at which the sample was recorded.
Month	the month at which the sample was recorded.
Day	the day at which the sample was recorded.
Hour	the hour at which the sample was recorded.
Minute	the minute at which the sample was recorded.
Cloud type	types of clouds ranging from 0 being how clear the cloud is till 15 which is smoke.
Dew point	the temperature below which water droplets begin to condense and dew can form in the atmosphere.
Temperature	the amount of heat that is present in a substance.
Pressure	the force generated on the Earth's surface resulting from the weight of the air above the surface.
Relative humidity	the amount of water vapour in the air expressed as a percentage of the amount needed for saturation at the same temperature.
Perceptible water	the depth of water in the column of the atmosphere if all the water in that column was visible as rain.
Wind direction	the direction the wind is originating from.
Wind speed	the current speed of the wind.
Fill flag	-

2.3 Relevance and Correlation Analysis using EDA

Exploratory data analysis or EDA is usually used in machine learning to clean the data and to better understand the relationship between the attributes of the dataset. Seaborn was used to fulfill that purpose. Seaborn is a data visualiza-

tion library regularly used in python to display the relationship between various attributes of the dataset. The below correlation heatmap matrix was generated before splitting the data into features and labels. Additionally, the correlation can be seen by observing the color palette. The darker the color, the higher the correlation.

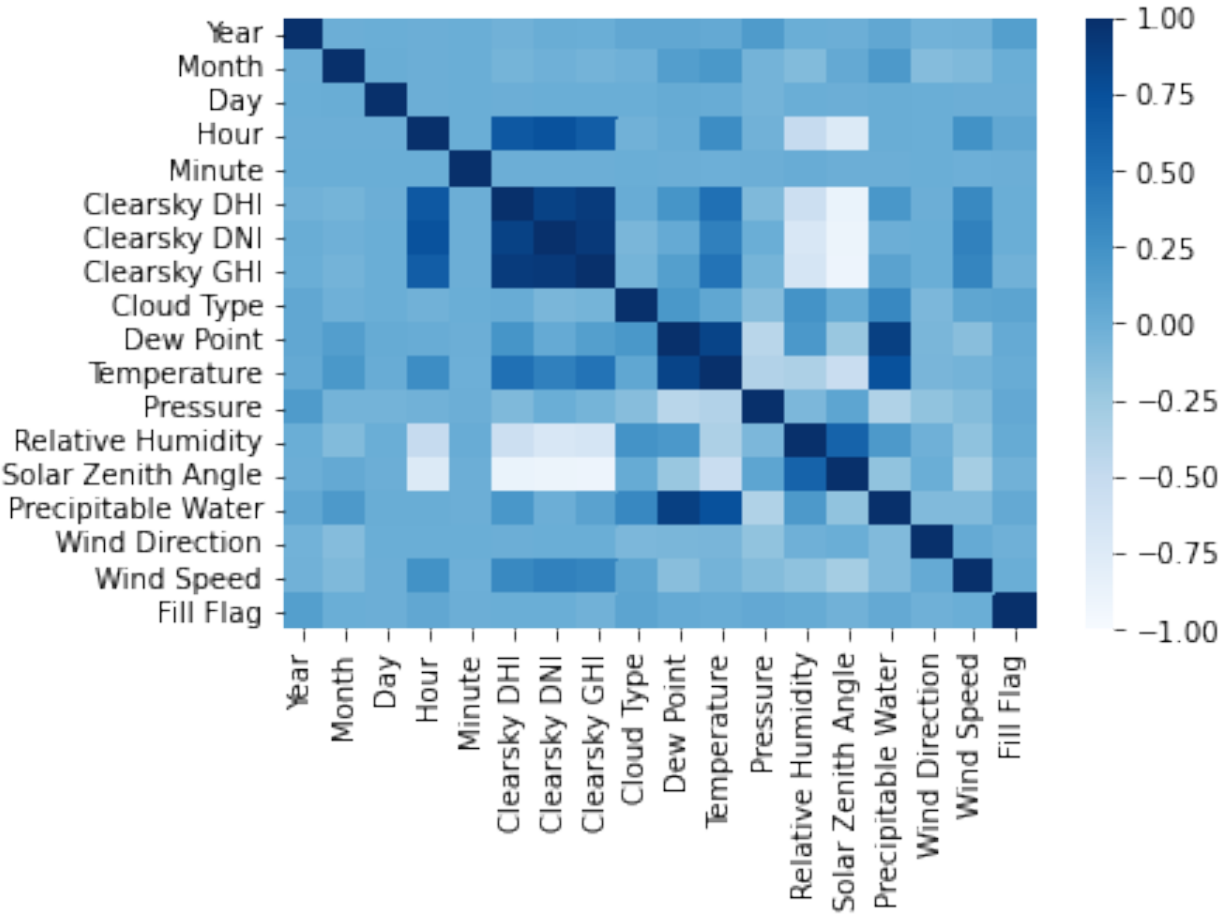

Figure 7: Correlation heatmap matrix for the features and labels

The diagonal line running through the matrix is completely correlated as shown by the color key. This expected since the attribute "Year" is surely going to correlate with itself. However, this matrix does show us useful correlations. Our labels 'Clearsky DHI', 'Clearsky DNI', and 'Clearsky GHI' (components of solar irradiance) seem to be correlated with the feature 'Hour' more so than the rest of the fea-

