APPLICATIONS OF MACHINE LEARNING IN HYPERLOCAL TEMPERATURE PREDICTION

by

SULAIMAN OWODUNNI

(Under the Direction of John Miller)

ABSTRACT

Weather prediction or forecasting is a science that has existed for a long time. There have been different methods recorded for these predictions with the use of tools like Numerical Weather Prediction. In the scientific community, there is a growing use of Machine Learning for this type of prediction and its ability to support large datasets also makes it a good alternative. We ran different machine learning models with the same dataset using different tools/frameworks which are Scalation, Scikit-Learn and Statsmodels. The result of this analysis is taken and compared against each other based on sample size, accuracy, machine learning models and the performance of the tools with respect to each other. The features or variables in the data are ranked to determine their level of importance when it comes to weather prediction. Some of the features present are temperature from weather stations, vegetation, and the density of man-made buildings. The accuracy is determined by the R^2, MAE and RMSE values, respectively. With the machine learning models used the best accuracy was achieved with Random Forest and the tool with the fastest computation time was Scikit-Learn.

INDEX WORDS: Weather forecasting, Machine learning, Feature importance, Regression.

APPLICATIONS OF MACHINE LEARNING IN HYPERLOCAL TEMPERATURE PREDICTION

by

SULAIMAN OWODUNNI

BS, Babcock University, Nigeria, 2017

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2023

APPLICATIONS OF MACHINE LEARNING IN HYPERLOCAL TEMPERATURE PREDICTION

by

SULAIMAN OWODUNNI

Major Professor:	John A Miller
Committee:	Lakshmish Ramaswamy
	Deepak Mishra

Electronic Version Approved:

Ron Walcott
Vice Provost for Graduate Education and Dean of the Graduate School
The University of Georgia
May 2023

DEDICATION

To MY MOM and DAD

ACKNOWLEDGEMENTS

Firstly, I would like to Thank God for seeing me through to this point. I would also, like to extend heartfelt gratitude towards my advisor, Dr. John A. Miller he has been very encouraging to me in my academics throughout my time in my program. I would like to thank Dr. Lakshmish Ramaswamy and Dr. Deepak Mishra for being part of my committee. I would also like to appreciate them for their individual constructive feedback, guidance, and insights towards helping me.

A special thanks to all the members of Dr Miller's lab for their interesting discussions throughout my research work. I would also like to thank my friend Okunoye Adetayo for contributing to different aspects of my thesis by interesting and inspirational discussions. Thanks to my family and friends for their support.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

CHAPTER 1

INTRODUCTION

Weather analysis is a field of study that has gained substantial traction over the decades. This analysis is usually carried out in terms of forecasting or prediction [1]. When we look at the concept of weather forecasting, which has been extensively researched, it involves the use either Numerical Weather Prediction (NWP) or Statistical Methods. NWP involves many calculations made rapidly using supercomputers because it requires a huge number of arithmetic operations for its computation. One of the best-known NWP models is the Global Forecast System (GFS). On the other hand, in statistical methods, they use data from the past to predict the future weather majorly with Time series models [2]. Due to the wide availability of massive weather observation data, the challenges in forecasting are learning the relationships between different weather observation variables and building a robust prediction model to exploit the hidden patterns in the datasets. It also requires gathering as much data as possible. While in weather prediction, the use machine learning has been growing with more studies planned for the future.

For prediction of the weather, it is important to know the features or variable that are essential. The use of NWP has always been the go-to mode but due to the wide availability of weather data (big data), the challenges in forecasting are learning the relationships between different weather observation variables, then using the knowledge to build a model for prediction and analysis of the datasets.

For certain types of weather prediction, some of the studies use machine learning tools such as deep learning techniques that involves the use of different types of neural networks like Multi-Layer Perceptron (MLP), Long Short-Term Memory Network (LSTM), possibly in

combination with Convolutional Neural Network (CNN). For prediction, the features found in the datasets of other studies often include temperature, wind speed, humidity, rainfall, and dewpoint. Finally, machine learning has been used most frequently for nowcasting and hyperlocal forecasting.

There have been very few hyperlocal based research projects when it comes to weather prediction for a location. In this study the goal is to predict the weather temperature using various Machine learning methods. We explore the effect of different types of regression models for the prediction of the weather for hyperlocal forecasting. We also carry out the research with different features in the dataset compared to what has been used in other studies and perform the analysis using regression models. Regression models show the relationships between variables observed in the data. The regression techniques used include Linear Regression, Random Forest Regression, Quadratic Regression, Gradient Boosting, and Decision Tree Regression. The results from all the techniques are compared and used to determine the effect of all the features present in the dataset. We use three different data science / machine learning frameworks to perform a comparative study of the results for each framework. The frameworks are Scalation, Scikit-Learn, and Statsmodels.

The rest of the paper is organized as follows: Chapter 2 discusses the Related Work on weather prediction, feature importance, and the different types of importance such as permutation and Shapley Additive explanations (SHAP). For Chapter 3, here we talk more about the methodology and architecture for the research study in terms of the evaluation and comparison of models with a focus on the features used and the modeling techniques used by all the tools being compared. The performance of the Scalation will be determined in the experiments and the results will be compared to the other frameworks with modern day alternatives, while also determining the best modelling techniques to be used for this based on this research study in

Chapter 4. Chapter 5 covers further findings and finally, contributions of this work, conclusions, and future work, are given in Chapter 6.

CHAPTER 2

LITERATURE REVIEW

There is substantial research work in the literature on weather prediction. These studies show that the use Machine Learning (ML) techniques for weather prediction have been achieving better performance compared traditional statistical methods [3]– [6]. These modern weather forecasting techniques involve a combination of computational models, observation and knowledge of weather trends and patterns. Popular models for the predictions are Multiple Linear Regression, Deep Learning, CNN, and LSTM based approaches.

A technique based on the exponential smoothing method to accurately predict temperature using historical values was proposed [7]. Their proposed method shows good performance in capturing the seasonal variability of temperature.

SARIMA (Seasonal Autoregressive Integrated Moving Average) techniques were used to predict the monthly mean air temperature in Nanjing city of China from 1951–2017 [8]. The forecasting accuracy of the SARIMA model was acceptable for most practical purposes. we could see that the daily temperature between 1980–2010 for four different European cities was forecasted [9]. Box-Jenkins and Holt-Winters seasonal auto regressive integrated moving average to forecast the future temperature values was another method used [9]. Holt-Winters is a way model and predict the sequence of values over a time value (time series) [10]. Similarly, Box-Jenkins is a model used to forecast based on inputs from a specified time series and it can analyze different types of time series data [11].

In addition to ground-based weather station data, satellite data were also explored to estimate temperature and analyze other atmospheric events [12] [13]. The implementation of several neural network architectures to predict the hourly air temperature for up to 24 hours in the Ararat valley of Armenia [12]. Therefore, statistical time series techniques and neural networks offer stable frameworks for forecasting ground-based air temperature.

A study was done on accurately predicting sea-surface temperature weeks to months is an important step toward long-term weather forecasting [14]. They claimed that standard atmosphere-ocean coupled numerical models provide accurate sea surface forecasts on the scale of a few days to a few weeks, but many important weather systems require greater foresight. They proposed machine-learning approaches to sea-surface temperature forecasting that is accurate on the scale of dozens of weeks. The approach is based on Koopman operator theory, a useful tool for dynamical systems modeling. With this approach, they predicted sea surface temperature in the Gulf of Mexico up to 180 days into the future based on a present image of thermal conditions and three years of historical training data.

Another study states that it is important to know why a model makes a certain prediction [15]. This is as important as the prediction's accuracy in the machine learning model. However, accuracy and interpretability are very important, especially when dealing with huge datasets. In response, various methods have recently been proposed to help users interpret the predictions of complex models, but it is often unclear how these methods are related and when one method is preferable over another, SHAP was introduced. SHAP assigns each feature an importance value for a particular prediction. Its novel components include (a) the identification of a new class of additive feature importance measures, and (b) theoretical results showing there is a unique solution in this class with a set of desirable properties. The new class unifies six existing methods, notable because several recent methods in the class lack the proposed desirable

properties. Based on insights from this unification, new methods show improved computational performance and/or better consistency with human intuition than previous approaches [15].

Research on Debiased MDI (Mean Decrease in Impurity) feature importance measure for random forests [16] has focused on the feature selection bias of MDI from both theoretical and methodological perspectives. Based on the original definition of MDI for a single tree, they derive a tight non-asymptotic bound on the expected bias of MDI importance of noisy features, showing that deep trees have higher (expected) feature selection bias than shallow ones [17]. However, it is not clear how to reduce the bias of MDI using its existing analytical expression. They derived a new analytical expression for MDI, and based on this new expression, they were able to propose a new MDI feature importance measure using out-of-bag samples, called MDI-oob. For both the simulated data and a genomic (chromatin immunoprecipitation) ChIP dataset, MDI-oob achieves state-of-the-art performance in feature selection from Random Forests for both deep and shallow trees.

In addition to MDI [15], [18], some other feature importance measures have been studied in the literature and used in practice:

- Split count, namely, the number of times a feature is used to split [19], can be used as a feature importance measure. This method has been studied in [20], [21] and is available in Extreme Gradient Boosting (XGBoost) [22].
- Mean decrease in accuracy (MDA) measures a feature's importance by the reduction in the model's accuracy after randomly permuting the values of a feature. The motivation of MDA is that permuting an important feature would result in a large decrease in accuracy while permuting an unimportant feature would have a negligible effect. Different permutation choices have been studied [20], [23].

According to a study, there have been various approaches to Sea Surface Temperature (SST) forecasting with physics-agnostic machine learning models [14]. Generic feed-forward neural network (NN) approaches to SST forecasting have been developed for at least two decades [24]. With the advent of deep learning techniques and hardware this decade [25], deeper and more complex architectures have been used. In recent years, Long-Short Term Memory [26]– [28] and Autoencoder [29] architectures have been popular. Another interesting approach has been to learn to model the error of numerical models with a neural network [30]. Application of NNs to physics and Earth science problems generally has a long history [15], [25], [31], as an abundance of data and difficulty in accurately modeling certain systems has made the approach appealing.

An expressly physics-informed approach to machine learning, however, was pioneered by [31], [32], who utilized neural networks (NNs) to model partial differential equations describing physical systems. Studies have also been done using data with different features or similar features as used in this research study where the goal was to determine the hotspot based on the temperature and determine the cause of this through regression analysis.

CHAPTER 3

EVALUATION AND COMPARISON OF MODELS

This chapter covers the methodology used in the collection, preprocessing, and analysis of the data for weather prediction. There have been many weather conditions and they mostly follow a linear trend [33]. We take into consideration the effect of all the features available for determining the weather based on the data and all external factors. In this study, we are focusing on predicting the weather based on the data record from several moving buses in the city of Athens, Georgia over a 6 month period. We compare the recorded data against the nearest weather station and other features to predict the weather based on the data available. The data recorded has different features which will be used in the regression analysis in predicting the weather. Also, this chapter discusses the different evaluation methodologies that are going to be used.

3.1 Data Management Procedure

The data that are to be used for prediction is obtained from devices attached to several buses in the city of Athens which record different features, and these features are later cleaned, transposed, and thoroughly checked for any discrepancy that may be in the record. The other aspect of the data used is obtained from the weather stations for that period. These data have already been prerecorded and available for usage during this study.

Data Preprocessing at the early stage of any big data system's lifecycle improves and refines the quality of the data [34]. Data Pre-Processing enhances the accuracy, efficiency, and Scalability of the classification and prediction models by applying the following sub-processes: data consolidation and integration, data reduction, data discretization, and data cleansing. This is when all the different data are brought together into one place and stored in a table-like or structured dataset for easy accessibility and reconciliation. Along the line, other fields were created based on existing ones just to allow for the ease of data manipulation.

In Machine learning, something that is always considered is building computational models with high prediction and generalization capabilities [35]. This process is used to reduce or minimize the amount of data that is to be processed. By doing this it allows the data to be processed efficiently and the cost of storage reduced, thereby increasing the rate of processing the data into the proposed system. Also, in the process of sampling, using things like the interquartile range allows an easy understanding of the data by showing the ranges of the most variables.

3.2 Data Cleaning

Data cleaning is the process of recognizing unfinished, unreliable, inaccurate, or nonrelevant parts of the data and restoring, remodeling, or removing the dirty data. Data cleaning is an essential part of data mining techniques that involve machine learning [36]. Machine learning (ML) is all about training and feeding data to algorithms to perform various compute-intensive tasks. Data cleaning involves the following steps:

1. Removing unwanted or duplicate records that might have occurred during data consolidation or recording.

2. Fixing any kind of errors like typos or structural errors.
3. Checking for outliers and determining how to handle them. Outliers are unusual values in your dataset, and they can distort statistical analyses and their assumptions. Deciding how to handle outliers depends on investigating their underlying cause.
4. Also, it is necessary to handle missing data. In this scenario, if it is not possible to trace the cause of the data loss then it is best to discard it if it is of no major consequence or effect. Alternatively, one may impute values for missing data.

It must be kept in mind that all the data cleaning also has an effect on the features available in the dataset.

3.3 Exploratory Data Analysis (EDA)

Exploratory data analysis is necessary for research analysis. The main goal is to analyze the data and detect any anomalies that may occur during the usage of data. EDA is expected to take place in the rudimentary stages of research right after data collection [37]. Things that take place in the EDA stage include, checking for outliers, testing assumptions, visual representation of the data and all the individual columns, and a correlation matrix that shows all the different relationships between all the features in the data. Most EDA techniques are visually representative with a few quantitative techniques [38]. With the correlation matrix, we are one step closer in the direction of determining which features are important and what effect they could have on the predictive analysis. The feature importance or selection is one of the major aspects of EDA.

