West Nile Virus Prediction Using Machine Classification

Frank Hiemstra
Dept of Electrical and Computer Engineering
University of Virginia
Charlottesville, VA 22903
fhh7ph@virginia.edu

Andrew Norton
Dept of Computer Science
University of Virginia
Charlottesville, VA 22903
apn4za@virginia.edu

Abstract

Chicago has seen an outbreak of the West Nile virus, a virus primarily spread to humans through disease-ridden mosquitoes. The aim of this project is to help the Chicago Department of Public Health (CDPH) predict where and when West Nile virus should be expected to be present in mosquitos based on historical data. In the past this task has been dealt with by using a statistical and mathematical model, which examines trends in weather data. Other researchers have used a machine learning approach, but stuck to linear regression, regression trees, and random forests. The teams solution involves using a Support Vector Classifier (SVC) to predict where and when West Nile virus will be present in the city of Chicago. With our SVC, we achieved a cross validation accuracy of 56% and a test accuracy of 65%. When compared with other classification techniques, the SVC outperformed the next best classifier by over 10%. The results of this project have the capability of containing West Nile virus and protecting the residents, and tourists of the city of Chicago.

1 Introduction of Target Task

Since 2002 Chicago has seen an outbreak of the West Nile virus, a virus primarily spread to humans through disease-ridden mosquitoes. The severity of this epi- demic is as serious as the consequences of being infected with the virus, which include but are not limited to a persistent fever, serious neurological illnesses, or even death. The first outbreaks recorded were in Israel in the 1950s. Since the first sightings of the virus in Chicago in 2002, the 2.72 million Chicago residents have been very concerned. However, infecting the city has much larger ramifications than harming only its citys residents. Chicago estimated it had 50.2 million people from around the globe tour the city in 2014. This magnifies West Nile virus from a city concern to a serious global threat. People visiting the city could become contaminated with the virus in Chicago and bring it back to their prospective homeland. The Chicago Department of Public Health (CDPH) has established a comprehensive surveillance and control program to combat the spread of the virus. This program involves surveying selected areas of the city, where samples of mosquitoes tested for the virus. Next, the city then sprays infected areas to eliminate these virus-carrying mosquitoes. Not all mosquitoes in Chicago carry the West Nile virus, so targeting which areas of the city have infected mosquitoes, and at what time, is an important task. Weather conditions, testing data from the CDPH program, and spraying data are all used in this project to predict when and where different species of mosquitoes will test positive for the West Nile Virus. The ability to predict this is crucial to not only city of Chicago residents, but also people around the world, as people come to the city for business and tourism. This project has the capability to give strategic, statistical-based direction for which areas in Chicago and at what time the CDPH should spray chemicals to kill mosquitoes. The task will be accomplished using regression and classification

In 2004 the Chicago Department of Public Health (CDPH) initiated a program to fight the virus. The program includes testing mosquitoes in various sites around Chicago for the virus, as well as spraying combative chemicals to eliminate these disease-carrying insects. Given this data from the past, along with NOAA weather data this project aims to give the CDPH a model to predict at what time different areas in Chicago will have mosquitoes infected with the virus. The weather has proven to be an indicator of West Nile virus, making this data valuable. This project aims to yield results that will stop the spread of West Nile virus from mosquitoes to people living in or visiting the city of Chicago by predicting the location and times of occurrences of West Nile virus in mosquitoes. Seeing that the city of Chicago is over 200 square miles, even a prediction with this accuracy would of far more practical use than simply guessing.

Predicting West Nile virus lends itself to falling into the medical and biological sciences fields. When dealing with diseases, particularly diseases which can impact humans, being able to predict their presence can lead to death or life as stated before. We took this into account in our prediction classification by sacrificing precision, in order to do our best to detect WNV. We believe it is far better to have a false positive, meaning we predict WNV is present when it is not, rather than a false negative, meaning we predict WNV is not present when it actual is present. When dealing with disease and human life, we believe this is a necessary sacrifice.

