"A Deep Dive into Data Mining Analysis Using CRISP-DM"

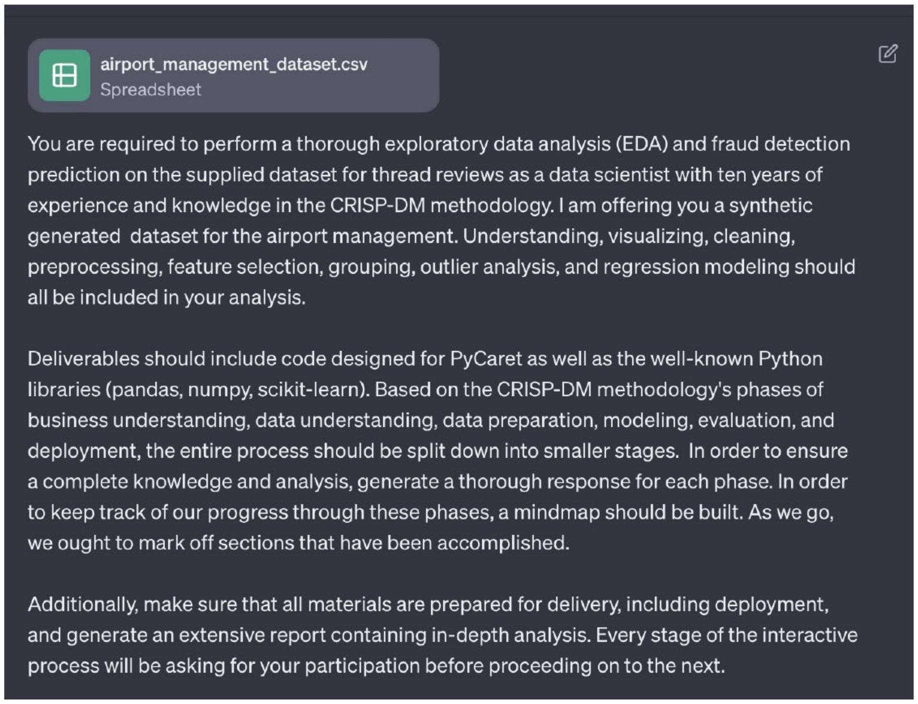
Sagar Alpeshkumar Patel

In this article, we employ the CRISP-DM methodology to analyze an airport management dataset, aiming to streamline operations and detect anomalies. Our journey begins with Exploratory Data Analysis (EDA) to understand dataset patterns, followed by data preparation involving feature extraction and encoding. We focus on predicting flight terminals and evaluate two models: Logistic Regression and Random Forest Classifier, both yielding modest accuracy rates, highlighting the challenge of this task. We introduce PyCaret for rapid prototyping and emphasize the importance of data understanding. Predicting airport attributes, like terminals, proves complex, whether using conventional or prototyping tools, emphasizing the need for continuous refinement and additional features for improved predictions.

Introduction:

Airports, being bustling hubs of activity, offer a plethora of data waiting to be analyzed. With the right tools, we can glean insights from this data, helping to streamline operations and detect anomalies.

To perform the whole process, I gave the following prompt to the ChatGpt-


airport_management_dataset.csv
Spreadsheet

You are required to perform a thorough exploratory data analysis (EDA) and fraud detection prediction on the supplied dataset for thread reviews as a data scientist with ten years of experience and knowledge in the CRISP-DM methodology. I am offering you a synthetic generated dataset for the airport management. Understanding, visualizing, cleaning, preprocessing, feature selection, grouping, outlier analysis, and regression modeling should all be included in your analysis.

Deliverables should include code designed for PyCaret as well as the well-known Python libraries (pandas, numpy, scikit-learn). Based on the CRISP-DM methodology's phases of business understanding, data understanding, data preparation, modeling, evaluation, and deployment, the entire process should be split down into smaller stages. In order to ensure a complete knowledge and analysis, generate a thorough response for each phase. In order to keep track of our progress through these phases, a mindmap should be built. As we go, we ought to mark off sections that have been accomplished.