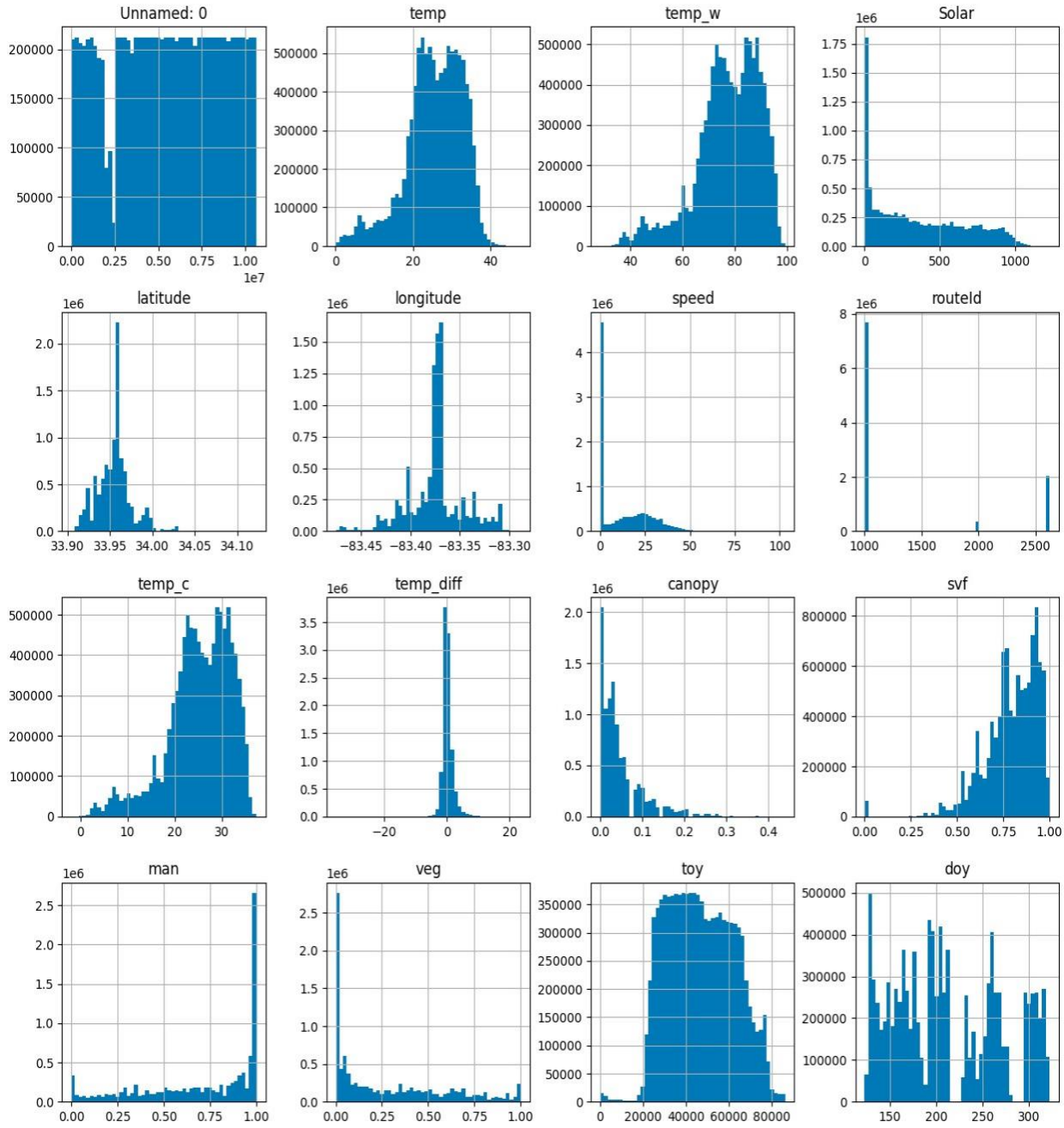


Figure 3.1 Histogram of features relevant to regression analysis

Figure 3.1 is a histogram display of all the features in the dataset. Showing their respective relationship.

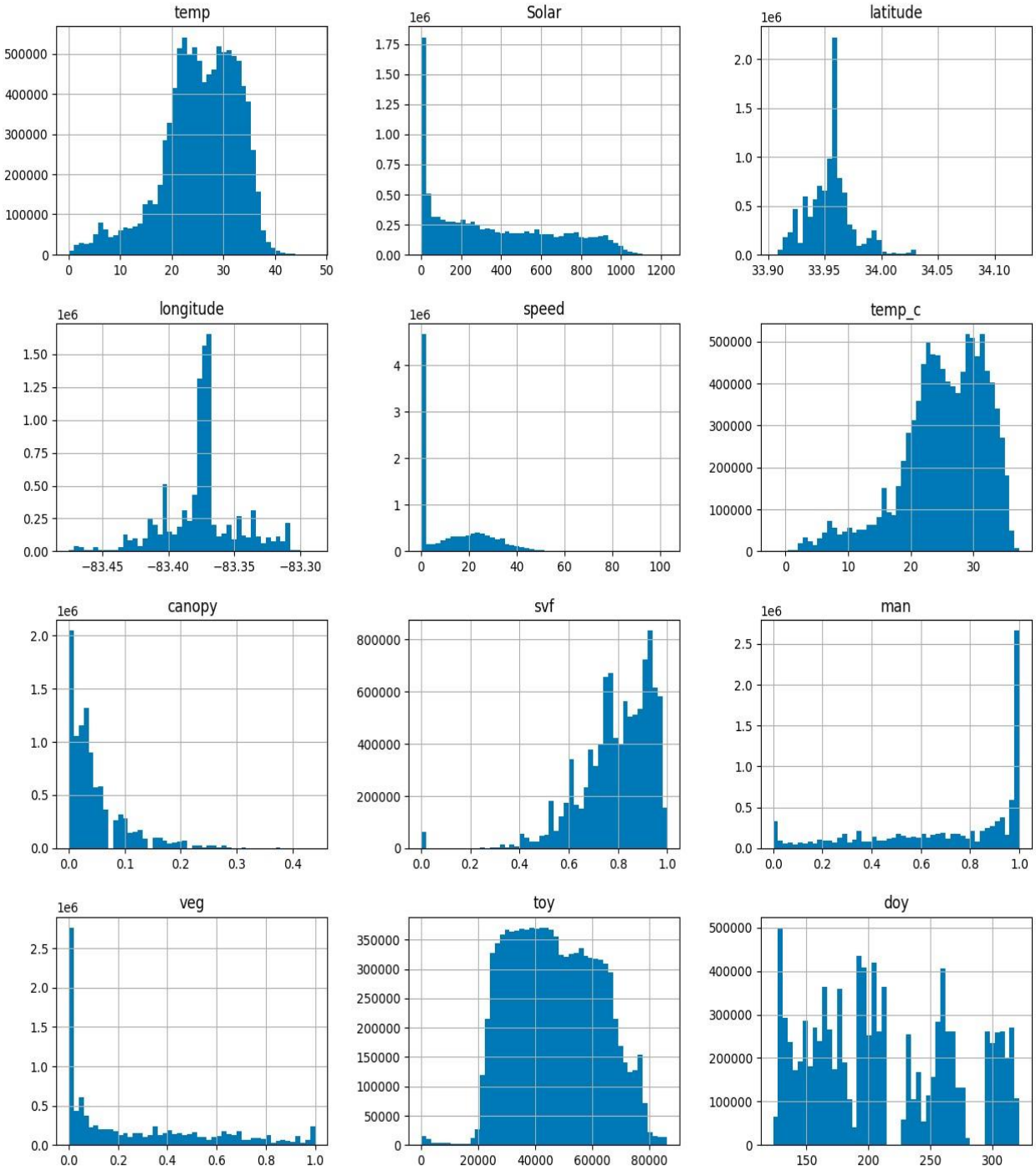


Figure 3.2 Histogram of features relevant to regression analysis

Figure 3.2 is a histogram display of all the features used in the regression analysis.

3.3.1 Correlation Matrix

This is one of the major steps in EDA, especially with large datasets. It is a technique to determine the degree of relationship between one variable or feature and another variable or feature in a dataset and it shows the possible outcomes from the relationships noticed. The matrix is square and symmetrical meaning the resulting figure or diagram is in form of a table that has N x N (N is the number of columns) rows and columns and the matrix is equal to its transpose. The values in the correlation matrix should always lie between +1 and -1. The closer to zero the value of the cell is the more independent the features are of each other. There is a diagonal 1.0 line going from the top left to the bottom right of the table [40].

The fact that there are positive and negative correlations implies that is linear but in opposite directions of each other. There are different types of correlation matrices namely Pearson, Spellman, and Kendall. These three are the majorly used and recognized correlation matrixes.

3.3.1.1 Pearson Correlation

In this correlation r, is the linear relationship between two variables in the dataset if they are normally distributed (basically they have a bell-shaped curve).

Calculate the Pearson r correlation:

$$r = \frac{\sum(x_i - \bar{x})\,(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2\,(y_i - \bar{y})^2}}$$

- r = correlation coefficient
- x_i = Values of the i-th x-variable in a sample
- $\bar{x}$ = Mean of the values of the i-th x- variable
- y_i = Values of the y-variable in a sample
- $\bar{y}$ = Mean of the values of the y- variable

Pearson's correlation coefficient $= \frac{Covariance(X,Y)}{(stdv(X)\ x\ stdv(Y))}$

X	temp	Solar	latitude	longitude	speed	temp_c	canopy	svf	man	veg	tod	doy
temp	1	0.5449	0.0754	-0.3128	-0.0171	0.9802	0.0877	-0.1683	0.0304	-0.0175	0.4689	-0.7055
Solar	0.5449	1	0.1482	-0.2342	0.0101	0.4933	0.092	-0.1135	0.0398	-0.0377	0.1541	-0.2873
latitude	0.0754	0.1482	1	-0.1146	-0.045	0.0492	-0.1191	-0.0484	0.5344	-0.5374	-0.0813	-0.0721
longitude	-0.3128	-0.2342	-0.1146	1	-0.0782	-0.3204	-0.3681	0.3418	0.1355	-0.1324	0.0837	0.4194
speed	-0.0171	0.0101	-0.045	-0.0782	1	-0.0117	-0.0197	-0.096	-0.1626	0.1639	-0.0167	0.0043
temp_c	0.9802	0.4933	0.0492	-0.3204	-0.0117	1	0.0974	-0.1773	0.0063	0.0071	0.4959	-0.7259
canopy	0.0877	0.092	-0.1191	-0.3681	-0.0197	0.0974	1	-0.4848	-0.4378	0.4325	-0.0553	-0.1482
svf	-0.1683	-0.1135	-0.0484	0.3418	-0.096	-0.1773	-0.4848	1	0.2677	-0.2606	0.0359	0.2358
man	0.0304	0.0398	0.5344	0.1355	-0.1626	0.0063	-0.4378	0.2677	1	-0.9923	-0.0379	-0.0223
veg	-0.0175	-0.0377	-0.5374	-0.1324	0.1639	0.0071	0.4325	-0.2606	-0.9923	1	0.0365	0.0025
tod	0.4689	0.1541	-0.0813	0.0837	-0.0167	0.4959	-0.0553	0.0359	-0.0379	0.0365	1	0.0749
doy	-0.7055	-0.2873	-0.0721	0.4194	0.0043	-0.7259	-0.1482	0.2358	-0.0223	0.0025	0.0749	1

Table 3.2: Pearson Correlation matrix of the full dataset of 10 million records

When looking at the Table 3.2, there is a cell with its value close to one. The negative sign implies the vector of both variables are in opposite directions. The closer to 1 the more dependent the variables are of each other.

3.3.2 Feature Importance

The features in the dataset before EDA has been completed “temp”, “temp_w”, “Solar”, “latitude”, “longitude”, “speed”, “lastUpdate”, “Time”, “routeId”, “temp_c”, “temp_diff”, “geometry”, “canopy”, “svf”, “man”, “veg”. They are all that was recorded from the bus except for the temp_c and the temp_diff which were added during the EDA data consolidation process. The features and their means are shown in Table 3.3 below.

Feature	Data	Meaning
“Temp,”	+25.06	Actual/measured temperature y (in degrees Celsius)
“temp_w,”	70.35	Temperature nearest weather station (in Fahrenheit)
“Solar,”	101	solar radiation (in watts per meter^2 reaching surface)
“Latitude,”	33.93921	(In degrees North of the Equator)
“Longitude,”	-83.38663	(In degrees West of Greenwich, UK)
“Speed,”	9	wind speed (in mph)
“lastUpdate,”	5/3/2019 10:15	date and time
“Time,”	5/3/2019 10:15	date and time
“routeId,”	2611	bus route id
“temp_c,”	21.30555556	Weather station temperature in Celsius
“temp_diff,”	3.754444444	this is y, the difference between temp and temp_c
“Geometry,”	POINT (279418.8026540396 3757981.575102321)	X, y coordinates distance from some origin becomes separate columns/features or ignored
“Canopy,”	0.05	percent covered by trees
“svf,”	0.86040455	Sky View Factor (in percent of sky visible at location)
“Man,”	1	percent man-made surface
“veg”	0	percent covered by vegetation
“doy”	365	Day of the year
“tod”	503623	time of the day

Table 3.3: List of all the features /columns in the dataset

Since one of the goals of this study is to predict the temperature, that would therefore make “temp” feature the most important. The remaining features present in the data will be used in conjunction with each other to predict the result. From the data available some of the features would have no use in their current state so they can either be converted to something else or completely remove. Below is a table showing all the features that are considered in the process of running the various models.

FEATURE	DATA
temp	+25.06
temp_w	70.35
solar	101
latitude	33.93921
longitude	-83.38663
speed	9
lastupdate	5/3/2019 10:15
time	5/3/2019 10:15
temp_c	21.30555556
canopy	0.05
svf	0.86040455
man	1
veg	0
doy	301
tod	50363

Table 3.4: List of all the columns/ features used in the models.

In Table 3.4, some features have been removed such as the “lastUpdate”, “Time”, “geometry”, and “temp_diff”. The three features were removed because it has no effect in the format that was and there is no means to read them as it is a string data type, and the last feature was removed seeing as it is a calculation of the temperature recorded and the temperature

recorded from the weather station. The “lastUpdate” column is however split into two columns namely the Time of Day “Tod” and Day of the Year “Doy”.

Another thing that was considered is the interquartile range of the most important feature which is the variable we are attempting to predict “Temp”. From the dataset, we got the maximum and minimum of the feature and then also got the interquartile range from 0.25-0.75. then we calculate the R^2 of each quartile and sum it up.

FEATURE	MAXIMUM	MINIMUM
temp	47.75000	0.06000
solar	1230.00000	0.00000
latitude	34.12194	33.90443
longitude	-83.28856	-83.47547
speed	103.00000	0.00000
temp_c	37.33333	-1.94444
canopy	0.44000	0.00000
svf	1.00000	0.00000
man	1.00000	0.00000
veg	1.00000	0.00000
tod	86339	0.00000
doy	322.00000	123.00000

Table 3.5: L-R, showing the minimum and maximum values of the individual features.

3.3.3 Variance Inflation Factor (VIF)

This is a tool used to measure the degree of multicollinearity of each factor [41], [42]. It could also be the amount of multicollinearity in a set of multiple regression variables. Multicollinearity occurs or exists when more than one predictor is correlated with one another. In this study, we performed analysis to determine the VIF based on the features used for prediction [41], [42]. When a feature has just a VIF value of 1, it implies there is no correlation between the features. If the value is greater than 10, it should be further studied. But in cases where the value exceeds 20, this shows high signs of multicollinearity [43].

