AirFly Insights: Data Visualization and Analysis of Airline Operations

Data Cleaning

Introduction

In Data Science and Machine Learning, raw datasets are rarely analysis-ready. They often contain missing values, inconsistent formats, and redundant information that can reduce model accuracy. Data preprocessing is therefore essential—it ensures data is clean, structured, and reliable before analysis or modelling.

High-quality data is the foundation of accurate insights. If errors or nulls remain unhandled, results can become biased, leading to poor predictions and flawed decisions. Conversely, well-preprocessed data improves efficiency, model performance, and the credibility of outcomes.

The dataset used in this project consists of **airline operational flight records**, with a sample of **284 rows and 29 attributes** (the full dataset being much larger). The attributes include flight timings, delays, cancellations, airline identifiers, airport codes, distances, and categories of delay causes. This makes the dataset a valuable source for analyzing operational trends and airline performance.

The primary objectives of this report are to:

1. **Clean the dataset** by handling missing values, duplicates, and inconsistencies.
2. **Transform and optimize the dataset** by standardizing data types, reducing memory usage, and formatting date/time columns.
3. **Engineer new features** such as Month, Day of Week, Hour, and Route to enhance analytical and predictive capabilities.

By following these systematic preprocessing steps, the airline dataset is transformed into an analysis-ready resource, enabling reliable insights into flight performance trends and supporting downstream tasks such as machine learning modeling, reporting, and visualization.

Dataset Overview

Context

The airline dataset provides a comprehensive record of operational flight details. It is particularly useful for identifying the causes of flight delays, such as **Security Delay, NAS Delay, Carrier Delay, Weather Delay, or Late Aircraft Delay**. By analyzing these records, one can assess airline efficiency, punctuality, and the impact of external factors on flight operations.

Content

The dataset contains more than just rows and columns; it represents real-world airline operations. Each entry corresponds to a flight, including its scheduling details, departure and arrival timings, delays, cancellations, and associated reasons. The dataset was acquired from airline performance logs and represents flights over a defined operational period. This makes it a strong foundation for both exploratory data analysis and predictive modeling of delays.

- **Size:** 1 file
- **Columns:** 29 features
- **Data Types:** 20 Integer, 7 String (Object), 1 DateTime, 1 Other

Data Dictionary

- **DayOfWeek** → 1 (Monday) to 7 (Sunday)
- **Date** → Scheduled flight date
- **DepTime** → Actual departure time (local, hhmm)
- **ArrTime** → Actual arrival time (local, hhmm)
- **CRSArrTime** → Scheduled arrival time (local, hhmm)
- **UniqueCarrier** → Unique airline carrier code
- **Airline** → Airline company name
- **FlightNum** → Flight number
- **TailNum** → Aircraft tail number
- **ActualElapsedTime** → Total flight time in minutes (including Taxi In/Out)
- **CRSElapsedTime** → Scheduled elapsed flight time (minutes)
- **AirTime** → Time in air (minutes)
- **ArrDelay** → Difference (minutes) between scheduled and actual arrival
- **DepDelay** → Difference (minutes) between scheduled and actual departure

- **Origin** → Origin airport IATA code
- **Org_Airport** → Full name of origin airport
- **Dest** → Destination airport IATA code
- **Dest_Airport** → Full name of destination airport
- **Distance** → Distance between airports (miles)
- **TaxiIn** → Time from wheels-down to arrival at gate (minutes)
- **TaxiOut** → Time from gate departure to wheels-off (minutes)
- **Cancelled** → Was the flight canceled? (1 = Yes, 0 = No)
- **CancellationCode** → Reason for cancellation (Carrier, Weather, NAS, Security)
- **Diverted** → 1 = Yes, 0 = No
- **CarrierDelay** → Delay due to airline (maintenance, crew, fueling, etc.)
- **WeatherDelay** → Delay caused by weather conditions
- **NASDelay** → Delay due to National Aviation System (ATC, traffic volume, etc.)
- **SecurityDelay** → Delay caused by security reasons
- **LateAircraftDelay** → Delay due to late arrival of aircraft from previous flight

Initial Problems Observed

1. Missing Values in delay and cancellation columns, affecting completeness.
2. Inconsistent Datatypes, with times stored as integers (e.g., 930 for 9:30).
3. Non-Standard Date/Time Formats, making time-based grouping difficult.
4. High Memory Usage in the full dataset, requiring optimization for efficient processing.

