**Analysis of NOAA Hurricane Data in the Atlantic**

Genesis Ewell, Scott Kotrla, Ian Lynch, Drew Murray

ADTA5230.100

10/10/2023

**Analysis of NOAA Hurricane Data in the Atlantic**

**Table of Contents**

Abstract 1

Problem Statement / Introduction 2

Tools Used for Analysis 2

Initial Dataset 3

Revised Dataset 6

Exploratory Data Analysis 9

Modeling Overview/Methods 15

Modeling Results 23

Limitations 24

Future Research 24

References 25

**Abstract**

This research focuses on the predictions of NextWindHigher with 12-14 different predictors. Several machine learning models, classifications, and algorithms are used to inform the appropriate audience on future hurricanes to prepare for possible financial loss, the severity of hurricanes and prepare those who are in the path of landfall. It was decided that Neural Networks was the best model to conclude the prediction of future hurricanes and uncover the financial loss due to the hurricanes.

**Problem Statement / Introduction**

The Atlantic region where hurricanes frequent is of significance due to the impact that they have on the region and the lives of those affected. Hurricanes are one of the most powerful natural disasters of nature and can produce a butterfly effect and cause other storms to occur such as tornadoes, flooding, and rip currents (Hurricanes | NOAA, 2020). Hurricanes are increasing in size and intensity each year which calls for data analysis to better understand these storms. The problem that we want to investigate uses advanced data analytics to predict the current max wind based on the geographical features of said hurricane - will it increase or decrease? Advanced methodologies and algorithms help us to understand past storm history to predict future storms and help citizens and first responders make better enhancements to post-hurricane procedures. There are a few challenges to completing analysis to prepare for the future such as data quality, understanding risk assessment and predictive modeling that come into play but once we understand and address those challenges then we are able to make decisions that will benefit the community in more ways than one. It is crucial to develop advanced techniques to alleviate the personal and financial impact of hurricanes in the Atlantic region.

**Tools Used For Analysis**

**Excel:** Excel is the spreadsheet application used in this project, to retrieve the dataset, and convert it into a comma separated file (CSV). Some preprocessing was implemented before importing the data into Python.

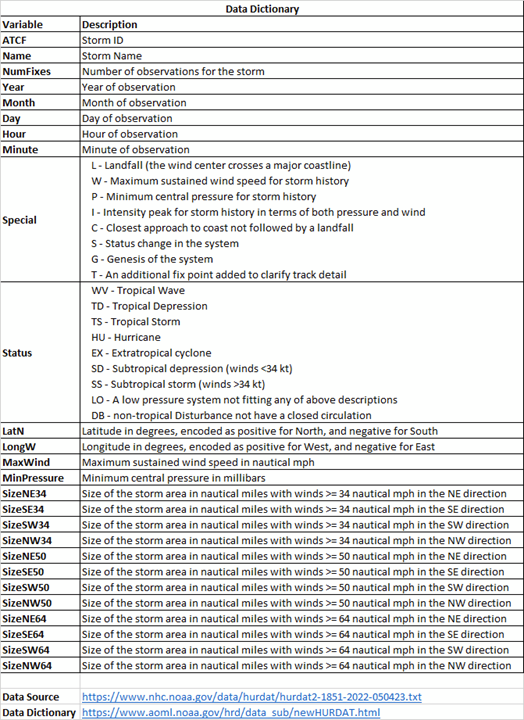
**Python:** Python was the programming language used for this analysis. It has libraries that are catered to data preprocessing, data visualization and machine learning.

* **Pandas:** Retrieved the NOAA Hurricane file and transformed into a pandas DataFrame, so the data can go through the preprocessing, data visualization and machine learning stages.
* **Matplotlib and Seaborn:** these libraries were used to execute data visualizations in order to detect patterns and insights before conducting the machine learning algorithms.
* **Sklearn:** This library was used to conduct the supervised and unsupervised machine learning methods, along with assessing its performance.
* **geopy:** Used to calculate and create columns for geographical data in the preprocessing phase.
* **Dmba:** Used to assess the accuracy of the classification machine learning models.

**Initial Dataset**

The hurricane dataset used in this project contains 7726 observations each with 26 variables after doing some preprocessing on the original data set. Each observation is part of a time series of storm movement and wind speed for all of the Atlantic storms between 2008 and 2022 starting with Gustav in 2008 with an average of 30.2 observations for each of the 256 storms. The two ID variables are ATCF and Name - ACTF is a unique identifier for a storm historically, while Name is only unique within a season. Because there are multiple observations over time for each storm, a single observation can only be specified with ACTF and a collection of date and time discrete numerical variables: Year, Month, Day, Hour, and Minute. NumFixes is the final variable that is the same for all observations of each storm - it is a discrete numerical variable that provides the number of observations in the dataset for a storm.

The remaining variables include two nominal categorical variables, two location variables, two measurement variables, and 12 size variables. The first nominal variable is Special, which denotes something special about the measurement, such as “L” for Landfall or “W” for Maximum sustained wind speed. The second nominal variable is Status, which denotes storm category, such as “TS” for Tropical Storm or “HU” for Hurricane. The full data dictionary is shown in ***Figure 1***, which includes descriptions for all classes of both of the nominal variables.