2 Previous work

The National Center for Atmospheric Research in Boulder, Colorado, released a report earlier this year associating meteorological conditions and incidence of West Nile virus in the United States [Hahn, *et al.*]. Without using machine learning, they looked at how variations in temperature or precipitation alter the likelihood of increased cases of West Nile virus. As a study with nation-wide scope, this "broad strokes" approach is useful for creating rough estimates of mosquito activity over large time intervals. However, applying machine learning to this problem could improve the specificity of their predictions.

In a more location-targeted study, the Geographic Information Systems and Spatial Analysis Laboratory at University of Illinois at Urbana-Champaign attempted to predict the spread of West Nile virus in the Chicago metropolitan area [Ruiz, *et al.*]. In their 2010 paper, they used linear regression, regression trees, and random forests to create a statistical model of the mosquito infection rate per week over the 2004-2008 timeframe. From 2011-2014, they used this model to predict mosquito infection rate. Earlier this year, Kaggle ran a competition focused on West Nile virus prediction in Chicago, based on a similar (but more current) dataset.

Another study, done by Kaoru Tachiiri et al, used a raster-based mosquito abundance model using basic geographic and temperature data. Some of the parameters they included in the model are spatial factors determined from the locations of their mosquito traps. They use this mosquito prediction in order to predict humans contacting the virus in British Columbia. Similarly to the first study discussed, applying machine learning to their problem could show large improvements.

The researchers from the National Center for Atmospheric Research (NCAR) and the Centers for Disease Control and Prevention (CDC) published a journal entry in The American Journal of Tropical Medicine and Hygiene, where they reveal that U.S. West Nile virus outbreaks occur at a higher incidence when temperatures in the previous year were above average. This is very interesting insight into the issue we are dealing with in this paper, which came from pure statistical analysis. The researchers came to this conclusion by conducted analysis of weather patterns pertaining to disease in various counties across the US from 2004 to 2012.

3 Relevance to machine learning

This problem, as we will approach it, is inherently rooted in machine learning. We aim to predict when and where mosquitoes carrying the West Nile virus will appear in Chicago based on historical data. We construct an algorithm that builds a model, which is learned from training data, and is used to make predictions from testing data. While one could approach this topic using more pure mathematical modeling or a simplified statistical model, machine learning will allow us to utilize far more data and create much more precise predictions. The problem we have at hand is classified

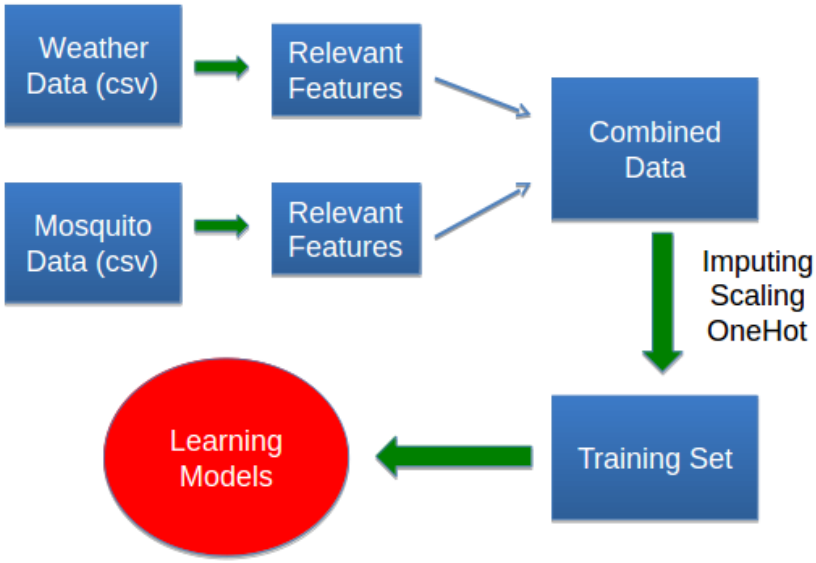


Figure 1: Data Processing Pipeline

as supervised learning, where we are performing binary classification, being a mosquito trap that contains West Nile virus, or does not contain West Nile virus.

