DATA SCIENCE PROJECT 2 – Airline Passenger satisfaction Prediction Project

Customer Satisfaction Prediction Data Science Project

Project:	Customer Satisfaction Prediction
Tools Used:	Jupyter Notebook
Technologies:	Machine Learning, Data Analytics
Difficulty Level:	Intermediate to Advanced
Libraries:	Pandas, NumPy, Matplotlib, Seaborn, Scikit-Learn, XGBoost, CatBoost
Dataset:	https://www.kaggle.com/datasets/teejmahal20/airline-passenger-satisfaction/data
GitHub Link:	https://github.com/Vicjosh07/customer_satisfaction_project

- **About Dataset**

Context

This dataset contains an airline passenger satisfaction survey. What factors are highly correlated to a satisfied (or dissatisfied) passenger? Can you predict passenger satisfaction

Content

Gender: Gender of the passengers (Female, Male)
Customer Type: The customer type (Loyal customer, disloyal customer)
Age: The actual age of the passengers
Type of Travel: Purpose of the flight of the passengers (Personal Travel, Business Travel)
Class: Travel class in the plane of the passengers (Business, Eco, Eco Plus)
Flight distance: The flight distance of this journey
Inflight wifi service: Satisfaction level of the inflight wifi service (0:Not Applicable;1-5)
Departure/Arrival time convenient: Satisfaction level of Departure/Arrival time convenient
Ease of Online booking: Satisfaction level of online booking
Gate location: Satisfaction level of Gate location
Food and drink: Satisfaction level of Food and drink
Online boarding: Satisfaction level of online boarding
Seat comfort: Satisfaction level of Seat comfort
Inflight entertainment: Satisfaction level of inflight entertainment
On-board service: Satisfaction level of On-board service

Leg room service: Satisfaction level of Leg room service
Baggage handling: Satisfaction level of baggage handling
Check-in service: Satisfaction level of Check-in service
Inflight service: Satisfaction level of inflight service
Cleanliness: Satisfaction level of Cleanliness
Departure Delay in Minutes: Minutes delayed when departure
Arrival Delay in Minutes: Minutes delayed when Arrival
Satisfaction: Airline satisfaction level (Satisfaction, neutral or dissatisfaction)

	id	Gender	Customer T	Age	Type of Tra	Class	Flight Dista	Inflight wif	Departure/	Ease of Onl	Gate locati	Food and d	Online boa	Seat comfo	Inflight ent	On-board s	Leg room s
0	19556	Female	Loyal Custo	52	Business tr	Eco	160	5	4	3	4	3	4	3	5	5	5
1	90035	Female	Loyal Custo	36	Business tr	Business	2863	1	1	3	1	5	4	5	4	4	4
2	12360	Male	disloyal Cu	20	Business tr	Eco	192	2	0	2	4	2	2	2	2	4	1
3	77959	Male	Loyal Custo	44	Business tr	Business	3377	0	0	0	2	3	4	4	1	1	1
4	36875	Female	Loyal Custo	49	Business tr	Eco	1182	2	3	4	3	4	1	2	2	2	2
5	39177	Male	Loyal Custo	16	Business tr	Eco	311	3	3	3	3	5	5	3	5	4	3
6	79433	Female	Loyal Custo	77	Business tr	Business	3987	5	5	5	5	3	5	5	5	5	5
7	97286	Female	Loyal Custo	43	Business tr	Business	2556	2	2	2	2	4	4	5	4	4	4
8	27508	Male	Loyal Custo	47	Business tr	Eco	556	5	2	2	2	5	5	5	5	2	2
9	62482	Female	Loyal Custo	46	Business tr	Business	1744	2	2	2	2	3	4	4	4	4	4
10	47583	Female	Loyal Custo	47	Business tr	Eco	1235	4	1	1	1	5	1	5	3	3	4
11	115550	Female	Loyal Custo	33	Business tr	Business	325	2	5	5	5	1	3	4	2	2	2
12	119987	Female	Loyal Custo	46	Business tr	Business	1009	5	5	5	5	4	5	5	5	5	5