***Figure 1. Hurricane dataset data dictionary***

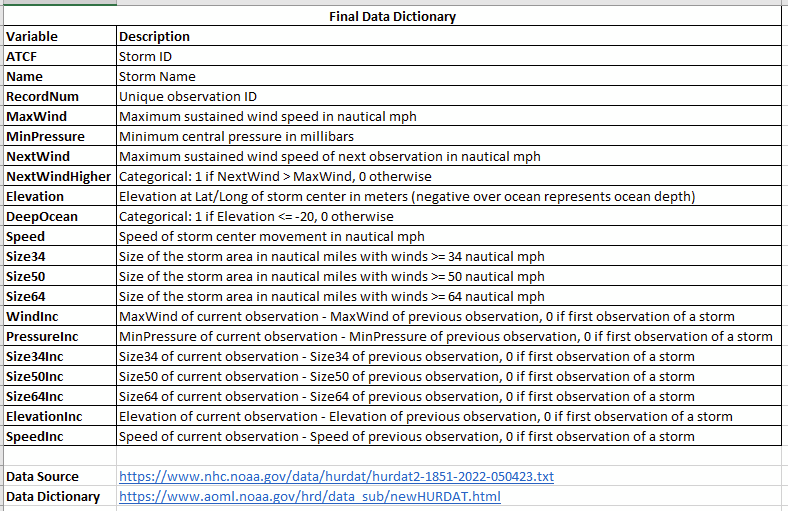
Location variables are LatN and LongW, which are continuous numerical variables that measure latitude and longitude in degrees. Latitude is encoded with a positive sign for North and a negative sign for South, while longitude is encoded with a positive sign for West and a negative sign for East. Measurement variables are MaxWind and MinPressure, which are continuous numerical variables that measure maximum sustained wind speed in nautical miles per hour (mph) and minimum pressure in millibars. The remaining 12 variables are the size variables: SizeNE34, SizeSE34, SizeSW34, SizeNW34, SizeNE50, SizeSE50, SizeSW50, SizeNW50, SizeNE64, SizeSE64, SizeSW64, and SizeNW64, which provide the size of the storm in nautical miles in a specific direction, where size is defined by a wind speed over a certain threshold. The directions are NE for Northeast, SE for Southeast, SW for Southwest, and NW for Northwest, and the wind speed thresholds are 34, 50, and 64 nautical mph.

The source dataset contains many more records going back 100 years, but prior to Gustav in 2008, there are significant quantities of missing data, so it was decided to focus on the last 15 years, where data is much more complete and there are still a significant number of records. An additional variable WindDirection was present in the last couple of years of data, but since it is not present in the majority of the records, it was dropped. In the original dataset, the ID variables and NumFixes metadata were not inline with the observations - they were header data for each storm, which makes the data difficult to work with. This was addressed with preprocessing in Excel to get the data into a CSV file that could be imported into Python for analysis. Also, dates and times were encoded as strings in “YYYYMMDD” and “HHMM” formats, which this was converted into the five separate date and time variables described previously. The original data had no column names, so column names were added to the dataset based on the data dictionary. Finally, the latitude and longitude values were converted from strings that included N, S, E, and W to numbers with direction represented by positive or negative sign.

**Revised Dataset**

Prior to Exploratory Data Analysis (EDA) and modeling, additional data was incorporated into the data set in the form of calculations and external lookups on variables within each observation as well as calculations between sequential observations in time. In addition, extra variables that are not expected to be used were removed. While the initial plan for target variable was related to wind speed at landfall, between the complexity of associating measurements days before landfall with landfall and the limited number of observations at landfall, the revised plan for target variable is to predict the whether the max wind speed of the next observation will be greater than the current wind speed. To support this, two new variables were created: NextWind, which is the MaxWind in nautical mph from the next observation in time if the next observation is the same storm, or 0 for the last observation in the series; amd NextWindHigher, which is a 1 if NextWind > MaxWind or 0 otherwise. While the general plan is to build a classification model to predict NextWindHigher, NextWind is also available as an outcome variable for a prediction model. The next challenge was identifying whether a given observation was over land or not. There is a Special code of L for Landfall, but there’s no way to tell after Landfall whether the storm stays over land or goes back over the ocean. Since storms generally decrease in speed over land, this was a potential large source of classification error. To resolve this issue, the Google Maps API was used to retrieve elevation for every observation using LatN and LongW, which has the added benefit of retrieving ocean depth when the coordinates are over the ocean. One important thing to note about the Google Maps API is that it takes Lat and Long encoded as N positive and E positive, requiring passing -LongW to the API. The API responses were encoded into two new variables Elevation, which is measured in meters; and DeepOcean, which is a 1 if Elevation <= -20 meters or 0 otherwise. -20 meters was chosen as the condition for DeepOcean because there were several measurements in the low negatives that made more sense to classify as ocean instead of land.

