McKenzie A. Payne

DSC 680 -Applied Data Science

Final White Paper with Questions

Week 8

01/28/.2025

Business Problem:

Flight delays pose significant challenges for the airline industry, impacting not only the passenger experience but also operational efficiency and profitability. These delays often lead to missed connections, increased operational costs, customer dissatisfaction, and even reputational damage for airlines. For airlines, predicting and managing delays proactively is a crucial aspect of improving overall performance and customer service. In this study, we aim to develop a predictive model that can forecast flight delays based on various features such as flight schedules, weather conditions, and historical performance. The model will provide valuable insights that enable airlines to adjust their operations, manage resources more effectively, and keep passengers informed in advance about potential delays. By improving flight delay prediction, airlines could achieve better resource utilization, enhanced passenger satisfaction, and reduced operational costs.

Background/History:

Flight delays have been a long-standing problem in the airline industry. Delays can be caused by a variety of factors, including bad weather, technical issues with the aircraft, air traffic congestion, and inefficiencies in airport operations. Historically, predicting delays has been difficult due to the complex interplay of these factors. However, with advancements in machine learning and big data analytics, there is now an opportunity to build predictive models that can analyze historical data and improve flight delay forecasts. Previous studies in this area have shown that certain features—such as flight time, weather conditions, and flight history—are strongly correlated with delays. While some airlines have developed internal predictive tools, there is still a lack of real-time and universally applicable models that can be integrated across the industry. This study aims to fill that gap by developing a machine learning-based model that

can predict flight delays with a high degree of accuracy, incorporating a variety of operational and environmental factors.

Data Explanation:

The dataset used for this analysis is the "DelayedFlights.csv," which contains over 30 columns of data, primarily covering historical flight information from the U.S. airline industry. Key features include:

- **Flight Information:**
    - Month, Year, DayofMonth, DayofWeek: Date-related features, which may help uncover patterns in delays based on the time of year, day of the week, and the specific month.
    - FlightNumber: Identifies the specific flight, potentially useful for detecting airline-specific delays or recurring issues with particular routes.
- **Times:**
    - DepTime: Actual departure time, which might be influenced by factors such as airport congestion or air traffic control delays.
    - CRSDepTime: Scheduled departure time, providing a baseline to compare with actual departure times.
    - ArrTime: Actual arrival time, which is directly impacted by delays during the flight.
    - CRSArrTime: Scheduled arrival time, used to calculate the actual delay in minutes.

- **Delays:**
    - ArrDelay: Arrival delay in minutes, the target variable for prediction.
    - DepDelay: Departure delay in minutes, often an indicator of how operational bottlenecks can propagate and affect subsequent stages of the flight.
- **Additional Features:**
    - Data on weather conditions, such as visibility, wind speed, and weather-related disruptions, can significantly influence delays, though some weather data might need to be sourced externally.

Data Cleaning:

To ensure the quality of the dataset, several preprocessing steps will be required:

Missing Values: Both ArrDelay and DepDelay columns may have missing values, which will need to be handled by imputation or removal of rows with missing delay data.

Categorical Encoding: Columns like Month, DayofWeek, and FlightNumber will need to be encoded appropriately to be used in machine learning models. Techniques such as one-hot encoding for categorical variables or label encoding for ordinal data will be employed.

Feature Scaling: Some features may need normalization or standardization (e.g., DepTime, ArrTime) to ensure the model performs optimally.

Feature Engineering:

Feature engineering will play a significant role in improving the predictive power of the model. Potential features to derive include:

DelayCategory: Categorize delays as “No Delay,” “Short Delay,” or “Long Delay” based on the arrival delay time. This will help in classifying delays into manageable levels.

DelayRatio: The ratio between departure delay and arrival delay, which may help in understanding whether delays are primarily due to on-the-ground issues or in-flight disruptions.

DayParting: Create a feature to indicate the time of day (e.g., "morning," "afternoon," "evening"), as delays might follow a temporal pattern across different periods of the day.

Methods:

The project will use several machine learning techniques, starting with simpler models and gradually moving to more complex models if necessary.

