Fabian IACOB

Mi412

Second session

Mathematical tools for data science

4SM 2023-2024

Content

Introduction

Following the pandemic, the airline industry suffered from a massive setback. To revitalize the industry, it is necessary to understand customer pain points and improve their satisfaction with the services provided. This project aims to predict passenger satisfaction levels based on survey data.

The goal is to find an adequate Machine Learning model capable of classifying what factors are highly correlated to a satisfied (or dissatisfied) passenger and predicting the latter' satisfaction.

Data analysis

The dataset consists of survey responses from air passengers with various features like “Gender”, “Customer Type”, “Age”, “Type of Travel”, “Class”, service ratings (“Inflight wifi service”, “Food and drink”), and delay times (“Departure Delay in Minutes”, “Arrival Delay in Minutes”). The train dataset has 103904 entries, and the test dataset has 25976 entries. The target variable is “Satisfaction” with two levels: “satisfied”, and “neutral or dissatisfied”.

Initial inspection of the datasets showed no missing values except for the “Arrival Delay in Minutes” column. The features were a mix of categorical and numerical data.

Data processing

We used the median strategy to impute the missing values of the “Arrival Delay in Minutes” column.

```
# Fix missing values for 'Arrival Delay in Minutes'
imputer = SimpleImputer(strategy='median')
train_data['Arrival Delay in Minutes'] = imputer.fit_transform(train_data[['Arrival Delay in Minutes']])
test_data['Arrival Delay in Minutes'] = imputer.transform(test_data[['Arrival Delay in Minutes']])
```

We used LabelEncoder to encode categorical variables: “Gender”, “Customer Type”, “Type of Travel”, and “Class”.

```
# Encode categorical variables
categorical_features = ['Gender', 'Customer Type', 'Type of Travel', 'Class']

# Using Label Encoding for simplicity
label_encoders = {}
for feature in categorical_features:
    le = LabelEncoder()
    train_data[feature] = le.fit_transform(train_data[feature])
    test_data[feature] = le.transform(test_data[feature])
    label_encoders[feature] = le
```

We used StandardScaler to standardize numerical features to ensure equal contribution to the model.

```
# Normalize numerical features
scaler = StandardScaler()
train_data[numerical_features] = scaler.fit_transform(train_data[numerical_features])
test_data[numerical_features] = scaler.transform(test_data[numerical_features])
```

Model selection and evaluation

Model selection

We have chosen to try four machine learning models for this project:

• Logistic regression

• K-Nearest Neighbors (KNN)

• Support Vector Machine (SVM)

• Random Forest

We spit the train dataset into training and validation sets and each model was trained and evaluated based on its accuracy and classification report.

Evaluation: Logistic regression

• Accuracy: **0.8777**

• Strengths: simple, performs well on linearly separable data.

• Weaknesses: assumes linearity, can underperform with more complex relationships.

Evaluation: K-Nearest Neighbors (KNN)

• Accuracy: **0.9272**

• Strengths: simple and intuitive, effective with a small amount of data.

• Weaknesses: computationally expensive, sensitive to irrelevant features and the choice of k.

Evaluation: Support Vector Machine (SVM)

• Accuracy: **0.9526**

• Strengths: effective in high-dimensional spaces, robust to overfitting.

• Weaknesses: memory-intensive, less effective on larger datasets.

Evaluation: Random Forest

• Accuracy: **0.9621**

• Strengths: high accuracy, robust to overfitting, handles both numerical and categorical data well.

• Weaknesses: can be computationally intensive, less interpretable than simpler models.

Comparison of models

• **Random Forest:** achieved the highest accuracy (0.9621) and provided well-balanced performance across satisfaction levels.

• **SVM:** Good performance with slightly less accuracy compared to Random Forest.

• **K-Nearest Neighbors**: Decent performance but lower accuracy.

• **Logistic Regression:** Lowest accuracy among the models.

So, we have chosen the Random Forest model as the final model due to its superior performance.

Feature importance analysis

To understand which features (or factors) contribute the most to passenger satisfaction prediction, we analyzed feature importance using the Random Forest model as it was the most precise one in previous tests.