	Feature	**Statsmodels VIF**	**Scalation VIF**
0	const	1.18E+07	NA
1	Solar	1.332850	1.332850
2	latitude	1.077040	1.077040
3	longitude	1.012665	1.012665
4	speed	1.201821	1.201821
5	temp_c	1.767113	1.767113
6	canopy	1.285954	1.285954
7	svf	1.335391	1.335391
8	man	13.597900	13.597900
9	veg	13.291810	13.291810
10	tod	1.199963	1.199963
11	doy	1.220459	1.220459

Table 3.1 The VIF values of all predictor features

All the features have values that inform us that there is no correlation among them except for two namely the “man” and the “veg” features. If you recall the correlation matrix for the dataset also shows that there is a high level of multilinearity between the “man” and “veg” columns of the data. The VIF values shown above also back up the claim. In Table 3.5, we also see that across both tools used in checking the VIF, the resulting values are equal.

3.4 Regression Model

Regression analysis in machine learning is a technique used as a model to analyze the relationship between the target feature otherwise known as the dependent variable or feature and the independent variables or remaining features of the dataset [44]. There are different algorithms or techniques for regression like Linear regression, Ridge, Lasso, Decision Tree, and Random Forest [45]. The goal of this technique could be to determine the predictor strength, forecast trend, feature importance, and time series.

3.4.1 Linear Regression Model

In this study, we would analyze simple linear regression and multiple linear regression. It consists of a predictor variable and a dependent variable related linearly [44]. The predictor error is the value obtained from the difference between the observed values and the predicted value. In this study we used both simple linear and the multiple linear regression. For the simple we ran a regression model with the predictor variable and every other feature, one after the other. While in the multiple we run all the predictor variables against the dependent variable at once. With this, it

is possible to determine the effect or influence of the predictors variables on the response variable which is also known as the feature importance [35].

Simple linear regression model equation.

$$y = \boldsymbol{b} \cdot \boldsymbol{x} + \varepsilon = b_0 + b_1 x + \varepsilon$$

Multiple linear regression model equation.

$$y = \boldsymbol{b} \cdot \boldsymbol{x} + \varepsilon = b_0 + b_1 x_1 + \cdots + b_k x_k + \varepsilon$$

Where:

- y: is the dependent/response variable.
- x: is the independent/predictor variable or features.
- $_k$: Is the number of features used.
- ε: is the error/residual term.

3.4.2 Quadratic Regression Model

This regression model technique involves the combination of linear models with two types of feature interaction [46]. It takes all the independent variables and takes the square of each independent variable and sums it up [47]. This is a bivariate case equation.

$$y = \boldsymbol{b} \cdot \boldsymbol{x} + \varepsilon = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_1^2 + b_4 x_2^2 + b_5 x_1 x_2 + \varepsilon$$

Where:

- $\boldsymbol{x}$ is the independent variable or features.
- y is the dependent or response variable.
- $\boldsymbol{b}$ is the parameter / coefficient vector.

3.4.3 Random Forest Regression Model (RF)

This type of machine learning algorithm falls under supervised learning. It involves the use of ensemble learning for regression analysis [48]. RF is a collection of decision trees each with its nodes and different data that leads to unique leaves. The algorithm is good for both regression and classification and is fast to train. However, RF may have substantial overfitting [49].

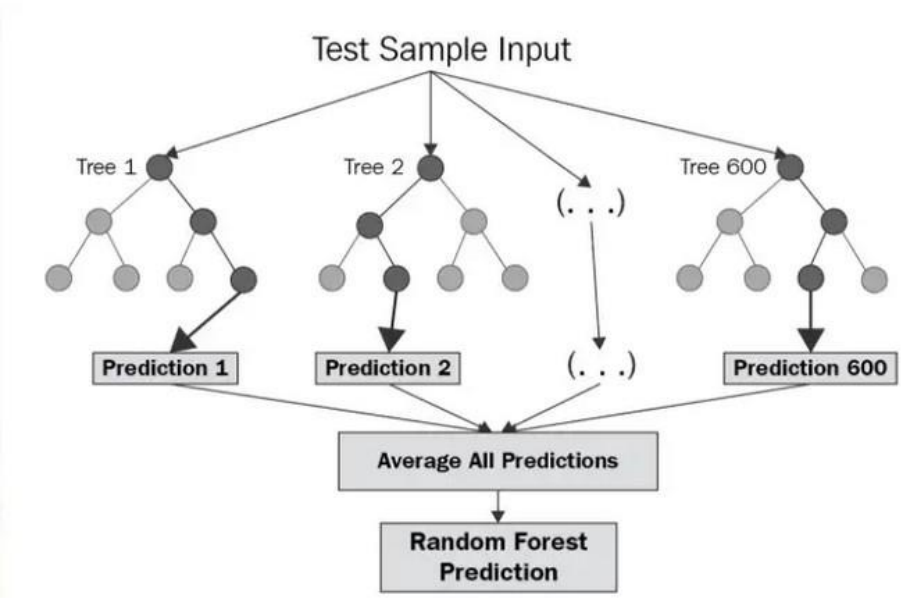


Figure 3.3 Random Forest tree split.

3.5 Model Evaluation and Result

A model is chosen and there are different criteria for each model. For any model from the above listed, we find the R-Squared of each of the features and predict the temperature based on all the features together. Another criterion used was splitting the data into sample sizes due to the volume of the whole dataset. The data sample size (m) used were one hundred Thousand (100,000), 1 million (1,000,000), 4 million (4,000,000), and 10 million (10,000,000). When

predicting based on all features in the sample size, we record the R-Squared (R^2), Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE) which is the square root of MSE and the feature importance for each case. To determine if the results obtained from the different regression models were accurate, we computed the total sum of the squares (SST), Sum of squares due to regression (SSR), and Sum of squares of errors (SSE). SST is a function of the SSR and SSE. In other words, we can say SST is the sum of the square of the difference between the target feature (y-value) and the mean of the target feature (mean of y-value).

$$SST = SSE + SSR$$

$$SST = \sum (y_i - \bar{y})^2$$

With SST, it is possible to determine the R^2, using the formula below. With this, we can attain that all the R^2's is equal to six decimal places.

$$R^2 = 1 - \frac{SSE}{SST}$$

Then in certain cases, we ran the model for 3 types of model validation procedures namely 1) In-sample, which involves using the entire dataset of the sample size selected in the model and determining the R^2, MAE, MSE, and RMSE. 2) Train and Test; in this case, the sample size is split in two in a ratio of choice but in this study, a ratio of 0.75-0.25 is used. The lesser size being the test size and the 0.75 of the sample datasets being used for training. In this case, the regression model fits on the train and predicts the test cases. In this study, the train and test are split in same way every time. 3) Cross-validation with K-fold; the data is split into subsets based on the number of folds suggested and then rotating the training and validation among them. [50]. MAE is the measure the of success of the model.

$$\text{SAE} = \sum |y_i - \hat{y}_i|$$

$$SSE = \sum (y_i - \hat{y}_i)^2$$

$$MAE = \frac{SAE}{m}$$

$$MSE = \frac{SSE}{m}$$

3.5.1 Predictive Tools

To run the model that was chosen in this study we had to pick what framework or programming language they would be from. As of today, some of the best things to consider when picking a tool for a machine learning study are the language, documentation, integrated development environment (IDE), execution speed, training speed and a graphical support [51]. There are different types of Machine learning (ML) namely, supervised, unsupervised, and reinforcement learning [52]. When determining the tool to be used in the study, we had to determine the type of ML we would choose which was concluded to be regression which is supervised learning. There are various languages but, in this study, we chose Python as the programming language and Scala. From these languages, we select a library or platform which are Scikit learn and Statsmodels for Python and Scalation for Scala. Some advantage as to why Python is one of the leading ML tools are it is an object-oriented language it is elegant and concise; it allows efficient execution and independency and finally, it can also be used in the development of large-scale software and has application in a neural network [53].

Scikit learn is a machine learning library that's readily available, open source and has several features most especially regression [54] which is our area of focus. It also has extensive documentation. Statsmodels is an API library that has just enough documentation for usage. Scalation is also chosen to compare it to the widely known ML tools that are all recognized as state-of-the-art libraries. Scalation is a library that has various uses such as database

management, knowledge base modeling, simulation modeling, machine learning modeling like regression and finally, it can be used for Time series.

So, we use these three tools separately to determine the best result possible for the dataset and check the level of the Scalation library in comparison to the Scikit and Statsmodels. By using these tools, we check to see if in the same condition we can replicate the same result across all test cases and reach a definitive conclusion on the importance of the feature, the predictive analysis of the data, and forecasting.

3.5.1.1 Scikit Learn (Sklearn)

This library has functions in it that allow for various cases that are to be used in this study such that one can import the library and uses the different methods to achieve the desired model design. In Sklearn, there is a function call for the train and test called "train-test-split" which accepts takes two important arguments mainly the dependent and the independent features that are required for the predictive analysis. Another function used is the cross-validation which takes arguments like the *k-fold*, regression model, and the dependent and independent dataset. Sklearn is such a good library as it can run all the regression techniques selected for this study.

3.5.1.2 Scalation

The Scalation framework is quite extensive in its capabilities. It has similar library functions just like Scikit-learn but the major difference is the language they are written off which in this case is Scala and runs using Simple Built Tool (SBT) compared to the other which uses Python. With this tool, it is also capable of running all the model techniques selected for this

study, and it also can run for various cases such as train and test, and cross-validation. It has a feature for displaying a summary. Some of the variables in this summary include p-value which is term that test the null hypothesis to determine if the coefficient is equal to zero (no effect). If the p-value is < 0.05, then the feature is meaningful in the model. Also, we have t-value which is the calculated difference represented in units of standard error. It measures the size of the difference relative to the variation in the sample data.

```
SUMMARY
    Parameters/Coefficients:
    Var      Estimate    Std. Error       t value         Pr(>|t|)        VIF
-------------------------------------------------------------------------------
    x0    1106.989080     155.217657      7.131850        0.000000        NA
    x1    0.001632        0.000014        117.354865      0.000000        1.362869
    x2    7.118445        0.484587        14.689729       0.000000        1.552540
    x3    16.170860       1.879344        8.604523        0.000000        1.361157
    x4    -0.002513       0.000423        -5.940543       0.000000        1.061933
    x5    0.964321        0.000715        1348.855660     0.000000        1.452905
    x6    -0.147287       0.075520        -1.950309       0.051139        1.652434
    x7    0.094918        0.034748        2.731604        0.006303        1.435868
    x8    0.141834        0.110029        1.289054        0.197379        67.093316
    x9    -0.168245       0.113702        -1.479700       0.138953        66.743799
    Residual standard error: 1.140726 on 99990.0 degrees of freedom
    Multiple R-squared:  0.966375,       Adjusted R-squared:  0.966372
    F-statistic: 319301.0873358779 on 9.0 and 99990.0 DF,  p-value: 0.0
-------------------------------------------------------------------------------
```

Figure 3.4 Summary result of Scalation

Figure 3.5 shows what the summary result looks like in Scalation with different factors like VIF, t-value, p-value, and R-squared.

3.5.1.3 Statsmodels

Its major advantage is its ability to track every kind of error or score when performing a linear regression and it also can summarize the whole model in such a way that all possible

values are easily seen and accessible. Some of the variables in this summary include p-value which is term that test the null hypothesis to determine if the coefficient is equal to zero (no effect). If the p-value is < 0.05, then the feature is meaningful in the model. Also, we have t-value which is the calculated difference represented in units of standard error. It measures the size of the difference relative to the variation in the sample data.

OLS Regression Results

Dep. Variable:	temp	R-squared:	0.966
Model:	OLS	Adj. R-squared:	0.966
Method:	Least Squares	F-statistic:	2.598e+07
Date:	Fri, 02 Dec 2022	Prob (F-statistic):	0.00
Time:	23:55:10	Log-Likelihood:	-1.7290e+07
No. Observations:	10064417	AIC:	3.458e+07
Df Residuals:	10064405	BIC:	3.458e+07
Df Model:	11		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	273.1177	1.459	187.154	0.000	270.258	275.978
Solar	0.0024	1.59e-06	1491.399	0.000	0.002	0.002
latitude	-4.7274	0.023	-208.827	0.000	-4.772	-4.683
longitude	1.3512	0.015	88.145	0.000	1.321	1.381
speed	-0.0120	3.28e-05	-365.842	0.000	-0.012	-0.012
temp_c	1.0080	8.39e-05	1.2e+04	0.000	1.008	1.008
canopy	-0.0107	0.009	-1.183	0.237	-0.028	0.007
svf	0.2973	0.003	88.677	0.000	0.291	0.304
man	-0.0818	0.005	-16.216	0.000	-0.092	-0.072
veg	-0.5612	0.005	-108.235	0.000	-0.571	-0.551
toy	-6.352e-06	3.03e-08	-209.622	0.000	-6.41e-06	-6.29e-06

Figure 3.5 Summary feature of the Statsmodels

Figure 3.6 shows what the summary result looks like in Statsmodels with different factors. like VIF, t-value, p-value, and R-squared. It is very similar to that of Scalation.

Feature	Linear R^2	Quadratic R^2	Random Forest R^2	Gradient Boosting R^2	Decision Tree R^2
Solar	0.307563	0.31619	0.335874	0.060811	0.328305
Lat	0.00179	0.003018	0.335874	0.006657	0.017646
Long	0.000111	0.002076	0.03404	0.005965	0.014989
speed	0.002834	0.004217	0.004417	0.000801	0.004395
Temp_c	0.95606	0.95637	0.95727	0.173599	0.95571
Canopy	0.00053	0.000709	0.002192	0.000397	0.001979
Svf	0.0004	0.001235	0.023567	0.003841	0.008153
Man	0.000469	0.000835	0.006125	0.001105	0.003618
Veg	0.000332	0.000534	0.005476	0.000972	0.003802
DOY	0.163001	0.449083	0.633813	0.114979	0.590484
TOD	0.109189	0.202092	0.260034	0.047067	0.255489

Table 3.2 $R^2$for Each feature in the individual regression model

Table 3.5 shows the R^2 of all the features that are being considered for the prediction analyses of the temperature. The R^2 are compared across different regression model techniques. Looking at the trend in the various R^2 values across all the model it shows that for most of the features, the R^2 is greatest with the Random Forest model while the least values are gotten in the gradient boosting. Also, the table shows that the "Temp_c" feature is the most important because it has the highest value of greater than (> 0.9).

CHAPTER 4

EXPERIMENTS AND RESULTS

For the experiment, as stated earlier, we would be running different models with certain criteria. the criteria are the data splitting procedure (i.e., the train and test splitting) and data sampling. For all the modeling technique used, the test cases selected were based on the following factors such as the procedure (In-sample (IN), Train and Test split (TNT) and cross-validation (CV) with k-fold = 4), Dataset size (100,000, 1,000,000), the type of feature selection to be used and some missing features. Cases were tested across all three tools earlier suggested. In some cases, we were able to test for a larger dataset sample size. This was limited because of the memory that would be required to run the model.

Here are the various cases used for the research.

