Documentation:

Setting up and Executing the Pipeline

Name: Visesh Agarwal
Email: visesh.agarwal@mca.christuniversity.in
Reg No.: 2347164

Folder Structure Overview

Akasa-Air-Analytics-Assignment project repository is organized as follows:

1. `datasets/`:
   - `aviation_data_cleaned.csv`: The cleaned dataset post data cleaning and preprocessing steps.
   - `normalized_data.csv`: The normalized dataset ready for analysis and visualization.
2. `reports/`:
   - `average_delay_airline.png`: Bar chart showing the average delay time per airline.
   - `average_delay_hour.png`: Line plot illustrating the average delay per departure hour.
   - `delay_distribution.png`: Histogram displaying the distribution of delays across flights.
   - `departure_vs_delay.png`: Scatter plot analyzing the relationship between departure times and delays.
   - `aviation_report.html`: A comprehensive HTML report summarizing all findings and visualizations.
3. **Config Files**:
   - `.env.example`: Template for the `.env` file that stores database credentials and environment variables.
   - `.gitignore`: Specifies which files and directories Git should ignore (e.g., `.env`, `__pycache__`).
4. **Project Documents**:
   - `AkasaAir-DataEngineerAnalytics-Task1.pdf`: The project task document outlining the assignment requirements.
   - `Assignment.ipynb`: The Jupyter Notebook that performs data loading, cleaning, analysis, and visualizations.

- `Assignment.pdf`: Exported PDF version of the notebook.
- `aviation_data.csv`: The raw aviation dataset before cleaning and transformation.
- `generate_report.py`: Script to generate an HTML report after running the analysis.
- `README.md`: The README file providing a general overview of the project and setup instructions.
- `requirements.txt`: A list of Python dependencies for the project.
- **`pipeline.py`**: A Python script that automates the entire project workflow from setup to report generation.

Libraries and Frameworks Used

The required packages (listed in `requirements.txt`) include:

- **`pandas`**: For data manipulation, cleaning, and analysis.
- **`python-dotenv`**: To securely manage environment variables, such as database credentials.
- **`matplotlib`** & **`seaborn`**: For generating various visualizations, including histograms, bar charts, and scatter plots.
- **`sqlalchemy`**: An SQL toolkit and ORM library used for communicating with the MySQL database.
- **`pymysql`**: Facilitates the connection between Python and the MySQL database.
- **`scipy`**: Provides essential tools for scientific and statistical analysis.

Pipeline Overview: `pipeline.py`

The `pipeline.py` script automates the entire workflow of setting up and executing the project, making it easier to handle complex steps without manual intervention.

Automated Setup and Execution (Recommended)

For a streamlined process, the `pipeline.py` script automates the setup, execution, and report generation.

Steps for Automated Pipeline:

1. **Clone the Repository**:

First, clone the project repository from GitHub:

```
git clone https://github.com/viseshagarwal/Akasa-Air-Analytics-Assignment.git
```

2. **Navigate to the Project Directory**:

   Navigate to the project folder where all files are located:

   ```
   cd Akasa-Air-Analytics-Assignment
   ```

3. **Run the `pipeline.py` Script**:

   Now, run the `pipeline.py` script to automate the entire process:

   ```
   python pipeline.py
   ```

What the `pipeline.py` Script Does:

- **Virtual Environment Creation**: It creates a virtual environment to isolate the project dependencies from your system.
- **Dependency Installation**: Automatically installs all the necessary Python libraries using `requirements.txt`.
- **Environment Setup**: Prompts for MySQL database credentials, generates a `.env` file, and configures the database connection.
- **Execution of Jupyter Notebook**: Attempts to run the `Assignment.ipynb` notebook. If Jupyter is not installed, it will open VS Code, allowing manual notebook execution.
- **Report Generation**: After running the notebook, the script generates an HTML report (`aviation_report.html`) in the `reports/` directory and opens it in your default web browser.

