Queen Mary University of London
School of Engineering and Materials Science

PREDICTING FLIGHT DELAYS

By Kristina Anne Baroma
Student Number: 200019273

DEN318 – Third-Year Project
Supervisor: Dr Jun Chen

SCHOOL OF ENGINEERING AND MATERIALS SCIENCE

ENGINEERING / MATERIALS
THIRD YEAR PROJECT
DEN318

APRIL 2022

DECLARATION

This report entitled

PREDICTING FLIGHT DELAYS

Was composed by me and is based on my own work. Where the work of others has been used, it is fully acknowledged in the text and in captions to table illustrations. This report has not been submitted for any other qualification.

Name Kristina Anne Baroma

Signed 

Date 25/04/2023

Abstract

Predicting flight delays is a critical challenge for the aviation industry, as delays can have significant financial and operational consequences for airlines and passengers. This study aims to predict flight delays using machine learning techniques, specifically the Random Forest algorithm. We employed the "flights.csv" dataset, which contains US flight data from 2015, and pre-processed the data to include relevant features and create a binary target variable. The impact of sample size on model performance and the effects of hyperparameter tuning were explored. Our findings indicate that increasing the sample size improves model performance, and hyperparameter tuning can help balance the trade-off between bias and variance. This study contributes to the field by demonstrating the potential of machine learning models to predict flight delays, thereby helping airlines and airports make informed decisions and improve operational efficiency.

Table of Contents

List of Tables

1 Introduction

Air travel is an essential mode of transportation for millions of people worldwide, and the aviation industry has grown tremendously over the years. However, one of the most significant issues faced by the industry is flight delays, which can cause inconvenience to passengers, disrupt travel plans, and result in significant economic losses. According to the United States Department of Transportation, flight delays in 2019 resulted in a cost of almost $32.9 billion to the US economy (U.S. Department of Transportation, 2020). In addition to economic losses, flight delays also have social and environmental implications, such as increased stress levels for passengers and increased carbon emissions.

Predicting flight delays is, therefore, a crucial task in the aviation industry, and accurate prediction can help airlines and airports better manage their operations, improve customer satisfaction, and minimise economic losses. Over the years, researchers have developed various models and algorithms to predict flight delays using machine learning techniques, such as gradient boosting, random forest, and support vector machines (Gopalakrishnan & Balakrishnan, 2017).

In this report, we aim to predict flight delays using a random forest algorithm and principal component analysis (PCA) for dimensionality reduction. We will use a dataset from Kaggle that contains information on flights departing from US airports in 2015. We will pre-process the data by encoding categorical variables, handling missing values, and creating a binary target variable for flight delays. We will then split the data into training and testing sets, scale the data, and apply PCA to reduce the number of features. Finally, we will train a random forest model on the reduced feature set and evaluate its performance using various performance metrics.

The rest of this report is structured as follows. In the subsequent section, we will review existing research and methods used in flight delay prediction, identify any gaps in existing research, and discuss how our study aims to address them. In the next section, we will describe the dataset used in our study and explain the pre-processing steps taken. We will then describe the methodology used, including the random forest algorithm and PCA for dimensionality reduction. In the experiments and results section, we will investigate the impact of sample size on model performance, perform hyperparameter tuning to optimise the model, and evaluate its performance using different performance metrics. In the discussion section, we will discuss the findings of our experiments, address any limitations of our study, and provide suggestions for future work. Finally, we will conclude the report by summarizing our main findings and restating the importance of the problem and the contribution of our research to the field.

2 Literature review

2.1 Overview of Flight Delays

Causes of flight delays

Flight delays have been a major concern in the aviation industry due to their significant impact on passengers, airlines, and the overall air transportation system. According to the

United States Bureau of Transportation Statistics (BTS), flight delays can be broadly categorised into five main causes: (i) National Aviation System (NAS) delays, (ii) weather-related delays, (iii) aircraft maintenance and technical issues, (iv) crew scheduling problems, and (v) security issues (U.S. Department of Transportation, 2020).

NAS delays refer to the disruptions in air traffic due to the limitations of the air traffic control system, heavy traffic volume, airport operations, and other non-extreme weather conditions (AhmadBeygi, et al., 2008). Weather-related delays, which account for a significant portion of flight delays, are caused by severe weather conditions such as thunderstorms, snowstorms, fog, and strong winds (Ball, et al., 2010). Aircraft maintenance and technical issues arise from mechanical problems or malfunctions in aircraft systems, leading to delays as airlines prioritise safety (Mazzeo, 2003). Crew scheduling problems may be caused by crew unavailability or exceeding the maximum allowable working hours set by regulations, leading to delays in flight operations (Kohl, et al., 2007)Security issues, although relatively rare, contribute to flight delays when additional screening or precautionary measures are needed (Barnhart, et al., 2003).

2.2 Machine Learning Techniques in Predicting Flight Delays

Machine learning techniques have been increasingly applied to predict flight delays due to their ability to learn complex patterns from large datasets, accommodate nonlinear relationships between variables, and provide valuable insights for decision-making (Hess, 2010). This section will provide an overview of the main machine learning techniques used in flight delay prediction, including supervised and unsupervised learning algorithms.