Case	**Model procedure**	**Dataset size**
case1	IN	100000
case2	IN	1000000
case3	IN	4000000
case4	IN	10000000
case5	TNT	100000
case6	TNT	1000000
case7	TNT	4000000
case8	TNT	10000000

case9	CV	100000
case10	CV	1000000
case11	CV	4000000
case12	CV	10000000

Table 4.1 The different cases that were used during the research across all the techniques.

4.1 Linear Regression

As earlier stated, we run the regression model across the 3 procedures and compare the results of the performance of all tools.

4.1.1 In-sample

We run the model for all the cases in of in-sample.

Model	Data count	R^2		
		SKLEARN	SCALATION	STATSMODEL
Regression (IN)	100000	0.966684	0.966649	0.966684
	1000000	0.946701	0.946346	0.946701
	4000000	0.952086	0.952027	0.952086
	10000000	0.965981	0.965863	0.965981

Table 4.2 R^2 for all cases with In-sample

Model	Data count	MAE		
		SKLEARN	SCALATION	STATSMODEL
Regression (IN)	100000	0.775389	0.774948	0.775389

	1000000	1.136827	1.138689	1.136827
	4000000	0.973337	0.974735	0.973337
	10000000	0.88723	0.888301	0.88723

Table 4.3 MAE for all cases with In-sample

Model	Data count	MSE		
		SKLEARN	SCALATION	STATSMODEL
Regression (IN)	100000	1.289168	1.290398	1.289168
	1000000	2.653883	2.671593	2.653883
	4000000	2.140999	2.143655	2.140999
	10000000	1.818389	1.824718	1.818389

Table 4.4 MSE for all cases with In-sample

Model	Data count	RMSE		
		SKLEARN	SCALATION	STATSMODEL
Regression (IN)	100000	1.135415	1.135957	1.135415
	1000000	1.629074	1.634501	1.629074
	4000000	1.463215	1.464123	1.463215
	10000000	1.348477	1.350821	1.348477

Table 4.5 RMSE for all cases with In-sample

Linear regression is one of the fastest and running the in-sample procedure show that all the tools return the same values of R^2, MAE, MSE, and RMSE. Then we look at the feature importance based on the tool. However, with Statsmodels, other than providing the p-values in the summary table, it is unable to determine the feature importance which serves as a

shortcoming for the tool. We also added a column called “One “which was used as the 1’s column in the predictive analysis.

Data count	**SKLEARN**	**SCALATION**
100,000	Temp, Solar, Veg, Latitude, One, Svf, Speed, Man, Longitude, Canopy	One, Temp, Solar, Man, Latitude, Longitude, Speed, Svf, Canopy, Veg
1,000,000	Temp, Solar, Speed, Svf, One, Canopy, Veg, Latitude, Longitude, Man	One, Temp, Solar, Speed, Svf, Latitude, Longitude, Canopy, Veg, Man

Table 4.6 Feature Importance of Linear regression In-sample

	0.25 Inter Quartile	**0.50 Inter Quartile**	**0.75 Inter Quartile**
R2	0.579048	0.96024	0.395587
MAE	0.730469	0.693425	1.097662
MSE	0.871116	1.211063	2.314713
RMSE	0.933336	1.100483	1.521418

Table 4.7 The result of using the running the model based on their interquartile range.

0.25 Inter Quartile	**0.50 Inter Quartile**	**0.75 Inter Quartile**
Temp_c	Temp_c	Temp_c
solar	solar	tod
speed	speed	speed

man	doy	solar
doy	veg	doy
veg	tod	veg
latitude	man	man
canopy	svf	canopy
svf	canopy	longitude
longitude	latitude	svf
tod	longitude	latitude

Table 4.8 feature importance across the interquartile range

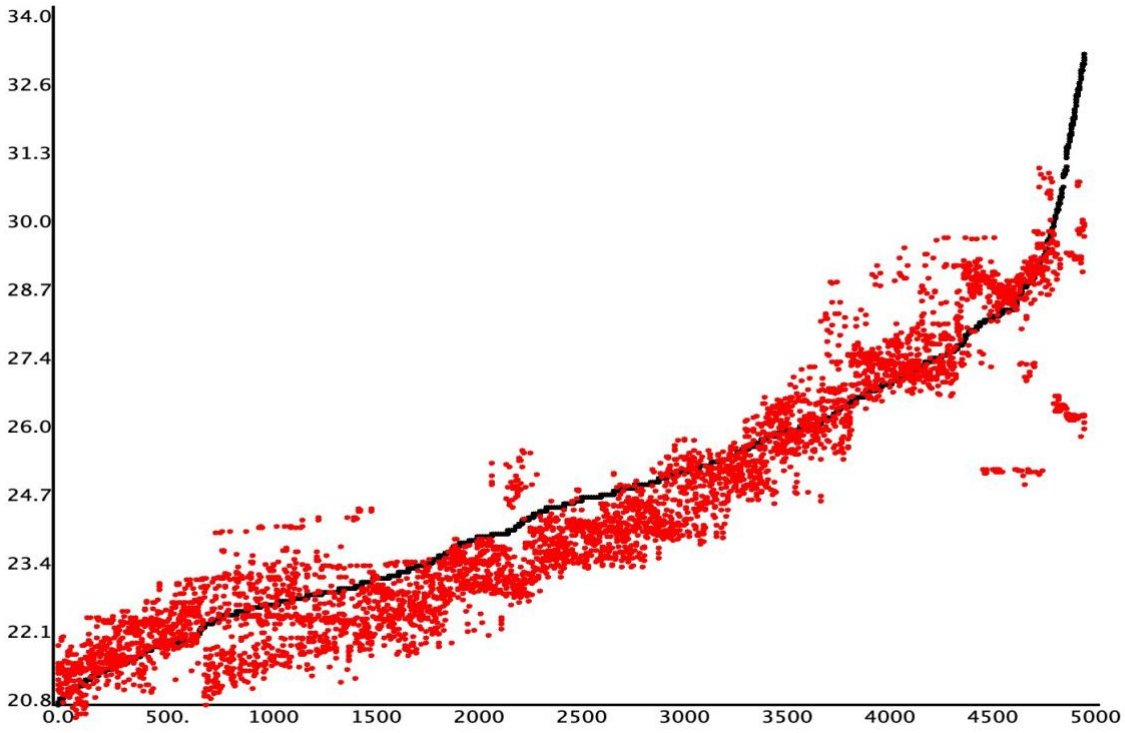


Figure 4.1 Regression data analysis

Figure 4.1 shows the regression of the data for a linear regression analysis with the full dataset.

4.1.2 Train and Test (TNT)

In this procedure, we train 75% of the data size used and test on the remaining 25%. The standard is to no use more than 30-40 % as your test size so the training data would have enough top use to predict.

		R2		
		SKLEARN	SCALATION	STATSMODEL
Regression (TNT)	100000	0.964991	0.966321	0.964991
	1000000	0.946646	0.899554	0.946646
	4000000	0.952005	0.959573	0.952005
	10000000	0.966034	0.945036	0.966034

Table 4.9 R^2 for Some cases with TNT

		MAE		
		SKLEARN	SCALATION	STATSMODEL
Regression (TNT)	100000	0.784825	0.779231	0.784825
	1000000	1.138447	1.315718	1.138447
	4000000	0.973342	0.950430	0.973342
	10000000	0.886692	0.989405	0.886692

Table 4.10 MAE for Some cases with TNT

		MSE

		SKLEARN	SCALATION	STATSMODEL
Regression (TNT)	100000	1. 333445	1.337294	1. 333445
	1000000	2.665112	3.666915	2.665112
	4000000	2.141742	2.327733	2.141742
	10000000	1.814328	2.414300	1.814328

Table 4.11 MSE for Some cases with TNT

		RMSE		
		SKLEARN	SCALATION	STATSMODEL
Regression (TNT)	100000	1.15474	1.156414	1.15474
	1000000	1.632517	1.914919	1.632517
	4000000	1.463469	1.525691	1.463469
	10000000	1.34697	1.553802	1.34697

Table 4.12 RMSE for Some cases with TNT

In this procedure of train and test, we noticed that there is a slight difference between the R^2's, and this can be seen as small since the split algorithm differs a little amongst the tools. Little things like stratifying or the fact that when splitting the data Scikit-Learn takes the first percentage as training and the remaining as a test, but Scalation does the opposite.

4.1.3 Cross Validation

This involves using a k-fold of 4, along with the linear regression model and the datasets for the x and y features. Since we are using a k-fold of 4, the splitting would be repeated 4 times across the simulation process.

		R2		
		SKLEARN	SCALATION	STATSMODEL
Regression (CV)	100000	0.927172	0.966268	
	1000000	0.913969	0.946707	
	4000000	0.947455		
	10000000	0.963228		

Table 4.13 R^2 for Some cases with CV

		MAE		
		SKLEARN	SCALATION	STATSMODEL
Regression (CV)	100000	0.799859	0.774319	
	1000000	1.194478	1.135749	
	4000000	0.981318		
	10000000	0.904477		

Table 4.14 MAE for Some cases with CV

		MSE		
		SKLEARN	SCALATION	STATSMODEL
Regression (CV)	100000	1.368108	1.304071	

	1000000	2.838935	2.659684	
	4000000	2.186107		
	10000000	1.874974		

Table 4.15 MSE for Some cases with CV

		RMSE		
		SKLEARN	SCALATION	STATSMODEL
Regression (CV)	100000	1.169661	1.141959	
	1000000	1.684913	1.630853	
	4000000	1.478548		
	10000000	1.369296		

Table 4.16 RMSE for Some cases with CV

In a bid to determine the feature importance, the data was split into its interquartile range and the resulting R2, MAE, MSE, RMSE, and order of feature importance was determined.

4.2 Quadratic Regression Model

4.2.1 In-sample

		R_2		
		SKLEARN	SCALATION	STATSMODEL
Quadratic (IN)	100000	0.973049	0.969228	0.973057

	1000000	0.951772	0.949685	0.951822
	4000000	0.954259	0.953365	0.954332
	10000000	0.967308	0.966291	0.967369

Table 4.17 R^2 for Some cases with INSAMPLE

		MAE		
		SKLEARN	SCALATION	STATSMODEL
Quadratic (IN)	100000	0.702975	0.757323	0.702457
	1000000	1.084209	1.099085	1.07931
	4000000	0.959556	0.966342	0.957272
	10000000	0.870559	0.884133	0.86845

Table 4.18 MAE for Some cases with INSAMPLE

		MSE		
		SKLEARN	SCALATION	STATSMODEL
Quadratic (IN)	100000	1.042871	1.190753	1.042586
	1000000	2.401387	2.505332	2.398923
	4000000	2.043901	2.083860	2.04066
	10000000	1.747467	1.801854	1.744234

Table 4.19 MSE for Some cases with INSAMPLE

		RMSE		
		SKLEARN	SCALATION	STATSMODEL

Quadratic (IN)	100000	1.021211	1.091216	1.021071
	1000000	1.549641	1.582824	1.548846
	4000000	1.429651	1.443558	1.428517
	10000000	1.321918	1.342331	1.320694

Table 4.20 RMSE for Some cases with INSAMPLE

4.2.2 Train and Test

		R_2		
		SKLEARN	SCALATION	STATSMODEL
Quadratic (TNT)	100000	0.973391	0.969090	0.97295
	1000000	0.952252	0.949366	0.951737
	4000000	0.953803	0.953472	0.954404
	10000000	0.967238	0.966645	0.967375

Table 4.21 R^2 for Some cases with TNT

		MAE		
		SKLEARN	SCALATION	STATSMODEL
Quadratic (TNT)	100000	0.69906	0.762410	0.699442
	1000000	1.078209	1.100023	1.079303
	4000000	0.963404	0.965666	0.959085
	10000000	0.871815	0.882998	0.868398

Table 4.22 MAE for Some cases with TNT

		MSE		
		SKLEARN	SCALATION	STATSMODEL
Quadratic (TNT)	100000	1.029404	1.227341	1.030893
	1000000	2.394409	2.514301	2.403534
	4000000	2.063146	2.068815	2.049233
	10000000	1.752139	1.794008	1.746152

Table 4.23 MSE for Some cases with TNT

		RMSE		
		SKLEARN	SCALATION	STATSMODEL
Quadratic (TNT)	100000	1.014596	1.107854	1.015329
	1000000	1.547388	1.585655	1.550334
	4000000	1.436366	1.438337	1.431514
	10000000	1.323684	1.339406	1.32142

Table 4.24 RMSE for Some cases with TNT

4.3 Random Forest Regression

This model was run using just 2 tools which are Scikit-learn and Scalation. Statsmodels does not have provision for running random forest regression thereby the need to exclude it for this modeling technique. In this technique, different hyperparameters were used such as the number of estimates, max depth, and random state.

4.3.1 In-sample

In this case, the hyperparameter used were several estimators set to 100 and the max depth of 10, and random state of 42.