4 Proposed method

The task at hand is supervised learning; in particular, we are facing a classification problem; given the weather history and disease test results for a set of areas, classify each area as containing or not containing West Nile Virus. As such, our first approach will be to use a Support Vector Classifier.

4.1 Preprocessing

For this problem, our features will include, but are not limited to: date, location, humidity, pressure, outside temperature, wind speed, past spraying history, past testing history. This task will require a fair amount of preprocessing (see Figure 1). First, some of our features contain categorical data. This means we will need to digitally discretize these features. For the continuous parameters, we will need to use the preprocessing technique of scaling to avoid numerical difficulties during calculations. As a first attempt, we will linearly scale, both the training and testing data, to fit a range of [0,1]. This is beneficial because it decreases the time it takes for gradient descent and other numerical methods to converge, as well as preventing certain features from being inherently weighted simply due to the size of the numbers involved. Better scaling may be achieved by centering the features so they have mean 0.5 and then dividing by the variance of the sample. There are missing values some of our feature data, which means we will need to do preprocessing to assign a value to each. Our first attempt at this will be imputation using nearest neighbor; however, our team would like to explore a more complex EM based approached to this preprocessing. If neither of the above work, we will substitute the missing value with the mean value of the feature.

4.2 Support Vector Classifier

Since we will be classifying our samples into only two distinct classes, we will use simple, binary classification techniques. As stated previously, we will use a Support Vector Classifier (Vapnik, 1995). Due to its simplicity, we started with a Linear Support Vector Classifier.

The performance of the Linear Support Vector Classifiers was compared with SVCs with other kernels using k-fold cross validation. After experimentation, we ended up using the polynomial kernel. After comparing that kernel with the linear and radial basis function kernels, it performed the

best. Next, we decided to experiment with the number of degrees of the polynomial kernel. Adding degrees only hurt the cross-validation accuracy, which led us to staying with only two degrees for the polynomial kernel.

The biggest successes we had in designing our SVC came when tuning the error penalty parameter C, and assigning weights to the classes. Originally, we did not assign weights to the classes. Because of the imbalance in our dataset, this led to poor results. We found a giant leap in cross-validation and test accuracy when weighting the classes based on the size of each in the data. Next, we found we needed a relatively high value for our C parameter; we chose C to be 85. This high value assigns a high penalty for misclassification, which works well for our imbalanced dataset, as well as trying to reduce false negatives.

4.3 Decision tree

In classification contests over the past several years, Decision Tree-based approaches have performed substantively well, but (due to the model complexity) have a tendency to overfit data. Because of the limited number of actual cases of West Nile Virus in our training data (see Evaluation Metrics section), there is a strong risk of overfitting. For this reason, we expect to use this model as simply a member of an ensemble instead of the primary classifier we are considering.

4.4 Brief Results

After experimentation, we decided to not use the ensemble approach. We found most of the classifiers gave an extreme number of false negatives, which will be further discussed in the following sections. Due to this, we found that using the SVC with a polynomial kernel outperformed our ensemble. The SVC was able to handle the imbalanced dataset well compared to the other classifiers.