Supervised Learning Algorithms

Supervised learning algorithms are widely used in flight delay prediction tasks, as they rely on labelled datasets to predict the target variable, which, in this case, is the flight delay. Some of the most used supervised learning algorithms in flight delay prediction include:

- Linear regression (LR): A simple and interpretable algorithm used to model the relationship between one or more independent variables (features) and the dependent variable (flight delay) (Ding, 2017)
- Decision trees (DT): A tree-based algorithm that recursively splits the dataset into subsets based on the values of the input features, with each split representing a decision rule (Manna, et al., 2017)
- Random forests (RF): An ensemble learning method that constructs multiple decision trees and aggregates their predictions to improve overall accuracy and reduce overfitting (Gui, et al., 2019)
- Support vector machines (SVM): A kernel-based algorithm that seeks to find the optimal hyperplane that maximises the margin between the two classes (flight delays and non-delays) (Chen, et al., 2008)
- Artificial neural networks (ANN): A computational model inspired by the human brain, consisting of interconnected neurons organised in layers, capable of learning complex, nonlinear relationships between input and output variables

(Khanmohammadi, et al., 2016)

Unsupervised Learning Algorithms

Unsupervised learning algorithms, which do not rely on labelled datasets, have also been employed in flight delay prediction, primarily for data pre-processing and feature extraction. Some examples of unsupervised learning algorithms used in this context are:

- Clustering: Algorithms like K-means and DBSCAN are used to group flights with similar characteristics, such as departure time or airport congestion, to identify patterns that might contribute to flight delays (Güvercin, et al., 2020)
- Dimensionality reduction: Techniques like Principal Component Analysis (PCA) and t-Distributed Stochastic Neighbour Embedding (t-SNE) are employed to reduce the dimensionality of the dataset, allowing for better visualisation and understanding of the underlying relationships between features (Wu, et al., 2019)

In summary, various machine learning techniques have been applied to predict flight delays, with supervised learning algorithms being the most commonly used due to their predictive capabilities. Unsupervised learning algorithms, on the other hand, play a crucial role in data pre-processing and feature extraction, enabling a better understanding of the underlying patterns in flight delay data.

2.3 Previous Research and Findings in Flight Delay Prediction

A growing body of research has focused on developing machine learning models to predict flight delays. This section highlights the key aspects of these studies, including dataset selection and pre-processing, as well as model evaluation and comparison.

Dataset Selection and Pre-processing

Selecting an appropriate dataset is crucial for developing accurate and generalizable flight delay prediction models. Previous studies have utilised datasets from various sources, such as the United States Bureau of Transportation Statistics (BTS), the Federal Aviation Administration (FAA), and EUROCONTROL (Rebollo & Balakrishnan, 2014). These datasets typically contain information on flight schedules, delays, weather conditions, and airport operations.

Data pre-processing is an essential step in developing machine learning models, as it helps address issues related to missing values, outliers, and inconsistent data formats. Techniques such as imputation, normalization, and feature encoding have been applied to ensure the quality and consistency of the input data (Gopalakrishnan & Balakrishnan, 2017)

Model Evaluation and Comparison

Several studies have compared the performance of different machine learning algorithms in predicting flight delays. For instance, one study compared the performance of decision trees, support vector machines and random forests, finding that random forests outperformed the other models (Choi, et al., 2016). Similarly, Gopalakrishnan and Balakrishnan (2017) compared linear regression, artificial neural networks, and support vector machines,

concluding that support vector machines provided the best results.

To evaluate the performance of flight delay prediction models, various metrics have been employed, including mean absolute error (MAE), mean squared error (MSE), root mean squared error (RMSE), and accuracy (Kodinariya & Makwana, 2013)In addition, studies have used cross-validation techniques to assess the generalizability of their models and to prevent overfitting (Gopalakrishnan & Balakrishnan, 2017).

In conclusion, previous research on flight delay prediction using machine learning has focused on selecting appropriate datasets, performing data pre-processing, and evaluating the performance of various algorithms. These studies have provided valuable insights into the factors affecting flight delays and have demonstrated the potential of machine learning techniques in improving the prediction accuracy and operational efficiency of the aviation industry.

2.4 Challenges and Future Directions in Flight Delay Prediction

While machine learning techniques have shown promising results in predicting flight delays, several challenges and opportunities remain to be addressed in future research. This section discusses the main challenges in flight delay prediction, along with potential future directions.

Data Quality and Availability

Data quality and availability are critical factors that influence the performance of machine learning models. Inaccurate, incomplete, or inconsistent data can lead to biased and unreliable predictions (Misra & Yadav, 2019). To mitigate these issues, future research could focus on collecting and integrating data from multiple sources, such as airport and airline databases, air traffic control systems, and social media platforms (Truong, 2021). Additionally, researchers could explore novel techniques for handling missing or noisy data, such as data imputation and data fusion methods (Shao, et al., 2022)

Feature Selection and Engineering

Identifying the most relevant features for flight delay prediction is essential for improving model accuracy and interpretability. Many studies have employed domain knowledge and expert opinions to select features, such as weather conditions, airport congestion, and flight schedules (Rebollo & Balakrishnan, 2014). However, there is still potential for discovering new features and relationships that can enhance model performance. Future research could explore advanced feature selection and engineering techniques, such as genetic algorithms, deep learning, and unsupervised feature learning (Li, et al., 2022).

Real-Time Prediction and Decision-Making

Many existing flight delay prediction models focus on forecasting delays several hours or even days in advance. However, accurate real-time predictions are crucial for enabling proactive decision-making and minimizing the impact of delays on passengers and airline operations (Hess, 2010). Future research could explore the development of real-time flight delay prediction models, incorporating dynamic and streaming data sources such as live weather updates, social media feeds, and air traffic control data (Truong, 2021).

