US Weather Severity prediction using classification model

Evelyn Zhi. Tang
Texas A M University
College Station

Julia Hann
Texas A M University
College Station

Abstract

Predicting severity of weather conditions help citizens and governments to prepare for natural disasters such as hurricanes and droughts. In the last 10 years, thanks to the breakthrough in the field of machine learning, researchers can now make better predictions on different severe weathers. We will explore and use supervised classification models in supervised machine learning to prognosticate the severity of weathers. Finally, we compare and access the chosen classification models and select the model with best performance. We also compare the performance of the same model with different tuned parameters. Our result also shows that using improper hyperparameter can lead to extremely low accuracy.

1. Introduction

As weather data availability increases, more and more research is done based on machine learning and deep learning techniques. The motivation of using machine learning techniques in weather prediction is to generate a data driven model with high accuracy. We use a few popular classification algorithms to classify the weather data: Decision tree, Random Forest, SVM, Logistic Regression, Stochastic Gradient Descent, Adaboost, Linear Discriminant Analysis, and Naive Bayes.

Preprocessing must also be done to the data. Some machine learning algorithms require numeric data, but not all the data available is numeric. As such LabelEncoder was used in order to convert the non-numeric data into numeric data. In addition, some models require that the data be normalized as they are sensitive to magnitude. As such, the data was normalized before being handled by the algorithms. Finally, the data set was split into a training and testing set. This is important for training and verification that the training was successful.

The weather data contains geological information, cities, states, time, type of weather, and severity of the weather events in the United States. Supervised learning was performed to predict the severity of the weather event.

The paper is organized as follows. Section II is about the methods used to analyze our data. Section III is our experimental results from running and implementing our algorithms. Finally Section IV is our conclusion.

2. Methods

2.1. Supervised Learning

The dataset we use for this project is the US weather event data from Kaggle. The dataset is posted by Sobhan Moosavi. It records weather events from 2016 to 2019 across multiple cities from 49 states in the U.S. These records are measured by airport based weather stations. The dataset contains numeric attributes such as latitude and longitude information, as well as categorical information such as type, severity, city, county, state.

Machine Learning models need to be trained by numeric data. So all the categorical information needs transformation. Scikit learn python package provides multiple types of encoders to convert categorical variables into numeric values. Popular options of encoders are OneHot encoder, Ordinal encoder ,Label encoder and Hashing encoders. For the weather dataset we are dealing with, we will use LabelEncoder to convert our categorical data into its numeric form. Class variables will be labelled with discrete positive numbers. Attributes like severity, type, timezone, city, county will be fed into the encoder one by one and a new dataframe is created with numeric categorical information along with the numerical information from the original dataframe.(McGinnis, 2016)

Normalization is a common practice before model training. Normalization helps machine learning models to reduce the number of iterations to converge. Normalization also helps when the used models are sensitive to magnitude. Although not every model we tried in this project is sensitive to magnitude, we will use the normalization technique in the code. Scikit Learn package provides multiple interfaces to perform numerous types of scaling. Popular options are MinMaxScaler, MaxAbsScaler, StandardScaler. In this project, we will use StandardScaler to scale our data. After scaling, the data will be with zero mean and unit vari-

ance.(Richmond)

Since there is only 1 dataset for training and testing, we need to split our dataset into training data and testing data. This can be done by using train test split methods provided by scikit learn package. We will feed our scaled attribute matrix and labels into the train test split method, and it will provide 4 outputs: the attributes for training, the attributes for testing, the labels for training, the labels for testing. Then the data is ready to be passed into the models.

2.2. Unsupervised Learning

The only unsupervised learning model we used in this project is Linear Discriminant Analysis. Unsupervised Learning does not take labels during training. LDA preserves the class information during training. More details about Linear Discriminant Analysis unsupervised model will be discussed in section 2.9

2.3. Naive Bayes

Naive Bayes is based on the Bayes probability function. It is called naive due to the fact that it is assumed that the different features are conditionally independent. Naive Bayes isn't highly referenced as an algorithm used for classification of weather, but it has been used to classify rainfall prediction. When using Naive Bayes it is important to choose explanatory variables wisely, as otherwise it could lead to a problem of overfitting. It is also important to note that for Naive Bayes to work correctly the data must be normalized. (Liu, Li, Dillon 2001)

2.4. Decision Tree

The Decision Tree algorithm is a generally popular choice due to it being able to identify the most important predictors and produces a human readable model. Decision Trees are also a popular choice for predicting various types of weather. It is also important to note that Decision Trees don't need numerical data and can work with heterogeneous data, like our data set has. Decision Trees generally work where they split the data on a variable until the leaves contain a singular constant value. The decision tree will split the data based off of the most relevant variable (Choi 2016 and Mcgovern 2017 and Mcgovern 2019).

