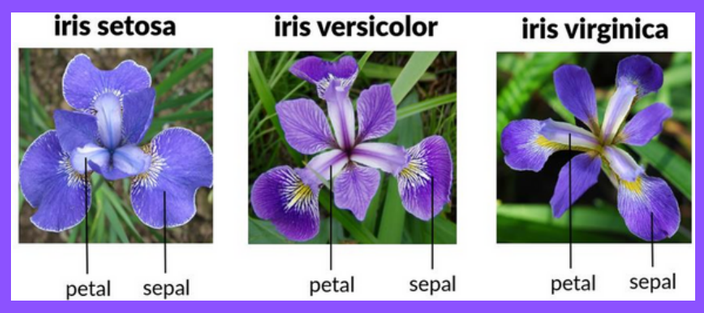
# SUPERVISED MACHINE LEARNING WITH IRIS DATASET

Logistic Regression, K-Nearest Neighbor (K-NN), Decision Tree, Random Forest, and Support Vector Machine (SVM)

by: Raniah Mufidah Admayana

Code in Google Colab = https://colab.research.google.com/drive/1c2TE-WVX2AMGtQK6HlsRuCYO2I1zwGFX?usp=sharing




Three classes of IRIS dataset (Mijwil & Abttan, 2021)

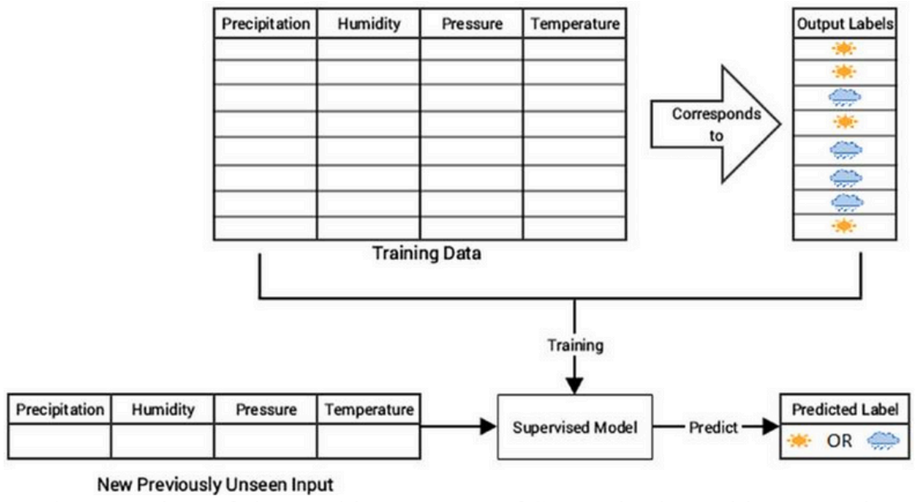
## INTRODUCTION

### IRIS DATASET

Iris dataset is a classic dataset in machine learning and statistics. It classifies three types of iris flowers based on sepal and petal measurements and was originally analyzed by R.A. Fisher in the 1930s (Marsland, 2014).

### SUPERVISED MACHINE LEARNING

Supervised Machine Learning is a method where a model learns from labeled data to recognize patterns. It uses this knowledge to predict the correct category or value for new data (Sarkar et al., 2017).




An overview of the working process of Supervised Machine Learning (Sarkar et al., 2017)

### RESEARCH PURPOSE

Determining the best Supervised Machine Learning model for the Iris dataset among Logistic Regression, K-NN, Decision Tree, Random Forest, and SVM.

## LITERATURE REVIEW

### SUPERVISED MACHINE LEARNING MODELS

**1. Logistic Regression**

Logistic Regression models the probability of binary classification, assigning the positive class (1) if it exceeds a threshold (commonly 50%) and the negative class (0) if below (Géron, 2017).

**2. K-NN**

K-NN classifies data based on the k closest points in the dataset. It measures the distance to find the nearest neighbors and assigns the most common class among them. The parameter k represents the number of neighbors considered for classification (Géron, 2017; Marsland, 2014).

**3. Decision Tree**

A Decision Tree uses a series of decision rules to split data at decision nodes, forming a tree-like structure. Each rule creates branches leading to new nodes, with terminal branches called leaves. This model is popular for its interpretability and serves as the foundation for various tree-based extensions (Albon, 2018).