		R^2		
		SKLEARN	SCALATION	STATSMODEL
Random forest (IN)	100000	0.991493	0.549745	
	1000000	0.965039	0.177244	
	4000000	0.963247	-	
	10000000	0.972626	-	

Table 4.29 R^2 for the Some cases with In-sample

		MAE		
		SKLEARN	SCALATION	STATSMODEL
Random forest (IN)	100000	0.379566	3.068913	
	1000000	0.900186	4.663062	
	4000000	0.859027	-	
	10000000	0.79949	-	

Table 4.30 MAE for all cases with In-sample

		MSE		
		SKLEARN	SCALATION	STATSMODEL
Random forest (IN)	100000	0.329196	17.422816	
	1000000	1.740783	40.967108	
	4000000	1.642268	-	

	10000000	1.463211	-	

Table 4.31 MSE for all cases with In-sample

		RMSE		
		SKLEARN	SCALATION	STATSMODEL
Random forest (IN)	100000	0.573756	4.174065	
	1000000	1.319387	6.400555	
	4000000	1.28151	-	
	10000000	1.209633	-	

Table 4.32 RMSE for all cases with In-sample

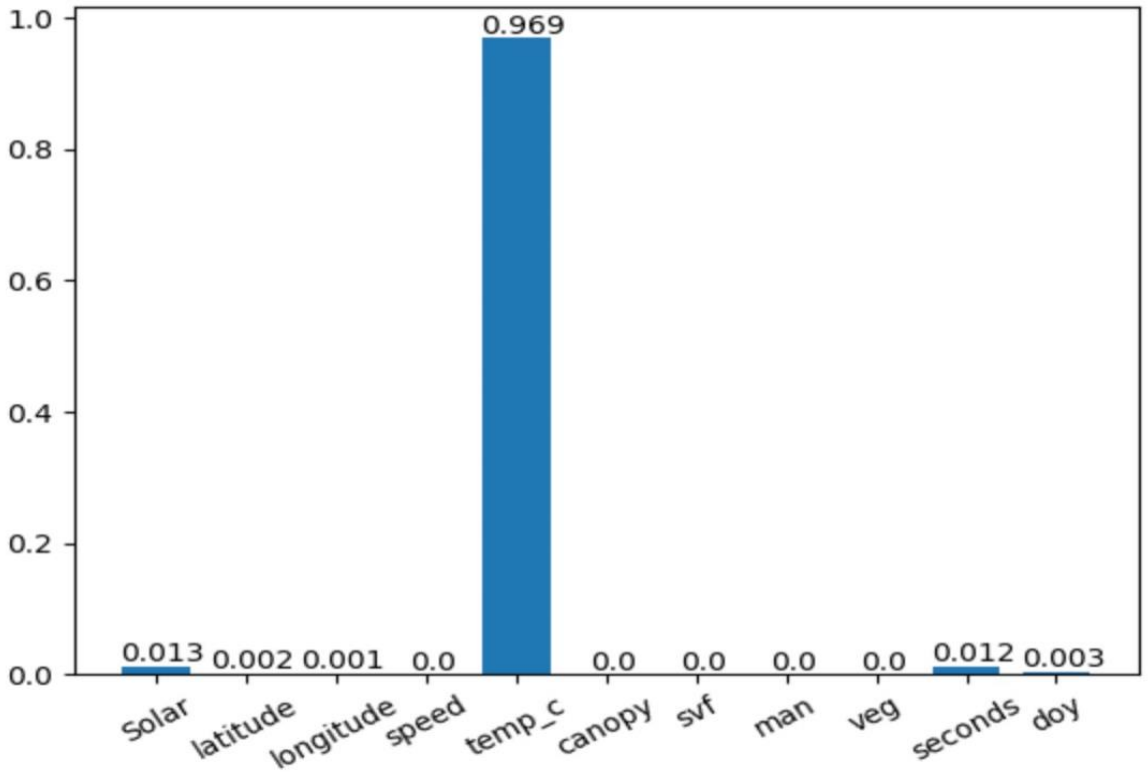


Figure 4.2 Feature Importance for 100k data

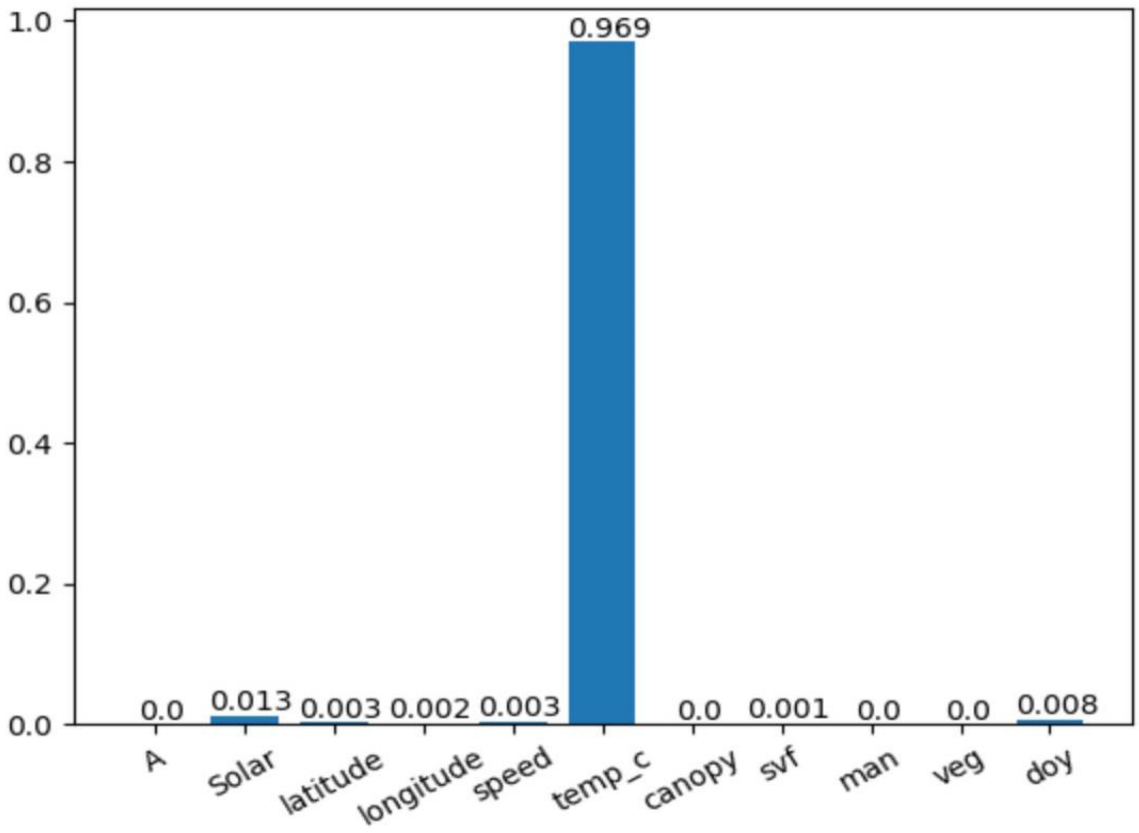


Figure 4.3 Feature Importance for 1Million dataset

4.3.2 ***Train and Test (TNT)***

In this case, the hyperparameter used include setting the number of estimators set to 100 and the max depth to 10, and the random state to 42.

		R^2		
		SKLEARN	SCALATION	STATSMODEL
Random forest (TNT)	100,000	0.991285	0.192361	
	1,000,000	0.96467	0. 178020	
	4,000,000	0.966392	-	

	10,000,000	0.972579	-	

Table 4.37 R^2 for Some cases with TNT

		MAE		
		SKLEARN	SCALATION	STATSMODEL
Random forest (TNT)	100,000	0.322104	4.635644	
	1,000,000	0. 817939	4.664281	
	4,000,000	0. 823472	-	
	10,000,000	0.800391	-	

Table 4.38 for Some cases with TNT

		MSE		
		SKLEARN	SCALATION	STATSMODEL
Random forest (TNT)	100,000	0.339238	32.069243	
	1,000,000	1.761318	40.816203	
	4,000,000	1.502221	-	
	10,000,000	1.467497	-	

Table 4.39 MSE for Some cases with TNT

		RMSE		
		SKLEARN	SCALATION	STATSMODEL
Random forest (TNT)	100,000	0.582441	5.662971	
	1,000,000	1.327147	6.388756	
	4,000,000	1.225651	-	

	10,000,000	1.211403	-	

Table 4.40 RMSE for Some cases with TNT

4.3 Gradient Boosting Regression

This model was run which are Scikit-learn. Statsmodels does not have provision for running random forest regression thereby the need to exclude it for this modeling technique. In this technique, the hyperparameters used were the number of estimates, max depth, random state, and learning rates. They were all set to 100, 10, 42, and 0.001 respectively.

4.3.1 In-sample

		R^2	MAE	MSE	RMSE
Gradient Boosting	100000	0.177977	3.440028	16.176899	4.022052
	1000000	0.865786	2.195257	7.727949	2.779919

Table 4.41 R^2 for Some cases with In-sample

4.3.2 Train and Test (TNT)

		R^2	MAE	MSE	RMSE
Gradient Boosting	100000	0.861961	1.904568	5.281008	2.298044
	1000000	0.84981	1.40695	2.728435	1.651798

Table 4.42 R^2 for the Some cases with In-sample

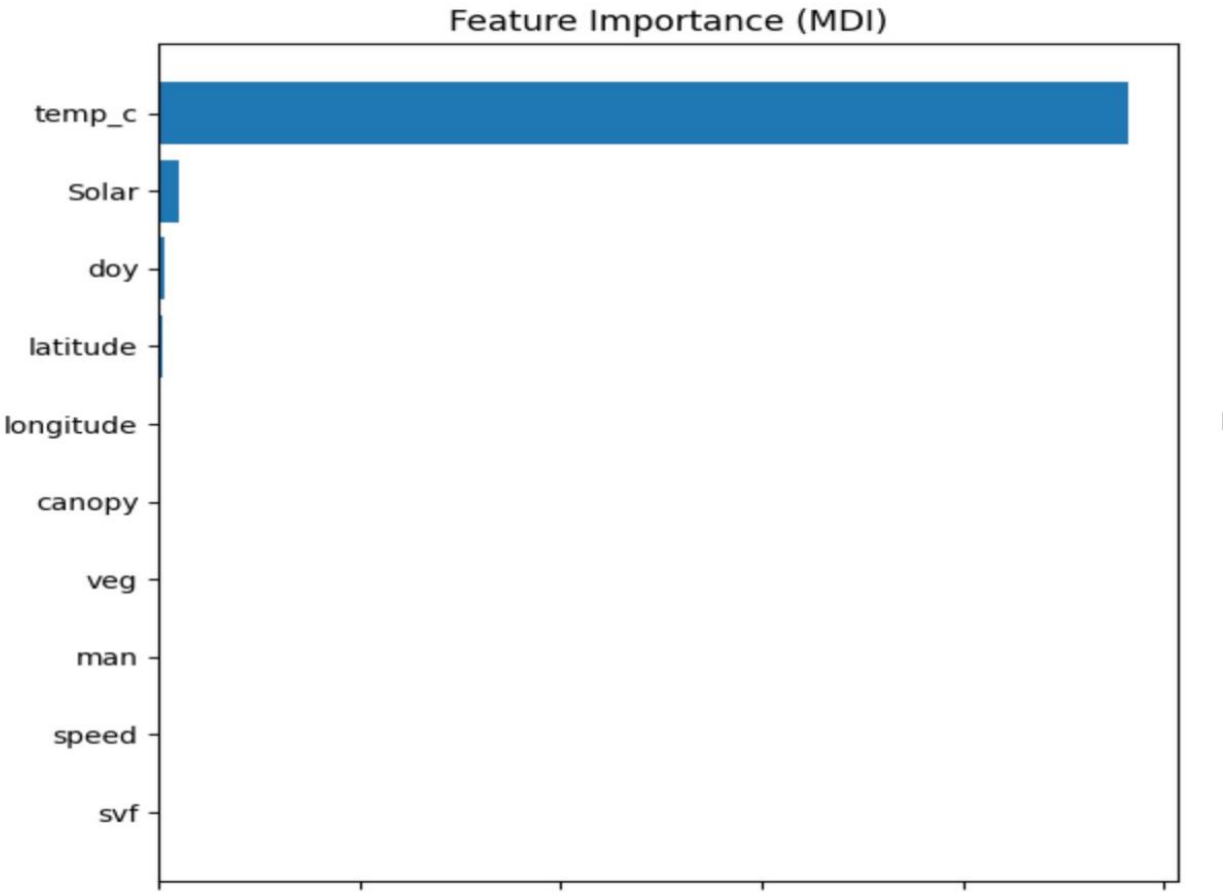


Figure 4.4 Feature importance in Gradient boosting for 100k data.

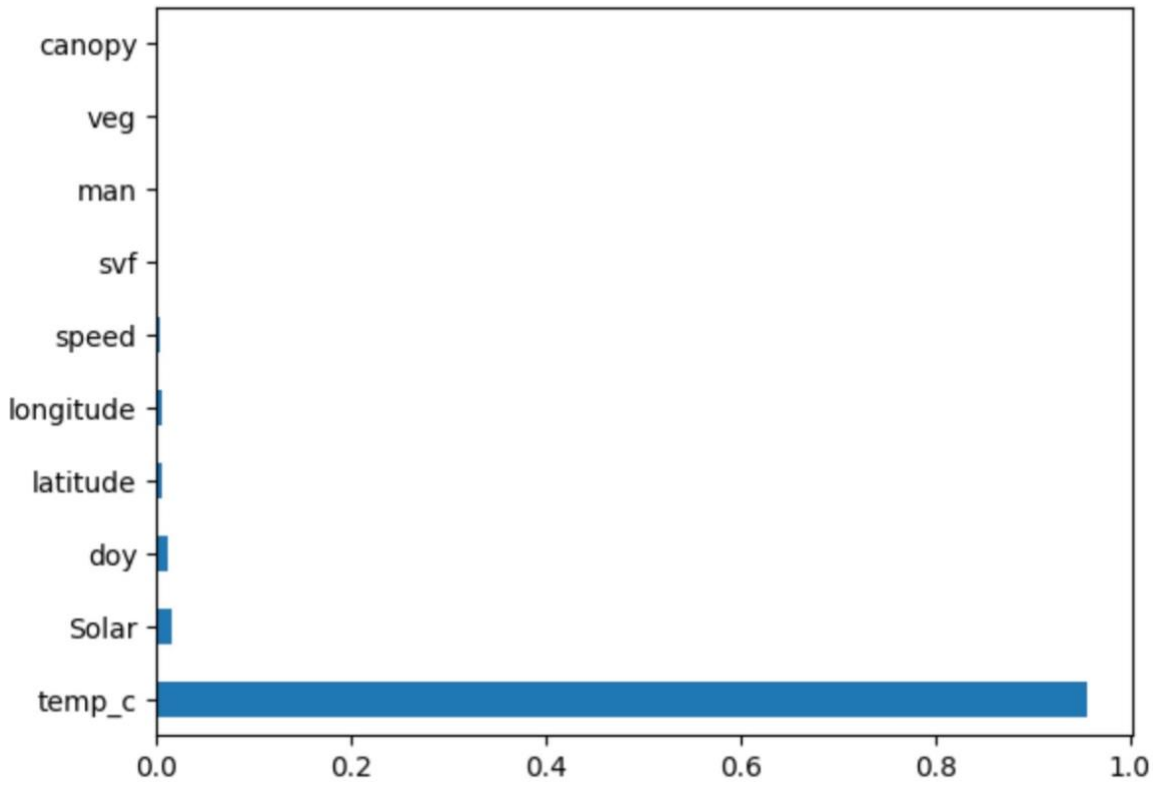


Figure 4.5 Feature importance in Gradient boosting for 1 million data.

4.5 Decision Tree Regression.

In this technique, we use max depth of 8, a random state of 42, and pruning factor of 0.2. Without pruning, the R2 was in the negatives.

4.5.1 In-sample

		R^2	MAE	MSE	RMSE
Decision Tree	100000	0.952758	1.073994	1.153463	0.758073
	1000000	0.941297	1.216663	2.922984	1.709674

Table 4.57 Decision tree for In-sample with all results.

4.5.1 Train and Test

		R^2	MAE	MSE	RMSE
Decision Tree	100000	-0.306629	3.713129	27.768691	5.269601
	1000000	0.819715	1.697062	4.973464	2.230126

Table 4.58 Decision tree for Train and test with all results.