2.5. Logistic Regression

Logistic regression classifies weathers from a probabilistic view and is considered as a linear classifier. Logistic regression is generally used for binary prediction. There is a multiple variable labelling version of Logistic regression called Softmax though. Logistic regression had been accessed by contingency statistics, and had better performance than SVM at 48-hour lead time and 72-hour lead time in prediction severe tornadic weather prediction. In practice, logistic regression is usually combined with cross validation. In the scikit learn package, there is a method called LogisticRegressionCV which combines the cross validation technique with Logistic Regression. Logistic regression is a popular model for classification tasks since the output of the model is probability. In deep learning image classification tasks, Logistic Regression as activation function is used in the last layer in the neural network. (Mercer 2009)

2.6. Random Forest

Random Forest is an algorithm that is related to the decision tree algorithm and is also widely known for use in weather. Random Forest, as it mostly a collection of decision trees, is considered to be more accurate than Decision Tree, but can produce not as sharp of a tree. Random Forest is capable of telling you the importance of each variable to the data set. The collection, or ensemble, or trees differentiates each tree by training them on seperate sets of data. These different sets of data come from bootstrapping the original data set. The end result is each tree will have different most relevant variables at each step. Random Forest utilizes the bagging technique and the feature random selection to make better predictions. (Choi 2016 and Mcgovern 2017 and Williams 2009)

2.7. Adaboost

Adaboost algorithm is also called Adaptive Boosting. Unlike other ensemble algorithms having multiple classifiers at the same time and aggregate their results, Adaboost algorithm improves a weak model through multiple iterations and in the end it ends up with a strong model. Adaboost can find a classification rule that minimizes the misclassification loss. The misclassification loss can be defined as $C(x) = \text{argmax}_k \text{Prob}(C = k, X = x)$. Adaboost aggregates multiple weak models sequentially(aggregates the models at all the stages) to approximate Bayes classifier (Ji Zhu, Saharon Rosset, Hui Zou, Trevor Hastie, 2006).

At each iteration, Adaboost algorithm will find out the mis-classified points.(Freund and Schapire ,1997) Then it adds weights to the mis-classified points and reduces weights to the correctly classified points, so the next classifier / the improved classifier can pay more attention to the mis-classified point. (Rojas, 2009) In this project, adaboost will classify the severity of different weather events. The attributes of the weather events will be used to build the classifier. The classifier will be a weak classifier at the beginning. The final classifier will be a linear combination of all the models at each stage.

2.8. Support Vector Machine

The Linear SVM(support vector machine) classifies data into regions in high dimensional space by using hyperplane. It defines the margin as the width between support vectors. Support vectors are the closest points towards the classifier.

Then it maximizes the margin to find the optimal classifier. Hard Margin suppose data are linearly separable. However, most of the real word data are linearly inseparable. To find the optimized classifier for linearly inseparable data, Soft Margin method and Kernel method were developed. The Soft Margin method allows the models to make some mistakes while at the same time minimizing the total sum of loss.The Kernel method transforms the data from its original coordinate system to another coordinate system. The dimension of the coordinate system can remain the same after changing the basis. In sklearn, it provides an api called LinearSVC, which is the linear support vector classifier. (Gordon)

2.9. Linear Discriminant Analysis

Latent Discriminant Analysis is a generalized form of Fisher's Linear discriminant. Latent Discriminant Analysis tries to treat y variable as the linear combination of independent variables x(wiki). It is similar to Principal Component Analysis in a way. Principal Component Analysis does not consider the class information of the data, that means, it assumes that all the data are from the same class. Latent Discriminant Analysis, however, takes into consideration of class information of the data. Both of them reduce the dimensionality of the data. Latent Discriminant Analysis projects the data from a high dimensional space into a lower dimensional space while at the same time during projection, it tries to maximize the between class distance of all the classes. The between class distance is measured by the variance. Since classes are taken into account, after reducing the dimension class information is also preserved and we take advantage of that to classify our model. Scikit learn package provides api to use the Latent Discriminant Analysis model.

