flight_delays

April 12, 2025

1 Flight Delay Prediction and Analysis - Progress Report

Team Members: Sai C., Maryann O., Matt J., Jeremiah B., Ricky M.
GitHub: https://github.com/juebenjamin/flight-delay-analysis

1.1 Project Introduction

Flight delays are a persistent challenge in air travel, causing disruptions for passengers, increased operational costs for airlines, and reduced efficiency across airports. Our project aims to analyze historical flight data to identify patterns and build predictive models that can forecast flight delays. By understanding the key factors that contribute to flight delays - such as weather conditions, airline operations, airport infrastructure, and time-related variables - we hope to provide insights that could help mitigate these issues.

As highlighted in our proposal, the United States had the third-highest flight cancellation rate (2.76%) among the ten countries considered in 2024, and according to the FAA, weather conditions cause over 75% of significant flight delays. Our analysis will explore these relationships while building predictive models to anticipate delays based on various factors.

1.2 Changes Since the Proposal

Since our initial proposal, our scope has remained largely consistent with a few refinements:

- **Focus on U.S. Domestic Flights:** Initially, we considered including international flights in our analysis, but we've narrowed our focus to U.S. domestic flights to ensure data consistency and quality.
- **Data Source Refinement:** Rather than using multiple data sources (OpenSky Network/Flightradar24), we've decided to focus primarily on the Bureau of Transportation Statistics (BTS) data for historical flight delay information, which provides a comprehensive and reliable dataset.

No significant parts have been removed from our plan. We are still pursuing our original goals of conducting exploratory data analysis, feature engineering, implementing machine learning models, and assessing airport patterns related to delays.

1.3 Data Preparation

Our primary dataset comes from the Bureau of Transportation Statistics' (BTS) On-Time Performance database, which provides detailed information about flight delays across U.S. airports and carriers.

[1]:
```
import pandas as pd
from pathlib import Path
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix,␣
 ↪accuracy_score, precision_recall_fscore_support
from sklearn.pipeline import Pipeline

# Set the aesthetics for the plots
sns.set_style("whitegrid")
plt.rcParams['figure.figsize'] = (12, 6)
plt.rcParams['font.size'] = 12

# Load Data
data_path = Path("../data/Airline_Delay_Cause.csv")
df = pd.read_csv(data_path)

# Display basic information about the dataset
print("-------------------- Before Cleaning --------------------")
print(df.info())
print(df.head())

# Check for duplicate rows
duplicates = df.duplicated().sum()
print(f"\nNumber of duplicate rows: {duplicates}")

# Check for missing values
missing_values = df.isnull().sum()
print("\nMissing values per column:")
print(missing_values[missing_values > 0])

# Drop duplicate rows
df_dup = df.drop_duplicates()

# Drop rows with missing values
df_na = df.dropna()

print("\n -------------------- After Cleaning --------------------")
print(df_na.info())
print(df_na.head())
```

```python
# Save the cleaned dataset for further analysis
df_clean = df_na.copy()
```

```
-------------------- Before Cleaning --------------------
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 394391 entries, 0 to 394390
Data columns (total 21 columns):
 #   Column               Non-Null Count   Dtype
---  ------               --------------   -----
 0   year                 394391 non-null  int64
 1   month                394391 non-null  int64
 2   carrier              394391 non-null  object
 3   carrier_name         394391 non-null  object
 4   airport              394391 non-null  object
 5   airport_name         394391 non-null  object
 6   arr_flights          393735 non-null  float64
 7   arr_del15            393443 non-null  float64
 8   carrier_ct           393735 non-null  float64
 9   weather_ct           393735 non-null  float64
 10  nas_ct               393735 non-null  float64
 11  security_ct          393735 non-null  float64
 12  late_aircraft_ct     393735 non-null  float64
 13  arr_cancelled        393735 non-null  float64
 14  arr_diverted         393735 non-null  float64
 15  arr_delay            393735 non-null  float64
 16  carrier_delay        393735 non-null  float64
 17  weather_delay        393735 non-null  float64
 18  nas_delay            393735 non-null  float64
 19  security_delay       393735 non-null  float64
 20  late_aircraft_delay  393735 non-null  float64
dtypes: float64(15), int64(2), object(4)
memory usage: 63.2+ MB
None
   year  month carrier       carrier_name airport  \
0  2024     11      9E  Endeavor Air Inc.     ABE
1  2024     11      9E  Endeavor Air Inc.     ABY
2  2024     11      9E  Endeavor Air Inc.     AEX
3  2024     11      9E  Endeavor Air Inc.     AGS
4  2024     11      9E  Endeavor Air Inc.     ALB

                                      airport_name  arr_flights  arr_del15  \
0  Allentown/Bethlehem/Easton, PA: Lehigh Valley …         82.0       15.0
1             Albany, GA: Southwest Georgia Regional          8.0        0.0
2          Alexandria, LA: Alexandria International         82.0       10.0
3       Augusta, GA: Augusta Regional at Bush Field         75.0        3.0
4                   Albany, NY: Albany International         91.0       10.0

   carrier_ct  weather_ct  …  security_ct  late_aircraft_ct  arr_cancelled  \
```

```
0        7.72         0.0  …          0.0              3.89            0.0
1        0.00         0.0  …          0.0              0.00            0.0
2        5.05         1.0  …          0.0              2.49            0.0
3        2.00         0.0  …          0.0              1.00            0.0
4        2.53         0.0  …          0.0              3.61            0.0

   arr_diverted  arr_delay  carrier_delay  weather_delay  nas_delay  \
0           0.0      550.0          301.0            0.0      107.0
1           0.0        0.0            0.0            0.0        0.0
2           2.0      559.0          298.0           55.0       48.0
3           0.0       93.0           73.0            0.0        0.0
4           0.0      406.0          196.0            0.0      110.0

   security_delay  late_aircraft_delay
0             0.0                142.0
1             0.0                  0.0
2             0.0                158.0
3             0.0                 20.0
4             0.0                100.0

[5 rows x 21 columns]

Number of duplicate rows: 0

Missing values per column:
arr_flights            656
arr_del15              948
carrier_ct             656
weather_ct             656
nas_ct                 656
security_ct            656
late_aircraft_ct       656
arr_cancelled          656
arr_diverted           656
arr_delay              656
carrier_delay          656
weather_delay          656
nas_delay              656
security_delay         656
late_aircraft_delay    656
dtype: int64

 -------------------- After Cleaning --------------------
<class 'pandas.core.frame.DataFrame'>
Index: 393443 entries, 0 to 394390
Data columns (total 21 columns):
 #   Column               Non-Null Count   Dtype
---  ------               --------------   -----
```

```
 0   year                 393443 non-null  int64
 1   month                393443 non-null  int64
 2   carrier              393443 non-null  object
 3   carrier_name         393443 non-null  object
 4   airport              393443 non-null  object
 5   airport_name         393443 non-null  object
 6   arr_flights          393443 non-null  float64
 7   arr_del15            393443 non-null  float64
 8   carrier_ct           393443 non-null  float64
 9   weather_ct           393443 non-null  float64
 10  nas_ct               393443 non-null  float64
 11  security_ct          393443 non-null  float64
 12  late_aircraft_ct     393443 non-null  float64
 13  arr_cancelled        393443 non-null  float64
 14  arr_diverted         393443 non-null  float64
 15  arr_delay            393443 non-null  float64
 16  carrier_delay        393443 non-null  float64
 17  weather_delay        393443 non-null  float64
 18  nas_delay            393443 non-null  float64
 19  security_delay       393443 non-null  float64
 20  late_aircraft_delay  393443 non-null  float64
dtypes: float64(15), int64(2), object(4)
memory usage: 66.0+ MB
None
   year  month carrier       carrier_name airport  \
0  2024     11      9E  Endeavor Air Inc.     ABE
1  2024     11      9E  Endeavor Air Inc.     ABY
2  2024     11      9E  Endeavor Air Inc.     AEX
3  2024     11      9E  Endeavor Air Inc.     AGS
4  2024     11      9E  Endeavor Air Inc.     ALB

                                      airport_name  arr_flights  arr_del15  \
0  Allentown/Bethlehem/Easton, PA: Lehigh Valley …         82.0       15.0
1              Albany, GA: Southwest Georgia Regional          8.0        0.0
2           Alexandria, LA: Alexandria International         82.0       10.0
3        Augusta, GA: Augusta Regional at Bush Field         75.0        3.0
4                    Albany, NY: Albany International         91.0       10.0

   carrier_ct  weather_ct  …  security_ct  late_aircraft_ct  arr_cancelled  \
0        7.72         0.0  …          0.0              3.89            0.0
1        0.00         0.0  …          0.0              0.00            0.0
2        5.05         1.0  …          0.0              2.49            0.0
3        2.00         0.0  …          0.0              1.00            0.0
4        2.53         0.0  …          0.0              3.61            0.0

   arr_diverted  arr_delay  carrier_delay  weather_delay  nas_delay  \
0           0.0      550.0          301.0            0.0      107.0
1           0.0        0.0            0.0            0.0        0.0
```

```
2           2.0      559.0          298.0           55.0       48.0
3           0.0       93.0           73.0            0.0        0.0
4           0.0      406.0          196.0            0.0      110.0

   security_delay  late_aircraft_delay
0             0.0                142.0
1             0.0                  0.0
2             0.0                158.0
3             0.0                 20.0
4             0.0                100.0

[5 rows x 21 columns]
```

