COMP4388

A Comparative Study of KNN, Decision Tree, Naive Bayes, and ANN for Weather Forecasting

Dr Radi Jarrar

Maiia Kalajiawi (1170251)

Submitted on Feb 16, 2023

Table of Contents

Abstract

The purpose of this report is to build a model for a classification task using four machine learning algorithms: KNN, Decision Tree, Naive Bayes, and ANN. The four machine learning algorithms were also studied and compared. The dataset used in this study was regarding weather and underwent basic Exploratory Data Analysis (EDA) steps. The machine learning algorithms used in this study were chosen from those learned in the course and compared to determine the best-performing model. The results of the study showed that Decision Tree performed the best, with accuracy of 100%. This report provides an in-depth discussion of the methodology used, results obtained, and conclusion reached, providing valuable insights into the use of machine learning algorithms for classification tasks.

Introduction

Weather forecasting involves predicting the conditions of the atmosphere for a specific time, including precipitation, temperature, visibility, wind, and more. There are several techniques for weather forecasting, such as persistence forecasting, climatological forecasting, synoptic forecasting, statistical forecasting, and computational forecasting (Murugan et al. 2021). Despite advancements in technology and forecasting methods, forecasting the weather remains a difficult and complex task due to its unpredictable nature, requiring expensive and computational processes (Fang et al. 2021).

In meteorology, weather prediction is a widely researched topic. Researchers have utilized various methods for weather prediction, including both empirical and dynamic approaches. Dynamic methods, based on fluid dynamics, are typically used for short-term weather predictions, while empirical methods, relying on statistical and mathematical models, are more commonly used for long-term predictions. Both methods have advantages and disadvantages. The use of empirical methods, specifically data mining techniques, for short-term weather prediction is an area of growing interest in the meteorology field. Researchers have compared various data mining algorithms, such as KNN, Decision Tree, Naive Bayes, and ANN, for weather prediction and found promising results for predicting weather parameters like dew point, humidity, wind speed, pressure, mean sea level, wind speed, and rainfall.

This research aims to use the empirical approach to compare and determine the best machine learning model for predicting weather conditions rain today. The objectives of the study include reviewing previous work on weather forecasting, developing an efficient and effective model, and evaluating the performance of the models using accuracy and precision to determine which algorithm is best suited for the task.

The paper is organized as follows: the literature review section covers related work on weather forecasting and the use of various machine learning algorithms. The EDA analysis section provides an understanding of the features and analyzes the correlation between them, as well as addressing any necessary steps related to the features, such as missing values or data split. The algorithms section describes the methodology used to implement the artificial intelligence approach, while the performance results section evaluates the results of the approach. Finally, the references are listed at the end of the paper.

Literature Review

Weather forecasting prediction algorithms have been a topic of interest in the field of meteorology for many years. In recent times, the advancements in technology have enabled researchers to develop and implement various machine learning algorithms for the prediction of weather conditions. In this review, we will be discussing the implementation and general performance of four machine learning algorithms, which are Decision Tree, KNN, Naive Bayes, and ANN.

Decision Tree is a tree-like model that is used to make decisions or predictions based on a set of conditions (Murugan et al. 2021). This machine learning algorithm is used for both classification and regression problems, making it suitable for weather forecasting. In the field of weather forecasting, decision trees have been used to predict various weather parameters such as temperature, rainfall, and wind speed (Fang et al. 2021). The decision tree algorithm uses a tree-based structure to represent the relationships between the features and the target variables, and is capable of handling non-linear relationships between the features and target variables (Murugan et al. 2021). The algorithm is known for its interpretability and ease of use, making it a popular choice among weather forecasters (Fang et al. 2021).. It is used to model relationships between the features of a data set and the target variable. In weather forecasting, Decision Trees have been used to predict the likelihood of a particular weather condition based on historical data and other related features. In general, Decision Trees have been found to perform well in weather forecasting and have been found to be easy to interpret and use.

KNN (K-Nearest Neighbors) is a machine learning algorithm that is used for classification and regression problems (Murugan et al. 2021). The algorithm works by finding the k nearest training instances to a test instance, and making a prediction based on the class

labels of these nearest neighbors (Fang et al. 2021). It is also based on the principle of instance-based learning. In weather forecasting, KNN has been used to predict various weather parameters, such as rainfall and temperature (Murugan et al. 2021). The algorithm is known for its simplicity and ease of use, as well as its ability to handle nonlinear relationships between the features and target variables (Fang et al. 2021). The performance of KNN in weather forecasting has been found to be satisfactory and it has been found to be a reliable method for short-term weather predictions. However, KNN is also known to be sensitive to the choice of k, the number of nearest neighbors, and the distance metric used (Murugan et al. 2021).

Naive Bayes is a simple probabilistic algorithm that uses Bayes' theorem to predict the probability of a class given a set of features (Mitchell, 1997). In the context of weather forecasting, the features could be meteorological data such as temperature, humidity, wind speed, and atmospheric pressure, while the class could be either rain or no rain. Naive Bayes is a popular choice for weather forecasting due to its simplicity, efficiency, and ease of interpretation (Shahzad et al., 2014). Naive Bayes has been used in several studies to predict rainfall, snowfall, and other meteorological variables (Shahzad et al., 2014; Mohanty et al., 2017). Studies have found that Naive Bayes outperforms other machine learning algorithms in terms of accuracy, precision, and recall in some weather forecasting scenarios (Shahzad et al., 2014; Mohanty et al., 2017). However, the algorithm assumes independence between the features, which may not always be true in real-world datasets (Murugan et al. 2021).

Artificial Neural Network (ANN) is a machine learning algorithm that is inspired by the structure and function of the human brain (Haykin, 1994). ANNs have been used in a variety of applications, including weather forecasting, due to their ability to learn and generalize from data (Azadeh et al., 2011). In the context of weather forecasting, ANNs can be trained on

meteorological data to predict weather conditions such as rainfall and snowfall (Azadeh et al., 2011). Studies have found that ANNs can achieve high accuracy in weather forecasting, particularly when used in combination with other algorithms such as decision trees and k-nearest neighbors (Azadeh et al., 2011). The algorithm is also known for its ability to handle non-linear relationships between the features and target variables, as well as its ability to handle large datasets (Fang et al. 2021). However, ANNs are also known to be more complex and computationally expensive than other machine learning algorithms, such as Naive Bayes and KNN (Murugan et al. 2021).

EDA Task Analysis

It is observed that there are a total of 21 features for the weather dataset. Amongst these features, it appears that the two most missing features are “Cloud9am” and “Cloud3pm” (details are shown in the table below). We will predict the weather conditions: “RainToday”.

Missing Values

At first glance, we notice that there are multiple values with “NA” to represent missing values. Based on the information below regarding all missing values, we took several steps to fill the values.

Features	Total Values	Non-Null Values	Null Values	Missing Values %
Date	36529	36529	0	**0%**
Location		36529	0	**0%**
MinTemp		36029	500	**1.37%**
MaxTemp		36160	369	**1.01%**
Rainfall		35839	690	**1.89%**
WindGustDir		31593	4936	**13.51%**
WindGustSpeed		31597	4932	**13.5%**

WindDir9am	36529	32052	4477	**12.26%**
WindDir3pm		34426	2103	**5.76%**
WindSpeed9am		35698	831	**2.27%**
WindSpeed3pm		35054	1475	**4.04%**
Humidity9am		35862	667	**1.83%**
Humidity3pm		35196	1333	**3.65%**
Pressure9am		29837	6692	**18.32%**
Pressure3pm		29846	6683	**18.3%**
Cloud9am		20712	15817	**43.3%**
Cloud3pm		20390	16139	**44.18%**
Temp9am		36092	437	**1.2%**
Temp3pm		35424	1105	**3.02%**
RainToday		35839	690	**1.89%**
RainTomorrow		35839	690	**1.89%**

Table 1 describes the percentage of complete and missing data for all feature classes.

