Abstract

This project evaluates the performance of four machine learning algorithms: k-Nearest Neighbors (kNN) and Naive Bayes (implemented from scratch), as well as Logistic Regression and Decision Tree (using scikit-learn) on the Iris dataset. The main aim is to determine the best classification model by analyzing metrics such as accuracy, precision, recall, and F1-score. Data preprocessing, exploratory data analysis, k-fold cross-validation, hyperparameter tuning, and feature importance analysis, were undertaken to ensure well made evaluations.

Introduction

This project explores and compares the performance of four algorithms on the Iris dataset. The primary goal is to determine the most effective model by analyzing their strengths, weaknesses, and performance metrics such as accuracy, precision, recall, and F1-score. The Iris dataset has remained one of the most studied datasets in the field of machine learning. It contains 150 samples of iris flowers, each belonging to one of three species: Setosa, Versicolor, and Virginica. Each sample is characterized by four features: sepal length, sepal width, petal length, and petal width. Despite its simplicity, the dataset's linear separability and balanced distribution make it a reliable benchmark for testing machine learning algorithms.

However, its limitations are also well-documented. The dataset's small size and clear separability often lead to near-perfect performance by many models, making it less reflective of real-world cases. Previous investigations into the Iris dataset have often used it as a starting point for introducing machine learning concepts, focusing on the ease with which it can be classified. While this has made the dataset popular in academic contexts, it has also highlighted the need to use additional, more complex datasets to test models in practical applications. Despite that, its well-balanced and well-structured nature makes it an excellent starting point for comparing algorithms and understanding their underlying mechanics, hence I chose to use this dataset.

Background

This project uses four machine learning algorithms: kNN, Naive Bayes, Logistic Regression, and Decision Tree, to classify the Iris dataset. Each of these algorithms

approaches the classification problem differently, and understanding their workings helps in evaluating their strengths and weaknesses.

1. **k-Nearest Neighbors (kNN)**

The kNN algorithm works by comparing a new data point to the closest points in the training dataset, known as neighbors. The number of neighbors considered, represented as k, is a key parameter. For example, if k=3, the algorithm checks the three closest points and assigns the new data point the majority class among them. This simplicity makes kNN easy to understand, but it can become slow when working with large datasets since it has to calculate distances for all training samples every time it makes a prediction.

2. **Naive Bayes**

Naive Bayes is based on probabilities. It uses a formula called Bayes' Theorem to calculate the likelihood of a data point belonging to each class and then assigns it to the most likely class. A key assumption of this algorithm is that all features are independent, which is rarely true in real-world data. However, this assumption works well in many cases on the contrary. Naive Bayes is fast, efficient, and especially useful when the data is simple or categorical.

3. **Logistic Regression**

Logistic Regression is a straightforward but powerful model, particularly for binary classification problems. It predicts the probability of a data point belonging to a specific class by using a mathematical function called the sigmoid. This function outputs values between 0 and 1, which can then be interpreted as probabilities. For multi-class problems like the Iris dataset, extensions like one-vs-rest are used. Logistic Regression is easy to interpret because the model's coefficients show how much each feature influences the prediction, but it may struggle with data that isn't linearly separable.

4. **Decision Tree**

Decision Tree works by splitting the data into smaller and smaller groups based on feature thresholds, like asking yes-or-no questions. For example, it might ask, "Is the petal length greater than 2.5?" to divide the data into two branches. It continues this process until the data is well-classified or a predefined depth is reached. The tree's structure is also easy to visualize and interpret. However, a Decision Tree may overfit the data, learning details that might not generalize well to new samples. To avoid this, parameters like max_depth can be used to limit the tree's growth.

Methodology

1. **Dataset preparation**: The Iris dataset was loaded using scikit-learn. Each sample includes four numerical features: sepal length, sepal width, petal length, and petal width. After ensuring that there are no missing values in the dataset (df.info() shows 150 non-null entries), the data was then divided into 80% training and 20% testing sets to maintain class proportions. To ensure optimal performance for distance-based algorithms like kNN, the feature values were standardized using MinMaxScaler, which scales the data to a range of 0 to 1.