Outcome:

After running the `pipeline.py` script:

- The virtual environment will be created, and all dependencies will be installed.
- The `Assignment.ipynb` notebook will be executed, and visualizations will be saved as `.png` files.
- A comprehensive HTML report (`aviation_report.html`) will be automatically generated and opened in your browser, summarizing the analysis and visualizations.

Manual Setup and Execution

In case the `pipeline.py` script fails or if you prefer to manually run the project, follow these steps:

Step 1: Set Up a Virtual Environment

1. **Navigate to the Project Directory**:

   Navigate to the project folder:

   ```
   cd Akasa-Air-Analytics-Assignment
   ```

2. **Create a Virtual Environment**:

   Create a virtual environment to manage project dependencies:

   ```
   python -m venv venv
   ```

3. **Activate the Virtual Environment**:
   - On **Windows**:

     ```
     venv\Scripts\activate
     ```

   - On **macOS/Linux**:

     ```
     source venv/bin/activate
     ```

Step 2: Install Dependencies

Once the virtual environment is activated, install all the required Python libraries:

```
pip install -r requirements.txt
```

Step 3: Set Up the `.env` File

1. **Copy the Example `.env` File**:

Duplicate the `.env.example` file to create a `.env` file:

```
cp .env.example .env
```

2. **Update the `.env` File**:

   Open the `.env` file and fill in your MySQL database credentials:

```
DB_USER=""
DB_PASSWORD=""
DB_HOST=""
DB_NAME=""
DB_PORT=""
```

Step 4: Database Setup

Ensure that your MySQL server is running, and the necessary database has been created. The database name should match the one specified in the `.env` file. Populate the database with the required data.

Step 5: Run the Jupyter Notebook

1. **Start Jupyter Notebook**:

   Launch Jupyter Notebook from the terminal:

```
jupyter notebook
```

2. **Execute the Notebook**:

   Open the `Assignment.ipynb` notebook in the browser and execute the cells sequentially:
   - The notebook will load, clean, and normalize the data.
   - Visualizations, such as bar charts and histograms, will be generated and saved in the `reports/` folder.

Step 6: Generate the HTML Report

Once the notebook analysis is complete:

1. **Run the Report Generation Script**:

Execute the `generate_report.py` script to generate an HTML report:

```
python generate_report.py
```

2. **View the Report**:

   The report will be saved as `aviation_report.html` in the `reports/` directory. You can open it in your browser to review the analysis.

Notes

- If Jupyter Notebook is not installed on your system, you can manually execute the notebook using any Python IDE, such as VS Code.
- Ensure that your MySQL server is running and the database is set up correctly before running any scripts.

Comprehensive Analysis Report

1. Data Overview

This Analysis report is based on aviation data, which includes flight details, departure and arrival times, airlines, and delay information. The goal of this analysis was to clean, preprocess, and analyze the dataset to identify key patterns and derive insights regarding flight delays.

---

2. Data Cleaning and Preprocessing

We performed the following steps to clean and normalize the data:

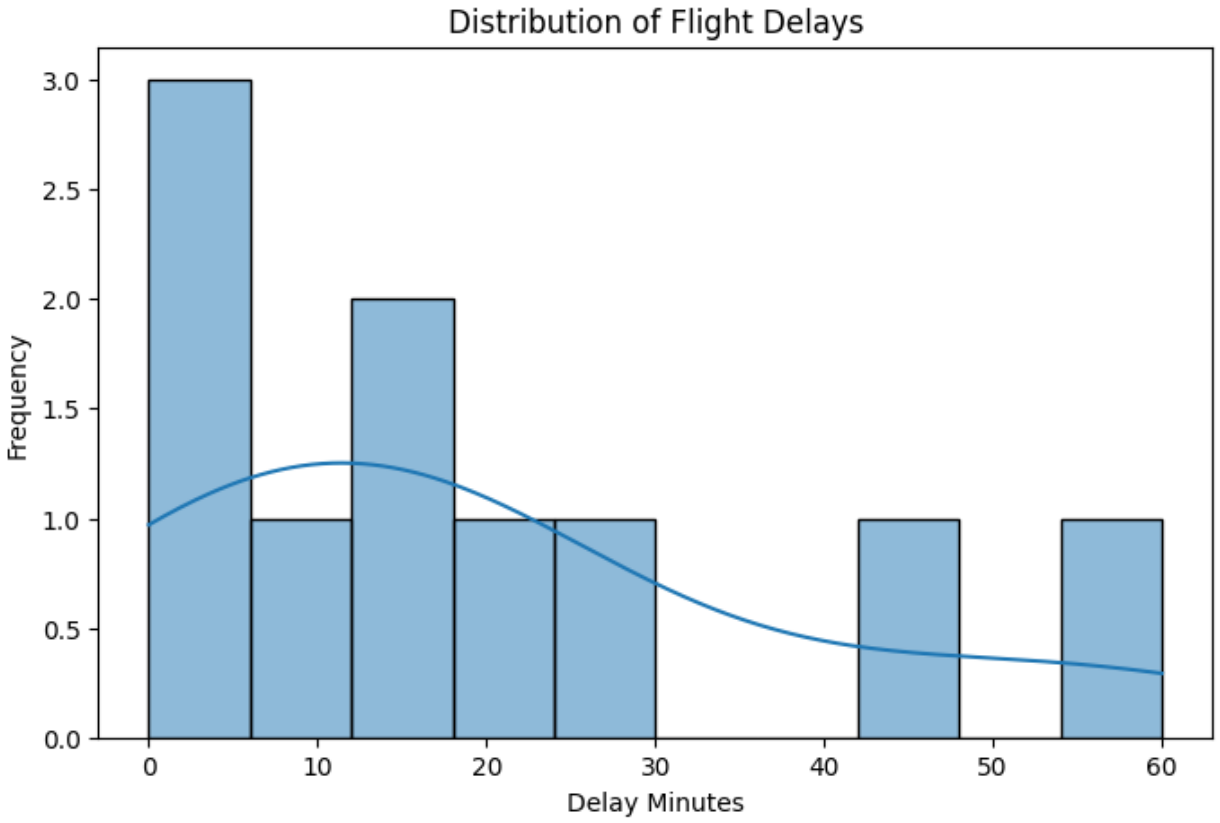
- **Handling Missing Values:**
  - Missing values in the `DelayMinutes` column were replaced with 0, representing no delay for those flights.
  - **Before:** 2 missing values in `DelayMinutes`.
  - **After:** No missing values.
- **Removing Duplicates:**
  - Duplicates based on flight number, times, and airline were identified and removed.
  - **Before:** 12 entries, 0 duplicates.
  - **After:** 12 unique entries retained.
- **Inconsistent Time Entries:**
  - We identified inconsistent time entries where departure times were later than arrival times. These were corrected or removed as appropriate.
  - **Before:** 2 inconsistent time entries.
  - **After:** All time inconsistencies resolved.
- **Data Normalization:**
  - Dates and times were standardized to a uniform format (YYYY-MM-DD for dates and 24-hour format for times).
  - New columns for combined `DepartureDateTime` and `ArrivalDateTime` were created for more precise time analysis.
  - **Flight Duration:** Calculated in minutes by taking the difference between `ArrivalDateTime` and `DepartureDateTime`.

---

3. Data Analysis & Visualizations

3.1 Flight Delay Distribution

- **Summary Statistics:**
  - **Mean Delay:** 19.5 minutes
  - **Standard Deviation:** 19.5 minutes
  - **Median Delay:** 15 minutes
  - **Max Delay:** 60 minutes
- **Distribution Visualization:**