Baggage ha	Checkin ser	Inflight ser	Cleanliness	Departure I	Arrival Dela	satisfaction
5	2	5	5	50	44	satisfied
4	3	4	5	0	0	satisfied
3	2	2	2	0	0	neutral or dissatisfied
1	3	1	4	0	6	satisfied
2	4	2	4	0	20	satisfied
1	1	2	5	0	0	satisfied
5	4	5	3	0	0	satisfied
4	5	4	3	77	65	satisfied
5	3	3	5	1	0	satisfied
4	5	4	4	28	14	satisfied
3	1	3	4	29	19	satisfied
2	3	2	4	18	7	neutral or dissatisfied
5	5	5	3	0	0	satisfied
5	3	5	5	117	113	satisfied

1. Introduction

Concept

This project aims to build a machine learning model to predict customer satisfaction based on survey and service data. The model classifies customers as satisfied or dissatisfied/Neutral while identifying key factors influencing satisfaction ***for airline passengers.***

Use Cases

- Businesses can proactively improve ***airline passenger's*** experience by targeting at-risk customers.
- Helps airlines understand the impact of different service attributes on satisfaction.
- Enhances decision-making in marketing, sales, and customer support.

Problem Addressed

- Understanding passenger's pain points and areas of service improvement.
- Creating an accurate predictive model for satisfaction classification.
- Handling missing data and categorical feature encoding effectively.

---

2. Data Collection & Preprocessing

Dataset

- **Sources:** I sourced for the dataset from kaggle.com which had an airline passenger's satisfaction dataset and was already cleaned and split into *train.csv* and *test.csv* files.
- **Features:** The dataset Included customer demographics, transaction history, and feedback metrics.
- **Target Variable:** The Satisfaction level was encoded for classification into two. (either satisfied or neutral/dissatisfied)

Dataset Overview (First Rows):

```
df_train.head(3)
```

	Gender	Customer Type	Age	Type of Travel	Class	Flight Distance	Inflight wifi service	Departure/Arrival time convenient	Ease of Online booking	Gate location	...	Inflight entertainment	On-board service	Leg room service	Baggage handling	Checkin service	I
0	Male	Loyal Customer	13	Personal Travel	Eco Plus	460	3	4	3	1	...	5	4	3	4	4	
1	Male	disloyal Customer	25	Business travel	Business	235	3	2	3	3	...	1	1	5	3	1	
2	Female	Loyal Customer	26	Business travel	Business	1142	2	2	2	2	...	5	4	3	4	4	

3 rows × 23 columns

Inflight service	Cleanliness	Departure Delay in Minutes	Arrival Delay in Minutes	satisfaction
5	5	25	18.0	0
4	1	1	6.0	0
4	5	0	0.0	1
4	2	11	9.0	0
3	3	0	0.0	1

2.1 Data Cleaning/Feature Engineering

- Dropped irrelevant columns (first two columns) to focus on the more relevant columns.
- Handled missing values:
  - I also Removed rows where Arrival Delay in Minutes was nulled (393 rows dropped, since the 393 rows in the Arrival Delay in Minutes were negligible to the overall performance of the model)

```
[13] df.dropna(inplace =True)

     df.shape

     (129487, 23)
```

2.2 Encoding Categorical Features

- Identified categorical columns and applied *LabelEncoder*.
- Printed the number of unique values in each categorical feature to confirm the different categories in each columns.

```
[18] print("Number of Uniques in the encoded data columns:")
     for i in object_col:
         num_uni = df[i].nunique()
         print(f"{i}: {num_uni}")
```

```
Number of Uniques in the encoded data columns:
Gender: 2
Customer Type: 2
Type of Travel: 2
Class: 3
satisfaction: 2
```

```
[26] encode = LabelEncoder()
     # Apply LabelEncoder to each column
     for col in object_col:
         df[col] = encode.fit_transform(df[col])
```

```
[27] df.head()
```