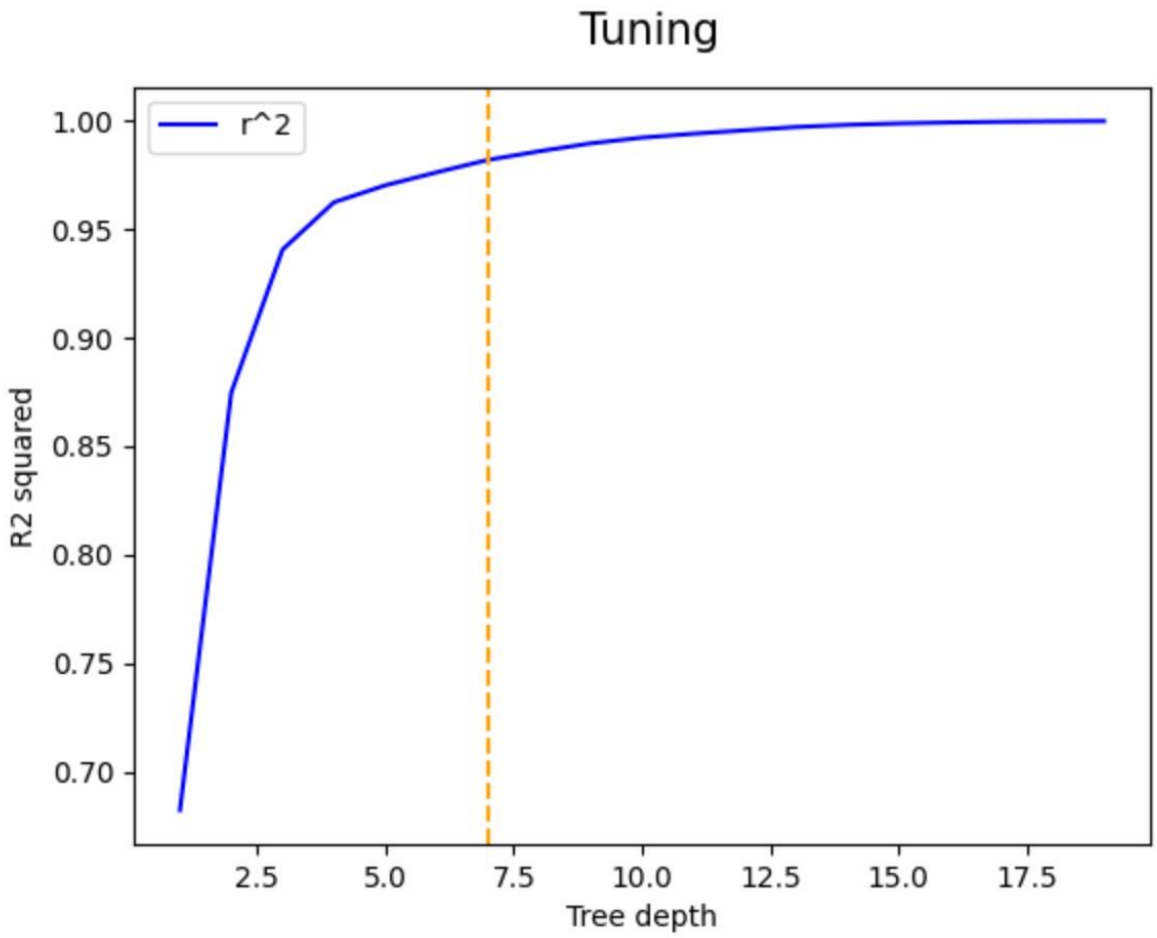


Figure 4.6 Graph of R^2against range of tree depths.

Figure 4.6 is a graph of the R^2 against different values of the tree depth to determine how well the R^2 performs in the model. It shows that the begins to remain constant from max depth of 8. This result was obtained using in-sample procedure.

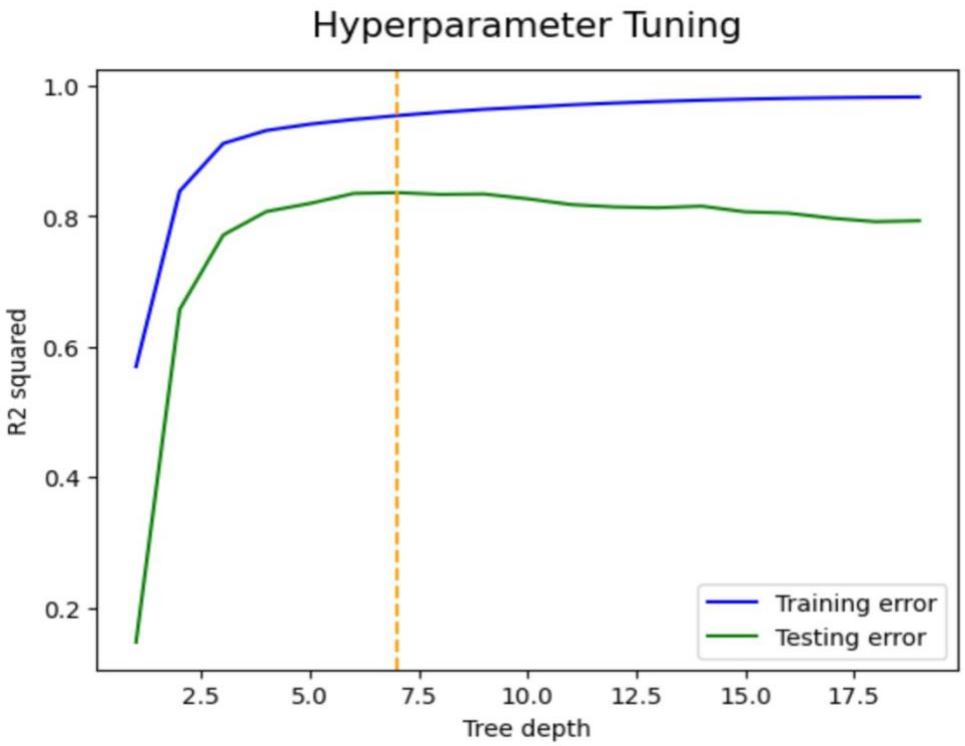


Figure 4.7 Train and test with 1 million R^2 vs tree depth

Figure 4.7 shows the R^2 plotted against a range of tree depth to determine where the best value is achieved. For the training, the value starts to even out at a max depth of 10. While for testing, the max depth is at 7. This result was obtained using the train and test procedure.

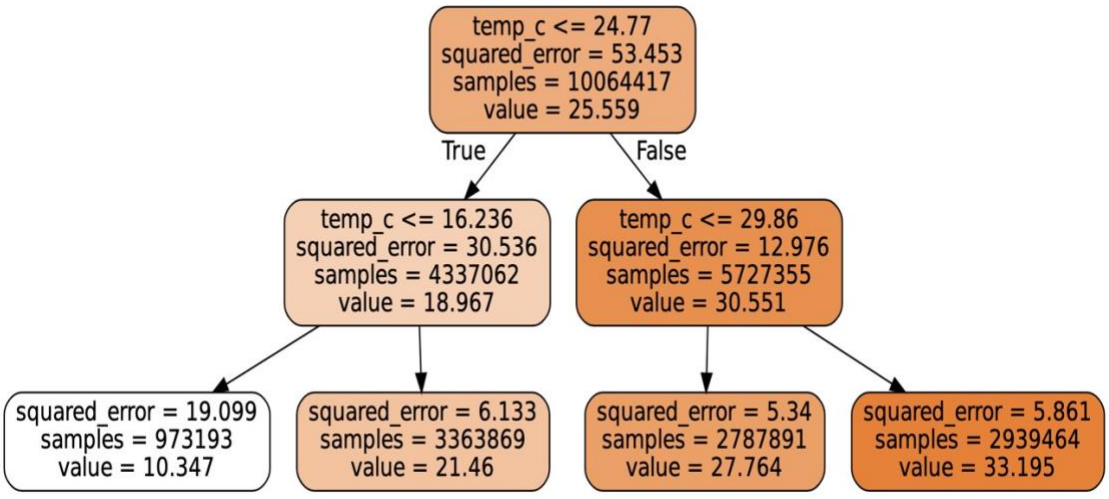


Figure 4.8 Sample of decision tree split.

Figure 4.8 A portion of the decision split tree that shows the tree with a max depth of 2.

Further look in to how the decision split is done can be found in the Appendix.

CHAPTER 5

FINDINGS

Across this study, we were able to use the different models to determine the best accuracy and we could see that in some cases across the different tools used, there were coherency between the result gotten in the different cases. Linear regression did well with all three frameworks with R^2 =0.967 and MAE =0.774 with sample size of 100,000. While in Random Forest we attained different results in the frameworks with Sklearn performing better with R^2 =0.991 and MAE = 0.374.

Other models were used to predict the best possible accuracy such as "AdaBoost" which worked well and outperformed the decision tree regressor on all techniques. I believe that it should be considered an important modeling technique in a future study. Partial least squares (PLS), SVR, and Gaussian process regression were all tried with the data to determine which tool gives the best result. K-Nearest Neighbors (KNN) was also used and gave good results for an in-sample procedure but performed badly in out-of-sample techniques. PLS performed well. SVR and Gaussian processes were not successful because they required excessively long time and large memory respectively.

5.1 Further Analysis

5.1.1 AdaBoost

AdaBoost which means adaptive boosting, can be used to boost the performance of any machine learning algorithm [55]. The algorithm builds a model and assigns equal weight to all datapoints

then assigns higher weights to points that are wrongly classified [56]. In this analysis, it was used to boost decision tree regression as it gives a negative value without pruning.

		R^2	MAE	MSE	RMSE
AdaBoost on DT	100000	0.80448	1.480828	4.15522	2.038436
	1000000	0.847558	1.541096	4.205367	2.050699

Table 5.1 AdaBoost result for the dataset.

5.1.2 Partial Least Squares (PLS)

PLS is a technique used in regression for reducing the predictors to uncorrelated small sets of the component and performs a partial least squares regression on this component [57]. It is majorly used for developing predictive models [58]. PLS is more robust because it does not assume the predictors are fixed.

		R^2	MAE	MSE	RMSE
PLS	10,000,000	0.7648	1.915231	6.488401	2.547234

Table 5.2 Partial least squares

Model	MAE
PLS	1.915231
Decision Tree Regression	1.697062
AdaBoost Regression	1.541096
Gradient boosting	1.40695
Linear Regression	1.139452
Quadratic Regression	0.871815
Random forest 100	0.800391

Table 5.3 Result of all MAE in the train and test procedure.

Table 3.3 Show the comparison between all the regression techniques performed using the train and test procedure. Random forest 1000 performs best because it has its feature in the hyperparameters changed such as the number of estimators set to 1000 and the max depth of 20. It goes to reason that with the further tuning of the hyperparameter, the random forest model would continue to perform better than the other techniques only thing would be to look out for overfitting. When compared to other research, we can see that this study agrees with the fact that by tuning the hyperparameter more, the random forest model gets better results [59]. The study uses similar dataset but with few different features such as the addition of the Land Surface Temperature (LST) data for the period.

5.2 Time Analysis

Using the different regression models, the time taken for each case was recorded across the different tools used. The time recorded are for the following models namely Linear, Quadratic and Random Forest Regression respectively. This is represented in the table below.

MODEL		**TIME**		
		SKLEARN	STATSMODEL	SCALATION
		seconds	seconds	seconds
LINEAR (IN)	100,000	0.0539	0.46474	0.39741
LINEAR (IN)	1,000,000	0.62899	1.33112	4.49475
LINEAR (IN)	4,000,000	3.96701	6.10624	9.62968
LINEAR (IN)	10,000,000	6.22217	9.979711	23.68394
LINEAR (TNT)	100,000	0.0483	0.69413	0.278
LINEAR (TNT)	1,000,000	0.54624	0.93864	2.14218
LINEAR (TNT)	4,000,000	3.395486	4.06347	6.68929
LINEAR (TNT)	10,000,000	7.32199	7.742461	18.64863
QUADRATIC (IN)	100,000	0.7314276	1.63516	5.41501
QUADRATIC (IN)	1,000,000	3.8663	9.81706	30.30904
QUADRATIC (IN)	4,000,000	27.18987	34.44873	164.22098
QUADRATIC (IN)	10,000,000	48.88017	67.43598	404.88054
QUADRATIC (TNT)	100,000	0.531189	1.43971	3.19706
QUADRATIC (TNT)	1,000,000	15.44459	20.65152	31.0256
QUADRATIC (TNT)	4,000,000	23.89573	24.25183	129.737
QUADRATIC (TNT)	10,000,000	35.95589	37.35108	320.12333
RF (IN)	100,000	32.7350	-	321.10798
RF (IN)	1,000,000	312.2934	-	17938 / 299 mins
RF (IN)	4,000,000	1478.7912	-	-
RF (IN)	10,000,000	4430.9147	-	-
RF (TNT)	100,000	25.5461	-	223.76450
RF (TNT)	1,000,000	311.7252	-	10119 / 168 min
RF (TNT)	4,000,000	1472.2623	-	-
RF (TNT)	10,000,000	3420.3180	-	-

Table 5.4 Time taken for each model to run different case.

The result from Table 5.4 shows that the time it takes for python models to run is faster than the time taken for scalation to run. Also, the results support the notion that the train test should always take less time than the in-sample seeing as the algorithm involves splitting of the data used in ratio 75:25.

During this research, it was found that the existing algorithm for the train and test split used in Scalation was considerably slow and this resulted into a high requirement of memory. This finding, brought about an update in the source code of Scalation to further improve the time taking for this process to run. A table showing the difference in the times is given below. In few cases, the previous algorithm was not able to generate a result due to the excessively high computational time. With respect to Random Forest regression, Statsmodels framework was excluded in the computation analysis because it currently has no RF modelling technique. The able above shows record for Scalation and Scikit Learn with the only case not available being that of the 10million in Scalation. In all the modeling techniques whose times have been recorded above, Python's Scikit-learn does runs considerably faster. It is also possible that with less data, Scalation will do as good as or probably better than Scikit-Learn

MODEL		TIME	
		New Scalation (s)	Old Scalation (s)
LINEAR (IN)	100,000	0.39741	0.3784
LINEAR (IN)	1,000,000	4.49475	6.4569
LINEAR (IN)	4,000,000	9.62968	12.2344
LINEAR (IN)	10,000,000	23.68394	25.091
LINEAR (TNT)	100,000	0.35471	18.4344
LINEAR (TNT)	1,000,000	2.14218	1878.3524
LINEAR (TNT)	4,000,000	6.68929	N/A
LINEAR (TNT)	10,000,000	18.64863	N/A
QUADRATIC (IN)	100,000	5.41501	4.1507
QUADRATIC (IN)	1,000,000	30.30904	42.4335
QUADRATIC (IN)	4,000,000	164.22098	169.8285

QUADRATIC (IN)	10,000,000	404.88054	409.3346
QUADRATIC (TNT)	100,000	3.19706	22.75983
QUADRATIC (TNT)	1,000,000	3.10256	1984.132
QUADRATIC (TNT)	4,000,000	129.737	N/A
QUADRATIC (TNT)	10,000,000	320.12333	N/A

Table 5.5 Comparison between the old time and new time after updating Scalation.

