UNIVERSIDADE DA CORUÑA
FACULTY DE INFORMATICA

FUNDAMENTALS OF MACHINE LEARNING FOR COMPUTER VISION

ACADEMIC YEAR 2024 – 2025

Final work : Training of Machine Learning models on the Airline Passenger Satisfaction dataset

NOWAK Julia

Teacher:
CERNADAS Eva
NETO Pedro

January 2025

Contents

1 Introduction

This report addresses a classification problem focused on analysing airline passenger satisfaction, detailing the process from data exploration to model evaluation. It provides a comprehensive overview of the dataset's characteristics, preprocessing steps, models training and the evaluation metrics used.

1.1 Problem

The aim of this project is to develop a predictive model to classify airline passengers as satisfied or dissatisfied based on various features in the dataset. Customer satisfaction is a critical factor in the airline industry, directly influencing loyalty and overall business performance. By identifying the key factors driving satisfaction, airlines can refine their services to better meet passenger needs. Using machine learning techniques, this project wants to find patterns in the data and explore how features such as flight distance or service quality contribute to passenger satisfaction.

1.2 Description of the database

The dataset selected for this project is the Airline Passenger Satisfaction dataset from Kaggle. The file that will be used, which contains 129,880 instances, each described by 24 features. These features include continuous variables such as the flight distance, categorical variables like gender, and ordinal variables such as satisfaction levels for specific elements ranged between 0 and 5. The target variable is the passenger satisfaction, classified as either "Satisfied" or "Neutral or Dissatisfied", making this a binary classification problem.

1.3 Code structure

The project contains 2 files. The first airline_passenger_satisfaction.csv is a CSV file containing the entire dataset. The second airline-passenger-satisfaction.ipynb is a jupyter notebook structured in 4 different steps. At first, the Jupyter explores the dataset, it instances and there features. Next, some preprocessing is applied on the dataset in order to provide the best possible training set for the training of the models. Furthermore, we focus on the training of the models, for this project we choose to implement four of them : "KNN", "Decision Tree", "LDA", "SVM". Finally, we analyse the performance of those models through several metrics.

2 Preprocessing

The preprocessing phase was a critical step in ensuring the dataset was clean, consistent, and ready for training multiple classification models. This section outlines the preprocessing pipeline, including the handling of outliers, feature scaling, and other essential steps.

2.1 Data Cleaning

The dataset was first examined for missing values. Any rows containing missing data were removed to focus on the most impactful features for classification.

2.2 Handling Outliers

Outliers were identified through statistical methods, including visualisations like boxplots. This ensured that the dataset was free from values that could disproportionately influence the models.

2.3 Feature Scaling

Feature scaling was performed to standardise the numerical features. Using the `StandardScaler` from `sklearn.preprocessing`, the features were transformed to have a mean of 0 and a standard deviation of 1. This step was particularly important for distance-based algorithms and ensured uniform contributions from all features.

2.4 Categorical Encoding

Categorical variables were transformed into numerical representations using the `LabelEncoder`. This conversion ensured that all input features were compatible with the machine learning algorithms used in the project.

2.5 Train-Test Split

The dataset was split into training and testing subsets using an 80-20 split ratio with the `train_test_split` function from `sklearn.model_selection`. This separation allowed for unbiased model evaluation on unseen data.

2.6 Cross-Validation Preparation

For robust performance evaluation, a cross-validation strategy was employed using `StratifiedKFold`. This technique ensures that each fold retained the same class distribution as the overall dataset, which is crucial for imbalanced datasets.

3 Model training

3.1 K-Nearest Neighbors (KNN)

The K-Nearest Neighbors (KNN) algorithm was implemented using `KNeighborsClassifier` from `sklearn.neighbors`. The model's performance was evaluated using two approaches: direct evaluation with varying k values and stratified k-fold cross-validation.

3.1.1 Implementation

As explained partially in the previous section the KNN model uses:

- **Feature Scaling:** Features were standardised with `StandardScaler` to ensure equal contribution in distance calculations.
- **Direct Evaluation:** The number of neighbors (k) was tested over a range of values $[3, 5, 9, 11, 13, 19, 29, 49]$. The model was trained on `x_train`, `y_train` and evaluated on `x_test`.
- **Cross-Validation:** A stratified k-fold cross-validation ($k = 5$) was performed with `StratifiedKFold`, maintaining class balance across folds. Mean accuracy was computed to assess robustness.

3.1.2 Observations

Lower k values resulted in overfitting, while higher values improved generalisation. Cross-validation provided robust performance estimates, with $k = 9$ achieving a balance between accuracy and generalisation.

3.2 Decision Tree

The Decision Tree was implemented using `DecisionTreeClassifier` from `sklearn.tree`. This algorithm is designed to capture non-linear relationships by recursively partitioning the feature space based on the most informative features.