	Gender	Customer Type	Age	Type of Travel	Class	Flight Distance	Inflight wifi service	Departure/Arrival time convenient	Ease of Online booking	Gate location	...	Inflight entertainment	On-board service	Leg room service	Baggage handling	Checkin service	Inflight service
0	1	0	13	1	2	460	3	4	3	1	...	5	4	3	4	4	5
1	1	1	25	0	0	235	3	2	3	3	...	1	1	5	3	1	4
2	0	0	26	0	0	1142	2	2	2	2	...	5	4	3	4	4	4
3	0	0	25	0	0	562	2	5	5	5	...	2	2	5	3	1	4
4	1	0	61	0	0	214	3	3	3	3	...	3	3	4	4	3	3

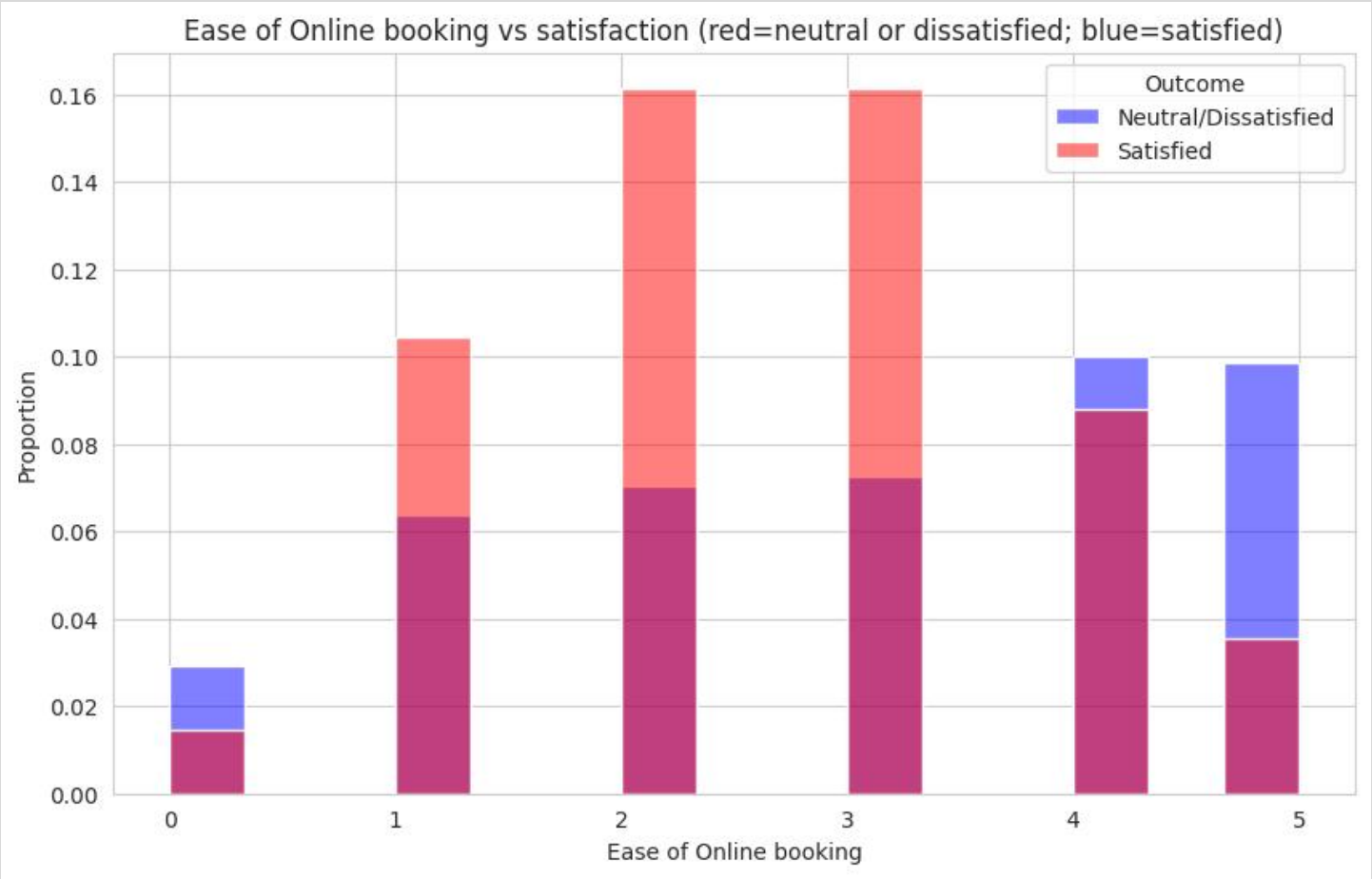
5 rows × 23 columns

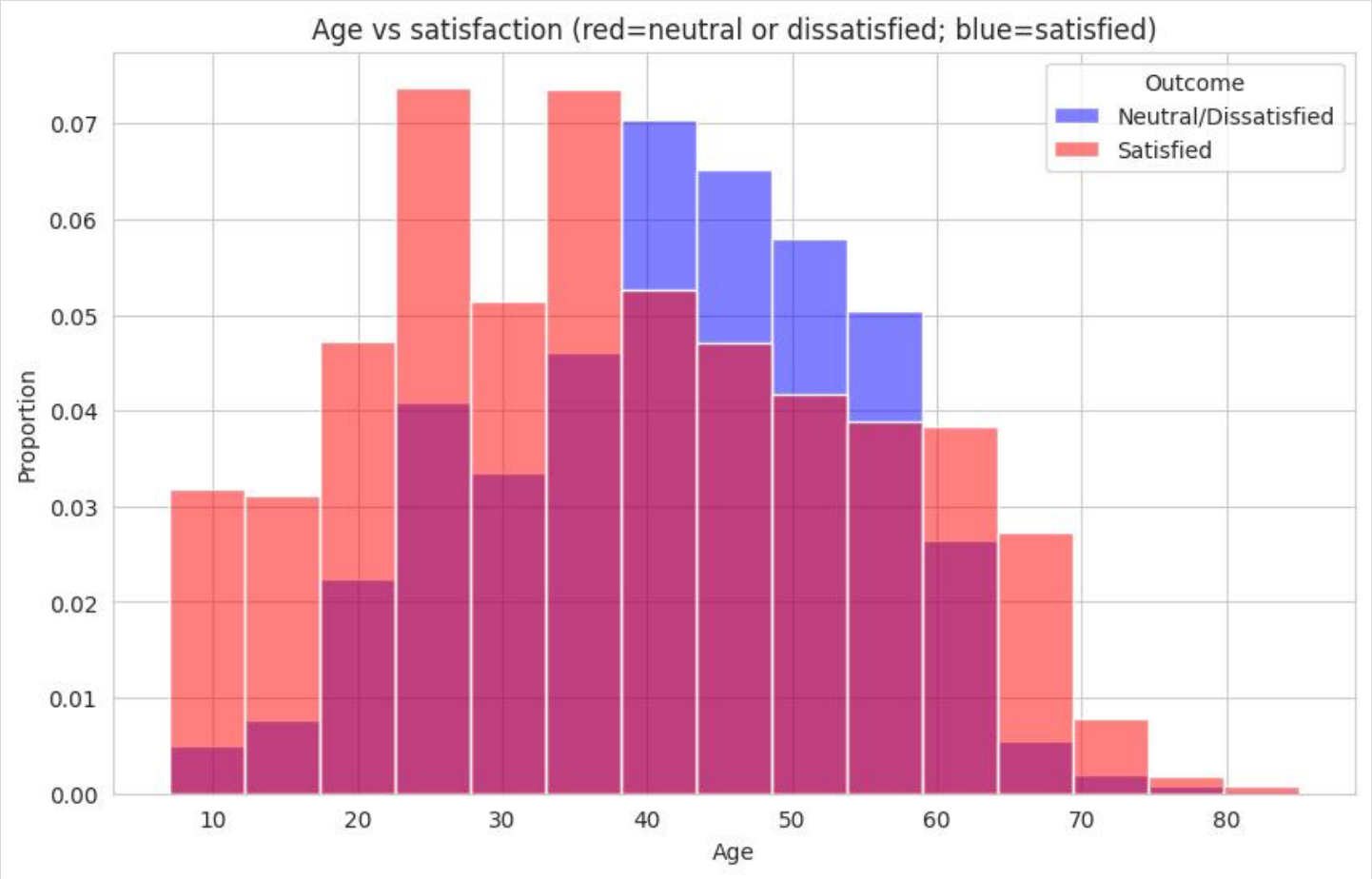
✓ 1s completed at 11:32 AM

3. Exploratory Data Analysis (EDA)

3.1 Feature Distribution

- To get an idea of the different distributions I Plotted histograms of key features to analyze distributions.





+ Code + Text

```python
# Define the function to plot histograms (outside the loop)
def plotHistogram(data, feature, label, title):
    sns.set_style("whitegrid")

    # Define a palette mapping
    custom_palette = {0: 'red', 1: 'blue'}

    # Plot the histogram
    plt.figure(figsize=(10, 6))
    sns.histplot(
        data=data,
        x=feature,
        hue=label,
        kde=False,
