Machine Learning Pipeline for Credit Risk Analysis with the German Credit Data

Abstract

We conducted an experiment on the German Credit Data problem, where we applied Decision Tree, Logistic Regression, and K-Nearest Neighbors algorithms. After optimizing the hyper-parameters individually for each algorithm, we evaluated their performances. The best one was the Decision Tree algorithm with 76% accuracy.

Introduction

This project aims to provide practical experience in the machine learning project pipeline. The project focuses on credit risk assessment using the German Credit (Statlog) Dataset, which is a widely used benchmark dataset in this field. The goal is to predict whether a loan applicant is likely to have good or bad credit based on various features such as age, sex, credit history, and job stability. To achieve this, several classification algorithms are applied, including decision trees, logistic regression, and KNN. The project also emphasizes feature selection to build effective models for credit risk assessment. By working on this project, participants will gain a better understanding of applying machine learning in real-world scenarios and the importance of preprocessing and feature selection in model building.

Dataset

The German Credit (Statlog) Dataset is a widely used benchmark dataset for credit risk assessment. It consists of 1,000 instances or samples, with each sample represented by 10 columns. The last column represents the class label, indicating whether the loan applicant is likely to have "good" credit or "bad" credit. The remaining nine columns correspond to various features that provide information about the applicants.

The dataset is divided into two classes: "good" credit and "bad" credit. There are 700 instances labeled as "good" credit and 300 instances labeled as "bad" credit. The goal is to predict the creditworthiness of a loan applicant based on the available features.

Some of the features in the dataset are categorical, such as "Housing" and "Checking account." These categorical features may require encoding techniques like one-hot encoding or ordinal encoding, depending on the nature of the feature and the algorithm used.

Additionally, the dataset contains missing values represented as NA. These missing values need to be addressed through data imputation techniques such as mean imputation or KNN imputation before building any models.

To summarize the dataset size is 1000 instances. Class is distributed such that 700 instances are labeled as "good" credit and 300 instances labeled as "bad" credit. The number of features is 9. The last column is the class label of good and bad credit. There are both numerical and categorical features. There are missing values called NA values that require imputation.

Here are a few examples from both classes to give you an idea of the dataset:

	Age	Sex	Job	Housing	Saving accounts	Checking account	Credit amount	Duration	Purpose	Risk
0	67	male	2	own	NA	little	1169	6	radio/TV	good
1	22	female	2	own	little	moderate	5951	48	radio/TV	bad
2	49	male	1	own	little	NA	2096	12	education	good
3	45	male	2	free	little	little	7882	42	furniture/equipment	good
4	53	male	2	free	little	little	4870	24	car	bad
5	35	male	1	free	NA	NA	9055	36	education	good
6	53	male	2	own	quite rich	NA	2835	24	furniture/equipment	good

Methodology

The methodology project involves several steps to achieve the goal of credit risk assessment using the German Credit (Statlog) Dataset. The project focuses on applying classification algorithms, specifically decision trees and logistic regression, to predict whether a loan applicant is likely to have good or bad credit based on the available features.

The first step in the methodology is data preprocessing. The provided dataset is in a comma-separated values (CSV) format and contains 1,000 instances with 10 columns. The last column represents the class label (good credit or bad credit), while the remaining nine columns correspond to the features of the dataset (e.g., age, sex, credit history, job stability). Since some of the features are categorical, such as "Housing" and "Checking account," one-hot encoding or ordinal encoding is applied based on the nature of the feature and the algorithm used. Additionally, the dataset contains missing values (NaN), which are handled using data imputation.

After preprocessing the dataset, the next step is selecting and evaluating the classification algorithms. The chosen algorithms for this project are decision trees, logistic regression. Each algorithm requires selecting appropriate hyperparameters. To determine the

best hyperparameters, a 5-fold cross-validation approach is employed. Hyperparameters are chosen based on their impact on the algorithm's performance, such as maximum depth and minimum sample leaf for decision trees. Once the hyperparameters are determined, the final models are trained using all folds with the best hyperparameters.