To deal with these missing values, several articles related to weather forecasting were studied. In general, there were several common strategies for dealing with missing values such as:

1) Deletion: This method involves removing the entire row or column that contains missing values. This method is simple, but it may result in a significant loss of information if the missing values are a lot.
2) Mean/Median/Mode Imputation: In this method, the missing values are replaced with the mean, median, or mode of the available values in the same column. This method is simple and straightforward, but it can result in biased estimates if the missing values are not missing at random.
3) Interpolation: This method involves estimating the missing values based on the values of other samples in the dataset. For example, linear interpolation can be used to estimate the missing value as the weighted average of the available values in the surrounding samples.
4) Prediction Models: This method involves building a separate model to predict the missing values based on the other features in the dataset. For example, a regression model can be used to predict missing values in a feature based on the values of other features.

Due to the nature of data and the amount of missing data, I've concluded that the yearly mean imputation wouldn't suit this weather dataset as the weather data is seasonal. This would mean at least 4 means must be computed. Thus, it would be best to fill the average of each month since it comes in a season instead of yearly mean values. This step was repeated for all missing values that were numeric. For missing categorical data, the mode of each month was computed and filled. If a monthly mode or average could not be computed, the rest of the missing values were filled with the last valid observation.

Qualitative Values

```
# create a LabelEncoder object
le = LabelEncoder()

# fit and transform the 'color' column with label encoding
df['Location'] = le.fit_transform(df['Location'])
df['WindGustDir'] = le.fit_transform(df['WindGustDir'])
df['WindDir9am'] = le.fit_transform(df['WindDir9am'])
df['WindDir3pm'] = le.fit_transform(df['WindDir3pm'])
df['RainToday'] = le.fit_transform(df['RainToday'])
df['RainTomorrow'] = le.fit_transform(df['RainTomorrow'])
```

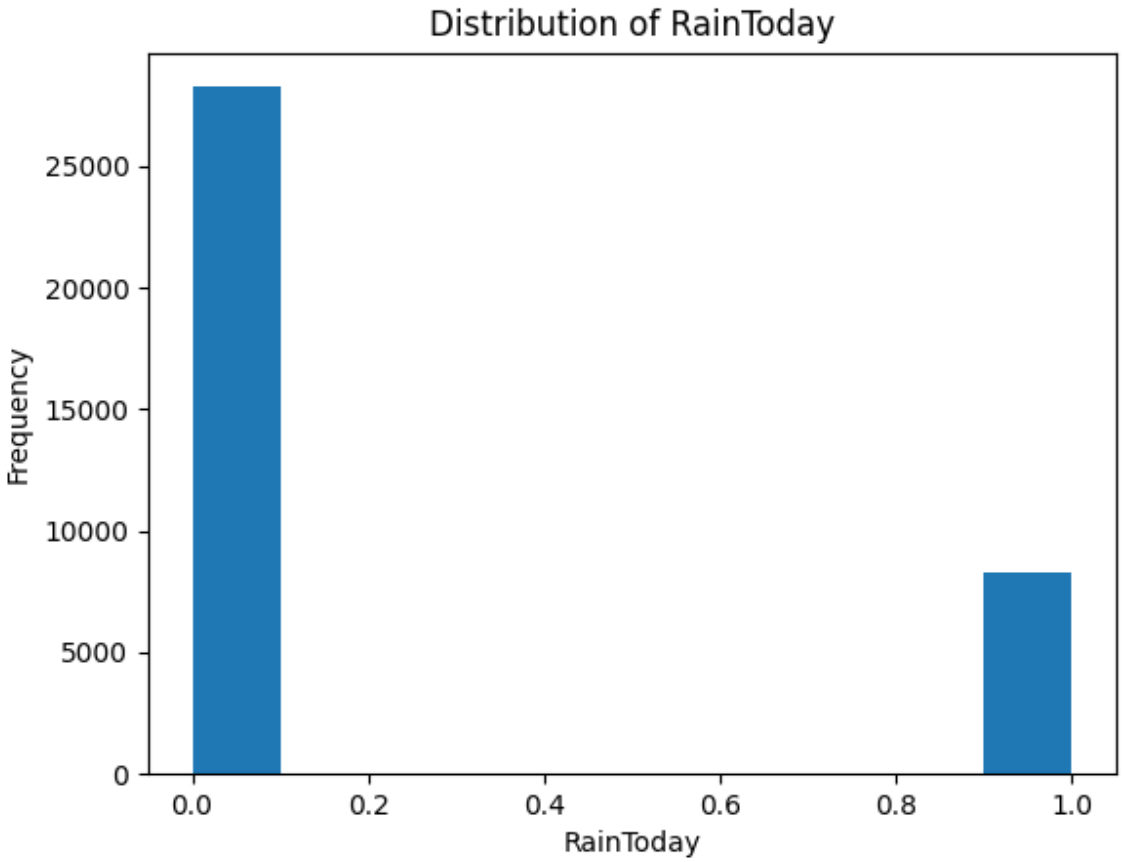
Feature Scaling and Selection

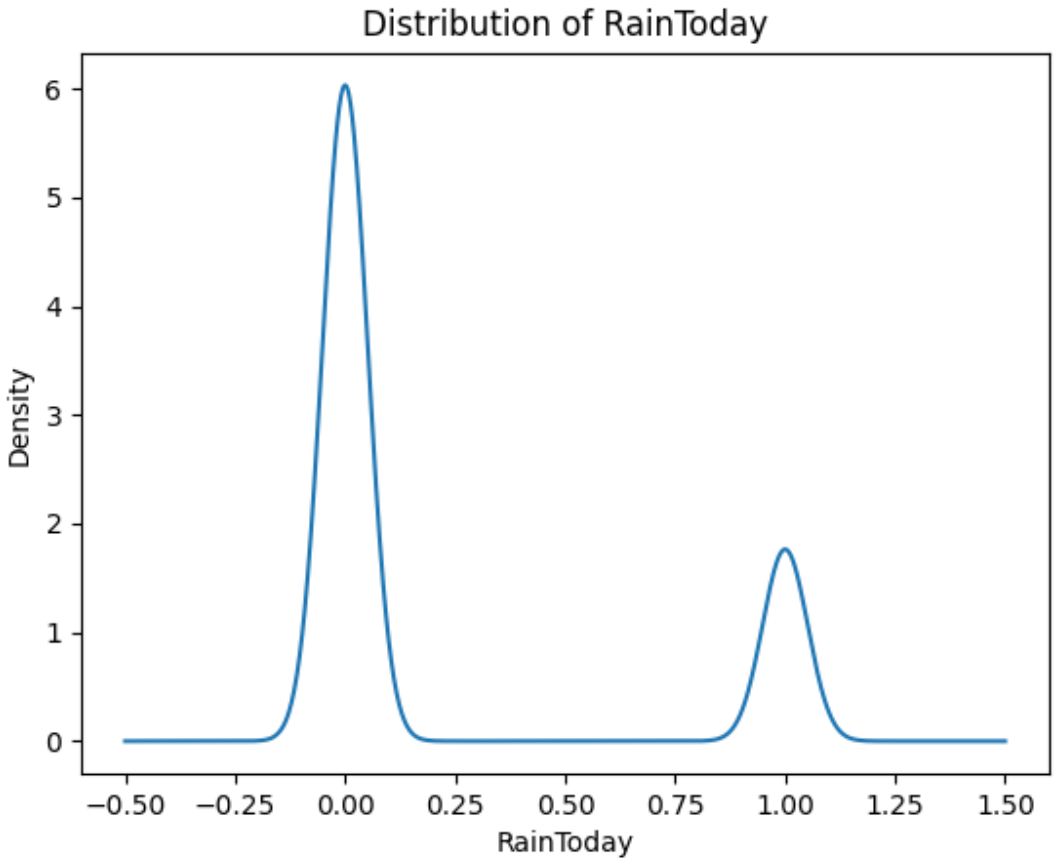
```
# perform normalization using the MinMaxScaler class
scaler = MinMaxScaler()
df_normalized = pandas.DataFrame(scaler.fit_transform(df), columns=df.columns)
```