tures relating to time. This correlation leads us to believe that the aforementioned components depend on the hour of the day more than the year, month, day, and minute which showed little correlation. As expected, our labels showed correlation with each other since they are components of solar irradiance. Moreover, the feature 'Temperature' correlated with our labels. This might be because a certain temperature must be reached in order for the components of solar irradiance to occur. Additionally, Wind Speed appears to be correlated with our labels perhaps because wind speed must be high or low in order for the previously mentioned components to occur. Surprisingly, there are several features that negatively correlated with our labels. Relative humidity is one of them possibly because it is partially responsible for the formation of clouds which scatter or block the rays of sunlight from reaching the earth. The components of solar irradiance mostly depend on the sun and the amount of light reaching the ground. Furthermore, it seems that the Solar zenith angle is also negatively correlated with the components of solar irradiance. Solar zenith angle is defined as the the angle between the sun's rays and the vertical direction. DHI, GHI, and DNI need to interact with a horizontal surface or a surface that is parallel to the ground. Thus, the solar zenith angle should always be around 90 degrees because the sun's rays are coming from a vertical direction and hitting a horizontal surface. This confirms to us that these components of solar irradiance will need to hit a horizontal surface to occur because if they do not the solar zenith angle will not be 90 degrees. At first glance we might think that due to this fact, the solar zenith angle should be correlated with our labels. However, if we take the entire earth into consideration, there is a higher probability for rays of sunlight to not hit a horizontal surface due to the fact that the earth is not flat. As a result, the solar zenith angle will almost always be not correlated with the components of solar irradiance. There are several features that showed low or no correlation with our labels. Hence, whether or not they fluctuate, will have little or no affect on our labels.

