Brandenburgische Technische Universität Cottbus-Senftenberg
Institute für Infomatik
Chair of Database and Information Systems

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Practical Training – Task 2
Implementation and Application of Classification Methods

Module: Foundations of Data Mining

Group 08

1 Shaik Fharook
MatrikelNr.: 5014962

2 Yalla Pavani Bhuvaneshwari
MatrikelNr.: 5011106

Date of Submission: 27-Jan-2025

Supervisors:

1 Prof. Dr.-Ing. habil. Ingo Schmitt

2 Alexander Stahl, M. Sc.

1. Introduction

This report presents an implementation and comparative analysis of three classification methods applied to real-world datasets. The study focuses on comparing a custom implementation of the K-Nearest Neighbors (KNN) algorithm against two established classifiers from the scikit-learn library: KNeighborsClassifier and Gaussian Naïve Bayes. The analysis aims to evaluate both the accuracy and computational efficiency of these methods across different classification scenarios.

2. Experiment Setup

2.1. Datasets

- **Breast Cancer Wisconsin (Diagnostic) Dataset:** A binary classification dataset with 569 samples and 30 features, sourced from the UCI Machine Learning Repository. Each sample represents a set of measures derived from cell nuclei images, such as mean radius, texture, and perimeter. The goal is to classify tumors as benign (Class 0) or malignant (Class 1) based on these cell image characteristics.
- **Wine Quality Dataset:** A multiclass classification dataset with 4898 samples of both red and white wines, and 11 features, sourced from the UCI Machine Learning Repository. Features include physiochemical properties such as alcohol content, pH, and residual sugar. The objective is to classify wine quality into six classes ranging from 3(lowest quality) to 8(highest quality).

2.2. Classifiers

- **CustomKNNClassifier:** An implementation of the K-Nearest Neighbours algorithm. It uses Euclidean distance to calculate similarity and a majority vote among the k-nearest neighbours to predict labels. Distances are computer pairwise, leading to high computational complexity for larger datasets.
- **KNeighborsClassifier:** The KNeighborsClassifier from scikit-learn, which is optimized using data structures such as KDTree or BallTree for faster neighbour searches. It employs the same principle of majority voting but is significantly faster due to its optimized implementation.
- **Gaussian Naïve Bayes:** A probabilistic classifier based on Bayes theorem. It assumes that all features are conditionally independent and follows a Gaussian distribution for continuous features. This simplicity makes it highly efficient but less effective for datasets with complex interdependencies among features.

2.3. Data Preprocessing

Before training, both datasets were standardized to have a mean of 0 and a standard division of 1 to ensure that all features have similar scales. This step ensures a no dominant distance-based calculations in KNN and probabilistic calculations in Naïve Bayes. The datasets were split into training and testing sets using stratifies sampling to maintain the class distribution in both splits. For each dataset, 80% of data was used for training, and the remaining 20% was used for testing.

2.4. Hyperparameter Tuning

- **Gaussian Naïve Bayes**: This model relies on the assumption that features are conditionally independent, and it was applied with its default parameters for both datasets. Hyperparameter tuning was not performed for Gaussian Naïve Bayes, as it generally works well with its default settings.
- **KNN Classifiers (CustomKNNClassifier or KNeighborsClassifier)**: For both KNN-based models, the default value of *k=5* was first experimented with, and the Euclidean distance metric was used for distance calculations. Afterward, a hyperparameter search was performed for *k* values ranging from *1 to 50* to find the optimal k for both classifiers. The optimal *k* was determined based on the accuracy achieved on the test set.