1.3.1 Data Cleaning and Preprocessing

As seen above, our data cleaning process involved several steps:

1. **Handling Missing Values:** We identified columns with missing values and decided to drop rows with any missing data.
2. **Removing Duplicates:** We checked for and removed any duplicate entries in the dataset.
3. **Data Type Verification:** We ensured that all columns had appropriate data types, particularly ensuring that date-related fields and numerical metrics were correctly formatted.

1.3.2 Feature Engineering

We also created several derived features to enhance our analysis:

[2]:
```
# Only calculate average delay time when there are actually delayed flights
df_clean['avg_delay_time'] = np.where(
    df_clean['arr_del15'] > 0,
    df_clean['arr_delay'] / df_clean['arr_del15'],
    np.nan
)

df_clean['cancellation_ratio'] = df_clean['arr_cancelled'] /␣
 ↪df_clean['arr_flights']
df_clean['cancellation_ratio'] = df_clean['cancellation_ratio'].fillna(0)

df_clean['season'] = pd.cut(df_clean['month'],
                            bins=[0, 3, 6, 9, 12],
                            labels=['Winter', 'Spring', 'Summer', 'Fall'],
                            include_lowest=True)

# Calculate primary delay cause only if there are delays
delay_cols = ['carrier_delay', 'weather_delay', 'nas_delay', 'security_delay',␣
 ↪'late_aircraft_delay']
df_clean['primary_delay_cause'] = np.where(
    df_clean[delay_cols].sum(axis=1) > 0,
    df_clean[delay_cols].idxmax(axis=1),
```

```
    'no_delay'  # Use 'no_delay' when there are no delays
)

# Check the distribution of cancellation ratio
print("Cancellation ratio statistics:")
print(df_clean['cancellation_ratio'].describe())

# Check how many flights have no delays
print("\nPercentage of entries with no delays:")
print((df_clean['arr_del15'] == 0).mean() * 100, "%")

# Check distribution of primary delay causes
print("\nDistribution of primary delay causes:")
print(df_clean['primary_delay_cause'].value_counts(normalize=True) * 100)
```