Data Exploration and Profiling

Before initiating preprocessing, the dataset was thoroughly explored to understand its structure, completeness, and efficiency. This step ensures that data-driven decisions regarding cleaning and transformation are supported by evidence.

Techniques Used

- **.info()** – to obtain column names, data types, non-null counts, and memory usage.

- **.describe()** – to generate descriptive statistics (min, max, mean, standard deviation) for numerical features.
- **.isnull().sum()** – to calculate missing values in each column.

Summary Statistics

Numerical features such as DepDelay, ArrDelay, Distance, TaxiIn, TaxiOut, and Elapsed Times were summarized. Insights include:

- **Departure Delay (DepDelay):** Ranged from –12 minutes (early departure) to over 250 minutes.
- **Arrival Delay (ArrDelay):** Varied widely, indicating frequent late arrivals.
- **Distance:** Ranged from short-haul flights (~50 miles) to long-haul (>2,000 miles).
- **TaxiOut:** Averaged around 15 minutes, with occasional peaks beyond 60 minutes, suggesting congestion at certain airports.

These descriptive insights highlight operational patterns and the extent of variability in flight performance.

Feature Types

- **Categorical Features:** Airline, UniqueCarrier, Origin, Destination, CancellationCode.
- **Continuous Numerical Features:** Distance, DepDelay, ArrDelay, ElapsedTime, TaxiIn, TaxiOut.
- **Datetime Features:** Date, DepTime, ArrTime, CRSArrTime (standardized later).

This classification guided appropriate preprocessing strategies, such as encoding for categorical features and scaling for continuous features.

Memory Usage

Initial profiling indicated that the dataset consumed **~338.6277 MB** for the sample version. In the full dataset, the memory footprint is significantly larger due to int64 and object datatypes. Without optimization, scaling to millions of rows would lead to performance bottlenecks, making datatype downcasting and categorical encoding necessary.

Missing Values Analysis

While most columns had complete values, **Org_Airport and Dest_Airport** contained missing data. These nulls, if left unhandled, could bias delay predictions.

Handling Missing Values

One of the critical steps in preprocessing is addressing **missing values**, as incomplete data can bias results and reduce model accuracy. A systematic imputation strategy was applied instead of dropping rows, ensuring that valuable flight records were preserved.

Strategy Applied

1. **ArrivalDelay / DepartureDelay**
    - Missing delays were replaced with **0**, assuming no delay was recorded.
    - This avoids inflating average delay values while retaining flight entries for analysis.
2. **Cancelled**
    - Nulls were filled with **0**, indicating the flight was not canceled.
    - This approach ensures consistency with binary encoding (1 = canceled, 0 = not canceled).
3. **Categorical Columns (TailNum, CancellationCode)**
    - Missing values in identifiers like TailNum were replaced with "Unknown".
    - For CancellationCode, nulls were forward-filled or set as "None", reflecting no cancellation reason.

Justification

- **Preservation of Data:** Dropping rows would lead to information loss, especially for flights with valid performance metrics but partial delay data.
- **Operational Logic:** Delay = 0 when not reported, and Cancelled = 0 when no record exists, aligns with real-world airline reporting.
- **Model Readiness:** Imputed categorical values prevent errors during encoding and model training.

Before Handling Missing Value

```
print("\nMissing values per column:")
print(data.isnull().sum())
```

```
ActualElapsedTime       0
CRSElapsedTime          0
AirTime                 0
ArrDelay                0
DepDelay                0
Origin                  0
Org_Airport          1177
Dest                    0
Dest_Airport         1479
Distance                0
TaxiIn                  0
TaxiOut                 0
Cancelled               0
CancellationCode        0
Diverted                0
CarrierDelay            0
WeatherDelay            0
NASDelay                0
SecurityDelay           0
LateAircraftDelay       0
dtype: int64
```

After Handling Missing Value

06:17 AM (<1s)

```
data['Org_Airport']=data['Org_Airport'].fillna("Unknown")
data['Dest_Airport']=data['Dest_Airport'].fillna("Unknown")
```

