Springboard Data Science Capstone Project I

Predicting the Occurrence of West Nile Virus in Chicago

Frew Berhe

December 13, 2019

Table of Contents

1. Introduction

West Nile Virus (WNV) is a disease caused by a virus which is most commonly spread to humans through infected mosquitoes. Around 20% of people who become infected with the virus develop symptoms ranging from a persistent fever, to serious neurological illnesses that can result in death. WNV is a disease spread from infected mosquitoes to humans. For many patients, mild symptoms go away on their own but for people with more severe diseases such as meningitis or encephalitis, these symptoms can become life-threatening. People over the age of 70 and with chronic conditions such as weakened immune systems or high blood pressure are at most risk if they are infected with WNV. Chicago has experienced one of the highest levels of WNV risk in this decade. According to the daily herald, 290 Illinois residents have been reported to be ill from this disease, the highest recorded number in Chicago since 2012. The cost of WNV varies from hospital to hospital. However, on average, each admitted patient to the hospital for WNV costs approximately $25,000. On average, the cost for each patient that receives chronic care for WNV is $22,000.

Chicago has had trouble controlling the spread of WNV. By 2004 the City of Chicago and the Chicago Department of Public Health (CDPH) had established a comprehensive surveillance and control program that is still in effect today. As a result, traps were set up to capture mosquitoes. Every week from late spring through the fall, these traps are frequently checked to see if any of the captured mosquitoes carry the

virus. Knowing which traps are more likely to have WNV present is important as it provides insight to the city as to where the city is best served to spray for the eradication of mosquitoes. Spraying can have adverse effects on the environment, and it is also very expensive, thus it's necessary to target specific areas and at specific times. Our goal is to predict where in Chicago WNV occurs to help the city prepare accordingly. The City of Chicago and the Chicago Department of Public Health (CDPH) can use such a model to get information on when and where the city will spray airborne pesticides to control adult mosquito populations. This in turn will play a significant role in the prevention and control of the disease.

2. Data Acquisition and Cleaning

The data set was acquired from the Kaggle Competition. Originally, the data has three datasets, the first dataset contains WNV testing & location information, the second dataset contains information about weather, and the third dataset contains spray data. We decided to exclude the spray data as there is not enough information about the sprays used to help guide our project. We checked the cleaned the dataset one by one and then merged them together.

The first dataset (testing & location data) has 12 columns and 10506 rows. After we dropped the unnecessary location related columns ('Trap id', 'Address', 'Block', 'Street', 'Latitude', 'Longitude', 'Address Accuracy'), we are left with only 4 columns namely date that the WNV test is performed, species of mosquitoes, the number of mosquitoes

caught, WNV presence in mosquitoes. We checked all the 6 columns for the missing values of and none of them have missing values.

The second dataset (weather data) It has 2944 rows and 22 columns. We dropped 4 unnecessary columns, and as a result we are left with 18 columns including Date, maximum temperature, minimum temperature, average temperature, departure from normal temperature, Dew Point, Wet Bulb, total precipitation. Since the majority of the columns data types is object, we changed them to numeric.

Even though the data have no null values, there were many 'M', 'I' values in most of the variables that indicate missing values, and we can also see that there ' T' values for the total precipitation column indicating trace amount. We converted the ' T' to 0.001. The data provided had weather report from different weather stations daily. Since the weather measured temperatures in different regions of Chicago, we averaged both weather data. Before we change the 'M' & '-' to missing values, we preferred to get the average of the two stations as most of the missing values are from station 2. As a result, we will have a smaller number of missing values. After we averaged the weather data of the two stations and removed the 'Station' column; Now all the variables have 1472 rows, except for 'station pressure' which has 1 missing value(nan). Hence, we filled the missing value by the mean of its column. Finally, we merged both datasets by their 'Date' column. 10506 rows and 24 columns. More details on acquiring, cleaning, merging, parsing these datasets can be found in this IPython notebook.

3. Data Exploration

3.1 Introduction to the cleaned data

Our cleaned data has 10506 rows and 24 columns. Our target variable is WNV positivity which is a categorical variable with 0 representing WNV not present and 1 representing presence of WNV in the mosquito. The 'Date' feature by itself is not helpful for our statistical analysis. However, we can extract many variables such as 'Day_of_week', 'Day_of_year', 'Week_of_year', 'Month', and 'Quarter' that can be important to our prediction model. Hence, we extracted these five features and removed the 'Date' feature from our dataset. We will go through most of the for features of the dataset to explore their relationship with WNV positivity rate. Details about each field can be found in Kaggle and *'noaa_weather_qclcd_documentation.pdf'.*

3.2 WNV positivity rate

Out of the 10506 tests for WNV presence in mosquitos only 551 (5.24%) indicate the presence of WNV in the mosquitos. This does not seem like a large number, but such rare occurrence can cause devastating disease outbreak. Therefore, it is important to understand when, where and how these rare events occur. In the following few sub-sections, we will go through many interesting features and explore the trend for WNV positivity rate. To start with, we plot the total number of WNV positives with the calendar variables extracted from the Date variable. We have five calendar variables such as quarter, month, week, day of week, and day of year.

3.3 Summary Statistics

Majority of the columns means except ('DewPoint', 'StnPressure', 'SeaLevel', 'ResultSpeed', 'ResultDir', 'AvgSpeed') are different from the median represented by the 50^{th} percentile. Besides, there is notably large difference between mean and 75^{th} percentile in 'Heat', 'NumMosquitos', 'PrecipTotal'. Therefore, we can understand that there are outliers in these variables stated above. To be more specific we can also check the difference b/n the 95th percentile and maximum value, as well as the 5th percentile and minimum value. There is big difference between 95 percentile and maximum value for 'Heat', 'PrecipTotal', & 'ResultSpeed'. This indicates the presence of extreme outliers in these three variables. There is no big difference between the 5th percentile and minimum value for all the variables

3.4 Visual exploratory data analysis

Regarding the skewness of the distribution of each feature, we can see from the figure below that 'StnPressure', and 'SeaLevel' appear to be normally distributed; while 'PrecipTotal', 'Heat', and 'NumMosquitos' are extremely skewed to the right. On the other hand, 'Sunrise', 'NumMosquitos', and 'WetBulb' appear to be bimodal, whereas 'Cool', 'SunSet', 'ResultSpeed', and 'ResultDir' appear to be multimodal. The remaining features are slightly skewed to either right or left.

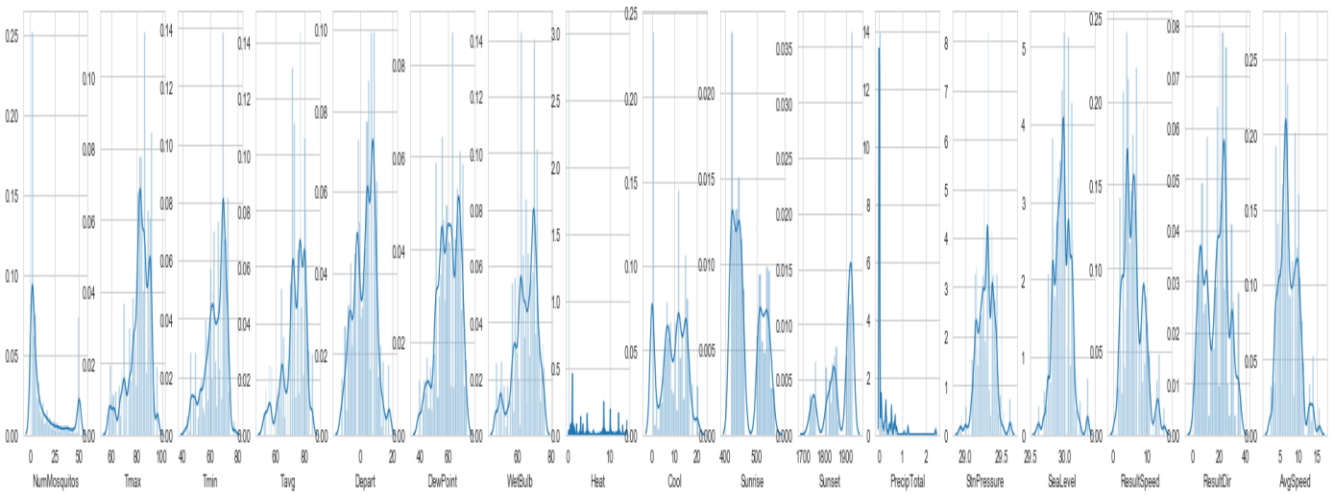

Figure 1: Distribution of all features.

A box plot (or box-and-whisker plot) shows the distribution of quantitative data in a way that facilitates comparisons between variables. The box shows the quartiles of the dataset while the whiskers extend to show the rest of the distribution.