2.10. Stochastic Gradient Descent

To understand Stochastic Gradient Descent, we first need to know Gradient Descent. Gradient Descent is overall a very popular algorithm. General Gradient Descent has been used frequently to predict weather, but not too many examples of stochastic gradient descent exist. Stochastic Gradient Descent is a variation of Gradient Descent. Instead of calculating the gradient based on all the data, stochastic gradient descent randomly samples a single record to update the gradient. The algorithm ends when it achieves linear convergence. Stochastic gradient descent has much faster convergence than regular gradient descent. (Bottou)

3. Experiment and Evaluation

3.1. Tools

The metric module of scikit learn package provides numerous evaluation tools for different tasks(classification, clustering, regression). In this project, we will use the accuracy score method and confusion matrix to evaluate our models. The accuracy score method calculates the percentage of the predicted labels of the model which matches the true labels in the dataset. Measuring accuracy using percentage is one of the most common methods to evaluate classification models. We also plot a confusion matrix for each individual model. The confusion matrix in scikit learn package shows the number of correctly predicted labels as well as the misclassified labels. The elements on the diagonal are the correctly predicted labels and the elements that are off diagonal are misclassified labels. The confusion matrix visually gives an overview of what labels are classified into what labels. The vertical axis represents the true labels and the horizontal axis represents the predicted labels.

3.2. Naive Bayes

The performance of Naive Bayes exceeds our expectations. The accuracy of the Gaussian Naive Bayes algorithm is 79.99 percent of accuracy. The confusion matrix shows that in testing data, data records with actual class label 0 and class label 2 tend to be classified into class 1. The high accuracy might of Gaussian Naive Bayes model might due to the simplicity of the weather dataset we used.

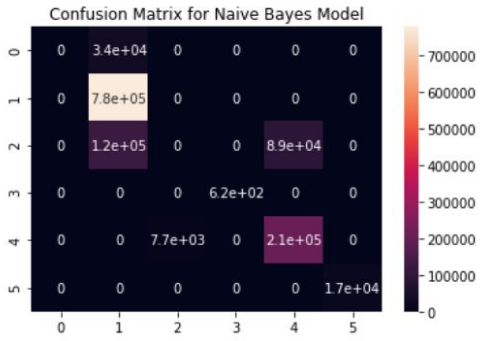

Figure 1. Confusion matrix for Naive Bayes model

3.3. Decision Tree

The Decision Tree algorithm also performs really well. The accuracy of the Decision Tree Algorithm is around 80 percent. We had implemented the decision tree model with different hyperparameter values. Hyperparameter "criteria" could be set to "gini" or "entropy". The "criteria" refers to how to measure impurity. Max feature hyperparameter is to set the maximum number of features to consider at each split. The result shows that when using the log2 option in max feature hyperparameter, the accuracy is lower compared with other decision tree models. The reason might be our dataset does not have many attributes (features), setting max features = "log2" might consider too few features.

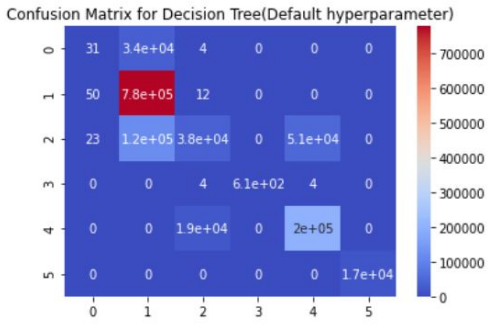

Figure 2. Confusion matrix for Decision Tree model. Criteria set to be gini impurity and splitter will split the node by best split.