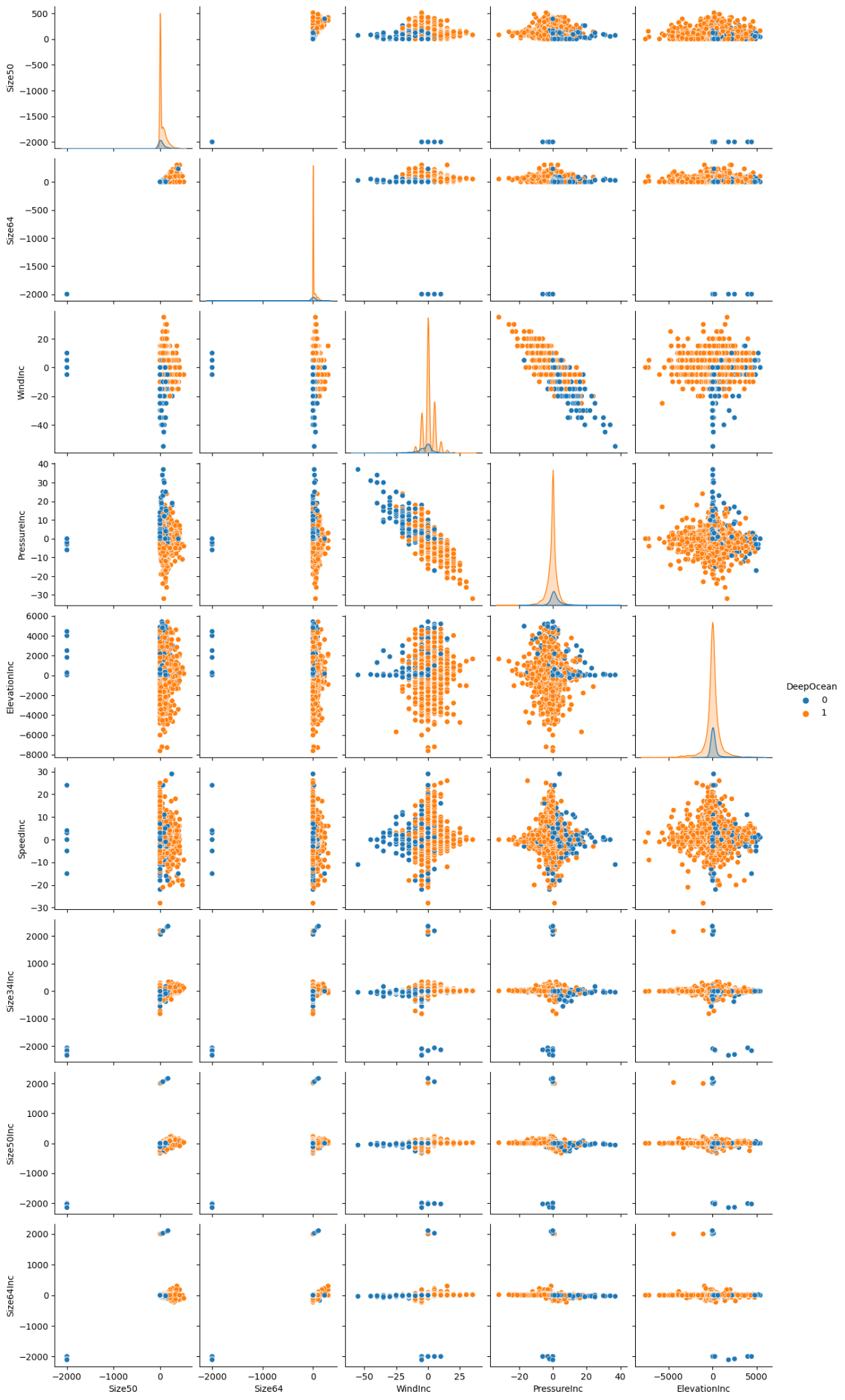
Another missing variable from each observation was speed of storm movement, which can be calculated from change in Latitude and Longitude over time. To assist with this calculation, LatN, LongW, and the change in time between previous and current observations were added to the current observation as PLatN, PLongW, and HoursPassed, with HoursPassed set to 0 if there is no previous observation of the same storm. The Latitude and Longitude measurements were used in the distance calculation of the geopy Python library to calculate nautical miles, which were then divided by HoursPassed to get speed in nautical mph, which was stored in the Speed variable. Next, the size variables were combined into a single measurement per wind speed. The first approach was to add the four measurements (NE, SE, SW, NE), which are basically the diagonals of a quadrilateral (a square if they are all the same), but there was concern that this was not a valid measurement of size. The second approach that was used in the final data set was to add opposite diagonals and take the longer pair, which would represent the maximum diameter of the storm at a given wind speed. This reduced the 12 size variables (at three speeds times four measurements) to three new size variables (at three speeds): Size34, Size50, and Size64. Next, several additional variables from the previous observation were compared to the current observation so that changes in variables over time could be calculated, which have the potential of being good predictors. These were encoded as the original variable names with “Inc” at the end, with an increase encoded positive and a decrease encoded as negative. Like before, for the first observation of a storm, these change variables are all encoded as 0. The new variables are WindInc from MaxWind, PressureInc from MinPressure, Size34Inc from Size34, Size50Inc from Size50, Size64In from Size64, ElevationInc from Elevation, and SpeedInc from Speed. Finally, RecordNum was added as a unique identifier of each observation, since the initial data set required pairing storms with all of the date and time variables.