2.5. Evaluation Metrics:

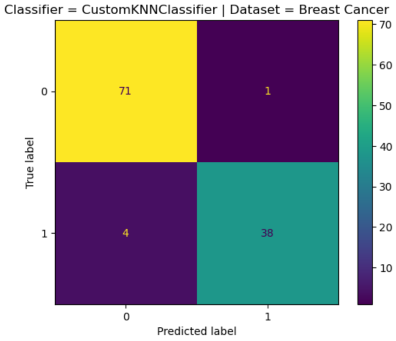
The performance of the models was evaluated using the following metrics:

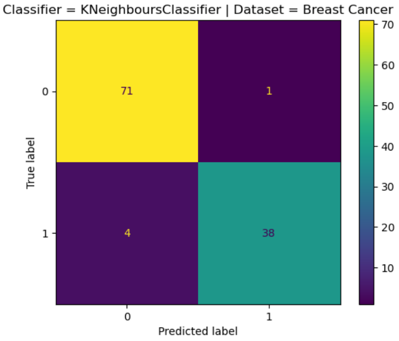
- **Test Accuracy:** The proportion of correctly classified instances.
- **Recall:** The ability of the classifier to correctly identify positive instances (useful in imbalanced datasets).
- **Precision:** The proportion of positive predictions that were actually correct.
- **Training Time:** The time taken to fit the model on the training data.
- **Prediction Time:** The time taken to make predictions on the test data.

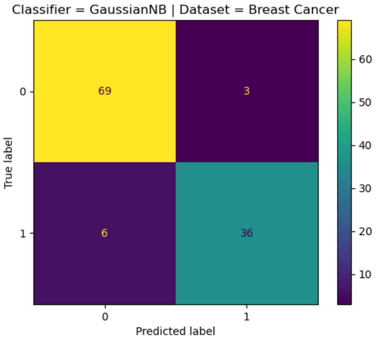
3. Presentation of Results

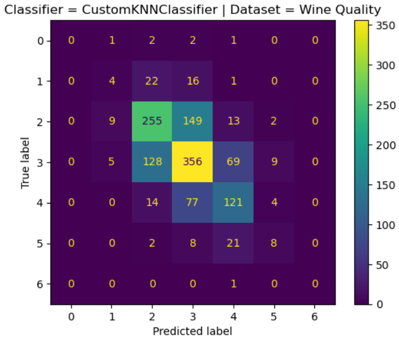
The table below summarizes the performance of the classifiers on both datasets.

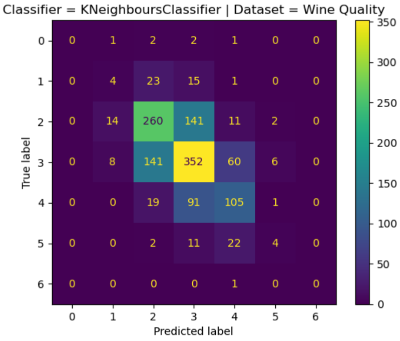
Dataset	Classifier	Test Accuracy	Recall	Precision	Training Time	Prediction Time
Breast Cancer Wisconsin Diagnostic	CustomKNNClassifier	0.9561	0.9048	0.9744	0.0	0.294
	KNeighborsClassifier	0.9561	0.9048	0.9744	0.001	0.007
	Gaussian Naïve Bayes	0.9211	0.8571	0.9231	0.001	0.0
Wine Quality Dataset	CustomKNNClassifier	0.5723	0.5723	0.5598	0.0	35.105
	KNeighborsClassifier	0.5577	0.5577	0.5433	0.006	0.112
	Gaussian Naïve Bayes	0.4277	0.4277	0.4453	0.003	0.001