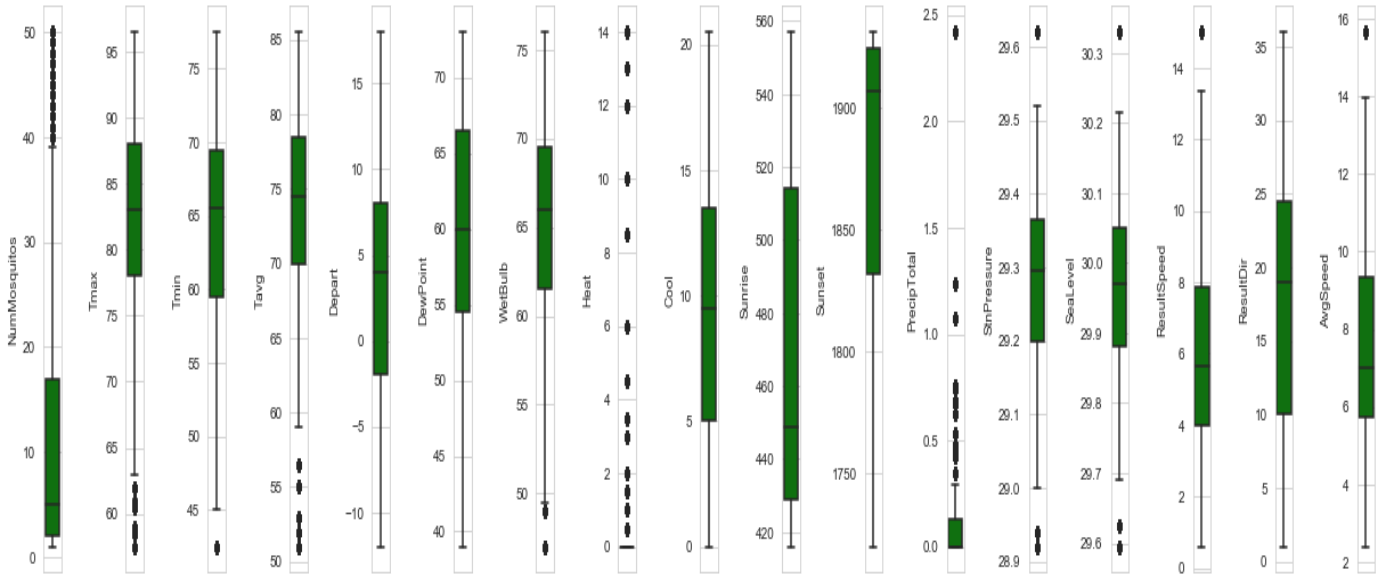

Figure 2: Box plot for all features

The black dot above or below the whiskers indicate the presence of outlier. 'DewPoint', 'Cool', 'Sunrise', 'Sunset', & 'ResultDir' are the features without outliers. All other columns show outliers.

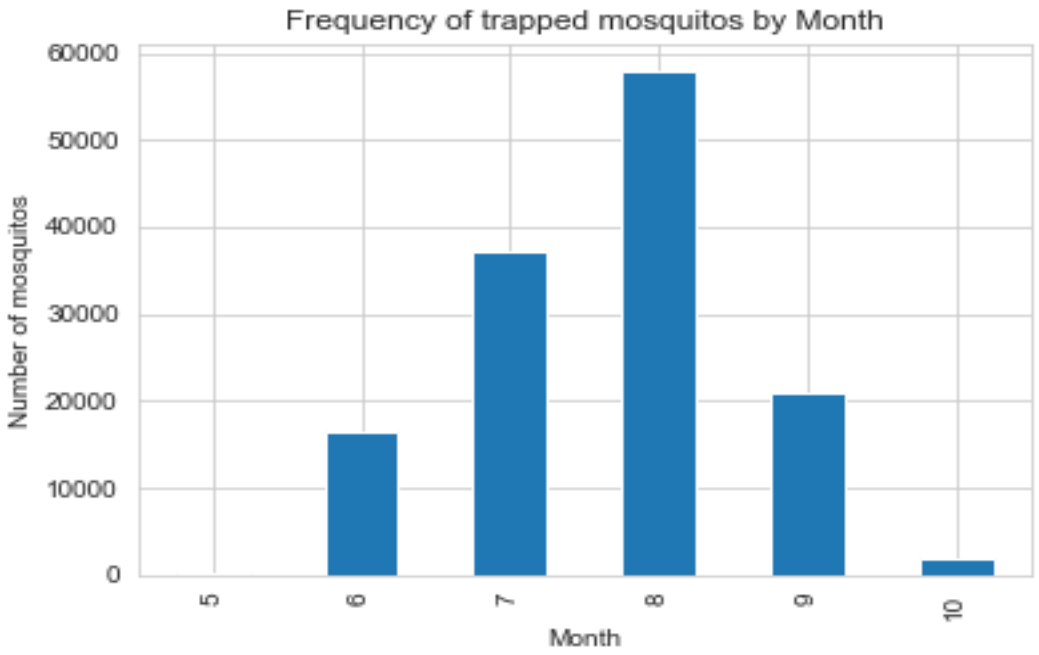

Figure 3: WNV positivity rate by month.

The highest WNV positivity rate was recorded in August followed by July, September, and June. WNV positivity rate is high in the third quarter, in August and September, which is in line with time of year when mosquito populations will be largest (August, July, June & September). Figure 3 shows the number of mosquitos trapped by month.

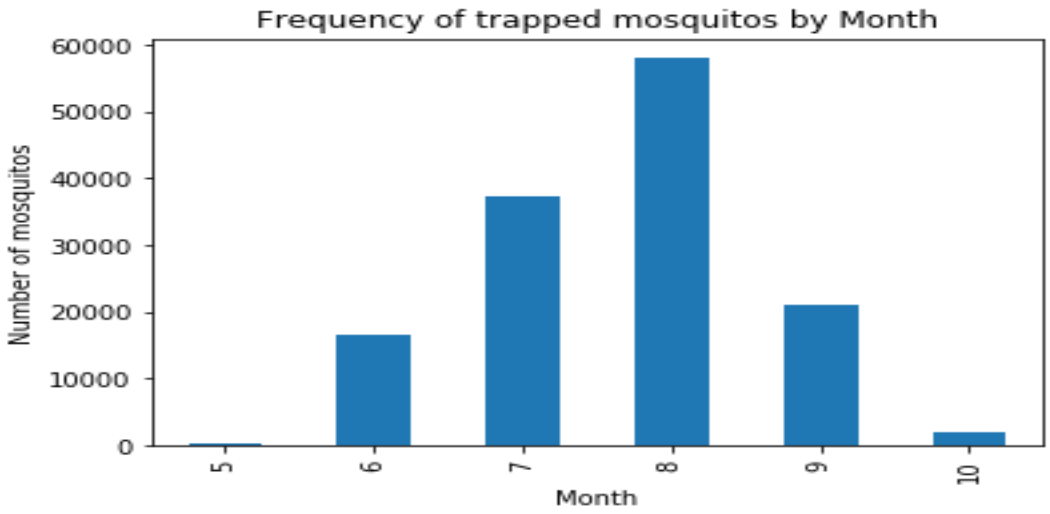

Figure 4: Number of mosquitoes caught in a trap by month.

On top of that, WNV positivity rate is high in Thursday, Tuesday and Wednesday as shown in figure 4. However, we don't have any data that was obtained during the weekends (Saturday, and Sunday). Mosquito traps were done from Monday to Friday only.

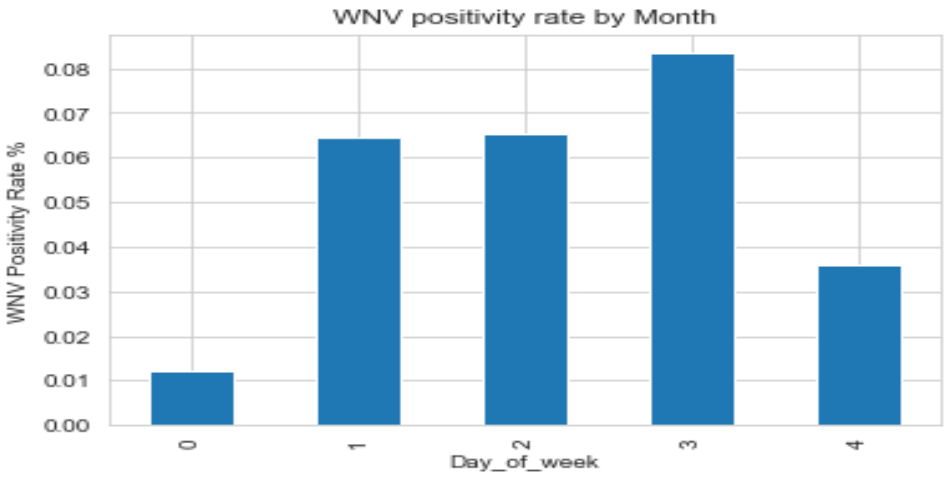

Figure 5: WNV positivity rate by Month.

We have seven species of Culex mosquitoes, namely *'CULEX PIPIENS/RESTUANS', 'CULEX RESTUANS', 'CULEX PIPIENS', 'CULEX SALINARIUS', 'CULEX TERRITANS', 'CULEX TARSALIS',* and *'CULEX ERRATICUS'*. *'CULEX PIPIENS/RESTUANS', 'CULEX RESTUANS', 'CULEX PIPIENS'* were the three dominant species. Knowing which mosquito species are more likely to carry the virus will be useful if the species tend to exist in different areas. *CULEX PIPIENS/RESTUANS* (262), *CULEX PIPIENS* (240), & *CULEX RESTUANS* (49) were the only species carrying WNV.