Once the variable additions were completed, the dataset had grown to 44 variables, so the next step was variable elimination. NumFixes is metadata about each storm, so it was dropped, and all of the date and time details were dropped; while there could be correlation between wind speed and time of year or time of day, it was thought to be a weaker predictor than the other variable. ATCF was considered for elimination, but since multiple storms can have the same name,it was kept to uniquely identify each storm. Special and Status were removed because they were believed to provide little information beyond what exists in other fields. LatN and LongW were removed; while there could be a correlation between wind speed and location, it’s not likely something that would show up in the model and tools used in this analysis beyond Speed and Elevation, which were already captured in new variables. As discussed previously, the 12 directional size variables were consolidated into new maximum size variables and can be removed. PLatN, PLongW, and HoursPassed were all intermediate calculation variables and they can be removed. The final variables are shown in the data dictionary in ***Figure 2*,** and include three ID variables, two outcome variables (one categorical and one numerical), one categorical predictor, and 14 numerical predictors.****

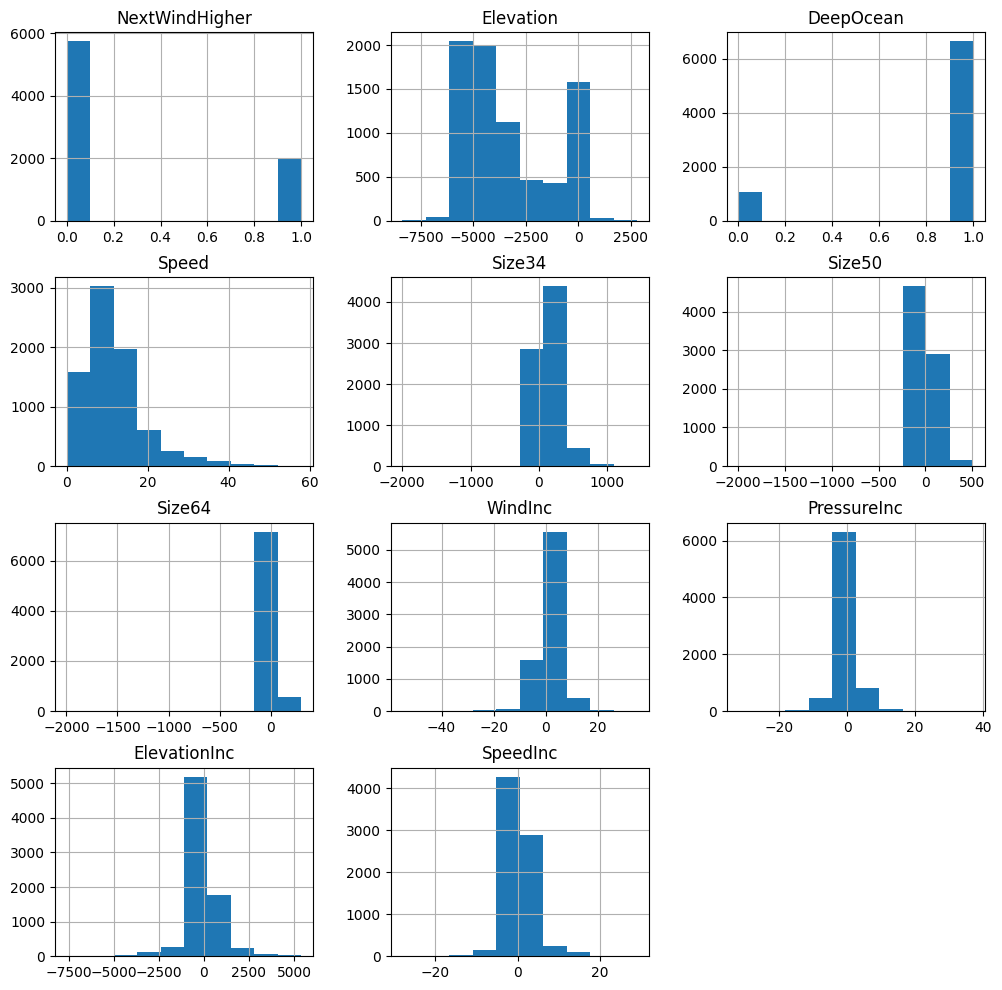
***Figure 2. Hurricane dataset final data dictionary***

**Exploratory Data Analysis**

After finalizing the data set, Exploratory Data Analysis (EDA) was completed on the data, starting with a series of scatterplot matrices to look for outliers and correlations across the 14 numerical variables with different colors for the DeepOcean categorical variable. To keep the plots readable, each plot has 5 variables on the x-axis, requiring 3 total plots to show all of the variables. The results are a 5x14 grid for the first 5 variables, a 5x9 grid for the next 5 variables, and a 4x4 grid for the last 4 variables. The 5x9 grid is shown in ***Figure 3.*** From the plot, the negative correlation between WindInc and PressureInc is clear, and outliers across all of the Size and SizeInc variables show up. The outliers are due to all 3 Size variables being reported as -1998 for 7 observations. To address this missing data, the 3 Size variables for these 7 observations was updated to the average Size of the observations before and after, which also required recalculation of the 3 SizeInc variables for the observations after the 7 with missing Size data.