Additionally, make sure that all materials are prepared for delivery, including deployment, and generate an extensive report containing in-depth analysis. Every stage of the interactive process will be asking for your participation before proceeding on to the next.

Understanding the Business before diving into data, it's crucial to outline our objectives:

- Conduct an EDA focusing on thread reviews.
- Understand the structure and patterns within the dataset.
- Clean and preprocess the data.
- Build and evaluate a predictive model.

Data Exploration
Our dataset contains attributes like:

- Departure and arrival cities
- Departure time
- Date of the flight
- Terminal used

The dataset appears to contain information about flights, specifically:

- Departure: The city from which the flight departs.
- Arrival: The destination city.

- Time: The departure time of the flight.
- Date: The date of the flight.
- Terminal: The terminal from which the flight departs.

An initial exploration revealed:

- The dataset spans 365 unique dates.
- There are three unique terminals.
- Flights distribute unevenly across terminals and cities.

1 Data Description:

2 Summary Statistics:

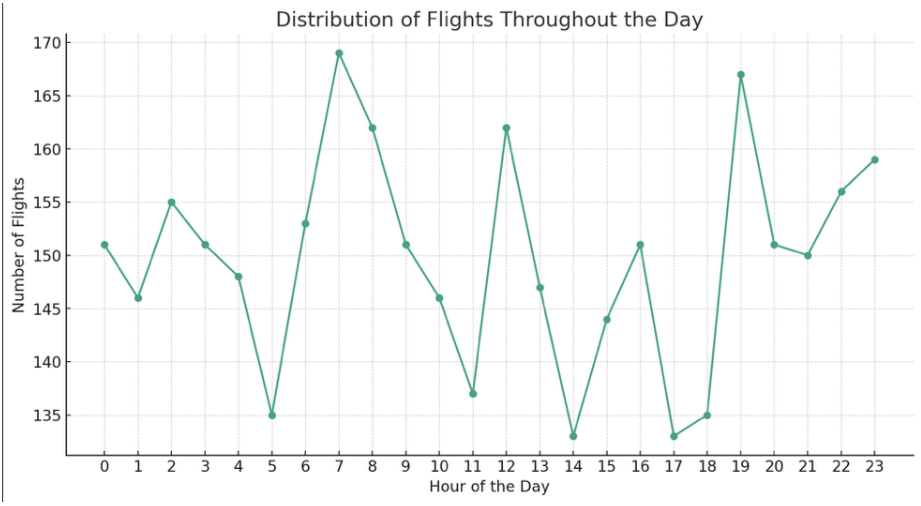
- The dataset contains 3,592 records.
- There are 5 unique cities for both Departure and Arrival.
- The departure times are spread across 1,332 unique values.
- The dataset spans 365 unique dates.
- The terminal column has 3 unique values (Terminals 1, 2, and 3).