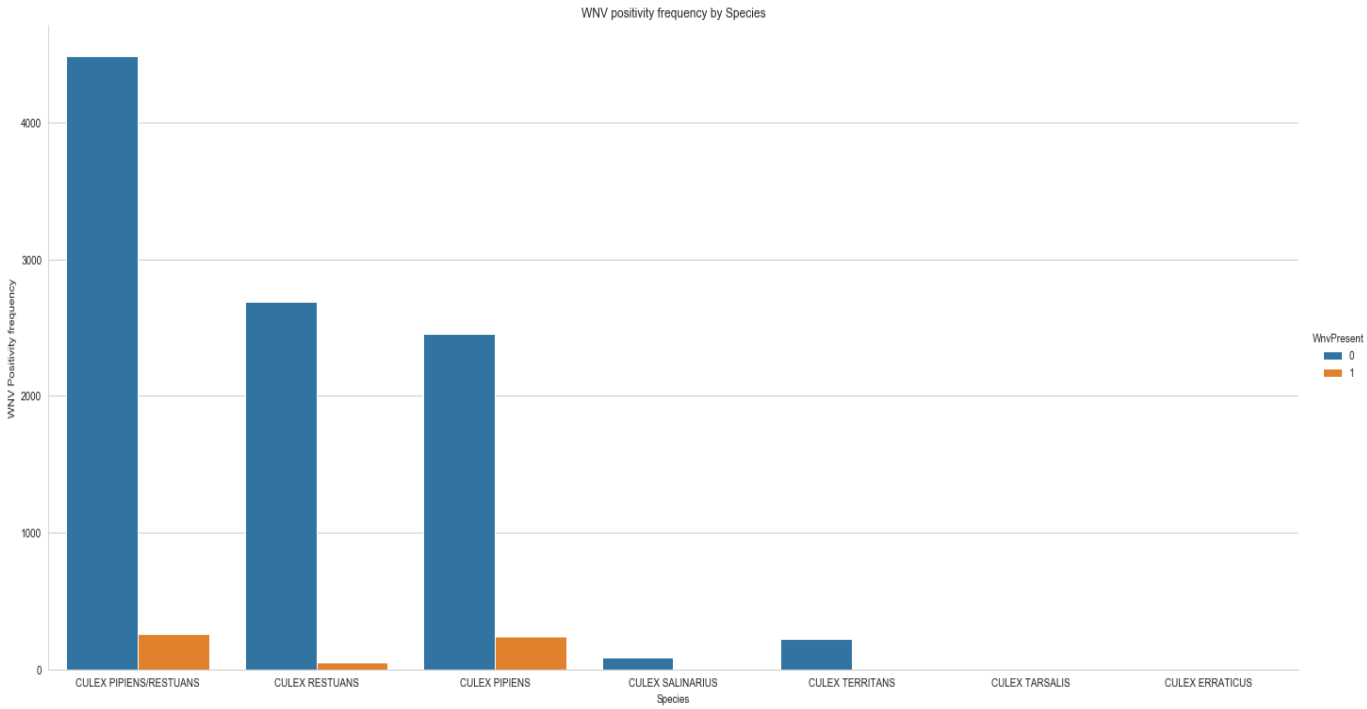

Figure 6: WNV positivity frequency by Species

Later in the machine learning part, we will filter our dataset to these three species only since the rest species have null value and as a result have no relevance in building prediction model.

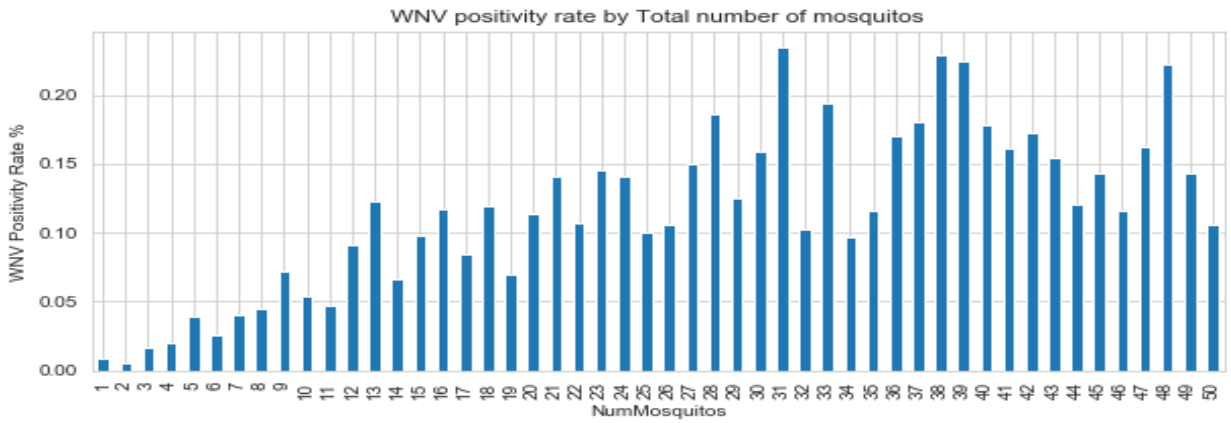

Figure 7: WNV positivity frequency by Species by Total number of mosquitoes in a trap

The above chart depicts that as the number of trapped mosquitoes increase WNV positivity rate also increases. This finding is straight forward that the more sample we have the more variety of mosquito species we get and consequently we will have more WNV positivity rate.

Weather Factors

There are many weather factors such as temperature, precipitation, dew point, pressure, windspeed, wind direction, humidity etc. but we will focus on only some factors here to keep the discussion short. A detailed data exploration can be found in this IPython notebook. The temperature data is described in four different forms namely maximum temperature, minimum temperature, average temperature, departure from normal temperature.

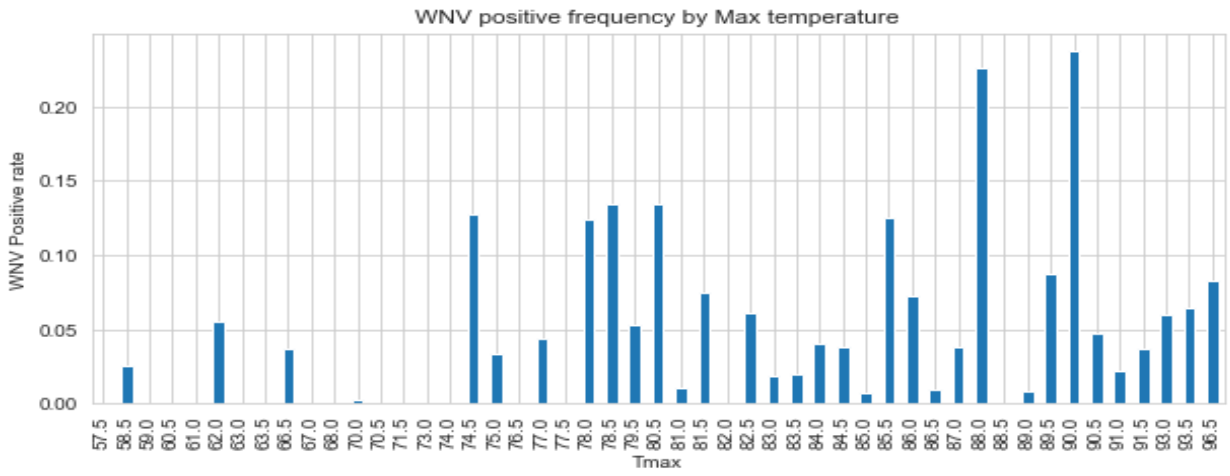

Figure 8: WNV positive frequency by Maximum temperature

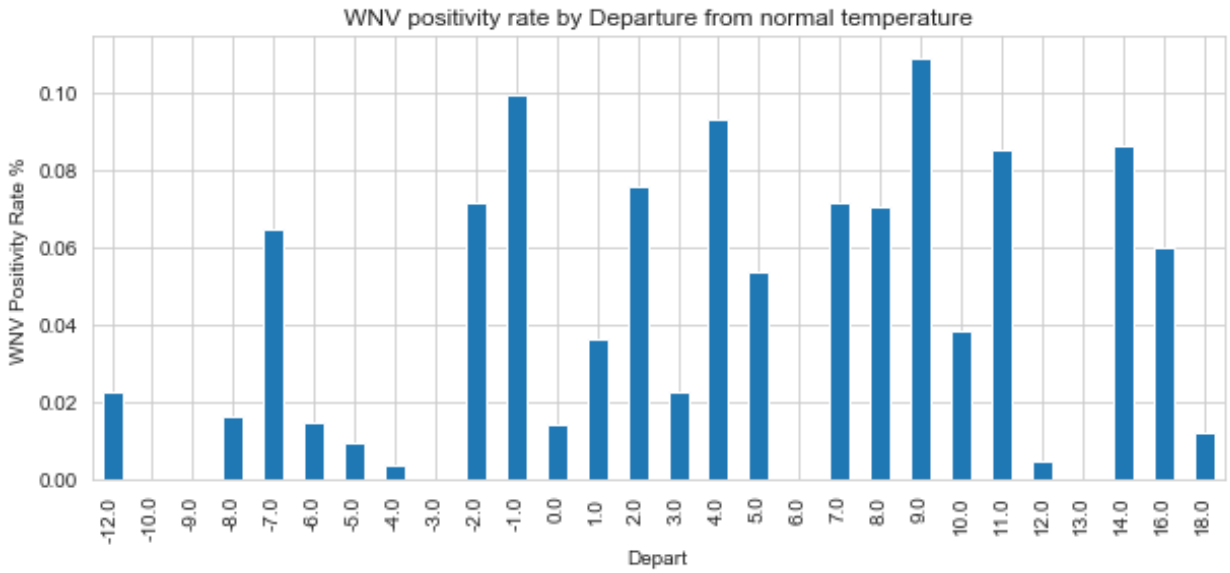

Figure 9: WNV positivity rate by Departure from normal temperature

The above two figures illustrate that as temperature increases the positivity rate for WNV also increases. This could be explained by the fact that hot and dry conditions are more favorable for WNV than cold and wet.