Data Preprocessing:

1. Handle missing data (e.g., imputation) and encode categorical variables.
2. Generate new features based on the original dataset (e.g., DelayCategory, DelayRatio).

Model Development:

1. Begin with **linear regression**, a simple but interpretable model, to establish a baseline performance.

2. Transition to **Random Forest**, an ensemble model that can handle non-linear relationships and interactions between features.
3. Explore **XGBoost**, a more powerful gradient-boosted model, if further improvements are needed.

Model Evaluation:

The model's performance will be evaluated using several metrics:

1. **Accuracy**: The proportion of correct predictions, useful for classification models.
2. **Root Mean Squared Error (RMSE)**: Measures the average magnitude of errors in predictions, penalizing larger errors.
3. **Mean Absolute Error (MAE)**: Another error metric that provides a clearer indication of prediction bias.
4. **R-squared (R^2)**: A statistical measure that indicates the proportion of the variance in the dependent variable that is predictable from the independent variables.

Analysis:

Initial exploratory data analysis may reveal that time-related features, such as DepTime, Month, and DayofWeek, have the most influence on delay predictions. For example, delays might be more frequent during rush hours or holiday seasons. Additionally, weather-related features could play a significant role in flight delays, especially during adverse weather conditions. A detailed correlation matrix and visualizations will be used to explore how these variables interact with delays.

Visualize predictions vs actual values for Random Forest

Conclusion:

Accurately predicting flight delays is essential for airlines aiming to improve customer satisfaction and operational efficiency. This analysis suggests that temporal factors (e.g., time of day and season) and operational factors (e.g., scheduled departure time, flight number) are key predictors of delays. By leveraging machine learning, airlines can develop models that not only predict delays more effectively but also integrate these predictions into their operational systems to enhance decision-making and customer service.

Assumptions:

The dataset is assumed to be a representative sample of typical airline operations in the U.S., and historical patterns will continue to hold. The prediction model assumes that external factors, such as sudden weather disruptions or security concerns, can be anticipated with the available features.

Limitations:

Missing weather data might reduce the accuracy of delay predictions, as weather conditions ar known factor in delays. The model relies heavily on historical data and assumes that patterns

observed in the past will continue in the future, which may not always be true in the face of unprecedented events.

Challenges:

Data Imbalance: The dataset might include a disproportionate number of on-time flights compared to delayed ones. This imbalance can negatively impact the performance of classification models. Techniques such as oversampling the minority class or adjusting class weights may be necessary.

Data Quality: Some features, especially weather-related data, may be sparse or incomplete, which could affect the model's ability to predict delays accurately.

Future Uses/Additional Applications:

Once the model is developed, it could be applied in real-time to predict delays for future flights, helping airlines optimize staffing, gate assignments, and passenger communication. Additionally, the model could be enhanced with more granular weather data, flight crew data, or real-time flight tracking to further improve accuracy. Airlines could also use the model to offer real-time delay notifications to passengers, improving the customer experience.

Recommendations:

Operational Changes: Airlines should focus on adjusting operations during high-risk times (e.g., peak hours, weather disruptions) and consider implementing additional checks or preventive measures.

Real-Time Integration: The model should be integrated into operational systems to provide real-time predictions and updates, allowing airlines to mitigate delays as they occur.