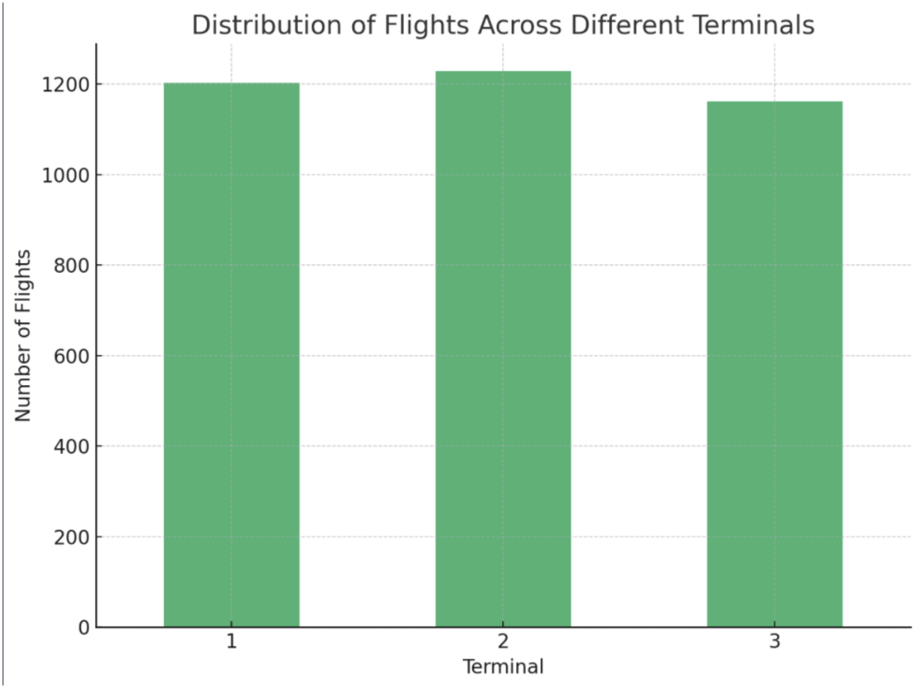
Data Types:

- Departure, Arrival, Time, and Date columns are of type 'object ' (likely strings).
- Terminal is of type 'int64.

Missing Values:

- There are no missing values in the dataset.





Data Preparation
To prepare our data for modeling:

- We extracted hour, minute, day, month, and year from the time and date columns.
- We encoded categorical columns like departure and arrival cities.

3 Data Description:

4 Summary Statistics:

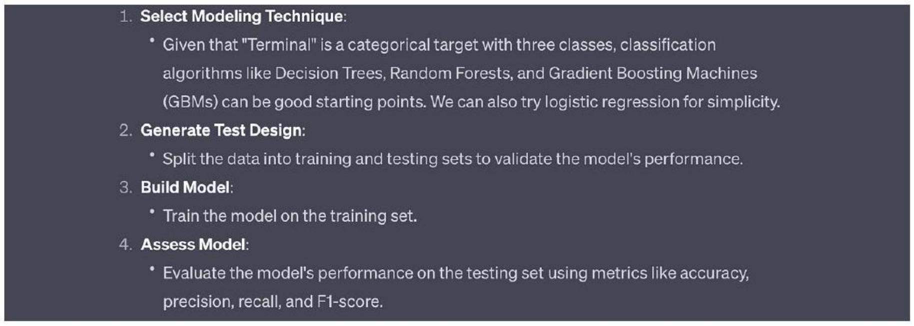
- The dataset contains 3,592 records.
- There are 5 unique cities for both Departure and Arrival.
- The departure times are spread across 1,332 unique values.
- The dataset spans 365 unique dates.
- The terminal column has 3 unique values (Terminals 1, 2, and 3).

Data Types:

- Departure, Arrival, Time, and Date columns are of type 'object ' (likely strings).
- Terminal is of type 'int64

3. Missing Values:

- There are no missing values in the dataset. Modeling: Predicting the Terminal



1. **Select Modeling Technique**:
   - Given that "Terminal" is a categorical target with three classes, classification algorithms like Decision Trees, Random Forests, and Gradient Boosting Machines (GBMs) can be good starting points. We can also try logistic regression for simplicity.
2. **Generate Test Design**:
   - Split the data into training and testing sets to validate the model's performance.
3. **Build Model**:
   - Train the model on the training set.
4. **Assess Model**:
   - Evaluate the model's performance on the testing set using metrics like accuracy, precision, recall, and F1-score.