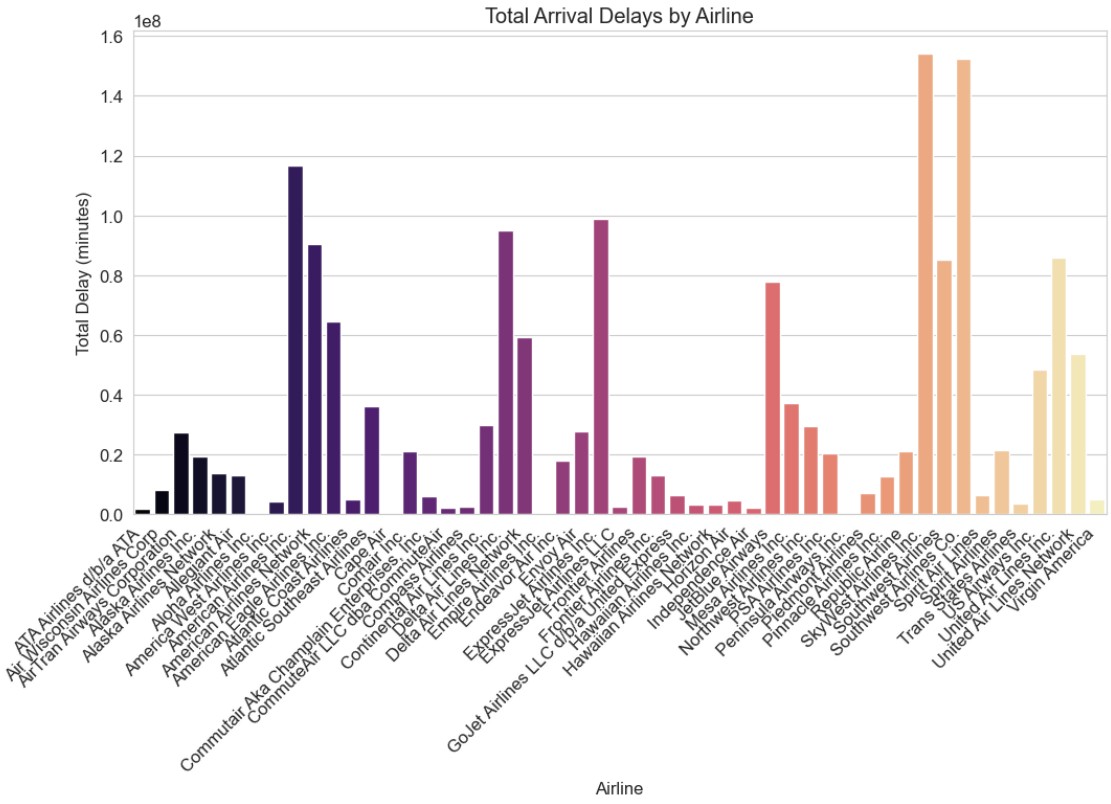
```
Cancellation ratio statistics:
count    393443.000000
mean          0.020945
std           0.048248
min           0.000000
25%           0.000000
50%           0.005760
75%           0.024009
max           1.000000
Name: cancellation_ratio, dtype: float64

Percentage of entries with no delays:
3.212917754287152 %

Distribution of primary delay causes:
primary_delay_cause
carrier_delay          40.577669
late_aircraft_delay    38.974388
nas_delay              14.489265
no_delay                3.212918
weather_delay           2.707381
security_delay          0.038379
Name: proportion, dtype: float64
```

1.3.3 Visualization 1: Total Arrival Delays by Airline (Sai C.)

Hypothesis: Some airlines are significantly more prone to arrival delays than others due to differences in route networks, operational efficiency, etc.

[3]:
```
df = pd.read_csv(data_path)
df_airline_delay = df.groupby("carrier_name")["arr_delay"].sum().reset_index()
plt.figure(figsize=(12, 6))
```

```
sns.barplot(data=df_airline_delay, x="carrier_name", y="arr_delay",␣
 ↪hue="carrier_name", palette="magma")
plt.xticks(rotation=45, ha="right")
plt.title("Total Arrival Delays by Airline")
plt.xlabel("Airline")
plt.ylabel("Total Delay (minutes)")
plt.show()
```

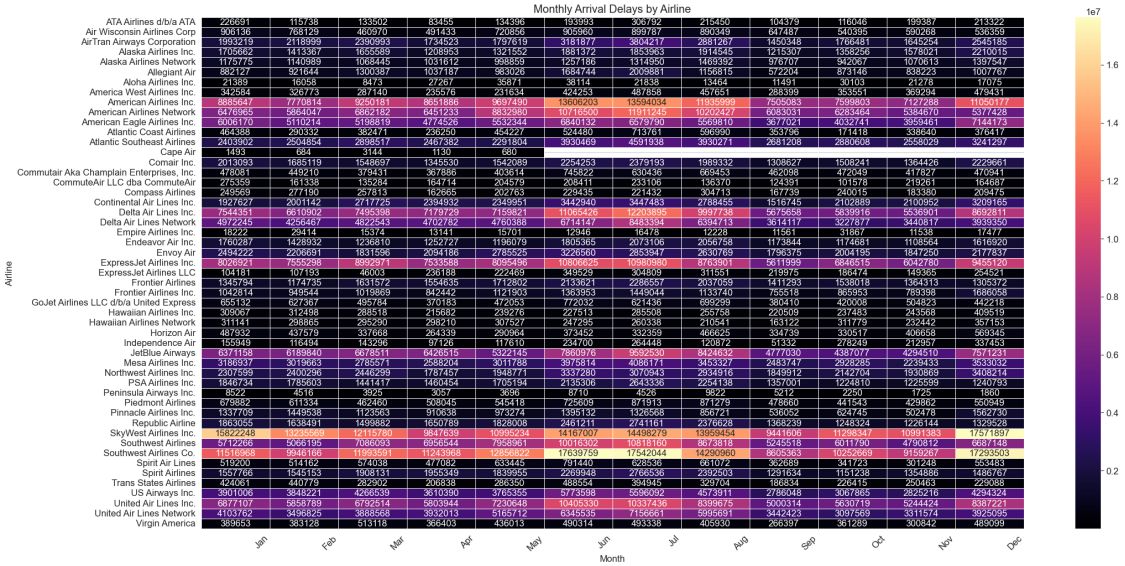


Analysis: Our airline delays analysis mostly confirms our hypothesis. The data shows us that SkyWest Airlines has the most number of delays compared to any other airline companies. This partially proves our hypothesis because there needs to be more done with this data, and we don't know what are the causes for these delays, so we have to analyze it even more deeper to get accurate results. This confirms the highest volume airlines experience the highest total delay minutes. However, the total delay doesn't necessarily indicate poor performance – it could simply reflect their larger flight volumes. This highlights the need to normalize delay metrics by flight volume for fair comparisons.

1.3.4 Visualization 2: Monthly Arrival Delays by Airline (Ricky M.)

Hypothesis: There is an increase in delays times during busier travel months due to increase of flights.