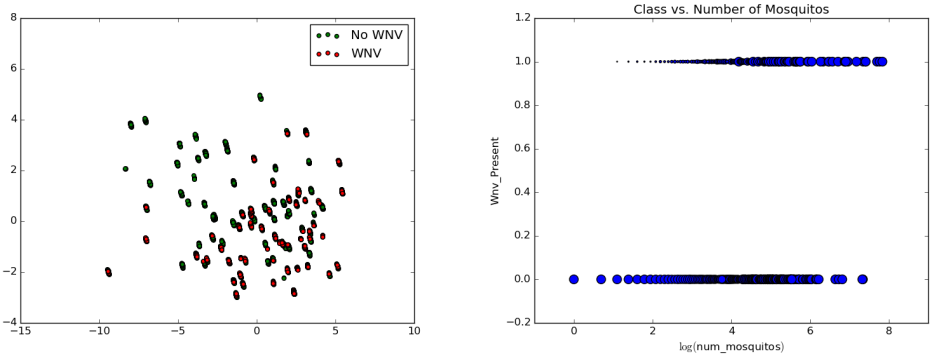
5 Experimental design and data

5.1 Data Source

The data used in this project came from the City of Chicago Department of Public Health. The department released the data via a Kaggle Competition (www.kaggle.com) under their Previous Work competitions. We have been given approximately 10,000 training samples, and 116,000 testing samples. Given the nature of the competition, we are not able to see which class each of the 116,000 testing samples are in. On Kaggles website we are able to upload our predictions for each sample, and they tell us the testing accuracy we have on the testing data.

5.2 Data Details

Of the ten-thousand training data samples we have been given, a few more than five hundred contain West Nile virus, giving us the imbalanced class nature. The data we used is broken up into three categories: weather data, mosquito data, and mosquito spray data. The weather data gives us twenty-one unique weather features per weather station. The features are as follows: Maximum Temperature, Minimum Temperature, Average Temperature, Dew Point, Wet Bulb, Heat, Cool, Sunrise, Sunset, CodeSum, Depth, Water, Snow Fall, Precipitation Total, Station Press, Sea Level Pressure, Resultant Speed, Resultant Direction, and Average Speed. We took weather data from two different weather stations in Chicago, giving us a sum of forty-two features. We decided to create our own feature, which we called Differential Pressure, which took the difference between the station pressure and sea level pressure. The mosquito data simply consisted of the address of the trap, species of mosquito in the trap, block number address of the trap, the street number of the trap, the trap identification number, another address location, latitude coordinate of the trap, the longitude coordinate of the trap, the accuracy of the address, and the number of mosquitos present in the trap, giving us another ten features to work with. The spray location and times table is simply a log of the time of day, and latitude/longitude of each occurrence of insecticide spraying the city of Chicago performed. This gave us an additional four features. We broke up the corresponding date related with both the weather and mosquito data into two features: month and day.



(a) 2D Projection of samples

(b) Class vs. Mosquito Count

Figure 2: Data Details

5.3 Data Statistics

It is important to note that even though we have a total of fifty-eight features, we did not use all of them. In our final submission, we used a total of thirty-eight features. Our data consisted of features with missing data samples, of which we needed to provide imputed values.

Figure 2a is a graph of a PCA feature dimensional reduction of the dataset. The green points are the WNV negative class, while the red points of the WNV present class. You can see two distinct clusters, however, the WNV present cluster contains many WNV negative points within it. This makes the prediction difficult.

In Figure 2b, we have a graph comparing the number of mosquitoes caught in a trap versus when WNV is present in that trap. The size of each circle represents the MLE of the probability of that class being selected for that number of mosquitos. As you can see, there is a fairly strong correlation between the number of mosquitos in the trap and WNV present.

The data we have was sufficient for us to achieve a 65% testing accuracy when classifying testing data. We are thankful for the data collected in giving us the ability to give a prediction with such accuracy.

5.4 Remarks

The feature data we used was assimilated all together for our use. Before doing any preprocessing we had to merge all the data together into a single CSV file. We did this by merging samples by date. This date day-month-year in a single feature After merging the files, we separated the date into day and month, throwing out the year. To handle the missing data, we took advantage of imputation. We imputed most features by inserting the mean value of that feature for the missing value. This proved to work well in most cases.