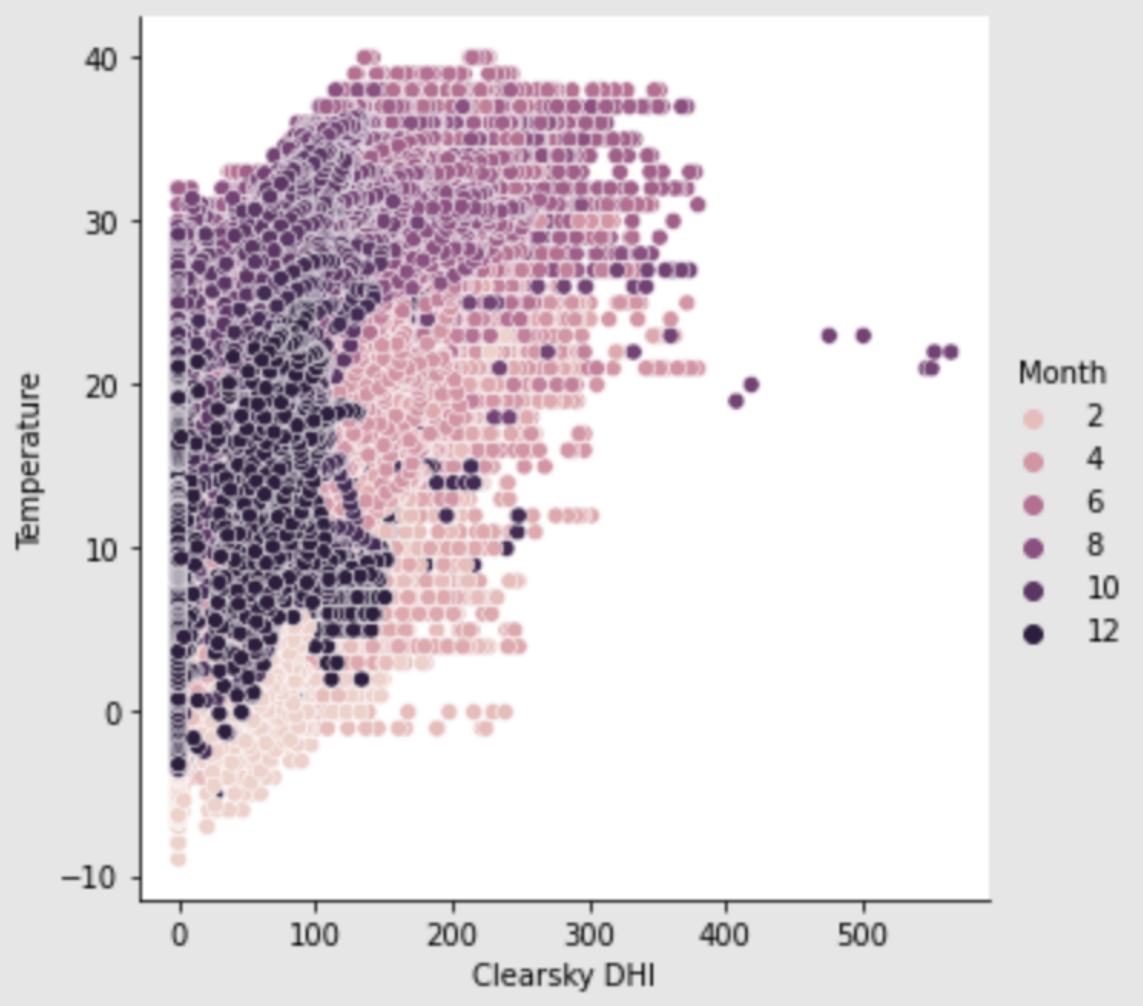

Figure 8: Relationship plot between DHI and the Temperature

Figure 8 displays a relationship plot generated using seaborn. The plot shows the month where DHI occurred with the corresponding temperature. It can be observed that the high values of DHI mostly occurred between February and June. Furthermore, the temperature recorded during those months was almost between -10 and 25. Note that the unit of measurement of the temperature was not specified however, it can be deduced to be celsius. It seems that DHI mostly occurs between the months of February and June when the temperature is between -10 and 25 degrees celsius possibly because there are other factors that are causing DHI to increase in that period of time. This was deduced because the majority of

the recorded data that was recorded between the months of August and December also experienced the same temperature that occurred between the months of February and June. This led us to believe that DHI levels increased due to other factors that only occur between February and June.

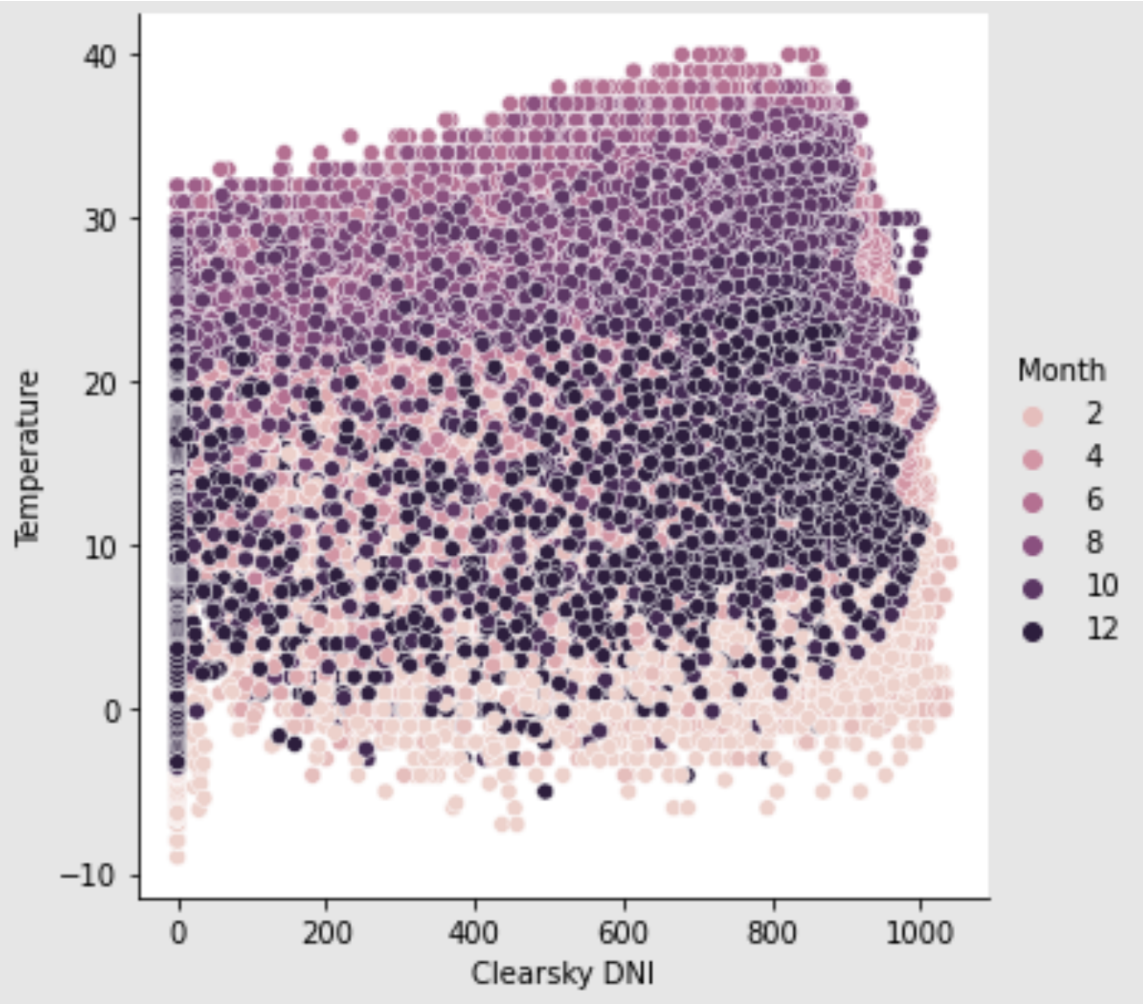

Figure 9: Relationship plot between DNI and the Temperature

The figure above shows a relationship plot also generated using seaborn. It is similar to figure 8 however, the y-axis is DNI. It seems that high instances of DNI occurs throughout the year regardless of the month and temperature. Interestingly, similar to figure 8, DNI did not occur a lot when the temperature was -10

degrees. It seems that extremely cold temperatures hinder the occurrence of DNI.