**4. Random Forest**

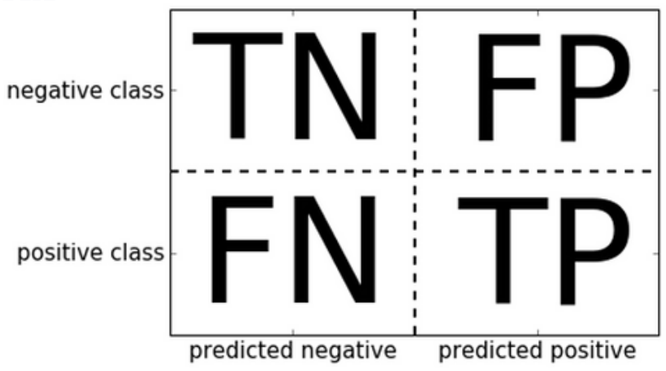
Random Forest is a collection of Decision Trees that work together (Albon, 2018).

**5. Support Vector Machine**

Support Vector Machine (SVM) determines the optimal boundary (hyperplane) that separates classes by creating the widest possible margin. To handle complex, non-linear data, SVM transforms it into a higher-dimensional space using kernel functions. However, it requires high computational power for large datasets (Marsland, 2014).

### MODEL EVALUATION

**Confusion Matrix**




Typical structure of a confusion matrix (Müller & Guido, 2016).

From the confusion matrix, we derive key **Performance metrics**:

- **Accuracy**: The proportion of correct predictions.
- **Precision**: The percentage of predicted positives that are actually correct.
- **Recall (Sensitivity)**: The percentage of actual positives correctly identified.
- **F1 Score**: The harmonic mean of precision and recall, balancing both metrics.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1\ Score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

(Sarkar et al., 2017)

## RESEARCH METHODOLOGY

**DATA SOURCE** → secondary data from the website https://www.kaggle.com/datasets/uciml/iris, accessed on December 17, 2024.

### RESEARCH VARIABLES

**Independent Variables (Features)**: represented by the symbol "X"

- Sepal Length (cm)
- Sepal Width (cm)
- Petal Length (cm)
- Petal Width (cm)

**Dependent Variable (Target/Label)**: represented by the symbol "y"
Species (Iris-setosa, Iris-versicolor, Iris-virginica)

### Analysis Steps

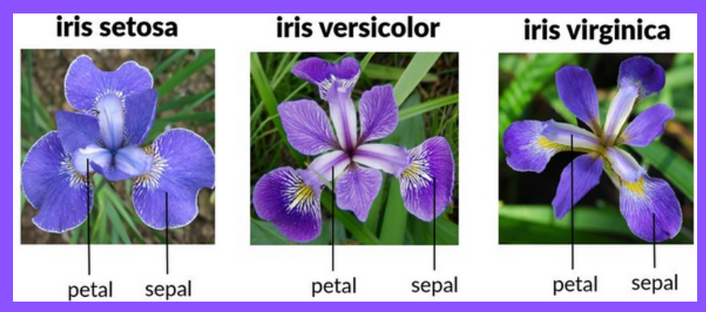
1. Identify the problem and input the dataset.
2. Exploratory Data Analysis
3. Split the dataset into training and testing sets.
4. Train the models and evaluate their performance.
5. Summarize the results and draw conclusions.

# SUPERVISED MACHINE LEARNING WITH IRIS DATASET

Logistic Regression, K-Nearest Neighbor (K-NN), Decision Tree, Random Forest, and Support Vector Machine (SVM)

by: Raniah Mufidah Admayana

Code in Google Colab = https://colab.research.google.com/drive/1c2TE-WVX2AMGtQK6HlsRuCYO2I1zwGFX?usp=sharing




Three classes of IRIS dataset (Mijwil & Abttan, 2021)

# DISCUSSION

## 1. INPUT THE DATASET

change all column names into lowercase

```
df = pd.read_csv('/content/iris.csv')
```

```
df.columns = df.columns.str.lower().str.replace(' ', '_')
```

```
df.head()
```

| | id | sepallengthcm | sepalwidthcm | petallengthcm | petalwidthcm | species |
|---|---|---|---|---|---|---|
| 0 | 1 | 5.1 | 3.5 | 1.4 | 0.2 | Iris-setosa |
| 1 | 2 | 4.9 | 3.0 | 1.4 | 0.2 | Iris-setosa |
| 2 | 3 | 4.7 | 3.2 | 1.3 | 0.2 | Iris-setosa |
| 3 | 4 | 4.6 | 3.1 | 1.5 | 0.2 | Iris-setosa |
| 4 | 5 | 5.0 | 3.6 | 1.4 | 0.2 | Iris-setosa |

