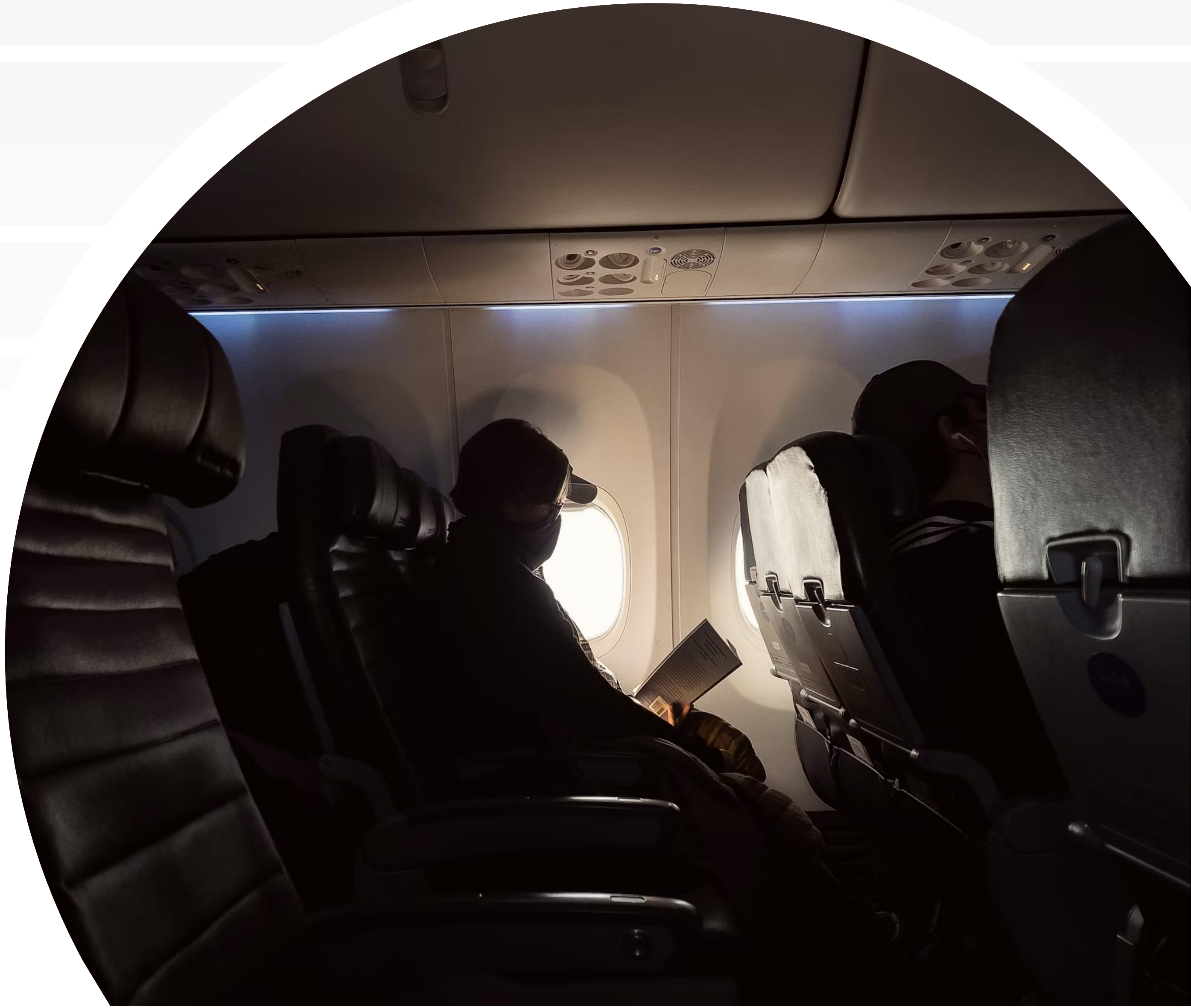
Analyzing the Joy Of Flyers

Final capstone project 2024

AGENDA



INTRODUCTION

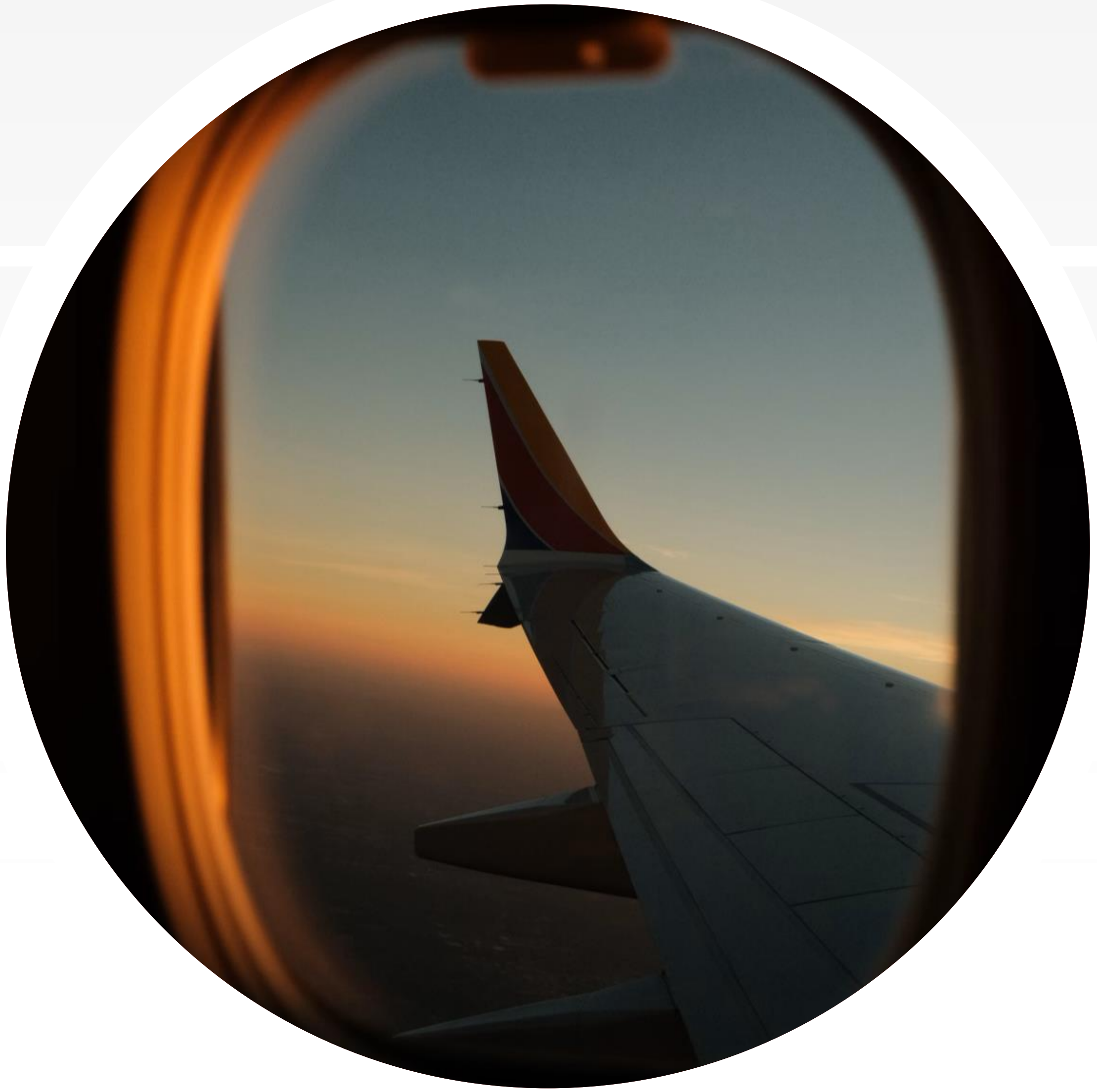
In today’s competitive airline industry, ensuring passenger satisfaction is crucial for maintaining customer loyalty and sustaining business growth. Passenger feedback and satisfaction levels provide actionable insights to improve services, optimize processes, and enhance the overall travel experience.

