Airline Delay Analysis

Group Members: Himanshu Gandhi (1770138), Venkata Shreya Kala (1764875), Shubham Prasad Sahoo (1824661)

1 Abstract

2 Introduction

3 Data

The dataset comprises Integer, Categorical, and Binary data types, reflecting various essential information crucial for... (). Figure **??** is a brief description of some of the important features of the dataset.

4 Methodology

4.1 Data Pre-processing and Analysis

Data pre-processing...

4.1.1 Null Value Handling

4.1.2 Outlier Treatment

4.1.3 Feature Engineering

4.1.4 Feature Selection

Correlation Heatmap:

Multicollinearity:

Redundant Feature Removal: .

4.2 Modeling Techniques

Logistic Regression

Logistic Regression is a classic classification algorithm, that predicts binary outcomes by modelling the relationship between features and a binary target variable. It utilizes the logistic function to compute the log odds of the event happening and then applies it to obtain the predicted probability. The logistic function transforms the input features into a probability score for the positive class.

The logistic regression formula represents the probability that the target variable of a flight being delayed or not equals 1 given the independent variables or features that could impact delay, where each β represents the coefficient for a feature:

Logistic or Sigmoid Function:

$$p = \frac{1}{1 + e^{-\text{logit}(p)}}$$

Logit Function for the model:

$$\text{logit}(p) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n$$

Logistic regression learns the relationship between the input features (such as flight departure time, airline, weather conditions, etc.) and the target variable- whether a flight is delayed. It learns each feature's coefficients (or beta values), indicating their importance in predicting flight delays. The logistic function is then applied to compute the chance of a flight getting delayed based on these coefficients. To implement logistic regression, we utilize the LogisticRegression function from sklearn.linear_model. The parameter used is C: The inverse of regularization strength, where smaller values indicate stronger regularization.

Support Vector Machine (SVM)
SVM constructs a hyperplane to separate classes by maximizing the margin between them, ensuring a clear class boundary while minimizing errors. The closest points to the hyperplane are called the support vectors. These determine the position of the separating hyperplane. Input data is mapped to a high-dimensional feature space to find the optimal hyperplane that separates the classes. Kernel methods are often employed to map the data to higher dimensions for an optimal hyperplane which is one of the parameters passed to the SVM Classifier (SVC) model from sklearn.svm. Other parameters include- - C: Regularization parameter and gamma: Kernel coefficient for the 'rbf' kernel. Higher gamma values could lead to overfitting as they allow for complicated decision boundaries.

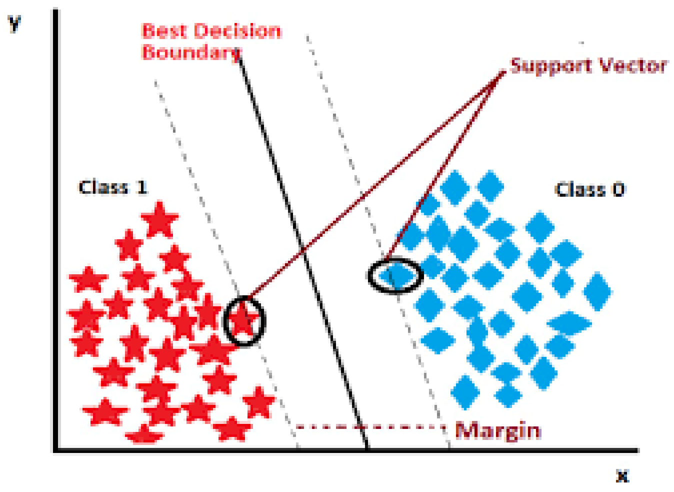

Figure 1: SVM (Https://statusneo.com/defying-convention-svm-the-maverick-of-ml-algorithms/)()

The working principle of SVM is illustrated in Figure 1().

SVM recognizes delayed flights by creating a decision boundary that separates flights into two categories: delayed and not delayed by analyzing various features to find patterns that could indicate delays.

Random Forest
Random Forest is an ensemble learning technique that combines multiple decision trees to make predictions. Each tree is trained on a random subset of the data, and then their predictions are combined to determine the most frequently predicted class. Each tree in the forest is trained and tested on a random subset of the data and features, which helps to reduce overfitting and improve generalization 2. So, it recognizes delayed flights by analyzing various features and their interactions. It has been implemented using the RandomForestClassifier function from sklearn.ensemble. The parameters used are the number of estimators- n_estimators: Number of trees in the forest. Another parameter is- max_depth: Maximum depth of each tree in the forest. It is essential to balance model complexity and performance by controlling this value to prevent overfitting.

