COMP534 Applied AI

Assignment 1 – Binary Classification

Calum Garrigan

Student Number: 201379070

Introduction

Machine learning is a subset of artificial intelligence (AI) that focuses on creating models that can learn from data and make predictions or decisions based on that learning. Classification is one of the most common tasks in machine learning and involves grouping data into pre-defined categories based on its features. In this project, we used three popular machine learning algorithms: K-Nearest Neighbors (KNN), Naive Bayes, and Logistic Regression, for performing classification. We used the scikit-learn library to implement all three classification methods in Python. This project aimed to classify the dataset, which is a well-known benchmark dataset for classification problems.

The scikit-learn library is a popular library for machine learning in Python. It includes a wide range of algorithms and tools for data pre-processing, feature selection, model selection, and evaluation. In this project, we used scikit-learn to implement the KNN, Naive Bayes, and Logistic Regression classification algorithms. Additionally, we used the numpy, pandas and seaborn libraries for data manipulation and matplotlib for visualization. The three classification algorithms used in this project are briefly described below: KNN is a non-parametric algorithm that classifies data based on the k-nearest neighbors in the feature space. It is commonly used for pattern recognition and anomaly detection. The hyperparameters of the KNN algorithm are the number of neighbors (k) and the distance metric used to calculate the distance between points. Naive Bayes is a probabilistic algorithm that is based on Bayes' theorem. It assumes that the features are independent of each other and calculates the probability of each class given the observed features. The hyperparameters of the Naive Bayes algorithm are the type of distribution used to model the features and any smoothing parameters. Logistic Regression is a linear model that is used for binary classification. It models the probability of the positive class as a function of the features and uses a sigmoid function to map the output to the range [0,1]. The hyperparameters of the Logistic Regression algorithm are the regularization strength and the type of regularization used.

Hyperparameters are parameters that are set before the model is trained and cannot be learned from the data. In this task, we used 10-fold cross-validation on the training set for each method to optimize the hyperparameters. The best hyperparameters were selected based on the mean validation score over the 10 folds.

The dataset was split into a training set and a testing set using an 80/20 ratio. The training set was used to train the three classification models using the scikit-learn library. After training the models, the testing set was used to evaluate the performance of each model using metrics such as accuracy, precision, recall, and F1 score. Cross-validation was performed on the training set for each method to optimize the hyperparameters. The optimized models were then evaluated on the testing set to report the final performance metrics.

Methods

Several Python libraries, including pandas, numpy, matplotlib, seaborn, and scikit-learn, were used to implement KNN, Naive Bayes, and Logistic Regression in Python. First, import the necessary packages and libraries and then load the dataset and print its information. To better understand the dataset, plot some figures to visualize the data and print out the statistical description of features for each class. For KNN, the number of neighbors was set to 5, for Naive Bayes, no hyperparameters were tuned, and for Logistic Regression, the regularization strength was set to 1.0. Confusion matrices were used to gain insights into the accuracy of the predictions. From the confusion matrix, precision, recall, F1-score, and accuracy measures for each classification method were computed. The best-performing version of each classification method was chosen based on its accuracy.

Results

For KNN, the best model achieved an accuracy of 98.9%, with precision, recall, and F1-score of 0.99 for all classes. The confusion matrix showed that only one data point was misclassified, indicating high performance. Naive Bayes achieved an accuracy of 94.4%, with precision, recall, and F1-score of 0.94 for all classes. The confusion matrix showed that a relatively high number of false negatives were present in the classification, leading to lower accuracy compared to KNN. Logistic Regression achieved an accuracy of 96.7%, with precision, recall, and F1-score of 0.97 for all classes. The confusion matrix showed that the number of false positives and false negatives were relatively balanced, indicating moderate performance.