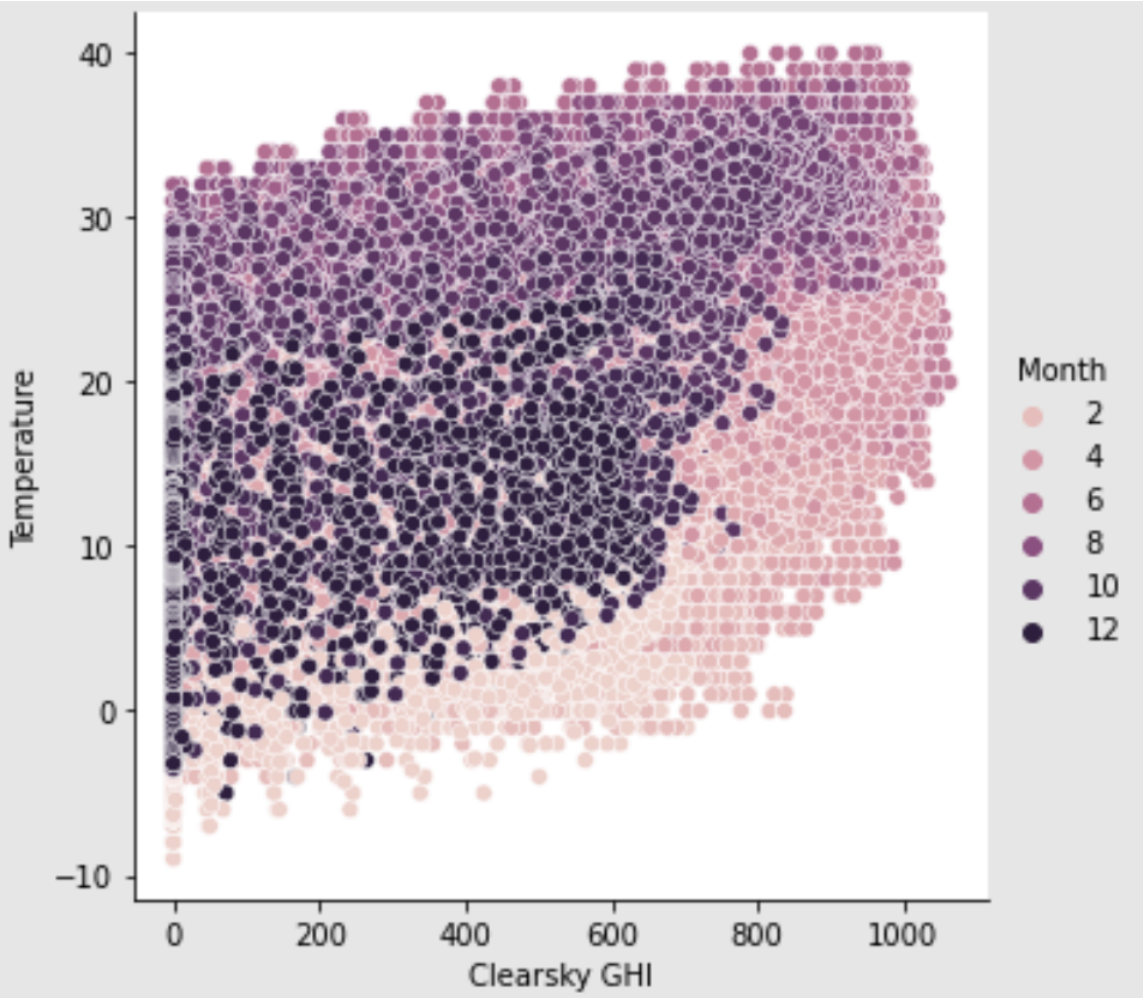

Figure 10: Relationship plot between GHI and the Temperature

Similar to figure 8, high values of GHI (700-1000) seem to occur between the months of February and June. GHI is defined as the maximum solar radiation at a specific time and place on the earth's surface when no cloud is present. It can be deduced that in most cases there are few cloud seen between the months of February and June since Spring and Summer occur in those months. As a result, high instances of GHI will occur between those months due to the lack of cloud formation.