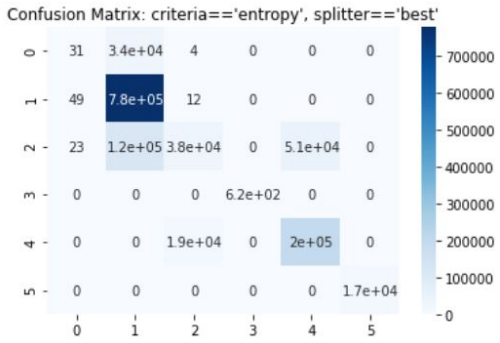

Figure 3. Confusion matrix for Decision Tree model. Criteria set to be Entropy and splitter will split the node by best split.

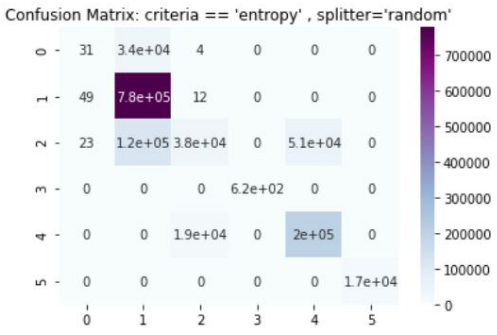

Figure 4. Confusion matrix for Decision Tree model. Criteria set to be Entropy and the splitter will split the node by random split.

3.4. Stochastic Gradient Descent

Stochastic gradient descent did not perform as well as the Decision Tree algorithm. The accuracy of Stochastic Gradient Descent is 78.61 percent. The confusion matrix showed that in testing data, records with actual class label 1 tend to be classified into other classes. And records with actual label 4 tend to be classified into class 2 and class 3.

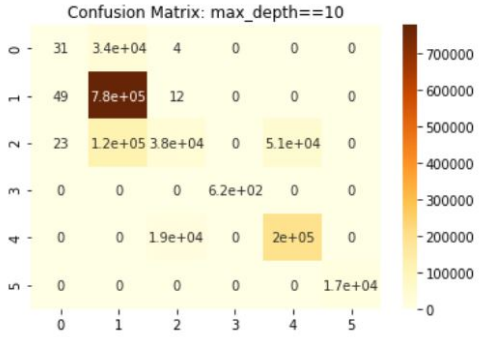

Figure 5. Confusion matrix for Decision Tree model. Default options for criteria and spiltter, and with maximum depth of the decision tree to be 10.

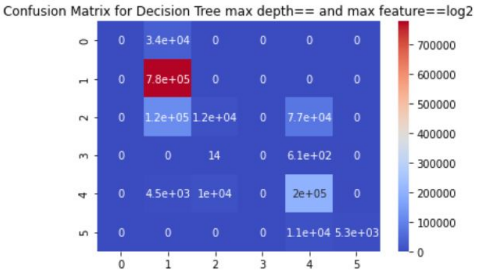

Figure 6. Confusion matrix for Decision Tree model. Default options set for criteria and splitter. Maximum depth of the decision tree set to be 5 and maximum number of features considered at each node is set to be number of log base 2 of the number of total features

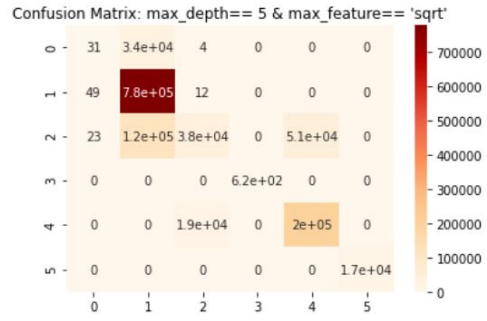

Figure 7. Confusion matrix for Decision Tree model. Default options set for criteria and splitter. Maximum depth of the decision tree set to be 5 and maximum number of features considered at each node is set to be number of square root of the number of total features

3.5. Logistic Regression

Logistic Regression did not perform well in classifying the weather severity in this project. The accuracy of Logis-