Model Interpretability and Explainability

Interpretability and explainability are critical aspects of machine learning models, particularly in safety-critical domains such as aviation. While complex models, such as deep learning and ensemble methods, often provide higher accuracy, they can be challenging to interpret and explain (Shao, et al., 2022). Future research could focus on developing interpretable and explainable machine learning models for flight delay prediction, allowing stakeholders to understand the underlying factors contributing to delays and make more informed decisions (Li, et al., 2022).

In conclusion, although significant progress has been made in predicting flight delays using machine learning techniques, several challenges and opportunities remain to be explored. By addressing these issues, future research can contribute to the development of more accurate, reliable, and interpretable flight delay prediction models, ultimately enhancing the efficiency and sustainability of the aviation industry.

3 Data Description and Pre-processing

3.1 Dataset Overview

The dataset used in this study is the "flights.csv" file, which was obtained from Kaggle (https://www.kaggle.com/code/fabiendaniel/predicting-flight-delays-tutorial/input?select=flights.csv). This dataset contains US flight data from 2015, including various features related to flight schedules, airlines, airports, and delays. The primary objective of this study is to predict flight delays using machine learning techniques.

3.2 Features and Target Variable

The dataset initially includes numerous features, such as YEAR, FLIGHT_NUMBER, TAIL_NUMBER, TAXI_OUT, WHEELS_OFF, AIR_TIME, WHEELS_ON, TAXI_IN, CANCELLED, CANCELLATION_REASON, DIVERTED, AIR_SYSTEM_DELAY, SECURITY_DELAY, AIRLINE_DELAY, LATE_AIRCRAFT_DELAY, and WEATHER_DELAY. However, several of these features are irrelevant to predicting flight delays and were dropped from the dataset. The remaining features were used to create a binary target variable, "DELAYED," which indicates whether a flight is delayed (1) or on-time (0).

3.3 Pre-processing Steps

The pre-processing steps performed on the dataset include the following:

- Handling missing values: Rows with missing values for the 'DEPARTURE_DELAY' feature were dropped. Remaining missing values were filled with the mean value of each respective column.
- Encoding categorical variables: Categorical features, such as MONTH, DAY, DAY_OF_WEEK, AIRLINE, ORIGIN_AIRPORT, and DESTINATION_AIRPORT, were encoded using one-hot encoding.
- Feature selection: Irrelevant columns were dropped, and the remaining features were

used as input for the machine learning model. The PCA algorithm was also applied to reduce dimensionality while retaining 95% of the variance in the data.

3.4 Data Splitting and Balancing

The dataset was split into training and testing sets, with an 80-20 ratio, using a random seed to ensure reproducibility. Data scaling was performed using the StandardScaler method from the scikit-learn library to standardise the features before applying PCA and training the model. No specific data balancing techniques were applied in this study, as the RandomForestClassifier can inherently handle imbalanced datasets through its algorithm.

4 Methodology

4.1 Machine Learning Algorithm: Random Forest

The Random Forest algorithm was chosen for this study due to its robustness and ability to handle a large number of features, as well as its capability to perform both classification and regression tasks. Random Forest is an ensemble learning method that constructs a multitude of decision trees at training time and outputs the class that is the mode of the classes or the mean prediction of the individual trees.

The key hyperparameters of the Random Forest algorithm include:

- n_estimators: The number of trees in the forest. Increasing the number of trees can improve the model's performance but may lead to longer training times.
- max_depth: The maximum depth of each tree in the forest. This parameter controls the complexity of the trees and can help prevent overfitting.
- min_samples_split: The minimum number of samples required to split an internal node. This parameter can also help prevent overfitting by ensuring that the trees do not grow too deep.
- min_samples_leaf: The minimum number of samples required to be at a leaf node. This parameter can help control the size of the trees and improve generalization.

4.2 Dimensionality Reduction using PCA

Principal Component Analysis (PCA) was employed for dimensionality reduction in this study. The rationale behind using PCA is to reduce the complexity of the dataset by transforming the original set of features into a smaller set of uncorrelated variables, known as principal components. This process helps to minimise the impact of the curse of dimensionality, which can degrade the performance of machine learning algorithms when dealing with high-dimensional data.

By using PCA, we can capture most of the information contained in the original features while reducing the number of dimensions, which can lead to faster training times and improved model performance.

4.3 Model Evaluation and Performance Metrics

To evaluate the performance of the Random Forest model, we used the following performance metrics:

- Accuracy: The proportion of correctly classified instances out of the total instances in the dataset. Accuracy is a widely used metric for classification problems, but it can be misleading when dealing with imbalanced data.
- Precision: The proportion of true positive instances out of the total predicted positive instances. Precision is useful when the cost of false positives is high.
- Recall: The proportion of true positive instances out of the total actual positive instances. Recall is important when the cost of false negatives is high.
- F1-score: The harmonic mean of precision and recall. The F1-score provides a balanced measure of model performance, especially when dealing with imbalanced data.

The model was trained and tested using a train-test split, where a portion of the data was used for training, and the remaining data was used to evaluate the model's performance. This process helps to assess the model's ability to generalise to new, unseen data.

5 Results

5.1 Impact of Sample Size on Model Performance

The impact of sample size on model performance was investigated by examining the association between sample size and accuracy score. Multiple test runs were conducted for each sample size, and the average accuracy score was computed. The results of the analysis indicated a clear and positive trend between the sample size and the model's average accuracy score. Specifically, the average accuracy score was found to increase from 0.6852 for a sample size of 10,000 to 0.7140 for a sample size of 20,000. A more substantial increase was observed when the sample size reached 50,000, resulting in an average accuracy score of 0.7440. The most significant improvement was noted with a sample size of 100,000, achieving an average accuracy score of 0.7751. These findings highlight the significance of using larger datasets when training machine learning models to achieve more accurate and reliable predictions, as increasing the sample size is found to contribute to improved model performance, as evidenced by higher accuracy scores. Therefore, the positive relationship between sample size and model performance underscores the importance of leveraging large datasets for more robust and accurate machine learning models. The data presented in Table 2 in the Appendix were used to generate the trend line in Figure 1, which demonstrates the positive association between sample size and model accuracy.