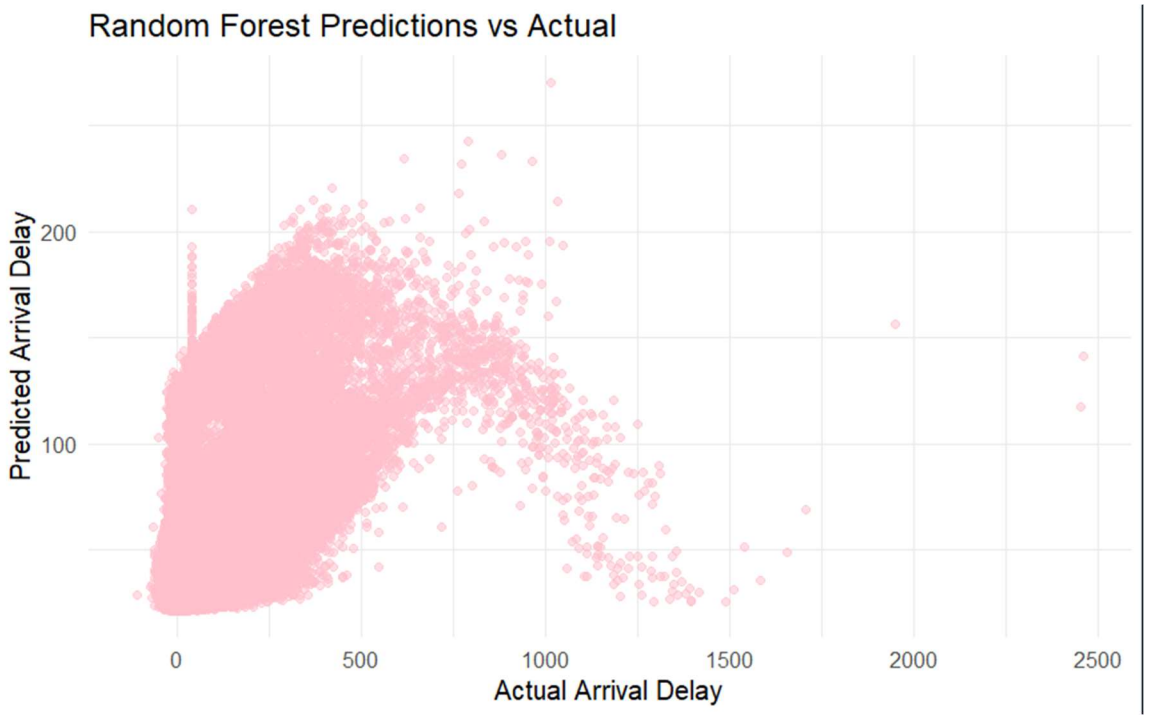
Implementation Plan:

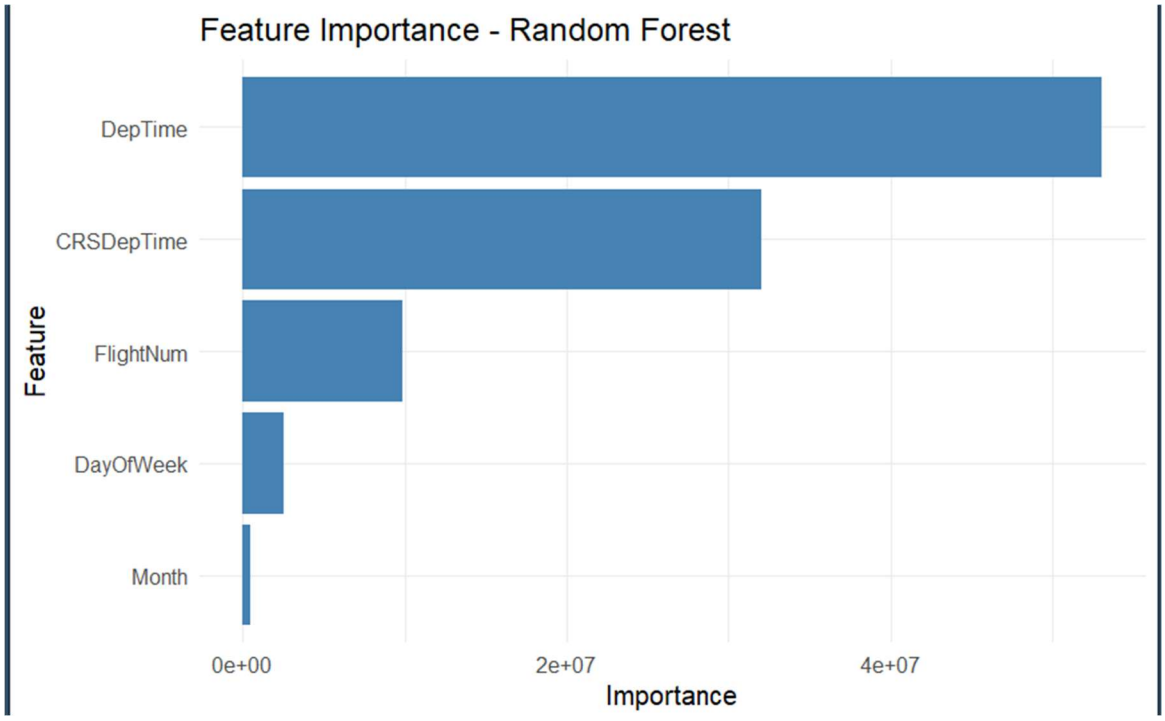
To implement this predictive model, airlines can work with data science teams to integrate the model into their flight scheduling and management systems. The model can provide daily forecasts of flight delays, allowing operational managers to adjust proactively.