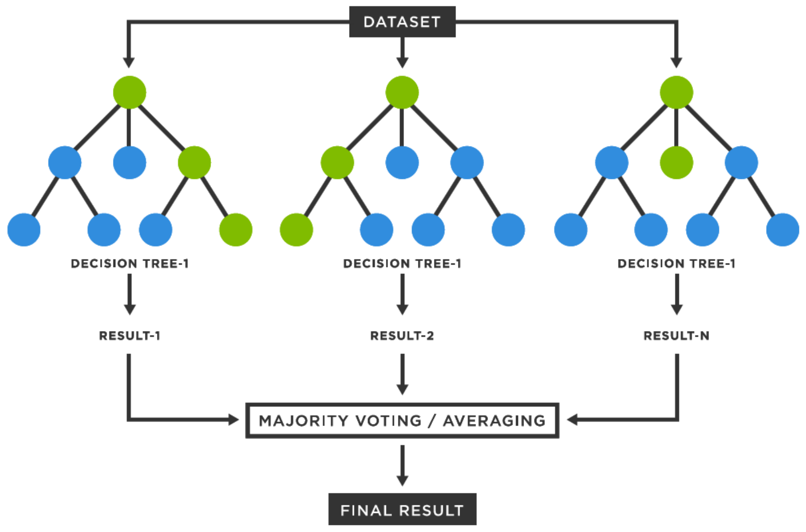

Figure 2: (https://medium.com/@denizgunay/random-forest-af5bde5d7e1e)()

K-Nearest Neighbours (KNN)
K-Nearest Neighbors (KNN) is a simple algorithm that emphasizes local patterns in the data. It classifies instances based on the majority class among their 'k' number of nearest neighbours in the feature space. The distance between a new observation and the training instances is calculated to determine the 'k' nearest neighbours. The label observed most frequently among these neighbours is then assigned to the new observation. It does not assume the underlying data distribution, making it a non-parametric method. It has been imported from sklearn.neighbors as KNeighborsClassifier and the number of neighbors to consider for classification, denoted as 'k' denoted by n_neighbors was passed as a parameter. Higher values of 'k' lead to smoother decision boundaries but may lead to an oversimplifed model.

This model identifies similar instances of delayed flights based on their features and assigns them the same label as their nearest neighbours in the feature space similar to the working given in Figure 3.

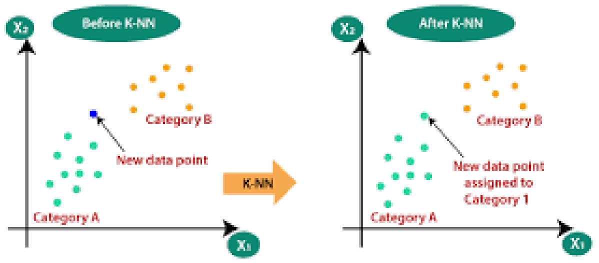

Figure 3: KNN (https://medium.com/@shankyp1000/knn-classifier-from-scratch-326d3d4e894e)()

XGBoost
XGBoost is an ensemble method that leverages boosting to combine weak learners, typically decision trees, to improve prediction accuracy or predictive power. It employs gradient boosting to construct sequential, interpretable trees by iteratively correcting errors and updating residuals. The loss function and regularization term are optimized to enhance model performance. XGBoost's functionality is given in the XGBClassifier from the xgboost library. Three parameters were passed to the model- n_estimators: Number of trees to build, max_depth: Maximum depth of each tree, learning_rate: Step size at each iteration, influencing the speed of learning and convergence.

The final prediction in XGBoost is obtained by summing up the predictions from all individual trees:

$$\hat{y} = \sum_{k=1}^{K} f_k(x_i)$$

This model works by iteratively improving predictions through sequential trees, enabling the identification of delayed flight instances based on various features in the dataset.

5 Results and Inferences

5.0.1 Metrics

We generated classification reports and plotted AUC-ROC curves for each of the classification models. Additionally, learning curves were plotted to demonstrate the model's performance as the training size increased.

Classification report: The classification report presents an overview of a classification model's performance, summarizing key metrics including precision (the accuracy of positive predictions), recall (the capability to identify positive instances), F1 score (a balanced measure of precision and recall), and accuracy.

Precision $= \frac{\text{True Positives}}{\text{True Positives+False Positives}}$

Recall $= \frac{\text{True Positives}}{\text{True Positives+False Negatives}}$

The F1 score is the harmonic mean of precision and recall:

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

The models were termed the best depending on the highest accuracy achieved, followed by a better precision, recall, and F1 score and then by considering the lesser difference in the training and testing accuracy if the former is higher.

Receiver Operating Characteristics (ROC): A graphical representation of the trade-off between True Positive Rate and False Positive Rate (analv)(). A high true positive rate and a low false positive rate indicate a well-performing model. The true Positive Rate is given by the proportion of correctly identified actual positives while the false positive rate indicates the proportion of actual negatives that are falsely identified as positives (analv)().

Area Under the Curve (AUC): Area under the ROC curve. An AUC score close to 1 indicates a good classifier, while a score close to 0 indicates a good classifier in reverse, and a score close to 0.5 suggests a poor classifier (or a random guess).

$$\text{AUC} = \frac{\sum_{x \in AN} \sum_{y \in AP} 1_{f(y) > f(x)}}{|AP| \times |AN|}$$

Learning Curve Learning curves provide valuable insights into the training of a model by plotting the change in a performance metric over time or with the number of steps. These curves are representations of the learning process, with the x-axis representing time or progress, and the y-axis representing the metric. These curves help in detecting issues, and optimizing prediction performance.(https://www.baeldung.com/cs/learning-curve-ml)().

6 Critique of the Model

7 Conclusion

8 References and Appendices

References

8.1 Appendices