**Title:**

**A Practical Guide to Decision Trees and Random Forests**

**Student name:**

**Student ID:**

# Introduction

In this tutorial, I explore and compare two well-known machine learning algorithms -Decision Trees and Random Forests. These models are often used for classification tasks because they’re easy to understand and usually work well out of the box. I wanted to see how each of them performs on a real-world dataset, so I used the Breast Cancer Wisconsin dataset, which contains information about tumors and whether they’re malignant or benign.

Decision Trees are simple and straightforward. They split data based on feature values to make predictions, kind of like asking a series of yes/no questions. But one big downside is that they can easily overfit meaning they learn the training data too well and don’t generalize as well to new, unseen data.

Random Forests try to fix that problem by building a bunch of different decision trees and combining their results. This helps reduce overfitting and usually gives better results overall. In this project, I’ll walk through how both models work, show how they perform on the dataset and also include some tuning of the models to improve accuracy.

The goal is to not only compare the results, but to understand why Random Forests tend to perform better, especially when we want more reliable results and better generalization even on clean, structured datasets like this one.

# Overview of Dataset

For this tutorial, I used the Breast Cancer Wisconsin dataset, which is included in the Scikit-learn library. It’s commonly used in machine learning projects because it’s clean, balanced, and based on real medical data. The dataset contains information about 569 tumor samples, each with 30 numerical features that describe characteristics of a cell nucleus, such as radius, texture, perimeter, area, and smoothness.

The goal is to classify each sample as either malignant (cancerous) or benign (non-cancerous). The dataset is already labeled, which makes it ideal for supervised learning techniques like Decision Trees and Random Forests.

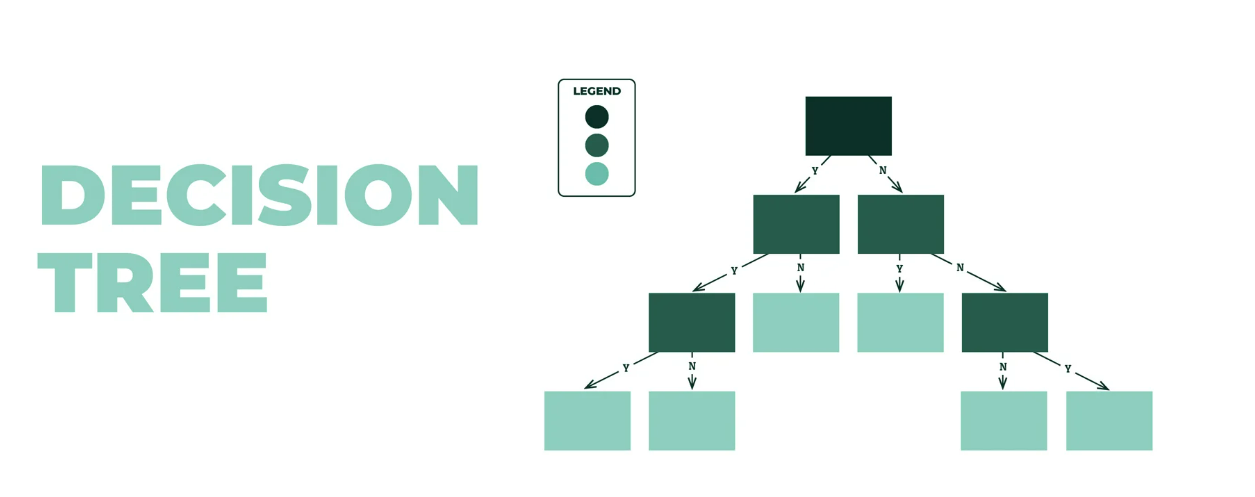
One reason this dataset works well for a comparison like this is that it’s not too large or messy, so I can clearly see how each model performs without spending too much time on data cleaning or preprocessing. It also has just enough features to give the models something to learn from, without being too complicated to interpret.

