A **Comparative Study of Weather Prediction Using Multiple Machine Learning Classifier Models**

John Cairo Q. Minerva, Maria Joeanne A. Gison, Ariane Marie L. Lavilla

Bachelor of Science in Computer Science

College of Information and Communications Technology, West Visayas State University

CCS 248: Artificial Neural Networks

Mr. Jan Carlo T. Arroyo

December 22, 2022

Introduction

A. General Topic

The weather is a natural phenomenon that is constantly altered by changes in many atmospheric variables. It is also the condition of the atmosphere in terms of temperature, moisture, air pressure, precipitation, and other factors at a specific time and location (StudyMoose, 2022). Weather forecasting or prediction has been a blessing of modern technology. It enables people to understand the nature of the atmosphere. Precise weather forecasting is one of the greatest challenges that people, and the modern world are experiencing today. Contrary to traditional methods, modern weather forecasting involves a combination of computer models, observation, and pattern recognition along with various trends (Hasan et al., 2019).

According to Chauhan, 2022, Naive Bayes is a probabilistic machine learning algorithm based on the Bayes Theorem, used in a wide variety of classification tasks. Bayes' Theorem is a simple mathematical formula used for calculating conditional probabilities. A conditional probability is a measure of the probability of an event occurring given that another event has (by assumption, presumption, assertion, or evidence) occurred. On the other hand, a random forest consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the highest number of votes becomes the model's prediction (Yiu, 2019). Logistic Regression is defined as a statistical analysis method used to foresee a binary outcome, such as yes or no, based on prior observations of the data set (Lawton et al, 2022).

In the modern world, more precise weather forecasts are always in demand. There are several needs for weather forecasting. Improved accurate weather predictions are always in demand from businesses, including farms, factories, satellite launchers, and even the general people. This paper focuses on weather forecasting by utilizing multiple machine learning algorithms namely Naive Bayes, Logistic Regression, and Random Forest to predict the weather based on the variables which are precipitation, maximum and minimum temperature, wind levels, and weather type and comparing their accuracies which can predict the outcome with the highest precision.

B. Problem

- What are the main factors needed to be able to predict the weather?
- Is there a significant difference in the accuracy of the results of multiple machine algorithms in weather forecasting?
- Does the model able to predict drastic weather changes by using climatic data elements?

C. Solution

- Analyze the dataset's weather data elements (precipitation, maximum and minimum temperatures, and wind levels).
- Build a machine learning model to predict the weather using multiple classifier algorithms and evaluate their accuracy percentage.
- Compare the model's results by its predicted and actual results using a confusion matrix.

Related Systems and Studies

Weather forecasting is important for people. It helps to make more informed daily decisions and to keep out of danger. Accurate Weather forecasting has been one of the most challenging problems around the world. Unlike traditional methods, modern Weather forecasting involves a combination of computer models, observation (by use of balloons and satellites), and knowledge of trends and patterns Using these methods are reasonably accurate forecasts can be made. Most of the computer models used for forecasting are run by forecast models based on complex formulas. The study of (Verma, et.al 2020) presents a review of Weather Forecasting using Artificial Neural Network (ANN) and studies the benefit of using it. An artificial Neural Network is an adaptive system that changes its structure based on external or internal information that flows through the network. The researchers' paper provides a survey of the available literature on some methodologies employed by different researchers to utilize ANN for Weather Forecasting.

However, the dissertation of (Tanvir-Hasan et al, 2019) in which the researchers have done many things to establish a relationship of recent input data and target data which is linear. But practically the relationship is nonlinear. After establishing the nonlinearity, many models have been made to get future weather data. As the weather data is nonlinear, Artificial Neural Network (ANN) has become an effective way of predicting weather data precisely and accurately. Neural Network is a system that can be trained with certain input and output. It creates its own structure based upon how it is trained. The researchers predicted weather data for a particular month of a season and compared the results for different functions and training methods of ANN.