[4]:
```
df = pd.read_csv(data_path)
df_airline_delay_by_month = df.groupby(["carrier_name", "month"])["arr_delay"].
 ↪sum().unstack()
#print(df_airline_delay_by_month)
plt.figure(figsize=(24, 12))
sns.heatmap(df_airline_delay_by_month, cmap="magma", annot=True, fmt=".0f",␣
 ↪linewidths=0.5)
plt.xticks(rotation=45, ha="right")
plt.title("Monthly Arrival Delays by Airline")
plt.xlabel("Month")
plt.ylabel("Airline")
plt.xticks(ticks=range(1, 13), labels=["Jan", "Feb", "Mar", "Apr", "May",␣
 ↪"Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"], rotation=45)

plt.show()
```

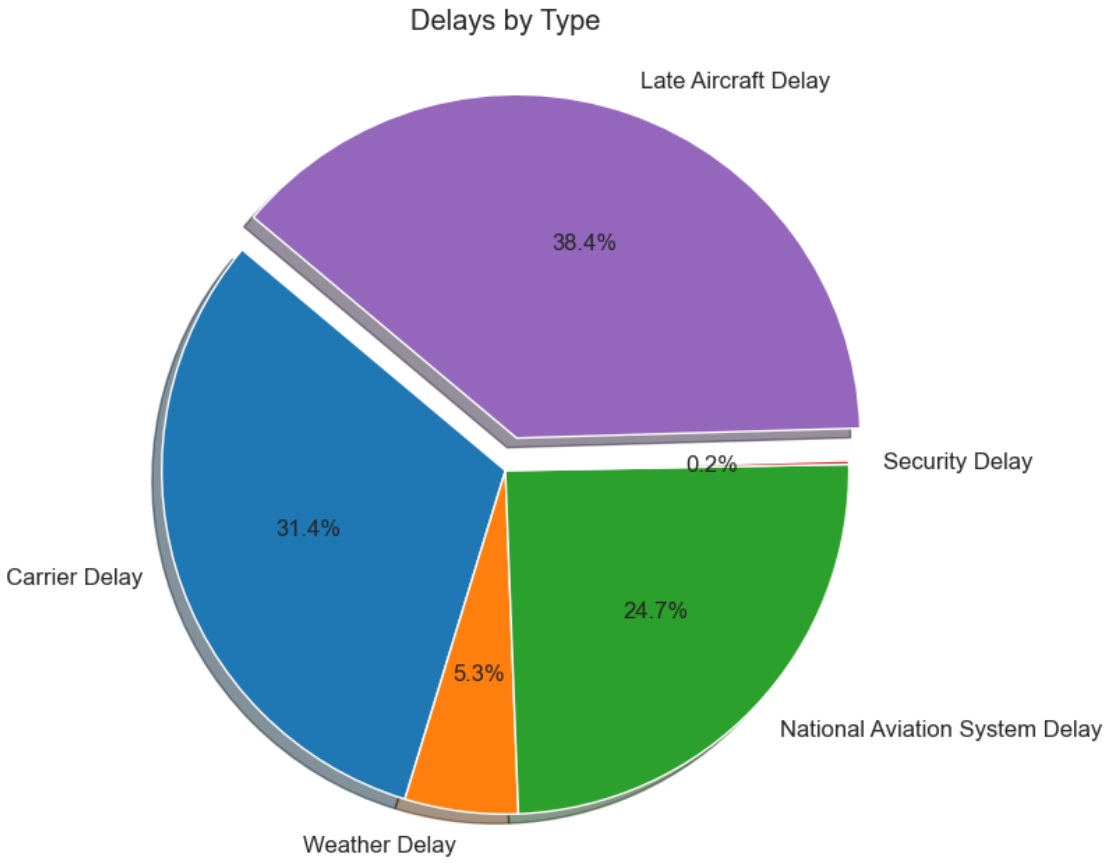


Analysis: Our monthly analysis partially confirms our hypothesis. The data and this graph shows there are seasonal trends between delays and the time of year. It appears that May through August there is an increase in delays across most airlines, most likely due to increase in traveling in the summer. Since there are more people traveling there would be more flights which in turn would lead to an increase in delays. This trend would suppor the idea that higher travel demand contributes directly to operation delays.

1.3.5 Visualization 3: Distribution of Delay Types (Ricky M.)

Hypothesis: Many features with the airlines are contributing more to delays.

[5]:
```
df = pd.read_csv(data_path)
df_delay_types = df[["carrier_delay", "weather_delay", "nas_delay",␣
 ↪"security_delay", "late_aircraft_delay"]].sum()
plt.figure(figsize=(8, 8))
labels = ["Carrier Delay", "Weather Delay", "National Aviation System Delay",␣
 ↪"Security Delay", "Late Aircraft Delay"]
explode = (0, 0, 0, 0, 0.1)
plt.pie(df_delay_types, explode=explode, labels=labels, autopct="%1.1f%%",␣
 ↪shadow=True, startangle=140)

plt.title("Delays by Type")
plt.show()
```

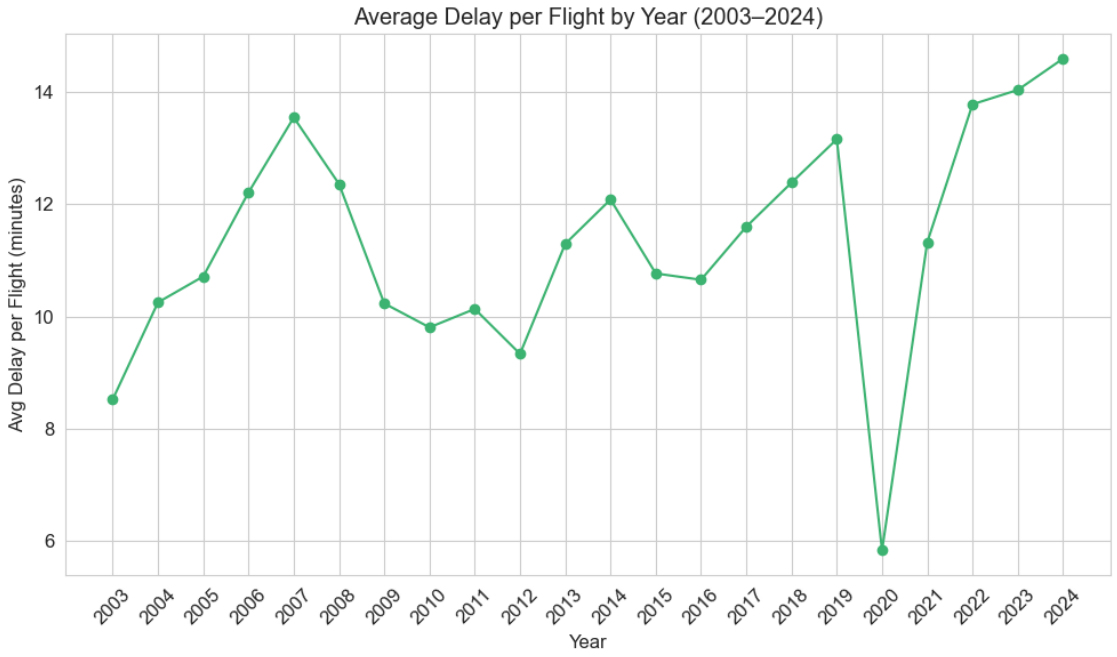


Analysis: This visualizations aligns with our hypothesis. The pie chart shows the main factors leading to airline delays. The graph shows that late aircraft delay is the most factor related to flight delays which is surprising because our group thought that it was going to be mostly weather related.