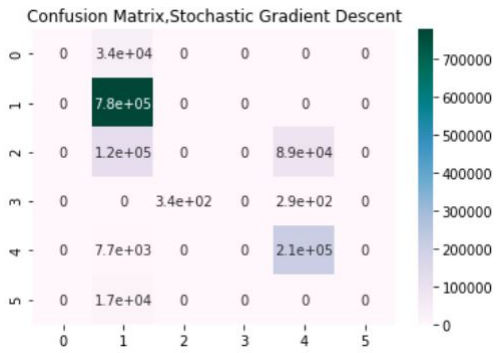

Figure 8. Confusion matrix for Stochastic Gradient Descent model. With tolerance set to be 0.005 and with default options for other hyperparameters.

tic Regression is around 54.47 percent. Nearly half of the testing data was mis-classified. In testing data, records with actual class label 0 tend to be classified into class 1, and records with class label 1 tend to be classified into class 0. Records with class label 2 did not get classified correctly completely. Most of the mis-classification happened when trying to classify class 1 and class 2 data.

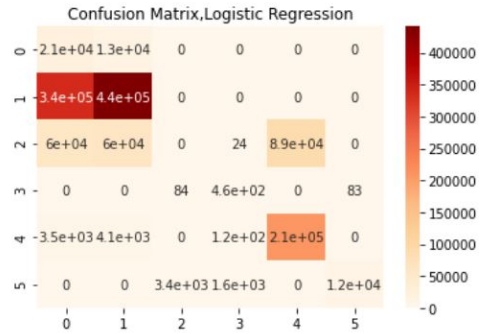

Figure 9. Confusion matrix for Logistic Regression model. The model with worst performance in this project. The low accuracy might due to the model itself or improper hyperparameter. Penalty is set to l2 penalty, solver is set to be lbfgs,fit intercept option is set to be false.

3.6. Random Forest

Random Forest achieved a high accuracy when predicting labels. As a variation from the family of Decision Tree models, the accuracy of the Random Forest model is close to the accuracy of the Decision Tree models we implemented. The accuracy of the Random Forest model is at 82.097percent. And the confusion matrix of the Random Forest model is similar to the confusion matrices of the Decision Tree models.

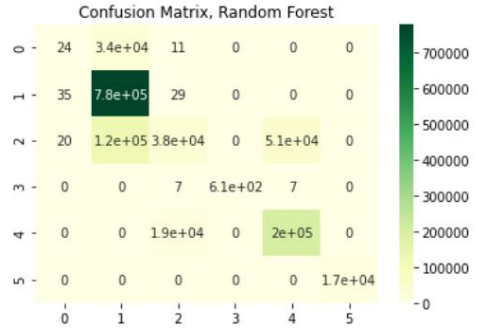

Figure 10. Confusion matrix for Random Forest model. with n estimator being set to 80 and max number of features set to auto detection, and criteria is set to gini impurity.

3.7. Adaboost

The accuracy of the Adaboost algorithm is 80.60 percent. From the confusion matrix we can tell records in class 0 and class 2 tend to be classified into class 1. The mis-classification pattern is very similar to the Decision Tree model. It makes sense since Adaboost is a variation of the family of Decision Tree models.

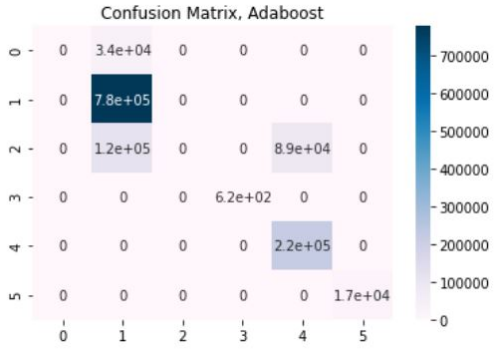

Figure 11. Confusion matrix for Adaboost model. All the hyperparameters are using default options for the Adaboost model provided by scikit learn.

3.8. Linear Discriminant Analysis

We had implemented 3 variations of Linear Discriminant Analysis models with hyperparameter tuning. All three models have accuracy around 78 percent. The shape of the confusion matrices of these 3 variations are similar. Some testing data with actual class label 2, or label 3 or label 5 did not get correctly classified. Overall, the Linear Discriminant Analysis model performed well. It is the only unsupervised learning method we tried in this project.