2. **Exploratory Data Analysis (EDA)**: EDA was conducted to understand the data distribution and relationships between the features. Pair plots show that petal length and petal width were the most discriminative features for separating the three species while the correlation heatmap confirmed strong relationships between these two features, highlighting their importance in the classification process.

3. **Algorithm selection:** 4 machine learning algorithms were selected based on their interpretability, performance and widespread usage. They were chosen to provide a mix of simplicity, computational efficiency and flexibility.

    a. kNN: A simple, instance-based algorithm that classifies samples based on proximity to neighbors.

    b. Naive Bayes: A probabilistic model that is efficient and works well on datasets with categorical or independent features.

    c. Logistic Regression: A parametric model that is effective for linear classification tasks and provides interpretable coefficients.

    d. Decision Tree: A non-parametric model that is intuitive and capable of handling non-linear decision boundaries.

4. **Cross-Validation:** To ensure reliable performance evaluation, (k-fold) 5-fold cross-validation was employed. This technique splits the training data into five subsets, training the model on four subsets and validating on the remaining one, then repeating the process five times. This will help mitigate the risk of overfitting by evaluating the model on multiple splits of the data, providing a robust estimate of its generalization ability.

5. **Hyperparameter tuning:** This is conducted to optimize the performance of the classifiers, ensuring that each algorithm is evaluated at its best.

6. **Model Training and Evaluation:** Each algorithm was trained on the prepared dataset and evaluated using accuracy, precision, recall, and F1-score. These metrics were chosen to provide a comprehensive view of the models' performance, especially since the Iris dataset has balanced class distributions. Feature importance was then analyzed for Decision Tree and Logistic Regression to understand the contributions of individual features.

Results

To verify consistency, 5-fold cross validation was performed and the average accuracy across folds were calculated:

- kNN: 96.67% (±2.98%)
- Naive Bayes: 96.00% (±3.89%)
- Logistic Regression: 93.33% (±4.22%)
- Decision Tree: 96.67% (±2.98%)

The cross-validation results show that kNN and Decision Tree maintained high and stable performance across multiple splits, reinforcing their reliability. Logistic Regression and Naive Bayes also performed well but had slightly more variability.

Hyperparameters were then tuned for each model:

- kNN: The best value for k was determined to be 3, which provided the highest cross-validated accuracy.
- For Gaussian Naive Bayes, there are no significant hyperparameters to tune because it relies on the dataset's statistics.
- Logistic Regression: A grid search was performed over the penalty term (l1, l2), regularization strength (C), and solver. The optimal parameters were C=100, penalty='l1', and solver='liblinear' which maximised performance.
- Decision Tree: The maximum depth (max_depth), minimum samples per split (min_samples_split), and minimum samples per leaf (min_samples_leaf) were tuned. The best parameters were max_depth=4, min_samples_split=2, and min_samples_leaf=1.

