Department of Computer Engineering

Bu-Ali Sina University

Machine Learning Course

Assignment 1: Testing various classifiers

By:

Ali Nafisi

Course Professor:

Muharram Mansoorizadeh

Fall 2023

TABLE OF CONTENTS

1 INTRODUCTION

In contemporary data-driven research, the selection and evaluation of appropriate machine learning classifiers are pivotal aspects, influencing the accuracy and effectiveness of predictive models. This document presents an in-depth exploration of four prominent classifiers: the Naive Bayes classifier, Decision Tree classifier, Random Forest classifier, and Multilayer Perceptron classifier. Each classifier's methodology, intricacies, and applications will be scrutinized to provide a comprehensive understanding of their respective strengths and limitations. Furthermore, this exploration will extend to a comparative analysis of these classifiers, leveraging experimental results obtained from diverse datasets, including the Iris dataset and the Optical Recognition of Handwritten Digits dataset. The ensuing sections elucidate the methodologies employed, present the experimental results, and draw insightful conclusions based on the findings.

2 METHODOLOGY

2.1 Naive Bayes classifier

The Naive Bayes classifier is a probabilistic machine learning algorithm based on Bayes' theorem, which describes the probability of an event occurring given the probability of another event. Specifically, the "naive" assumption in Naive Bayes refers to the assumption that features used to describe an observation are conditionally independent, given the class label. Despite this simplifying assumption, Naive Bayes has demonstrated effectiveness in various real-world applications, particularly in text classification and spam filtering.

The algorithm calculates the probability of a particular class label given a set of features by leveraging Bayes' theorem. In the context of classification, the class label is the output, and the features are the input variables. The algorithm calculates the likelihood of observing the given features under each class and combines it with the prior probability of each class to determine the posterior probability. The class with the highest posterior probability is then assigned to the observation.

One key advantage of the Naive Bayes classifier lies in its computational efficiency and simplicity. The assumption of independence among features allows for a straightforward calculation of probabilities, especially when dealing with high-dimensional datasets. However, this assumption may not always hold in real-world scenarios, leading to potential limitations in performance when the features are correlated. Despite this limitation, Naive Bayes remains a valuable tool, particularly in scenarios where computational efficiency and interpretability are paramount.

2.2 Decision Tree classifier

A Decision Tree classifier is a popular machine learning algorithm used for both classification and regression tasks. The fundamental idea behind a Decision Tree is to recursively partition the dataset into subsets based on the values of different features, ultimately leading to a tree-like structure of decision nodes. Each decision node represents a specific feature, and the branches emanating from it correspond to the possible values of that feature. At the leaf nodes of the tree, the final predictions or classifications are made. The decision-making process involves evaluating the features at each node to determine the optimal split, with the goal of maximizing the homogeneity of the resulting subsets in terms of the target variable.

The construction of a Decision Tree involves iteratively selecting the best features and split points to partition the data at each node. The "best" split is typically determined by metrics such as Gini impurity or information gain, which quantify the degree of impurity or uncertainty in a node.

By repeatedly partitioning the data based on these criteria, the algorithm constructs a tree that captures complex decision boundaries and relationships within the dataset. Decision Trees are known for their interpretability, as the resulting structure allows users to visualize and understand the decision-making process, making them particularly useful for explaining the rationale behind predictions.

While Decision Trees have the advantage of interpretability, they can be prone to overfitting, capturing noise in the training data. To mitigate this, techniques such as pruning and setting a minimum number of samples required to split a node are often employed. Additionally, ensemble methods like Random Forests, which aggregate multiple Decision Trees, can enhance predictive performance and generalization by reducing overfitting.