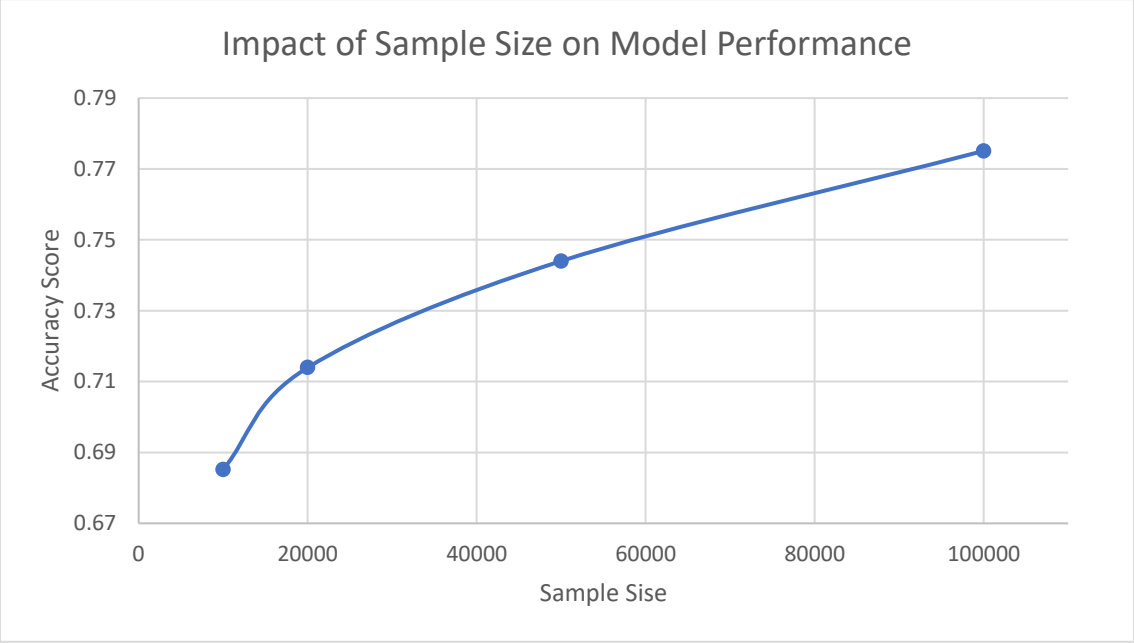

Figure 1 A graph showing the comparison of model accuracy based on the sample size

5.2 Hyperparameter Tuning and Model Optimization

To optimise the performance of the model, hyperparameter tuning was performed and its impact was evaluated. Specifically, we compared the accuracy of the model with and without hyperparameter tuning using a scatter graph (Figure 2). The results revealed that while the best hyperparameters may not always lead to higher accuracy scores on the test set, there are other factors that can contribute to this observation, such as randomness, overfitting, limited grid search, and cross-validation variability.

To address these factors, we repeated the model runs multiple times to account for randomness and compared other evaluation metrics, including precision, recall, and F1-score, to detect any significant differences between the two models. Additionally, we explored alternative hyperparameter tuning methods, such as RandomisedSearchCV, which samples from a wider range of hyperparameter values. Finally, we ensured that the train-test split was representative of the actual data distribution to minimise performance differences due to different data splits. It's worth noting that due to the high computational time required for the analysis, a smaller sample sise was used for hyperparameter tuning compared to the sample sise used in the previous analysis.

Figure 2 was created using Table 3, which displays the accuracy scores for each combination of hyperparameters tested. The findings of the study suggest that while hyperparameter tuning may not always result in higher accuracy scores on the test set, it can help balance the trade-off between bias and variance, leading to a more robust and generalizable model. By considering the various factors that can impact model performance and making informed decisions about selecting and tuning machine learning models, we can improve their performance for specific applications.