# What Are Decision Trees and Random Forests?

## Decision Trees

A Decision Tree is a model that splits data into branches based on feature values, kind of like asking a series of questions. Each split tries to group the data in a way that improves prediction accuracy. For example, it might split based on whether the “mean radius” is above or below a certain number.

One of the best things about Decision Trees is that they’re easy to understand. You can actually visualize how the model makes decisions. But there’s a catch they often overfit the training data, especially when the tree is very deep. That means they can do well on training data but struggle with new, unseen data.



**Step 1: Start at the Root**  
The model begins with the full dataset and looks for the best feature to split on the one that gives the most information gain. It usually uses **Gini Impurity** or **Entropy** to measure how “pure” a split is.

**Step 2: Split the Data**  
Once the best feature is chosen, the data is divided into two groups. The model repeats this process for each group, looking for the next best feature to split on.

**Step 3: Keep Splitting Until a Stopping Point**  
The tree keeps splitting the data until one of the stopping conditions is met, like:

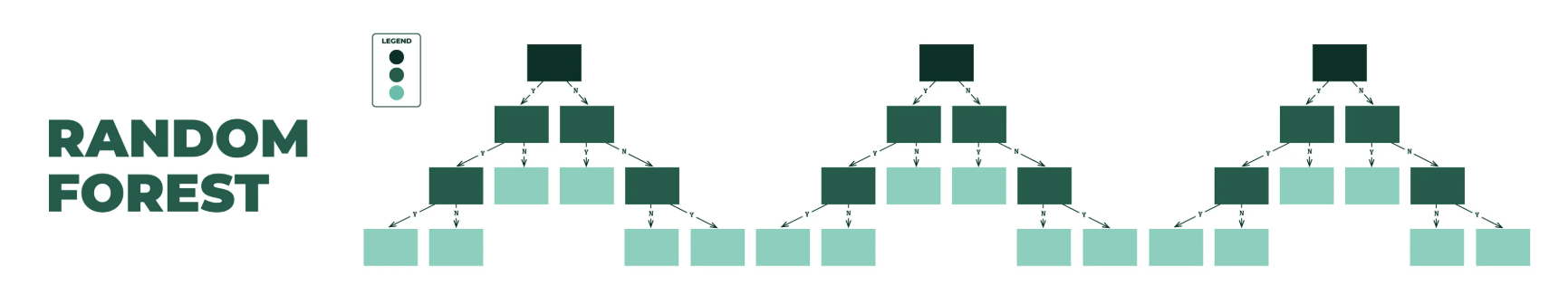
* All samples belong to the same class
* The max tree depth is reached
* There are too few samples left to split

**Step 4: Make Predictions**  
Once the tree is built, new data flows through it, following the branches based on feature values, until it reaches a leaf node and that’s the prediction.

## Random Forests

A Random Forest is basically a collection of Decision Trees. Instead of relying on just one tree, it creates many trees using different random subsets of the data and features and then combines their predictions. This process is called bagging, and it helps reduce overfitting and increase accuracy.

Random Forests are usually more accurate than a single Decision Tree because they average out the “mistakes” individual trees might make. The trade-off is that they’re harder to interpret, but they’re much more reliable in most real-world tasks including this one.



**Step 1: Create Bootstrapped Datasets**  
Instead of using the whole dataset, Random Forest takes random samples with replacements to build each tree. So each tree sees slightly different data.

**Step 2: Train Trees on Random Features**  
Each tree doesn’t get access to all the features. At each split, it randomly selects a subset of features, which adds diversity and reduces overfitting.

**Step 3: Build Multiple Decision Trees**  
Using this randomness, the model builds many individual trees usually 100 or more each trained slightly differently.

**Step 4: Aggregate the Results**  
When making predictions, all the trees "vote." For classification tasks like this one, the Random Forest takes a majority vote from all trees to decide the final prediction.

# Implementation and Evaluation

To test and compare the models, I used the Breast Cancer dataset from Scikit-learn. I split the data into 80% training and 20% testing. Then, I trained both a Decision Tree and a Random Forest classifier using the default settings first, and later tuned them to improve performance.