Moreover, the research of (Durran et al, 2020) presented a significantly improved data-driven global weather forecasting framework using a deep convolutional neural network (CNN) to forecast several basic atmospheric variables on a global grid. New developments in this framework include an off-line volume-conservative mapping to a cubed-sphere grid, improvements to the CNN architecture, and the minimization of the loss function over multiple steps in a prediction sequence. The cubed-sphere remapping minimizes the distortion on the cube faces on which convolution operations are performed and provides natural boundary conditions for padding in the CNN. The improved model produces weather forecasts that are indefinitely stable and produce realistic weather patterns at lead times of several weeks and longer. For short- to medium-range forecasting, our model significantly outperforms persistence, climatology, and a coarse-resolution dynamical numerical weather prediction (NWP) model. Unsurprisingly, the forecasts are worse than those from a high-resolution state-of-the-art operational NWP system. The data-driven model is able to learn to forecast complex surface temperature patterns from a few input atmospheric state variables. On annual time scales, the model produces a realistic seasonal cycle driven solely by the prescribed variation in top-of-atmosphere solar forcing. Although it currently does not compete with operational weather forecasting models, the data-driven CNN executes much faster than those models, suggesting that machine learning could prove to be a valuable tool for large-ensemble forecasting.

Furthermore, the study of (Fente, et.al 2018) used different weather parameters that were collected from the national climate data center then using the Long-short term memory (LSTM) technique, the neural network is trained for different combinations. In the prediction of future weather conditions using LSTM the neural network is trained using different combinations of weather parameters, the weather parameters used are temperature, precipitation, and wind speed. pressure, dew point visibility, and humidity. After training of the LSTM model using these parameters, the prediction of future weather is done.

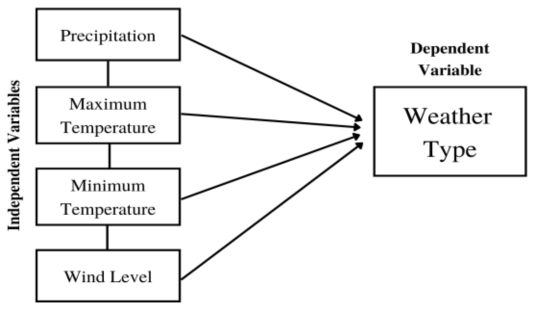
Methods

In this section, the proposed model to predict weather using multiple machine learning classifiers are introduced. The proposed model was coded and executed using Python 3 and Jupyter Notebook 6.3.0 in an Intel(R) Core(TM) i3-1005G1 CPU @ 1.20GHz 1.20 GHz laptop.

A. Conceptual Framework

The conceptual framework shows the relationship between the independent and dependent variables in the study. In this research, the independent variables are precipitation, maximum and minimum temperature, and wind level which are factors that were needed in order to predict the weather. While the dependent variable is the weather type (drizzle, fog, rain, and sun) which is the outcome based on the factors affecting it.

Figure 1. Conceptual Framework



B. Dataset

A publicly available dataset for weather prediction obtained from https://www.kaggle.com/datasets/ananthr1/weather-prediction with 1461 instances collected from climatic data was utilized. The dataset comprises 6 variables where variables 1 to 5 are the input features, while variable 6 is the output feature. The dataset was divided into 70% training, and 30% testing sets used in the neural networks. The indexed description of the dataset is presented in Table 1.

Table 1 Dataset Features

No.	Variables	Type
1	date	date - object
2	precipitation	millimeter - float
3	temp_max	Celcius - float
4	temp_min	Celcius - float
5	wind	mph - float
6	weather	weather type - object

C. Algorithm Process

C.1. Data Preprocessing

Before using the dataset as input to the machine learning model, the preprocessing step is first initiated. The following steps were conducted as follows:

1. Checking for null values
2. Describing the instances' data types
3. Creating a weather label variable from the weather column
4. Deleting the weather column and naming the weather label into weather variable

There were no instances that contain null values subject to removal or for filling yet the weather variable contains an object data type so it should be changed into a float data type instead so that the model could recognize it and predict it more accurately. The weather data were changed and labeled as 0: 'drizzle', 1: 'fog', 2: 'rain', 3: 'snow', and 4: 'sun'.

C.2. Testing and Training the Data and Model Building

The ‘date’ variable was removed from the dataset and the data were divided into independent and dependent components. For the x input the ‘weather’ variable was dropped remaining only 5 variables while the y output only contains the ‘weather’ variable for classifying the output. Then, the dataset was split into a 70:30 ratio, 70% for training and 30% for testing. The shapes of the training and testing data were shown in which (1022, 5) for x training data, (1022,) for y training data, (439, 5) for x testing data, and (439,) for y testing data.