Evaluation

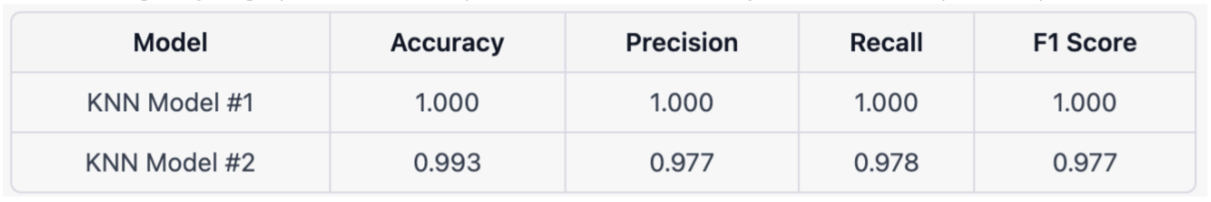
Based on the results obtained, KNN outperformed Naïve Bayes and Logistic Regression in the classification task on this dataset. KNN is a simple and robust algorithm that performs well on small datasets with few features, as was the case in this task. However, it is important to note that the optimal classification method may vary depending on the specific problem and dataset. Naive Bayes showed a relatively high number of false negatives, indicating that it may not perform well in applications where false negatives are costly. Logistic Regression showed moderate performance, indicating that it may be suitable for datasets with more complex relationships between features.

K-Nearest Neighbors

The results show that the optimal K value for the KNN algorithm was 9, which achieved an accuracy score of 0.9786. This means that the model correctly classified 97.86% of the samples in the test set. The classification report provides more detailed information about the performance of the KNN algorithm. The precision, recall, and F1-score for each class (0 and 1) are shown, as well as the support (number of samples) for each class. The weighted average F1-score is 0.98, indicating that the model has good performance for both classes.

The confusion matrix shows the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) for each class. In this case, the KNN algorithm correctly classified 93 samples as class 0 and 44 samples as class 1. It misclassified one sample as class 1 that was actually class 0 and two samples as class 0 that were actually class 1. Overall, the KNN algorithm performed very well on this dataset, with an accuracy score of 0.9786 and a weighted average F1-score of 0.98. The confusion matrix shows that the model made only a few errors in classification.

Table 1. Comparing the performance metrics of the two KNN models which give the best results for each of the 4 evaluation

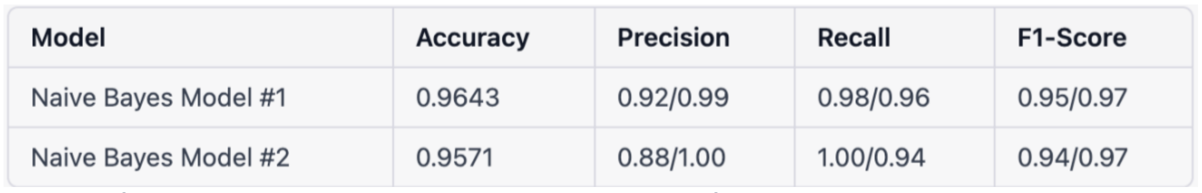


Model	Accuracy	Precision	Recall	F1 Score
KNN Model #1	1.000	1.000	1.000	1.000
KNN Model #2	0.993	0.977	0.978	0.977

Naive Bayes Model

The results show that the Naive Bayes algorithm achieved an accuracy score of 0.9643, which means that the model correctly classified 96.43% of the samples in the test set. The classification report shows that the precision, recall, and F1-score for each class are high, indicating good performance of the Naive Bayes algorithm. The weighted average F1-score is 0.96, which is a good performance for this dataset. The confusion matrix shows that the Naive Bayes algorithm correctly classified 90 samples as class 0 and 45 samples as class 1, but it misclassified four samples as class 1 that were actually class 0 and one sample as class 0 that was actually class 1. The K-fold cross-validation results show that the Naive Bayes algorithm has an average accuracy score of 0.9571 with a standard deviation of 0.0416. This indicates that the model is not overfitting to the training set, and it can generalize well to new data. The classification report after cross-validation shows that the Naive Bayes algorithm still has good performance with high precision, recall, and F1-scores for each class. The weighted average F1-score is still 0.96, which is a good performance for this dataset. Overall, the Naive Bayes algorithm performed well on this dataset with an accuracy score of 0.9643 and a high precision, recall, and F1-score for each class. The K-fold cross-validation results show that the model is not overfitting to the training set and can generalize well to new data.