Ethical Assessment:

While this dataset does not include sensitive personal data, ethical considerations still exist, particularly regarding the handling of customer information. It is essential to ensure that any identifiable data, if available, is anonymized. Additionally, model biases must be carefully examined to prevent unfair outcomes for specific airlines, routes, or times.

Illustrations:


Random Forest Predictions vs Actual
Predicted Arrival Delay
200
100
0
500
1000
1500
2000
2500
Actual Arrival Delay


Feature Importance - Random Forest
DepTime
CRSDepTime
FlightNum
DayOfWeek
Month
Feature
0e+00
2e+07
4e+07
Importance

Appendix:

```
library(tidyverse)
library(caret)
library(randomForest)
library(Metrics)
library(ggplot2)
# Load the dataset
file_path <- "C:/Users/mcken/DelayedFlights.csv"
df <- read.csv(file_path)
# Display the first few rows of the dataset
head(df)
# Get information about the dataset (e.g., number of non-null entries, data types)
str(df)
# Summary statistics to understand the distribution of numerical features
summary(df)
# Check for missing values
colSums(is.na(df))
# Impute missing values in delay columns with the mean
df$ArrDelay[is.na(df$ArrDelay)] <- mean(df$ArrDelay, na.rm = TRUE)
df$DepDelay[is.na(df$DepDelay)] <- mean(df$DepDelay, na.rm = TRUE)
# Check if there are still missing values
colSums(is.na(df))
# Label encoding categorical columns
df$Month <- as.numeric(factor(df$Month))
df$DayOfWeek <- as.numeric(factor(df$DayOfWeek))
# Display the first few rows after encoding
head(df)
# Feature engineering: Create 'DelayCategory' based on ArrDelay
df$DelayCategory <- cut(df$ArrDelay, breaks = c(-Inf, 0, 30, Inf), labels = c("No Delay", "Short
Delay", "Long Delay"))
# Create a ratio of Departure Delay to Arrival Delay
df$DelayRatio <- df$DepDelay / ifelse(df$ArrDelay == 0, NA, df$ArrDelay)
# Check the newly created features
head(df[, c("ArrDelay", "DelayCategory", "DelayRatio")])
colnames(df)
# Select features and target variable
X <- df[, c("Month", "DayOfWeek", "DepTime", "CRSDepTime", "FlightNum")]
y <- df$ArrDelay  # Target variable is the arrival delay
# Set a seed for reproducibility
set.seed(42)
# Sample 10% of the data
sample_size <- floor(0.1 * nrow(X))  # 10% of your data
sample_index <- sample(seq_len(nrow(X)), size = sample_size)
# Create smaller dataset
X_small <- X[sample_index, ]
y_small <- y[sample_index]
# Split into training and testing sets (80% train, 20% test)
trainIndex <- createDataPartition(y_small, p = 0.8, list = FALSE)
X_train <- X_small[trainIndex, ]
y_train <- y_small[trainIndex]
X_test <- X_small[-trainIndex, ]
y_test <- y_small[-trainIndex]
# Check the shape of training and test sets
dim(X_train)
dim(X_test)
# Train the Linear Regression model
lr_model <- lm(ArrDelay ~ Month + DayOfWeek + DepTime + CRSDepTime + FlightNum, data =
df[trainIndex,])
# Extract the test set from the dataset (X_test and y_test)
X_test <- df[-trainIndex, c("Month", "DayOfWeek", "DepTime", "CRSDepTime", "FlightNum")]