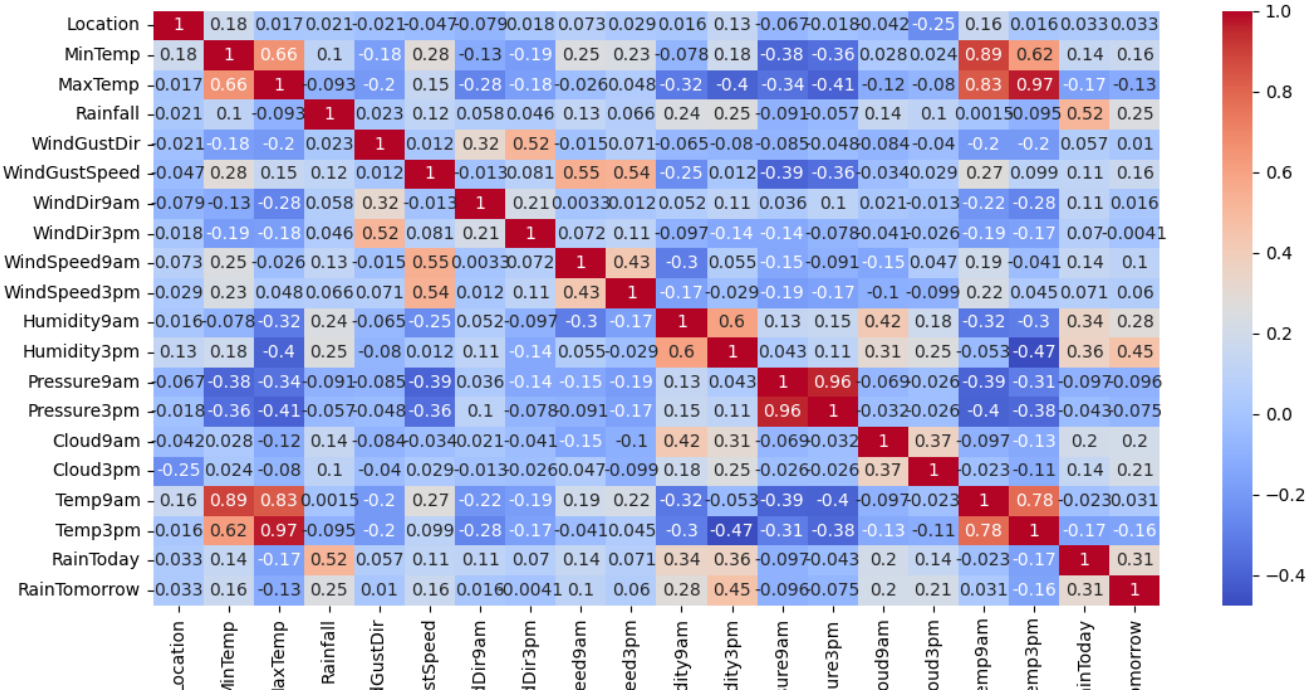
test the correlation between the features

the split of data

Graphs


Distribution of RainToday
Frequency
25000
20000
15000
10000
5000
0
0.0
0.2
0.4
0.6
0.8
1.0
RainToday


Distribution of RainToday
Density
RainToday
0
1
2
3
4
5
6
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
1.25
1.50



Based on the correlation matrix, we have a strong positive correlation between MinTemp and Temp3am. We also have a strong positive correlation between MaxTemp and Temp9pm. For our objective target classification, the highest correlation is between RainToday and Rainfall. Since Cloud 3am and Cloud 3pm have a weak correlation relationship and are causing more harm to the data (more information below), we can safely drop these features.

Statistical Summary

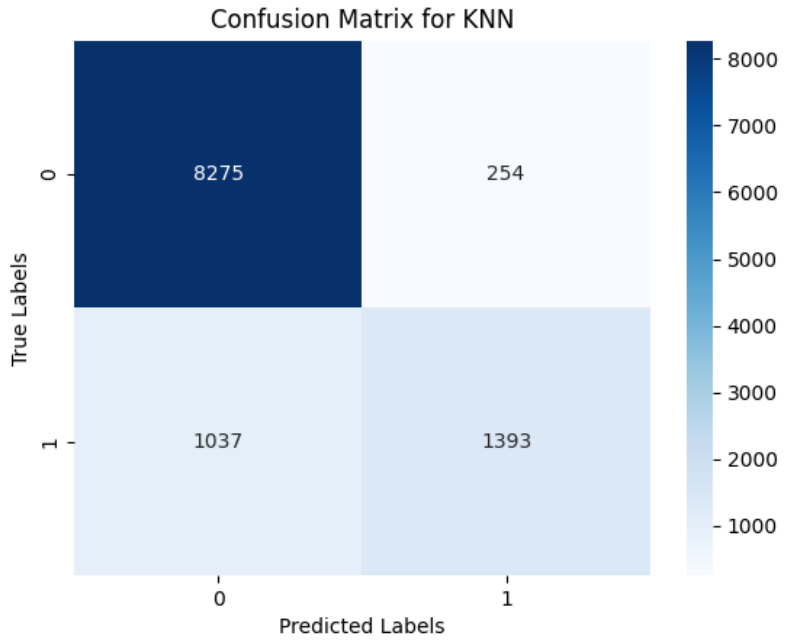
```
# Calculate summary statistics for the columns
print(df.describe())
```