The feature importances were extracted and visualized as follows.

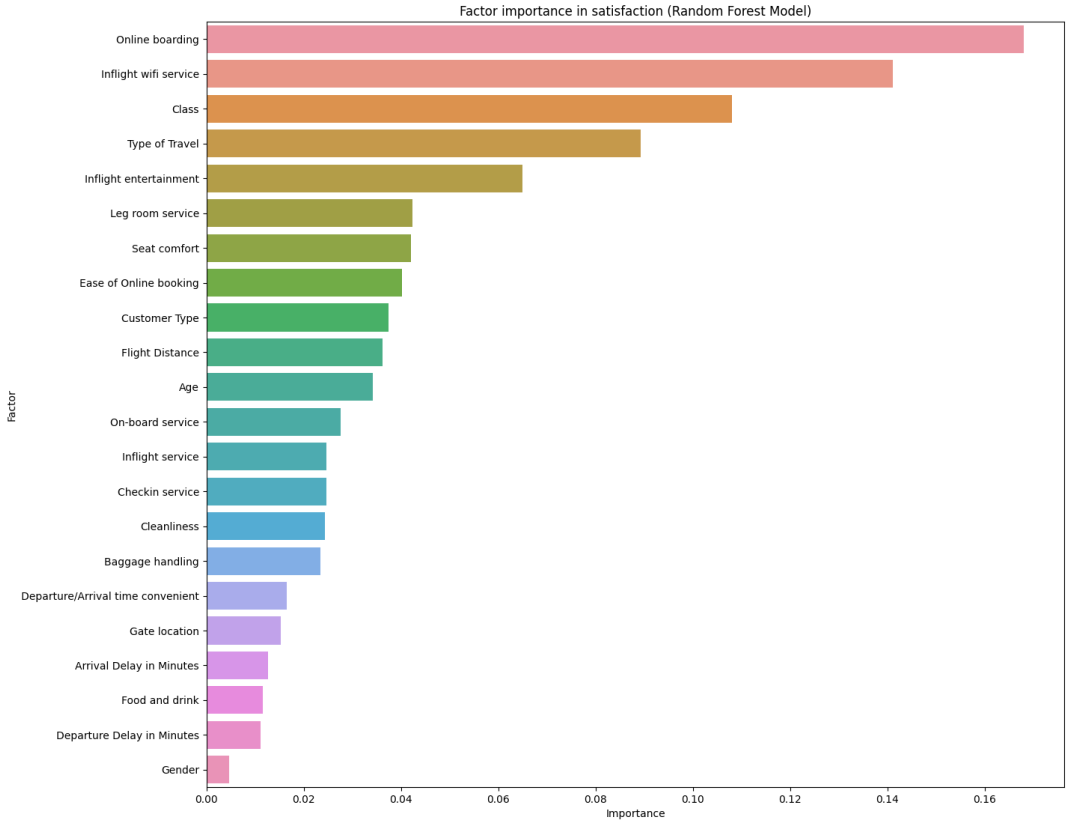
Extract Feature Importance:

The importance of each feature was obtained from the trained Random Forest model. These importance scores help in understanding the influence of each feature on the model's predictions.

Visualize Feature Importance:

We created a bar plot to visualize the importance of each feature, providing a clear view of the most and least influential factors.

Here is the diagram of factors sorted by their importance:



Factor importance in satisfaction

This visualization highlights that the top features influencing passenger satisfaction include online boarding, inflight wifi service, class, type of travel, and inflight entertainment. These insights can guide airlines in prioritizing improvements in these areas to enhance overall passenger satisfaction.

Conclusion

We identified Random Forest as the best model for predicting passenger satisfaction. We successfully processed the data and trained several models. Random Forest model achieved high accuracy and balanced performance across satisfaction levels and it permitted to identify key factors influencing passenger satisfaction, such as online boarding, inflight wifi service, class, type of travel.

Appendix

• GitHub: https://github.com/Fabian-Iacob/Ma412_Second_Session_Project