## 2. EXPLORATORY DATA ANALYSIS

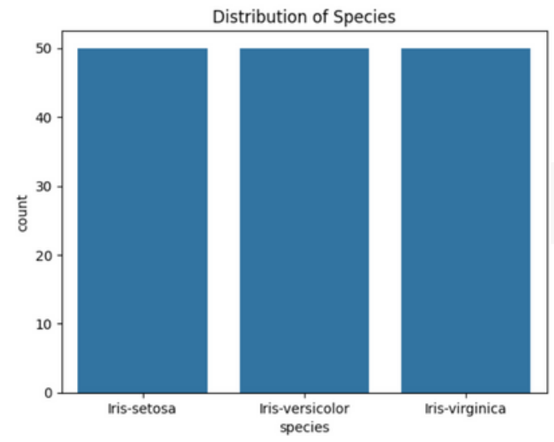
Clean, explore, and visualize the data

```
df.isnull().sum()
# To see the missing values
```

| | 0 |
|---|---|
| id | 0 |
| sepallengthcm | 0 |
| sepalwidthcm | 0 |
| petallengthcm | 0 |
| petalwidthcm | 0 |
| species | 0 |

No missing value

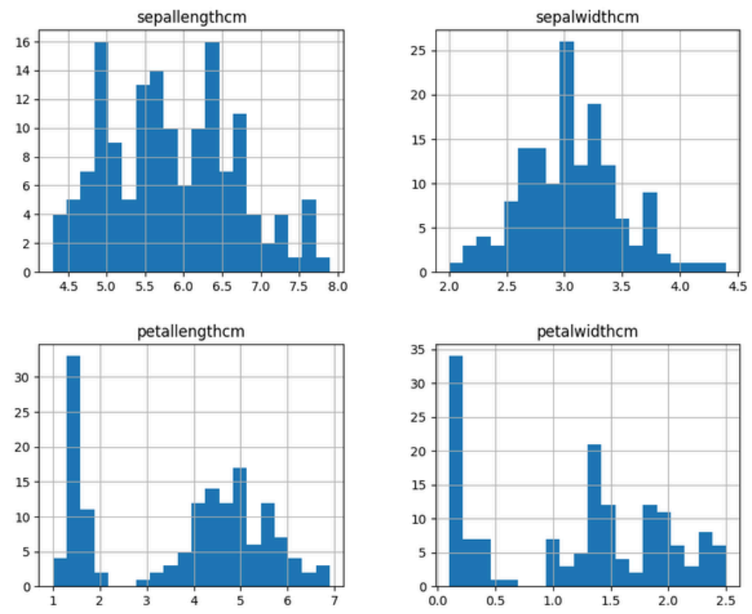
```
# Species distribution
sns.countplot(x='species', data=df)
plt.title('Distribution of Species')
plt.show()
```




```
# Species distribution
print(df['species'].value_counts())
```

```
species
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: count, dtype: int64
```