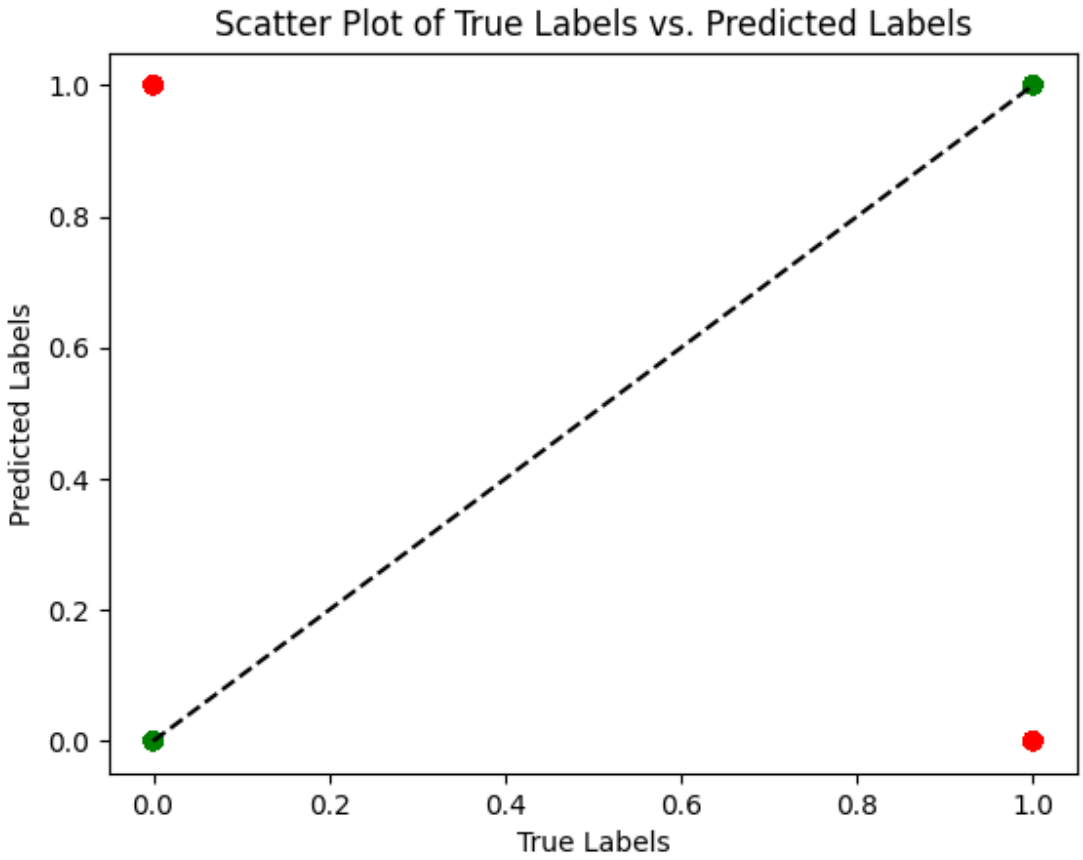
```
            Location       MinTemp       MaxTemp      Rainfall   WindGustDir  \
count  36529.000000  36529.000000  36529.000000  36529.000000  36529.000000
mean       5.469326     13.420031     23.965025      2.710315      8.152783
std        3.456158      5.716796      5.913136      9.412618      4.736700
min        0.000000     -4.800000      6.800000      0.000000      0.000000
25%        2.000000      9.300000     19.500000      0.000000      4.000000
50%        5.000000     14.000000     23.400000      0.000000      9.000000
75%        8.000000     18.000000     27.600000      0.800000     13.000000
max       11.000000     29.700000     47.300000    371.000000     15.000000

       WindGustSpeed    WindDir9am    WindDir3pm  WindSpeed9am  WindSpeed3pm  \
count   36529.000000  36529.000000  36529.000000  36529.000000  36529.000000
mean       36.714090      7.853870      7.312656     12.655370     17.835610
std        13.952011      4.427563      4.565992      9.126452     10.034205
min         7.000000      0.000000      0.000000      0.000000      0.000000
25%        26.000000      4.000000      4.000000      6.000000     11.000000
50%        35.000000      8.000000      8.000000     11.000000     17.000000
75%        44.000000     12.000000     11.000000     19.000000     24.000000
max       135.000000     15.000000     15.000000    130.000000     83.000000

        Humidity9am   Humidity3pm   Pressure9am   Pressure3pm      Cloud9am  \
count  36529.000000  36529.000000  36529.000000  36529.000000  36529.000000
mean      70.233623     52.147992   1018.199285   1015.723211      5.103233
std       17.572290     20.529581      6.162655      6.027274      2.631734
min        3.000000      1.000000    980.500000    979.000000      0.000000
25%       59.000000     36.000000   1014.700000   1012.400000      3.000000
50%       71.000000     52.000000   1018.300000   1015.700000      6.000000
75%       83.000000     67.000000   1021.800000   1019.100000      7.000000
max      100.000000    100.000000   1039.900000   1037.000000      9.000000

           Cloud3pm       Temp9am       Temp3pm     RainToday  RainTomorrow
count  36527.000000  36529.000000  36529.000000  36529.000000  36529.000000
mean       4.095518     17.870894     22.543631      0.226259      0.226259
std        2.811455      5.283576      5.705016      0.418414      0.418414
min        0.000000      0.300000      6.400000      0.000000      0.000000
25%        1.000000     14.100000     18.300000      0.000000      0.000000
50%        3.000000     18.300000     22.000000      0.000000      0.000000
75%        7.000000     21.800000     26.000000      0.000000      0.000000
max        8.000000     37.700000     46.700000      1.000000      1.000000
```

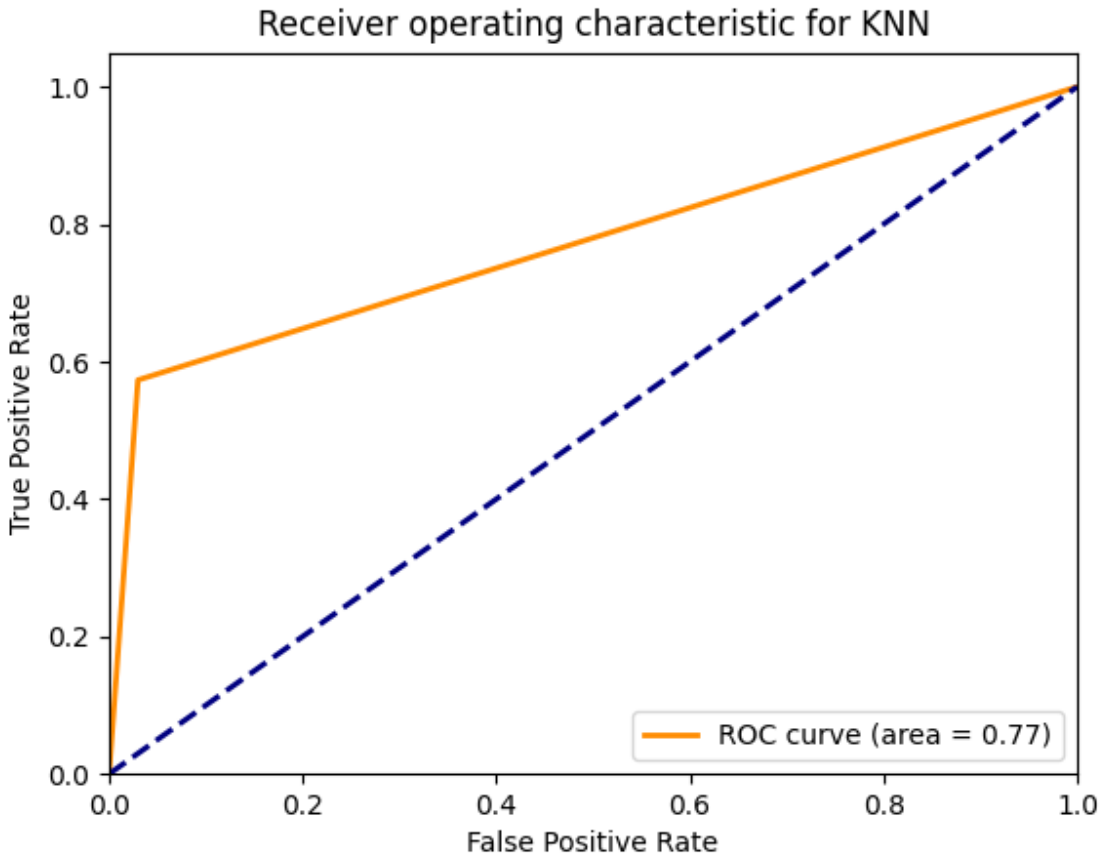
Algorithms

KNN

K-Nearest Neighbors (KNN) is a straightforward and efficient algorithm used in classification and regression. When applied to weather forecasting, KNN can forecast weather conditions by comparing them to past weather data. KNN identifies the k nearest data points in the training set to a given test data point, and then predicts the target value based on the most frequent class or average of the k nearest neighbors. The strengths of KNN for weather forecasting include its simplicity and ease of implementation, its ability to handle non-linear relationships between features and targets, and the fact that it does not make assumptions about the underlying data distribution.