Feature scaling was then implemented in the model using the sklearn python library. According to (Vashisht, 2022) feature scaling is a technique for normalizing the variety of independent variables or features in data. It is often carried out during the data preparation stage and is sometimes referred to as data normalization in the context of data processing. Each feature's value in the data is standardized so that it has a zero mean and unit variance. In order to create the new data point, the following formula is used after determining the distribution mean and standard deviation for each feature:

$$x' = \frac{x - \bar{x}}{\sigma}$$

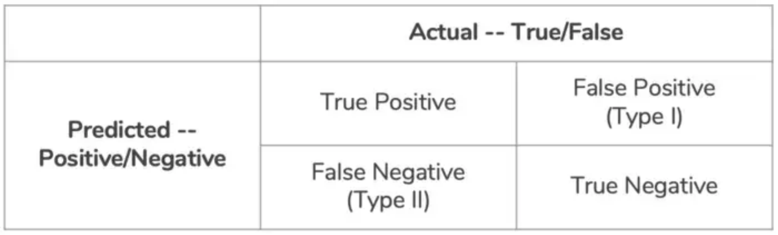
Here, σ is the standard deviation of the feature vector, and $\bar{x}$ is the average of the feature vector.

After feature scaling the input and output data, the next step is building the model using multiple classifiers namely Logistic Regression, Gaussian Naive Bayes, and Random Forest. These classifiers are utilized using the sklearn python library. The x input data and y output data were fitted into the classifiers in building the machine learning model.

D. Evaluation Metrics

To test the accuracy and performance of the model, the confusion matrix was considered as the evaluation technique. A confusion matrix is a table that lists how many guesses a classifier made correctly and incorrectly. It is employed to evaluate a classification model's effectiveness. It may be used to calculate performance measures like accuracy, precision, recall, and F1-score in order to assess the effectiveness of a classification model (Suresh, 2021). Confusion matrices are a widely used measurement when attempting to solve classification issues. Both binary classification and multiclass classification issues may be solved with it (Kulkarni et al., 2020). Figure 2 shows the types of errors in the Confusion Matrix (Ragan, 2018).

Figure 2. Types of Errors



	Actual -- True/False	
Predicted -- Positive/Negative	True Positive	False Positive (Type I)
	False Negative (Type II)	True Negative

From our confusion matrix, we can calculate five different metrics measuring the validity of our model.

- Precision – how many correctly predicted cases actually turned out to be positive. The formula for precision is shown below:

$$Precision = \frac{TP}{TP + FP}$$

- Recall - how many of the actual positive cases we were able to predict correctly with our model. The value of recall can be solved by using the formula:

$$Recall = \frac{TP}{TP + FN}$$

- F-1 score - is a harmonic mean of Precision and Recall, and so it gives a combined idea about these two metrics. It is maximum when Precision is equal to Recall. The f-1 score is calculated using:

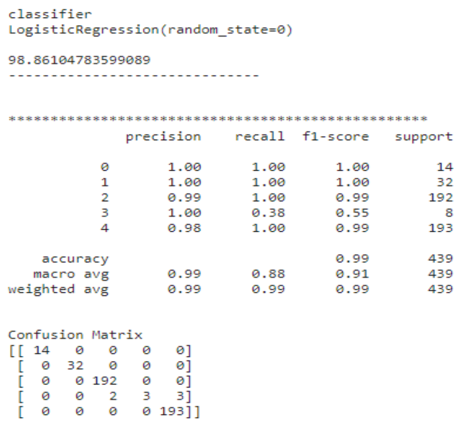
$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Results and Discussion

The machine learning model was coded and executed in Windows 11 Home Single Language version 21H1 environment using Python 3.10, Jupyter Notebook 6.3.0, and Google Colab in an Asus Vivobook with a 1.20GHz Quad-core i3 10th gen processor, 8GB RAM, and 2GB Intel UHD Graphics Card. The results of each classifier model were analyzed by their accuracy, classification report, and confusion matrix.

Figure 3 shows the classification report, accuracy, and confusion matrix of Logistic Regression. With random_state=0, we get the same train and test sets across different executions.