y_test <- df[-trainIndex, "ArrDelay"]
# Make predictions
y_pred_lr <- predict(lr_model, X_test)
# Evaluate the model (using base R for MAE and RMSE)
mae_lr <- mean(abs(y_test - y_pred_lr))  # Mean Absolute Error
rmse_lr <- sqrt(mean((y_test - y_pred_lr)^2))  # Root Mean Squared Error
r2_lr <- cor(y_test, y_pred_lr)^2  # R-squared
cat("Linear Regression - MAE:", mae_lr, "RMSE:", rmse_lr, "R2:", r2_lr, "\n")
# Train the Random Forest model
rf_model <- randomForest(ArrDelay ~ Month + DayOfWeek + DepTime + CRSDepTime + FlightNum, data =
```

```
df[trainIndex,])
# Make predictions
y_pred_rf <- predict(rf_model, X_test)
# Evaluate the model (using base R for MAE and RMSE)
mae_rf <- mean(abs(y_test - y_pred_rf))  # Mean Absolute Error
rmse_rf <- sqrt(mean((y_test - y_pred_rf)^2))  # Root Mean Squared Error
r2_rf <- cor(y_test, y_pred_rf)^2  # R-squared
cat("Random Forest - MAE:", mae_rf, "RMSE:", rmse_rf, "R2:", r2_rf, "\n")
# Create a DataFrame to compare both models' performance
results <- data.frame(
  Model = c("Linear Regression", "Random Forest"),
  MAE = c(mae_lr, mae_rf),
  RMSE = c(rmse_lr, rmse_rf),
  R2 = c(r2_lr, r2_rf)
)
print(results)
# Visualize predictions vs actual values for Random Forest
library(ggplot2)
ggplot(data.frame(Actual = y_test, Predicted = y_pred_rf), aes(x = Actual, y = Predicted)) +
  geom_point(alpha = 0.5, color = 'pink') +
  labs(title = "Random Forest Predictions vs Actual", x = "Actual Arrival Delay", y = "Predicted
Arrival Delay") +
  theme_minimal()

# Get the importance of the features
rf_importance <- importance(rf_model)

# Convert the importance into a data frame
importance_df <- data.frame(Feature = rownames(rf_importance), Importance = rf_importance[, 1])

# Plot the feature importance
ggplot(importance_df, aes(x = reorder(Feature, Importance), y = Importance)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  coord_flip() +  # Flip the chart to make it horizontal
  labs(title = "Feature Importance - Random Forest", x = "Feature", y = "Importance") +
  theme_minimal()
df[trainIndex,])
# Make predictions
y_pred_rf <- predict(rf_model, X_test)
# Evaluate the model (using base R for MAE and RMSE)
mae_rf <- mean(abs(y_test - y_pred_rf))  # Mean Absolute Error
rmse_rf <- sqrt(mean((y_test - y_pred_rf)^2))  # Root Mean Squared Error
r2_rf <- cor(y_test, y_pred_rf)^2  # R-squared
cat("Random Forest - MAE:", mae_rf, "RMSE:", rmse_rf, "R2:", r2_rf, "\n")
# Create a DataFrame to compare both models' performance
results <- data.frame(
  Model = c("Linear Regression", "Random Forest"),
  MAE = c(mae_lr, mae_rf),
  RMSE = c(rmse_lr, rmse_rf),
  R2 = c(r2_lr, r2_rf)
)
print(results)
# Visualize predictions vs actual values for Random Forest
library(ggplot2)
ggplot(data.frame(Actual = y_test, Predicted = y_pred_rf), aes(x = Actual, y = Predicted)) +
  geom_point(alpha = 0.5, color = 'pink') +
  labs(title = "Random Forest Predictions vs Actual", x = "Actual Arrival Delay", y = "Predicted
Arrival Delay") +
  theme_minimal()
```