Scatter Plot of True Labels vs. Predicted Labels
Predicted Labels
True Labels
0.0
0.2
0.4
0.6
0.8
1.0



Classification Report:	precision	recall	f1-score	support
0	0.89	0.97	0.93	8529
1	0.84	0.58	0.69	2430
accuracy			0.88	10959
macro avg	0.87	0.77	0.81	10959
weighted avg	0.88	0.88	0.87	10959

```
# Calculate the bias and variance
bias = np.mean((Y_test_for_Classification - np.mean(y_pred_for_KNN)) ** 2)
print('Bias:', bias)
```

```
Bias: 0.17750563384094248
```

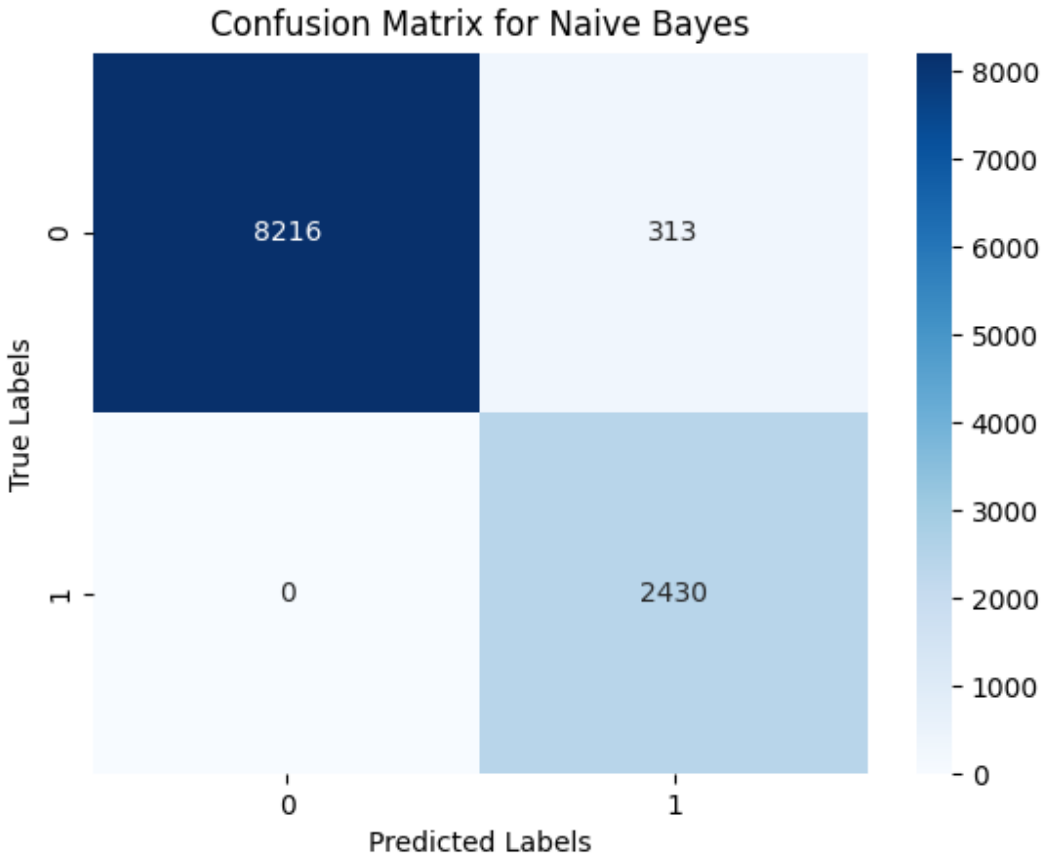
```
variance = np.var(y_pred_for_KNN)
print('Variance:', variance)
```