***Figure 3. 5 x 9 Scatterplot matrix***

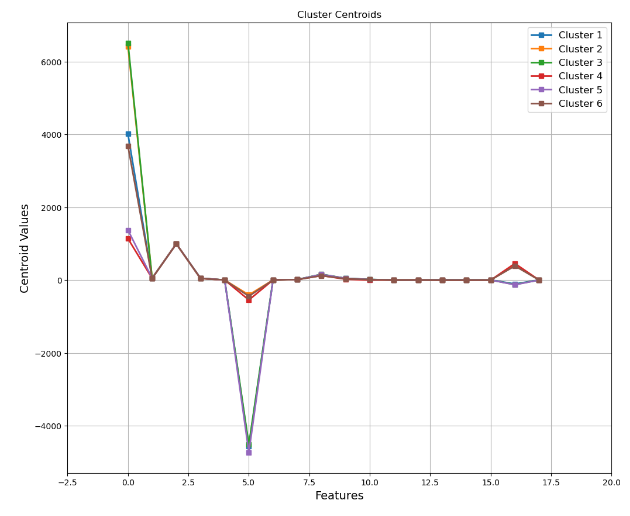
****

***Figure 4. 4x3 Histograms of Numerical Variables***

***Figure 4*** are histograms of all the numerical variables in the dataset besides the variable RecordNum. When analyzing ***Figure 4***, columns like NextWindHigher and DeepOcean only have bars at zero and 1, indicating that they are binary variables. Unfortunately, none of the columns pass the assumption of normality, which was accounted for when developing the machine learning models. Furthermore, the range of values in these histograms are diverse meaning that executing these models without any data transformation will increase the variance of the machine learning models which can lead to poor generalizations when trying to predict new data. In order to combat that, we used normalization techniques like min-max scaling and standardization.

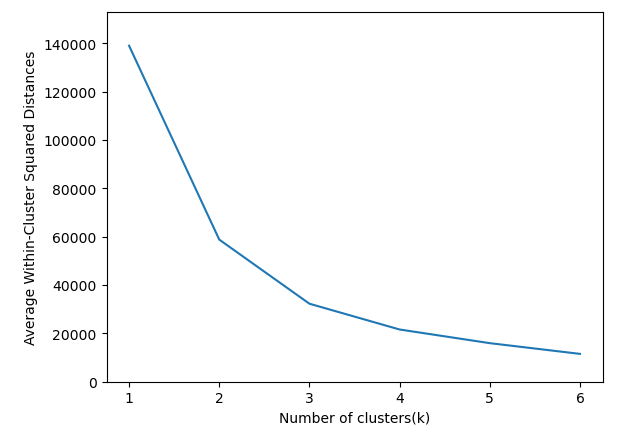
It was decided that hierarchical clustering would be better suited for exploratory data analysis. For hierarchical clustering, scikit-learning, pandas, and scipy were the libraries used in Python for analysis. Hierarchical clustering is a clustering method that arranges the clusters into a natural hierarchy. Clusters are created by measuring the distance and dissimilarity of the data points (Shmueli & Gedeck, 2019). Hierarchical clustering was used to determine the likelihood of the NextWindHigher variables based off of NextWind and Speed.

Euclidean distance was used to measure the dissimilarity measure between NextWindHigher and Speed. The results yield higher distances, before normalization, which means the rows are a little more dissimilar in the terms of these numeric variables. After normalization with using the sample standard deviation, the distances are smaller which is yielding more similar values. Distances ranged between 1.16 to 3.65 at the highest upon observation. The closest distance was between Hurricanes Gustav and Josephine at 0.26 and the furthest distance was between Hurricanes Gustav and Nicole at 3.65. The distance of the centroids between Cluster 0 and Cluster 1 for NextWind (which is equivalent to NextWindHigher) is 0.63 which is indicative of the data being separated and the clusters are full. This further proves the similarity of the data values. ***Figure 5*** shows a line graph visualization of the cluster centroids to show the similarities of the clusters.



***Figure 5. Line Graph of Centroid Clusters***

After further analysis, it was determined that 6 clusters are needed. This is shown by an “elbow chart” line graph in ***Figure 6*** that depicts the overall decline if we choose to add more clusters. With the “elbow chart” we can learn the similarities and differences of the clusters. For example, there is a big decline, (distance) between Cluster #1 and Cluster #2 but between Cluster #3 and Cluster #4 there is a small distance. With this information in mind, we can determine the cluster validity. The ratio is very small so we know that we have valid separated clusters.



***Figure 6. Elbow Chart of k=6***

**Modeling Overview / Methods**

A total of 7 different classification models were developed to predict NextWindHigher using 60% of the data for training, leaving 40% of the data for selecting the best model. The same training data was used for all models by always setting the random seed to 7. The following modeling algorithms were used:

* **K Nearest Neighbors:** A supervised machine learning algorithm that classifies the observations based on similar records in the data. K are the number of records that are similar to the new record. The class with the most votes out of k will be the class that the new record will be assigned to (Shmueli & Gedeck , 2019).
* **Neural Network:** Derived from the human biological neural network. It’s a machine supervised machine learning algorithm that utilizes different combinations of weights and bias in three layers, input, hidden and output layer to predict the outcome variable. This algorithm also falls under a subset of machine learning methods called Deep Learning (Shmueli & Gedeck, 2019).
* **Logistic Regression:** Derived from linear regression, but uses the logit function to convert the continuous outcome into a binary outcome where the values range from 0 to 1, which indicates the probability of the predicted class (Shmueli & Gedeck, 2019).
* **Linear Discriminant Analysis:** Classical statistical technique that classifies new records based on the statistical distance of each class mean. It is also utilized to highlight the aspects that distinguish the classes (Shmueli & Gedeck, 2019).
* **Random Forest:** An ensemble method that combines multiple classification trees that improves the predictive power in comparison to a single classification tree. It utilizes a random sampling technique called bootstrapping (Shmueli & Gedeck, 2019).
* **Gradient Boosted Tree Classifier:** An ensemble method that combines multiple classification trees and concentrates on the misclassified records from the previous tree (Shmueli & Gedeck, 2019).
* **K-Means clustering:** A clustering method where the user specifies the number of clusters, and the algorithm assigns the records to each cluster. K is the number of clusters that the user specified (Shmueli & Gedeck, 2019).

**K Nearest Neighbors**

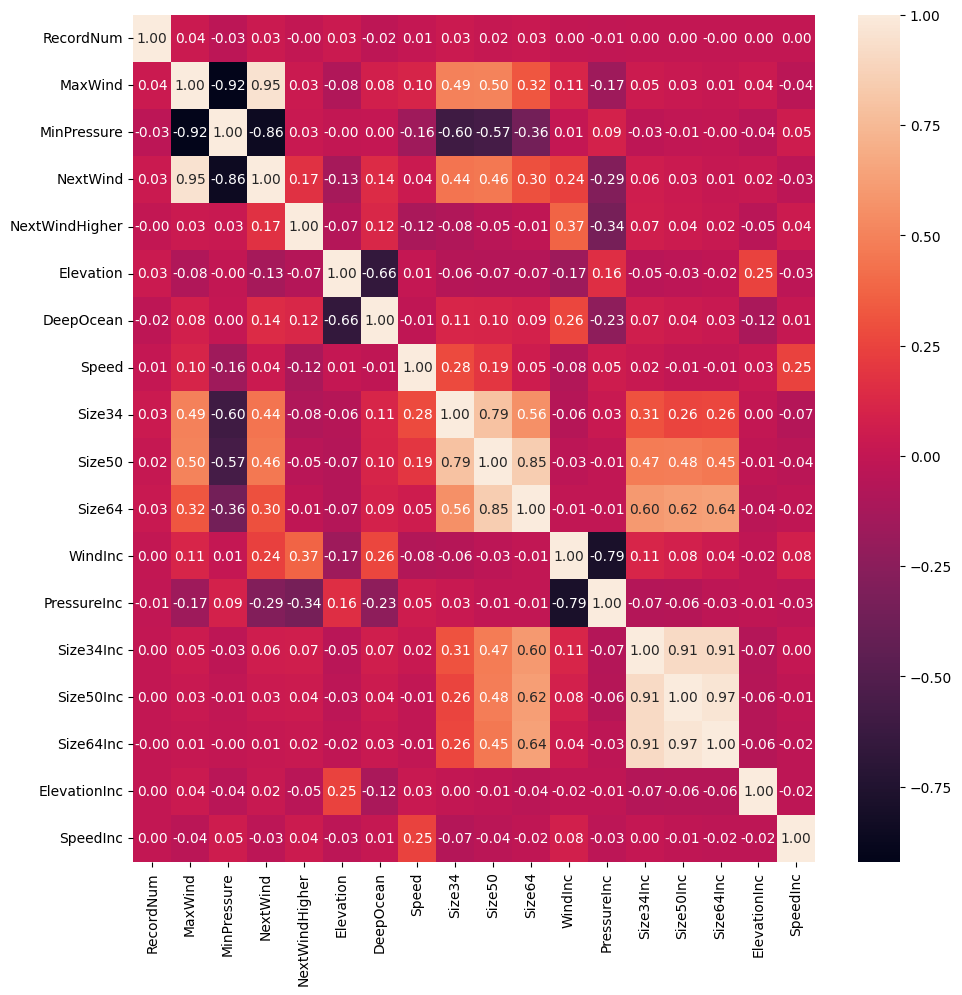
To develop the best K Nearest Neighbors (KNN) model, the GridSearchCV and KNeighborsClassifier functions from the sklearn library were used to find the best k using the 14 numerical variables as predictors, cv of 5, and a k search range of 1 to 70, where 70 is the square root of the samples in the training partition. Before running the model, the predictor variables were standardized to a range of 0-1. The best results were at k=21 with a score of 0.779. The classificationSummary function from the dmba library shows 79.4% accuracy on the training partition and 77.8% accuracy on the validation partition, which shows that the model fits pretty well. However, since 74.3% of the validation observations are 0, the model results are not very good, with a True Positive Rate (TPR) of only 30.7% on the validation data. In order to increase the TPR, the cutoff was reduced until the True Negative Rate (TNR) was lower than the TPR and then backed up one step. At the resulting cutoff of 28.6%, the TPR increases to 66.3%, while the TNR falls to 80.9% on the training data. The results at the selected cutoff on the validation data are a 61.3% TPR, 79.6% TNR, and an unweighted average of 70.4%. Because the model cutoff is being adjusted, GridSearchCV was rerun with roc\_auc scoring, resulting in a score of 0.763 at k=65. After adjusting the cutoff to 26.2% , the results on the validation data are 65.8% TPR, 74.6% TNR, and an unweighted average of 70.2%, which is a small decline. The cutoff was also calculated using the roc\_curve function by finding the maximum of TPR + TNR, but the results on the validation data were worse at TPR of 59.4%, TNR of 80.6%, and an unweighted average of 70.0% at a cutoff of 30.8%. The final results for KNN are k=21 using standard scoring, a cutoff of 28.6%, 61.3% TPR, 79.6% TNR, and an unweighted average of 70.4%.