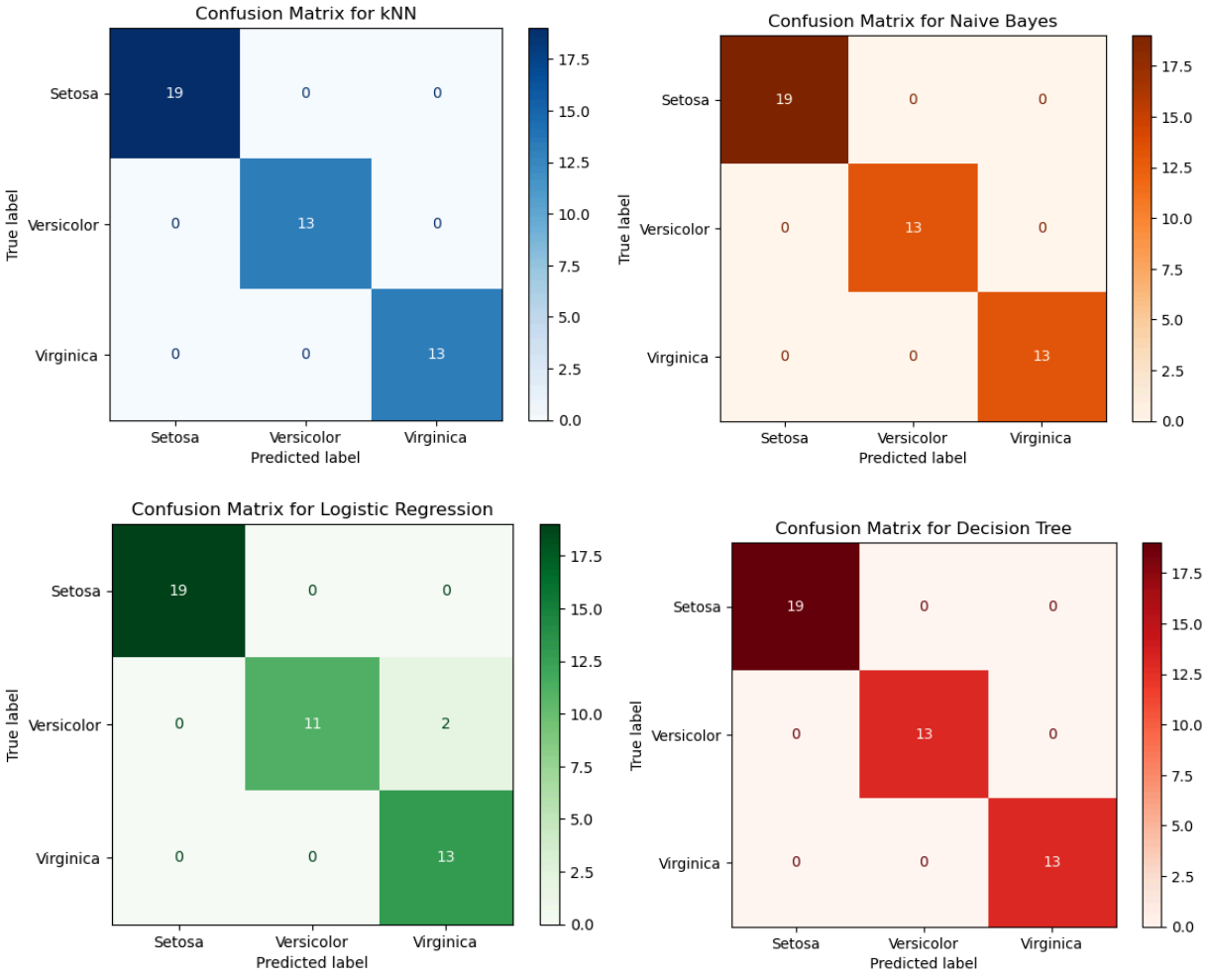
After training and evaluating the four models on the iris dataset, the following results were observed:

Model	Accuracy	Precision (avg)	Recall (avg)	F1-score (avg)

kNN	100%	100%	100%	100%
Naive Bayes	98%	98%	98%	98%
Logistic Regression	100%	100%	100%	100%
Decision Tree	100%	100%	100%	100%

All four models performed exceptionally well, with kNN, Logistic Regression, and Decision Tree achieving perfect scores. Naive Bayes, while slightly behind, still achieved a very high accuracy of 98%. These results highlight the simplicity and separability of the Iris dataset, which allowed most models to excel.

To better illustrate the four models' effectiveness, confusion matrices were constructed as shown below.



These matrices show that kNN, Naive Bayes and Decision Tree models made no classification errors, while Logistic Regression made 2 misclassifications in the Versicolor class. Hence, showing the consistently high accuracy of the models while also highlighting the minor challenge Logistic Regression faced when distinguishing Versicolor samples.

Evaluation

kNN emerged as one of the top performers, achieving perfect accuracy with no misclassifications across all three classes. Its strength lies in its simplicity and effectiveness for small, well-separated datasets like Iris. However, the algorithm's reliance on distance calculations makes it computationally expensive for larger datasets. Moreover, its performance is sensitive to the choice of the hyperparameter k and the scaling of features, requiring careful preprocessing and tuning to ensure optimal results.