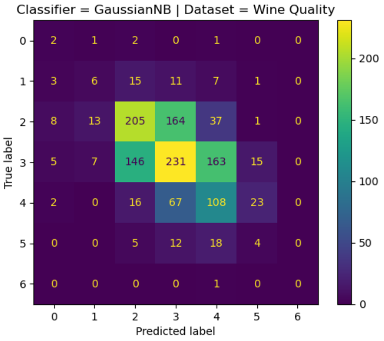








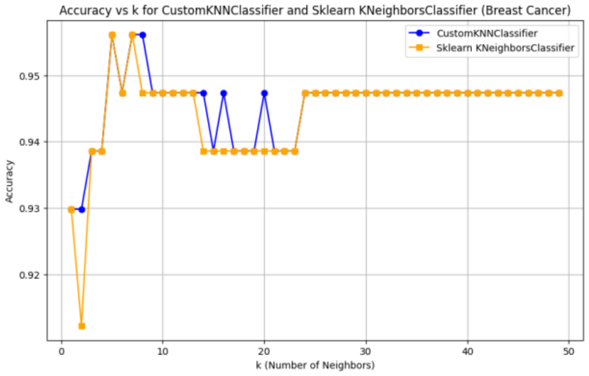


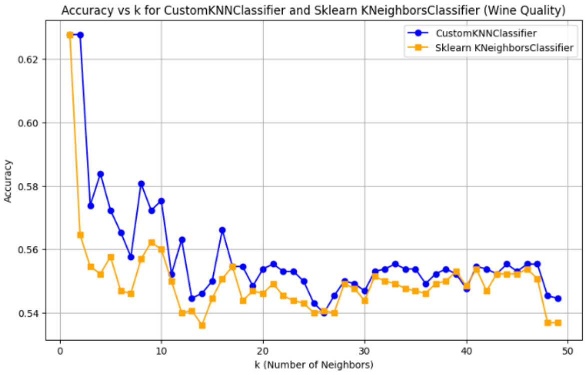


3.1. Optimal *k* Analysis

Through hyperparameter tuning, it was found that the optimal value of k for both CustomKNNClassifier and KNeighborsClassifier was:

- **Breast Cancer Wisconsin Diagnostic Dataset:** *k=5,* with the highest accuracy of 0.9561. This value of k struck a balance between underfitting and overfitting, resulting in a well generalized model.
- **Wine Quality Dataset:** *k = 1,* with the highest accuracy of 0.6277. A smaller k was more effective due to the noise and complexity in the dataset, where more localized decision-making (i.e., considering fewer neighbours) yielded better performance.





4. Discussion

For Breast Cancer Wisconsin Diagnostic dataset, it is found that both CustomKNNClassifer and KNeighborsClassifier performed exceptionally well, with nearly identical results. The high accuracy, recall, and precision indicate that these models are well-suited for binary classification problems where the classes are clearly separable. It was observed that the KNN Classifiers, especially with an optimal k of 5, outperformed Gaussian Naïve Bayes in cased where feature correlations are strong. Naïve Bayes, despite being computationally faster, struggled to maintain the same level of accuracy due to its strong assumptions about the feature independence.

It is noted that the performance on the Wine Quality dataset was notably lower, especially for Gaussian Naïve Bayes, which suffered from the feature corelation issue. KNN-based models performed better than Naïve bayes, but the results were still relatively poor. This suggests that the Wine quality dataset's complexity requires a more sophisticated approach perhaps using more advanced ensemble methods or neural networks to achieve higher accuracy.

The training time for all classifiers was found to be quite fast, especially for Gaussian Naïve Bayes. However, KNN-based models were slower during predictions, particularly with the Wine Quality dataset, due to the larger number of samples and features. The training time for KNN models was relatively low, but the prediction time increased significantly as the dataset grew. This trade-off should be considered when using KNN for large datasets.

5. Conclusion

This study demonstrated that KNN-based classifiers (CustomKNNClassifier ad KNeighborsClassifier) are highly effective for binary classification tasks, as seen in their excellent performance on the Breast Cancer dataset. The optimal value of *k (5)* struck a balance between underfitting and overfitting, resulting in high accuracy, precision, and recall. However, for more complex datasets, such as Wine Quality, the performance of KNN Classifiers was suboptimal. This indicates a need for more localized decision-making, as evident from the suboptimal *k* value of 1 for this dataset.

Gaussian Naïve Bayes, while computationally efficient, struggled in scenarios where feature dependencies were significant, particularly for wine quality dataset. The findings highlight the trade-offs between computational efficiency and model performance, emphasizing that algorithm selection should consider the complexity and structure of the dataset.