Figure 3. Logistic Regression Classification Report



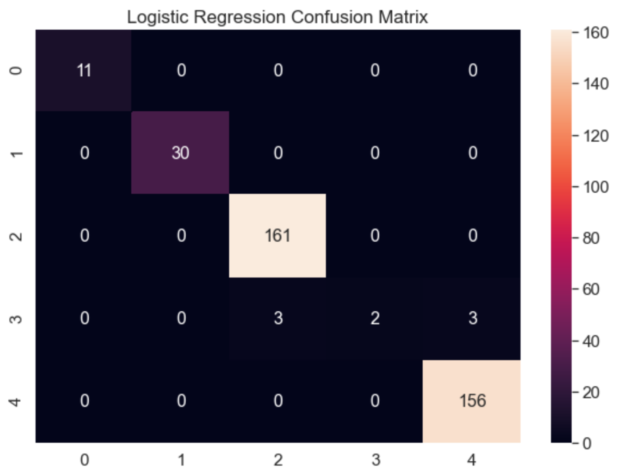
```
classifier
LogisticRegression(random_state=0)

98.86104783599089
------------------------------

************************************************
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        14
           1       1.00      1.00      1.00        32
           2       0.99      1.00      0.99       192
           3       1.00      0.38      0.55         8
           4       0.98      1.00      0.99       193

    accuracy                           0.99       439
   macro avg       0.99      0.88      0.91       439
weighted avg       0.99      0.99      0.99       439

Confusion Matrix
[[ 14   0   0   0   0]
 [  0  32   0   0   0]
 [  0   0 192   0   0]
 [  0   0   2   3   3]
 [  0   0   0   0 193]]
```

By implementing the logistic regression classifier, the weather label 0 (drizzle) obtained a score of 1.00, 1.00, and 1.00 for precision, recall, and f-1 score respectively. The weather label 1 (fog) acquired a score of 1.00, 1.00, and 1.00 for precision, recall, and f-1 score respectively. The weather label 2 (rain) got a score of 0.99, 1.00, and 0.99 for precision, recall, and f-1 score respectively. The weather label 3 (snow) attained a score of 1.00, 0.38, and 0.55 for precision, recall, and f-1 score respectively. The weather label 4 (sun) achieved a score of 0.98, 1.00, and

0.99 for precision, recall, and f-1 score respectively. With an overall accuracy rate of 98.8104783599 percent.

Figure 4 shows the confusion matrix graph plot of the logistic regression classifier. It was coded using matplotlib and seaborn python libraries.

Figure 4. Logistic Regression Confusion Matrix

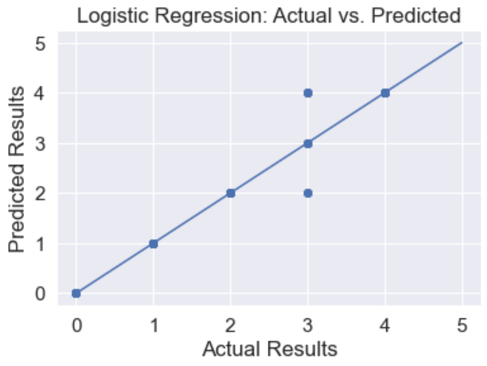


By understanding the confusion matrix of the derived results of logistic regression, the classifier was able to predict all scores for weather labels 0 (drizzle), 1 (fog), 2 (rain), and 4 (sun). While it wrongly forecasted three values for row 3 (snow). It predicted three values for column 2 (rain), two values for column 3 (snow), and another three for column 4 (sun).

Figure 5 indicates the scatterplot of actual vs. predicted weather values of logistic regression as well as the test and train scores of the aforementioned model. It was coded using the matplotlib python library.

Figure 5. Logistic Regression Actual vs. Predicted Results

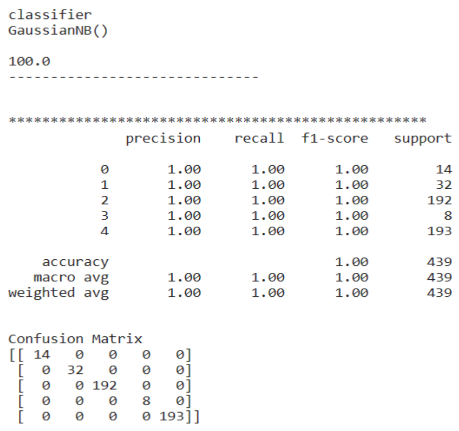
```
Train score:
0.9941291585127201
Test score:
0.9886104783599089
```



The logistic regression gained a train score of 0.994129 and a test score of 0.9886104. The logistic regression classifier was able to give an accurate weather prediction with only a narrow margin of error.