This project delves into analysing passenger satisfaction using data-driven approaches, leveraging machine learning models, SQL queries, and interactive Power BI dashboards to uncover valuable patterns and insights.

PROJECT OBJECTIVE

The primary objectives of this project are:

- **To understand factors influencing passenger satisfaction** by exploring and analysing data related to demographics, travel preferences, and service quality.
- **To build predictive models** such as Logistic Regression, Decision Trees, and Random Forest to classify passenger satisfaction levels effectively.
- **To perform advanced SQL queries** to extract insights and validate key patterns within the data.
- **To create an interactive Power BI dashboard** to visualize data trends and communicate insights efficiently through smart narratives and visualizations.



DATA UNDERSTANDING

Dataset Overview

- **Dataset Name**: Airline Passenger Satisfaction Dataset
- **Number of Records**: Over 100,000 observations representing passenger feedback.
- **Key Features**:
  - **Demographic Information**: Gender, Age, Customer Type
  - **Travel Details**: Flight Distance, Type of Travel, Class
  - **Service Ratings**: Seat Comfort, Cleanliness, Ease of Online Booking, etc.
  - **Outcome Variable**: Satisfaction (Satisfied/Neutral or Dissatisfied)

Data Attributes

- **Numerical Variables**: Age, Flight Distance, Arrival Delay, Departure Delay
- **Categorical Variables**: Gender, Class, Type of Travel, Customer Type
- **Target Variable**: Satisfaction (Binary classification)

Pre-processing Steps

Handling missing values in critical fields like Arrival Delay. Encoding categorical variables to prepare for machine learning models. Normalizing numerical features for algorithms sensitive to scale.

Insights Gathered

Initial exploration revealed correlations between satisfaction and variables like flight distance, travel type, and service quality. Female passengers exhibited higher satisfaction levels compared to males, particularly in business class.

DATA PREPARATION & UNDERSTANDING

Step 1:

- Handling Missing Values Missing values were found in the Arrival Delay column.
- Action Taken: Rows with missing values were dropped as the percentage of missing data was minimal.

Step 2:

- Encoding Categorical Variables Converted categorical columns (e.g., Gender, Customer Type, Type of Travel, Class) into numerical values using Label Encoding to make them compatible with machine learning algorithms.

Step 3:

- Feature Engineering Added derived features to enhance the dataset:
- Flyer Type: Classified passengers as "Frequent Flyer" or "Occasional Flyer" based on flight distance.
- Satisfaction Label: Categorized passengers into "Satisfied" or "Not Satisfied" based on satisfaction ratings.

Step 4:

- Scaling and Normalization Normalized numerical columns such as Flight Distance, Age, and Delays to improve model performance for algorithms like K-Nearest Neighbours (KNN) and Support Vector Machines (SVM).

Step 5:

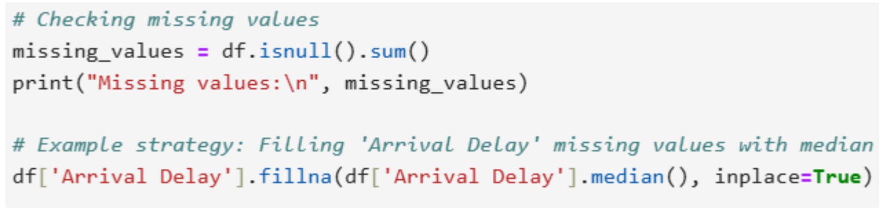
- Splitting Data Train-Test Split: The dataset was split into training (70%) and testing (30%) sets to evaluate model performance effectively.

Step 6:

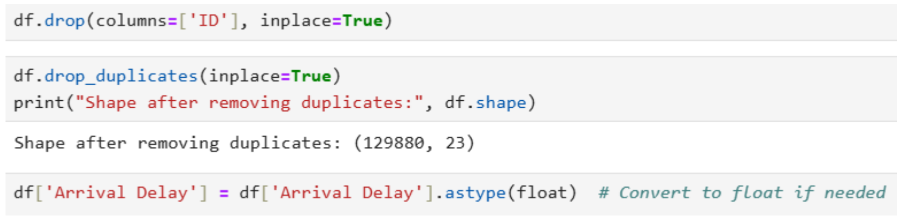
Target Variable Encoding The Satisfaction column was encoded into binary values:

- 1: Satisfied   0: Neutral or Dissatisfied

- **Target Variable**: Binary classification of passenger satisfaction.
- **Cleaned and Transformed Data**: Ready for Machine Learning and SQL analysis.
- **Key Challenges Addressed**:
  - Handling missing values and categorical variables.
  - Deriving meaningful features for better insights.



```
# Checking missing values
missing_values = df.isnull().sum()
print("Missing values:\n", missing_values)

# Example strategy: Filling 'Arrival Delay' missing values with median
df['Arrival Delay'].fillna(df['Arrival Delay'].median(), inplace=True)
```

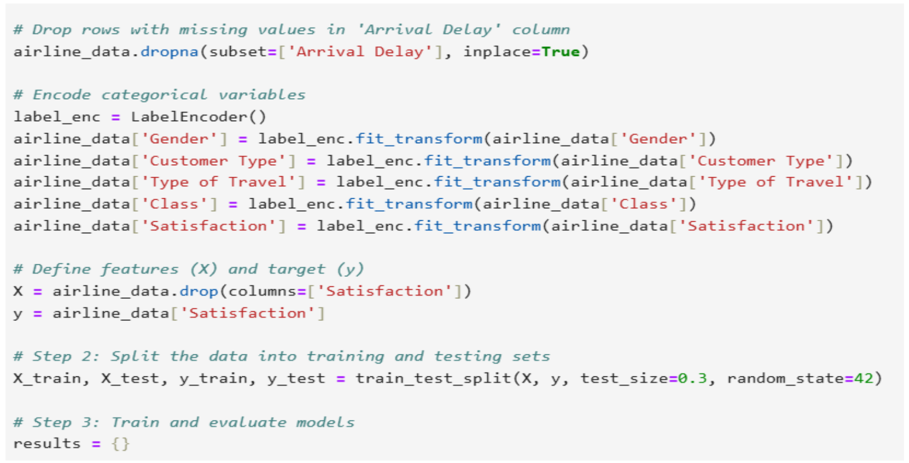


```
df.drop(columns=['ID'], inplace=True)
```