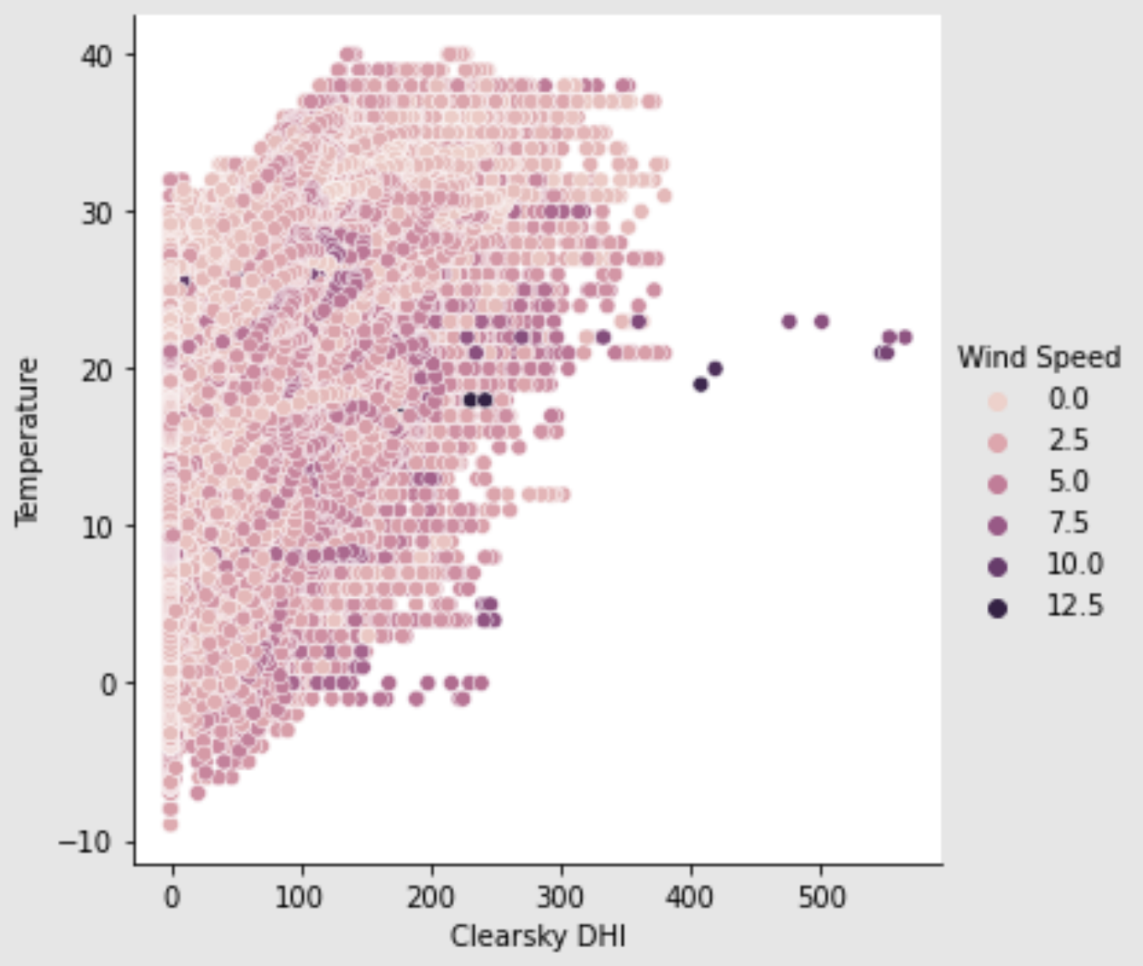

Figure 11: Relationship plot between GHI and the Temperature

Figure 11 has the same values on the x and y axis however the hue now depicts the wind speed not the month. It can be observed that high values of DHI occur when there is a high wind speed and low values of DHI occur when there is a low wind speed. It is important to note that wind speed, temperature, and month were used in these relationship plot because the correlation heat matrix showed a high correlation between them and the labels of the dataset i.e. the values that we want to predict (DNI, GHI, and DHI).