Figure 6 shows the classification report, accuracy, and confusion matrix of Gaussian Naïve Bayes.

Figure 6. Gaussian Naïve Bayes Classification Report

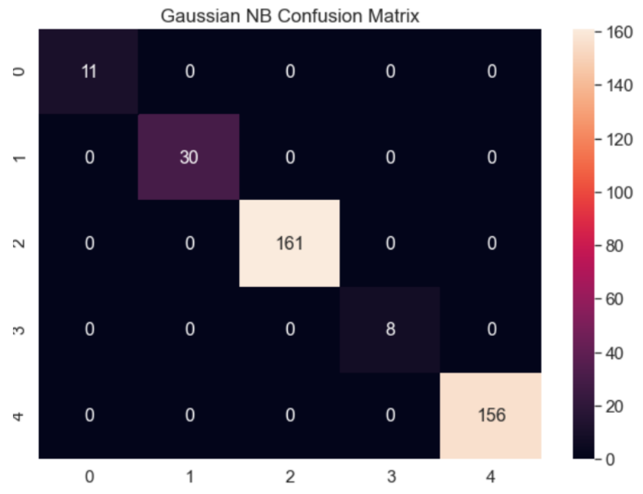


```
classifier
GaussianNB()

100.0
------------------------------

**************************************************
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        14
           1       1.00      1.00      1.00        32
           2       1.00      1.00      1.00       192
           3       1.00      1.00      1.00         8
           4       1.00      1.00      1.00       193

    accuracy                           1.00       439
   macro avg       1.00      1.00      1.00       439
weighted avg       1.00      1.00      1.00       439

Confusion Matrix
[[ 14   0   0   0   0]
 [  0  32   0   0   0]
 [  0   0 192   0   0]
 [  0   0   0   8   0]
 [  0   0   0   0 193]]
```

By implementing the gaussian naïve bayes classifier, all the weather labels (0 (drizzle), 1 (fog), 2 (rain), 3 (snow), and 4 (sun) obtained the scores of 1.00, 1.00, and 1.00 for precision, recall, and f-1 score, respectively. This results in an overall accuracy rate of 100.0 percent.

Figure 7 shows the confusion matrix graph plot of the gaussian naïve bayes classifier. It was coded using matplotlib and seaborn python libraries.

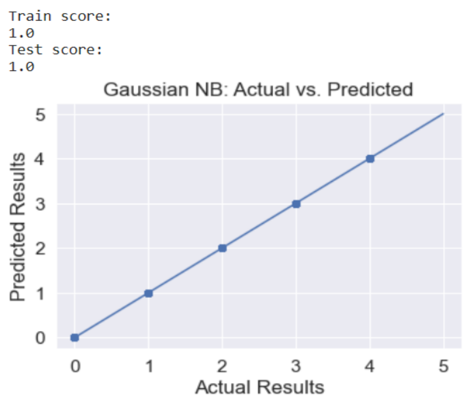
Figure 7. Gaussian Naïve Bayes Confusion Matrix



By understanding the confusion matrix of the derived results of gaussian naïve bayes, the classifier was able to correctly predict all scores for weather labels 0 (drizzle), 1 (fog), 2 (rain), 3 (snow), and 4 (sun).

Figure 8 indicates the scatterplot of actual vs. predicted weather values of gaussian naïve bayes as well as the test and train scores of the aforementioned model. It was coded using the matplotlib python library.