## Training the Models

* **Decision Tree**: Trained on all 30 features without limiting depth.
* **Random Forest**: Used 100 trees (n\_estimators=100) with automatic feature selection at each split.

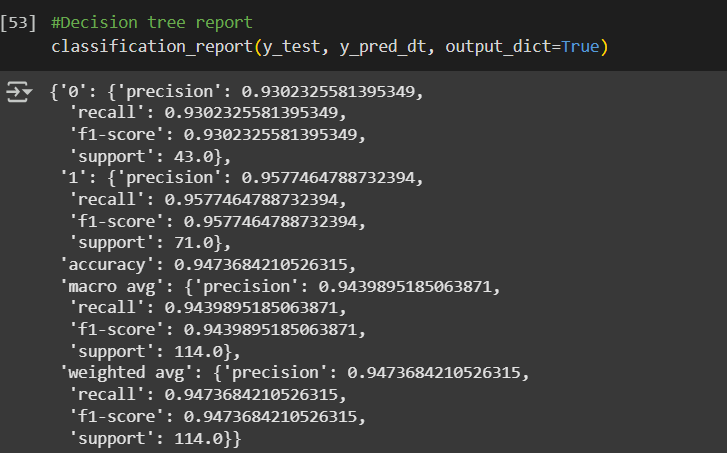
I used accuracy\_score and classification\_report to measure performance, along with confusion matrices to visualize the results.

## Implementation and Evaluation

I trained both a Decision Tree and a Random Forest on the Breast Cancer dataset. After splitting the data (80% train / 20% test), I evaluated both models using accuracy and classification metrics.

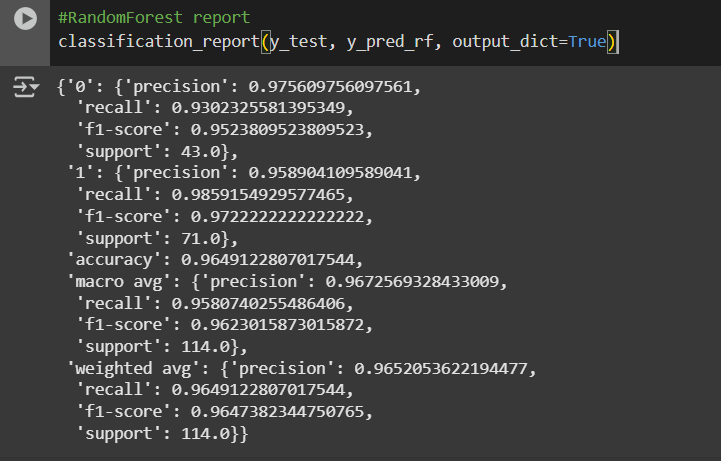
## Classification Report of Decision Tree

The Decision Tree model achieved an overall accuracy of 94.7% on the test set. It performed slightly better on class 1 (benign tumors) with a precision and recall of 95.8%, while class 0 (malignant tumors) had slightly lower but still strong values of 93.0%. The macro and weighted averages for precision, recall, and F1-score were all around 94.4–94.7%, showing that the model handled both classes fairly well. These results indicate that the Decision Tree provided a balanced and reliable performance, though not as high as Random Forest.



## Classification Report of Random Forest

The Random Forest model achieved a slightly higher accuracy of 96.5% on the test set. It performed exceptionally well on both classes. For malignant tumors (class 0), it had a precision of 97.6% and an F1-score of 95.2%. For benign tumors (class 1), it showed a recall of 98.6% and an F1-score of 97.2%. The macro and weighted averages were all above 96%, highlighting the model’s balanced performance across both classes. Compared to the Decision Tree, Random Forest was more consistent and made fewer classification errors.