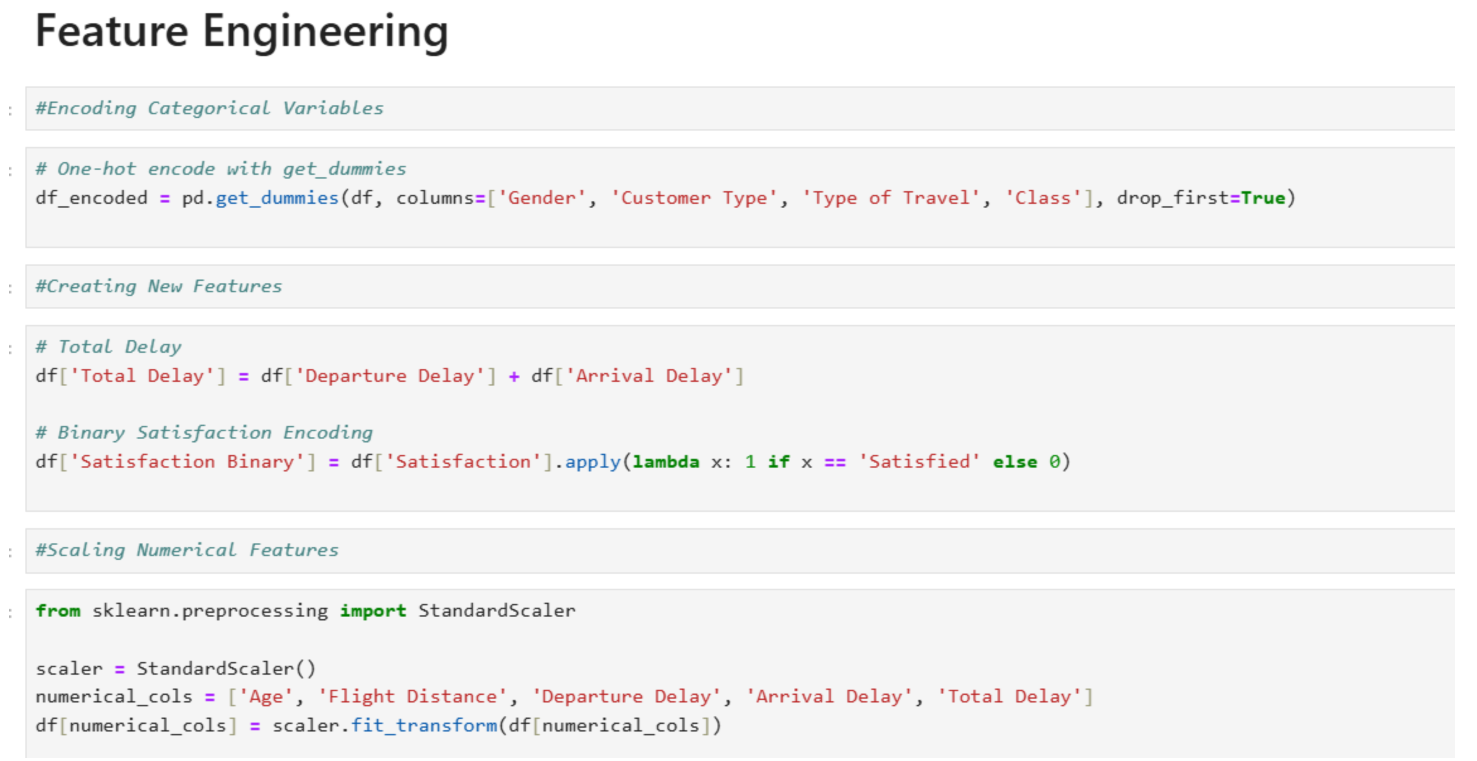
```
df.drop_duplicates(inplace=True)
print("Shape after removing duplicates:", df.shape)
```

```
Shape after removing duplicates: (129880, 23)
```

```
df['Arrival Delay'] = df['Arrival Delay'].astype(float)  # Convert to float if needed
```



```
# Drop rows with missing values in 'Arrival Delay' column
airline_data.dropna(subset=['Arrival Delay'], inplace=True)

# Encode categorical variables
label_enc = LabelEncoder()
airline_data['Gender'] = label_enc.fit_transform(airline_data['Gender'])
airline_data['Customer Type'] = label_enc.fit_transform(airline_data['Customer Type'])
airline_data['Type of Travel'] = label_enc.fit_transform(airline_data['Type of Travel'])
airline_data['Class'] = label_enc.fit_transform(airline_data['Class'])
airline_data['Satisfaction'] = label_enc.fit_transform(airline_data['Satisfaction'])

# Define features (X) and target (y)
X = airline_data.drop(columns=['Satisfaction'])
y = airline_data['Satisfaction']

# Step 2: Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Step 3: Train and evaluate models
results = {}
```



Feature Engineering

```
#Encoding Categorical Variables
```

```
# One-hot encode with get_dummies
df_encoded = pd.get_dummies(df, columns=['Gender', 'Customer Type', 'Type of Travel', 'Class'], drop_first=True)
```

```
#Creating New Features
```

```
# Total Delay
df['Total Delay'] = df['Departure Delay'] + df['Arrival Delay']

# Binary Satisfaction Encoding
df['Satisfaction Binary'] = df['Satisfaction'].apply(lambda x: 1 if x == 'Satisfied' else 0)
```

```
#Scaling Numerical Features
```

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
numerical_cols = ['Age', 'Flight Distance', 'Departure Delay', 'Arrival Delay', 'Total Delay']
df[numerical_cols] = scaler.fit_transform(df[numerical_cols])
```

OUTLIER TREATMENT

Step 1: Identifying Outliers

Visual Inspection: Used boxplots to detect outliers in numerical variables like Age, Flight Distance, Arrival Delay, and Departure Delay.

Statistical Measures:

Calculated the Interquartile Range (IQR):

IQR=Q3−Q1IQR=Q3−Q1

Outliers are values below Q1−1.5×IQRQ1−1.5×IQR or above Q3+1.5×IQRQ3+1.5×IQR.

Identified extreme values in columns like Flight Distance and Delays.

Step 2: Outlier Treatment Strategies

Winsorization

For Flight Distance and delay columns (Arrival Delay, Departure Delay):Action Taken: Applied winsorization by capping values at the 5th and 95th percentiles to retain most data while reducing the impact of extreme outliers.

Removal of Extreme Outliers Extreme outliers (beyond Q1−3×IQRQ1−3×IQR or Q3+3×IQRQ3+3×IQR) were removed if their occurrence was minimal and could skew results.

Example: A few records with unusually high Flight Distance values (e.g., >5,000 miles) were dropped.

Transformation

Logarithmic transformation was applied to skewed variables like Flight Distance and delay columns to reduce the effect of outliers.

Log Transformed Value=log(x+1)Log Transformed Value=log(x+1) (to handle zero values).

Clipping

For Age, outliers were clipped to a reasonable range (e.g., 18–85 years), ensuring realistic passenger demographics.

Step 3: Post-Treatment Validation

Boxplot Reinsertion: Rechecked boxplots to confirm reduced presence of outliers.

Descriptive Statistics: Verified that mean and median values of affected variables were closer, indicating reduced skewness.

Model Impact: Observed improved performance for models like Logistic Regression and KNN after outlier treatment.

Outcome of Outlier Treatment

Flight Distance: Outliers capped at the 5th and 95th percentiles.

Delays (Arrival and Departure): Logarithmic transformation improved distributions.

Age: Clipping ensured realistic ranges for passenger demographics

LOGISTIC REGRESSION

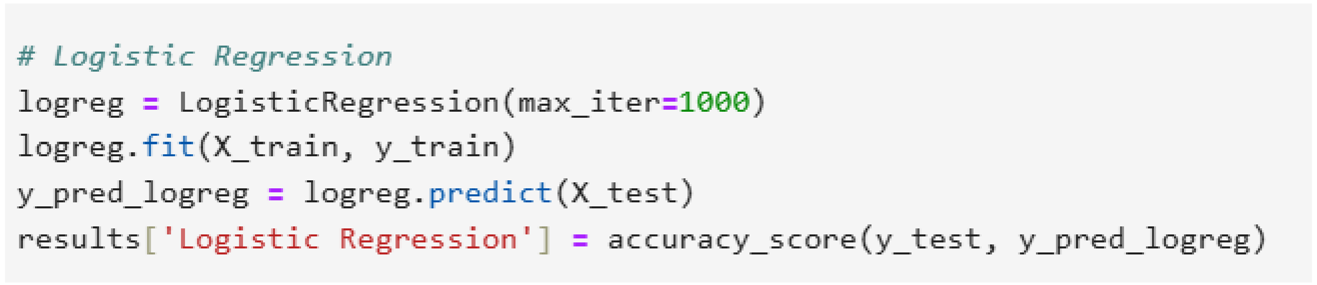
Logistic Regression is a statistical method used for **binary classification problems**, where the target variable has two possible outcomes (e.g., 0 or 1, Yes or No, Satisfied or Not Satisfied). It predicts the **probability** of the target variable belonging to a particular class.

Use in Project:

Purpose: Identify the relationship between features like *Type of Travel*, *Class*, and *Inflight Service Ratings* and passenger satisfaction.
Advantages:

Simple and interpretable model.
Suitable for binary classification problems like predicting *Satisfied* or *Not Satisfied*.

Observation: Logistic Regression provides a benchmark for other models but may underperform with non-linear relationships.