Figure 8. Gaussian Naïve Bayes Actual vs. Predicted Results

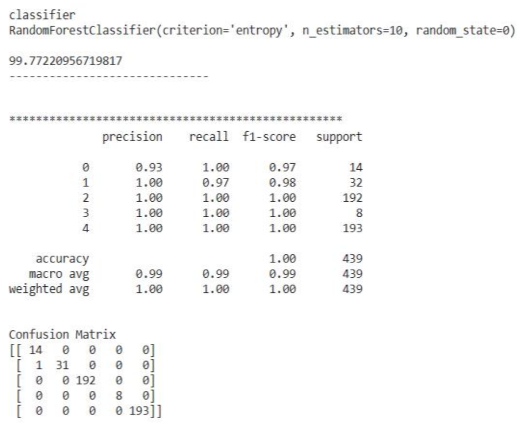


```
Train score:
1.0
Test score:
1.0
```

The gaussian naïve bayes gained a train score of 1.0 and a test score of 1.0. The gaussian naïve bayes classifier was able to accurately predict the weather.

Figure 9 shows the classification report, accuracy, and confusion matrix of Random Forest. With criterion='entropy', n_estimators=10, and random_state=0, we get the same train and test sets across different executions.

Figure 9. Random Forest Classification Report

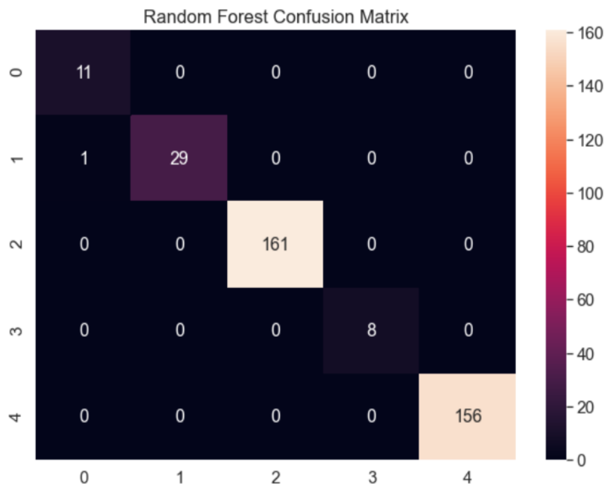


```
classifier
RandomForestClassifier(criterion='entropy', n_estimators=10, random_state=0)

99.77220956719817
------------------------------


**************************************************
              precision    recall  f1-score   support

           0       0.93      1.00      0.97        14
           1       1.00      0.97      0.98        32
           2       1.00      1.00      1.00       192
           3       1.00      1.00      1.00         8
           4       1.00      1.00      1.00       193

    accuracy                           1.00       439
   macro avg       0.99      0.99      0.99       439
weighted avg       1.00      1.00      1.00       439


Confusion Matrix
[[ 14   0   0   0   0]
 [  1  31   0   0   0]
 [  0   0 192   0   0]
 [  0   0   0   8   0]
 [  0   0   0   0 193]]
```

By implementing the random forest classifier, the weather label 0 (drizzle) obtained a score of 0.93, 1.00, and 0.97 for precision, recall, and f-1 score respectively. The weather label 1 (fog) acquired a score of 1.00, 0.97, and 0.98 for precision, recall, and f-1 score respectively. The weather label 2 (rain) got a score of 1.00, 1.00, and 1.00 for precision, recall, and f-1 score respectively. The weather label 3 (snow) attained a score of 1.00, 1.00, and 1.00 for precision, recall, and f-1 score respectively. The weather label 4 (sun) achieved a score of 1.00, 1.00, and 1.00 for precision, recall, and f-1 score respectively. With an overall accuracy rate of 99.77220956719817 percent.

Figure 10 shows the confusion matrix graph plot of the random forest classifier. It was coded using matplotlib and seaborn python libraries.

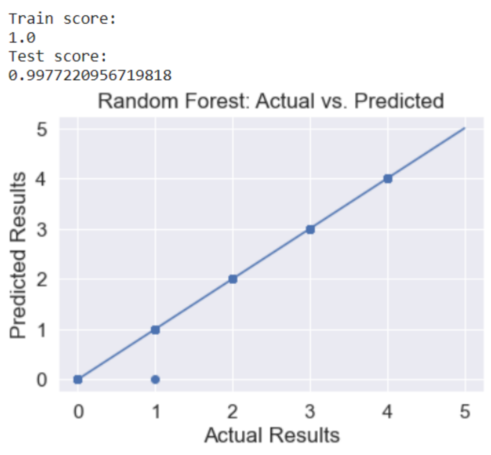
Figure 10. Random Forest Confusion Matrix



By understanding the confusion matrix of the derived results of random forest, the classifier was able to predict all scores for weather labels 0 (drizzle), 2 (rain), 3 (snow), and 4 (sun). While it wrongly forecasted two values for row 1 (fog). It gave one value for column 0 (drizzle) and 29 for column 1 (fog).