## Feature Importance (Random Forest)

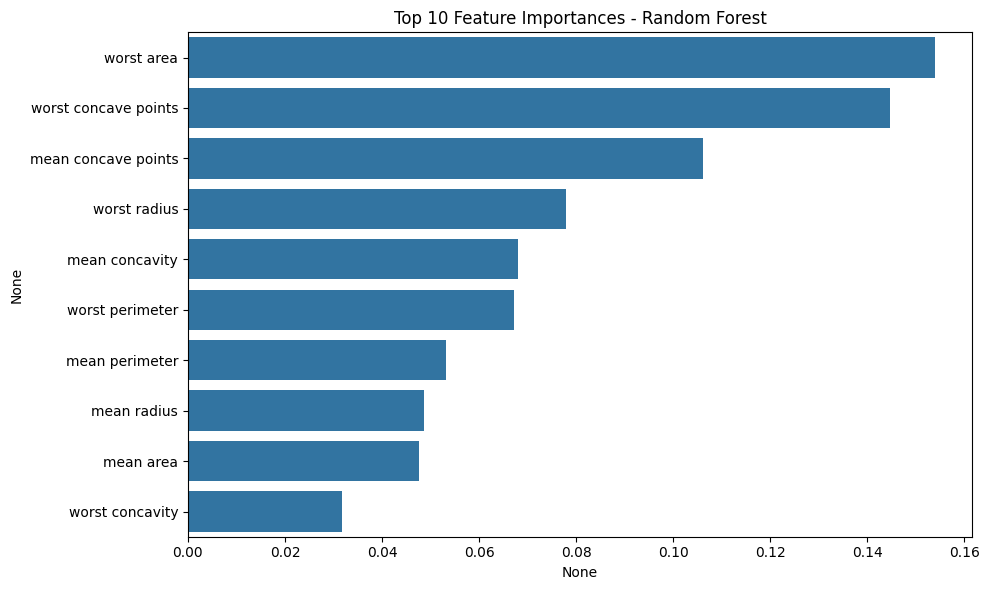
One of the advantages of using Random Forests is that they can show which features were most important in making predictions. After training the model, I checked the top features based on how much they contributed to splitting the data in the trees.

The most important features were:

* worst area
* worst concave points
* mean concave points
* worst radius
* mean concavity

These features had the highest influence on the model’s decisions, which makes sense since they relate to the size and shape of the tumor, which are key indicators when diagnosing breast cancer.

The feature importance plot gave a visual summary, making it easier to interpret which aspects of the data mattered most to the model. This kind of insight can be especially useful in healthcare, where understanding *why* a model makes certain predictions can help support clinical decisions.



# Hyperparameter Tuning

After testing both models with their default settings, I used GridSearchCV to try and improve their performance by tuning a few important hyperparameters.

## Decision Tree Tuning

For the Decision Tree, I focused on max\_depth, which controls how deep the tree can go. Deeper trees can capture more detail but are also more likely to overfit. The grid search tested several values, and the best result came with:

* max\_depth = 4

However, even with tuning, the test accuracy stayed the same at 94.7%, which shows that the default tree was already close to optimal for this dataset. The only change was a slight drop in training accuracy, meaning the tuned version was less overfitted.

## Random Forest Tuning

For the Random Forest, I tested a combination of n\_estimators (number of trees) and max\_depth. The best parameters found were:

* n\_estimators = 150
* max\_depth = 8

Even with these settings, the test accuracy stayed at 96.5%, just like before tuning. This suggests that the Random Forest was already well-generalized out of the box, and further tuning had little effect on performance likely due to the clean and structured nature of the dataset.

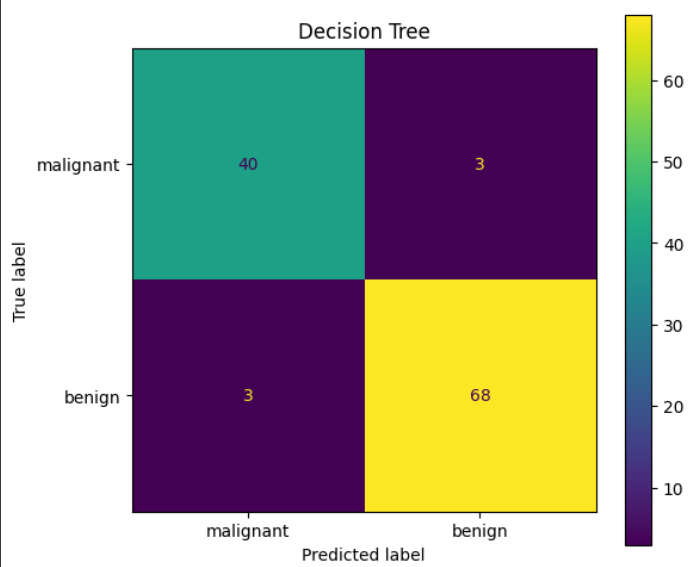
The tuning didn’t lead to a noticeable improvement in accuracy for either model, but it did help confirm that the models were already performing well. This is a good reminder that not every dataset needs aggressive tuning, especially when the defaults already give strong results.