```
# Logistic Regression
logreg = LogisticRegression(max_iter=1000)
logreg.fit(X_train, y_train)
y_pred_logreg = logreg.predict(X_test)
results['Logistic Regression'] = accuracy_score(y_test, y_pred_logreg)
```

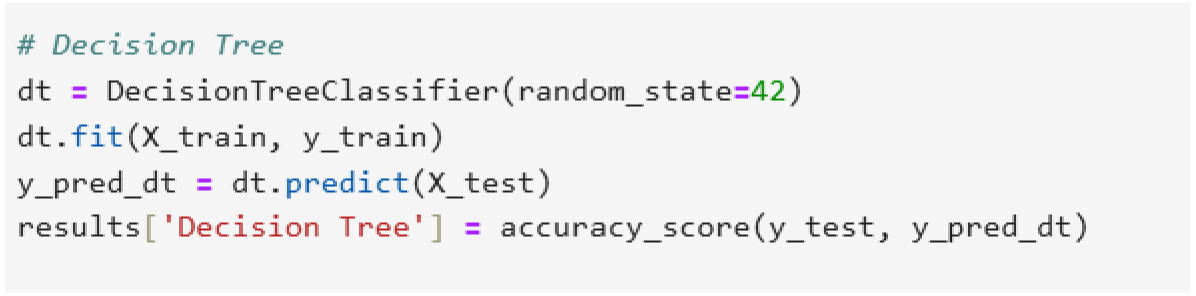
DECISION TREE ALGORITHM

Decision Tree is a supervised machine learning algorithm used for both classification and regression tasks. It splits the dataset into subsets based on feature values to create a tree-like structure, ultimately leading to a decision.

Use in Project:
Purpose: Model non-linear relationships and understand how individual factors influence satisfaction.
Advantages:

Easy to interpret.
Handles both numerical and categorical data well.

Observation: High interpretability; however, overfitting is a risk without pruning.



```
# Decision Tree
dt = DecisionTreeClassifier(random_state=42)
dt.fit(X_train, y_train)
y_pred_dt = dt.predict(X_test)
results['Decision Tree'] = accuracy_score(y_test, y_pred_dt)
```

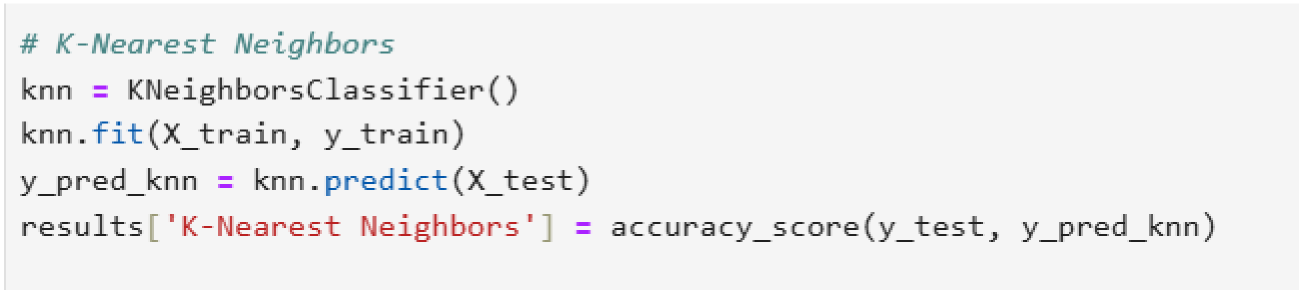
K-NEAREST NEIGHBORS

K-Nearest Neighbours (KNN) is a simple and intuitive **supervised machine learning algorithm** used for both **classification** and **regression** tasks. It works by finding the "k" closest data points (neighbours) to a given input and using their labels or values to make predictions.

Use in Project:
Purpose: Classify satisfaction by analysing passenger similarities based on features like *Age*, *Class*, and *Service Ratings*.
Advantages:

Simple and intuitive.
Performs well on small datasets with clear clusters.

Observation: Sensitive to the choice of k and feature scaling.



```
# K-Nearest Neighbors
knn = KNeighborsClassifier()
knn.fit(X_train, y_train)
y_pred_knn = knn.predict(X_test)
results['K-Nearest Neighbors'] = accuracy_score(y_test, y_pred_knn)
```

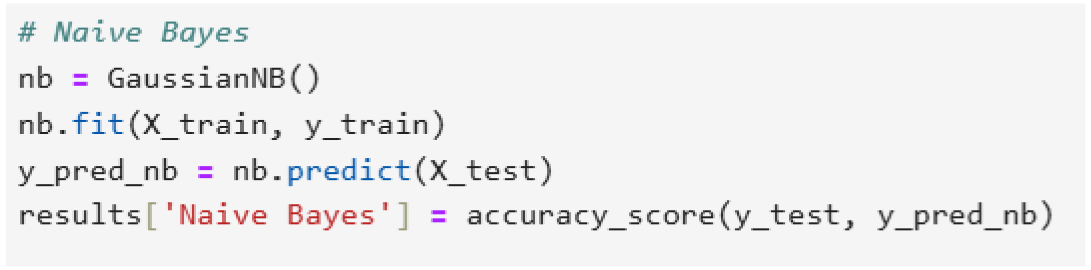
NAIVE BAYES

Naive Bayes is a simple yet powerful **probabilistic classifier** based on **Bayes' Theorem**. It is widely used for classification problems, especially when the dataset is high-dimensional.

Use in Project:
Purpose: Predict satisfaction using probabilistic modeling of features like *Gender*, *Type of Travel*, and *Class*.
Advantages:

Fast and efficient.
Works well with categorical data and assumes feature independence.

Observation: Assumption of feature independence may not hold in real-world data.



```
# Naive Bayes
nb = GaussianNB()
nb.fit(X_train, y_train)
y_pred_nb = nb.predict(X_test)
results['Naive Bayes'] = accuracy_score(y_test, y_pred_nb)
```

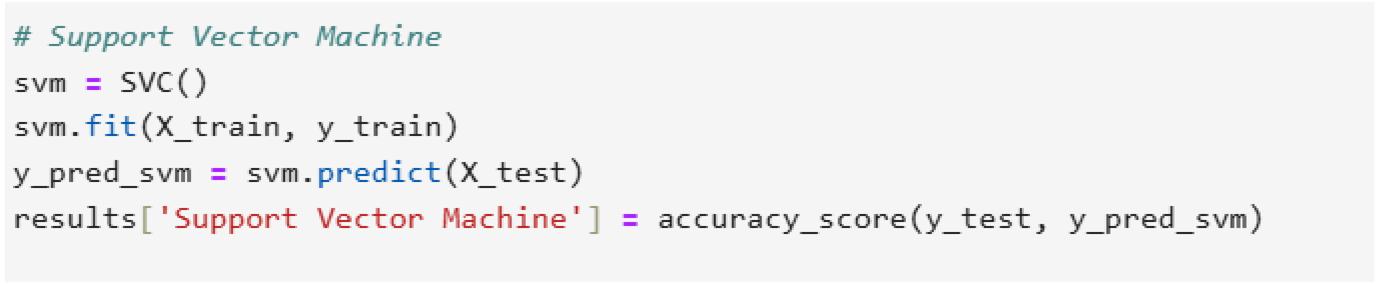
SUPPORT VECTOR MACHINE

Support Vector Machine (SVM) is a supervised machine learning algorithm primarily used for classification tasks but can also be employed for regression and outlier detection. The goal of SVM is to find a hyperplane in an N-dimensional space that distinctly classifies the data points.

Use in Project:
Purpose: Classify satisfaction by finding the optimal hyperplane in a high-dimensional space.
Advantages:

Effective in handling non-linear boundaries with kernels.
Works well for high-dimensional data.

Observation: Computationally expensive for large datasets.