**Neural Network**

To develop the best Neural Network (NN) model, the GridSearchCV and MLPClassifier functions from the sklearn library were used to find the size of a a single hidden layer using the 14 numerical variables as predictors, cv of 5, and a size for each hidden layer between 1 and 14, where 1 is the number of output classes -1, and 14 is the number of predictors. Before running the model, the predictor variables were standardized to a range of 0-1. The best results were at 7 nodes with a score of 0.782. The grid search was then rerun with two hidden layers, each with 1 to 14 nodes, and the best results were at two layers each with 4 nodes with a score of 0.783. The classificationSummary function from the dmba library shows 79.5% accuracy on the training partition and 77.0% accuracy on the validation partition, which shows that the model fits pretty well. However, since 74.3% of the observations are 0, the model results are not very good, with a True Positive Rate (TPR) of only 50.1% on the validation data . In order to increase the TPR, the cutoff was reduced until the True Negative Rate (TNR) was lower than the TPR and then backed up one step. At the resulting cutoff of 26.5%, the TPR increases to 75.7%, while the TNR falls to 75.9% on the training data. The results at the selected cutoff on the validation data are a 71.8% TPR, 75.1% TNR, and an unweighted average of 73.5%. Because the model cutoff is being adjusted, GridSearchCV was rerun with roc\_auc scoring, resulting in a score of 0.804 for one layer with 1 node and a score of 0.798 for two layers each with 3 nodes. Using the one-layer model and adjusting the cutoff to 26.7%, the validation TPR is 73.2% and the TNR is 75.0%, with an unweighted average 74.1%, which is a decent improvement over the original model. The cutoff was also calculated using the roc\_curve function by finding the maximum of TPR + TNR, which resulted in a small improvement with a validation TPR of 73.8%, TNR of 74.7%, and unweighted average of 74.2% at a cutoff of 26.4%. The final results for NN are one hidden layer with 1 node using roc\_auc scoring, a cutoff of 26.4%, 73.8% TPR, 74.7% TNR, and an unweighted average of 74.2%.

**Logistic Regression**

The dimension reduction technique used for both logistic regression and linear discriminant analysis was a correlation heatmap shown in ***Figure 7***. The predictors, Size34Inc, Size50Inc, Size64Inc, and MaxWind were dropped to reduce the possibility of multicollinearity when fitting the algorithms. NextWind was removed because it was the numerical equivalent to NextWindHigher. RecordNum was dropped as well because it was for identification purposes for the observations. Next, the predictor variables were standardized because they were measured in different units, and algorithms may show bias to measurements that have a higher range of values than others.



***Figure 7. 18x18 correlation heatmap***

For logistic regression, python’s library, sklearn, was used to execute the logistic regression algorithm that had L2 regularization. Sci-kit learn’s GridSearchCV was implemented as an exhaustive search tool to find the best C parameter of the logistic regression model between the values of 0.001, .01,7,8,9 and 10, and then performs a cv (cross-validation) of 5 different partitions. The logistic regression model used the training data for fitting. After the GridSearchCV was carried out, C being equal to 1 was the best value and it achieved an accuracy score of 77.80%, with a specificity of 94.42%, and a sensitivity of 29.89% at a cutoff value of 50% using the training data. Since the sensitivity was extremely low, the cutoff value needed to be adjusted. At a cutoff value of 26.2%, sensitivity was able to increase to 72.4%, while the specificity decreased to 73.1%, and overall accuracy decreased to 72.7%. For the purpose of this project, the priority is to have the sensitivity to being equal or close to equal to the specificity. To assess how well this logistic regression model generalized with unseen data, the validation data was used with the sklearn’s ClassificationReport and dmba’s ClassificationSummary function which includes the accuracy along with a confusion matrix. The model obtained an accuracy of 72.5%, with a sensitivity score of 72.2% and a specificity score of 72.8% at a cutoff value of 26.2%. Because the sensitivity score was higher than the specificity score, the algorithm performed slightly better at classifying when NextWindHigher was 1 than 0. When it comes to the overall accuracy score when assessing the validation data, the training accuracy scored slightly higher. Meaning that the model was overfitting, but not significantly enough to impact generalization of the model.

**Linear Discriminant Analysis**

For linear discriminant analysis, python’s sklearn was also used to execute the algorithm. Because linear discriminant analysis is simple and easy to compute, an exhaustive search tool was not used. The algorithm was fitted using the training data, obtaining an accuracy score of 76.79%, with a specificity of 97.09%, and a sensitivity of 18.25% at a cutoff value of 50%. Since the sensitivity was extremely low, cut off value needed to be adjusted. At a cut off value of 26.2%, the model achieved a sensitivity score of 70.3%, and a specificity score of 76.1%, and an overall accuracy score of 73.0% Next, to assess how well this linear discriminant analysis model generalized when it comes to unseen data, the validation data was used along with dmba’s ClassificationSummary function. This includes the accuracy and a confusion matrix. The model achieved an accuracy score of 72%, with a sensitivity score of 72.4% and a specificity score of 73%. The linear discriminant analysis model did a better job at classifying when NextWindHigher was 0 than 1. Lastly, the overall accuracy score for the validation data was slightly lower meaning that there is some overfitting. However, it is significant enough to impact the generalization.