the amount of data between species is evenly distributed

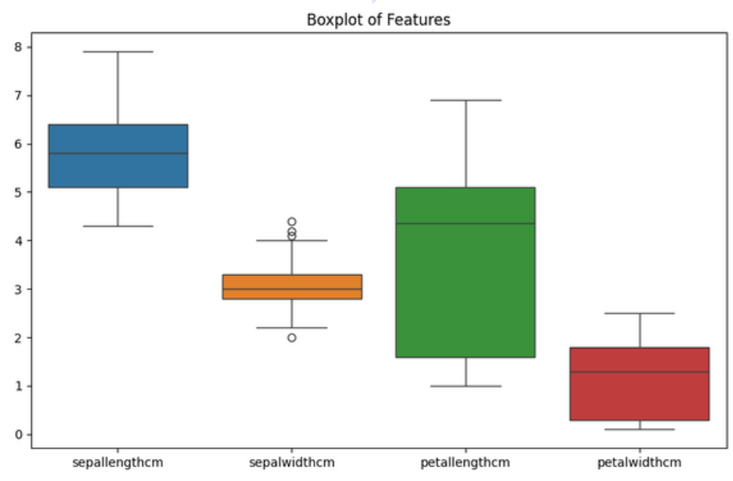


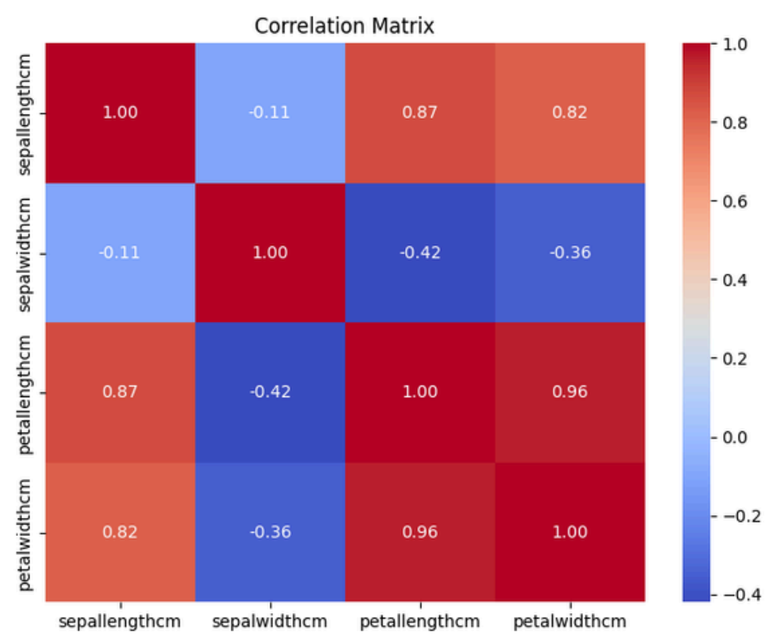

### Data Distribution & Outlier Detection

```
df.drop(columns=['id']).describe()
```

| | sepallengthcm | sepalwidthcm | petallengthcm | petalwidthcm |
|---|---|---|---|---|
| count | 150.000000 | 150.000000 | 150.000000 | 150.000000 |
| mean | 5.843333 | 3.054000 | 3.758667 | 1.198667 |
| std | 0.828066 | 0.433594 | 1.764420 | 0.763161 |
| min | 4.300000 | 2.000000 | 1.000000 | 0.100000 |
| 25% | 5.100000 | 2.800000 | 1.600000 | 0.300000 |
| 50% | 5.800000 | 3.000000 | 4.350000 | 1.300000 |
| 75% | 6.400000 | 3.300000 | 5.100000 | 1.800000 |
| max | 7.900000 | 4.400000 | 6.900000 | 2.500000 |

Outliers can be ignored



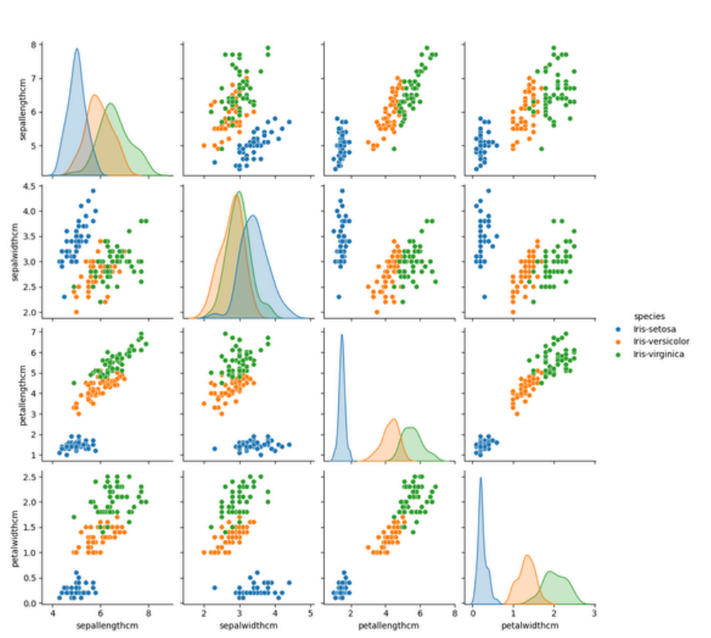

### Correlation between features

Correlation Matrix shows the strength of the relationship between features.

