Report for Exercise 7 in WSI

Adam Szostek, Index Number 331443

February 11, 2025

1 Implemented Algorithm

1.1 Naive Bayes Classifier

The Naive Bayes classifier is a simple yet effective algorithm used in classification tasks. It is based on Bayes' theorem, which describes the relationship between the probabilities of events and their conditions:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}, \tag{1}$$

where:

- $P(A|B)$ - the probability of event A occurring given that event B has occurred,
- $P(B|A)$ - the probability of event B occurring given that event A has occurred,
- $P(A)$ - the prior probability of event A,
- $P(B)$ - the prior probability of event B.

The algorithm assumes that the input features are independent of each other (the naive assumption), which significantly simplifies the computations. The classifier selects the class C_k for which the conditional probability $P(C_k|\mathbf{x})$ is the highest, where $\mathbf{x}$ is the feature vector. This can be expressed as:

$$C_k = \arg\max_{C_k} P(C_k) \prod_{i=1}^{n} P(x_i|C_k), \tag{2}$$

where x_i is the i-th feature of the vector $\mathbf{x}$, and n is the number of features.

In the implementation of the classifier in Python, the following assumptions were adopted:

- **Estimation of Prior Class Probabilities**: The value $P(C_k)$ for each class is calculated as the ratio of the number of samples belonging to that class to the total number of samples in the training set:

$$P(C_k) = \frac{\text{number of samples in class } C_k}{\text{total number of samples}}$$

- **Estimation of Feature Distribution Parameters**: It is assumed that the features for each class are distributed according to a normal distribution. For each class, the mean (μ) and variance (σ^2) are calculated for each feature:

$$P(x_i|C_k) = \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{(x_i - \mu_i)^2}{2\sigma_i^2}\right)$$

- **Calculating Conditional Probabilities**: To assign a class to a sample, the logarithms of the conditional probabilities are calculated for all classes. The conditional probability $P(C_k|\mathbf{x})$ is proportional to the sum of the logarithm of the prior probability of the class and the logarithms of the conditional probabilities of the features given the class:

$$\log P(C_k|\mathbf{x}) \propto \log P(C_k) + \sum_{i=1}^{n} \log P(x_i|C_k)$$

  The class with the highest value is assigned to the sample.

2 Planned Numerical Experiments

2.1 Dataset

The implementation of the Naive Bayes classifier was tested on the Iris Data Set. This dataset contains information about 150 samples from three species of irises. The goal is to train a model to classify samples into the correct species based on four features contained in the samples.

2.2 Model Evaluation

The model was evaluated using cross-validation. This is a model evaluation technique that involves splitting the data into k groups (folds) and iteratively testing the model on one group while training it on the remaining ones. The process can be described as follows:

1. Split the dataset $\mathcal{D}$ into k equal (or nearly equal) parts: $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_k$.
2. For each part i (from 1 to k):
   (a) Use $\mathcal{D} \setminus \mathcal{D}_i$ as the training set.
   (b) Use $\mathcal{D}_i$ as the testing set.
   (c) Train the model and calculate the performance metric (e.g., accuracy, F_1, etc.).
3. The average metric from the k iterations is the final model evaluation:

$$\text{Score} = \frac{1}{k} \sum_{i=1}^{k} \text{Score}_i$$

2.3 Comparison with Other Models

Additionally, a plot was created comparing the average accuracy results obtained from 5-fold cross-validation for three other models used in classification problems. These are SVC (with an RBF kernel), Decision Tree, and K-Nearest Neighbors. Implementations from the sklearn library were used for the models.

3 Obtained Results

Below is the confusion matrix obtained by performing 5-fold cross-validation for the implemented Naive Bayes classifier.