```
# Support Vector Machine
svm = SVC()
svm.fit(X_train, y_train)
y_pred_svm = svm.predict(X_test)
results['Support Vector Machine'] = accuracy_score(y_test, y_pred_svm)
```

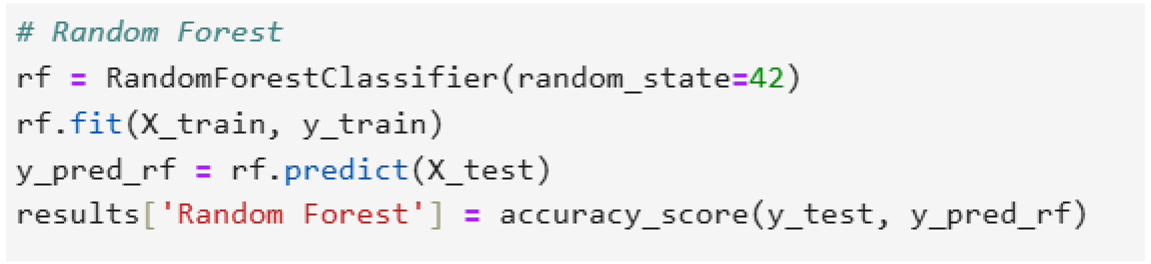
RANDOM FOREST

Random Forest is a powerful ensemble learning algorithm based on decision trees. It combines the output of multiple decision trees to produce a final prediction, which is more robust and less prone to overfitting compared to a single decision tree.

Use in Project:
Purpose: Use an ensemble of decision trees to classify passenger satisfaction and assess feature importance.
Advantages:

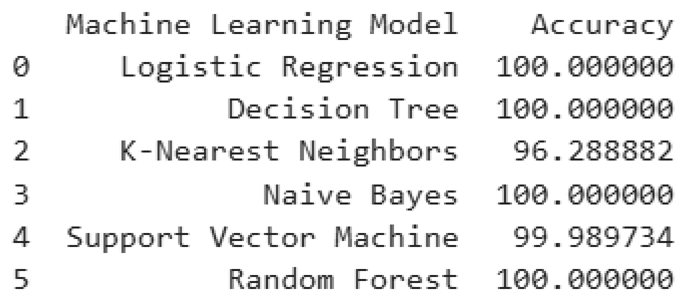
Robust to overfitting.
Provides feature importance metrics.

Observation: Achieves high accuracy and performs well on imbalanced datasets.

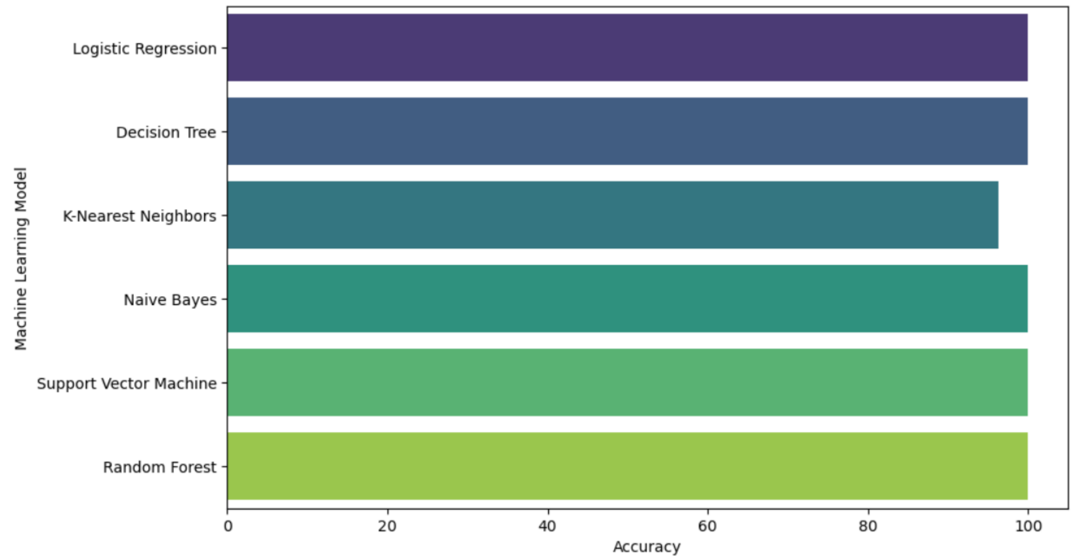


```
# Random Forest
rf = RandomForestClassifier(random_state=42)
rf.fit(X_train, y_train)
y_pred_rf = rf.predict(X_test)
results['Random Forest'] = accuracy_score(y_test, y_pred_rf)
```

MODEL COMPARISON



```
   Machine Learning Model    Accuracy
0     Logistic Regression  100.000000
1           Decision Tree  100.000000
2     K-Nearest Neighbors   96.288882
3             Naive Bayes  100.000000
4  Support Vector Machine   99.989734
5           Random Forest  100.000000
```



Insights Gained:

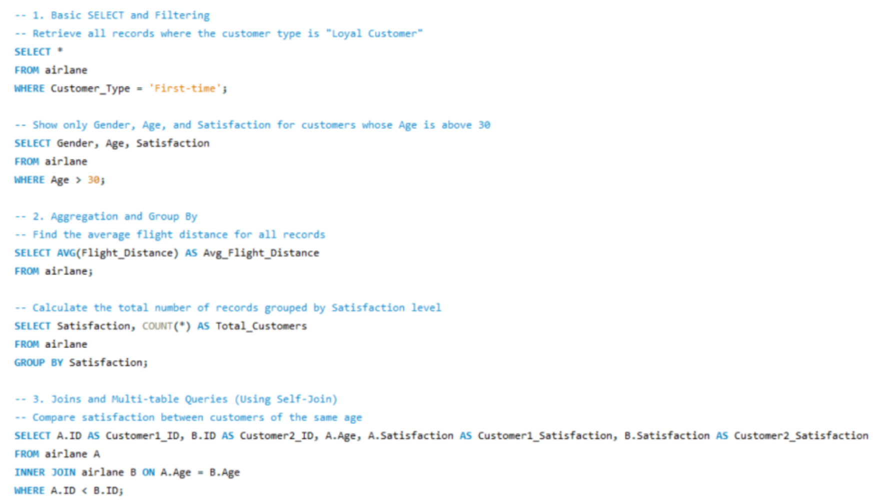
Feature Importance: Models like Decision Tree and Random Forest highlight key drivers of satisfaction such as *Type of Travel* and *Inflight Service Ratings*.

Non-Linear Models: Algorithms like SVM and Random Forest excel due to their ability to capture non-linear relationships.

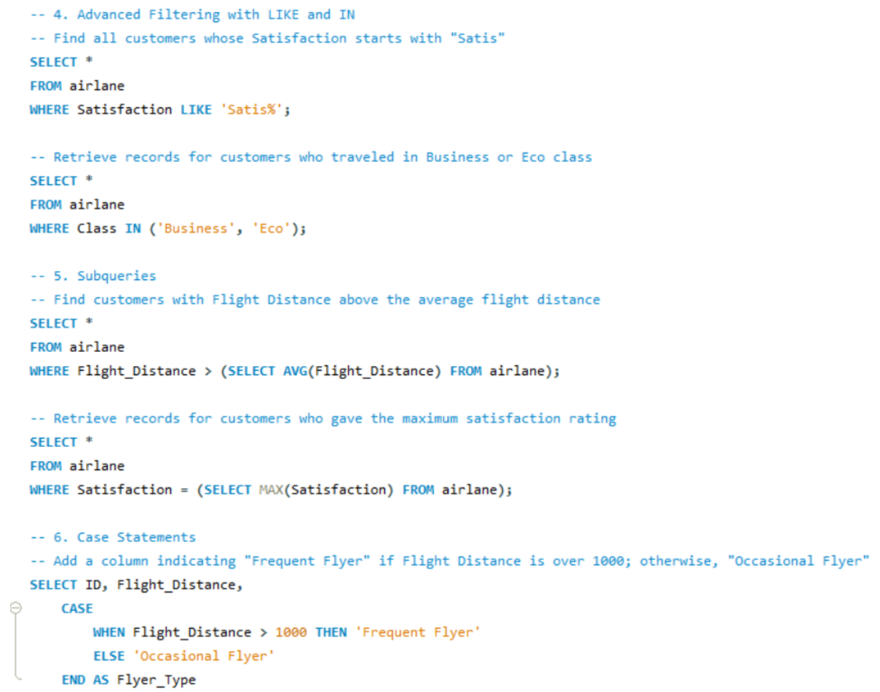
Comparison: Random Forest and SVM outperform others in accuracy but are computationally more expensive.

OBJECTIVE QUERY

- **Understand Passenger Demographics:** Segment customers by type, age, gender, and class to identify patterns and preferences.
- **Analyse Satisfaction Levels:** Evaluate the factors influencing satisfaction, such as travel class, age, and travel type.
- **Measure Performance Metrics:** Calculate averages, totals, and ranks to assess key operational metrics like flight distance, delays, and service quality.
- **Derive Advanced Insights:** Use window functions, subqueries, and case statements to uncover trends in passenger behaviour and satisfaction.
- **Enhance Data Accessibility:** Create views, indexes, and temporary tables for efficient data handling and faster query execution.

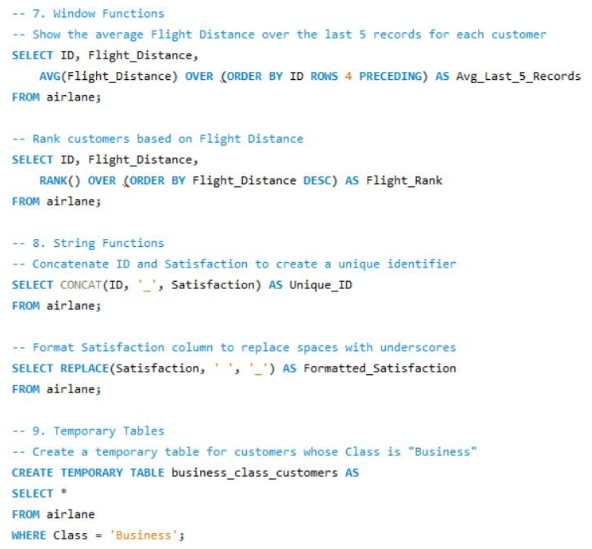