2.3 Random Forest classifier

The Random Forest classifier is an ensemble learning method that leverages the power of multiple Decision Trees to improve predictive accuracy and generalization. Unlike a single Decision Tree, which may be susceptible to overfitting, a Random Forest aggregates the predictions of multiple trees, thereby reducing the impact of individual tree idiosyncrasies. The ensemble is constructed by training each Decision Tree on a randomly sampled subset of the training data, and at each split, a random subset of features is considered. By introducing randomness in both data sampling and feature selection, the Random Forest mitigates overfitting and enhances the model's robustness to variations in the dataset.

One key feature of Random Forests is the concept of "bagging" (Bootstrap Aggregating), where each tree is trained on a bootstrapped sample of the original dataset. This involves randomly selecting samples with replacement, creating diverse subsets for individual trees. Furthermore, the algorithm introduces randomness in feature selection by considering only a subset of features at each split. The final prediction of the Random Forest is obtained by aggregating the predictions of all individual trees, often through a majority voting scheme for classification tasks.

Random Forests excel in handling high-dimensional datasets and capturing complex relationships in the data while mitigating overfitting. They are less sensitive to outliers and noise compared to individual Decision Trees, making them a versatile choice for a wide range of applications. The interpretability of Random Forests is somewhat reduced compared to a single Decision Tree, but their superior predictive performance and robustness make them a popular choice in machine learning practice.

2.4 Multilayer perceptron classifier

The Multilayer Perceptron (MLP) classifier is a type of artificial neural network designed for supervised learning tasks, particularly for complex classification problems. It consists of multiple layers of nodes (neurons) organized into an input layer, one or more hidden layers, and an output layer. Each connection between nodes is associated with a weight, and each node applies an activation function to the weighted sum of its inputs. The architecture's depth and non-linear activation functions enable MLPs to learn intricate relationships within the data, making them well-suited for tasks involving non-linear decision boundaries.

The training process of an MLP involves adjusting the weights to minimize the difference between the predicted output and the actual target values. This is typically achieved through backpropagation, an optimization algorithm that propagates the error backward through the network, updating weights in the direction that minimizes the error. The choice of activation functions, the number of hidden layers, and the number of nodes in each layer are hyperparameters that can significantly impact the model's performance and training dynamics. Common activation functions include the sigmoid, hyperbolic tangent (tanh), and rectified linear unit (ReLU).

MLPs are known for their ability to model complex relationships and patterns in data, making them applicable to a wide range of tasks, including image and speech recognition, natural language processing, and other domains with high-dimensional input data. However, their training can be computationally demanding, and they may be prone to overfitting, especially with limited data. Regularization techniques, such as dropout, are often employed to mitigate overfitting in MLPs. The versatility of MLPs, coupled with advancements in optimization algorithms and regularization techniques, has contributed to their widespread use in various machine learning applications.

3 EXPERIMENTAL RESULTS

3.1 Datasets

3.1.1 Iris dataset

The Iris dataset is a widely used and well-known collection of data in the field of machine learning and statistics. Introduced by the British biologist and statistician Sir Ronald A. Fisher in 1936, the dataset comprises measurements of four features from three different species of iris flowers: setosa, versicolor, and virginica. The four features include the lengths and widths of the sepals and petals, all measured in centimeters. With a total of 150 instances, each species has 50 samples, making the dataset balanced. The Iris dataset serves as a fundamental resource for pattern recognition, classification, and clustering tasks, making it a cornerstone in introductory data science studies and a benchmark for evaluating the performance of various machine learning algorithms. Its simplicity, clarity, and real-world applicability have contributed to its enduring popularity in both academic and practical contexts.

3.1.2 Optical recognition of handwritten digits dataset

The Optical Recognition of Handwritten Digits dataset, commonly referred to as the MNIST dataset, is a seminal collection in the realm of machine learning, specifically designed for the task of handwritten digit recognition. Comprising 8x8 pixel images, each datapoint in the dataset represents a grayscale depiction of a handwritten digit (0 through 9). With 10 distinct classes corresponding to the digits, the dataset encompasses approximately 180 samples per class, resulting in a total of 1797 samples. The dimensionality of each image is set at 64, and the features within each image are represented as integers ranging from 0 to 16. This dataset has been instrumental in advancing the field of image classification, providing a compact yet comprehensive set of examples that researchers and practitioners alike use to assess the efficacy of various machine learning algorithms in recognizing and distinguishing handwritten digits. The modest size and standardized format of MNIST have contributed to its widespread adoption as a benchmark in the development and evaluation of digit recognition models.