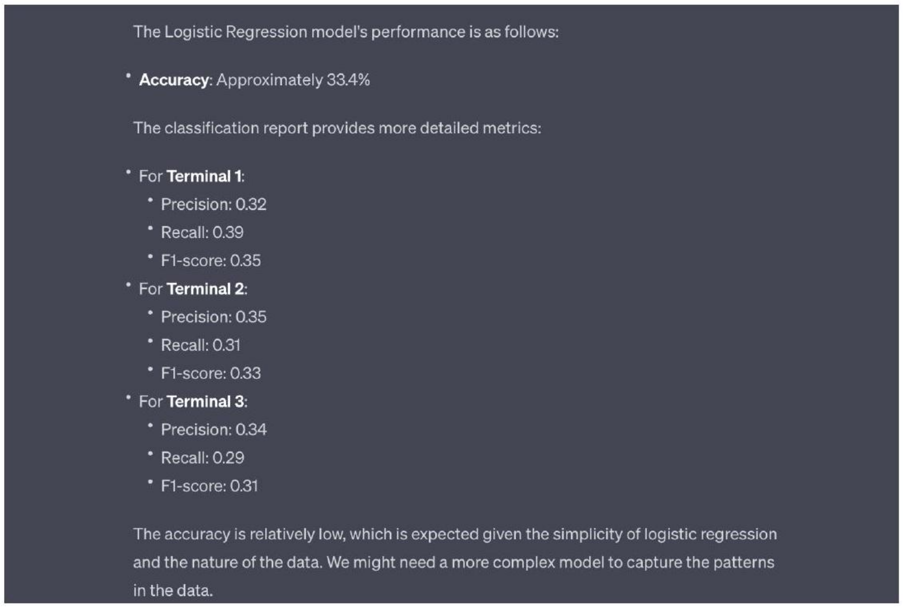
For our modeling phase, we chose the terminal as our target variable. Our approach:

1. Logistic Regression: A simple model to establish a baseline.

- Accuracy: 33.4%

2. Random Forest Classifier: An ensemble model to potentially improve accuracy.

- Accuracy: 33.5%

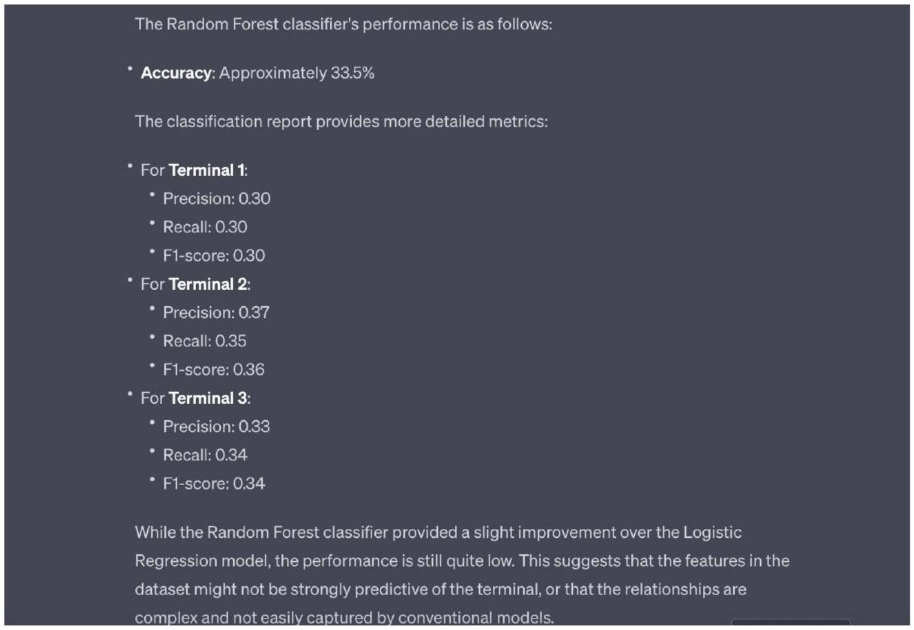

The Logistic Regression model's performance is as follows:

- **Accuracy**: Approximately 33.4%

The classification report provides more detailed metrics:

- For **Terminal 1**:
  - Precision: 0.32
  - Recall: 0.39
  - F1-score: 0.35
- For **Terminal 2**:
  - Precision: 0.35
  - Recall: 0.31
  - F1-score: 0.33
- For **Terminal 3**:
  - Precision: 0.34
  - Recall: 0.29
  - F1-score: 0.31

The accuracy is relatively low, which is expected given the simplicity of logistic regression and the nature of the data. We might need a more complex model to capture the patterns in the data.