        bins=15,
        palette=custom_palette,
        alpha=0.5,
        stat="probability"  # Display as proportions
    )
    plt.title(title)
    plt.xlabel(feature)
    plt.ylabel('Proportion')
    plt.legend(labels=["Neutral/Dissatisfied", "Satisfied"], title="Outcome")
    plt.show()
```

+ Code + Text

```python
# Loop through each feature except the label (Outcome)
for feature in df.columns[:-1]:  # Exclude the last column (assumed label)
    plotHistogram(
        data=df,
        feature=feature,
        label=df.columns[-1],  # The label column (assumed to be the last column)
        title=f"{feature} vs {df.columns[-1]} (red=neutral or dissatisfied; blue=satisfied)"
    )
```

4. Data Splitting & Scaling

- I Splitted the dataset into training (80%) and testing (20%) sets.
- Then, I Applied *StandardScaler* for feature normalization. As to avoided bias due to large numerical features in some columns.

```
[17] x = df[df.columns[:-1]]  # All columns except the last one
     y = df[df.columns[-1]]   # The last column
     x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)

[18] scaler = StandardScaler()
     x_train = scaler.fit_transform(x_train)
     x_test = scaler.fit_transform(x_test)
```

5. Model Training & Evaluation

Models Used: I looped through and tested different models to pick the most optimal with different hyper–parameters; The following were Explored: Decision Tree, SVM, Logistic Regression, Random Forest, K-Nearest Neighbors, Gradient Boosting, Naive Bayes, XGB, CatBoost.

```
7m
# Models to test with hyperparameters
models = {
    "Decision Tree": DecisionTreeClassifier(random_state=0),
    "SVM": SVC(random_state=1, class_weight='balanced'),
    "Logistic Regression": LogisticRegression(random_state=42),
    "Random Forest Classifier": RandomForestClassifier(max_depth=25, random_state=42),
    "K-Nearest Neighbors": KNeighborsClassifier(n_neighbors=7),
    "Gradient Boosting": GradientBoostingClassifier(random_state=1),
    "Naive Bayes": GaussianNB(),
    "XGB": xgb.XGBClassifier(random_state=42),
    "CatBoost": CatBoostClassifier(iterations=500, random_state=42, learning_rate=0.1),
}

# Train and evaluate each model
for model_name, model in models.items():
    print(f"Training {model_name}...")

    # Train the model
    model.fit(x_train, y_train)
    y_pred = model.predict(x_test)
```