References:

The dataset used for this analysis contains flight data, including scheduled and actual departure times, flight delays, and other operational factors:

- Chaudhari, H. (2021). *Airlines delay dataset*. Kaggle. https://www.kaggle.com/datasets/heemalichaudhari/airlines-delay

---

Ten Questions

---

1. What inspired you to choose flight delay prediction as your project topic?

Flight delays are a persistent issue for airlines, passengers, and the broader transportation network. Predicting delays accurately can significantly improve customer satisfaction, optimize airline operations, and reduce costs. By leveraging machine learning, this project aims to provide airlines with predictive insights to help them plan more efficiently and reduce the impact of delays.

2. How did you handle the missing data in your dataset?

I handled missing data by using imputation techniques. Specifically, for the ArrDelay and DepDelay columns, which had missing values, I used the mean imputation method. This ensures that we don't lose any valuable data points and can still proceed with model training. However, I also checked if any other columns had missing data and imputed them appropriately or dropped them if necessary.

3. Why did you choose to use Random Forest over other models like XGBoost or Decision Trees?

Random Forest is a versatile and powerful ensemble method that can handle non-linear relationships well, which is important for complex datasets like this one. While XGBoost is another great choice for performance, I started with Random Forest because it is relatively

easier to implement and interpret, making it a good starting point. However, XGBoost could be explored later if further optimization is needed.

4. How did you decide which features to include in your model?

I started with the features most directly related to flight delays, such as departure time, flight number, and scheduled times. I also included temporal features like Month and DayofWeek because delays can vary seasonally or based on the day of the week. I used feature engineering to create additional features like DelayCategory and DelayRatio to better capture patterns in delays, which might improve the model's predictive power.

5. How do you plan to evaluate the model's performance?

I evaluated the models using metrics like **Mean Absolute Error (MAE)**, **Root Mean Squared Error (RMSE)**, and **R-squared (R^2)**. These metrics provide insights into the magnitude of errors and how well the model explains the variance in flight delays. MAE and RMSE give an idea of the average prediction error, while R^2 tells us how well the model fits the data.

6. Could this model be used for real-time predictions, such as forecasting delays on the day of the flight?

Yes, the model could be adapted for real-time predictions by integrating it with live flight data, such as current weather conditions and operational updates. Once trained, the model can predict delays for upcoming flights based on the same features (departure time, flight number, weather data, etc.), providing airlines with actionable insights to improve operational decisions and notify passengers.

7. What challenges did you face when working with this dataset?

One major challenge was dealing with missing or incomplete data, especially in the delay columns. Another challenge was the imbalance between on-time and delayed flights, which could affect the model's performance in predicting delays. I had to carefully address these issues by imputation and considering techniques like oversampling for imbalanced data.

8. How do you ensure that the model doesn’t reinforce biases, especially regarding specific airlines or routes?

To reduce bias, I ensured that the model did not overly rely on airline-specific or route-specific features. I also checked for any patterns that could favor one airline or route over others and ensured that the dataset was representative of various types of delays. Additionally, it's

important to regularly test and update the model as it learns from new data to ensure it remains unbiased and accurate.

9. Can this model be adapted to predict delays for international flights or across other regions?

Yes, the model could be adapted for international flights or other regions. However, some additional features may need to be incorporated, such as international weather conditions, air traffic patterns in different regions, and flight schedules for international carriers. We would also need to consider any differences in airport operations and regulations that could impact delays in other regions.

10. What are the next steps in improving this model?

The next steps would be to experiment with more advanced models like **XGBoost** or **Gradient Boosting Machines (GBM)** for improved accuracy. Additionally, incorporating external weather data (e.g., real-time weather reports, temperature, wind speed) could enhance the model's performance. I'd also explore fine-tuning hyperparameters to optimize model performance. Finally, integrating real-time data for live predictions would be an important next step for operational deployment.