Trade-off:

- Performance vs. Scalability: While kNN performs exceptionally well for small datasets, its computational cost makes it less suitable for larger, high-dimensional datasets.

Similarly, Naive Bayes performed excellently, benefiting from its assumption of feature independence, which suited the Iris dataset well. The model's efficiency in both training and prediction phases makes it computationally efficient. However, the assumption of feature independence, while simplifying calculations, limits its flexibility when dealing with highly correlated or overlapping features, which could pose challenges on more complex datasets.

Trade-off:

- Simplicity vs. Flexibility: Naive Bayes excels in speed and simplicity but lacks the flexibility to handle non-linear relationships effectively.

In contrast, Logistic Regression exhibited slight weaknesses, misclassifying two samples in the Versicolor class. While its linear nature makes it computationally efficient and easy to interpret, it struggles when the dataset is not linearly separable, as evidenced by the errors in classifying overlapping regions. Its coefficients also provide valuable insights into feature importance, adding an extra layer of interpretability.

Trade-off:

- Interpretability vs. Complexity Handling: Logistic Regression is interpretable and robust for linear problems but less effective for more complex, non-linear data.

Finally, the Decision Tree also demonstrated perfect classification. Its ability to handle non-linear relationships and provide interpretable, rule-based decisions makes it a powerful model. However, Decision Trees are known to be sensitive to slight variations in the data, which can lead to changes in the tree structure. While overfitting is a common concern with Decision Trees, this issue was mitigated in this project through careful tuning of hyperparameters, such as max_depth.

Trade-offs:

- Flexibility vs. Stability: Decision Tree's ability to model non-linear relationships is a significant advantage, but it can be less stable compared to simpler models like Logistic Regression.

Due to the simplicity of the Iris dataset, generalizability of these findings is limited. While the dataset serves as a good benchmark, it does not fully test the robustness of the models under challenging conditions, such as noisy, imbalanced, or high-dimensional data. Future studies could address these limitations by applying the models to more complex datasets, incorporating noise, or touching on additional classifiers. Furthermore, the dataset assumes that the classes are well-separated, which might not always be the case in real-world datasets with overlapping or ambiguous classes. Overall, while the Iris dataset's simplicity allowed for near-perfect performance, complex real-world scenarios would likely reveal a wider performance gap among these models, providing deeper insights into their respective trade-offs.

Conclusion

The results demonstrated that the Iris dataset's simplicity and separability allowed all models to achieve high accuracy, with three of the four classifiers (kNN, Naive Bayes, and Decision Tree) achieving perfect classification.

In terms of computational efficiency, Naive Bayes and Logistic Regression proved to be the fastest models due to their streamlined training and prediction processes. In contrast, kNN was computationally expensive during the prediction phase because of its instance-based nature, which requires distance calculations for all training samples. Decision Tree provided a balance between accuracy and computational cost, excelling due to optimal hyperparameter tuning.

The trade-offs between performance and interpretability were also evident. Logistic Regression and Decision Tree provided insights into feature importance and decision-making processes, making them more transparent and interpretable. On the

other hand, while kNN and Naive Bayes performed exceptionally well, their underlying decision processes are less intuitive.

All in all, the simplicity of the Iris dataset limited the ability to differentiate between the classifiers significantly. However, this project successfully demonstrated the strengths and weaknesses of each model. The results suggest that for a well-separated and balanced dataset, simpler models such as Naive Bayes or kNN are sufficient. Whereas for more complex tasks or datasets with overlapping features, models like Decision Tree might be more suitable.

In conclusion, while all algorithms performed well, I would recommend kNN for the Iris dataset. It achieved perfect accuracy, and its simplicity aligns well with the characteristics of a well-separated and balanced dataset like Iris.

References

Machine learning. MIT Press. (2024, June 18). https://mitpress.mit.edu/9780262018029/machine-learning/

Duda, R. O., Hart, P. E., & Stork, D. G. (2000). *Pattern Classification.* Wiley-Interscience.