+ Code + Text

```python
# Model evaluation
print(f"{model_name} Model Performance:")
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))

# Visualize the top 10 feature importances
plt.figure(figsize=(8, 5))  # Set figure size

# Ensure x_train is a DataFrame before extracting feature names
if isinstance(x_train, np.ndarray):
    x_train_df = pd.DataFrame(x_train, columns=x.columns)  # Convert back to DataFrame
else:
    x_train_df = x_train  # Keep it as is if already a DataFrame

# Extract actual feature names
feature_names = x_train_df.columns

if hasattr(model, "feature_importances_"):
    feature_importances = pd.Series(model.feature_importances_, index=feature_names)
    feature_importances.nlargest(10).plot(kind='barh', color="skyblue")
    plt.title(f'Top 10 Feature Importances - {model_name}')
    plt.xlabel('Importance Score')
    plt.ylabel('Feature')
    plt.show()
```

Connected to Python 3 Google Compute Engine backend

+ Code + Text

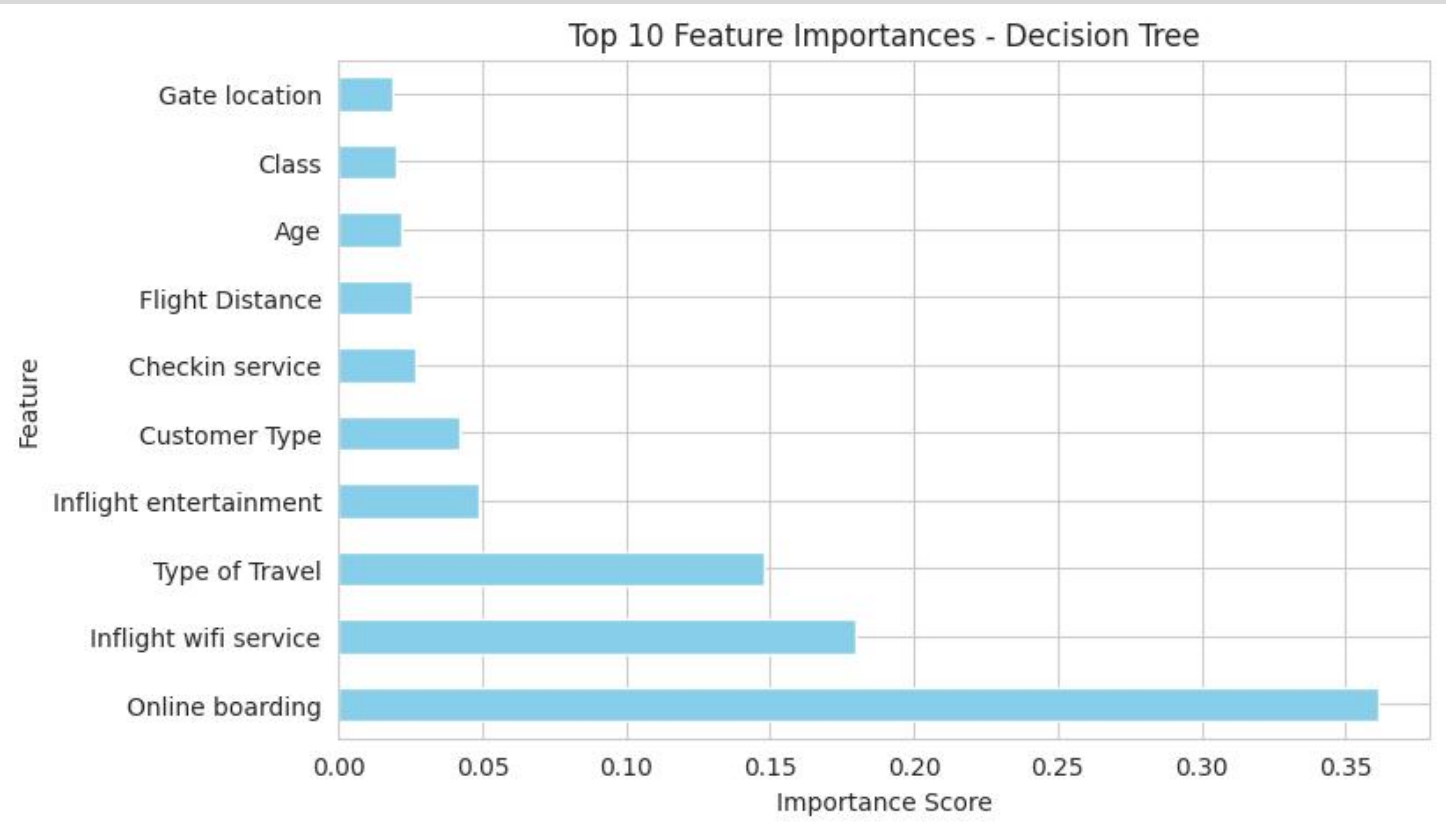
```python
elif hasattr(model, "coef_"):  # For models like Logistic Regression
    coefficients = pd.Series(abs(model.coef_[0]), index=feature_names)
    coefficients.nlargest(10).plot(kind='barh', color="orange")
    plt.title(f'Top 10 Feature Importances (Coefficients) - {model_name}')
    plt.xlabel('Coefficient Magnitude')
    plt.ylabel('Feature')
    plt.show()

else:
    print(f"{model_name} does not support feature importances directly.")
```