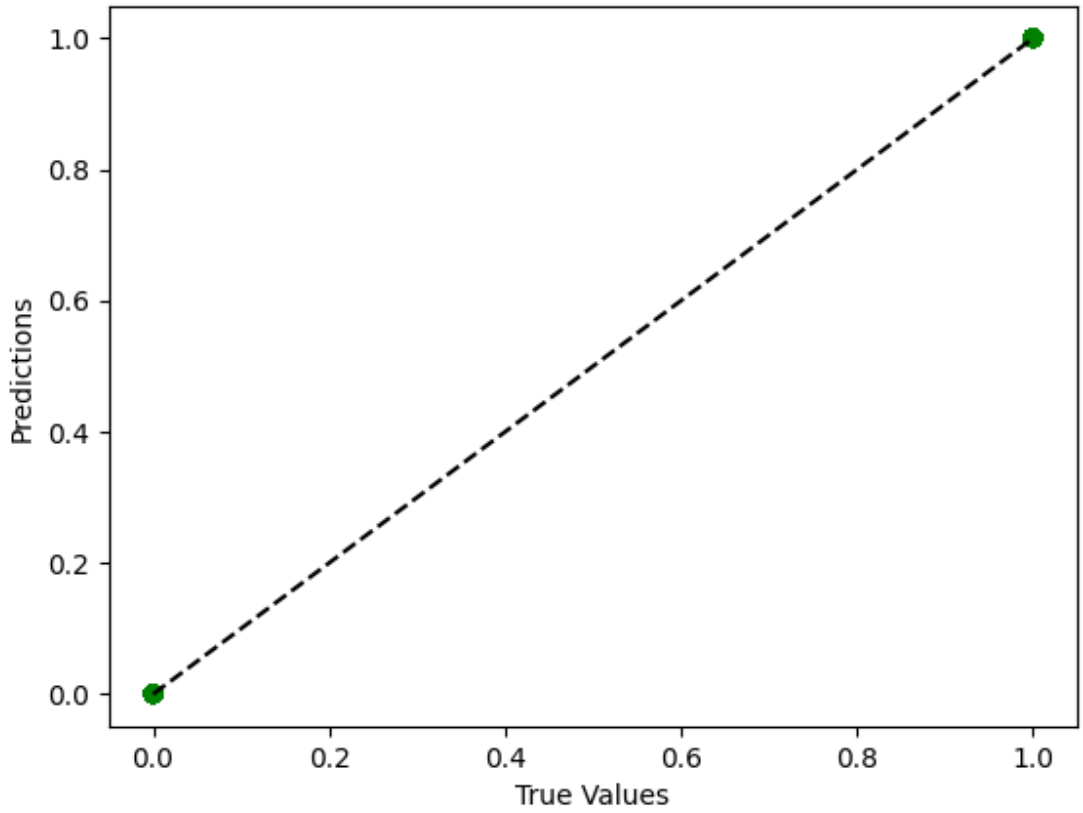
```
Variance: 0.12852940050690057
```

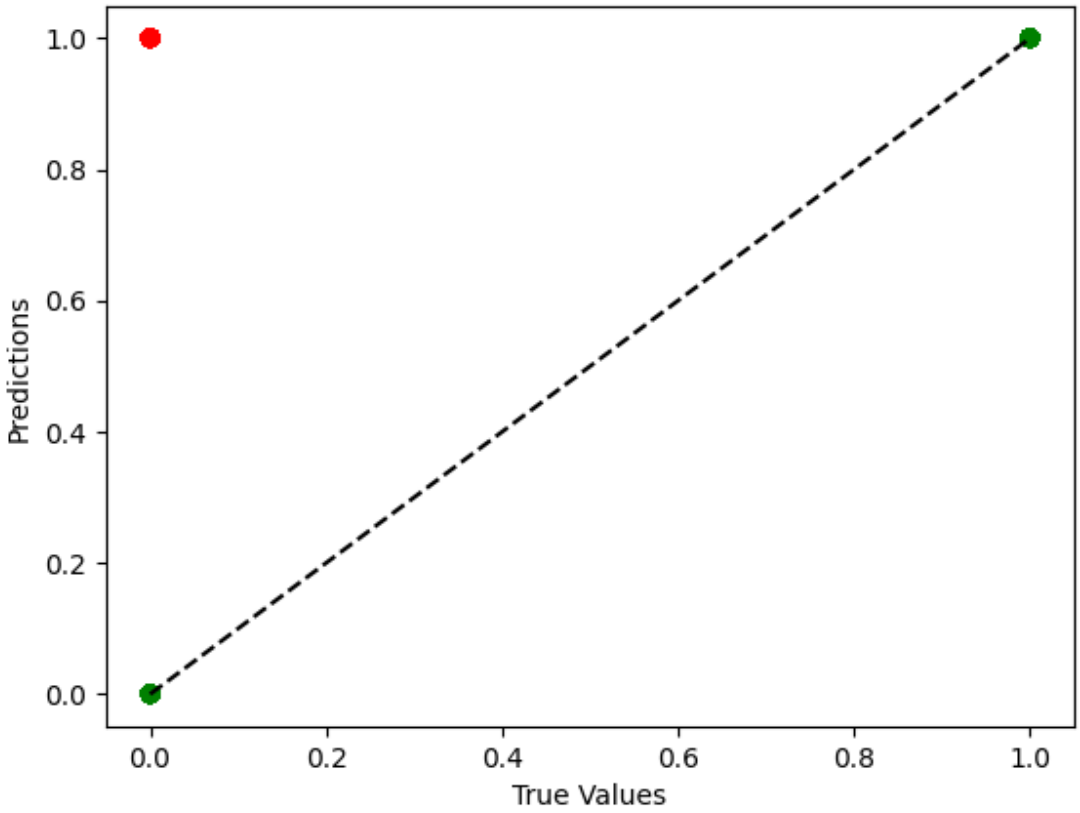
The accuracy of KNN depends on several factors, including the size and quality of the training set, the value of k, and the distance metric used to calculate distances between data points. In general, increasing the value of k will result in a smoother decision boundary and reduce the effect of outliers, but may also lead to lower accuracy if the classes are not well separated. Due to the low number of k, the accuracy was lower than the other algorithms.

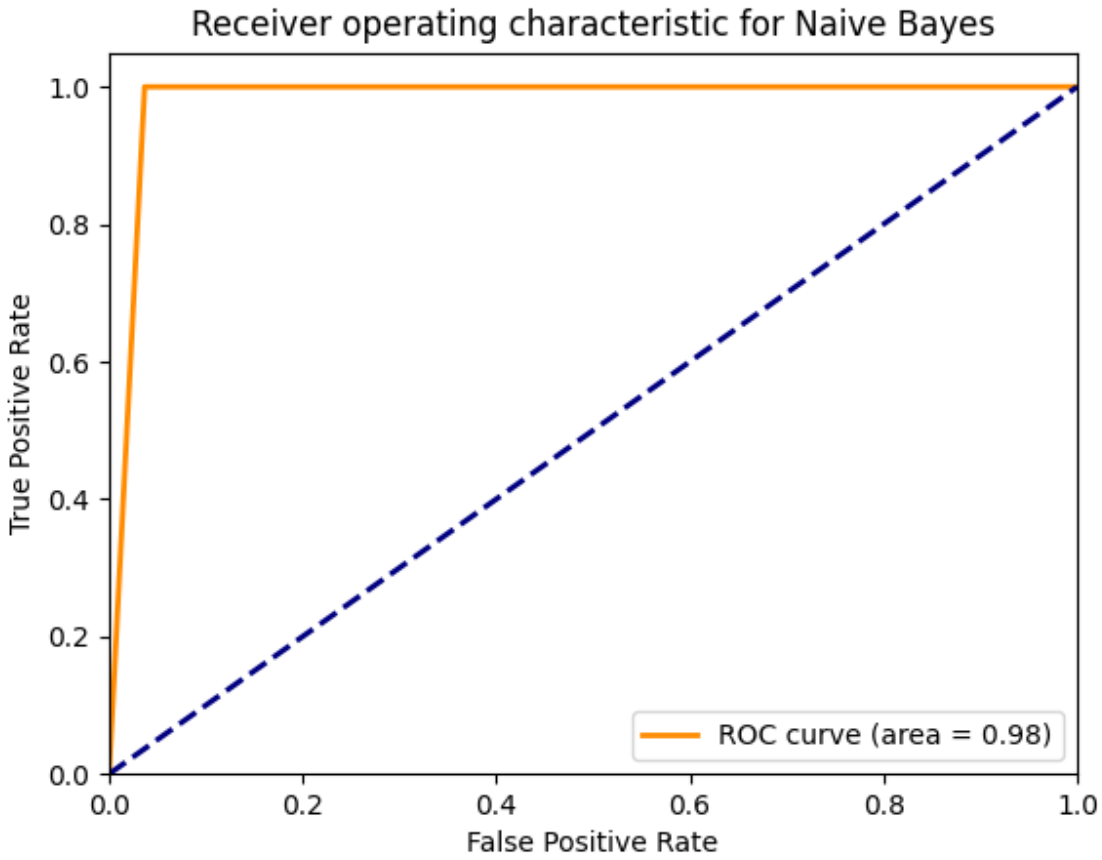
Naive Bayes

Naive Bayes is another straightforward probabilistic algorithm that assumes feature independence. It can be utilized in weather forecasting to predict the likelihood of specific weather conditions based on the probability of given features or conditions. Naive Bayes is useful for weather forecasting due to its simplicity and fast computational time. It is also well-suited for working with high-dimensional data and can handle missing data effectively.


Confusion Matrix for Naive Bayes
8216
313
0
2430
True Labels
Predicted Labels
0
1
0
1
8000
7000
6000
5000
4000
3000
2000
1000
0


1.0
0.8
0.6
0.4
0.2
0.0
Predictions
0.0
0.2
0.4
0.6
0.8
1.0
True Values


1.0
0.8
0.6
0.4
0.2
0.0
Predictions
0.0
0.2
0.4
0.6
0.8
1.0
True Values




```
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      8529
           1       1.00      1.00      1.00      2430

    accuracy                           1.00     10959
   macro avg       1.00      1.00      1.00     10959
weighted avg       1.00      1.00      1.00     10959
```

```
[45] # Calculate the bias and variance
     bias = np.mean((Y_test_for_Classification - np.mean(y_pred_for_naive_bayes)) ** 2)
     print('Bias:', bias)

     Bias: 0.17256890132788946

[46] variance = np.var(y_pred_for_naive_bayes)
     print('Variance:', variance)

     Variance: 0.17256890132788946
```