# Confusion Matrix

To better understand how each model performed, I used confusion matrices to visualize the number of correct and incorrect predictions for both classes.

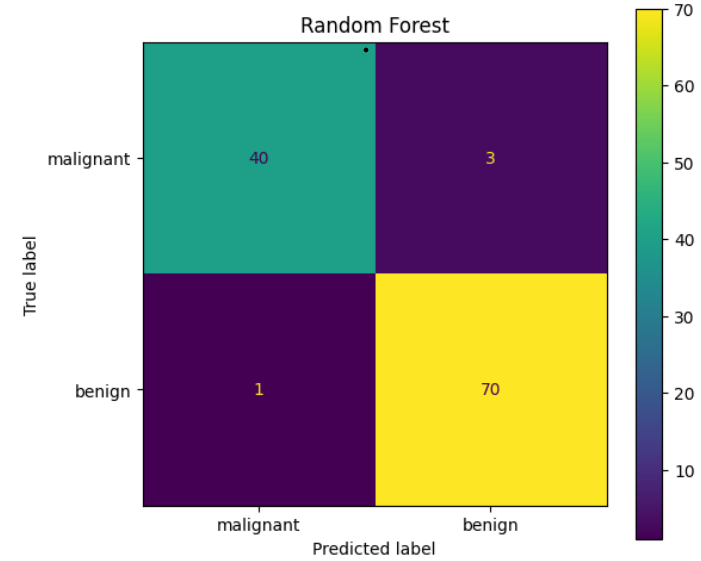
## Decision Tree

The confusion matrix for the Decision Tree showed that while it made good predictions overall, it misclassified a few malignant tumors as benign. This can be risky in medical applications because false negatives (missing a malignant case) can have serious consequences.



## Random Forest

The Random Forest had fewer misclassifications. It correctly identified more malignant tumors compared to the Decision Tree and had a more balanced performance across both classes. This confirms that Random Forest is better at generalizing and handling edge cases.



# Conclusion and Analysis

In this tutorial, I explored two classification models: Decision Trees and Random Forests, using the Breast Cancer Wisconsin dataset. Both models were tested on the same dataset and gave solid results, but Random Forest performed slightly better overall.

The Decision Tree achieved a test accuracy of 94.7 percent. It was easy to understand and visualize, but made a few more errors, especially with malignant cases. Even after tuning, the performance stayed the same, which showed that the model had already reached its limit with this dataset.

The Random Forest achieved a higher test accuracy of 96.5 percent, with fewer misclassifications and more balanced predictions across both classes. It also provided feature importance scores, which helped explain which tumor characteristics had the biggest influence on the model's decisions.

Although hyperparameter tuning was applied to both models, it did not change the final test accuracy. This showed that the default settings were already well-suited for the dataset, which is clean and well-structured.

Overall, Random Forest proved to be the better model in this comparison. It was more accurate, consistent, and reliable, making it a stronger option for classification tasks in medical diagnosis and similar fields.

# References

* Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1), 5–32.
* Quinlan, J. R. (1986). Induction of Decision Trees. *Machine Learning*, 1, 81–106
* Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
* UCI Machine Learning Repository. Breast Cancer Wisconsin (Diagnostic) Data Set.
* Scikit-learn Documentation – Decision Trees
* Scikit-learn Documentation – Random Forests