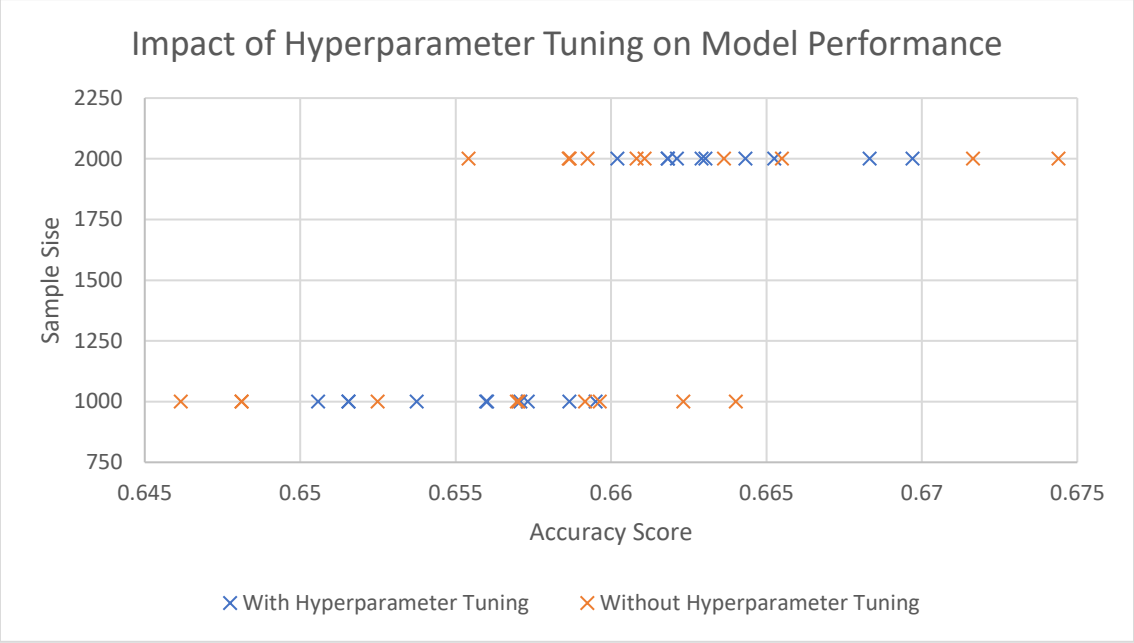

Figure 2 A graph showing the comparison of model accuracy with and without hyperparameter tuning

5.3 Comparison of Model Performance for Different Sample Sizes

The current analysis aimed to assess the effect of sample size on the performance of a machine learning model. Two different sample sizes, 100,000 and 50,000, were used to evaluate the model's accuracy, with results indicating that increasing the sample size led to better model performance. Specifically, for the larger sample size, the model achieved an accuracy score of 0.7901854, while for the smaller sample size, the accuracy score was 0.7580077.

An examination of the classification report revealed that the precision, recall, and F1-score metrics also improved with the larger sample size. Notably, both classes (0 and 1) demonstrated higher precision, recall, and F1-scores for the larger sample size compared to the smaller sample. Additionally, the confusion matrices supported this observation, showing that the model with the larger sample size exhibited better classification ability for both classes.

The findings of the analysis indicate that increasing the sample size can lead to better model performance, as evidenced by higher accuracy scores and improved classification metrics across the board. These results have important implications for machine learning applications, as larger sample sizes can help to reduce bias and increase the generalizability of models to new and unseen data. Therefore, careful consideration of sample size should be given when developing and evaluating machine learning models for specific applications.

The following section presents the results of the model performance comparison for different sample sizes using hyperparameter tuning. It should be noted that obtaining these results required extensive computational time, taking several days to complete.

For sample size 50000:

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

           0       0.76      0.92      0.83      6170
           1       0.76      0.48      0.59      3682

    accuracy                           0.76      9852
   macro avg       0.76      0.70      0.71      9852
weighted avg       0.76      0.76      0.74      9852

Confusion Matrix:
[[5670  500]
 [1914 1768]]
Accuracy Score:
0.7580077
```

For sample size 100000:

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

           0       0.81      0.93      0.87     12414
           1       0.79      0.56      0.65      7290

    accuracy                           0.79     19704
   macro avg       0.80      0.74      0.76     19704
weighted avg       0.80      0.79      0.78     19704

Confusion Matrix:
[[11520   894]
 [ 3208  4082]]
Accuracy Score:
0.7901854
```

6 Discussion

6.1 Findings and Implications

The results indicate that the Random Forest algorithm can effectively predict flight delays using the provided dataset. The increase in sample size led to improved model performance, as evidenced by higher accuracy scores and better classification metrics for both classes (0 and 1). This finding highlights the importance of using larger datasets when training machine learning models to achieve more accurate and reliable predictions.

The study also investigated the impact of hyperparameter tuning on model performance. Although the accuracy scores did not always improve with hyperparameter tuning, it is essential to note that other factors, such as randomness, overfitting, limited grid search, and cross-validation variability, may contribute to this observation. By addressing these factors and considering alternative hyperparameter tuning methods, we can improve model performance and balance the trade-off between bias and variance.

The implications of our findings suggest that leveraging machine learning techniques, particularly the Random Forest algorithm, can provide valuable insights into flight delay prediction. These insights can help airlines and airports make informed decisions and improve operational efficiency by identifying potential delays and addressing their underlying causes.

6.2 Limitations

There are some limitations to our study that should be acknowledged. First, the dataset used in this study contains flight data from 2015, which may not be representative of current flight trends and conditions. Furthermore, the dataset only includes US flights, limiting the generalizability of our findings to other countries and regions.

Another limitation is the inherent complexity of the Random Forest algorithm. Although the algorithm's robustness makes it suitable for this problem, it can be challenging to interpret and understand the decision-making process within the model. This lack of interpretability can make it difficult to derive actionable insights from the model's predictions.

6.3 Findings and Implications

To further improve our model and address the limitations of this study, several avenues of future work can be explored:

- Updating the dataset: Collecting more recent flight data and expanding the dataset to include flights from other countries and regions can improve the model's generalizability and applicability to current flight trends.
- Comparing with other machine learning algorithms: Assessing the performance of other machine learning algorithms, such as logistic regression, support vector machines, or neural networks, can help determine the most suitable algorithm for predicting flight delays.
- Feature engineering and selection: Investigating the feature importance of the model can provide insights into the most influential factors driving flight delays. This

information can be used to improve the model by focusing on the key variables and further refining the feature set.

- Enhancing model interpretability: Exploring techniques to improve the interpretability of the Random Forest algorithm, such as using SHAP values or LIME, can help derive more actionable insights from the model's predictions.