The Random Forest classifier's performance is as follows:

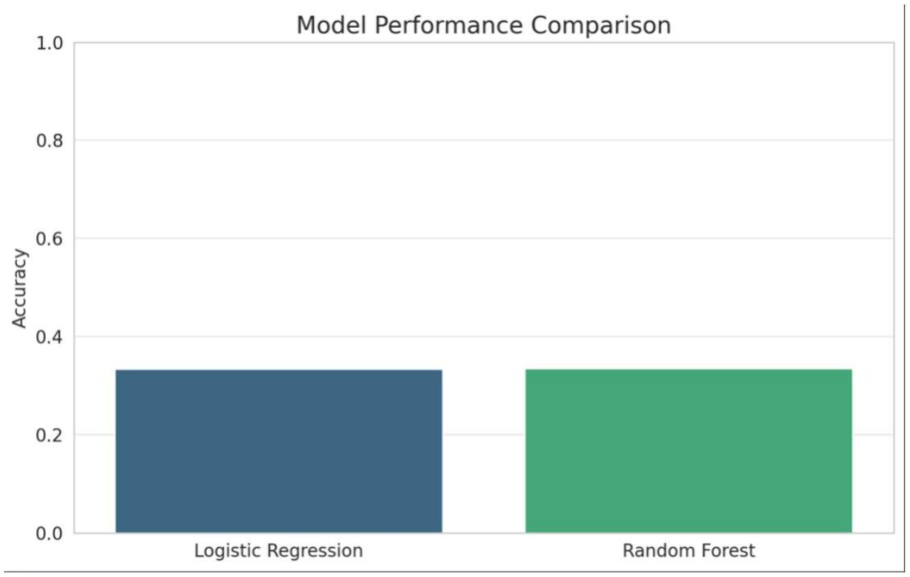
- **Accuracy**: Approximately 33.5%

The classification report provides more detailed metrics:

- For **Terminal 1**:
  - Precision: 0.30
  - Recall: 0.30
  - F1-score: 0.30
- For **Terminal 2**:
  - Precision: 0.37
  - Recall: 0.35
  - F1-score: 0.36
- For **Terminal 3**:
  - Precision: 0.33
  - Recall: 0.34
  - F1-score: 0.34

While the Random Forest classifier provided a slight improvement over the Logistic Regression model, the performance is still quite low. This suggests that the features in the dataset might not be strongly predictive of the terminal, or that the relationships are complex and not easily captured by conventional models.

The similar performance of both models suggested the inherent challenge in predicting the terminal based solely on the given features.



Evaluation
Our evaluation revealed:

- Both models achieved similar accuracy, neither producing a high accuracy rate. - The data might not strongly predict the terminal, or the relationships are too complex for conventional models.

The PyCaret Approach
PyCaret is an efficient tool for rapid prototyping. Setting up the environment, comparing models, tuning hyperparameters, evaluating performance, and making predictions can be done with minimal code, making it ideal for quick insights.