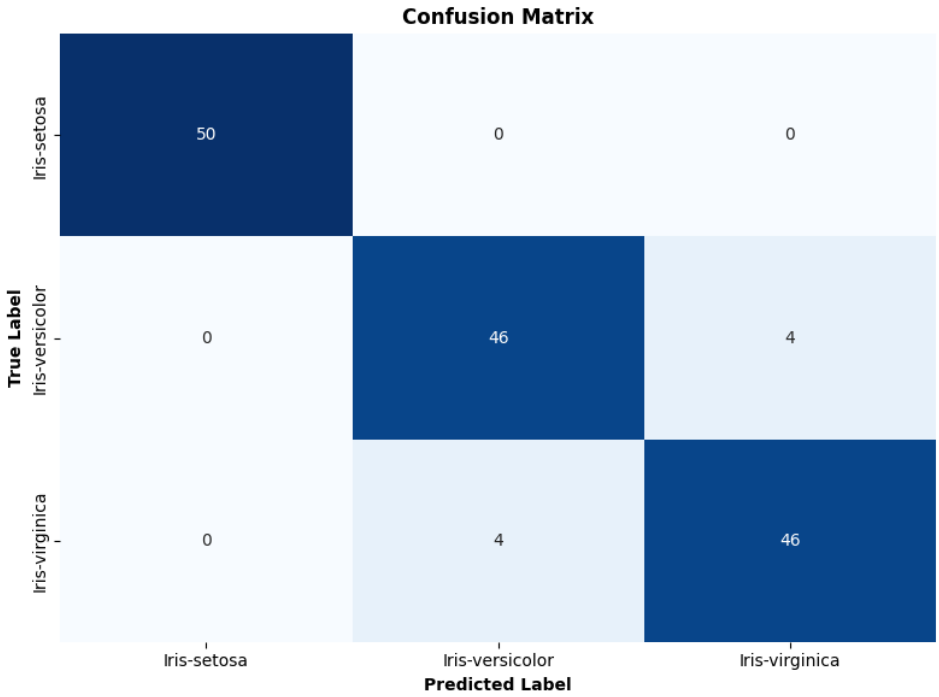

Figure 1: Confusion Matrix for Naive Bayes Classifier

The following plot shows the accuracy obtained for each fold of cross-validation and the average of these results.