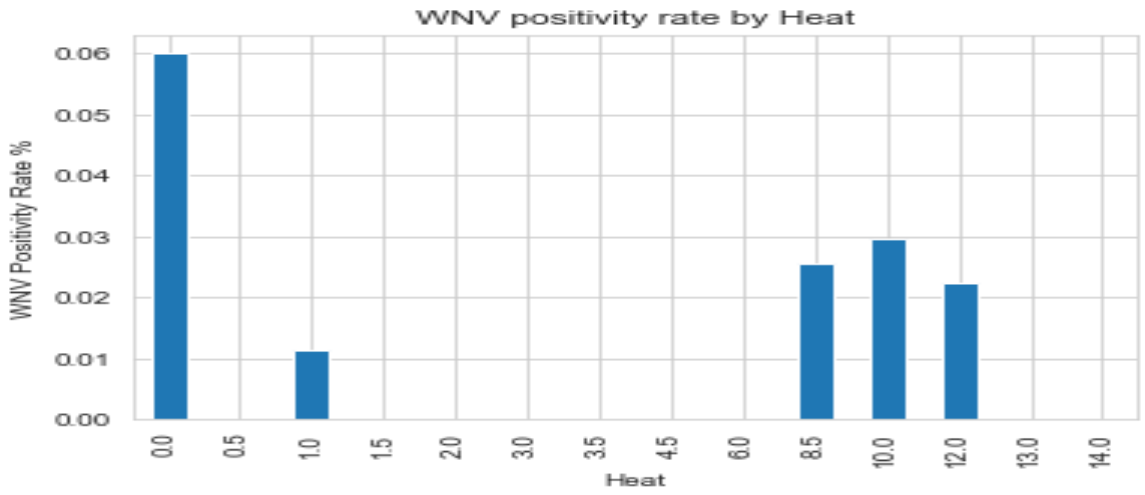

Figure 10: WNV positivity rate by Heat

WNV positivity is associated with low heating days and high cooling days. Low heating days and high cooling days are an indication of hot weather which is favorable to the WNV growth. Majority of the WNV positivity rates occurred at 0.00 heating degree days.

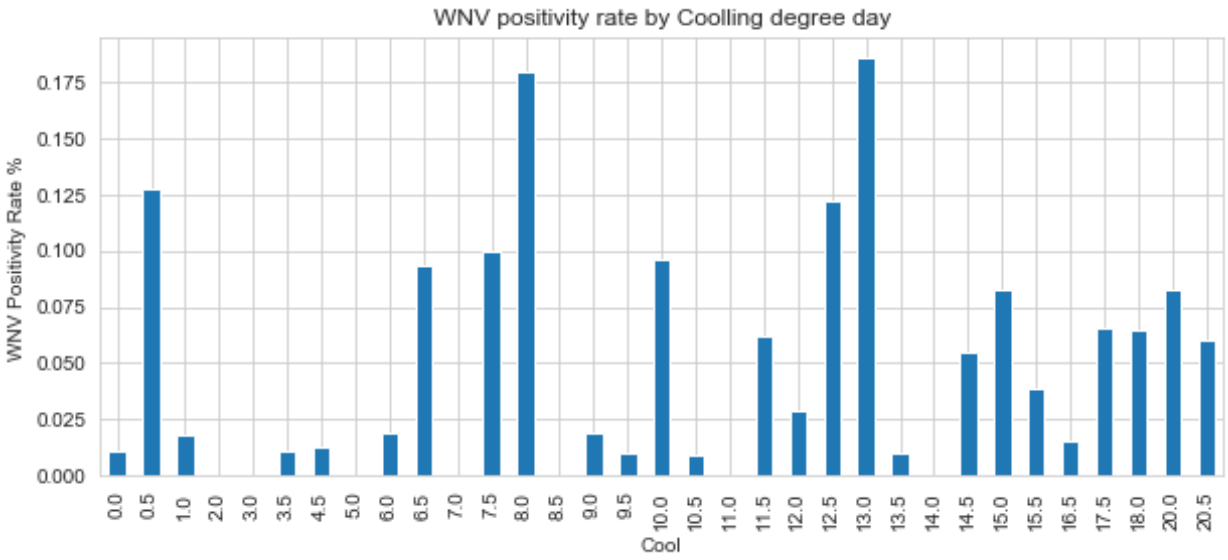

Figure 11: WNV positivity rate by Cooling degree day

The distribution of WNV positivity rate over 'DewPoint' and 'WetBulb' is similar. This may indicate these two variables may be highly correlated. Besides, the positivity rate for

WNV increases with both 'DewPoint' and 'WetBulb'. The possible explanation for the association between Wet bulb and WNV is that, Wet bulb is an indicator of evaporation rate, thus decreases in moisture, thus less moisture tends to be favorable for WNV.

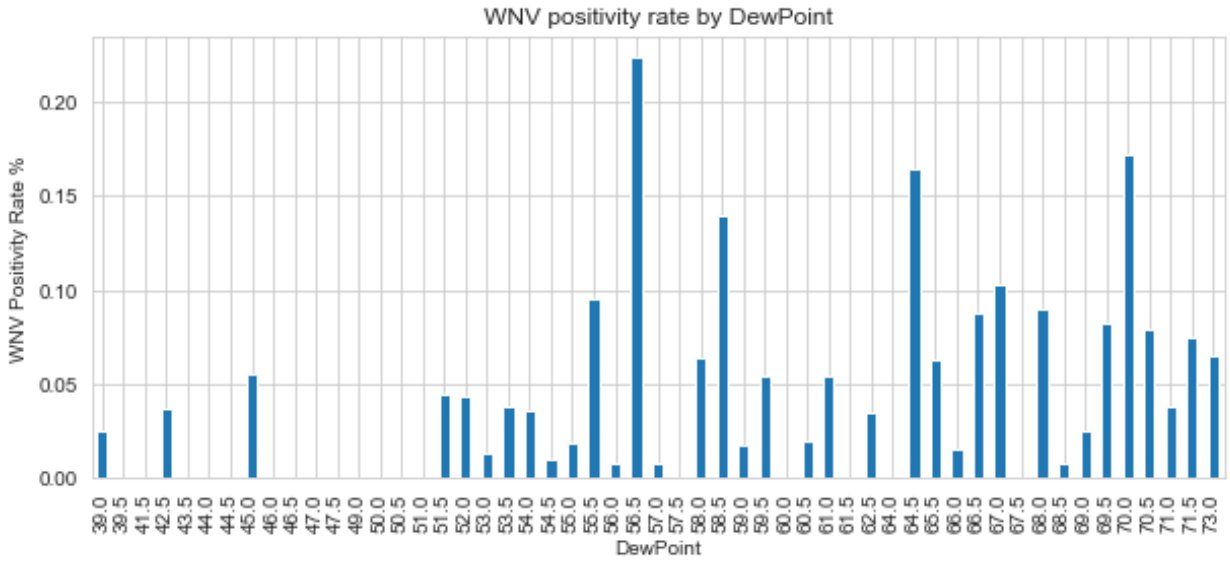

Figure 12 WNV positivity rate by Dew Point.

The WNV positivity rate increases slightly with total precipitation, though the data looks bimodal. Precipitation represents an increase in moisture, which can be favorable weather condition for the growth of mosquito population.

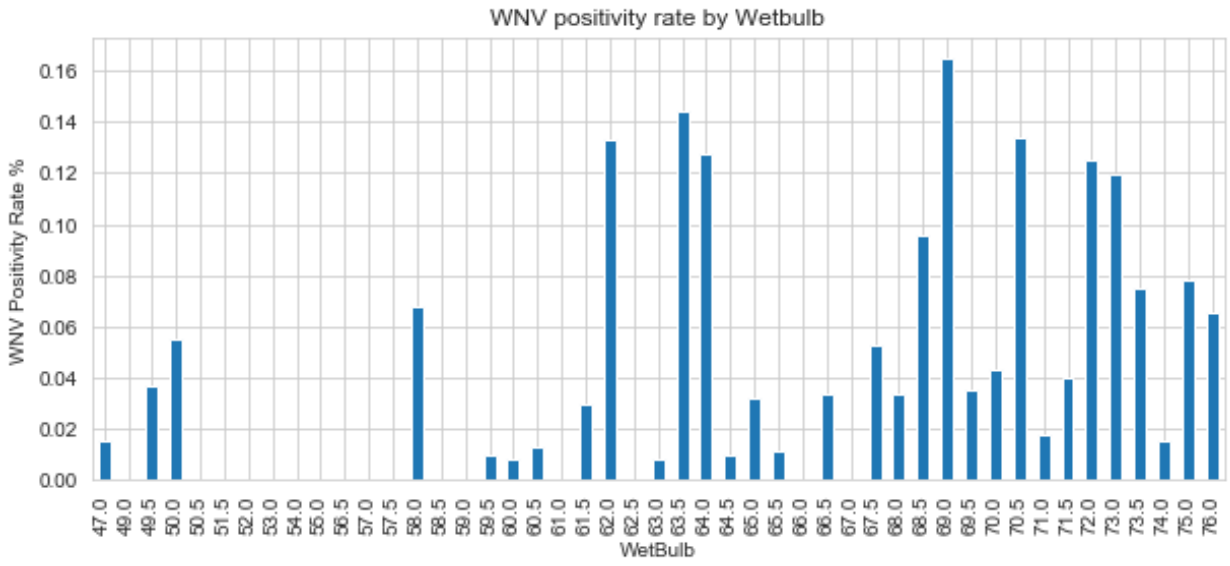

Figure 13: WNV positivity rate by Wet bulb

3.5 Outliers Handling

Data values that are either less than the 5th percentile or above the 95th percentile are removed from the dataset. We removed 107 rows that lies in the mentioned range. We used the boundary of 5th percentile and 95th percentile because it is the maximum interval that includes all the WNV positives. We tried to use different intervals such as 25th and 75th percentile, 10th and 90th percentile, but all of them excludes all the 551 WNV positives which in turn would make prediction impossible.