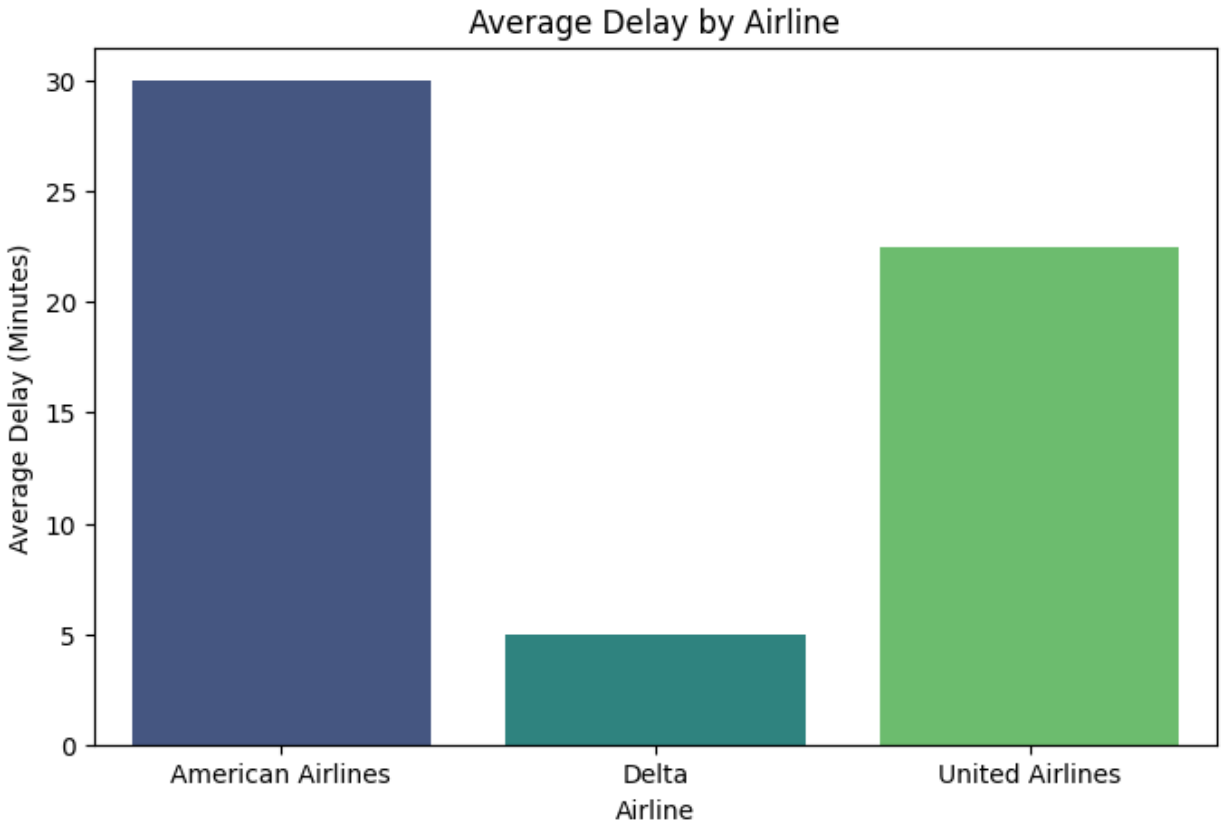


  - The majority of flights experienced delays between 0 and 10 minutes.
  - The distribution was right-skewed, indicating that most flights had relatively short delays, but a few experienced significant delays.

3.2 Average Delay by Airline

- **Insight:**
  - **American Airlines:** Highest average delay (30 minutes).
  - **Delta Airlines:** Lowest average delay (5 minutes).
  - **United Airlines:** Moderate average delay (22.5 minutes).