3 Prediction Results

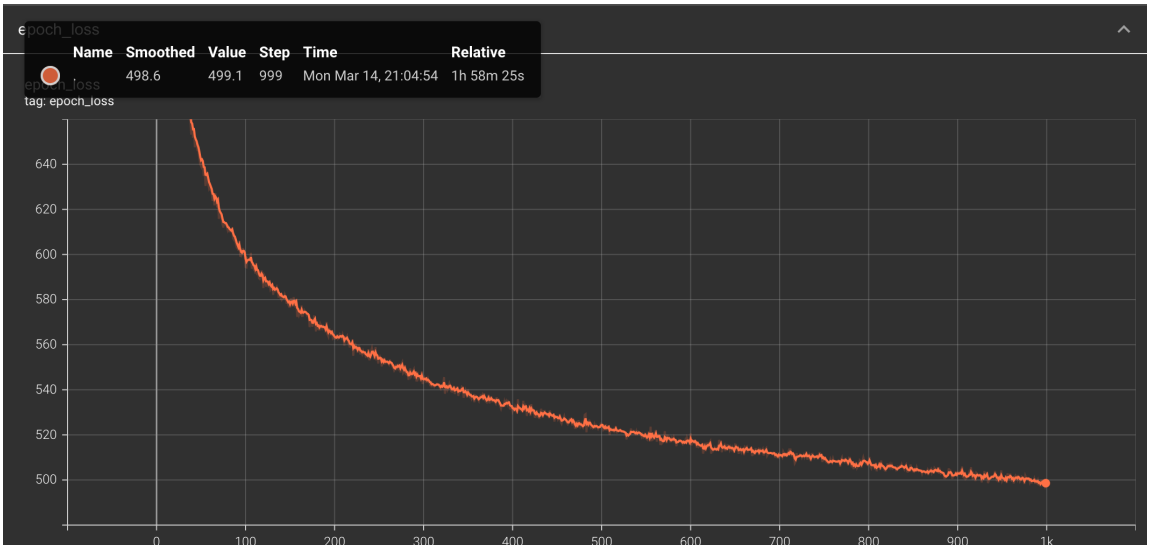

Figure 12: Prediction results on the training set without normalization

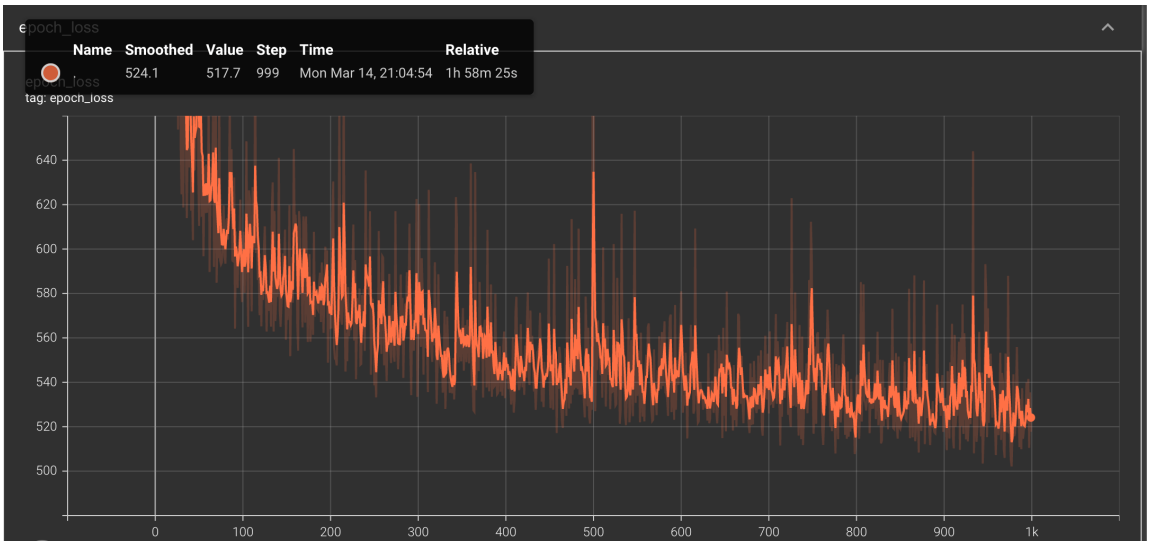

Figure 13: Prediction results on the validation set without normalization

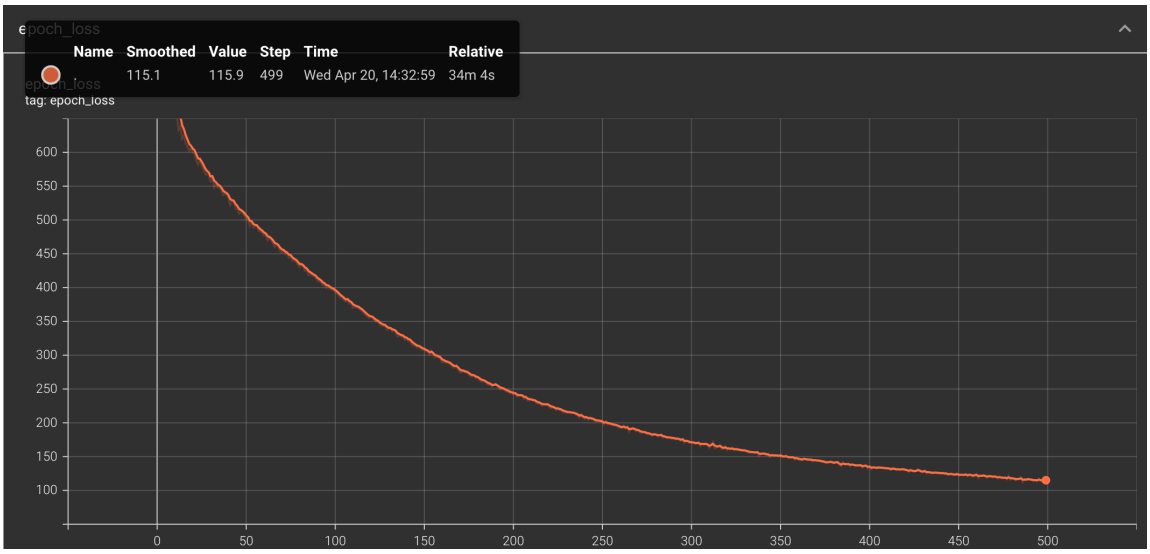

Figure 14: Prediction results on the validation set with normalization

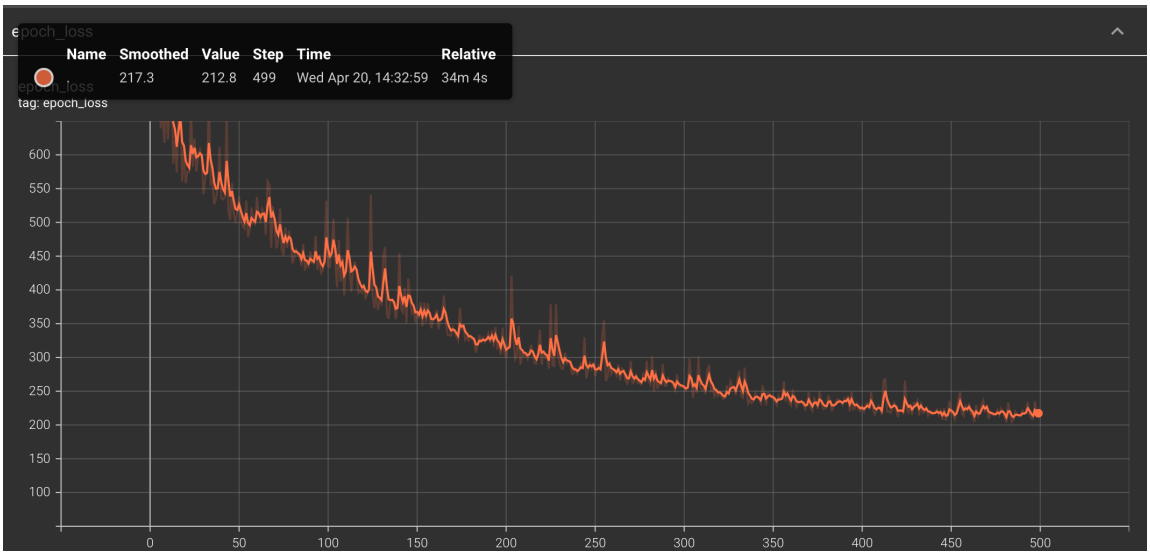

Figure 15: Prediction results on the validation set with normalization

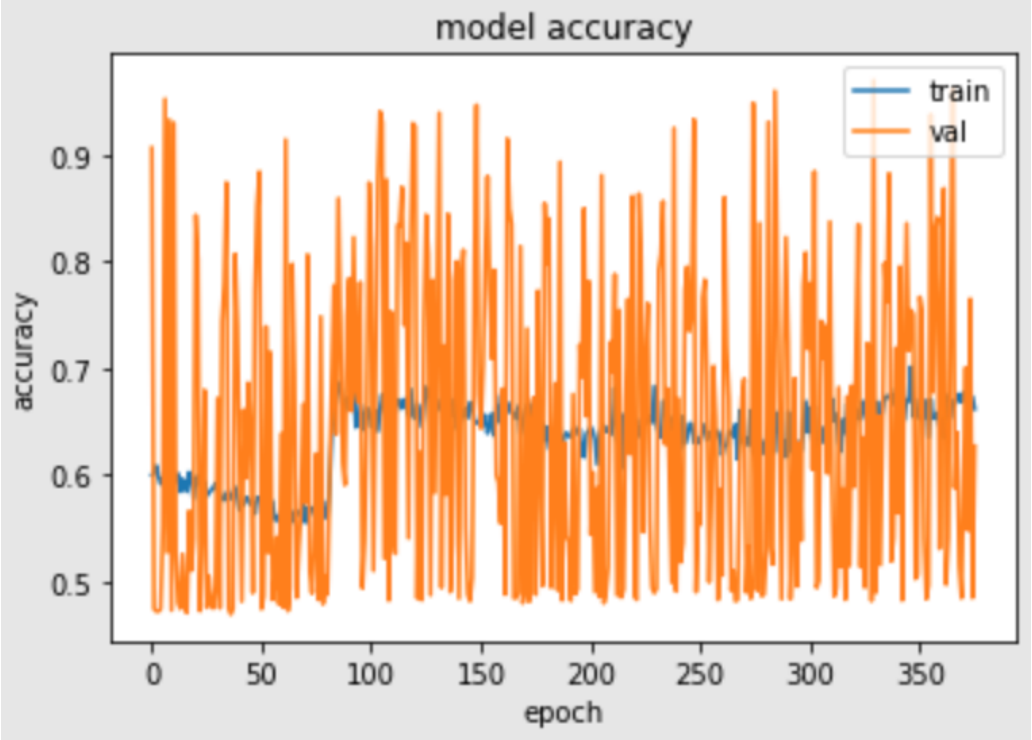

Figure 16: Accuracy on training set and validation set

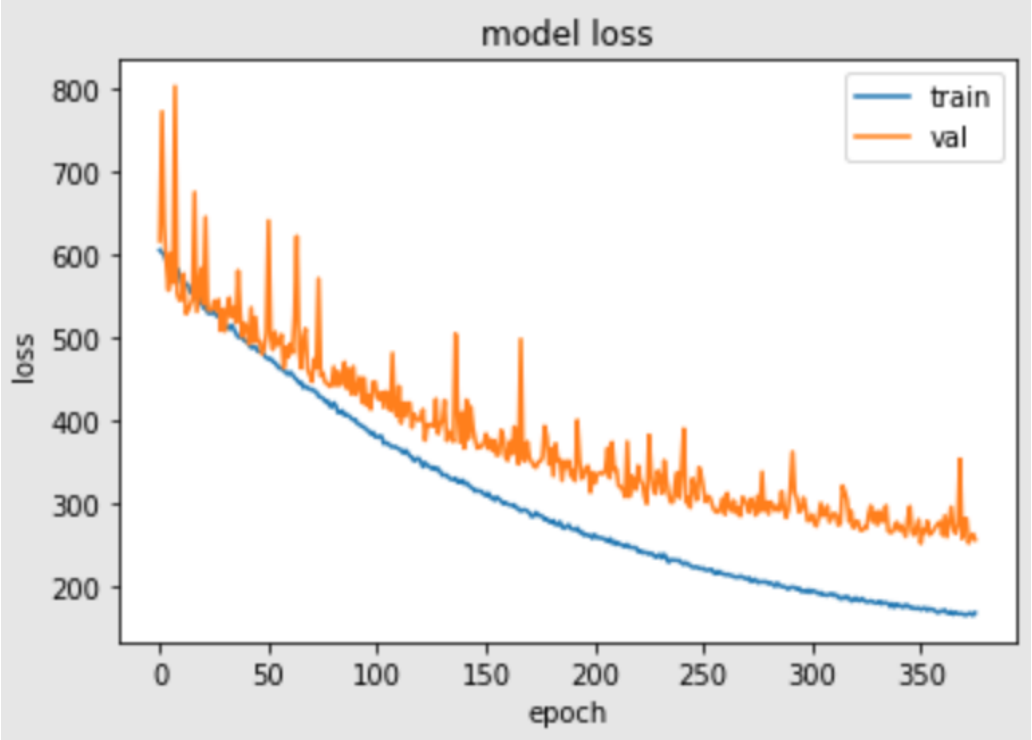

Figure 17: Loss on training set and validation set

4 Results Analysis and Comparisons

4.1 Effects of normalization on results

The results obtained show that the model is well-trained. First of all, figures 8 and 9 emphasize the importance of data normalization. Normalization is the process of rescaling the data into values between 0 and 1. This leads to equal importance between all of the variables. As a result, each variable is given equal weight, and no one can severely affect model performance simply because it is larger. Furthermore, normalization is crucial because it enables the model to understand variables that are measured at different scales. While figure 8 might show that the model is learning and gradually decreasing its error, figure 9 shows the opposite. It confirms that the model did not generalise well to new data due to the constant prediction fluctuations and the fact that the model did not performed better on the training set. Hence, the model does not perform well when given new data. Figures 10 and 11 display how well the model was trained. Figure 11 emphasizes