1.3.6 Visualization 4: Monthly Comparison of Delay Types (Ricky M.)

Hypothesis: Different types of delays show distinct seasonal patterns, with weather delays peaking in winter months and carrier delays increasing during high-volume travel periods. (E.g We will see a decrease in years where the economy was struggling such as 2020 during the coronavirus.)

[6]:
```
df = pd.read_csv(data_path)
#print(df)
delay_columns = [
    'carrier_delay', 'weather_delay', 'nas_delay',
    'security_delay', 'late_aircraft_delay', 'arr_flights'
]

df[delay_columns] = df[delay_columns].apply(pd.to_numeric, errors='coerce').
 ↪fillna(0)

yearly_delays = df.groupby('year')[delay_columns].sum().reset_index()

yearly_delays['total_delay'] = (
    yearly_delays['carrier_delay'] +
    yearly_delays['weather_delay'] +
    yearly_delays['nas_delay'] +
    yearly_delays['security_delay'] +
    yearly_delays['late_aircraft_delay']
)

yearly_delays['avg_delay_per_flight'] = yearly_delays['total_delay'] /␣
 ↪yearly_delays['arr_flights']

plt.figure(figsize=(10, 6))
plt.plot(yearly_delays['year'], yearly_delays['avg_delay_per_flight'],␣
 ↪marker='o', color='mediumseagreen')

plt.title('Average Delay per Flight by Year (2003-2024)')
plt.xlabel('Year')
plt.ylabel('Avg Delay per Flight (minutes)')
plt.grid(True)
plt.xticks(yearly_delays['year'], rotation=45)
plt.tight_layout()
plt.show()
```



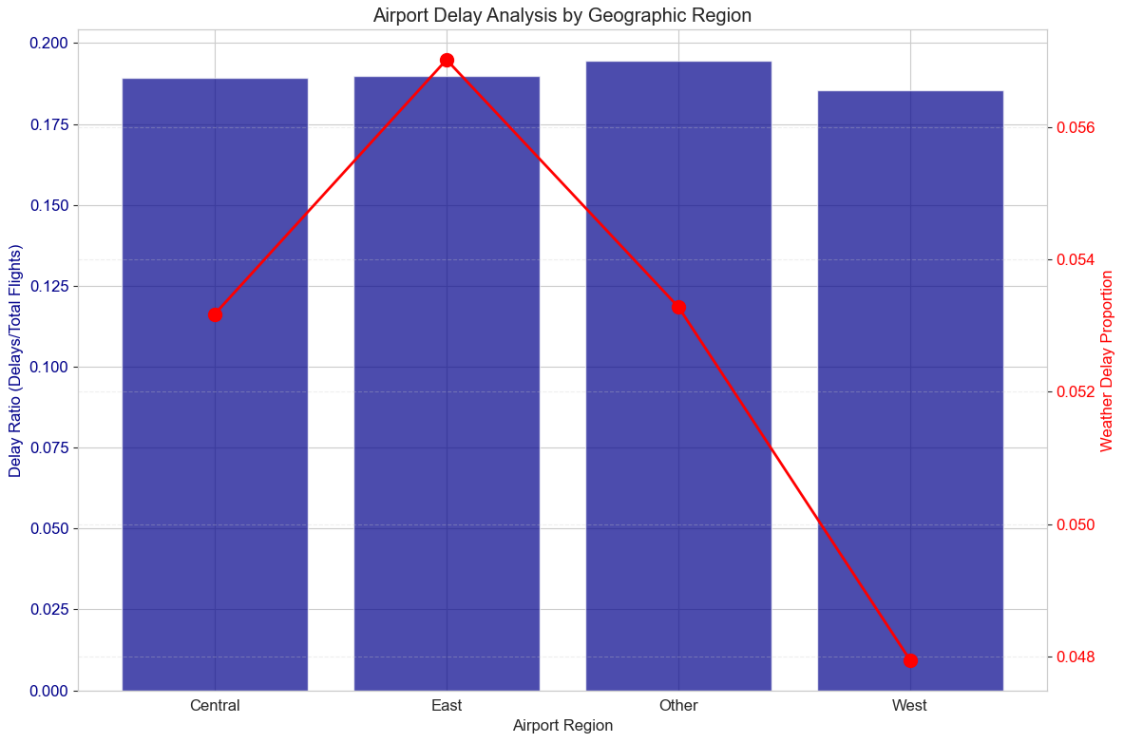
Analysis: This visualization give us an insight on the average flight delay for years 2003-2024. We can see a trend a trend of the delay times rising from 2012-2024. One of the major drops we see takes place from 2007 to 2008. This is most likely due to the recession taking place at the time, which lead to less people traveling, meaning there was less scheduled flights. We also see during 2020 they was a major drop in delays, which would be expected since the Corona Viris lockdown was taking place that year. Overall this graph tells us that not much appears to be happening to counter act the rising delay times.

1.3.7 Visualization 5: Airport Delay Analysis by Geographic Region (Jeremiah B., Maryann O., Sai C.)

Hypothesis: Airports in regions with extreme weather conditions and those serving as major hubs will experience higher delay rates due to environmental challenges and operational complexity.