3.2 Comparing performance of models

Tables 1 and 2 present the performance metrics of each model applied to the Iris and Optical Recognition of Handwritten Digits datasets. The Iris dataset was stratified into distinct training and validation sets, with a partitioning ratio of 70 percent for training and 30 percent for validation. Simi-

Table 1: Performance of models (with their default parameters) on Iris dataset

	Train data				Validation data			
	Accuracy	**Precision**	**Recall**	**F1-score**	**Accuracy**	**Precision**	**Recall**	**F1-score**
Naive Bayes	0.94	0.95	0.95	0.95	0.98	0.98	0.97	0.97
Decision Tree	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Random Forest	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Multilayer Perceptron	0.95	0.96	0.95	0.95	1.00	1.00	1.00	1.00

Table 2: Performance of models (with their default parameters) on Optical recognition of handwritten digits dataset

	Train data				Validation data			
	Accuracy	**Precision**	**Recall**	**F1-score**	**Accuracy**	**Precision**	**Recall**	**F1-score**
Naive Bayes	0.80	0.85	0.80	0.80	0.77	0.82	0.77	0.76
Decision Tree	1.00	1.00	1.00	1.00	0.84	0.84	0.84	0.84
Random Forest	1.00	1.00	1.00	1.00	0.97	0.97	0.97	0.97
Multilayer Perceptron	1.00	1.00	1.00	1.00	0.98	0.98	0.98	0.98

larly, the Optical Recognition of Handwritten Digits dataset underwent a comparable division, with an 80 percent allocation for training and a 20 percent allocation for testing. The results indicate commendable performance across all models concerning both the training and validation subsets of the Iris dataset, underscoring the dataset's inherent simplicity. Consequently, motivated by the desire to assess model robustness in more intricate scenarios, we opted to evaluate their performance on the Optical Recognition of Handwritten Digits dataset, known for its greater complexity relative to the Iris dataset. This strategic shift enables a nuanced comparison of model effectiveness across datasets with varying degrees of complexity.

3.3 Best Hyperparameters for each model

In pursuit of optimizing model performance, a systematic grid search methodology was employed to identify the most efficacious combination of hyperparameters for each individual model. This rigorous exploration was conducted on the Optical Recognition of Handwritten Digits dataset, yielding consequential findings as delineated below.

1. Naive Bayes
    - Var smoothing: 0.06579
2. Decision Tree
    - Criterion: entropy
    - Max depth: 9
    - Max features: sqrt

3. Random Forest
    - Max depth: 16
    - Max features: 0.5
    - N estimators: 60
4. Multilayer Perceptron
    - Alpha: 0.01
    - Hidden layer sizes: 64
    - Max iter: 1000
    - Solver: adam

4 CONCLUSION

In culmination, this study has delved into the methodologies and performances of four prominent machine learning classifiers: the Naive Bayes classifier, Decision Tree classifier, Random Forest classifier, and Multilayer Perceptron classifier. Through a meticulous examination of experimental results on distinct datasets, encompassing the Iris dataset and the Optical Recognition of Handwritten Digits dataset, the classifiers' efficacy in diverse contexts has been evaluated. The comparative analysis has unveiled nuances in their performances, elucidating scenarios where one classifier may outshine another. Additionally, insights into the best hyperparameters for each model have been gleaned through rigorous experimentation. As the exploration draws to a close, the discernments gleaned from this study contribute valuable considerations for the judicious selection and deployment of machine learning classifiers in various applications, paving the way for informed decision-making in predictive modeling endeavors.