Machine Learning Fundamental Study

Vincent Chan
University of Hawaii at Manoa
ICS 435: Machine Learning

February 12, 2025

Abstract

In this paper, we implement and analyze the performance of three machine learning classifiers: K-Nearest Neighbors, Decision Tree, and Random Forest. Our focus is on classification tasks, specifically evaluating these models on the Breast Cancer dataset from scikit-learn. We assess model performance using Accuracy, Precision, Recall, and F1-score. Additionally, we explore the impact of hyperparameter tuning through an ablation study.

1 Introduction

The objective of this project is to classify breast cancer tumors as benign or malignant using three different machine learning models: K-Nearest Neighbors, Decision Tree, and Random Forest. We aim to evaluate their effectiveness using a range of performance metrics and investigate how hyperparameter tuning affects model performance.

2 Model Descriptions

2.1 K Nearest Neighbors

KNN is a learning algorithm that classifies a sample based on the majority class among its nearest neighbors. The hyperparameter (k) denotes denotes the number of neighbors to take in account when classifying a point.

2.2 Decision Tree

A Decision Tree is a tree-structured model that recursively splits the dataset based on feature values to classify instances. The depth of the tree significantly impacts performance, with deeper trees potentially overfitting.

2.3 Random Forest

Random Forest is an ensemble learning method that constructs multiple decision trees and aggregates their predictions. It introduces randomness through bootstrapped sampling and feature selection at each split, improving generalization in comparison to a single Decision Tree.

3 Experiment Design

3.1 Dataset Description

For the experimentation, the Breast Cancer dataset from scikit-learn is used, which includes 30 features that describe various characteristics of tumors. The dataset is divided into two classes: malignant (labeled as 1) and benign (labeled as 0). It contains a total of 569 samples, with 80% of the data (455 samples) used for training and the remaining 20% (114 samples) reserved for testing. The random seed used for the split is 50. Since KNN uses distance-based calculations the features are scaled

using StandardScaler before applying the KNN algorithm to ensure accurate calculations. The archive dataset can be found at
https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic

3.2 Experimental Setup

Each classifier is trained with various hyperparameter configurations to explore their effects on performance. For the KNN classifier, the number of neighbors (n_neighbors) is tested with values 5, 1, 10, 15, 20. The Decision Tree classifier is trained with different values for the maximum depth (max_depth), specifically None, 3, 5, 10, 20. For the Random Forest classifier, the number of estimators (n_estimators), the maximum depth (max_depth), and minimum samples split are varied, with n_estimators taking values 100, 200, max_depth set to None, 5, 10, and min_samples_split to 2,4. These configurations allow us to assess the impact of hyperparameter choices on classifier performance.

3.3 Evaluation Metrics

To assess model performance, we use several metrics that provide a comprehensive understanding of how well the classifiers are performing. **Accuracy** measures the proportion of correctly classified samples out of all samples. **Precision** is the ratio of correctly predicted positive cases to all predicted positives, indicating how many of the predicted positives are actually correct. **Recall** calculates the proportion of actual positive cases that are correctly identified by the model, reflecting the model's ability to detect positive instances. Finally, the **F1-Score** is the harmonic mean of Precision and Recall, offering a balance between false positives and false negatives, especially in cases where there is an uneven class distribution. These metrics together help in evaluating the trade-offs between different types of errors and the overall performance of the model.

3.4 Source Code and Implementation

https://github.com/eve-liya/ICS435/blob/main/435_assignment1.ipynb

3.5 Model Performance

The table below summarizes the results of the performance of each model with different hyperparameters on the test set. In boldface are the models that performed the best.