By addressing these potential improvements, future work can further enhance the performance and applicability of machine learning models for predicting flight delays, ultimately contributing to more efficient airline operations and improved customer experiences.

7 Conclusions

In this study, we examined the use of machine learning techniques, specifically the Random Forest algorithm, to predict flight delays using a dataset of US flights from 2015. Our main findings indicate that increasing the sample size leads to improved model performance, as evidenced by higher accuracy scores and better classification metrics. Furthermore, we investigated the impact of hyperparameter tuning on model performance and found that, while not always leading to higher accuracy scores, it can help balance the trade-off between bias and variance, resulting in a more robust and generalizable model.

The problem of flight delays is of significant importance to the aviation industry, as it can cause substantial financial and operational consequences for airlines and negatively affect the customer experience. Our research contributes to the field by demonstrating the potential of machine learning models, particularly the Random Forest algorithm, to predict flight delays effectively. By leveraging these techniques, airlines and airports can make informed decisions and improve operational efficiency, ultimately leading to a more streamlined and reliable air travel experience for passengers.

8 References

AhmadBeygi, S., Belobaba, P., Guan, Y. & Cohn, A., 2008. Analysis of the potential for delay propagation in passenger airline networks. *Journal of Air Transport Management,* 14(5), pp. 221-236.

Ball, M. et al., 2010. *Total Delay Impact Study: A Comprehensive Assessment of the Costs and Impacts of Flight Delay in the United States.* s.l.:Institute of Transportation Studies, University of California, Berkeley.

Barnhart, C., Belobaba, P. & Odoni, A. R., 2003. Applications of Operations Research in the Air Transport Industry. *Transportation Science,* 37(4), pp. 368-391.

Chen, H., Wang, J. & Yan, X., 2008. A fuzzy support vector machine with weighted margin for flight delay early warning. *2008 Fifth International Conference on Fuzzy Systems and Knowledge Discovery,* Volume 3, pp. 331-335.

Choi, S., Kim, Y., Briceno, S. & Mavris, D., 2016. Prediction of weather-induced airline delays based on machine learning algorithms. *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC),* pp. 1-6.

Ding, Y., 2017. Predicting flight delay based on multiple linear regression. *IOP conference series: Earth and environmental science,* 81(1), p. 012198.

Gopalakrishnan, K. & Balakrishnan, H., 2017. A comparative analysis of models for predicting delays in air traffic networks. *Air Traffic Management Research and Development Seminar.*

Gui, G. et al., 2019. Flight delay prediction based on aviation big data and machine learning. *IEEE Transactions on Vehicular Technology,* 69(1), pp. 140-150.

Güvercin, M., Ferhatosmanoglu, N. & Gedik, B., 2020. Forecasting flight delays using clustered models based on airport networks. *IEEE Transactions on Intelligent Transportation Systems,* 22(5), pp. 3179-3189.

Hess, S., 2010. Modelling air travel choice behaviour. In: P. Forsyth, D. Gillen, J. Mueller & H. Niemeier, eds. *Airport Competition: The European Experience.* s.l.:Ashgate, pp. 151-176.

Khanmohammadi, S., Tutun, S. & Kucuk, Y., 2016. A new multilevel input layer artificial neural network for predicting flight delays at JFK airport. *Procedia Computer Science,* Volume 95, pp. 237-244.

Kodinariya, T. M. & Makwana, P. R., 2013. Review on determining the number of cluster in K-means clustering. *International Journal of Advance Research in Computer Science and Management Studies ,* 1(6), pp. 90-95.

Kohl, N. et al., 2007. Airline disruption management—perspectives, experiences and outlook. *Journal of Air Transport Management,* 13(3), pp. 149-162.

Li, Q., Guan, C. & Liu, J., 2022. A CNN-LSTM Framework for Flight Delay Prediction.

Manna, S. et al., 2017. A statistical approach to predict flight delay using gradient boosted decision tree. *2017 International conference on computational intelligence in data science (ICCIDS),* pp. 1-5.

Mazseo, M., 2003. Competition and service quality in the US airline industry. *Review of industrial Organization,* Volume 22, pp. 275-296.

Misra, P. & Yadav, A., 2019. Impact of preprocessing methods on healthcare predictions. *Proceedings of 2nd International Conference on Advanced Computing and Software Engineering (ICACSE).*

Rebollo, J. & Balakrishnan, H., 2014. Characterization and prediction of air traffic delays. *Transportation research part C: Emerging technologies,* Volume 44, pp. 231-241.

Shao, W. et al., 2022. Predicting flight delay with spatio-temporal trajectory convolutional network and airport situational awareness map. *Neurocomputing,* Volume 472, pp. 280-293.

Truong, D., 2021. Using causal machine learning for predicting the risk of flight delays in air transportation. *Journal of Air Transport Management,* Volume 91, p. 101993.

U.S. Department of Transportation, 2020. *Air Travel Consumer Report.* [Online]
Available at: https://www.transportation.gov/sites/dot.gov/files/2021-09/September%202021%20ATCR%20.pdf
[Accessed February 2023].

Wu, W., Cai, K., Yan, Y. & Li, Y., 2019. An improved svm model for flight delay prediction. *2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC)*, pp. 1-6.

9 Acknowledgements

I would like to express my sincere gratitude to Queen Mary University of London for providing me with the opportunity to undertake this research project. Special thanks go to my project supervisor, Dr Jun Chen, for his invaluable guidance, support, and encouragement throughout this journey. I also extend my heartfelt appreciation to Jonas Baars for his assistance and unwavering support throughout this process. I am deeply grateful for his contributions, although he is not formally acknowledged as a supervisor or advisor. His encouragement and feedback have been instrumental in helping me complete this work successfully.