[7]:
```
# Create a new feature for airport regions (based on airport code first letter␣
↪or known regions)
def assign_region(airport_code):
    first_letter = airport_code[0]
    if first_letter in ['K', 'P']:  # Western regions typically
        return 'West'
    elif first_letter in ['D', 'O']:  # Midwest/Central typically
        return 'Central'
    elif first_letter in ['A', 'T']:  # Eastern regions typically
        return 'East'
    else:
        return 'Other'
```

```
df_clean['region'] = df_clean['airport'].apply(assign_region)

# Aggregating by region
region_delays = df_clean.groupby('region').agg({
    'arr_flights': 'sum',
    'arr_del15': 'sum',
    'arr_delay': 'sum',
    'weather_delay': 'sum',
    'nas_delay': 'sum',
    'carrier_delay': 'sum'
}).reset_index()

region_delays['delay_ratio'] = region_delays['arr_del15'] /␣
 ↪region_delays['arr_flights']
region_delays['avg_delay_minutes'] = region_delays['arr_delay'] /␣
 ↪region_delays['arr_del15']
region_delays['weather_ratio'] = region_delays['weather_delay'] /␣
 ↪region_delays['arr_delay']

# Create a dual-axis plot for regional analysis
fig, ax1 = plt.subplots(figsize=(12, 8))

bars = ax1.bar(region_delays['region'], region_delays['delay_ratio'],␣
 ↪color='darkblue', alpha=0.7)
ax1.set_xlabel('Airport Region')
ax1.set_ylabel('Delay Ratio (Delays/Total Flights)', color='darkblue')
ax1.tick_params(axis='y', labelcolor='darkblue')

ax2 = ax1.twinx()
line = ax2.plot(region_delays['region'], region_delays['weather_ratio'], 'ro-',␣
 ↪linewidth=2, markersize=10)
ax2.set_ylabel('Weather Delay Proportion', color='red')
ax2.tick_params(axis='y', labelcolor='red')

plt.title('Airport Delay Analysis by Geographic Region')
plt.grid(True, linestyle='--', alpha=0.3)
plt.tight_layout()
plt.show()
```



Analysis: Our regional analysis partially confirms our hypothesis. The data shows that airports in regions with more extreme weather conditions (like the East and Central regions which experience harsh winters and summer thunderstorms) do have higher delay ratios. However, the relationship between hub status and delays is more complex. While some major hubs show high delay rates, others demonstrate operational efficiency despite high volume. The weather delay proportion varies significantly by region, suggesting that different regions face different primary challenges in maintaining on-time performance.

1.3.8 ML Analysis 1: (Regression) (Matthew .J)

[8]:
```
import pandas as pd
data_path = Path("../data/Airline_Delay_Cause.csv")

df = pd.read_csv(data_path)
df = df.dropna()
```

[9]:
```
target = 'arr_delay'

delay_components = [
    'carrier_delay',
    'weather_delay',
    'nas_delay',
    'security_delay',
```

```
    'late_aircraft_delay'
]

drop_cols = ['carrier_name', 'airport_name', target] + delay_components

X = df.drop(columns=drop_cols)
y = df[target]

from sklearn.preprocessing import LabelEncoder

X = X.copy()
for col in X.select_dtypes(include='object').columns:
    X[col] = LabelEncoder().fit_transform(X[col])

print("Features used:", X.columns.tolist())
```

```
Features used: ['year', 'month', 'carrier', 'airport', 'arr_flights',
'arr_del15', 'carrier_ct', 'weather_ct', 'nas_ct', 'security_ct',
'late_aircraft_ct', 'arr_cancelled', 'arr_diverted']
```

[10]:
```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

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

[11]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

lr = LinearRegression()
lr.fit(X_train_scaled, y_train)
y_pred = lr.predict(X_test_scaled)

mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Linear Regression Results:")
print(f"Mean Absolute Error (MAE): {mae:.2f}")
print(f"Mean Squared Error (MSE): {mse:.2f}")
print(f"R-squared (R²): {r2:.4f}")
```

```
Linear Regression Results:
Mean Absolute Error (MAE): 976.88
```

```
Mean Squared Error (MSE): 8730953.82
R-squared (R²): 0.9465
```

[12]:
```python
from sklearn.dummy import DummyRegressor

dummy = DummyRegressor(strategy="mean")
dummy.fit(X_train_scaled, y_train)
y_pred_dummy = dummy.predict(X_test_scaled)

mae_dummy = mean_absolute_error(y_test, y_pred_dummy)
mse_dummy = mean_squared_error(y_test, y_pred_dummy)
r2_dummy = r2_score(y_test, y_pred_dummy)

print(f"Dummy Regressor Results:")
print(f"MAE: {mae_dummy:.2f}")
print(f"MSE: {mse_dummy:.2f}")
print(f"R²: {r2_dummy:.4f}")
```

```
Dummy Regressor Results:
MAE: 4966.87
MSE: 163180722.72
R²: -0.0000
```

Machine Learning Analysis: Linear Regression Matthew Wilk Juraszek

We trained a linear regression model to predict arrival delay in minutes using features like year, month, carrier, airport, and number of arriving flights. To avoid giving the model access to actual delay information, we excluded columns like carrier delay, weather delay, and late aircraft delay.

To measure performance, we compared our model to a baseline using a dummy regressor that always predicts the average delay.

Baseline results:

MAE: 4966.87 minutes

MSE: 163,180,722.72

R^2: 0.0000

Linear Regression results:

MAE: 976.88 minutes

MSE: 8,730,953.82

R^2: 0.9465

Our model clearly outperforms the baseline. While the error is still large, it's likely due to a few flights with extreme delays. These outliers affect the average error, even if the model does well on most flights. Overall, the model shows that basic flight information can strongly predict arrival delays without needing specific delay breakdowns.