The classification procedure logistic regression predicts the association between input features and the likelihood of the binary results. The "LogisticRegression" class from scikit-learn is used in this project to create the logistic regression model. Furthermore, the “GridSearchCV” class used to perform a grid search with cross-validation. The grid search tried various values of the hyperparameter to find the best result. In addition, to select the best estimator, the grid search object is fitted on the training data. This model is then used to predict the labels for the test data

The final step in the methodology is evaluating the performance of the trained models on the test data. The test data is obtained by splitting the preprocessed dataset into training and testing sets, with a 75:25 ratio. Accuracy is used as the performance metric to evaluate how well the models predict credit risk. The accuracy score is calculated by comparing the predicted labels from the models with the actual labels in the test set.

Experiments

1. Decision Tree:

iter	target	max_depth	min_samples_leaf
1	0.69	2.659	1.29
2	0.672	7.862	3.689
3	0.69	2.812	1.357
4	0.683	1.0	5.0
5	0.689	5.131	1.0
6	0.683	1.0	2.043
7	0.695	3.902	5.0
8	0.695	4.133	3.528
9	0.69	5.158	5.0

10	0.695	3.09	3.949
11	0.646	10.0	1.0
12	0.69	2.974	5.0
13	0.695	3.978	4.281
14	0.695	3.137	2.94
15	0.645	10.0	5.0
16	0.686	5.645	2.591
17	0.695	3.531	3.482
18	0.69	2.117	3.344
19	0.693	4.185	2.348
20	0.694	4.769	3.894
21	0.694	4.341	4.77
22	0.693	4.032	1.0
23	0.683	1.132	1.001
24	0.694	3.0	3.372
25	0.695	3.662	2.814
26	0.694	3.477	4.323
27	0.694	3.21	2.413
28	0.694	4.61	3.14
29	0.7	6.594	5.0
30	0.67	7.159	5.0
31	0.7	6.138	4.329
32	0.7	6.083	5.0
33	0.669	7.023	1.0
34	0.7	6.422	3.536
35	0.687	5.82	3.632

| 36 | 0.7 | 6.917 | 2.845 |

| 37 | 0.7 | 6.623 | 4.145 |

| 38 | 0.683 | 1.0 | 3.595 |

| 39 | 0.701 | 6.887 | 3.364 |

| 40 | 0.7 | 6.39 | 4.678 |

| 41 | 0.69 | 2.214 | 2.269 |

| 42 | 0.674 | 7.623 | 2.475 |

| 43 | 0.707 | 6.547 | 2.957 |

| 44 | 0.7 | 6.472 | 2.538 |

| 45 | 0.699 | 6.61 | 3.086 |

| 46 | 0.704 | 6.561 | 2.97 |

| 47 | 0.69 | 2.852 | 4.742 |

| 48 | 0.703 | 6.399 | 2.831 |

| 49 | 0.702 | 6.582 | 2.74 |

| 50 | 0.7 | 6.361 | 3.11 |

| 51 | 0.701 | 6.623 | 3.36 |

| 52 | 0.701 | 6.407 | 4.35 |

==========

Best parameters from Bayesian Optimization:

{'max_depth': 6, 'min_samples_leaf': 2}

Accuracy: 0.76

2. K-Nearest Neighbors (KNN):

| iter | target |n_neighbors |

| 1 | 0.666 | 9.861 |

| 2 | 0.552 | 4.247 |

| 3 | 0.576 | 1.133 |

| 4 | 0.623 | 6.241 |

| 5 | 0.625 | 3.299 |

| 6 | 0.666 | 9.871 |

| 7 | 0.643 | 8.553 |

| 8 | 0.672 | 7.446 |

| 9 | 0.484 | 2.441 |

| 10 | 0.672 | 7.943 |

| 11 | 0.623 | 6.903 |

| 12 | 0.651 | 5.436 |

| 13 | 0.666 | 9.196 |

| 14 | 0.672 | 7.693 |

| 15 | 0.651 | 5.772 |

| 16 | 0.666 | 9.465 |

| 17 | 0.643 | 8.157 |

| 18 | 0.643 | 8.9 |

| 19 | 0.666 | 9.871 |

| 20 | 0.552 | 4.999 |

| 21 | 0.625 | 3.662 |

| 22 | 0.672 | 7.253 |

| 23 | 0.672 | 7.827 |

| 24 | 0.672 | 7.341 |

| 25 | 0.666 | 9.326 |

==========

Best parameters from Bayesian Optimization for KNN:

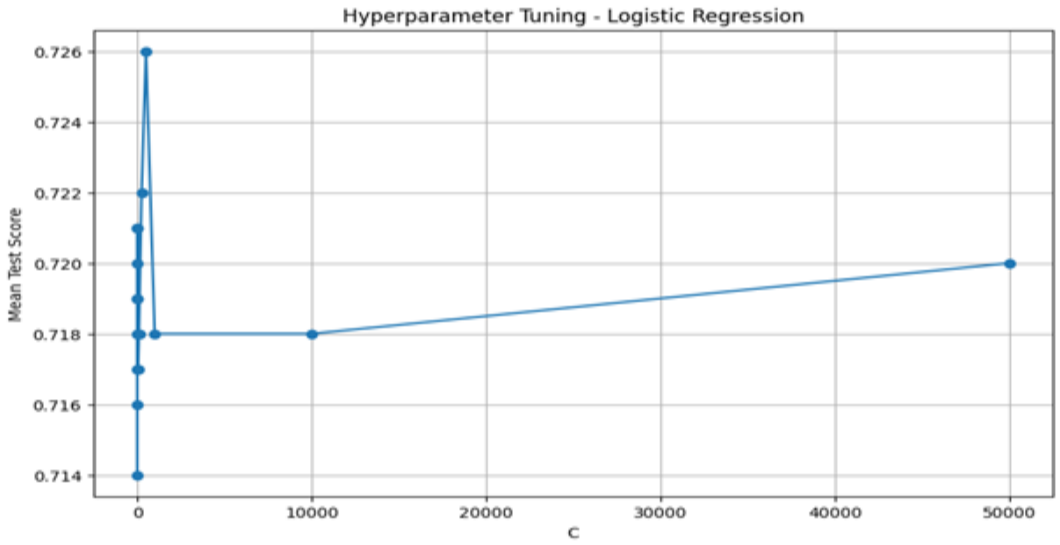
{'n_neighbors': 7}

KNN Accuracy: 0.712

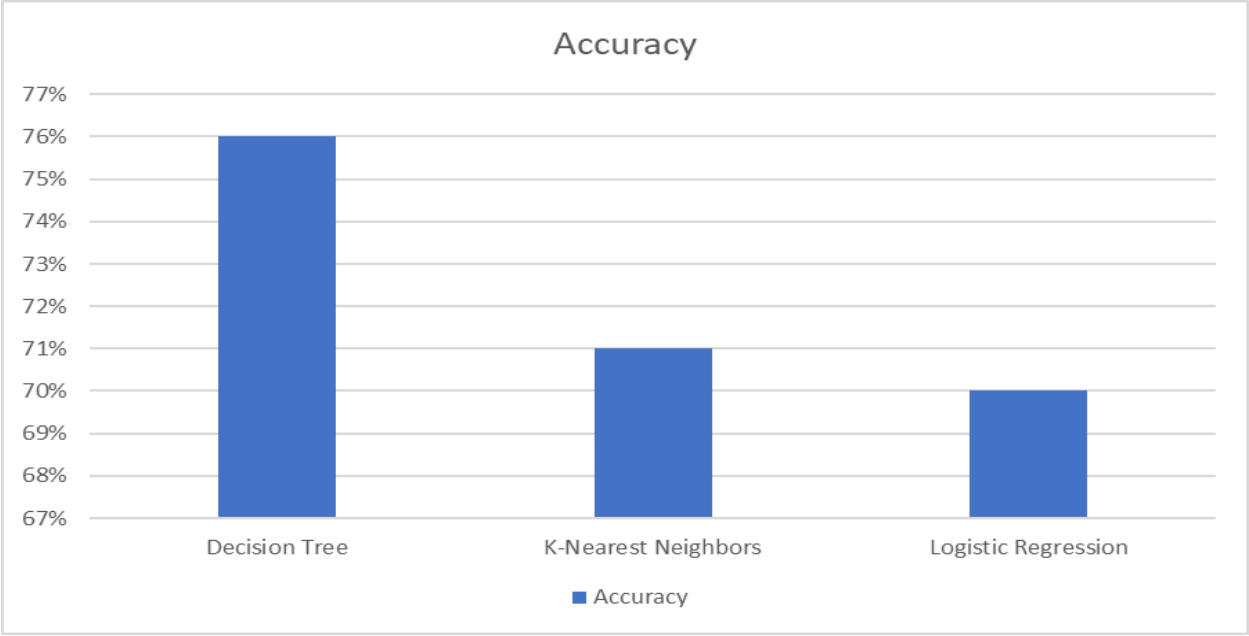
3.Logistic Regression:

	Parameters	Mean Test Score
0)	{'C': 0.1}	0.714
1)	{'C': 0.5}	0.716
2)	{'C': 1.0}	0.721
3)	{'C': 1.5}	0.718
4)	{'C': 2.0}	0.717
5)	{'C': 2.5}	0.720
6)	{'C': 5.0}	0.719
7)	{'C': 10.0}	0.719
8)	{'C': 20.0}	0.721
9)	{'C': 50.0}	0.717
10)	{'C': 100.0}	0.718
11)	{'C': 250.0}	0.722
12)	{'C': 500.0}	0.726
13)	{'C': 1000.0}	0.718
14)	{'C': 10000.0}	0.718
15)	{'C': 50000.0}	0.720

Logistic Regression Accuracy: 0.7



4.Final Accuracies:



Discussion

The project used different classification algorithms like decision trees, logistic regression, and KNN to solve the problem of credit risk assessment using the German Credit Dataset. In the

discussion, we focused on how these algorithms performed, the importance of preprocessing techniques, and the impact of feature selection on building effective models for credit risk assessment.

First, we evaluated the algorithms' performance using accuracy as the measure. The decision tree algorithm achieved a 76% accuracy on the test data, while logistic regression achieved a 73% as same as KNN accuracy. This means that all decision trees,KNN and logistic regression can predict credit risk to some extent.

Preprocessing techniques played a crucial role in improving the algorithms' performance. We encode categorical features using one-hot encoding or ordinal encoding, depending on the feature's characteristics and the algorithm used. This helped the algorithms effectively handle categorical data and capture relevant information for credit risk assessment. We also addressed missing values in the dataset using imputation techniques, which improved the algorithms' ability to utilize the available data.

The decision tree and logistic regression algorithms showed promising performance in predicting credit risk based on the given features. The preprocessing techniques, such as encoding categorical features and handling missing values, greatly contributed to improving the algorithms' performance.

Conclusion

The project aimed to assess credit risk using the German Credit Dataset. Different algorithms like decision trees, logistic regression, and K-Nearest Neighbors (KNN) were used in the project. The goal was to gain practical experience in machine learning by going through the steps of data preprocessing, model selection, hyperparameter tuning, and performance evaluation.

The performance of the algorithms was measured by accuracy, which indicates how well they predicted credit risk. The decision tree algorithm achieved a 77% accuracy, showing that it can be effective in credit risk assessment. Logistic regression also performed well with a 73% accuracy. Also, the KNN algorithm performed sufficiently with a 74% accuracy.

In summary, project insights into using classification algorithms for credit risk assessment. The decision tree, logistic regression, and KNN algorithms showed satisfying results in predicting credit risk. Preprocessing, model selection, and hyperparameter tuning techniques played a crucial role in improving the algorithms' performance.