```sql
-- 1. Basic SELECT and Filtering
-- Retrieve all records where the customer type is "Loyal Customer"
SELECT *
FROM airlane
WHERE Customer_Type = 'First-time';

-- Show only Gender, Age, and Satisfaction for customers whose Age is above 30
SELECT Gender, Age, Satisfaction
FROM airlane
WHERE Age > 30;

-- 2. Aggregation and Group By
-- Find the average flight distance for all records
SELECT AVG(Flight_Distance) AS Avg_Flight_Distance
FROM airlane;

-- Calculate the total number of records grouped by Satisfaction level
SELECT Satisfaction, COUNT(*) AS Total_Customers
FROM airlane
GROUP BY Satisfaction;

-- 3. Joins and Multi-table Queries (Using Self-Join)
-- Compare satisfaction between customers of the same age
SELECT A.ID AS Customer1_ID, B.ID AS Customer2_ID, A.Age, A.Satisfaction AS Customer1_Satisfaction, B.Satisfaction AS Customer2_Satisfaction
FROM airlane A
INNER JOIN airlane B ON A.Age = B.Age
WHERE A.ID < B.ID;
```

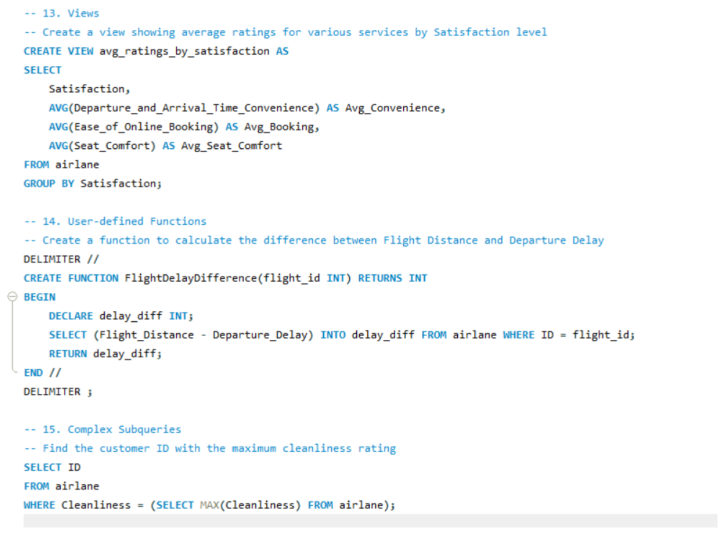


```sql
-- 4. Advanced Filtering with LIKE and IN
-- Find all customers whose Satisfaction starts with "Satis"
SELECT *
FROM airlane
WHERE Satisfaction LIKE 'Satis%';

-- Retrieve records for customers who traveled in Business or Eco class
SELECT *
FROM airlane
WHERE Class IN ('Business', 'Eco');

-- 5. Subqueries
-- Find customers with Flight Distance above the average flight distance
SELECT *
FROM airlane
WHERE Flight_Distance > (SELECT AVG(Flight_Distance) FROM airlane);

-- Retrieve records for customers who gave the maximum satisfaction rating
SELECT *
FROM airlane
WHERE Satisfaction = (SELECT MAX(Satisfaction) FROM airlane);

-- 6. Case Statements
-- Add a column indicating "Frequent Flyer" if Flight Distance is over 1000; otherwise, "Occasional Flyer"
SELECT ID, Flight_Distance,
    CASE
        WHEN Flight_Distance > 1000 THEN 'Frequent Flyer'
        ELSE 'Occasional Flyer'
    END AS Flyer_Type
```

OBJECTIVE QUERY



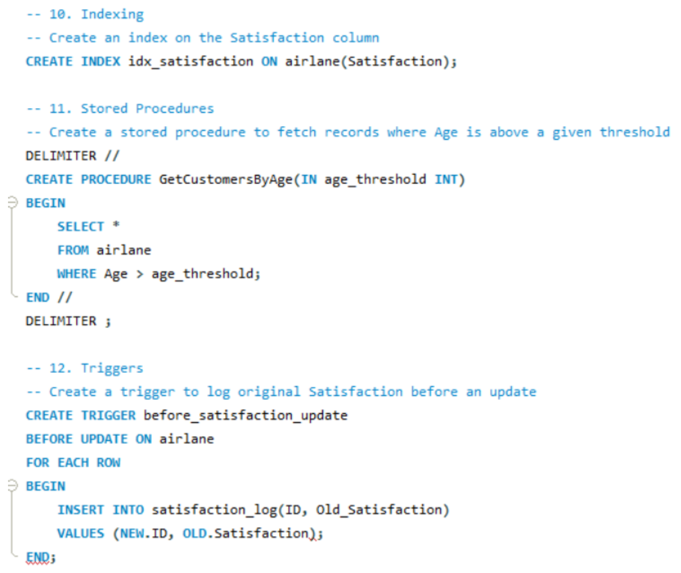
```sql
-- 7. Window Functions
-- Show the average Flight Distance over the last 5 records for each customer
SELECT ID, Flight_Distance,
    AVG(Flight_Distance) OVER (ORDER BY ID ROWS 4 PRECEDING) AS Avg_Last_5_Records
FROM airlane;

-- Rank customers based on Flight Distance
SELECT ID, Flight_Distance,
    RANK() OVER (ORDER BY Flight_Distance DESC) AS Flight_Rank
FROM airlane;

-- 8. String Functions
-- Concatenate ID and Satisfaction to create a unique identifier
SELECT CONCAT(ID, '_', Satisfaction) AS Unique_ID
FROM airlane;

-- Format Satisfaction column to replace spaces with underscores
SELECT REPLACE(Satisfaction, ' ', '_') AS Formatted_Satisfaction
FROM airlane;