6 Evaluation Metrics and Model Selection Design

6.1 Model Selection Design

When selecting a model we took into account:

- Data overfitting
- Data underfitting
- The Imbalance in the data
- Variance
- Bias

As with all classification, the above factors play a role. Each of the factors boil down to model complexity versus prediction accuracy. The more features we used in from our dataset certainly increased our complexity of the model. Also, when tuning each of the different classifiers we can vary to complexity. This is seen in the SVC classifer by which kernel we picked. When we used a polynomial kernel with more degrees of freedom, the prediction accuracy went down; however, when we used more features we got greater prediction accuracy. Understanding these trade-offs is a fine balance. When comparing variance and bias, we saw in our lower complexity models had a higher bias but lower variance, whereas our more complex models had a lower bias but high variance. The SVC model we ended up choosing had a high variance and low bias, which worked well for us.

Support Vector Classifier at first did not fit the data correctly. Figuring out how to weight the classes was critical for the success of this classifier, because of the imbalanced dataset the underrepresented class needs to have a higher weight. Also, the parameter weighting the penalty for an error, C, was critical for achieving our objective. Not only is it important to pick the optimal C in an imbalanced dataset, we wanted to penalize the system more for a false negative (prediction WNV negative when it was actually WNV present) as opposed to a false positive (predicting present when actually WNV not present), as we are dealing with a disease potentially effecting human life.

In the same vein, we will use k-fold cross validation to help select the right number of neighbors to consider in the K-Nearest-Neighbors classifier. Because the fewer neighbors considered increases the complexity of the model and our dataset is quite imbalanced, selecting the K value is important. If we choose too low of a K value, we will tend to overfit our data, but if K is too high, then we will not be able to predict with accuracy. In our experimentation this approach did not have much success.

6.2 Evaluation Metrics

When evaluating each of our models, we took into account the following metrics:

- Cross-validation accuracy
- Testing Accuracy
- Confusion Matrix
- Precision
- Recall
- F1-Score

Using the above techniques, we were able to parameterize each different classifier we experimented with, as well as select a classifier. The cross-validation accuracy proved to be extremely important for us, as it is a better metric to consider than testing accuracy when considering a SVC, which we believed would be the best classification technique we could use. These were also important in the other classifiers. A confusion matrix, precision, recall, and F1-score were all also vital in seeing how each classifier preformed. A confusion matrix shows the number of true positives, true negatives, false positives, and false negatives when testing a classifier. This deemed to be extremely important when seeing the tradeoff between false positives and false negatives. This matrix also allowed us to determine the next three metrics. Precision is the number of true positives divided by the sum of the number of true positives plus the number of false positives. Precision tells us how certain we are a sample actually has WNV when it is classified as WNV present. Recall is the number of true positive over the sum of the number of true positives and number of false negatives. This gives us the confidence we should have in our classifiers ability to detect WNV present in a given sample. The F1-score is a measure of the tests accuracy, using a proportion of the recall and precision. The best F1-score has a score of 1 being the best and 0 the worst. The F1-score metric is as follows:

$$\frac{2 \cdot recall \cdot precision}{recall + precision}$$

7 Experimental Results

Designing a machine classification system is an iterative process; to improve our machine, we must conduct experiments. In this section, we will discuss the results we received.

7.1 3-Fold Cross Validation

To compute an accuracy statistic for our model during model selection and tuning, we must only use the training data; otherwise, we risk fitting our model to the testing data (and thus distorting our model performance). We also cannot test and train on the same data; otherwise, we'd end up with a much better apparent performance than we actually have. Thus, we use k-fold cross validation to split our data into k subsections. The evaluator will iterate through, training on $k-1$ of the subsections and testing on the remaining subsection. Performing this k times (one for each possible choice of the testing data), we get multiple accuracy statistics. These may then be used to compute an average accuracy and standard deviation to give us some indicator of how our classifier will perform on the testing set.

Because our training set was heavily imbalanced, we used *stratified* k-fold cross validation. In the stratified variant, we create our subsections of the training set such that the proportion of positively and negatively classified samples is the same as (or as close as possible to) the ratio in the original training set. While this does have the disadvantage of possibly giving accuracy results dependent on the positive/negative classification ratio, this is not a problem in our situation because the testing set is guaranteed to be a 50/50 split of positive and negative samples.