Model	Accuracy	Precision	Recall	F1-Score
KNN (k=5)	0.964 912	0.944 444	1.000 000	0.971 429
KNN (k=1)	0.938 596	0.942 029	0.955 882	0.948 905
KNN (k=10)	0.956 140	0.943 662	0.985 294	0.964 029
KNN (k=15)	0.938 596	0.906 667	1.000 000	0.951 049
KNN (k=20)	0.947 368	0.918 919	1.000 000	0.957 746
Decision Tree (depth=None)	0.921 053	0.940 299	0.926 471	0.933 333
Decision Tree (depth=3)	0.938 596	0.969 231	0.926 471	0.947 368
Decision Tree (depth=5)	0.929 825	0.954 545	0.926 471	0.940 299
Decision Tree (depth=10)	0.921 053	0.927 536	0.941 176	0.934 307
Decision Tree (depth=20)	0.921 053	0.927 536	0.941 176	0.934 307
Random Forest (trees=100, depth=None min_samples_split=2)	0.938 596	0.942 029	0.955 882	0.948 905
Random Forest (trees=200, depth=None min_samples_split=2)	0.938 596	0.942 029	0.955 882	0.948 905
Random Forest (trees=100, depth=5 min_samples_split=2)	0.938 596	0.942 029	0.955 882	0.948 905
Random Forest (trees=100, depth=10 min_samples_split=2)	0.947 368	0.955 882	0.955 882	0.955 882
Random Forest (trees=100, depth=None min_samples_split=4)	0.938 596	0.942 029	0.955 882	0.948 905
Random Forest (trees=200, depth=10 min_samples_split=4)	0.938 596	0.942 029	0.955 882	0.948 905

Table 1: Model Performance.

3.6 Ablation Study

The performance of the K-Nearest Neighbors (KNN) models varies based on the choice of k. The KNN model with k=1 shows lower overall accuracy (0.9386) and an F1-score of 0.9489, suggesting potential overfitting. Since k=1 creates highly flexible decision boundaries that closely follow training examples, it may struggle to generalize well. In contrast, the KNN model with k=5 achieves the best overall performance, with an F1-score of 0.9714, indicating a strong balance between precision and recall. As k increases to 10, the model maintains high accuracy but begins to smooth decision boundaries, leading to slightly lower performance. When k=15 and k=20, recall remains high (1.0), but accuracy drops slightly, suggesting a loss of precision and a possible trend toward underfitting.

The decision tree models show varying results depending on depth. The fully grown decision tree (unrestricted depth) has lower accuracy (0.9211) and an F1-score of 0.9333, which may indicate overfitting. However, a decision tree with depth 3 achieves a better balance, with an accuracy of 0.9386 and an F1-score of 0.9474. Increasing the depth further to 5 and 10 provides stable but marginally reduced performance, suggesting that deeper trees do not necessarily improve generalization. Beyond depth 10, no significant improvements are observed, reinforcing the idea that excessive depth leads to diminishing returns.

The random forest models demonstrate stable and consistent performance across different configurations. Most models achieve an accuracy of 0.9386 and an F1-score of 0.9489, showing that variations in depth or tree count do not significantly impact results. However, when the depth is explicitly set to 10, there is a slight improvement in accuracy (0.9474) and F1-score (0.9559), suggesting that some depth optimization can be beneficial. Increasing the number of trees from 100 to 200 does not lead to noticeable gains, indicating that additional trees beyond a certain point do not provide further advantages.

Overall, the best-performing model is KNN with k=5, as it effectively balances flexibility and generalization. The KNN model with k=1 and the decision tree with unrestricted depth exhibit signs of overfitting, while KNN with k=20 trends toward underfitting. Random forest models are robust and consistent, though increasing their complexity does not necessarily yield better results.

4 Conclusion

This report evaluated KNN, Decision Tree, and Random Forest models using accuracy, precision, recall, and F1-score. The KNN model with k=5 achieved the highest performance, with an F1-score of 0.9714, while k=1 led to overfitting and larger values like k=20 showed signs of underfitting. Decision trees performed best at moderate depths, with deeper trees offering diminishing returns. Random Forest models exhibited stable performance across different configurations, with depth tuning providing minor improvements. Overall, the best-performing model was KNN with k=5, demonstrating that, for this dataset, a carefully chosen k in KNN can provide strong generalization without requiring more complex ensemble methods.