3. Modeling

Out of the 7 mosquito species, only three species (*CULEX PIPIENS/RESTUANS* (262), *CULEX PIPIENS* (240), & *CULEX RESTUANS* (49)) were carrying WNV. The remaining

four species have null value for WNV positivity. Hence, we will be forced to remove these four species as they have no relevance to our prediction algorithm. As a result, 865 observations are filtered out and our data frame contains 10085 rows, and 51 columns.

Since the data is already labeled (0 for WNV not present, and 1 for WNV present) a supervised machine learning algorithm is a perfect choice to build a predictive model. Moreover, since there are only two outcomes (or classes) in the data (0 and 1), we use binary classification algorithms. The models are trained using the 70% of the data and the remaining 30% is used to evaluate the performance of the models. Since the proportion of WNV positives out of the total 10506 tests is 5.45%, our original dataset is highly imbalanced. Mostly, all standard algorithms are not well suited for learning with highly imbalanced data. There are many methods of handling imbalanced dataset. We used under sampling method with ratio of 1 WNV positive to 4 WNV negatives to get a total of 551 WNV positive and 2204 WNV negatives. We used eight different classification algorithms to build predictive models by taking the imbalanced data issue into account.

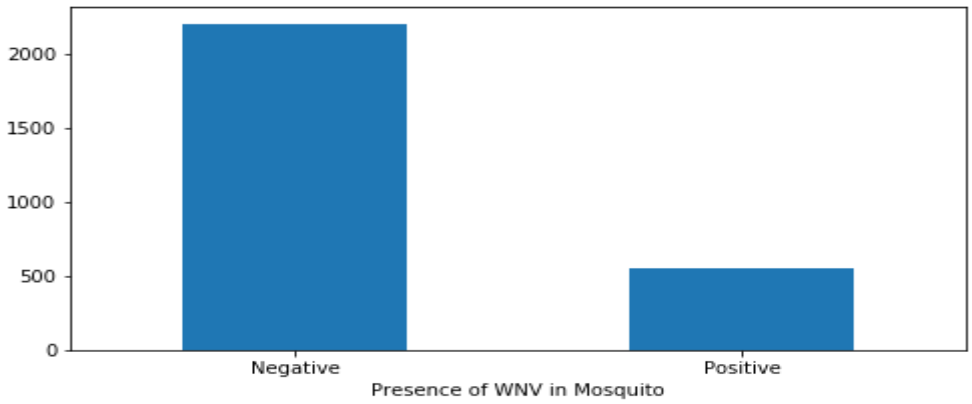

Figure 14: Total WNV Negatives and Positives

4.1 Feature Engineering

Our dataset is a time series data, and there are 16 variables in our weather data that needs feature engineering. Hence, we created lag variable starting from 1 up to 14 days for each of the 16 weather variables. After creating the lag variables, we ended up having a total of 248 columns. We are having missing values as a result of creating the lag variables and we handled it by filling them by their respective column median. We chose median instead of mean, because all the weather variables are skewed.

4.2 Data Pre-processing

Before feeding the data into any machine learning algorithms, there are some preprocessing steps that must be performed on the data. We outline these steps below.

4.2.1 Handling Categorical variable:

To run the different machine learning algorithms on this data, we would have to convert all non-numeric features into numeric ones. The feature 'Species' is the only categorical variable in our dataset, and we converted it into numeric dummy/indicator by using pandas 'get_dummies' method. This also increased the number of columns from 248 to 250 as the categorical column Species has the categories.

4.2.2 Data splitting:

The second step involves splitting the label encoded dataset into train and test datasets. In this project we split them in to 70%-30% ratio.

4.2.3 Scaling:

For some algorithms, it is necessary that we scale the values of all features to lie within a fixed range. We scaled features by using standard scaler such that all features have values between 0 and 1. We applied scaling to KNN and SVM only.

4.3 Feature selection

4.3.1 Attribute Relevance Analysis

Information value is one of the most useful techniques to select important variables in a predictive model. It helps to rank variables on the basis of their importance. There are some prerequisites that need to be fulfilled beforehand. First, the data must be clean that there shouldn't be any missing value. We fulfilled this criterion as we have no missing value. Second, there should not be any continuous attributes. We fixed this by binning all continuous attributes in to five or two bins depending on the value of the attribute. We also made sure that each bin has at least 5% of the observations. Finally, we calculated the IV score and got 199 variables with IV-score 0.01 - 0.8.

4.3.2 Correlation with the target variable

We have done correlation matrix between all the features and the target variable. As a result, we got 81 features with correlation coefficient of greater than 0.11. Most of the features has weak correlation of less than 0.2 correlation coefficient with the target variable.

4.3.3 Variance Inflation Factor (VIF)

Variance Inflation Factor (VIF) is used to detect the presence of multicollinearity among predictor variables. Variance inflation factors (VIF) measure how much the variance of the estimated regression coefficients is inflated as compared to when the predictor variables are not linearly related. We calculated VIF for those 234 features with either IV-score of 0.01 - 0.8 or with correlation coefficient of greater than 0.11. Out of which, we got eight features with VIF <= 5.

4.4 Modeling Pipeline and Evaluation Metric

Once the data is pre-processed, we feed them to classification algorithm to build the model. In order to evaluate the performance of the model, we test the model on the test dataset. Before making predictions on test dataset, we use the exact same pre-processing steps (except for scaling) that we used for training dataset and apply them on the test dataset. We used a class called pipeline from Python's scikit learn library, to combine all the steps including the scaling, and classifier learning steps into one. This pipeline is then applied directly on the test dataset. Furthermore, we used Grid Search with 5-fold cross-validation, in order to apply hyperparameter tuning for the classifiers.

Regarding evaluation metric, we were cautious in selecting the proper metric as we have an imbalanced data. Accuracy is not a good metric for such datasets. We want to have high true positive rate (or recall) with WNV present tagged as positive class. At the same time, we do not want lots of false positives or less precision. Most of the time, the choice of a good metric depends on business needs. In this project, we keep in mind all elements of a confusion matrix. Also, we want a metric which is threshold invariant, so

F1-score is also not a great choice. Thus, area under the curve (AUC) of receiver operating characteristic (ROC) curve and Log Loss will be our two primary evaluation metrics for our imbalanced data set.

4.5 Logistic Regression

After testing in the 30% holdout data, this algorithm led to a very poor model. Its ROC AUC score is 0.5379 which is a little better than the one predicted by chance or luck. In addition, the Log Loss for the logistic regression is 0.4443 which is somewhat larger as compared to the other models. Upon trying a different metric (such as F1-score and recall) for optimization we obtained non-zero values for precision & recall; and its accuracy and f1 score is 0.7838 and 0.72 respectively. The ROC curves are shown in Fig 15.

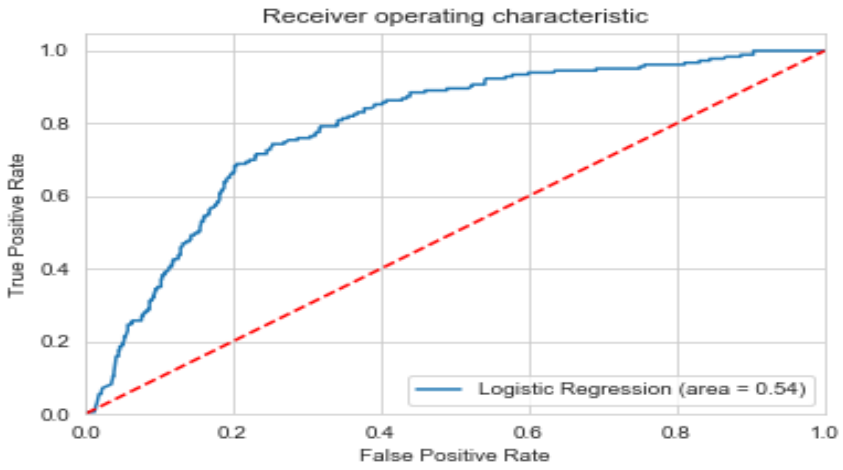

Figure 15: ROC AUC curve for Logistic Regression Classifier

4.6 K Nearest Neighbors (KNN)

For KNN we used scaling to standardize the data as well as grid search for selecting the optimal number of neighbors. Using number of neighbors (neighbors=1) means each

sample is using itself as reference, that's an overfitting case. For our data, the best value selected for the number of neighbors is 10. Apart from the number of neighbors, we optimized the 'P' which is the power parameter for the Minkowski metric. When p=1, this is equivalent to using manhattan_distance(l1), and euliddean_distance(l2) for p=2. For arbitrary p, minkowski distance (l_p) is used. In most cases, the choice is always between l1 and l2 but it's interesting to see the results of higher minkowski distances. For our data, using l1 seems to be better than l2 and other l_p distances.