the importance of normalizing the data. The model is able to reach an MSE of 115 on the training set and 217 on the validation set which is better than the notebook that used this dataset on Kaggle. The validation set experienced much less fluctuations because the data has scaled down to values between 0 and 1. Although, it still experienced minor fluctuations. This is because the model is observing new data so it is prone to make mistakes. Furthermore, the model is exploring different regions in the loss function every time the weights get updated. In the end, when evaluating the model on the test set, the model achieved an MSE of 215 which confirmed that the model was trained well. Since the test set contains sample data not seen by the model, achieving an MSE similar to the validation MSE proves that the model is robust and can generalize well to new data.

4.2 Accuracy and Loss

Accuracy is used to assign a numeric value to how accurate a model predicted its results.x The model simply predicts if something is correct or incorrect. For example, when predicting if an email is spam or not, a well-trained model will have high accuracy i.e. a low percentage of false positives and false negatives. When the email is not spam, the model will output a low accuracy. This is the reason why accuracy is mostly used in classification problems because the output can either be correct or incorrect. Since the current problem is regression, accuracy is not a suitable metric to determine how good the model is performing. In regression, the output is always continuous. As a result, if the model predicts a value of 250, when the actual value is 240, then accuracy cannot help in this situation because the model is attempting to predict a value as close as possible to the actual value. The definite performance of a regression model can only be monitored through the error it achieved. The error is calculated by subtracting the difference between the actual value and the predicted value. Thus, the smaller the difference, the smaller the error, the more accurate the model is. Figure 12 shows the performance of the regression model when accuracy is used as a metric. It can be seen clearly that accuracy cannot be used a metric for regression models. On the other hand, figure 13 