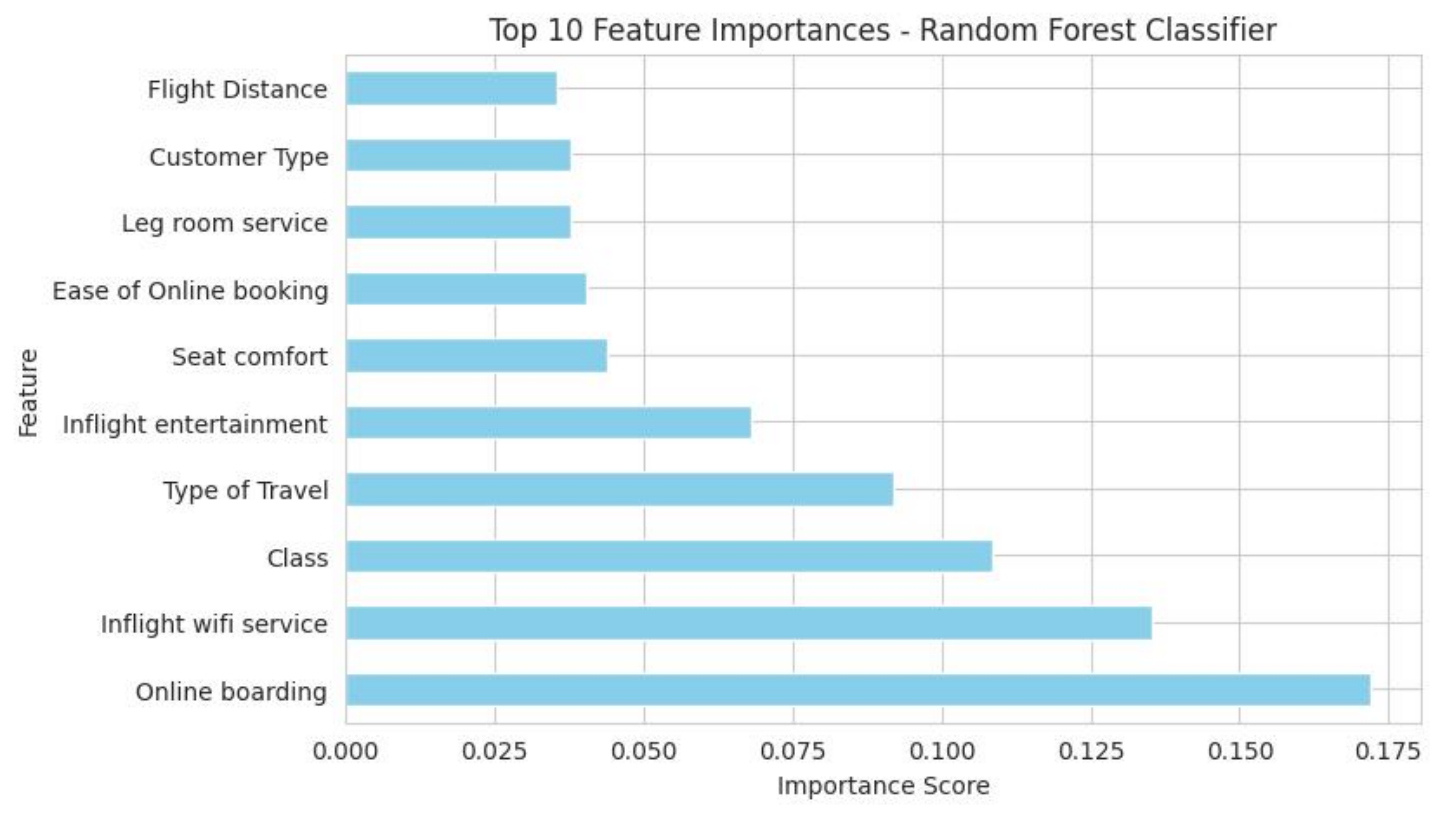
```
Accuracy: 0.9474476793574793
Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.95      0.95     14668
           1       0.94      0.94      0.94     11230

    accuracy                           0.95     25898
   macro avg       0.95      0.95      0.95     25898
weighted avg       0.95      0.95      0.95     25898

Confusion Matrix:
 [[13979   689]
 [  672 10558]]
```