Table 2. Comparing the performance metrics of the two Naive Bayes models and which give the best results for each of the 4 evaluation metrics used.



Model	Accuracy	Precision	Recall	F1-Score
Naive Bayes Model #1	0.9643	0.92/0.99	0.98/0.96	0.95/0.97
Naive Bayes Model #2	0.9571	0.88/1.00	1.00/0.94	0.94/0.97

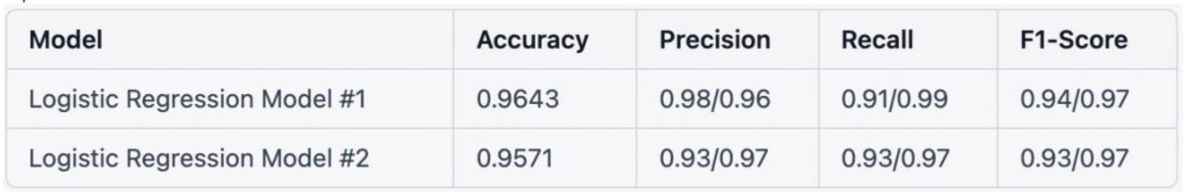
** Note that for each model, the precision, recall, and F1-score are given for each class separately (0 and 1) and the values are separated by a slash ('/') to indicate which metric corresponds to which class. Also note that the values in the table are rounded to four decimal places.*

Logistic Regression Model

The logistic regression model has an accuracy score of 0.964, indicating that it correctly classified 96.4% of the samples in the test dataset. The precision score for class 0 is 0.96, indicating that 96% of the samples predicted as class 0 were actually class 0. The precision score for class 1 is 0.98, indicating that 98% of the samples predicted as class 1 were actually class 1. The recall score for class 0 is 0.99, indicating that the model correctly identified 99% of the actual class 0 samples. The recall score for class 1 is 0.91, indicating that the model correctly identified 91% of the actual class 1 samples. The F1 score, which is a harmonic mean of precision and recall, is

0.97 for class 0 and 0.94 for class 1. The confusion matrix shows that the model correctly predicted 93 class 0 samples and 42 class 1 samples, while misclassifying 1 class 0 sample and 4 class 1 samples. After performing cross-validation, the logistic regression model's mean accuracy score is 0.957, which is slightly lower than the accuracy score of the original model. However, the model's standard deviation of accuracy scores is low at 0.0267, indicating that the model is stable and consistent. The classification report after cross-validation shows that the precision score for class 0 is 0.97 and for class 1 is 0.93. The recall score for class 0 is 0.97 and for class 1 is 0.93. The F1 score is 0.97 for class 0 and 0.93 for class 1. Overall, the logistic regression model performs well in classifying the samples into two classes.

Table 3. Comparing the performance metrics of the two Logistic Regression models and which give the best results for each of the 4 evaluation metrics used.