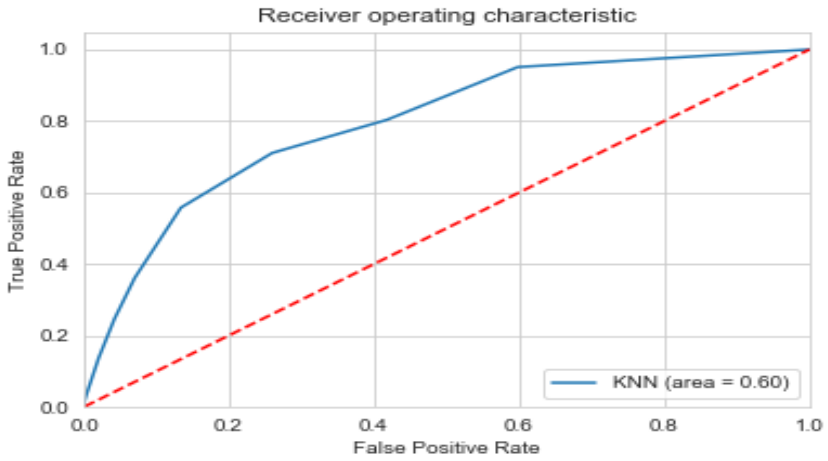

Figure 16: ROC AUC curve for KNN classifier

It has better ROC AUC score of 0.6019 and accuracy of 0.8004 as compared to logistic regression. However, it has the worst Log Loss of 0.7573 when compared to all the rest seven classifiers. Its F1-Score is 0.76. The ROC curves are shown in Fig 16.

4.7 Random Forest Classifier

Random forest models reduce the risk of overfitting by introducing randomness by building multiple trees (n_estimators), by drawing observations with replacement (i.e., a

bootstrapped sample), and by splitting nodes on the best split among a random subset of the features selected at every node. Thus, the random forest model has many hyperparameters to be optimized. Most data scientists see *number of trees*, *tree depth* and *the learning rate* as most crucial parameters. Since there is no learning rate in random forest, we optimized both the *number of trees*, *tree depth* hyperparameters *u*sing 5-fold cross validation. For the random forest model, we do not have to scale the features and so we skipped the scaling step of the pre-processing section.

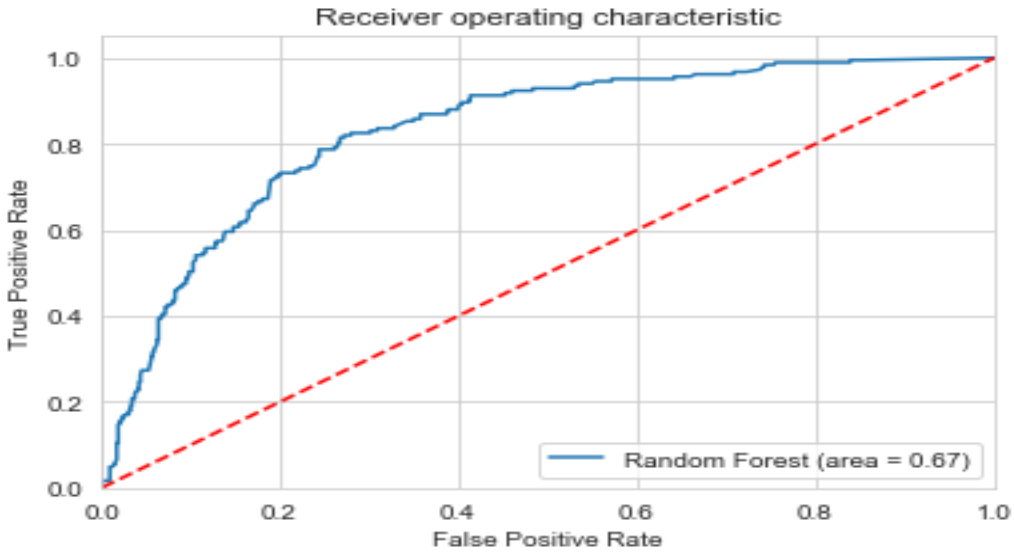

Figure 17: ROC AUC curve for Random Forest Classifier

The best parameters selected by the grid search are 100 for number of trees, and 13 for the maximum number of levels in each decision tree. The ROC AUC score and Log Loss for this algorithm is 0.6746 and 0.4513. This algorithm has also better accuracy of 0.8162 and F1_score of 0.80. The ROC curves are shown in Fig 17.

4.8 Support Vector Machine (SVM)

Support Vectors Classifier tries to find the best hyperplane to separate the different classes by maximizing the distance between sample points and the hyperplane. This is the second algorithm that needs scaling. Hence, we applied scaling to standardize the data, as well as grid search to tune the hyperparameters. The optimum hyperparameters obtained from the grid search are (C=1, gamma=0.1, and kernel = linear). The ROC curves are shown in Fig 18.

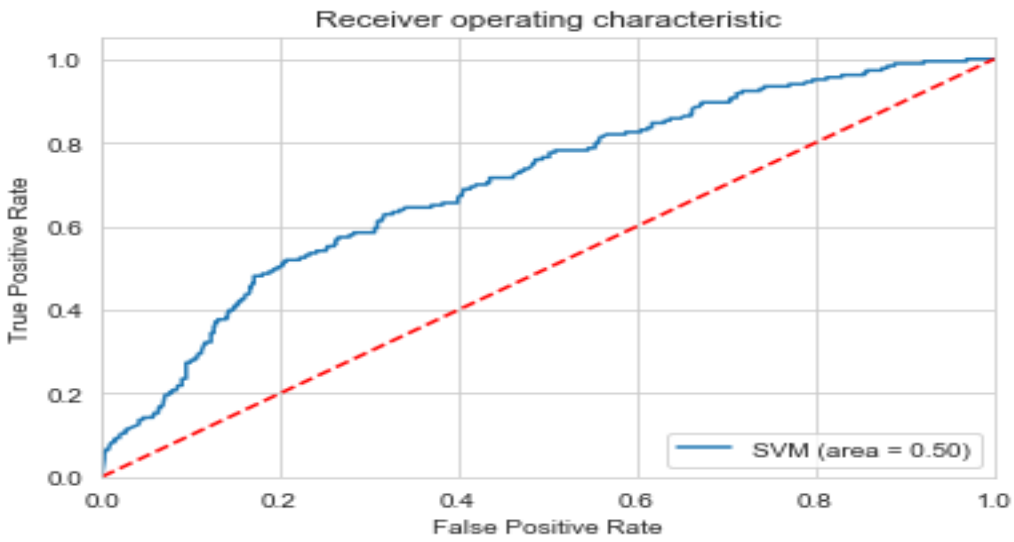

Figure 18: ROC AUC curve for SVM classifier

This is the worst algorithm that produced ROC AUC score 0.5 which is exactly the same as the one predicted by chance or luck. On top of that, the F1_score, precision and recall for WNV present is 0 indicating poor model. The Log Loss for this algorithm is also bad (0.503). This algorithm has an accuracy of 0.7787 and F1_score of 0.68.

4.9 Extremely Randomized Trees (Extra Trees Classifier)

Extra Trees Classifier is an ensemble learning method fundamentally based on decision trees. Extra Trees Classifier, like Random Forest, randomizes certain decisions and subsets of data to minimize over-learning from the data and overfitting. In addition, Extra Trees is like Random Forest, in that it builds multiple trees and splits nodes using random subsets of features, but with two key differences: it does not bootstrap observations (meaning it samples without replacement), and nodes are split on random splits, not best splits. In Extra Trees, randomness doesn't come from bootstrapping of data, but rather comes from the random splits of all observations. In ET, we don't need to use feature scaling, but we used grid search for hyperparameter tuning.

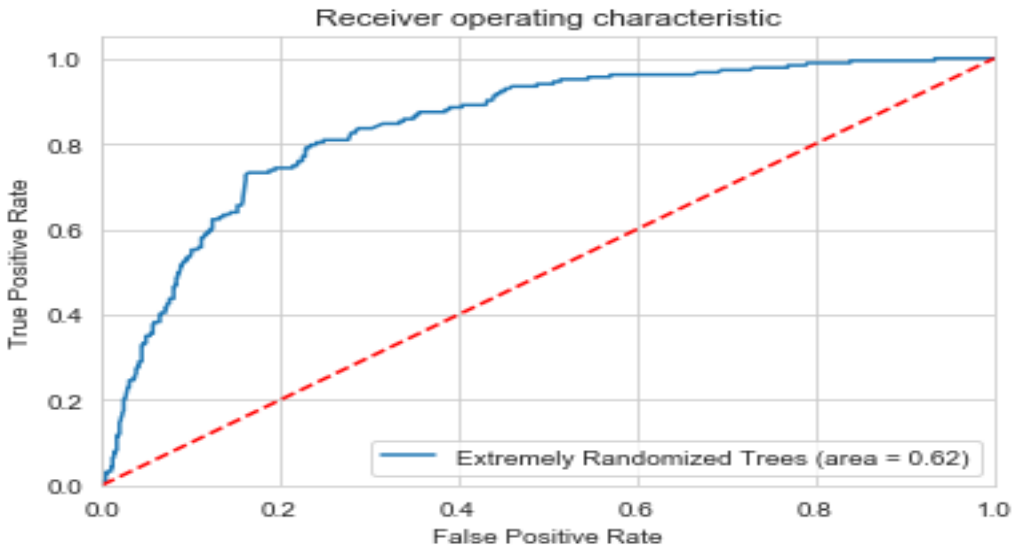

Figure 19: ROC AUC curve for Extremely Randomized Trees classifier

The best parameters selected by the grid search were 100 for number of trees, 14 for the minimum number of data points placed in a node before the node is split, 13 for the maximum number of levels in each decision tree. The ROC AUC score and Log Loss for this algorithm is 0.6172 and 0.3968. This algorithm has also better accuracy of 0.8089 and F1_score of 0.78. The ROC curves are shown in Fig 19.