5.3 Memory Usage Analysis

Another, goal of this research is to determine how much memory is used by the different tools when running the different regression models. Again, we can conclude that Python uses less memory than Scalation. To reduce the memory required for the models especially Random Forest, the general algorithm used for quick sort which is used in scalation had to be reconfigured in other to accommodate a high dataset as it works suitably well with a dataset as low as 10,000 but not as high as 50,000. With this, it was not running for the selected test cases of 100,000 and others. The issue was always a stack overflow error but with the new redesign of the algorithm, we have been able to run the process and reduce the time and memory consumption significantly.

MODEL		**MEMORY**	
		PYTHON	SCALATION
LINEAR (INSAMPLE)	100,000	83 MB	292 MB
LINEAR (INSAMPLE)	1,000,000	358 MB	1.79 GB
LINEAR (INSAMPLE)	4,000,000	1.18 GB	4.54 GB
LINEAR (INSAMPLE)	10,000,000	3.54 GB	19.40 GB
LINEAR (TNT)	100,000	35 MB	214 MB
LINEAR (TNT)	1,000,000	277 MB	1.95 GB
LINEAR (TNT)	4,000,000	1.15 GB	4.10 GB
LINEAR (TNT)	10,000,000	3.42 GB	12.97 GB

QUADRATIC (INSAMPLE)	100,000	119 MB	2.01 GB
QUADRATIC (INSAMPLE)	1,000,000	308 MB	7.03 GB
QUADRATIC (INSAMPLE)	4,000,000	8.32 GB	15.43 GB
QUADRATIC (INSAMPLE)	10,000,000	20.86 GB	27.53 GB
QUADRATIC (TNT)	100,000	147 MB	1.82 GB
QUADRATIC (TNT)	1,000,000	591 MB	7.93 GB
QUADRATIC (TNT)	4,000,000	11.30 GB	10.47 GB
QUADRATIC (TNT)	10,000,000	23.79 GB	27.37 GB
RF (IN)	100,000	140 MB	409 MB
RF (IN)	1,000,000	437 MB	3.85 GB
RF (IN)	4,000,000	1.52 GB	12.64 GB
RF (IN)	10,000,000	5.19 GB	-
RF (TNT)	100,000	145 MB	792 MB
RF (TNT)	1,000,000	1.08 GB	2.11 GB
RF (TNT)	4,000,000	1.91 GB	9.19 GB
RF (TNT)	10,000,000	4.38 GB	-

Table 5.6 Memory consumed/ used by each model to run different case.

5.3 Scientific Study

During this study, other research was reviewed, and it showed that there have been different ways or strategies of analyzing and forecasting ambient temperature. In a study, using the temperature data in Norway, the research used the Random Forest (RF) algorithm to predict the ambient air temperature (Tair). The analysis of the model was performed by determining the RMSE and R^2 based on the number of RF trees. It also suggests RF models are good for hyperlocal response for ambient temperature which our research agrees with [61]. The study also shows that certain features such as vegetation, building structure have effects on Tair. In terms of accuracy, they achieved an R^2 of 0.5 and 0.43. It is good to

notice that a possible reason for the low R^2 value is due to the large surface area being covered in the study compared to the concentrated area cover in our research.

Another Study used linear regression models, Gradient boosting, Random Forest and Voting regressor etc. to determine the air temperature. The study compared the outputs of the different models and concluded that Random Forest performs better than the others [59]. Like the research performed in this study and others Random Forest looks to be performing exceptionally well when it comes to predicting the ambient Air temperature.

In a study where just like this research, hyperlocal temperature prediction was the aim. But in this study, the climate of an urban area was the focus, and this area was somewhere in Chicago. The research made use of urbanized weather research forecast model (uWRF). This model was used to predict the temperature based on some factors such as building height, impervious surface fraction, canopy height, vegetation coverage, development intensity that are available and then compared against other weather station readings for the same period. The R^2, RMSE and MAE where all considered as the measure of accuracy. The research focuses majorly on the accuracy of the suggested model uWRF. The model was able to achieve an $R^2 = 0.82$ when compared to the Global Historical Climatology Network (GHCN). The uWRF model uses an adoption of the Gaussian process model to estimate the air temperature. Different test groups of GPR models with different combinations of the variables were made and then the best group was selected for the model with the best performance. This nominated model is further tuned via the hyperparameter optimization process.

Title	Author	Method	Data Features

Hyperlocal mapping of urban air temperature using remote sensing and crowdsourced weather data. (03/26/2020)	Zander Venter, Oscar Brousse, Igor Esau, Fred Meier	Random Forest.	vegetation, building structure, canopy, LST, Infrared, Light wavelengths, sun exposure and wind.
Urban ambient air temperature estimation using hyperlocal data from smart vehicle-borne sensors. (08/19/2020)	Yanzhe Yin, Navid Tonekaboni, Andrew Grundstein, Deepak R. Mishra, Lakshmish Ramaswamy, John Dowd	Random Forest, Voting Regressor, Gradient Boosting, Linear Regression	LST, Weather station temperature, vegetation, building, soil, canopy cover, water, and solar radiation.
Hyper-local temperature prediction using detailed urban climate informatics. (04/11/2023)	Peiyuan Li, Ashish Sharma	Urban weather research forecast (uWRF) model based of GPR.	building height, impervious surface fraction, canopy height, vegetation coverage, development intensity, Weather station (GHCN)

Table 5.7 Summary comparison of other studies.

Table 5.7 shows some papers and the types of models used in their predictive research. In the table, we see the similarities in the features and notice that vegetation, building structures and canopy

are common amongst the study which are also features that are present in the research focused on in this paper. The features have a high impact in ambient temperature prediction. In conclusion when it comes to the prediction of ambient air temperature, the LST or weather station data, vegetation and building coverage are the most important features as corroborated by this studies and other research.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

The research in this thesis involves using machine learning models along with a different set of features in a dataset for hyperlocal weather prediction. The ML models used the results to determine how important some of the features were and how influential ML is in weather forecasting. In this study, the results obtained from the regression analysis supports the fact that one of the effects of certain features overshadows the rest. When compared to other studies, the machine learning techniques differ as most tend to work more with deep learning models compared to the regression models used here. Also, when using the regression models, one common trend with other research is that the accuracy of the models increases as the complexity increases.

In terms of feature importance, we went with the basic feature importance as it is readily available across all regression models compared to the permutation importance or SHAP. Especially with the fact that SHAP takes a long time in running which is longer than the time it takes to run the model before the importance check. Based on the features such “man”, “veg” and “canopy” which are available in the data used for this study, it is possible to determine hot spot locations present.

Several regression modeling techniques were used in this research which are Linear Regression, Quadratic Regression, Random Forest Regression, Gradient Boosting, and Decision Tree regression. Other regression models were notably used such as K-Nearest Neighbors, AdaBoost, Gaussian processes, Partial Least Squares. When we look at the models, in terms of accuracy, the more complex the algorithm or tunning of the parameter the better the results achieved which is why Random Forest is one of the best. Another factor considered in this tool is

the efficiency of the tools which looks at the time and space required for the model, and we conclude that the best is the linear regression followed closely by Quadratic regression while the least are SVR and the Gaussian Process which take a long time or require large memory. The results obtained from the models are all easy to understand and explain based on how they were designed.

In this Thesis, we also aim to check the consistency between three tools which are Scalation, Scikit Learn, and Statsmodels for prediction analysis. We test the tools to determine how well it stands in comparison to each other based on some factors such as the resulting errors, the accuracy, and more. By comparing these tools, on a large dataset, we were also able to determine the best features of the data based on the selected modeling techniques used. In this study, we were able to see that the most important feature is the "temp_c" from the dataset but if we isolate that feature, we obtain results showing that "solar", "tod", "doy", and "speed" are also very important. We also saw that the best modeling technique used was the Random Forest.

Also, we saw that all the tools compared have different advantages over the others. One of the advantages of Scikit learn is that it requires less memory when running large datasets compared to Scalation, an advantage of Scalation is that it can run all the regression algorithms or techniques like Scikit-learn which Statsmodels currently does not achieve. Statsmodels and Scalation are both capable of producing a general summary of all the model that was run and calculating other factors such as the SST, SSR, variance inflation factor, etc. while Scikit Learn is not equipped with this capability [60]. Regarding time or speed, the Python frameworks, both Scikit Learn and Statsmodels are considerably faster than Scalation. As a framework or library, Scalation can run feature selection such as forward and backward feature selection whereas the Python tools will require the use of other external libraries to perform this. In a large view, the Python tools are easier to use because they it is dynamically typed, while Scalation is statically

typed. Although dynamically typed languages like python are slower, but they can attain good performance using C/C++ libraries such as Basic Linear Algebra Subprograms (BLAS) and Linear Algebra Package (LAPACK).

The research showed that the highest R^2 values were obtained in the random forest model in Scikit Learn and it shows that is where the most important feature is the temperature in the independent variables. It also shows that the MAE is lowest when we use the random forest regression of about 0.80 with the dataset of one million but there was a case with 100,000 dataset that resulted in 0.37.

In terms of future work, the study of more hyperlocal areas for weather prediction can be considered and what other features can be used in predicting the weather. For a different location, the features that will be considered will differ from what has been used previously. Also, the use of Machine Learning Models can be explored using Python tools like Keras and PyTorch. ML tools will likely be better suited for revised research in weather forecasting and prediction. Scalation can investigate increasing its processing speed, it can also be checked to make sure that it is using the same conditions as the other tools, for example in the train and test split, its split is the opposite of what the Python's Scikit Learn tool uses. Also, tuning of Neural Networks should also be considered as both Python and Scalation have capabilities for this. While Quadratic regression was considered, other forms of symbolic regression can be considered. Finally, Multivariate time series forecasting could be applied to achieve more accurate results.

REFERENCE

[1] D. S. Roy, “Forecasting the Air Temperature at a Weather Station Using Deep Neural Networks,” *Procedia Comput Sci*, vol. 178, pp. 38–46, 2020, doi: 10.1016/J.PROCS.2020.11.005.

[2] T. Anjali, K. Chandini, K. Anoop, and V. L. Lajish, “Temperature Prediction using Machine Learning Approaches,” *2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies, ICICICT 2019*, pp. 1264–1268, Jul. 2019, doi: 10.1109/ICICICT46008.2019.8993316.

[3] Y. Radhika and M. Shashi, “Atmospheric Temperature Prediction using Support Vector Machines,” *International Journal of Computer Theory and Engineering*, pp. 55–58, 2009, doi: 10.7763/IJCTE.2009.V1.9.

[4] S. S. Baboo and I. K. Shereef, “An Efficient Weather Forecasting System using Artificial Neural Network,” *International Journal of Environmental Science and Development*, pp. 321–326, 2010, doi: 10.7763/IJESD.2010.V1.63.

[5] L. Goddard, S. J. Mason, S. E. Zebiak, C. F. Ropelewski, R. Basher, and M. A. Cane, “Current approaches to seasonal to interannual climate predictions,” *International Journal of Climatology*, vol. 21, no. 9, pp. 1111–1152, Jul. 2001, doi: 10.1002/JOC.636.

[6] R. R. Chaves, R. S. Ross, and T. N. Krishnamurti, “Weather and seasonal climate prediction for South America using a multi-model superensemble,” *International Journal of Climatology*, vol. 25, no. 14, pp. 1881–1914, Nov. 2005, doi: 10.1002/JOC.1230.

[7] H. Wang, M. S. Pathan, Y. H. Lee, and S. Dev, “Day-Ahead Forecasts of Air Temperature,” *2021 IEEE USNC-URSI Radio Science Meeting (Joint with AP-S Symposium), USNC-URSI 2021 - Proceedings*, pp. 94–95, 2021, doi: 10.23919/USNC-URSI51813.2021.9703606.

[8] P. Chen, A. Niu, D. Liu, W. Jiang, and B. Ma, “Time Series Forecasting of Temperatures using SARIMA: An Example from Nanjing,” *IOP Conf Ser Mater*

Sci Eng, vol. 394, p. 052024, Aug. 2018, doi: 10.1088/1757-899X/394/5/052024.

[9] M. Murat, I. Malinowska, M. Gos, and J. Krzyszczak, "Forecasting daily meteorological time series using ARIMA and regression models**," *Int. Agrophys*, vol. 32, pp. 253–264, 2018, doi: 10.1515/intag-2017-0007.

[10] "Holt-Winters Forecasting and Exponential Smoothing Simplified - Orange Matter." https://orangematter.solarwinds.com/2019/12/15/holt-winters-forecastingsimplified/ (accessed Dec. 09, 2022).

[11] "Box-Jenkins Model: Definition, Uses, Timeframes, and Forecasting." https://www.investopedia.com/terms/b/box-jenkins-model.asp (accessed Dec. 09, 2022).

[12] H. Astsatryan and R. Abrahamyan, "Air temperature forecasting using artificial neural network for Ararat valley Network for Sustainable Ultrascale Computing (NESUS)-COST Action View project Second Administrative Level Boundaries dataset (SALB) View project", doi: 10.1007/s12145-021-00583-9.

[13] S. Manandhar, Y. H. Lee, Y. S. Meng, F. Yuan, and S. Dev, "A potential low cost remote sensing using GPS derived PWV," *International Geoscience and Remote Sensing Symposium (IGARSS)*, vol. 2018-July, pp. 903–906, Oct. 2018, doi: 10.1109/IGARSS.2018.8517797.

[14] J. Rice, W. Xu, and A. August, "Analyzing Koopman approaches to physicsinformed machine learning for long-term sea-surface temperature forecasting,"
2020, Accessed: Dec. 04, 2022. [Online]. Available: https://github.com/JRice15/physics-informed-

[15] S. M. Lundberg, P. G. Allen, and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions", Accessed: Dec. 04, 2022. [Online]. Available: https://github.com/slundberg/shap

[16] X. Li, Y. Wang, S. Basu, K. Kumbier, and B. Yu, "A Debiased MDI Feature Importance Measure for Random Forests".

[17] "Classification and Regression Trees - Leo Breiman - Google Books." https://books.google.com/books?id=MGlQDwAAQBAJ&printsec=frontcover&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false (accessed Dec. 04, 2022).

[18] P. Wei, Z. Lu, and J. Song, “Variable importance analysis: A comprehensive review,” *Reliab Eng Syst Saf*, vol. 142, pp. 399–432, Jul. 2015, doi: 10.1016/J.RESS.2015.05.018.

[19] C. Strobl, A.-L. Boulesteix, A. Zeileis, and T. Hothorn, “Bias in random forest variable importance measures: Illustrations, sources and a solution,” 2007, doi: 10.1186/1471-2105-8-25.

