Data Analysis and Classification Report

IMT2021034(PRACHODAY),IMT2021023(KALYAN),IMT2021072(RAM SAI KOUSHIK)

21-10-2023

1 Introduction

This report provides a comprehensive overview of the data preprocessing, exploratory data analysis (EDA), and the machine learning models employed to analyze and predict passenger contentment in the airline industry.

2 Data preprocessing and Exploratory Data analysis

2.1 Removing Duplicate Rows

Duplicate rows in a dataset can introduce biases and inaccuracies in analysis. Therefore, it is essential to identify and remove them. In this step, we remove duplicate rows from the dataset and set the 'id' column as the index.

2.2 Initial Data Inspection

To ensure data quality, we perform an initial inspection by checking for missing values. Identifying columns with null values is crucial for data preprocessing and imputation.

Null Value Percentages

The following column contains null values:

- 'Arrival Delay in Minutes'

We will address these missing values in the subsequent data preprocessing steps.

This initial data inspection allows us to understand the extent of missing data, which will guide our data imputation strategy during preprocessing.

2.3 Data Distribution Visualization

In this section, we present histograms for selected columns to observe the distribution of data. The histograms provide insights into the central tendencies and spread of each variable. The excluded columns ('Unnamed: 0', 'id', 'Departure Delay in Minutes', 'Arrival Delay in Minutes') are not included in the visualizations.

The histograms reveal the distribution patterns for the remaining columns, which will be useful for further analysis and modeling.

2.4 Scatter Plot: Arrival and Departure Delays

In this section, we present scatter plots for the "Arrival Delay in Minutes" and "Departure Delay in Minutes" columns. The scatter plots illustrate the distribution of data points for these two variables.

Most of the values are concentrated in the 0-200 minute range, suggesting a correlation between arrival and departure delays.

This concentration of data points in this range provides valuable insights into the distribution of delays among different flights. It underscores the significance of punctuality and its impact on passenger satisfaction.

2.5 Encoding Categorical Columns

In this section, we'll walk through the process of identifying categorical columns that require encoding and performing one-hot encoding for these columns. The encoding process enhances the compatibility of categorical data for machine learning models.

2.5.1 Identification of Categorical Columns

First, we need to identify the columns in our dataset that require encoding. We do this by examining each column's data type. If a column's data type is 'object' (indicating it contains text or categorical values) and it's not the 'contentment' column, we add it to the list of columns requiring encoding.

The following columns require encoding:

- 'Gender'
- 'Traveler Type'
- 'Type of Travel'
- 'Class'

2.5.2 One-Hot Encoding

Now that we've identified the columns, we'll proceed with one-hot encoding using the OneHotEncoder from scikit-learn. One-hot encoding transforms categorical variables into binary vectors, making them suitable for machine learning models. We create a binary column for each category within the original categorical columns.

2.5.3 Encoding the 'contentment' Column

The encoding process ensures that the dataset is now ready for machine learning modeling. By creating binary columns for categorical variables and encoding the target variable, we've prepared the data for our analysis.

2.6 Handling Missing Values

In this section, we address missing values within the dataset. To ensure data completeness and integrity, we fill the null values with the mean of the corresponding column. This strategy is commonly employed to handle missing data, ensuring that the imputed values are representative of the overall distribution.

The code iterates through the columns with missing values and fills them with the mean of the respective column. This data preprocessing step is essential for ensuring that the dataset is complete and ready for further analysis and machine learning modeling.

2.7 Outlier Detection and Removal

In this section, we employ boxplots to detect outliers in each feature. Outliers are data points that significantly deviate from the majority of data and can impact the integrity of our analysis. After identifying outliers, we perform removal for the 'Traveler Type' and 'Checkin Service' columns.

2.7.1 Boxplot Analysis

To identify outliers, we visualize each feature using boxplots. Boxplots provide a clear visual representation of the data's distribution, helping us spot potential outliers.

The boxplots allow us to identify columns with potential outliers.

2.7.2 Outlier Removal

After identifying the columns with outliers, we perform outlier removal for the 'Flight distance' and 'Checkin Service' columns. We use a custom outlier removal class, 'outlierremoval', to apply the removal process.