**Random Forest**

To develop a Random Forest classifier model, the process was started with using a full tree classifier using the entire set of dimensions available. It was determined that the variables of importance were the Elevation, Speed, MinPressure, and Size34 columns. Ultimately, after going through the process it was determined that it was underfit against the validation set, at an accuracy of about 71.3%. The number of terminal leaves from this tree was 1,395. It was then attempted to be adapted via the grid search function.

After using the grid search function through the sklearn library, it was possible to find a more optimal depth selection process and to utilize the more important variables present, namely the MinPressure and the Elevation columns. The initial parameter search included 5 cross validation partitions, a parameter grid of depth including values 10, 20, 30, and 40. The minimum sample split was 20, 40, 60, 80, 100. The allowed interval of minimum impurity decrease was 0, 0.0005, 0.005, and 0.01. From here the initial score was about 0.7954. After several trials, the grid search function’s optimized parameters were a max depth of 5, a minimum sample split of 108, and a minimum impurity decrease of 0.001. The score yielded from this was 0.7968. Using the best tree, the predictions and actual values from the validation set were used to construct a confusion matrix. It was apparent that there was an overwhelming likelihood of choosing to predict that the next wind would not be higher. This yielded a result of 78.1% accuracy. The number of terminal leaves on this “best tree” was 12, as opposed to the full tree of 1,395.

From here, the random forest model was constructed since the original classification tree work was performed. Using an estimator count of 1,500, a random forest model was constructed to fit the training set. The results demonstrated an accuracy of 77.5%.

The positive predictive value using the random forest model was 58.3% and the negative predictive value was 82.1%. The sensitivity was 44.03% and the specificity was 89.1%. Overall, not optimal for our goals.

**Gradient Boosted Tree Classifier**

After the preliminary work was performed on the original tree set, it was then that construction of the gradient boosted classifier was produced using the training data set. The performance of this particular model yielded an accuracy of 78.1%. The positive predictive value was 60.9%, with a negative predictive value of 81.73%. The sensitivity for the gradient boosted tree model was 41.4% and the specificity was 90.8%. This appears to be a stronger model than the Random Forest model.

**Modeling Results**

Based on the preliminary findings of these models, it is possible that the Neural Net model may be the best candidate to move forward with further research into hurricane prediction. With an accuracy of 77.0%, a TPR of 73.8%, and an unweighted TPR/TNR average of 74.2%, and substantial positive and negative predictive rates, it could truly be the springboard for a reliable model to use on future hurricanes.

**Limitations**

Each author in this paper computed two machine learning algorithms, and presented their results for time efficiency. However, this led to inconsistencies when comparing and contrasting the results of each algorithm.

Because different techniques for normalizations were used when executing the machine learning algorithms, min-max and standardization, it may have had an influence on the variability on assessing performance for each model. In future studies, using only one normalization technique when computing machine learning algorithms would be more beneficial.

A two way split of the data was implemented in this project to estimate the generalization of the algorithms, however, adding a holdout set is a better and more effective way to maximize how well the algorithms generalizes (Geron, 2023). Furthermore, sklearn’s GridSearchCV is a great tool to find the best parameters of an algorithm, but it can be computationally expensive depending on the algorithm. So, if runtime is a concern when rerunning the code, an effective alternative is sklearn’s HalvingGridSearchCV (Geron, 2023).

**Future Research**

The Neural Network model presented in this paper could be used to predict the future severity of hurricanes in an effort to mitigate large financial losses and the loss of lives of those in the path of a landfall. Based on the findings of the NOAA, the financial losses due to hurricanes and other climate-related disasters from 1980 to August 2023 totaled over $2.6 trillion. In 2022 alone, the total of climate-related disasters in the US was over $165 billion (“Hurricane Costs”, 2023).

**References**

Final dataset: <https://drive.google.com/file/d/1Xa7WVwz5VNJUAYjK2kcxxRWsEtgh-lA4/view?usp=sharing>

Data source:   
<https://www.nhc.noaa.gov/data/hurdat/hurdat2-1851-2022-050423.txt>

Data dictionary:  
<https://www.aoml.noaa.gov/hrd/data_sub/newHURDAT.html>

Category 5 definition:  
<https://www.nhc.noaa.gov/aboutsshws.php>

Geron, A. (2023). Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow. *O’Reilly.*

Shmueli, S., Gedeck, P. (2019). Data Mining for Business Analytics. *Wiley*.

“Hurricanes | National Oceanic and Atmospheric Administration.” *National Oceanic and Atmospheric Administration*, 1 May 2020, www.noaa.gov/education/resource-collections/weather-atmosphere/hurricanes.

“Hurricane Costs.” (2023, October 5th).*NOAA Office Of Coastal Management.*<https://coast.noaa.gov/states/fast-facts/hurricane-costs.html>