4.10 AdaBoost Classifier

Ada-boost or Adaptive Boosting is an iterative ensemble method. It combines multiple classifiers to increase the accuracy of classifiers. AdaBoost classifier builds a strong classifier by combining multiple poorly performing classifiers so that you will get high accuracy strong classifier. The basic concept behind Ada-boost is to set the weights of classifiers and training the data sample in each iteration such that it ensures the accurate predictions of unusual observations.

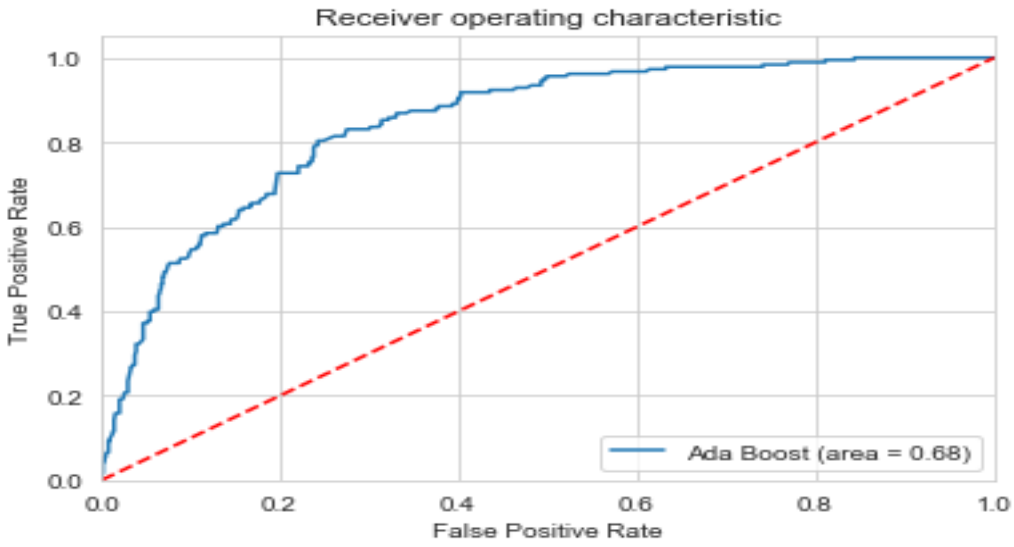

Figure 20: ROC AUC curve for Ada Boost classifier

For this algorithm also, we do not need to use any feature scaling. We carried out hyperparameter tuning for the number of trees and the optimum number of trees is 200. The ROC AUC score and Log Loss for this algorithm is 0.6812 and 0.6865. The Log Loss for of this algorithm is large. Th This algorithm has also better accuracy of 0.8234 and F1_score of 0.81. The ROC curves are shown in Fig 20.

4.11 Gaussian Naïve Bayes

Naive Bayes is a statistical classification technique based on Bayes Theorem. It is one of the simplest supervised learning algorithms. Naive Bayes classifier is the fast, accurate and reliable algorithm. Naive Bayes classifiers have high accuracy and speed on large datasets. Any Naive Bayes algorithm assumes that the features are independent for a given class. Violating this assumption may result in poor metrics. Therefore, we can try to look at correlations amongst features and remove the ones that are highly correlated. However, we don't have a feature that have correlation coefficient values closer to 1. All the eight features have correlation coefficient values of < 0.5. We removed the top 3, and top 1 features with highest correlations and trained the model again. Removing features did not help in improving the results. The ROC curves are shown in Fig 21.

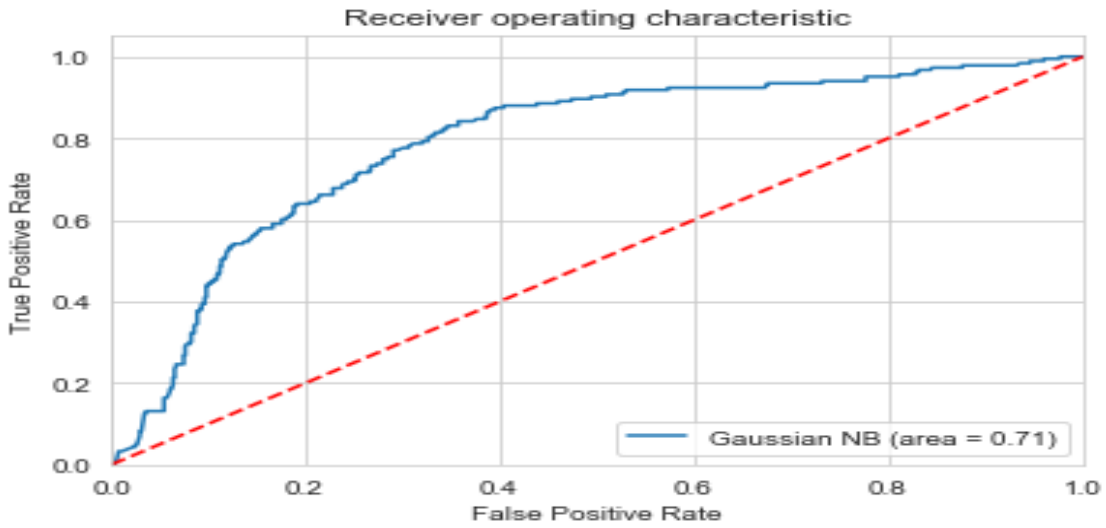

Figure 21: ROC AUC curve for Gaussian Naive Bayes classifier

We didn't use scaling and grid search for Gaussian Naive Bayes classifier. We got better metrics score from this algorithm. This classifier predicted the best ROC AUC score of 0.7088 among all the classifiers. However, its Log Loss (0.5561) is somewhat larger. The accuracy and F1 score of this classifier are 0.7811 and 0.79 respectively.

4.12 Gradient Boosting

Gradient Boosting is an ensemble boosting method based on using weak learners (almost always decision trees) trained sequentially to form a strong model. It's obvious that rather than random guessing, a weak model is far better. In a boosting, algorithms first, divide the dataset into sub-dataset and then predict the score or classify things. Then it again divides the remaining misclassified datasets into sub data and so on. Unlike in the random forest, it learns from its mistakes in each iteration. It means that in a random forest, all the trees are independent, but in the case of boosting each successive model learns from the mistakes from the ones before it. There are many hyperparameters in a Gradient Boosting controlling both the entire ensemble and

individual decision trees. We do not need to use feature scaling for Gradient Boosting as well, but we used grid search for three hyperparameters.

The optimum hyperparameters selected by the grid search are 150 for number of trees, 2 for maximum number of levels in each decision tree, and a learning rate of 0.15. Despite the optimum maximum number of levels in each decision tree is 2, we got better results when we increase it to 3. Thus, we decided to increase the maximum number of levels in each decision tree parameter to 3. This is the algorithm that has the lowest Log Loss of 0.3816 and the second-best ROC AUC score of about 0.7043 next to Gaussian naïve Bayes. It has also the best accuracy score of 0.8319 and F1-Score of 0.82. Overall this is the finest classifier that has the best evaluation metrics as compared to the rest of the seven classifiers. The ROC curves are shown in Fig 22.

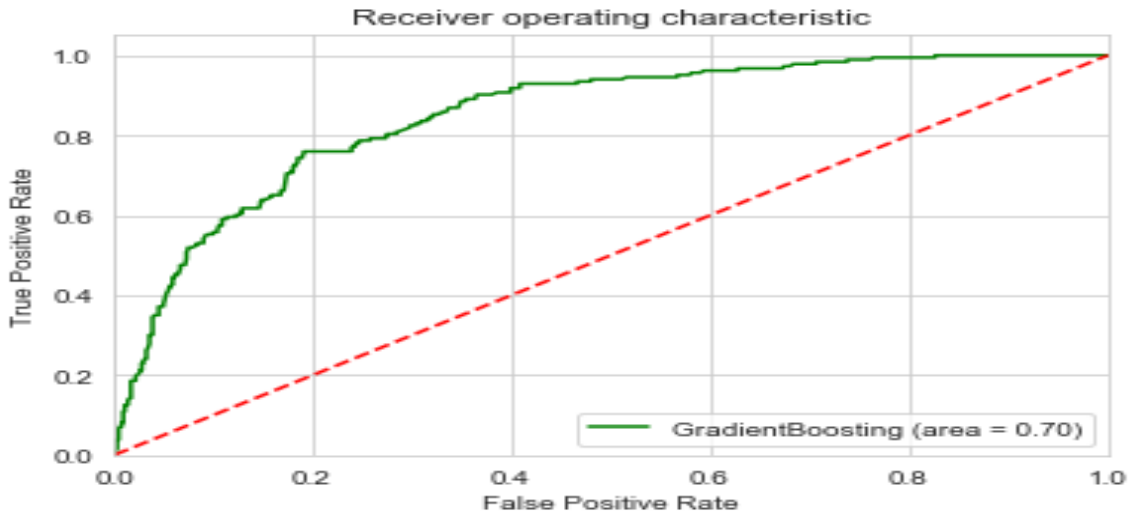

Figure 22: ROC AUC curve for Gradient Boosting Classifier

With all the optimized hyperparameter we can calculate the feature importance. Figure 23 shows all the eight features with their respective feature importance's. 'Number of

Mosquitos' is the strongest feature that predicts the presence or absence of WNV in mosquito; followed by Resultant wind Speed & Depart (lag_14). While 'Heat' is the weakest feature in classifying the presence or absence of WNV in mosquito.