The custom 'outlierremoval' class helps us identify and remove outliers from these columns, ensuring the data is more robust for analysis and modeling.

This section documents the process of outlier detection and removal, enhancing the integrity of our dataset for further analysis.

2.8 Correlation Analysis

In this section, we delve into the correlation between various columns to understand how different features relate to each other. Specifically, we explore the correlation heatmap, which visualizes the strength and direction of relationships between numeric columns.

2.8.1 Correlation Heatmap

To assess the correlation between columns, we generate a heatmap. The heatmap helps us identify potential patterns and relationships within the dataset. It's a valuable tool for understanding which variables are strongly related and which are not.

The heatmap provides insights into the correlations between different columns. Notably, we observe a very high correlation between 'Arrival Delay' and 'Departure Delay.' Based on our previous exploratory data analysis (EDA), it's evident that most values in these columns fall within the 0-200 minute range.

This high correlation suggests a strong relationship between the delays at departure and arrival. Understanding these correlations is crucial for our analysis and can guide further modeling and feature selection.

The heatmap, combined with our EDA findings, enhances our comprehension of the dataset.

2.9 Defining Feature (X) and Target (Y) Columns

In this section, we prepare our dataset for machine learning by defining the feature (X) and target (Y) columns. Feature columns represent the input variables used to make predictions, while the target column contains the values we want to predict.

In the code, we create the feature matrix X by dropping the 'contentment' and 'Unnamed: 0' columns, as well as 'Departure Delay in Minutes' and 'Arrival Delay in Minutes.' These columns are excluded as they are often used as predictors, and it's important to set them apart from the target variable.

The target vector Y is defined as the 'contentment' column.

2.10 Feature Normalization

In this section, we address the normalization of feature columns using the StandardScaler. Normalization is a crucial preprocessing step in machine learning that ensures all features are on the same scale.

The code iterates through each feature column, applying StandardScaler to standardize the values. This process scales the features to have a mean of 0 and a standard deviation of 1.

2.11 Pre Processing Steps Tried

the following are some of the pre processing steps that we tried.

2.11.1 Addressing Class Imbalance

In this section, we tackle the issue of class imbalance within the dataset. Class imbalance can affect the performance of machine learning models, particularly when one class significantly outweighs the other. To mitigate this issue, we employ the RandomOverSampler (ROS) technique to balance the number of samples in each class.

In the code, we utilize the ROS to oversample the minority class, ensuring that the 'discontent' and 'content' classes have an equal number of samples. This resampling technique helps mitigate class imbalance and reduces the risk of bias towards the majority class.

3 Model Training and Prediction

In this section, we complete the final steps of our analysis, which include training classification models, and making predictions on the test data.

3.1 Model Training

We divided the dataset into training and validation sets using the `train_test_split` function. The training set is used to train our classification models, while the validation set allows us to assess their performance.

Next, we trained the dataset using four different classification models:

- Random Forest
- Logistic Regression
- XGBoost
- K-Nearest Neighbors (KNN)

3.2 Model Evaluation

We evaluated the performance of each model by calculating the accuracy score on both the validation and training datasets. The accuracy score provides insights into how well the models are classifying passenger contentment.

3.2.1 Logistic Regression

The accuracy for Logistic Regression on the test set is 0.8765699436985708, while the accuracy on the training set is 0.8740059911215909.

3.2.2 Random Forest

The accuracy for Random Forest on the test set is 0.9629469226697464, while the accuracy on the training set is 0.9999879696353596.

3.2.3 XG Boost

The accuracy for XG Boost on the test set is 0.962850680910447, while the accuracy on the training set is 0.9763843942109885.

3.2.4 K-Nearest Neighbors (KNN)

The accuracy for KNN on the test set is 0.9272412299696838, while the accuracy on the training set is 0.9467536061018009.

3.3 Making Predictions on the Test Data

To make predictions on the test data, we first loaded the test dataset then We used the trained Random Forest model, which demonstrated promising performance on the validation dataset, to make predictions on the test data.

The final predictions were saved in a CSV file for submission.

This section marks the conclusion of our analysis.