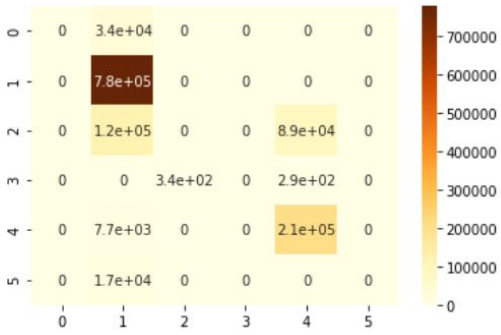

Figure 12. Confusion matrix for the Linear Discriminant Analysis (LDA) model. The only unsupervised learning model used in this project. Solver set to be sigular value decomposition and leaving other hyperparameters set to be default options.

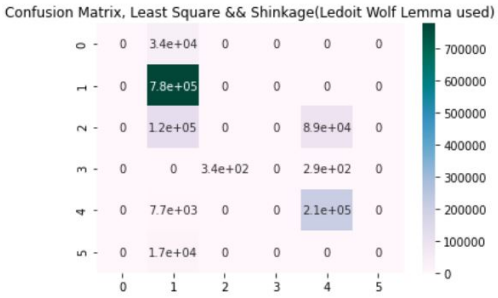

Figure 13. Confusion matrix for Linear Discriminant Analysis(LDA) model. Solver set to be Least Sqaure solution with auto shrinkage.

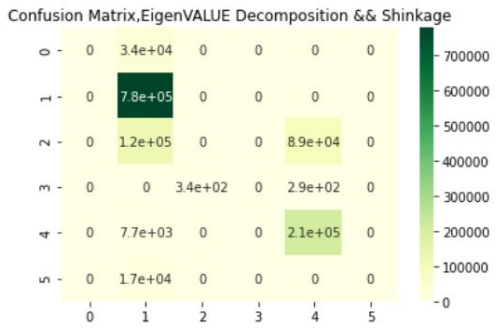

Figure 14. Confusion matrix for Linear Discriminant Analysis(LDA) model. Solver set to be eigenvalue decomposition with shrinkage set to be auto.

4. Conclusion

As technology continues to progress, predicting the weather will be as important as ever. Using the data that was gathered we were able to successfully predict the weather around various parts of the United States.

We expected Decision Tree or Logistic Regression to be the best algorithm to predict the weather (Huang, Liu, Ling 2003 and Pranckevičius, Marcinkevičius 2017). All methods, but Logistic Regression, were around 80 percent of accuracy. Random Forest had the highest accuracy at 82 percent and Logistic Regression was the worst at 54 percent. Neither of the methods we expected to have the best results had the best results. We are pretty happy with the results from our tests and general high accuracy of our models.

If we were to do another experiment on this dataset, we would focus on the Random Forest and see if there were other ways we could boost the accuracy. We will also try to tune the parameter of the Logistic Regression to see if we can get a better result in future to improve the prediction results.

In this project, we learned that for simple datasets, simple models like Naive Bayes or Decision Tree models performed really well. Logistic Regression did not perform well, howeverm this might due to setting improper hyperparameters. In this project, we have tried different values of hyperparameters in the same model. For example, we have tuned the hyperparameters for the decision tree model and the Linear Discriminant Analysis(LDA) model to see if they will acheive different results.

One more thing to notice, the dataset is highly skewed. From the confusion matrices, it is obvious that most of the data records are with true class label 1. The data is inbalance. This explained why Logistic Regression performed the worse. Since the hyperparameter in Logistic Regression is set to be "balance". This experiment shows that improper hyperparameter can lead to catastrophic consequences. Choosing the right hyperparameters plays an important role in training machine learning models. When training models, we need to let models know that our dataset is skewed or inbalanced, otherwise, models can perform really bad.

5. Reference

[1] S. Choi, Y. J. Kim, S. Briceno, and D. Mavris, "Prediction of weather-induced airline delays based on machine learning algorithms, " 2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC), 2016.

[2] J. Huang, J. Lu, and C. Ling, "Comparing naive Bayes, decision trees, and SVM with AUC and accuracy," Third IEEE International Conference on Data Mining, 2003.

[3] S. D. Jadhav and H. P. Channe, "Comparative Study of K-NN, Naive Bayes and Decision Tree Classification Techniques," International Journal of Science and Research (IJSR), vol. 5, no. 1, pp. 1842–1845, May 2016.