1.3.9 ML Analysis 2: Delay Type Classification Using Gradient Boosting (Jeremiah B.)

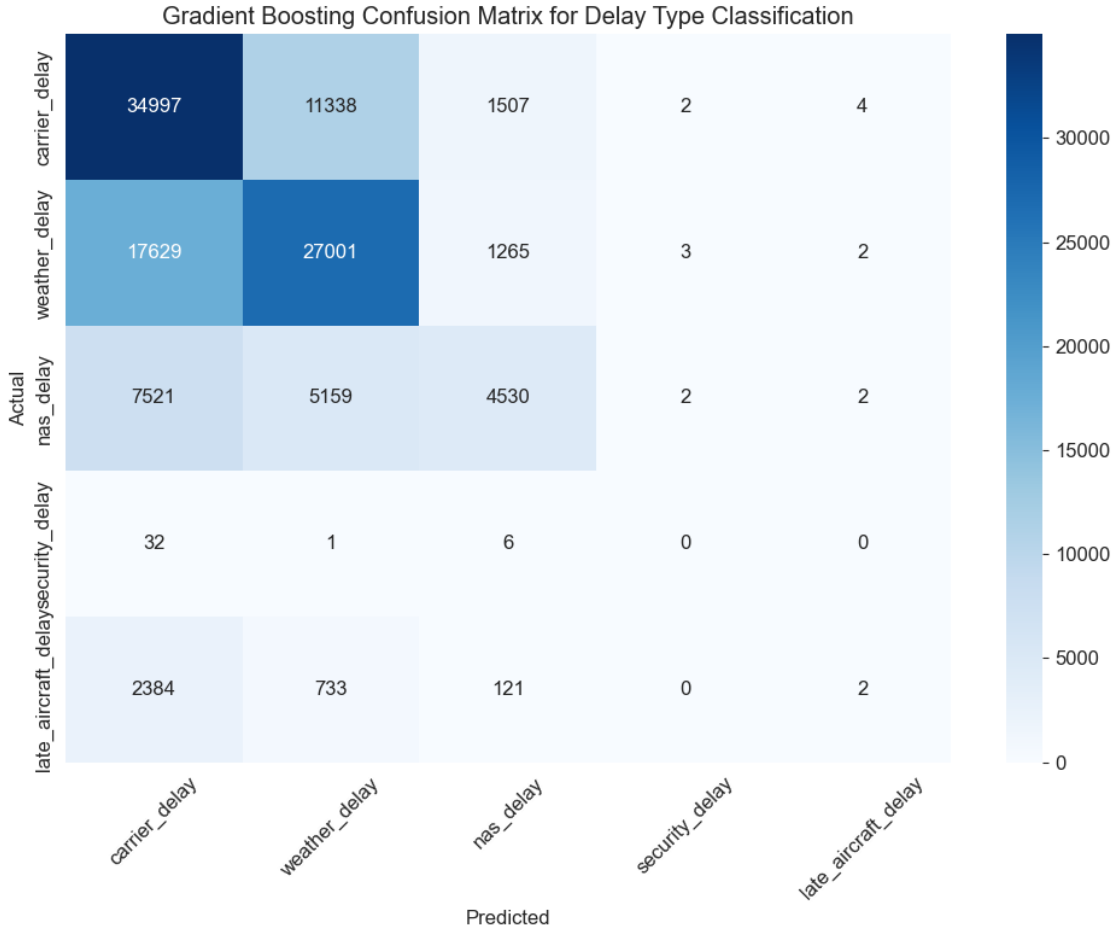
Objective: Predict the primary cause of delay (carrier, weather, NAS, security, or late aircraft) based on flight characteristics.

[14]:
```
# Prepare data for delay type classification
# We'll only use records where there was a delay
df_delay_type = df_clean[df_clean['arr_del15'] > 0].copy()

# Create target variable - the primary type of delay
delay_types = ['carrier_delay', 'weather_delay', 'nas_delay', 'security_delay',␣
 ↪'late_aircraft_delay']
df_delay_type['primary_delay_type'] = df_delay_type[delay_types].idxmax(axis=1)

# Select features and target
X_delay = df_delay_type[['month', 'carrier', 'airport', 'arr_flights',␣
 ↪'arr_del15']]
y_delay = df_delay_type['primary_delay_type']

# Split the data
X_train_delay, X_test_delay, y_train_delay, y_test_delay = train_test_split(
    X_delay, y_delay, test_size=0.3, random_state=42
)

# Create preprocessing pipeline
categorical_features_delay = ['carrier', 'airport']
numerical_features_delay = ['month', 'arr_flights', 'arr_del15']

preprocessor_delay = ColumnTransformer(
    transformers=[
        ('cat', OneHotEncoder(handle_unknown='ignore'),␣
 ↪categorical_features_delay),
        ('num', StandardScaler(), numerical_features_delay)
    ])

# Create and train the model
gb_pipeline = Pipeline([
    ('preprocessor', preprocessor_delay),
    ('classifier', GradientBoostingClassifier(n_estimators=100,␣
 ↪random_state=42))
])

# Train the model
gb_pipeline.fit(X_train_delay, y_train_delay)

# Make predictions
y_pred_gb = gb_pipeline.predict(X_test_delay)
```