- **Visualization:**

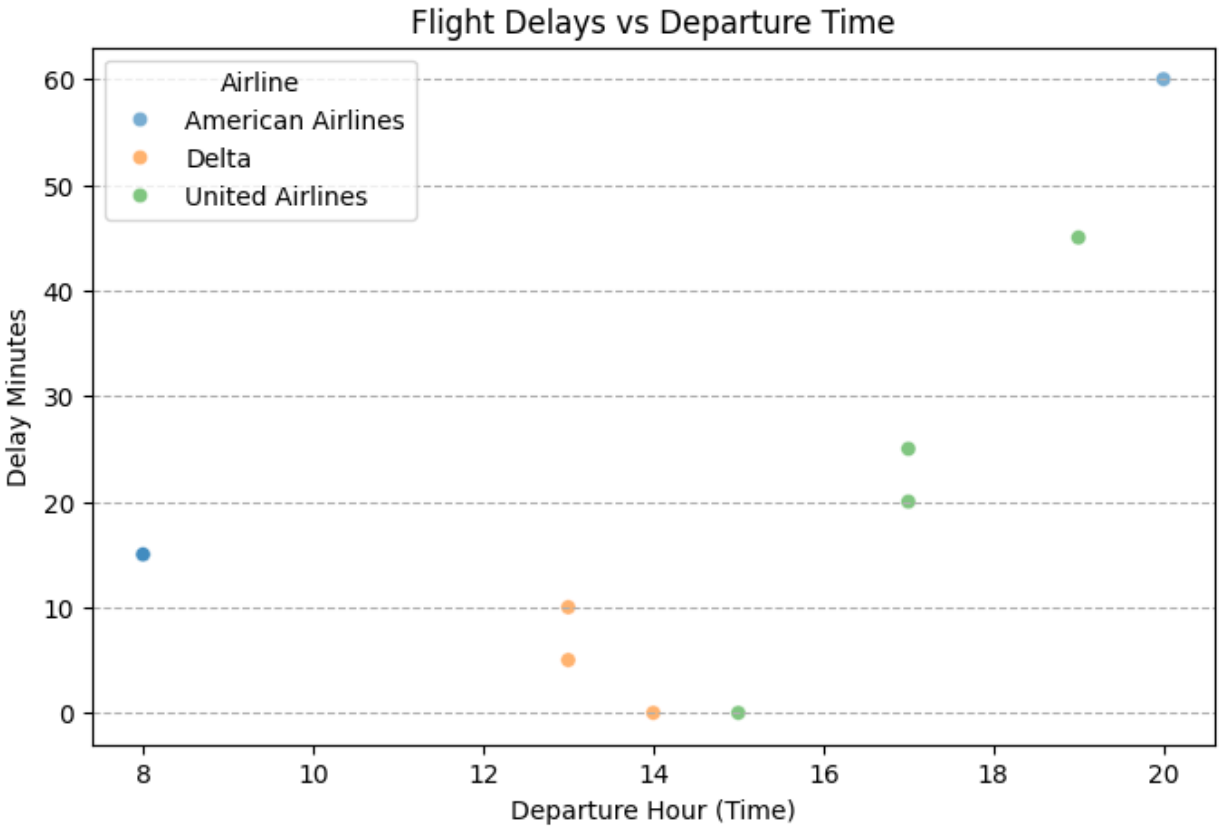


This bar chart illustrates the varying operational performance across airlines.

3.3 Impact of Departure Time on Delays

- **Insight:**
  - Delays increase significantly after 4 PM, particularly for American and United Airlines.
  - Delta Airlines demonstrates the most punctuality across all time periods.
  - Morning flights tend to have fewer delays.