5.1 Model Performance Comparison

MODEL	ACCURACY (%)	PRECISION (AVG)	RECALL (AVG)	F1-SCORE (AVG)
Random Forest	96%	0.96	0.96	0.96
Support Vector Machine (SVM)	95%	0.95	0.95	0.95
Gradient Boosting	94%	0.94	0.94	0.94
Logistic Regression	88%	0.97	0.87	0.87
K-Nearest Neighbors (KNN)	93%	0.93	0.92	0.93
Naive Bayes	86%	0.86	0.86	0.86
XGBoost	89%	0.89	0.88	0.89
CatBoost	**96%**	**0.97**	**0.96**	**0.96**
Decision Tree	95%	0.95	0.95	0.95

5.2 Visualizing the top 10 Features Importance in each model





6. Challenges Faced

The major challenge I faced was while selecting the parameters to use during the plotting of the histogram. Picking a visualization between the features distributions was a bit hard, because I had to look up different documentation to get that snippet of code.

---

7. Future Improvements

Some of the improvements I recommend is:

- Deploying the model as an API for real-time predictions.
- Design a recommendation algorithm to suggest improvements based on predicted satisfaction levels.
- Example suggestions:
  - **Dissatisfied customers:** Offer better customer service follow-ups.
  - **Neutral customers:** Introduce personalized incentives and promotions.
  - **Satisfied customers:** Encourage brand advocacy through referrals.

---

8. Conclusion

I was able to successfully build a customer satisfaction prediction model using multiple machine learning approaches. We can us this in real time predictions where we take different range of features on a ticket and predict the satisfaction level of an airline passengers. I am hoping for Future enhancements focusing on deployment and real-time analytics.

Olaleye Joshua - UMIP271961