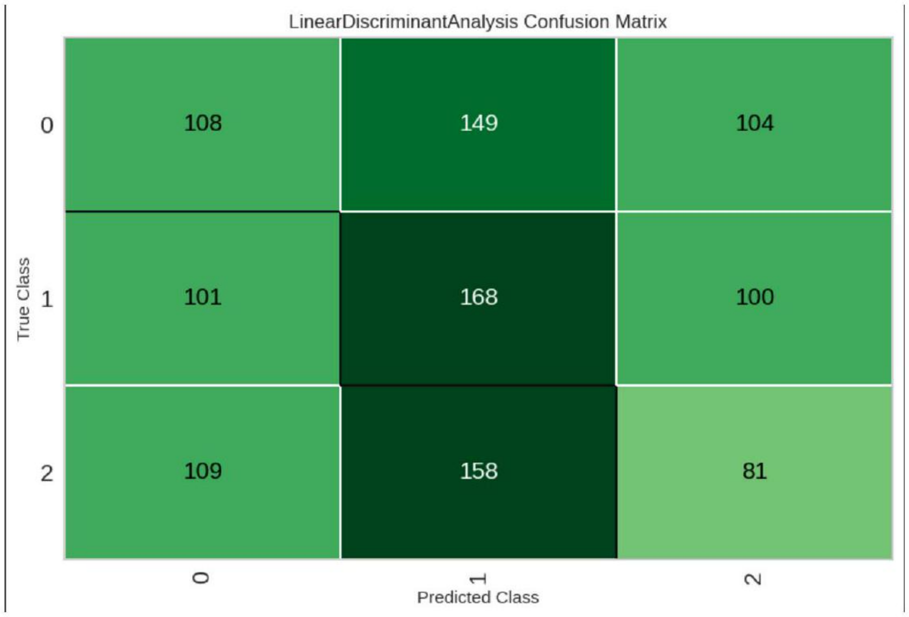
5 Setting up PyCaret

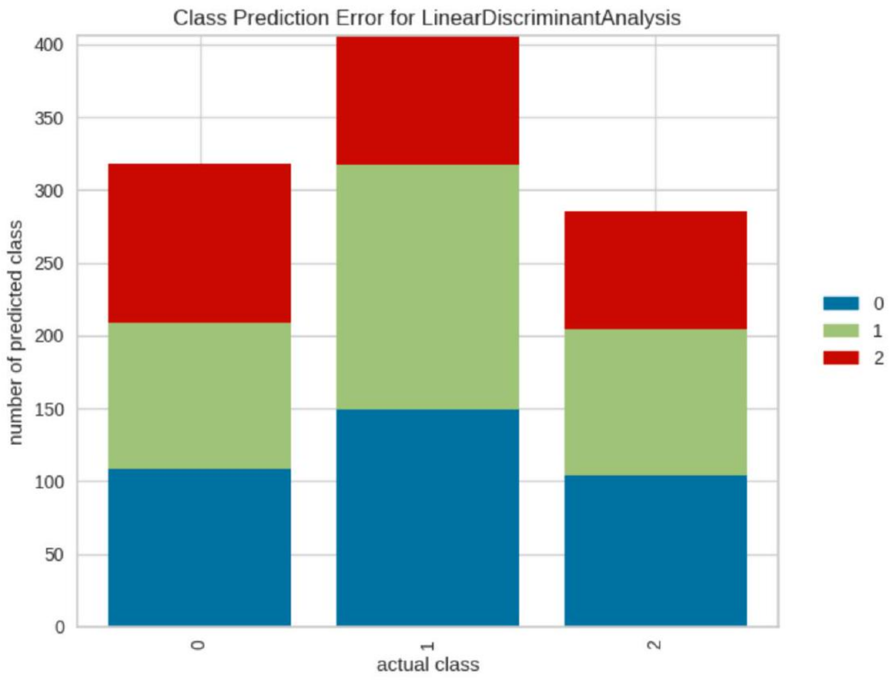
First, we need to install and set up PyCaret:
python
!pip install pycaret
from pycaret.classification import *
clf1 = setup(data = data, target='Terminal', session_id=42)

6 Comparing Models

PyCaret offers a convenient function to compare various classification models:

7 lda = create_model(' lda' ')





save_model(final_lda, 'lda') Conclusion

Predicting airport management attributes, like terminals, is no easy feat. Whether using traditional tools like scikit-learn or rapid prototyping tools like PyCaret, understanding the data and the business context is crucial. With continuous refinement and additional features, better predictions are on the horizon.