Below are the results from running 3-fold cross validation with various models:

Model	**Accuracy**	**Std. Deviation**
Support Vector Classifier	0.5642	0.366
Naive Bayes	0.2109	0.140
Random Forest	0.6704	0.405
Decision Tree	0.6234	0.395
5-Nearest Neighbor	0.7346	0.327

As we can see here, the Support Vector Classifier did not perform the best; rather, the 5-NN and tree based approaches did. However, they all had a very large standard deviation, and the approaches that did perform well are prone to over-fit data.

We tuned each of these models to perform their best (these results shown above), and then submitted to the Kaggle server to obtain the testing accuracy in the next section. We found that a C-value of 85 worked best for the SVC, and that $5-NN$ was the best at prediction while still avoiding overfitting.

7.2 Testing Accuracy

As part of the Kaggle competition, the testing data is kept hidden from the contestants; this is primarily an anti-cheating measure, but keeping testing data separate from training data helps ensure that our model is not "looking at the answers" to help with its classification.

When we submit to Kaggle, we receive two results: a "public" score and a "private" score (this is another feature of Kaggle competitions, which allows closed-source programs to be judged on a separate dataset than open-source programs). Unfortunately, the program is only run once on a large dataset, instead of multiple times on various data sets so we could obtain a standard deviation as above. The table below summarizes our results for the private scoreboard:

Model	**Accuracy**
Support Vector Classifier	0.646
Naive Bayes	0.535
Random Forest	0.513
Decision Tree	0.530
5-Nearest Neighbor	0.502

Interestingly enough, the SVC and NB models performed much better on this dataset than on the training set. We believe this to be because both the SVC and the NB models take into account the high imbalance of the training set, while the other models do not. The Naive Bayes model accounts for this by considering the probability of a sample being classified in a particular class, given no other knowledge ($P(C_i)$ for class C_i). The `SKLearn` library for SVC also includes a "`class_weight`" parameter for imbalanced datasets; this adjusts the SVC's C value to be instead `class_wt[i]*C`. This weighting is inversely proportional to the class frequency in the input data, and greatly improved the performance of the classifier when we started using it. (The SVC went from classifying *everything* as a negative sample–and thus 50% testing accuracy–to being the best of all 5 of the models we tried.)

The Random Forest and Decision Tree algorithms did not perform as well as their training results would lead us to believe. This is most likely do to overfitting on a largely imbalanced dataset. The poor performance of the 5-NN classifier should come as no surprise with this dataset. If one glances at the PCA plot (Figure 2a) from the Data Statistics section, they will see that many "positive" datapoints in the training set exist in very close proximity to several other "negative" datapoints. Although this is simply a projection into two dimensional space, it is representative of the behavior in the high-dimensional space–samples that should be classified as positive do occur near samples that should be classified as negative. This causes the K-NN classifier to perform poorly.

7.3 Precision, Recall, and F1-Score

While the above results do provide us with an indication of how accurate (or inaccurate) certain classification methods are, it doesn't tell the whole story. When attempting to eliminate West Nile Virus, a false negative is much more dangerous than a false positive. Thus, we computed the confusion matrices for each of the five classifiers considered above. Each confusion matrix has the following layout:

True Positive	False Negative
False Positive	True Negative

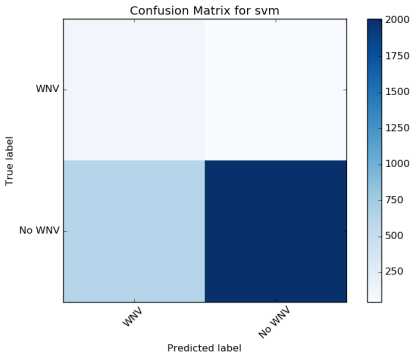
Thus, we want a high-valued main diagonal, but we'd rather have a large entries in the lower-left than the upper-right on the off diagonal, since–for our application–it is better to kill more mosquitos needlessly than to let some live that might have WNV.