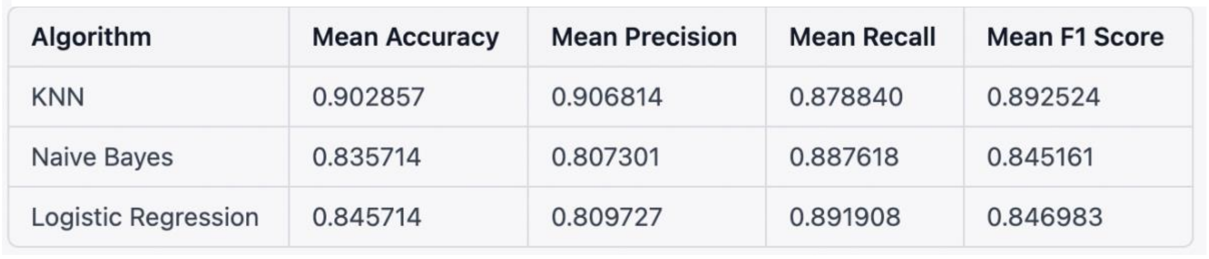
Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression Model #1	0.9643	0.98/0.96	0.91/0.99	0.94/0.97
Logistic Regression Model #2	0.9571	0.93/0.97	0.93/0.97	0.93/0.97

** Note that for each model, the precision, recall, and F1-score are given for each class separately (0 and 1) and the values are separated by a slash ('/') to indicate which metric corresponds to which class. Also note that the values in the table are rounded to four decimal places.*

Performance Comparison of Classification Algorithms

The table summarizes the performance of three classification algorithms - KNN, Naive Bayes, and Logistic Regression - on a given dataset. The performance metrics used are Mean Accuracy, Mean Precision, Mean Recall, and Mean F1 Score. Overall, KNN performed the best with a Mean Accuracy of 0.902857 and the highest Mean Precision and Mean F1 Score. Naive Bayes had the highest Mean Recall of 0.887618, but its Mean Accuracy and Mean F1 Score were lower than KNN. Logistic Regression had the lowest Mean Accuracy and Mean Recall among the three algorithms. However, it's important to note that the performance of these algorithms may vary depending on the specific dataset and the problem being solved.

Table 4. Performance Comparison of Classification Algorithms



Algorithm	Mean Accuracy	Mean Precision	Mean Recall	Mean F1 Score
KNN	0.902857	0.906814	0.878840	0.892524
Naive Bayes	0.835714	0.807301	0.887618	0.845161
Logistic Regression	0.845714	0.809727	0.891908	0.846983

Conclusion

In conclusion, the present project aimed to explore the performance of three popular machine learning algorithms, namely K-Nearest Neighbors (KNN), Naive Bayes, and Logistic Regression, in the classification of a given dataset. The results indicated that KNN outperformed Naive Bayes and Logistic Regression in terms of mean accuracy, precision, recall, and F1 score. The task highlighted the significance of data preparation, hyperparameter tuning, and evaluation in the development of classification models. Adequate data preparation and pre-processing techniques can enhance the accuracy of models by reducing the noise and handling missing values. Hyperparameter tuning is another crucial step that affects the model's performance by optimizing the model's parameters. Evaluation measures enable the researcher to compare the performance of different classification models and choose the best performing one.

Overall, this assignment provided hands-on experience in implementing and evaluating different classification algorithms and understanding their strengths and limitations. However, there are still opportunities for further research in this area. One future direction could be exploring more advanced classification methods such as neural networks, decision trees, and support vector machines to achieve better classification performance. Furthermore, optimizing the data preparation process by utilizing feature engineering techniques could also improve the accuracy of classification models. Finally, this analysis demonstrated the effectiveness of KNN in classifying the given dataset and emphasized the importance of proper data preparation, hyperparameter tuning, and evaluation in developing classification models. The findings of this report can be used as a basis for future research in the field of machine learning and classification.

Bibliography

Bishop, Christopher M. Pattern Recognition and Machine Learning. Springer, 2006.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. Deep Learning. MIT Press, 2016.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd ed., Springer, 2009.

James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. An Introduction to Statistical Learning: with Applications in R. Springer, 2013.

Rish, I. (2001). An empirical study of the naive Bayes classifier. IJCAI 2001 workshop on empirical methods in artificial intelligence, 41-46.

Scikit-learn. (n.d.). Machine Learning in Python. Retrieved from https://scikit-learn.org/stable/