[4] J. Liu, B. Li, and T. Dillon, "An improved naive Bayesian classifier technique coupled with a novel input solution method [rainfall prediction], " IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and

Reviews), vol. 31, no. 2, pp. 249–256, 2001.

[5] A. Mcgovern, K. L. Elmore, D. J. Gagne, S. E. Haupt, C. D. Karstens , R. Lagerquist, T. Smith, and J. K. Williams, "Using Artificial Intelligence to Improve Real-Time Decision-Making for High-Impact Weather," Bulletin of the American Meteorological Society, vol. 98, no. 10, pp. 2073–2090, 2017.

[6] A. Mcgovern, C. D. Karstens, T. Smith, and R. Lagerquist, "Quasi-Operational Testing of Real-Time Storm-Longevity Prediction via Machine Learning," Weather and Forecasting, vol. 34, no. 5, pp. 1437–1451, 2019.

[7] A. E. Mercer, C. M. Shafer, C. A. Doswell, L. M. Leslie, and M. B. Richman, "Objective Classification of Tornadic and Nontornadic Severe Weather Outbreaks," Monthly Weather Review, vol. 137, no. 12, pp. 4355–4368, 2009.

[8] S. Moosavi, M. H. Samavatian, A. Nandi, S. Parthasarathy, and R. Ramnath, "Short and Long-term Pattern Discovery Over Large-Scale Geo-Spatiotemporal Data," Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD 19, 2019.

[9] M. A. Nayak and S. Ghosh, "Prediction of extreme rainfall event using weather pattern recognition and support vector machine classifier, " Theoretical and Applied Climatology, vol. 114, no. 3-4, pp. 583–603, Jun. 2013.

[10] T. Pranckevičius and V. Marcinkevičius, "Comparison of Naive Bayes, Random Forest, Decision Tree, Support Vector Machines, and Logistic Regression Classifiers for Text Reviews Classification," Baltic Journal of Modern Computing, vol. 5, no. 2, 2017.

[11] Y. Radhika and M. Shashi, "Atmospheric Temperature Prediction using Support Vector Machines," International Journal of Computer Theory and Engineering, pp. 55–58, 2009.

[12] J. K. Williams, D. Ahijevych, S. Dettling, and M. Steiner, "Combining observations and model data for short-term storm forecasting," Remote Sensing Applications for Aviation Weather Hazard Detection and Decision Support, 2008.

[13] L. Bottou, "Large-Scale Machine Learning with Stochastic Gradient Descent," Proceedings of COMPSTAT2010, pp. 177–186, 2010.

[14] Ji Zhu, Saharon Rosset, Hui Zou, Trevor Hastie "Multi-class AdaBoost". January 12, 2006

[15] Ra 'ul Rojas, AdaBoost and the Super Bowl of Classifiers ,A Tutorial Introduction to Adaptive Boosting. Computer Science Department Freie University at Berlin, Christmas 2009

[16] R. Berwick, Village Idiot, An Idiot's guide to Support vector machines (SVMs), Lecture Slides.

[17] Bernhard E. Boser, Isabelle M. Guyon, Vladimir N. Vapnik. A Training Algorithm for Optimal Margin Classi ers.

[18] Geoff Gordon, Support Vector Machines and Kernel Methods, Lecture Slides, Jun, 2004.

[19] Aleix M. MartõÂnez, Member, IEEE, Avinash C. Kak. "PCA versus LDA", Feb, 2001

[20] Richmond, Michael. The Magnitude Scale. http://spiff.rit.edu/classes/phys440/lectures/mag/mag.html.

[21] Wikipedia,https://en.wikipedia.org/wiki/Logistic_regression https://en.wikipedia.org/wiki/Linear_discriminant_analysis

[22] Scikit Learn https://scikit-learn.org/stable/modules/svm.html https://scikit-learn.org/stable/modules/naive_bayes.html https://scikit-learn.org/stable/modules/sgd.html https://scikit-learn.org/stable/modules/cross_validation.html

[23] Wang, Zhangyang. All course materials and lectures, resourses on personal website http://people.tamu.edu/~atlaswang/SpringCSCE421.html