5 Figures

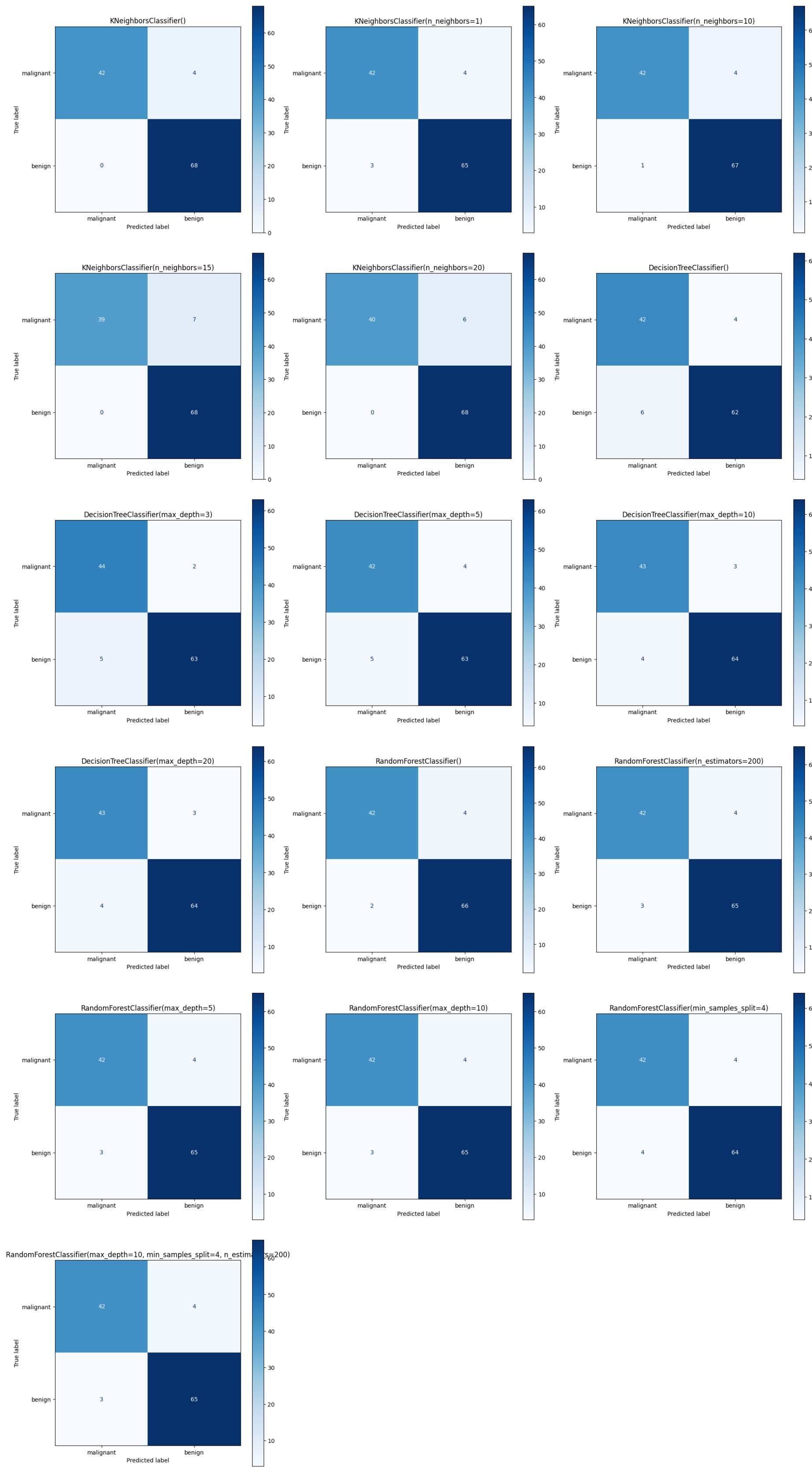

Figure 1: Confusion Matrices