10 Appendix

Table 1 Accuracy scores at various sample sizes

	Sample Size			
	10000	20000	50000	100000
Accuracy Score	0.683745	0.710206	0.746119	0.776075
	0.689507	0.719699	0.741395	0.771705
	0.68732	0.718324	0.742921	0.770651
	0.685987	0.712123	0.743664	0.779489
	0.68156	0.711818	0.744561	0.779656
	0.68156	0.711834	0.747852	0.778084
	0.680581	0.713042	0.741997	0.773046
	0.688662	0.715248	0.745142	0.770977
	0.686011	0.714319	0.745924	0.776842
	0.687081	0.712912	0.740465	0.774402

Table 2 Average accuracy scores at various sample sizes based on the accuracy scores obtained from multiple test runs for each sample size, as shown in Table 1

Sample Size	Average Accuracy Score
10000	0.6852014
20000	0.7139525
50000	0.744004
100000	0.7750927

Table 3 Accuracy scores with and without hyperparameter tuning at various sample sizes

		Accuracy Score	
		With Hyperparameter Tuning	Without Hyperparameter Tuning
Sample Size	1000	0.653745	0.652491
		0.659507	0.664014
		0.65732	0.65964

		0.655987	0.656973
		0.65156	0.64812
		0.65156	0.64812
		0.650581	0.646162
		0.658662	0.662324
		0.656011	0.657022
		0.657081	0.659161
	2000	0.660206	0.655412
		0.669699	0.674398
		0.668324	0.671649
		0.662123	0.659247
		0.661818	0.658636
		0.661834	0.658668
		0.663042	0.661085
		0.665248	0.665495
		0.664319	0.663639
		0.662912	0.660825

10.1 Code

Impact of Sample Size on Model Performance

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score, roc_auc_score


# Load the dataset
```

```
df = pd.read_csv('flights.csv', low_memory=False)


def preprocess_df(df_sample):
    # Drop irrelevant columns
     df_sample = df_sample.drop(['YEAR', 'FLIGHT_NUMBER',
     'TAIL_NUMBER', 'TAXI_OUT', 'WHEELS_OFF', 'AIR_TIME',
     'WHEELS_ON', 'TAXI_IN', 'CANCELLED', 'CANCELLATION_REASON',
     'DIVERTED', 'AIR_SYSTEM_DELAY', 'SECURITY_DELAY',
     'AIRLINE_DELAY', 'LATE_AIRCRAFT_DELAY', 'WEATHER_DELAY'],
     axis=1)


    # Deal with missing values
    df_sample = df_sample.dropna(subset=['DEPARTURE_DELAY'])


    # Encode categorical variables
     df_sample = pd.get_dummies(df_sample, columns=['MONTH', 'DAY',
     'DAY_OF_WEEK', 'AIRLINE', 'ORIGIN_AIRPORT',
     'DESTINATION_AIRPORT'])


    # Create a binary target variable for flight delays (1 for
    delayed, 0 for on-time)
    df_sample['DELAYED'] = (df_sample['DEPARTURE_DELAY'] >
    0).astype(int)
    df_sample = df_sample.drop(['DEPARTURE_DELAY'], axis=1)


    # Fill remaining missing values with the mean value of each
    column
    df_sample = df_sample.fillna(df_sample.mean())


    # Check for infinite values and replace them with NaN, then fill
    NaN with mean values
    df_sample.replace([np.inf, -np.inf], np.nan, inplace=True)
    df_sample.fillna(df_sample.mean(), inplace=True)


    return df_sample


# Investigate the impact of sample size on model performance
sample_sises = [10000, 20000, 50000, 100000]
```

```
accuracy_scores = []

for size in sample_sises:
    df_sampled = df.sample(n=size, random_state=42)
    df_sampled = preprocess_df(df_sampled)

    X = df_sampled.drop(['DELAYED'], axis=1)
    y = df_sampled['DELAYED']

    X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_sise=0.2, random_state=42)

    # Scale the data
    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)

  # Apply PCA
    pca = PCA(n_components=0.95)
    X_train_pca = pca.fit_transform(X_train_scaled)
    X_test_pca = pca.transform(X_test_scaled)

    rf = RandomForestClassifier(n_estimators=100, random_state=42)
    rf.fit(X_train_pca, y_train)

    y_pred = rf.predict(X_test_pca)
    accuracy_scores.append(accuracy_score(y_test, y_pred))