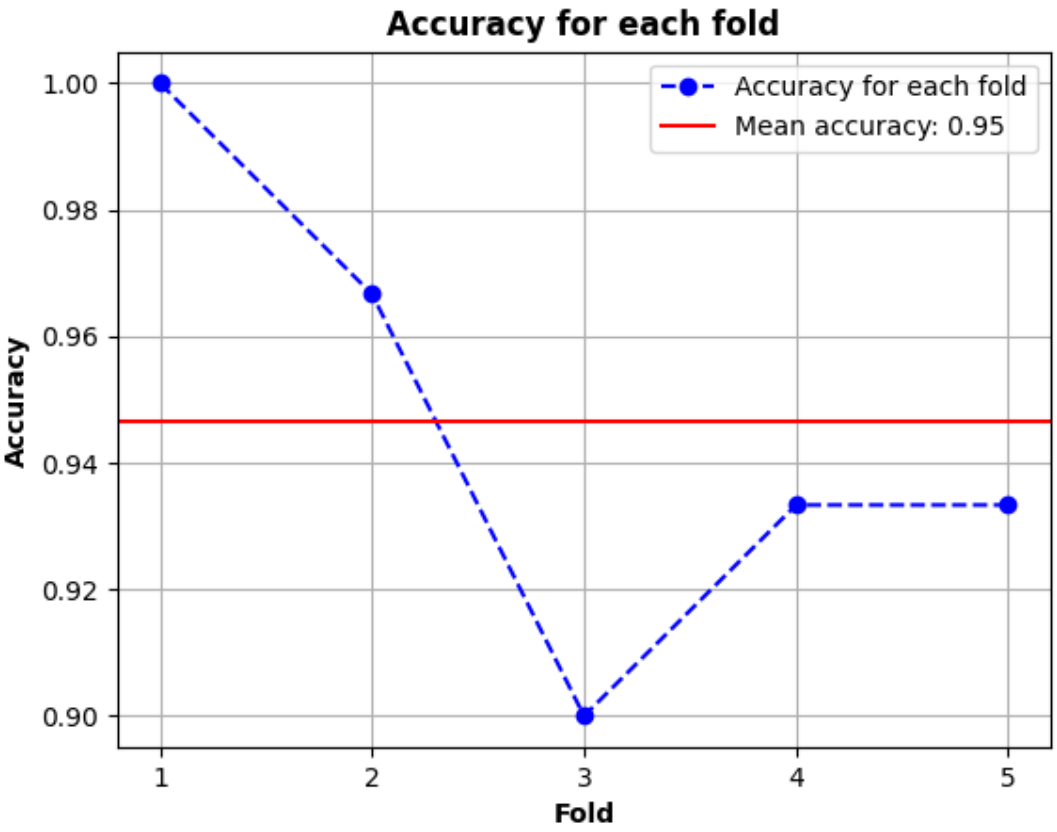

Figure 2: Cross-Validation Accuracy Scores

Below is a comparison of the average accuracy from 5-fold cross-validation for four different types of classifiers, including the tested Naive Bayes algorithm.

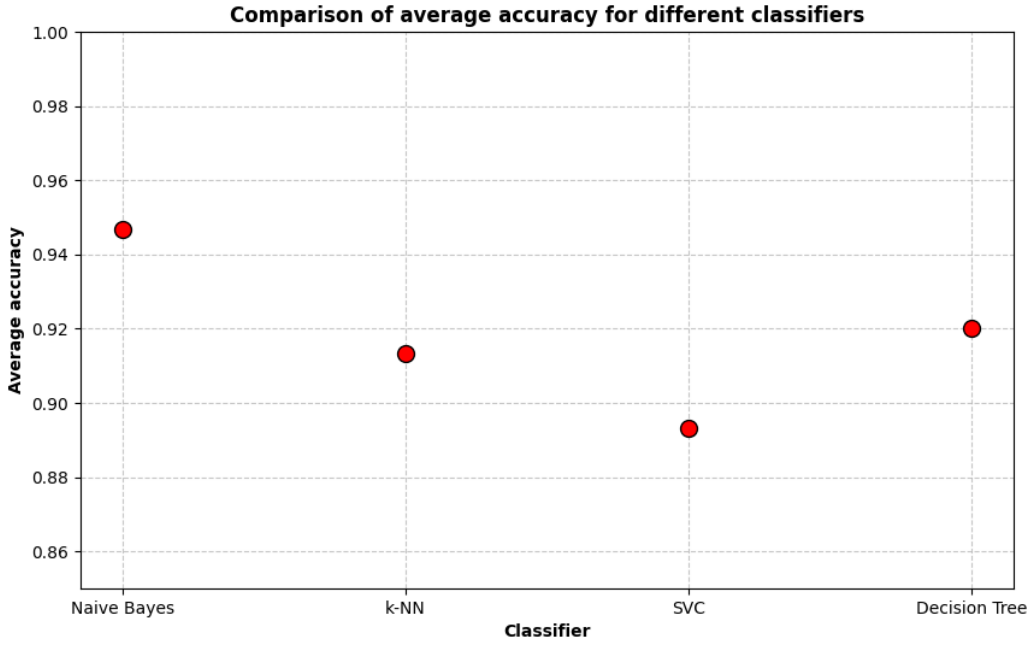

Figure 3: Comparison of Classifier Accuracies

4 Conclusions from the Results

Based on the results presented above, it can be confidently stated that the Naive Bayes classifier achieves very high accuracy on the tested dataset. A key observation is the very small size of the Iris Dataset, which contains only 150 samples in total. Such a number of samples fits well with less complex algorithms like the Naive Bayes classifier. This can cause a significant overestimation of the obtained results, which explains the accuracy of 1.0 in the first fold. Another important issue is feature independence, which is a crucial assumption of the algorithm (allowing the correct use of Bayes' theorem). It is very rarely fully satisfied, but in the case of the tested dataset, it is likely that the data are close to independent, which again explains the very high accuracy. Additionally, the comparison with other classifiers supports this thesis, as the Naive Bayes classifier achieves higher classification accuracy than the other models, suggesting that the algorithm selection was correct for the tested dataset, thereby helping to confirm the validity of the assumptions. Moreover, the assumption of a normal distribution for iris features may be particularly accurate due to the natural distribution of their features observed in reality.
In summary, the Naive Bayes classifier is a simple yet effective algorithm that is well-suited for classifying simple data. However, caution should be exercised with the obtained results due to the small number of samples.