3.2.1 Implementation

As partially explained previously, the Decision Tree model includes the following steps:

- **Model Configuration:** The classifier was initialised with a `max_depth` parameter to control tree complexity and prevent overfitting. A random seed (`random_state` = 42) ensured reproducibility.
- **Training and Prediction:** The model was trained using `x_train` and `y_train`, and predictions were made on `x_test`.

3.2.2 Observations

The Decision Tree classifier effectively identified patterns in the data, with deeper trees capturing more details but could be overfitting. In our example we went for a max depth of 5 which balance the capture of details and avoid overfitting.

3.3 Linear Discriminant Analysis (LDA)

The Linear Discriminant Analysis (LDA) classifier was implemented using `LinearDiscriminantAnalysis` from `sklearn.discriminant_analysis`. LDA is a linear classification method that projects the dataset on a lower-dimensional space while trying to maximising the class separability.

3.3.1 Implementation

As this model doesn't require any parameters we only trained it once on the training set and evaluated it on the test set.

3.4 Support Vector Machine (SVM)

The Support Vector Machine (SVM) classifier was implemented using `SVC` from `sklearn.svm`. SVM is a powerful algorithm for classification tasks, particularly effective in high-dimensional spaces and when classes are not linearly separable.

3.4.1 Implementation

The SVM model was configured and trained with the following steps:

- **Kernel Selection:** The kernel function was set to `rbf` as it performed better then `linear` or `poly`.
- **Regularisation Parameter (C):** The regularisation parameter C was set to 1.0, to maximise the margin .

3.4.2 Observations

The SVM classifier demonstrated robust performance which will be developed in the next section.

4 Model Evaluation and Results

This section presents the performance of the models implemented. The models were evaluated using metrics such as accuracy, precision, recall, F1-score, Cohen's Kappa, and confusion matrices.

4.1 Performance Comparison

The table below summarises the key metrics for the best-performing configurations of each model:

Model	Accuracy	Precision	Recall	F1-Score	Cohen's Kappa
KNN (k=9)	91.60%	92%	91%	91%	0.83
Decision Tree (depth=14)	95.48%	96%	95%	95%	0.91
LDA	86.91%	87%	86%	87%	0.74
SVM (kernel=RBF)	89.97%	90%	89%	90%	0.79

Table 1: Performance metrics for the best-performing configurations of each model.

4.2 Observations and Insights

- **K-Nearest Neighbors (KNN):** - Achieved a balanced accuracy of 91.60% with $k = 9$, highlighting its effectiveness in handling this dataset. - Cross-validation further supported its robustness, with a mean accuracy of 93.65%.
- **Decision Tree:** - The Decision Tree achieved the highest accuracy of 95.48%, with a depth of 14. - The visualisation revealed that features like `Online Boarding` and `In-flight Wifi Service` were the most important predictors of satisfaction. - We noticed that a higher depth improved the performances until a certain point where the model started to overfit instead of capture the generalisation.
- **Linear Discriminant Analysis (LDA):** - LDA had an accuracy of 86.91%, showing it is effective for linearly separable data.
- **Support Vector Machine (SVM):** - The SVM with an RBF kernel achieved an accuracy of 89.97%. - Its performance was competitive, but training was computationally expensive and this lead to fewer hyperparameters' optimisation than expected.

4.3 Conclusion

In conclusion the Decision Tree (95.48%) outperformed other models, leveraging its ability to model complex relationships and identify key features such as `Online Boarding` and `In-flight Wifi Service` as the visualisation showed us.

The small difference between the Cohen's Kappas and the accuracies suggests the dataset is well-balanced. This is indeed what we conclude after analyzing the dataset.

High F1-scores across the models confirm their ability to balance precision and recall effectively. This suggests the dataset does not heavily penalise either false positives or false negatives, making it well-suited for general classification tasks without severe trade-offs.

Overall, the analysis shows that the dataset is well-structured and allows for effective classification with minimal noise. Which conducts to model with a high performance to classify the passengers satisfaction in order to help plane companies to improve their offers.

5 Conclusion

This project explored the development and evaluation of machine learning models to classify airline passenger satisfaction, leveraging a well-structured dataset with balanced classes and diverse features. The preprocessing, including data cleaning, outlier handling, feature scaling, and categorical encoding, ensured the dataset was optimised for training.

Among the four models implemented—K-Nearest Neighbors (KNN), Decision Tree, Linear Discriminant Analysis (LDA), and Support Vector Machine (SVM), the Decision Tree demonstrated the highest accuracy at 95.48%, effectively capturing complex relationships in the data. It also identified key predictors of passenger satisfaction,