Name: Shreyas Aditya Baksi
Roll no: SE22UCSE317

Project Title: Flight Delay Prediction - Analysis and Report

1. Project Overview

The aim of this project is to build a robust Deep Neural Network (DNN) model that can predict whether a flight will be delayed by 15 minutes or more. This binary classification task leverages historical flight and delay data, and is optimized using a 1D Convolutional Neural Network (CNN) architecture for structured time-independent tabular data.

---

2. Approach

1. Load and preprocess a large dataset (`flights_clean.csv` with ~398,000 rows).
2. Engineer new features like `delay_per_flight` and `is_peak_month`.
3. Normalize numerical features and label-encode categorical ones.
4. Reshape the tabular data for compatibility with a 1D CNN.
5. Train the model using a class-weighted loss to handle imbalance.
6. Evaluate performance on a separate test set.
7. Predict delays on an unseen dataset and optionally compare with other models.

---

3. Technical Implementation Analysis

3.1 Deep-Neural Networks Fundamentals

The model is based on a sequential DNN architecture using a **1D Convolutional Neural Network**, which extracts local patterns across features treated as sequential data. The architecture includes:

- 3 Conv1D layers with LeakyReLU and BatchNorm
- MaxPooling and Dropout for regularization
- A Dense layer followed by a sigmoid-activated output

This design allows the model to generalize well even across complex, structured data.

3.2 Optimizer Implementation

- **Optimizer Used:** Adam Optimizer
- **Initial Learning Rate:** 0.001
- **Learning Rate Scheduling:** ReduceLROnPlateau
- **Loss Function:** Binary Crossentropy
- **Evaluation Metrics:** Accuracy and AUC (Area Under the Curve)

This ensures faster convergence and adaptability during training.

3.3 Preprocessing

- **Null Handling:** Dropped rows with null values.
- **Target Variable:** Derived from `arr_delay` — 1 if delay ≥ 15 mins, else 0.
- **Feature Engineering:**
  - `delay_per_flight = total_delay / arrival_flights`
  - `is_peak_month` based on common travel months
- **Encoding:** Label encoding for `carrier`, `airport`
- **Scaling:** StandardScaler for all numerical columns
- **Train/Val/Test Split:** 70/15/15 (stratified)

---

4. Dataset Used

- **Source File:** `flights_clean.csv`
- **Rows:** 398,233
- **Features:** Includes carrier, airport, delay metrics, cancellation info, etc.
- **Target Variable:** `Delayed` (binary: 1 or 0)

A secondary file `unseen_flights_100_balanced.csv` was used for external validation.(as train dataset)

---

5. Why did I Use 1D CNN Model for this Project?

CNNs are not just for images—they can capture local patterns in structured data. Each feature is treated as a timestep, allowing the model to learn spatial (feature-wise) correlations. Unlike feedforward networks, 1D CNNs:

- Extract localized dependencies
- Require fewer parameters
- Generalize better with feature-based input like this

---

6. How CNN Differs from Other 3 Models Used

- **Feedforward Neural Network**: Lacks spatial or sequential awareness. Simpler, but underperforms.
- **LSTM**: Effective for sequences over time, but overkill for non-temporal structured data.
- **Transformer**: Captures long-range dependencies; complex and requires large datasets.
- **1D CNN**: Captures local patterns, efficient, regularized well with dropout and pooling layers.

The CNN-1D bridges the gap between structure and complexity, offering a lightweight yet efficient solution.

---

7. Analysis of the OUTPUT / RESULT / GRAPH

- **Test Loss:** `0.6792`
- **Test Accuracy:** `0.9679`
- **Test AUC:** `0.5180` (indicating a model that may require improvements in classification separation)

Although accuracy seems high, **AUC is near random**, suggesting that the class imbalance or feature distribution may have skewed results. The validation accuracy oscillated, and overfitting was somewhat controlled via early stopping.

Plots like training vs validation accuracy/loss were generated and saved (`cnn_training_performance.png`), clearly showing training dynamics.

---

8. Improvements Over Basic Model

- Used 3 Conv1D layers instead of a flat feedforward network

- Added dropout and batch normalization to prevent overfitting
- Class weighting handled imbalance
- Engineered new meaningful features

---

9. Learning Outcomes

- Deeper understanding of how CNNs can apply beyond images
- Preprocessing structured data for DNN input
- Training stabilization using callbacks like ReduceLROnPlateau and EarlyStopping
- Model evaluation using AUC, not just accuracy

---

10. Challenges Encountered

- **Class Imbalance:** Majority class dominated learning; AUC reflects this
- **Low AUC despite high accuracy:** Indicates poor model calibration
- **File handling in Colab:** Required explicit upload and reloading for unseen dataset
- **Feedforward model missing:** Comparison was skipped due to missing H5 file

---

11. Recommendations for Further Improvement

- Perform **SMOTE or other resampling** for class balance
- Use **F1-score and Precision/Recall** in addition to AUC
- Tune the **kernel size and filter count** in CNN layers
- Add **attention mechanism** for learning feature importance
- Investigate **ensemble techniques** for better stability

---

12. Conclusion

This project demonstrates the feasibility of applying 1D CNNs to structured airline delay data. Despite strong preprocessing and a well-designed model, the outcome points to challenges in capturing class boundaries, evident in the AUC score. However, the exercise offered strong learning in modeling pipelines, CNN implementation, and real-world data preparation. With further enhancements, this model can serve as a solid foundation for production-grade delay prediction systems.