In Figure 3, we see a graphical depiction of the confusion matrices for each classifier. This lets us see, at a glance, that the SVM classifier has the desired trait of a strong main diagonal and most of the incorrectly classified results are in the lower-left quadrant.

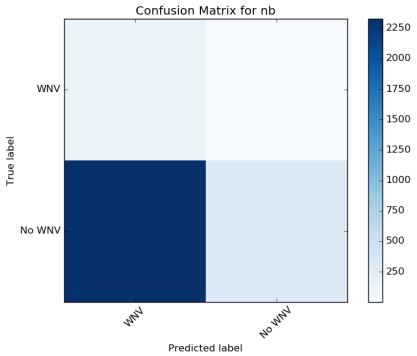
8 Conclusion and Discussions

In this project we were able to use collected, supervised data to make predictions about where and when mosquitos will carry West Nile virus in the city of Chicago. Using machine-learning techniques, our team was able to make this supervised data useful in predicting West Nile virus. Our team realized this task as a binary classification problem, and designed fitted models to accomplish the classification. As with many disease-related prediction models in the medical field, we allowed for a greater number of false positives rather than false negatives, as a false negative gives a greater risk of people in the city of Chicago contacting West Nile virus. The challenge of this task was the class imbalance of our training set. We found that when fitting our classifiers with this imbalanced dataset, assigning class weights was very important to the success of the model. Of the experimented classifiers, we found a Support Vector Classifier (SVC) to give us the greatest success. It performed best in evaluation metrics we performed on all the experimented classifiers.

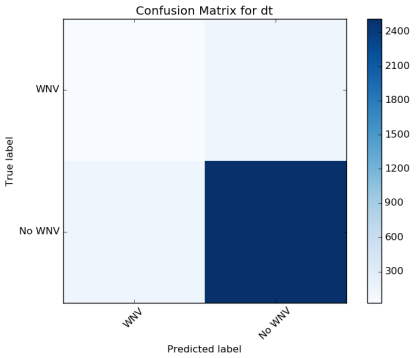
The authors would love to see continued work on the prediction of West Nile virus in mosquitos. Area of future work would include over sampling the underrepresented class by adding synthetic data points using the Synthetic Minority Over-sampling Technique commonly referred to as SMOTE [Chawla et al.]. Another area of future work could come in under sampling the overrepresented class by randomly taking away data samples of the training data. Imbalanced datasets come up in a variety of fields, such as economics, consumerism, medical, sports, engineering; which elevates the



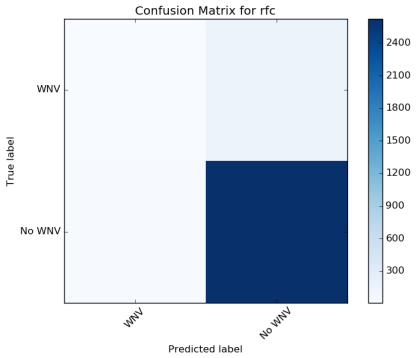
(a) SVM Confusion Matrix



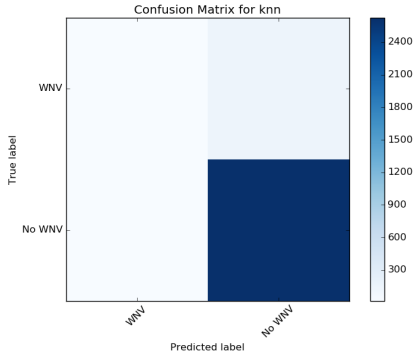
(b) Naive Bayes Confusion Matrix



(c) D-Tree Confusion Matrix



(d) Random Forest Confusion Matrix



(e) 5-NN Confusion Matrix

Figure 3: Confusion Matrix (Graphical Description)

importance and relevance of classifying imbalanced datasets. We expect to see the area of research to continue to be a pursued in the present and near future.