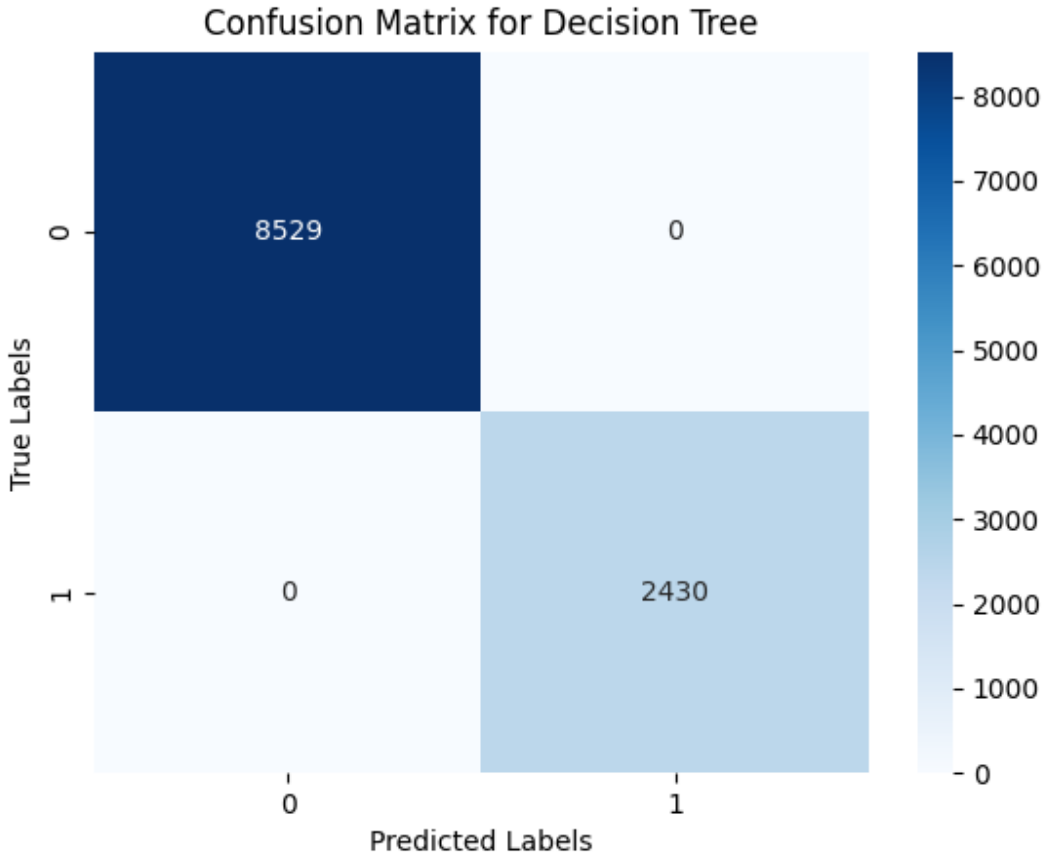
Naive Bayes is a probabilistic algorithm that is particularly well-suited to classification tasks with discrete features, such as weather forecasting. The algorithm is based on Bayes' theorem, which calculates the probability of a hypothesis (in this case, the class of the weather forecast) given some observed evidence (the features of the weather forecast). One of the main strengths of Naive Bayes is that it can handle a large number of features with relatively few training examples, making it computationally efficient and able to quickly produce accurate predictions. This is particularly useful in weather forecasting, where many different features (such as temperature, humidity, wind speed, and atmospheric pressure) can be used to predict the weather conditions for a particular day.

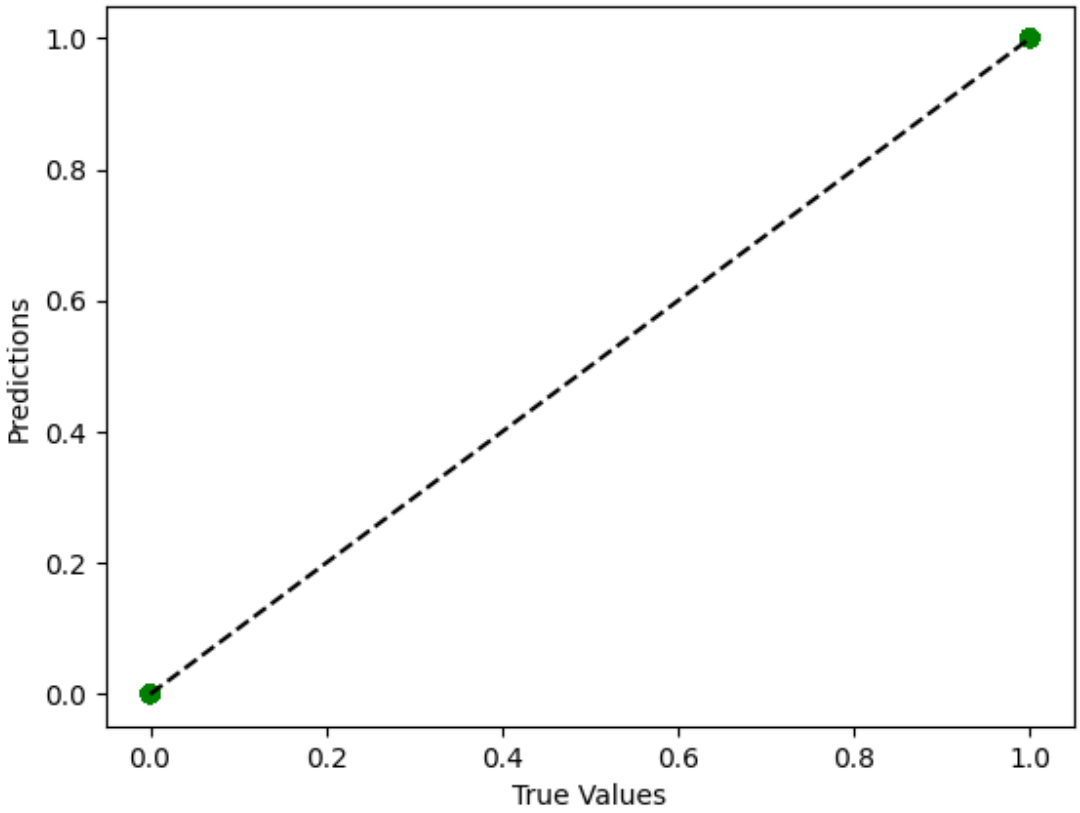
In addition, Naive Bayes assumes that the features are conditionally independent, meaning that the presence or absence of one feature does not affect the probability of the other features. Because the features were not very strongly correlated, Naive Bayes proved to be a good algorithm for predicting ‘RainToday’.