[20] C. Strobl, A.-L. Boulesteix, T. Kneib, T. Augustin, and A. Zeileis, “Conditional variable importance for random forests,” 2008, doi: 10.1186/1471-2105-9-307.

[21] S. Basu, K. Kumbier, J. B. Brown, and B. Yu, “Iterative random forests to discover predictive and stable high-order interactions,” *PNAS*, vol. 115, no. 8, pp. 1943–1948, 2018, doi: 10.1073/pnas.1711236115.

[22] T. Chen and C. Guestrin, “XGBoost: A Scalable Tree Boosting System”, Accessed: Dec. 04, 2022. [Online]. Available: https://github.com/dmlc/xgboost

[23] S. Janitza, E. Celik, and A.-L. Boulesteix, “A computationally fast variable importance test for random forests for high-dimensional data,” 2015, Accessed: Dec. 04, 2022. [Online]. Available: http://www.stat.uni-muenchen.de

[24] F. T. Tangang, W. W. Hsieh, and B. Tang Oceanography, “Forecasting the equatorial Pacific sea surface temperatures by neural network models,” *Clim Dyn*, vol. 13, pp. 135–147, 1997.

[25] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, “Deep learning,” 2015, doi: 10.1038/nature14539.

[26] Yuting Yang, Junyu Dong, Xin Sun, Estanislau Lima, Quanquan Mu, and Xinhua Wang, “A CFCC-LSTM Model for Sea Surface Temperature Prediction,” *IEEE*, vol. 15, no. 2, Feb. 2018, Accessed: Dec. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8241465

[27] C. Xiao *et al.*, “A spatiotemporal deep learning model for sea surface temperature field prediction using time-series satellite data,” *Environmental Modelling and Software*, vol. 120, Oct. 2019, doi: 10.1016/J.ENVSOFT.2019.104502.

[28] Q. Zhang, H. Wang, J. Dong, G. Zhong, and X. S. Member, “Prediction of Sea Surface Temperature using Long Short-Term Memory”, Accessed: Dec. 04, 2022. [Online]. Available: http://www.esrl.noaa.gov/psd/

[29] N. B. Erichson, L. Mathelin, Z. Yao, S. Brunton, M. W. Mahoney, and J. Kutz, "Shallow Learning for Fluid Flow Reconstruction with Limited Sensors and Limited Data," *undefined*, 2019.

[30] K. Patil, M. C. Deo, and M. Ravichandran, "Prediction of Sea Surface Temperature by Combining Numerical and Neural Techniques," *J Atmos Ocean Technol*, vol. 33, no. 8, pp. 1715–1726, Aug. 2016, doi: 10.1175/JTECH-D-15-0213.1.

[31] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations", Accessed: Dec. 04, 2022. [Online]. Available: https://github.com/

[32] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations".

[33] V. Agilan and N. v. Umamahesh, "Modelling nonlinear trend for developing nonstationary rainfall intensity–duration–frequency curve," *International Journal of Climatology*, vol. 37, no. 3, pp. 1265–1281, Mar. 2017, doi: 10.1002/JOC.4774.

[34] Ashish Juneja and Nripendra Narayan Das, "Big Data Quality Framework: PreProcessing Data in Weather Monitoring Application." https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8862267 (accessed Dec. 03, 2022).

[35] "(PDF) Machine Learning -Regression." https://www.researchgate.net/publication/357992043_Machine_Learning_-Regression#pf14 (accessed Dec. 03, 2022).

[36] "Data Cleaning in Machine Learning: Best Practices and Methods." https://www.einfochips.com/blog/data-cleaning-in-machine-learning-best-practicesand-methods/ (accessed Dec. 03, 2022).

[37] M. Komorowski, D. C. Marshall, J. D. Salciccioli, and Y. Crutain, "Exploratory data analysis," *Secondary Analysis of Electronic Health Records*, pp. 185–203, Jan. 2016, doi: 10.1007/978-3-319-43742-2_15/FIGURES/18.

[38] J. Camacho, R. A. Rodríguez-Gómez, and E. Saccenti, "Group-Wise Principal Component Analysis for Exploratory Data Analysis," *https://doi.org/10.1080/10618600.2016.1265527*, vol. 26, no. 3, pp. 501–512, Jul. 2017, doi: 10.1080/10618600.2016.1265527.

[39] “Is it possible to get the correlation value greater 1. If so, why?. | ResearchGate.” https://www.researchgate.net/post/Is_it_possible_to_get_the_correlation_value_greater_1_If_so_why (accessed Dec. 02, 2022).

[40] “What is a Correlation Matrix? - Displayr.” https://www.displayr.com/what-is-acorrelation-matrix/ (accessed Dec. 02, 2022).

[41] T. A. Craney and J. G. Surles, “Model-Dependent Variance Inflation Factor Cutoff Values,” *https://doi.org/10.1081/QEN-120001878*, vol. 14, no. 3, pp. 391–403, 2007, doi: 10.1081/QEN-120001878.

[42] “10.7 - Detecting Multicollinearity Using Variance Inflation Factors | STAT 462.” https://online.stat.psu.edu/stat462/node/180/ (accessed Dec. 04, 2022).

[43] R. Salmerón, C. B. García, and J. García, “Variance Inflation Factor and Condition Number in multiple linear regression,” *https://doi.org/10.1080/00949655.2018.1463376*, vol. 88, no. 12, pp. 2365–2384, Aug. 2018, doi: 10.1080/00949655.2018.1463376.

[44] “6 Types of Regression Models in Machine Learning You Should Know About | upGrad blog.” https://www.upgrad.com/blog/types-of-regression-models-inmachine-learning/ (accessed Dec. 02, 2022).

[45] “Regression in machine learning | 10 Popular Regression Algorithms.” https://www.jigsawacademy.com/popular-regression-algorithms-ml/ (accessed Dec. 02, 2022).

[46] J. A. Miller, “Introduction to Computational Data Science Using ScalaTion,” 2022.

[47] “Quadratic Polynomial Regression Model Solved Example - VTUPulse.” https://www.vtupulse.com/machine-learning/quadratic-polynomial-regressionmodel-solved-example/ (accessed Dec. 04, 2022).

[48] “Random Forest Algorithm for Machine Learning | by Madison Schott | Capital One Tech | Medium.” https://medium.com/capital-one-tech/random-forestalgorithm-for-machine-learning-c4b2c8cc9feb (accessed Dec. 07, 2022).

[49] “Random Forest Regression. Random Forest Regression is a… | by chaya | Level Up Coding.” https://levelup.gitconnected.com/random-forest-regression209c0f354c84 (accessed Dec. 07, 2022).

[50] “What is Cross-Validation?. Testing your machine learning models… | by

Mohammed Alhamid | Towards Data Science.” https://towardsdatascience.com/what-is-cross-validation-60c01f9d9e75 (accessed Dec. 02, 2022).

[51] A. Khan and S. Zubair, “Machine Learning Tools and Toolkits in the Exploration of Big Data,” *International Journal of Computer Sciences and Engineering*, vol. 6, no. 12, pp. 570–575, Dec. 2018, doi: 10.26438/IJCSE/V6I12.570575.

[52] Dr. S. Veena, T. Shankari, S. Sowmiya, and M. Varsha, “(PDF) A SURVEY ON TOOLS USED FOR MACHINE LEARNING,” *International Journal of Engineering Applied Sciences and Technology*, vol. 4, no. 9, pp. 116–119, Jan. 2020, Accessed: Dec. 02, 2022. [Online]. Available: https://www.researchgate.net/publication/341371202_A_SURVEY_ON_TOOLS_USED_FOR_MACHINE_LEARNING

[53] S. Rong and Z. Bao-Wen, “The research of regression model in machine learning field,” *MATEC Web of Conferences*, vol. 176, Jul. 2018, doi: 10.1051/MATECCONF/201817601033.

[54] V. Vinothina, “MACHINE LEARNING TOOLS-AN OVERVIEW”.

[55] “AdaBoost Algorithm - A Complete Guide for Beginners - Analytics Vidhya.” https://www.analyticsvidhya.com/blog/2021/09/adaboost-algorithm-a-completeguide-for-beginners/ (accessed Dec. 09, 2022).

[56] “A Guide To Understanding AdaBoost | Paperspace Blog.” https://blog.paperspace.com/adaboost-optimizer/ (accessed Dec. 09, 2022).

[57] H. Abdi, “Partial Least Squares (PLS) Regression”, Accessed: Dec. 09, 2022. [Online]. Available: http://www.utdallas.edu/

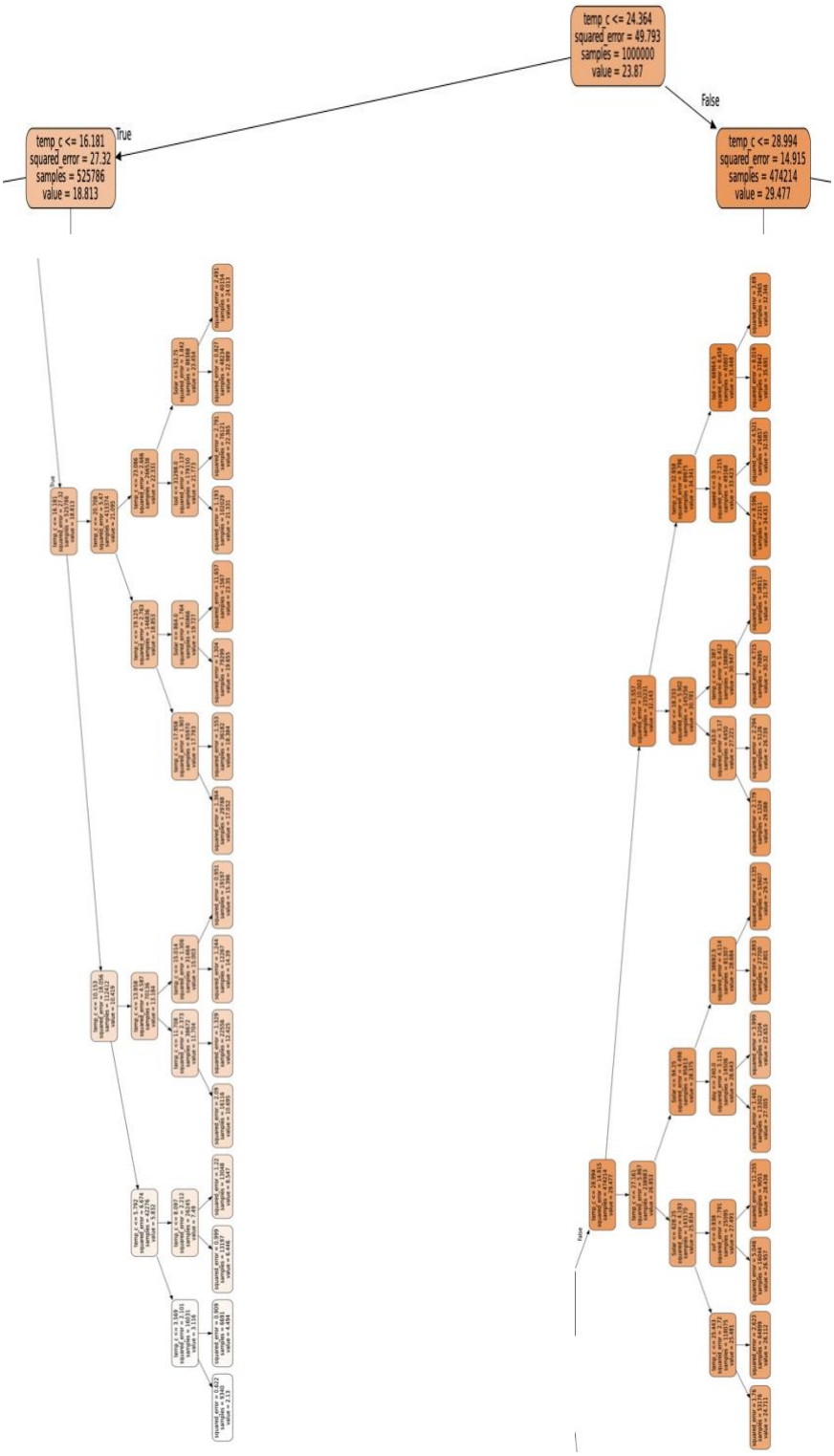
[58] “What is partial least squares regression?”.

[59] Y. Yanzhe, T. Navid Hashemi, G. Andrew, R. M. Deepak, and R. Lakshmish, “Urban ambient air temperature estimation using hyperlocal data from smart vehicle-borne sensors,” vol. 84, Jul. 2022, Accessed: Dec. 07, 2022. [Online]. Available: https://pdf.sciencedirectassets.com/271803/1-s2.0-S0198971520X00056/1-s2.0-S0198971520302714/main.pdf

[60] “5. Estimating Standard Error and Significance of Regression Coefficients — Data Science Topics 0.0.1 documentation.” https://datascience.oneoffcoder.com/estimating-standard-error-coefficients.html

(accessed Dec. 08, 2022).

[61] Venter, Z. S., Brousse, O., Esau, I., & Meier, F. (2020). Hyperlocal mapping of urban air temperature using remote sensing and crowdsourced weather data. Remote Sensing of Environment, 242. https://doi.org/10.1016/J.RSE.2020.111791



Appendix A: Decision tree diagram of max depth of 8.

```
import pandas as pd
import csv import
math


# Function to write the split file def
create_split_file(split_rows, header, split_file_name):
with open(split_file_name, 'w') as split_file:        output
= csv.writer(split_file, delimiter=",")        # Add header
to rows if you want        split_rows.insert(0, header)
output.writerows(split_rows) split_file_rows = 100000
file_path = "/home/student/gdf_all.csv"
# Read and split the rows split_rows = [] with
open(file_path) as csv_file:    # Read the CSV
file    csv_reader = csv.reader(csv_file,
delimiter=',')
   # Loop over all rows to get the number of row and all values of the
row    for row_num, row in enumerate(csv_reader):        # Get the
header from the first row        if row_num == 0:            header = row

      else:
         # If we have reached the split mark, create a split file
if row_num % split_file_rows == 0:
```

```
            # Get the number of the split file and convert to string
file_suffix = str(int(row_num / split_file_rows))
            # Create the name of the split file
split_file_name = "100k/100k-" + file_suffix + ".csv"
            # Output the split file
create_split_file(split_rows, header, split_file_name)
            # Make split rows start with this row
split_rows = [row]
        # If we haven't reached the split point, add the row to the current split_row
        else:
            split_rows.append(row)
    # When it is the end of file then output all the leftover
rows    split_rows.append(row)    file_suffix =
str(math.ceil(row_num / split_file_rows))    split_file_name
= "100k/100k-" + file_suffix + ".csv"

    create_split_file(split_rows, header, split_file_name)
```

Appendix B: The code used to split the data into different sizes.