depicts the loss of the model. In this case, the loss is assigned to MSE. The model performed extremely well on both the training and validation sets. Through the usage of MSE as a loss function, it can clearly be seen that as the epochs increased, the model predicted more accurate results i.e. it predicted results closer to the actual results hence, the error decreased. One thing to note is that figure 13 is the same as figure 10 and 11 but displayed on one graph.

5 Limitations

This project experienced several limitations. First of all, the current model was supposed to be trained using several machine learning algorithms other than a neural network such as randomforest and XGBoost. The results of each ML algorithm were supposed to be compared to observe which one had the highest performance and the most accurate results. Unfortuneately, this did not work out at all because using randomforest on a large dataset like the one at hand takes a lot of time to train and due to Lebanon's economic crisis the electricity is unstable. As a result, the internet would stop working in the middle of a training session which would lead to it to forcefully stop. Note that due to the current dataset being large, the training session takes up to 3 hours. Since Google Colab was being used to train the model, a stable internet connection had to be maintained.

6 Concluding Remarks

In conclusion, the objectives stated at the beginning of this report were achieved. The model was able to properly utilize thee features it was given to accurately predict the labels. Furthermore, the achieved MSE was very accurate and comparable to the only notebook that used this very dataset. In the future, continuing this project in a place that has a stable internet connection will allow different machine learning algorithms too be implemented on this model. This project greatly enhanced my machine learning knowledge and motivated me to perform more projects or research regarding machine learning.

References