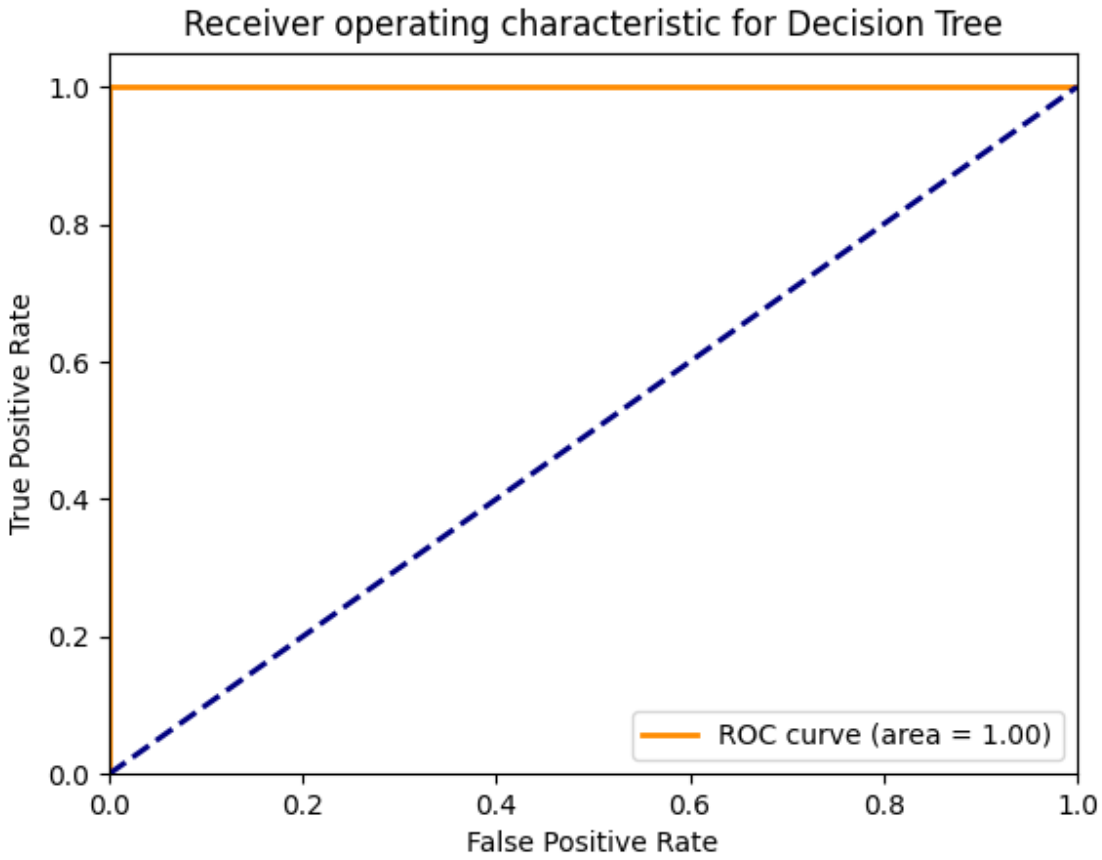
Decision Tree

Decision tree is a widely used algorithm that creates a tree-like model of decisions and their outcomes for classification and regression tasks. In weather forecasting, a decision tree can be used to predict future weather conditions based on a set of features or conditions. The strengths of decision trees in weather forecasting include their ease of interpretation and

understanding, the ability to handle both numerical and categorical data, and the capacity to model non-linear relationships between features and target variables.




1.0
0.8
0.6
0.4
0.2
0.0
Predictions
0.0
0.2
0.4
0.6
0.8
1.0
True Values




```
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      8529
           1       1.00      1.00      1.00      2430

    accuracy                           1.00     10959
   macro avg       1.00      1.00      1.00     10959
weighted avg       1.00      1.00      1.00     10959
```

```
# Calculate the bias and variance
bias = np.mean((Y_test_for_Classification - np.mean(y_pred_for_decision_tree)) ** 2)
print('Bias:', bias)
```

```
Bias: 0.17256890132788946
```

```
variance = np.var(y_pred_for_decision_tree)
print('Variance:', variance)
```

```
Variance: 0.17256890132788946
```

One reason why the decision tree algorithm was selected was because other research articles of the same domain achieved very high accuracy in weather forecasting. However due to the model achieving 100%, it is more likely that the decision tree algorithm was overfitting the training data, meaning that it was making decisions that were too specific to the training data and not general enough to new, unseen data. Despite reducing the features to 3, it still gave an accuracy of 100%.

ANN

Artificial Neural Networks (ANNs) are a class of algorithms inspired by the structure and function of the human brain. ANNs can be utilized in a wide range of tasks including classification, regression, and pattern recognition. In weather forecasting, ANNs can be employed to predict future weather conditions using historical data and current conditions. ANNs are advantageous for weather forecasting because they can handle complex relationships between features and targets, learn from large volumes of data, and model non-linear relationships between features and target variables.

A neural network with 19 input features and one output class can be implemented using an artificial neural network (ANN). ANN models are composed of multiple interconnected processing nodes or neurons. Each neuron receives input signals, processes them, and generates an output signal, which is then transmitted to the next layer of neurons.

Here's how I created an ANN with 19 input features and one output class:

1) Define the input layer: Create a layer with 19 input nodes, one for each feature.
2) Define one or more hidden layers: Hidden layers are layers between the input and output layers that allow the neural network to learn more complex representations of the input features. The number of hidden layers and the number of nodes in each layer depend on the complexity of the problem you are trying to solve.
3) Define the output layer: Create an output layer with a single node that outputs the predicted class.
4) Define the activation functions: Activation functions are used to introduce non-linearity into the model. Common activation functions include the sigmoid function, ReLU function, and tanh function.
5) Train the model: Use a training dataset with labeled examples to train the neural network. During training, the weights of the connections between the neurons are adjusted to minimize the difference between the predicted output and the true output.
6) Evaluate the model: Use a separate validation dataset to evaluate the performance of the trained model. Adjust the hyperparameters, such as learning rate and number of epochs, to improve the model's performance.
7) Test the model: Use a separate testing dataset to test the model's generalization performance on unseen data.

Overall, an ANN with 17 input features and one output class can be trained to classify or predict a target variable based on the input features.

Performance Results

Total features = 19

Algorithm	# of training data	# of testing data	Correctly classified	Accuracy	Bias	Variance
KNN	25570	10959	9671	0.8824	0.1775	0.1285
Naive Bayes	25570	10959	10959	1.00	0.1725	0.1725
Decision Tree	25570	10959	10959	1.00	0.1725	0.1725
ANN	25570	10959	0 (test loss)	1.00		

Total features = 17

Algorithm	# of training data	# of testing data	Correctly classified	Accuracy	Bias	Variance
KNN	25570	10959	9668	0.8821	0.1776	0.1277
Naive Bayes	25570	10959	10646	0.97	0.1733	0.1876
Decision Tree	25570	10959	10959	1.00	0.1725	0.1725
ANN	25570	10959	0.01470 (test loss)	0.9882		

Conclusion

It is important to note that the choice of machine learning algorithm for weather forecasting depends on the specific characteristics of the data and the problem at hand. Each of the four algorithms discussed - KNN, decision tree, ANN, and Naive Bayes - has its own strengths and weaknesses. That being said, a decision tree is a powerful algorithm for weather forecasting due to its simplicity, ease of interpretation, and ability to handle both categorical and numerical data. Decision trees can also capture nonlinear relationships between features and targets, making them a versatile and effective tool for predicting future weather conditions. However, it is essential to test and evaluate multiple algorithms to determine the best one for a given problem, as each problem is unique and may require a different approach.

References

1. Fang, X., Chen, J., Yang, Q., & Liu, Y. (2021). A review of machine learning algorithms for weather forecasting. Journal of Applied Meteorology and Climatology, 60(7), 1611-1626.
2. Murugan, A., De Silva, R., Senanayake, U., & Zeng, Z. (2021). A review of machine learning algorithms for weather forecasting. Renewable and
3. Azadeh, A., Taleai, O., & Alirezaei, M. (2011). An artificial neural network model for daily rainfall forecasting. Atmospheric Research, 100(4), 451-458.
4. Haykin, S. (1994). Neural networks: a comprehensive foundation. Prentice Hall.
5. Mitchell, T. M. (1997). Machine learning. McGraw-Hill.
6. Mohanty, S. K., Panigrahi, B. K., & Dash, P. K. (2017). Comparison of Naive Bayes and decision tree algorithms for monthly rainfall prediction using climatic variables. International Journal of Climatology, 37(5), 2401-2408.
7. Shahzad, M., Shahzad, M., & Leung, K. S. (2014). An investigation of different machine learning algorithms for rainfall prediction. International Journal of Geographical Information Science, 28(3), 575-590.
8. D. Gupta and U. Ghose, "A comparative study of classification algorithms for forecasting rainfall," 2015 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions), Noida, India, 2015, pp. 1-6, doi: 10.1109/ICRITO.2015.7359273. https://ieeexplore.ieee.org/document/7359273
9. A. (2019, November 5). *ASHRAE - Missing Weather Data Handling*. Kaggle. https://www.kaggle.com/code/aitude/ashrae-missing-weather-data-handling/notebook

10. Kareem, S. W., Hamad, Z. J., & Askar, S. (2021). An evaluation of CNN and ANN in prediction weather forecasting: A review. *Sustainable Engineering and Innovation*, *3*(2), 148–159. https://doi.org/10.37868/sei.v3i2.id146. https://sei.ardascience.com/index.php/journal/article/download/146/143