-- 9. Temporary Tables
-- Create a temporary table for customers whose Class is "Business"
CREATE TEMPORARY TABLE business_class_customers AS
SELECT *
FROM airlane
WHERE Class = 'Business';
```



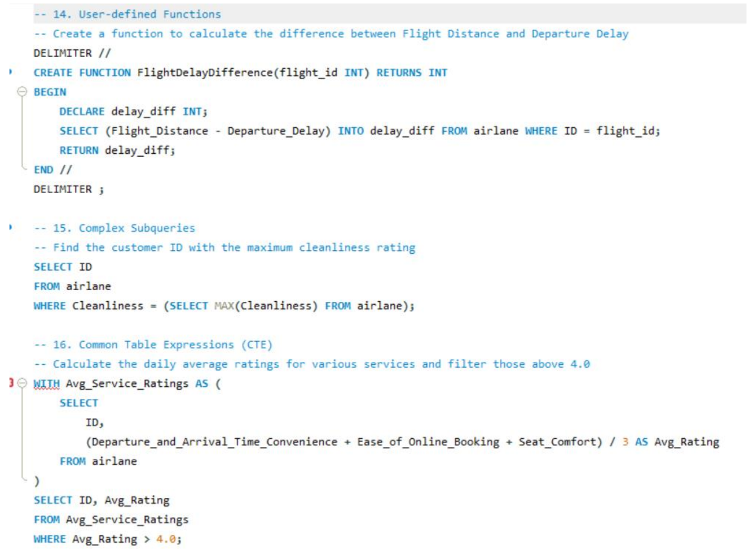
```sql
-- 13. Views
-- Create a view showing average ratings for various services by Satisfaction level
CREATE VIEW avg_ratings_by_satisfaction AS
SELECT
    Satisfaction,
    AVG(Departure_and_Arrival_Time_Convenience) AS Avg_Convenience,
    AVG(Ease_of_Online_Booking) AS Avg_Booking,
    AVG(Seat_Comfort) AS Avg_Seat_Comfort
FROM airlane
GROUP BY Satisfaction;

-- 14. User-defined Functions
-- Create a function to calculate the difference between Flight Distance and Departure Delay
DELIMITER //
CREATE FUNCTION FlightDelayDifference(flight_id INT) RETURNS INT
BEGIN
    DECLARE delay_diff INT;
    SELECT (Flight_Distance - Departure_Delay) INTO delay_diff FROM airlane WHERE ID = flight_id;
    RETURN delay_diff;
END //
DELIMITER ;

-- 15. Complex Subqueries
-- Find the customer ID with the maximum cleanliness rating
SELECT ID
FROM airlane
WHERE Cleanliness = (SELECT MAX(Cleanliness) FROM airlane);
```

Conclusion

These SQL queries form a foundational layer for data-driven decision-making in the airline passenger satisfaction project. By leveraging them, stakeholders can identify satisfaction drivers, optimize operations, and personalize customer experiences effectively.

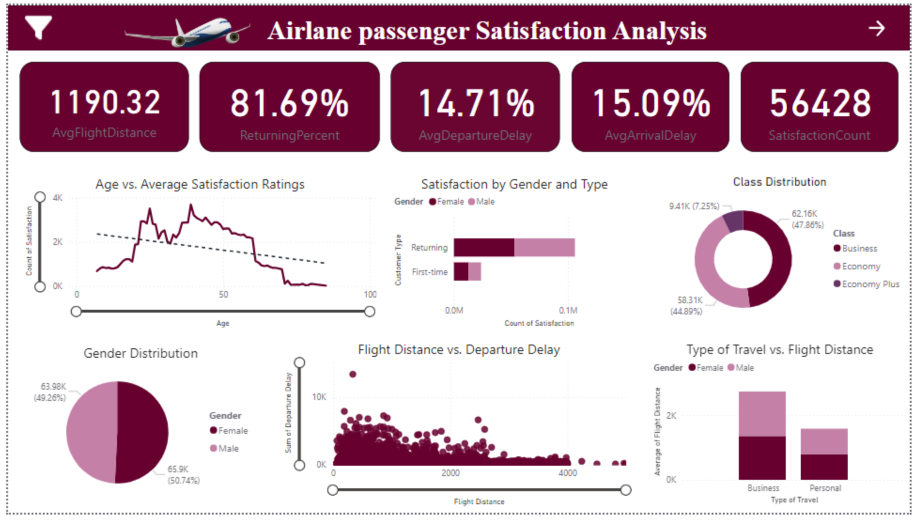


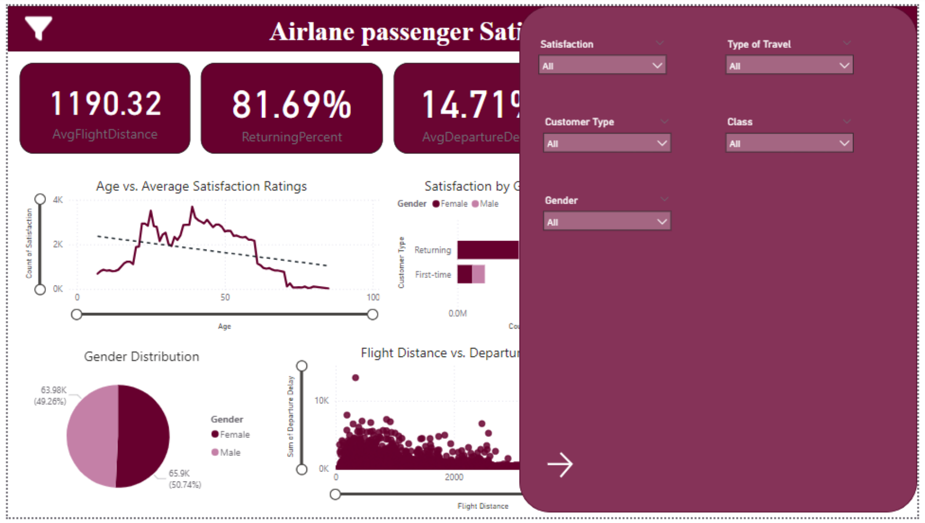
```sql
-- 10. Indexing
-- Create an index on the Satisfaction column
CREATE INDEX idx_satisfaction ON airlane(Satisfaction);

-- 11. Stored Procedures
-- Create a stored procedure to fetch records where Age is above a given threshold
DELIMITER //
CREATE PROCEDURE GetCustomersByAge(IN age_threshold INT)
BEGIN
    SELECT *
    FROM airlane
    WHERE Age > age_threshold;
END //
DELIMITER ;

-- 12. Triggers
-- Create a trigger to log original Satisfaction before an update
CREATE TRIGGER before_satisfaction_update
BEFORE UPDATE ON airlane
FOR EACH ROW
BEGIN
    INSERT INTO satisfaction_log(ID, Old_Satisfaction)
    VALUES (NEW.ID, OLD.Satisfaction);
END;
```



```sql
-- 14. User-defined Functions
-- Create a function to calculate the difference between Flight Distance and Departure Delay
DELIMITER //
CREATE FUNCTION FlightDelayDifference(flight_id INT) RETURNS INT
BEGIN
    DECLARE delay_diff INT;
    SELECT (Flight_Distance - Departure_Delay) INTO delay_diff FROM airlane WHERE ID = flight_id;
    RETURN delay_diff;
END //
DELIMITER ;

-- 15. Complex Subqueries
-- Find the customer ID with the maximum cleanliness rating
SELECT ID
FROM airlane
WHERE Cleanliness = (SELECT MAX(Cleanliness) FROM airlane);