- **Visualization:**

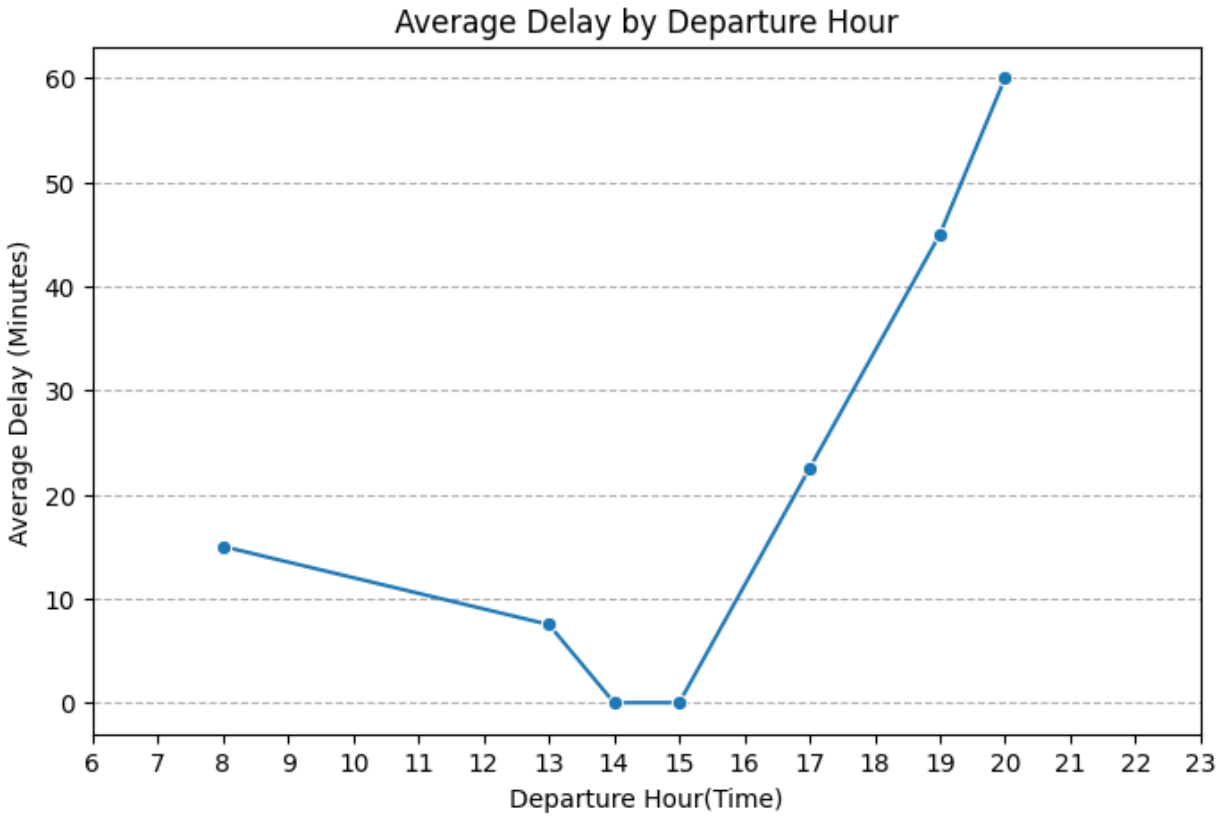


A scatter plot shows the relationship between departure time and delays, with a clear peak in delays during the late afternoon and evening.

3.4 Delay Trends Across Departure Hours

- **Insight:**
  - Delays peak at around 8:00 AM and 8:00 PM, with minimal delays between 2:00 PM and 3:00 PM.

- **Visualization:**



A line plot showing that delays increase significantly in the evening, with a peak at 8:00 PM.

3.5 ANOVA Analysis: Differences in Delays Between Airlines

- **ANOVA Test Result:**
  - **F-statistic:** 1.44
  - **p-value:** 0.299
  - **Conclusion:** There is no statistically significant difference in delays between the airlines.

---

4. Key Insights

- **Delay Distribution:** Most flights experienced delays of less than 30 minutes, with a significant portion of delays under 10 minutes.
- **Airline Performance:**
  - **American Airlines** had the highest average delays, especially during peak periods.
  - **Delta Airlines** consistently outperformed the others with the lowest delay times.
  - **United Airlines** had moderate delays, peaking during the evening hours.

- **Impact of Departure Times:**
  Flights departing in the evening tend to experience longer delays, with the highest delays recorded after 5 PM.

---

5. Recommendations

- **Operational Adjustments for American Airlines:**
  American Airlines should allocate more resources during peak delay periods (morning and evening) to improve punctuality and reduce delays.
- **Improved Scheduling:**
  Airlines may benefit from adjusting flight schedules, particularly around high-delay times such as late afternoon and early evening, to minimize bottlenecks and congestion.
- **Resource Allocation:**
  Airlines should optimize staff and ground crew deployment during peak delay periods to ensure smoother operations and minimize disruptions.