9 Team justification

We have a biological issue at hand with mosquitoes in the city of Chicago carrying the West Nile virus. A large step in solving this problem will come from analyzing data in order to make an adaptive, predictive solution. This task will be accomplished through an intense effort involving statistics, mathematics, and computer programming. Our team consists of two members with different, complimentary backgrounds that meet and exceed the required skills needed to solve this problem. Andrew is a Computer Science and Mathematics double major, whereas Frank is an Electrical Engineering graduate student. Not only are the technical skills met, our team understands the scope and the significance of the issue in order to get to the root problem. The presence of the West Nile virus in Chicago carries implications much further than a city concern, but rather a world issue. Our team will treat this project as such.

In order to come up with a solution to this problem, many solutions should be examined. Our team has a wide breadth of knowledge of various classification techniques which will allow us to investigate different solutions. In a classroom setting we have been exposed to expository teaching on classification subject matter, while being given real world problems to solve. This team is motivated by helping others, and is excited to use their college education to help the city of Chicago. Furthermore, one of our team has had exposure to working with weather data and analyzing trends in both an academic research environment, and in industry. The intelligence of our group is met with great curiosity and vigor to accomplish this task.

This project holds significance for the second author, Andrew Norton, because one could apply the results of this project to protecting the residents of Chicago. As a third year undergraduate, he is still deciding on the focus of study for his future research; however, he certainly wants to see his efforts to advance a field protect mankind from the dangers of the world.

This project holds significance for Frank Hiemstra because of his research in control theory. The results of this project in making predictions based on supervised data directly correlate to tuning PID controllers. As the field of machine learning and predictive analysis expands, more and more correlations to control theory will arise.

10 Contributions

This project was completed in the collaboration between Frank Hiemstra and Andrew Norton. We worked extremely close together, making the workload as even as possible between the two. We both worked almost the same number of hours on the project, and as well as each was involved in every step of the project. If we were to give ourselves titles, Frank led the experimental design/analysis, whereas Andrew led the programming. This project evolved by: coming up with the project idea, understanding the problem, applying machine learning techniques, writing the code to implement the techniques, analyzing results, and making design decisions. Each of us was involved in every aspect of the way. Practically, we had to submit a project proposal in the beginning, followed by mid-phase report, final presentation, and lastly this final report. Again, we each were extremely involved in all four of these tasks with Andrew taking the lead on implementing classification by programming, while Frank taking the lead on designing solutions. We really enjoyed working with one another and found applying machine learning to this project fascinating.

References

Hahn, M., Monaghan, A., Hayden, M., Eisen, R., Delorey, M., Lindsey, N., . . . Fischer, M. (2015). Meteorological Conditions Associated with Increased Incidence of West Nile Virus Disease in the United States, 2004-2012. *American Journal of Tropical Medicine and Hygiene*, 1013-1022.

N. V. Chawla, K.W. Bowyer, L. O. Hall, W. P. Kegelmeyer, SMOTE: Synthetic Minority Over-Sampling Technique, Journal of Artificial Intelligence Research, vol. 16, 321-357, 2002.

Ruiz, M., Chaves, L., Hamer, G., Sun, T., Brown, W., Walker, E., . . . Kitron, U. (n.d.). Local impact of temperature and precipitation on West Nile virus infection in Culex species mosquitoes in northeast Illinois, USA. *Parasites Vectors Parasites & Vectors*, 19-19.

West Nile Virus Prediction. (n.d.). Retrieved October 12, 2015, from https://www.kaggle.com/c/predict-west-nile-virus

Vapnik, Vladimir, and Corinna Cortes. "Support-Vector Networks." Machine Learning 20 (1995): 273-97. http://csee.wvu.edu/x̌inl/library/papers/comp/ML/svm.pdf.