Figure 11 indicates the scatterplot of actual vs. predicted weather values of random forest as well as the test and train scores of the aforementioned model. It was coded using the matplotlib python library.

Figure 11. Random Forest Actual vs. Predicted Results



```
Train score:
1.0
Test score:
0.9977220956719818
```

The random forest gained a train score of 1.0 and a test score of 0.9977220956719818. The random forest classifier was able to give an accurate weather prediction with only a narrow margin of error.

Conclusion

This study benchmarks the performance of three machine learning classifier models namely Logistic Regression, Gaussian Naïve Bayes, and Random Forest which were used to predict the weather based on five variables - precipitation, maximum and minimum temperature, wind levels, and weather type. While such prediction methods have been implemented previously, this study further compares the performance and accuracy to find out which machine learning classifier algorithm gave the most accurate weather prediction.

The results revealed that the Gaussian Naïve Bayes classifier model had the highest accuracy in predicting the weather when compared with Logistic Regression and Random Forest with 100%, 98.9%, and 99.8%, respectively.

Supplementary Materials

The dataset that was used in the study is publicly available on Kaggle (https://www.kaggle.com/datasets/ananthr1/weather-prediction) for anyone to use. The Python source code notebook used for building the machine learning model is available on the researcher - John Cairo Minerva's GitHub account which was made into a public repository (https://github.com/caiiimnrv/weather-prediction.git).

References

Fente, D.N., & Kumar Singh, D. (2018). Weather Forecasting Using Artificial Neural Network. *Second International Conference on Inventive Communication and Computational Technologies (ICICCT),* 2018, pp. 1757-1761, doi: 10.1109/ICICCT.2018.8473167. https://ieeexplore.ieee.org/document/8473167

Kulkarni, A., Chong, D., & Batarseh, F. A. (2020). Foundations of data imbalance and solutions for a data democracy. Data Democracy, 83–106. https://doi.org/10.1016/b978-0-12-818366-3.00005-8

Lawton, G., Burns, E., & Rosencrance, L. (2022, January 20). logistic regression. Business Analytics. https://www.techtarget.com/searchbusinessanalytics/definition/logistic-regression

Ragan, A. (2018, October 12). Taking the Confusion Out of Confusion Matrices - Towards Data Science. *Medium*. https://towardsdatascience.com/taking-the-confusion-out-of-confusion-matrices-c1ce054b3d3e

Rahul, G. K., Singh, S., & Dubey, S. (2020). Weather Forecasting Using Artificial Neural Networks. ICRITO 2020 - IEEE 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions), 21–26. https://doi.org/10.1109/ICRITO48877.2020.9197993

Saxena, A., Verma, N., & Tripathi, D. K. C. (n.d.). A Review Study of Weather Forecasting Using Artificial Neural Network Approach. www.ijert.org

StudyMoose. (2022, November 1). Predicting the Weather: Forecasting Free Essay Example. https://studymoose.com/predicting-the-weather-forecasting-essay

Tanvir-Hasan, Md., Fattahul-Islam, K.M., Sifat-Rahman, Md., & Song Li (2019). Weather Forecasting Using Artificial Neural Network. *Lecture Notes in Computer Science, 11633. https://link.springer.com/chapter/10.1007/978-3-030-24265-7_15#:~:text=As%20the%20weather%20data%20is,upon%20how%20it%20is%20trained.*

Vashisht, R. (2022, October 10). Machine Learning: When to perform a Feature Scaling? Atoti. https://www.atoti.io/articles/when-to-perform-a-feature-scaling/

Weyn, J. A., Durran, D. R., & Caruana, R. (2020). Improving Data-Driven Global Weather Prediction Using Deep Convolutional Neural Networks on a Cubed Sphere. Journal of Advances in Modeling Earth Systems, 12(9). https://doi.org/10.1029/2020MS002109