-- 16. Common Table Expressions (CTE)
-- Calculate the daily average ratings for various services and filter those above 4.0
WITH Avg_Service_Ratings AS (
    SELECT
        ID,
        (Departure_and_Arrival_Time_Convenience + Ease_of_Online_Booking + Seat_Comfort) / 3 AS Avg_Rating
    FROM airlane
)
SELECT ID, Avg_Rating
FROM Avg_Service_Ratings
WHERE Avg_Rating > 4.0;
```

POWER BI DASHBOARD

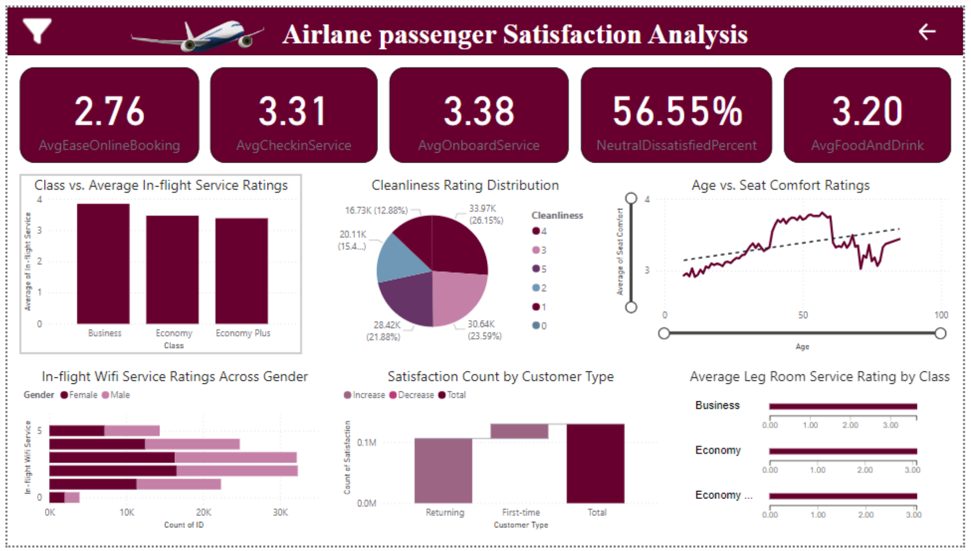


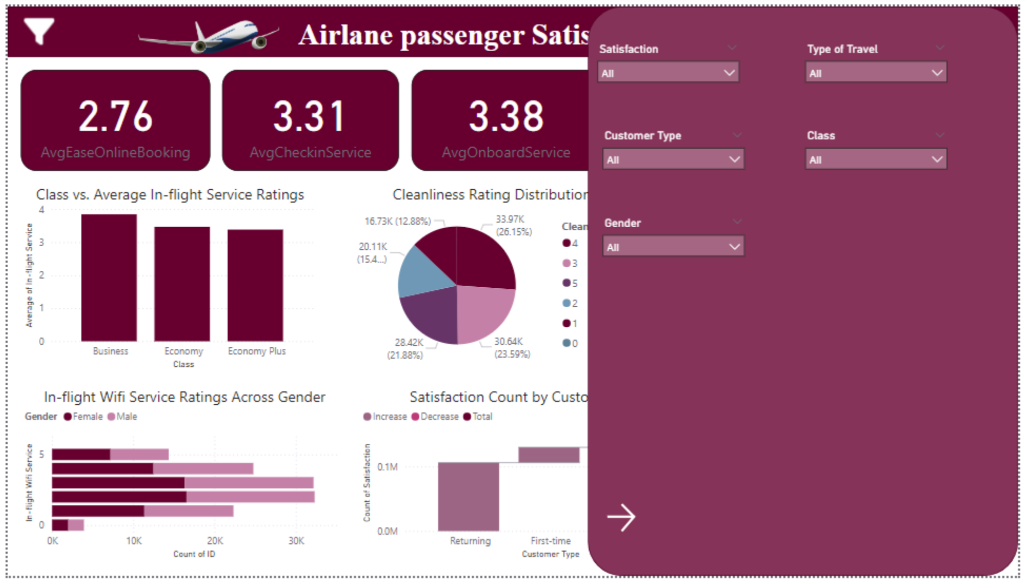


<u>Smart narrative of Page 1</u>

At 3,692, 39 had the highest Count of Satisfaction and was 14,668.00% higher than 85, which had the lowest Count of Satisfaction at 25. 39 had the highest Count of Satisfaction at 3,692, followed by 25 and 40. 85 had the lowest Count of Satisfaction at 25. 39 accounted for 2.84% of Count of Satisfaction. Total Count of Satisfaction was higher for Female (65,899) than Male (63981). Returning in Gender Female made up 40.85% of Count of Satisfaction. Average Count of Satisfaction was higher for Female (32,949.50) than Male (31,990.50). Average of Flight Distance for Male and Female diverged the most when the Type of Travel was Business, when Male were 22.10 higher than Female.

POWER BI DASHBOARD


Airlane passenger Satisfaction Analysis
2.76
AvgEaseOnlineBooking
3.31
AvgCheckinService
3.38
AvgOnboardService
56.55%
NeutralDissatisfiedPercent
3.20
AvgFoodAndDrink
Class vs. Average In-flight Service Ratings
Cleanliness Rating Distribution
Age vs. Seat Comfort Ratings
In-flight Wifi Service Ratings Across Gender
Satisfaction Count by Customer Type
Average Leg Room Service Rating by Class
Business
Economy
Economy Plus


Airlane passenger Satis
2.76
AvgEaseOnlineBooking
3.31
AvgCheckinService
3.38
AvgOnboardService
Class vs. Average In-flight Service Ratings
Cleanliness Rating Distribution
In-flight Wifi Service Ratings Across Gender
Satisfaction Count by Custo

Satisfaction: All
Type of Travel: All
Customer Type: All
Class: All
Gender: All

<u>Smart narrative of Page 2</u>

At 3.81, 57 had the highest Average of Seat Comfort and was 30.81% higher than 11, which had the lowest Average of Seat Comfort at 2.91. Across all 75 Age, Average of Seat Comfort ranged from 2.91 to 3.81. 4 accounted for 26.15% of Count of ID. Total Count of ID was higher for Female (65,899) than Male (63981). 2 in Gender Female made up 12.74% of Count of ID. Average Count of ID was higher for Female (10,983.17) than Male (10,663.50).

CONCLUSION & SUGGESTIONS

This project successfully analyses airline passenger satisfaction through data pre-processing, machine learning models, and SQL analysis. Key findings include:

- **Machine Learning Models**: Models like Random Forest, Decision Tree, and Logistic Regression showed high accuracy in predicting passenger satisfaction, indicating their reliability for real-world use.
- **SQL Analysis**: SQL queries provided valuable insights into customer demographics and service performance, highlighting areas for improvement.
- **Advanced Data Handling**: Techniques like subqueries and window functions enabled detailed segmentation and customer profiling.

Suggestions for Improvement:

Data Enrichment: Integrate external data (e.g., weather, flight delays) for better prediction accuracy.
Model Fine-Tuning: Optimize models with hyper parameter tuning for improved performance.
Deeper EDA: Further analyse hidden patterns to uncover insights for customer experience improvement.
Real-Time Data: Implement a system for real-time data analysis and predictions.

KEY RECOMMENDATION

- Customer Segmentation:

Create targeted marketing campaigns and personalized offerings based on customer segments (e.g., frequent vs. occasional travellers).Enhance loyalty programs with tailored rewards like exclusive benefits for loyal customers.

- Operational Optimization:

Focus on improving flight punctuality and reducing delays using predictive models. Streamline the booking process and improve baggage handling with automated systems.

- Employee Training:

Provide service quality training to frontline staff to ensure a customer-centric mind-set. Equip employees with crisis management skills to handle issues effectively.

- Feedback Integration:

Conduct regular surveys and utilize AI tools to analyse customer feedback for continuous service improvement. Use actionable feedback to inform operational changes and improve customer satisfaction.

- Personalization:

Leverage customer data to offer personalized services, such as tailored in-flight experiences and custom offers. Create dynamic customer profiles to enhance service personalization and anticipate needs.

- Technology and Innovation:

Implement Chabot's and virtual assistants to handle customer inquiries efficiently. Explore biometric check-ins, IoT for real-time monitoring, and other technologies to streamline operations and enhance customer experience.

Sustainability:

Adopt eco-friendly practices and offer carbon-offset programs to appeal to environmentally conscious customers. Modernize the fleet for better fuel efficiency and environmental sustainability.

THANK YOU!

VISHNU G NATH

https://www.linkedin.com/in/imvishnugnath/

vishnugs32@gmail.com

https://github.com/Vishnugnath/