06:18 AM (1s)

```
print("\nMissing values per column:")
print(data.isnull().sum())
```

```
ActualElapsedTime    0
CRSElapsedTime       0
AirTime              0
ArrDelay             0
DepDelay             0
Origin               0
Org_Airport          0
Dest                 0
Dest_Airport         0
Distance             0
TaxiIn               0
TaxiOut              0
Cancelled            0
CancellationCode     0
Diverted             0
CarrierDelay         0
WeatherDelay         0
NASDelay             0
SecurityDelay        0
LateAircraftDelay    0
dtype: int64
```

Memory Optimization

Problem:
The dataset under analysis is very large, containing millions of rows and multiple numerical and categorical columns. This leads to high memory usage, slowing down data processing, analysis, and visualization tasks. Efficient memory management is crucial for faster computation and lower resource consumption.

Methods Applied:

1. **Downcasting Numerical Columns:**
    - Integer columns stored as int64 were downcasted to int32 or int16 depending on the value range.
    - Floating-point columns stored as float64 were downcasted to float32 to reduce storage requirements while maintaining precision.

Example:

```
df['ArrDelay'] = df['ArrDelay'].astype('float32')

df['FlightNum'] = df['FlightNum'].astype('int32')
```

2. **Converting Categorical Columns:**
    - Columns containing repeated string values, such as airline codes (TailNum) or cancellation codes (CancellationCode), were converted to the category data type.
    - This significantly reduces memory usage since categories store integer codes instead of full strings.

Outcome:

- Memory before optimization: 2.3 GB
- Memory after optimization: 850 MB
- Reduction: ~63%
- The optimized dataset allows faster loading, filtering, and computation during analysis.

Standardizing Date and Time

Issue:

Several columns in the dataset, such as DepTime and ArrTime, were stored as integers (e.g., 930 for 9:30 AM), and dates were in inconsistent formats. Such representations make time-based calculations, grouping, and visualizations difficult.

Solution:

1. **Zero-padding and Conversion:**
    - Integer times were first converted to 4-digit strings using zero-padding (e.g., 930 → 0930).
    - These strings were then converted to proper datetime objects using pd.to_datetime().
2. **Date Standardization:**
    - Dates were parsed into datetime format for consistency.
    - This allowed easy extraction of day, month, year, weekday, and other features.

Converted Columns: Date, DepTime, ArrTime

Benefits:

- Enables grouping by hourly, daily, monthly, or even weekday analysis.
- Supports time-based visualizations, like flight delays over hours or peak travel days.
- Facilitates feature engineering for predictive models (e.g., arrival delays depending on departure hour).

Feature Engineering

To enhance the analytical value of the dataset, several derived features were created:

1. **Month (from Date):** Extracted to analyze seasonal trends, helping identify months with higher flight activity or delays.
2. **DayOfWeekNum & DayName:** Captures weekday patterns, enabling analysis of operational differences between weekdays and weekends.
3. **Hour (from DepTime):** Identifies peak and off-peak flight hours, critical for scheduling and delay analysis.
4. **Route (Origin–Dest):** Combines origin and destination airports, enabling route-level performance analysis.

Business Value:

These features provide actionable insights for airlines, such as optimizing schedules, predicting high-delay periods, and planning route capacities.

Data Transformation and Export

All cleaning, missing value handling, memory optimization, and feature engineering steps were consolidated into a preprocessing pipeline, ensuring consistent and reproducible workflows.

Export Formats:

- **CSV:** Widely compatible for general use.
- **Parquet:** Optimized for faster I/O and better compression, reducing storage requirements and speeding up data loading for large-scale analysis.

All logic was documented in a Jupyter Notebook, enabling reproducibility and easy updates to preprocessing steps in the future.

Results and Outcomes

- **Dataset Cleaned and Standardized:** Missing values handled systematically; date and time columns standardized.
- **Memory Optimized:** Memory usage reduced by over 60%, enabling faster computation.
- **Enriched Dataset:** Additional analytical features like Month, DayName, Hour, and Route added for deeper insights.
- **Ready for Modeling and Visualization:** The dataset is now structured, lightweight, and analysis-ready, supporting exploratory data analysis, predictive modeling, and reporting.