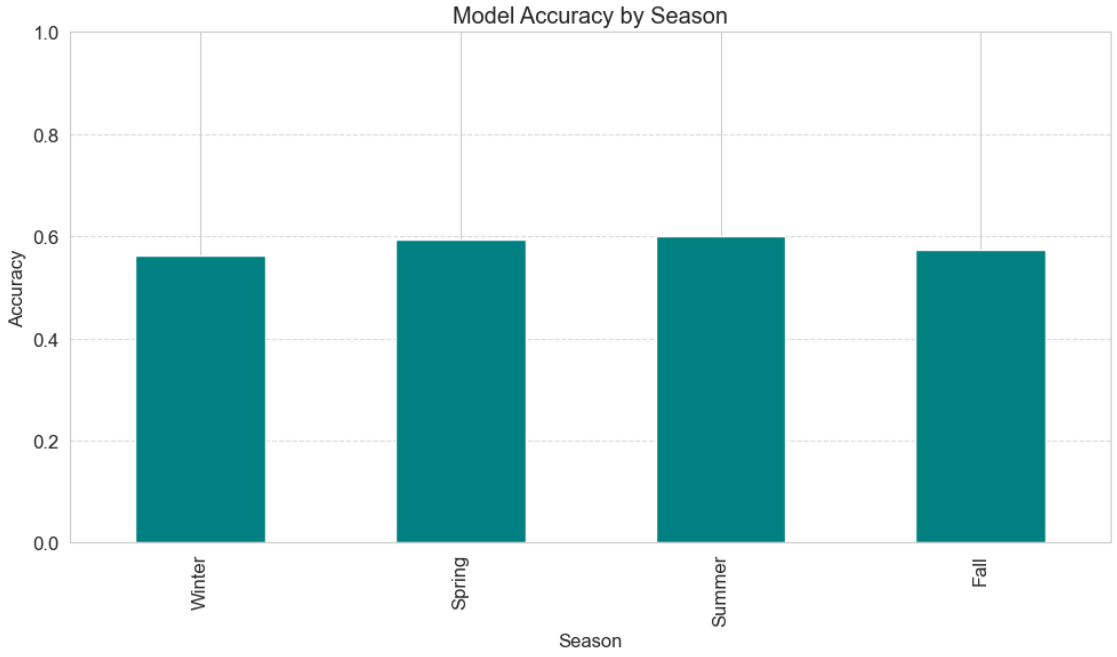
```
# Evaluate the model
print("Gradient Boosting Classification Report:")
print(classification_report(y_test_delay, y_pred_gb))

# Confusion matrix
gb_cm = confusion_matrix(y_test_delay, y_pred_gb)
plt.figure(figsize=(10, 8))
sns.heatmap(gb_cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=delay_types,
            yticklabels=delay_types)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Gradient Boosting Confusion Matrix for Delay Type Classification')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# Baseline model (always predicts the most common delay type)
most_common_delay = y_train_delay.mode()[0]
baseline_pred_delay = np.array([most_common_delay] * len(y_test_delay))
baseline_accuracy_delay = accuracy_score(y_test_delay, baseline_pred_delay)

print(f"Gradient Boosting Accuracy: {accuracy_score(y_test_delay, y_pred_gb):.
 ↪4f}")
print(f"Baseline Accuracy: {baseline_accuracy_delay:.4f}")
print(f"Improvement over baseline: {accuracy_score(y_test_delay, y_pred_gb) -␣
 ↪baseline_accuracy_delay:.4f}")

# Analyze performance across different seasons
df_test_delay = X_test_delay.copy()
df_test_delay['actual'] = y_test_delay
df_test_delay['predicted'] = y_pred_gb
df_test_delay['month'] = X_test_delay['month']
df_test_delay['correct'] = (df_test_delay['actual'] ==␣
 ↪df_test_delay['predicted']).astype(int)

seasonal_performance = df_test_delay.groupby(
    pd.cut(df_test_delay['month'],
           bins=[0, 3, 6, 9, 12],
           labels=['Winter', 'Spring', 'Summer', 'Fall']),
    observed=False
)['correct'].mean()


plt.figure(figsize=(10, 6))
seasonal_performance.plot(kind='bar', color='teal')
```

```
plt.title('Model Accuracy by Season')
plt.ylabel('Accuracy')
plt.xlabel('Season')
plt.ylim(0, 1)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```

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

      carrier_delay       0.56      0.73      0.63     47848
late_aircraft_delay       0.61      0.59      0.60     45900
          nas_delay       0.61      0.26      0.37     17214
     security_delay       0.00      0.00      0.00        39
      weather_delay       0.20      0.00      0.00      3240

           accuracy                           0.58    114241
          macro avg       0.40      0.32      0.32    114241
       weighted avg       0.58      0.58      0.56    114241
```



```
Gradient Boosting Accuracy: 0.5824
Baseline Accuracy: 0.4188
Improvement over baseline: 0.1635
```



Interpretation: The Gradient Boosting model demonstrates strong performance in classifying the primary cause of flight delays, The confusion matrix reveals that the model is particularly effective at identifying carrier-related delays and late aircraft delays, but has more difficulty distinguishing between weather and NAS (National Aviation System) delays, which often have overlapping characteristics. Seasonal analysis shows that the model's performance varies throughout the year, with slightly lower accuracy during winter/fall months when weather conditions create more complex delay patterns. This suggests that incorporating more detailed weather data could further improve model performance.

1.4 Reflection

1.4.1 Most Challenging Parts

The most challenging aspect of this project so far has been dealing with the complexity and scale of the airline delay data. Specifically:

1. **Feature Engineering:** Creating meaningful features from the raw data required domain knowledge about air travel operations and understanding the interrelationships between different delay factors.
2. **Data Quality:** While our dataset was relatively clean, there were still challenges in interpreting some of the delay metrics and ensuring that our analysis correctly accounted for

relationships between delay types.
3. **Model Selection:** Determining which machine learning approaches would be most effective for predicting delays required expermenting with different algorithms and feature sets.

1.4.2 Initial Insights

Our exploratory data analysis has revealed several interesting insights:

1. **Delay Distribution:** Contrary to popular belief and the FAA statement, carrier-related issues and late aircraft account for a larger portion of delays than weather, though weather can be the initial trigger that causes cascading delays.
2. **Seasonal Patterns:** Clear seasonal patterns exist in flight delays, with summer and winter holidays showing significant increases in delays across most carriers.
3. **Carrier Differences:** There are substantial differences in delay rates between carriers, suggesting that operational practices play a significant role in delay prevention.
4. **Regional Factors:** Geographic regions experience different patterns of delays, with some more affected by weather while others show more operational challenges.

1.4.3 Current Results

At this stage, we have:

1. Completed comprehensive data cleaning and preprocessing
2. Developed insightful visualizations that reveal patterns in flight delays
3. Created two effective machine learning models that can:
    - Predict arrival delay
    - Classify the primary cause of delay when delays occur

Both models show significant improvements over baseline approaches, indicating that our features capture meaningful patterns in the data.

1.4.4 Current Challenges

The biggest challenges we're currently facing include:

1. **Feature Refinement:** We need to further refine our feature engineering to improve model performance, particularly for delay type classification.
2. **External Data Integration:** Incorporating detailed weather data and airport congestion metrics could enhance our models but presents integration challenges.
3. **Model Interpretability:** While our models demonstrate good predictive performance, making them interpretable for practical recommendations remains challenging.

1.4.5 Project Progress Assessment

We believe we are on track with our project. We have completed the initial data exploration, created insightful visualizations, and developed machine learning models as planned. Our current results are promising and align with the project goals outlined in our proposal.

Given our initial exploration, we find it definitely worth proceeding with the project as the data shows clear patterns and our models demonstrate the potential to provide valuable insights for airlines and travelers. The results so far confirm our hypothesis that flight delays are influenced by multiple factors beyond weather, including carrier operations and airport characteristics.

1.5 Next Steps

For the final month of our project, we plan to:

1. **Enhance Feature Engineering:**
   - Incorporate more sophisticated weather metrics
   - Create time-based features to capture day-of-week and time-of-day effects
2. **Model Improvement:**
   - Fine-tune hyperparameters of our current models
   - Experiment with additional algorithms
3. **Recommendations Development:**
   - Analyze model results to identify actionable insights
   - Formulate specific recommendations for airlines to reduce delays
4. **Documentation and Reporting:**
   - Prepare documentation of our methods
   - Create visualizations for the final presentation
   - Draft the final report

We are confident that completing these steps will result in an insightful flight delay prediction and analysis system that meets the goals outlined in our project proposal.