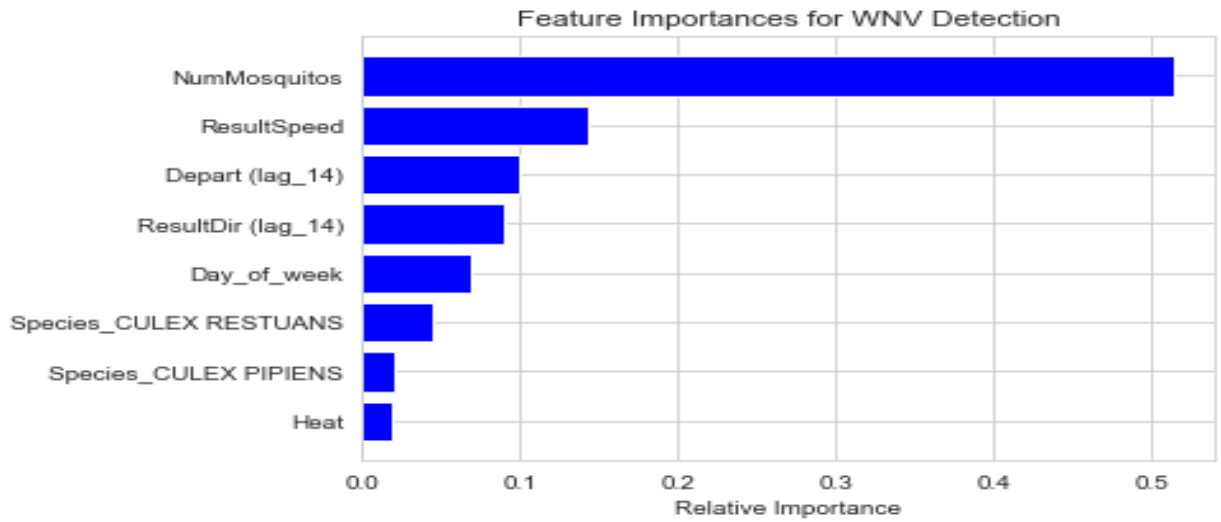

Figure 23: Feature Importance of all the eight features from Gradient Boosting Model

4.13 Model Comparisons

We have used Logistic Regression, Gaussian Naive Bayes, Random Forest, Gradient Boosting and Extra Randomized Trees classifiers to build a model to predict WNV positivity likelihood. Based on testing the models on the holdout dataset (50% of the whole data), we found different performance of all models. We used ROC AUC, Log Loss, F1 score, and accuracy in order to compare the eight algorithms. For good models the value for the Log Loss should be close to zero whereas for ROC AUC, F1 score, and accuracy the value should be close to one. Out of the eight classifications algorithms we used, Gradient Boosting, Random Forest, Gaussian NB, and Ada-Boosting are the four best classifiers with best metrics. Though the Gaussian NB

Classifier has the best ROC_AUC score (0.7088), its Log Loss is larger than that of Gradient Boosting Classifier and Random Forest Classifier. Therefore, Gradient Boosting Classifier is the one with the lowest Log Loss, best accuracy score and second-best ROC AUC score (0.7043) which makes it the best algorithm among the eight classifiers we used. The results of various evaluation metrics scores for all models are shown in Tab. 1.

Table 1: Top four best performer models in descending order.

Model	**ROC AUC**	**Log Loss**	**Accuracy**	**F1-Score**
Gradient Boosting	0.7043	0.3816	0.8319	0.82
Random Forest	0.6746	0.4513	0.8162	0.80
Gaussian NB	0.7088	0.5561	0.7811	0.76
Ada-Boosting	0.6812	0.6865	0.8234	0.81
Extremely Randomized	0.6172	0.3968	0.8089	0.78
KNN	0.6019	0.7573	0.8004	0.76
Logistic Regression	0.5379	0.4443	0.7838	0.72
SVM	0.500	0.5034	0.7787	0.68

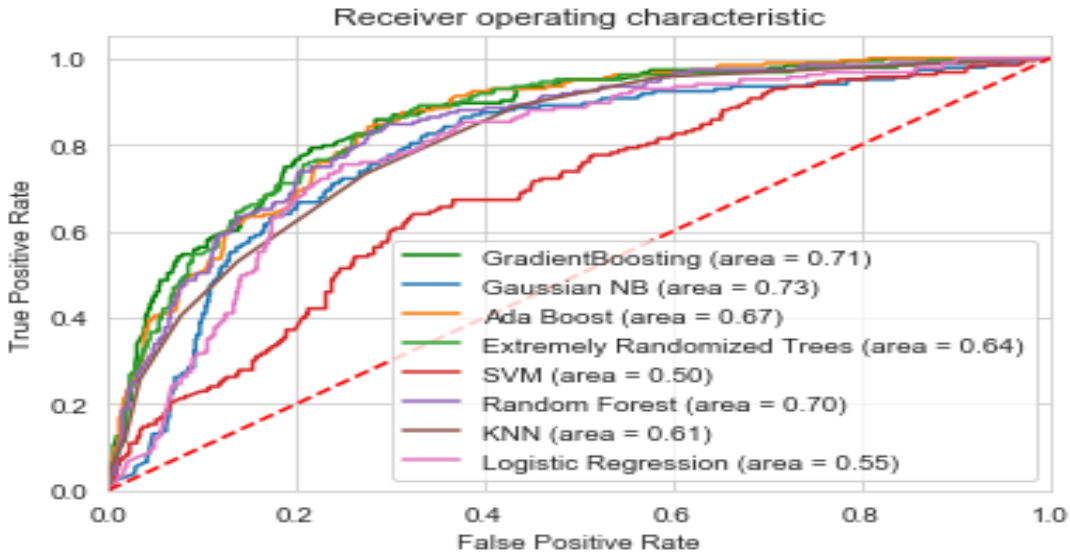

Figure 24. Model Comparisons by ROC AUC Curves

4. Using Model and Recommendations

In order to make predictions on new data, one would need to filter out some features, then perform pandas get dummies method on categorical feature (Species variable) and take only the eight important features (stated in Tab. 1). Table 2 contains a list of all the eight features that we need to feed to the Gradient boosting classification model. The table is sorted by the feature importance values obtained from the Gradient Boosting model, with highest on the top. Features such as Depart (lag_14), ResultDir (lag_14) are features that are we engineered from the 'Depart', and 'ResultDir' variables of the weather dataset respectively. We also extracted the 'Day_of_week' feature from the date variable of both datasets. So, one would need to calculate these features beforehand. Once the features are ready Gradient Boosting model can be run on the new dataset and predictions can be made. Finally, by doing so one can achieve

remarkable results. Thus, the City of Chicago and the Chicago Department of Public Health (CDPH) can apply this model and as a result will be able to identify when and where the city will spray airborne pesticides to control adult mosquito populations, thereby significantly reduce morbidity and mortality due to the disease.

Our goal is to predict where in Chicago WNV occurs to help the city prepare accordingly. The City of Chicago and the Chicago Department of Public Health (CDPH) can use such a model to get information on when and where the city will spray airborne pesticides to control adult mosquito populations. This in turn will play a significant role in the prevention and control of the disease.

Table 2: Description of the final set of features for the Gradient Boosting model

Features	Source type	Data type	OHE required	Description
NumMosquitos	main data	Numerical	No	Number of mosquitoes caught in trap
ResultSpeed	Weather data	Numerical	No	Resultant wind speed
Depart(lag_14)	Weather data (Engineered)	Numerical	No	Departure from normal temperature (14 after)
ResultDir(lag_14)	Weather data (Engineered)	Numerical	No	Resultant wind direction (14 after)
Day_of_week	date variable	Numerical	No	Day of week
Species_CULEX RESTUANS	main data (Species)	Categorical	Yes	CULEX RESTUANS species of mosquito
Species_CULEX PIPIENS	main data (dummy variable of Species)	Categorical	Yes	CULEX PIPIENS species of mosquito
Heat	Weather data	Numerical	No	Heat

5. Assumptions and Limitations

Since our dataset is time series data, it is likely that there are some correlations among our features. In this project in order to implement models such as Gaussian Naïve Bayes model, we assume that all the features are independent, i.e. there are no correlations. Other than this assumption, we also have some limitations in the data which might have reduced the robustness of the machine learning model that we developed. In order to make a prediction on new data, the prediction model is dependent on how good the prediction of the weather will be after certain days. This means that the model will predict better if the weather prediction is better. Furthermore, we are unable to use the spray data as there is not enough information about the sprays used to help guide our project. Incorporating this data could have played a role in improving the model.

6. Conclusions

We explored the main dataset (WNV test data) merged along with the weather dataset and some engineered weather data to determine their impact on WNV test positivity on mosquitoes. The overall WNV positivity rate was 5.45%, indicating an imbalanced data. After doing exploratory data analysis, and preprocessing we used eight different supervised classification algorithms (Logistic Regression, KNN, SVM, Random Forest, Extremely Randomized Trees, Ada-Boosting, Gaussian Naive Bayes, and Gradient Boosting) to train the predictive model by using 70% of the whole data. The remaining

30% was used to evaluate the model. We found that the Gradient Boosting classifier gives the best model performance with ROC_AUC score of 0.70, Log Loss of 0.38, accuracy of 0.83. In terms of features, we only need eight features to make the prediction. Of which 1 from the main data, 2 from weather data, 2 engineered from the weather data, 2 dummy variables of a categorical variable from the main data, and 1 extracted from the date variable of the merged data. We believe that the model is fair enough to predict WNV detection. However, this model can be further improved by overcoming the aforementioned limitations.