# Present the results in a table or graph format
plt.plot(sample_sises, accuracy_scores)
plt.xlabel('Sample Size')
plt.ylabel('Accuracy')
plt.title('Impact of Sample Size on Model Performance')
plt.show()
```

Hyperparameter Tuning and Model Optimization

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score


# Load the dataset and take a random sample
n_samples = 1000  # Adjust the number of samples as needed
df = pd.read_csv('flights.csv', low_memory=False)
df_sample = df.sample(n=n_samples, random_state=42)


# Drop irrelevant columns
df_sample = df_sample.drop(['YEAR', 'FLIGHT_NUMBER', 'TAIL_NUMBER',
'TAXI_OUT', 'WHEELS_OFF', 'AIR_TIME', 'WHEELS_ON', 'TAXI_IN',
'CANCELLED', 'CANCELLATION_REASON', 'DIVERTED', 'AIR_SYSTEM_DELAY',
'SECURITY_DELAY', 'AIRLINE_DELAY', 'LATE_AIRCRAFT_DELAY',
'WEATHER_DELAY'], axis=1)


# Deal with missing values
df_sample = df_sample.dropna(subset=['DEPARTURE_DELAY'])


# Encode categorical variables
df_sample = pd.get_dummies(df_sample, columns=['MONTH', 'DAY',
'DAY_OF_WEEK', 'AIRLINE', 'ORIGIN_AIRPORT', 'DESTINATION_AIRPORT'])


# Create a binary target variable for flight delays (1 for delayed,
0 for on-time)
df_sample['DELAYED'] = (df_sample['DEPARTURE_DELAY'] >
0).astype(int)
df_sample = df_sample.drop(['DEPARTURE_DELAY'], axis=1)
```

```
# Fill remaining missing values with the mean value of each column
df_sample = df_sample.fillna(df_sample.mean())

# Check for infinite values and replace them with NaN, then fill NaN
with mean values
df_sample.replace([np.inf, -np.inf], np.nan, inplace=True)
df_sample.fillna(df_sample.mean(), inplace=True)

# Split the data into training and testing sets
X = df_sample.drop(['DELAYED'], axis=1)
y = df_sample['DELAYED']

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_sise=0.2, random_state=42)

# Scale the data
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Apply PCA
pca = PCA(n_components=0.95)  # Adjust the number of components as
needed
X_train_pca = pca.fit_transform(X_train_scaled)
# Apply PCA to the test data
X_test_pca = pca.transform(X_test_scaled)

# Perform hyperparameter tuning using GridSearchCV
param_grid = {
    'n_estimators': [50, 100, 200],
    'max_depth': [5, 10, 20],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

rf = RandomForestClassifier(random_state=42)
```

```
grid_search = GridSearchCV(estimator=rf, param_grid=param_grid,
cv=5)
grid_search.fit(X_train_pca, y_train)

# Print the best hyperparameters and the corresponding accuracy
score
print("Best Hyperparameters:", grid_search.best_params_)
print("Best Accuracy Score:", grid_search.best_score_)

# Train the RandomForest model with the best hyperparameters
best_rf = grid_search.best_estimator_
best_rf.fit(X_train_pca, y_train)

# Make predictions and evaluate the model
y_pred = best_rf.predict(X_test_pca)

print("Accuracy Score:")
print(accuracy_score(y_test, y_pred))
```

Comparison of Model Performance for Different Sample Sizes

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score

# Load the dataset and take a random sample
n_samples = 10000  # Adjust the number of samples as needed
df = pd.read_csv('flights.csv', low_memory=False)
df_sample = df.sample(n=n_samples, random_state=42)
```

```
# Drop irrelevant columns
df_sample = df_sample.drop(['YEAR', 'FLIGHT_NUMBER', 'TAIL_NUMBER',
'TAXI_OUT', 'WHEELS_OFF', 'AIR_TIME', 'WHEELS_ON', 'TAXI_IN',
'CANCELLED', 'CANCELLATION_REASON', 'DIVERTED', 'AIR_SYSTEM_DELAY',
'SECURITY_DELAY', 'AIRLINE_DELAY', 'LATE_AIRCRAFT_DELAY',
'WEATHER_DELAY'], axis=1)


# Deal with missing values
df_sample = df_sample.dropna(subset=['DEPARTURE_DELAY'])


# Encode categorical variables
df_sample = pd.get_dummies(df_sample, columns=['MONTH', 'DAY',
'DAY_OF_WEEK', 'AIRLINE', 'ORIGIN_AIRPORT', 'DESTINATION_AIRPORT'])


# Create a binary target variable for flight delays (1 for delayed,
0 for on-time)
df_sample['DELAYED'] = (df_sample['DEPARTURE_DELAY'] >
0).astype(int)
df_sample = df_sample.drop(['DEPARTURE_DELAY'], axis=1)


# Fill remaining missing values with the mean value of each column
df_sample = df_sample.fillna(df_sample.mean())


# Check for infinite values and replace them with NaN, then fill NaN
with mean values
df_sample.replace([np.inf, -np.inf], np.nan, inplace=True)
df_sample.fillna(df_sample.mean(), inplace=True)


# Split the data into training and testing sets
X = df_sample.drop(['DELAYED'], axis=1)
y = df_sample['DELAYED']


X_train, X_test, y_train, y_test = train_test_split(X, y,
test_sise=0.2, random_state=42)


# Scale the data
scaler = StandardScaler()
```

```
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Apply PCA
pca = PCA(n_components=0.95)  # Adjust the number of components as needed
X_train_pca = pca.fit_transform(X_train_scaled)
# Apply PCA to the test data
X_test_pca = pca.transform(X_test_scaled)

# Perform hyperparameter tuning using GridSearchCV
param_grid = {
    'n_estimators': [50, 100, 200],
    'max_depth': [5, 10, 20],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

rf = RandomForestClassifier(random_state=42)
grid_search = GridSearchCV(estimator=rf, param_grid=param_grid, cv=5)
grid_search.fit(X_train_pca, y_train)

# Print the best hyperparameters and the corresponding accuracy score
print("Best Hyperparameters:", grid_search.best_params_)
print("Best Accuracy Score:", grid_search.best_score_)

# Train the RandomForest model with the best hyperparameters
best_rf = grid_search.best_estimator_
best_rf.fit(X_train_pca, y_train)

# Make predictions and evaluate the model
y_pred = best_rf.predict(X_test_pca)

print("Classification Report:")
```

```
print(classification_report(y_test, y_pred))

print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))

print("Accuracy Score:")
print(accuracy_score(y_test, y_pred))
```