- a number close to 1 or -1 → have strong correlation
- 0 → don't have a correlation

(Sarkar et al., 2017)

Features that have strong correlation
- petallengthcm & petalwidthcm (0.96)
- sepallengthcm & petallengthcm (0.87)
- sepallengthcm & petalwidthcm (0.82)




## 3. SPLIT THE DATASET

80% Train Data → 120 Data
20% Test Data → 30 Data

```
X = df.drop(columns=['id', 'species'])
y = df['species']
```

```
# 80% train, 20% test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)
```

## 4. MACHINE LEARNING MODELS

### 4.1 Logistic Regression

```
# Creating and training a Logistic Regression model
model_logistic_regression = LogisticRegression()
model_logistic_regression.fit(X_train, y_train)

# Predict
y_pred = model_logistic_regression.predict(X_test)

# Calculate the accuracy
accuracy_logistic_regression = accuracy_score(y_test, y_pred)
print(f'Accuracy of Logistic Regression Model: {accuracy_logistic_regression * 100:.2f}%\n')

print('Classification Report:')
print(classification_report(y_test, y_pred))
```

Accuracy = 96.67%

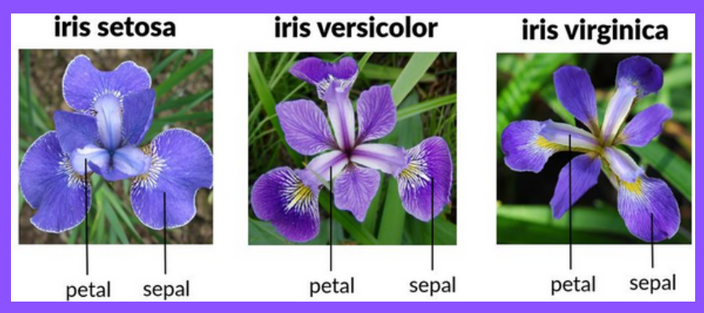
```
Accuracy of Logistic Regression Model: 96.67%

Classification Report:
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        11
Iris-versicolor       1.00      0.92      0.96        13
 Iris-virginica       0.86      1.00      0.92         6

       accuracy                           0.97        30
      macro avg       0.95      0.97      0.96        30
   weighted avg       0.97      0.97      0.97        30
```

# SUPERVISED MACHINE LEARNING WITH IRIS DATASET

Logistic Regression, K-Nearest Neighbor (K-NN), Decision Tree, Random Forest, and Support Vector Machine (SVM)

by: Raniah Mufidah Admayana

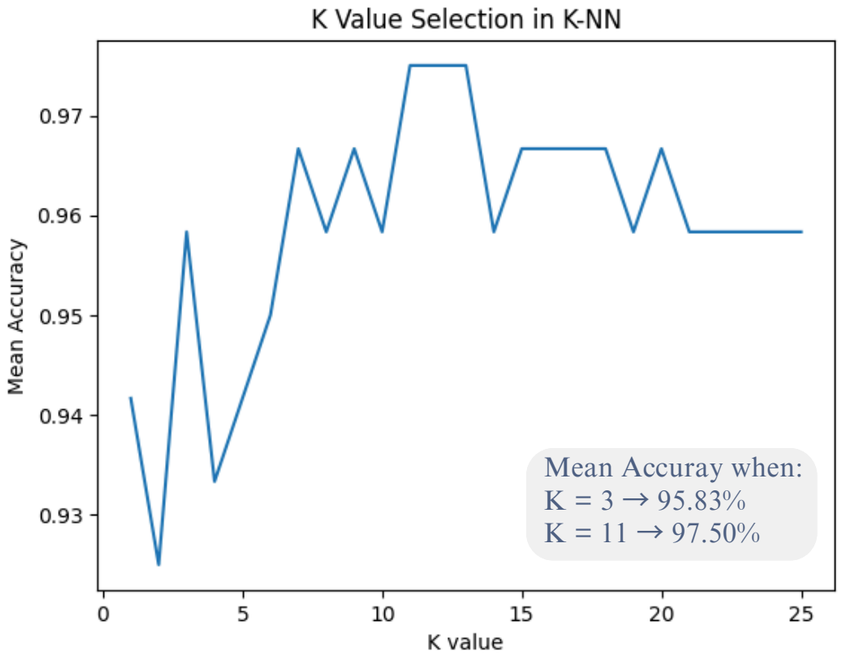
Code in Google Colab = https://colab.research.google.com/drive/1c2TE-WVX2AMGtQK6HlsRuCYO2I1zwGFX?usp=sharing




Three classes of IRIS dataset (Mijwil & Abttan, 2021)

## 4.2 K-Nearest Neighbor

### 1. Search the best K




### 2. Compare the accuracy of K-NN models for both K values after applying them to test data

K = 3: Accuracy of KNN model = 100.00%

Classification Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Iris-setosa | 1.00 | 1.00 | 1.00 | 11 |
| Iris-versicolor | 1.00 | 1.00 | 1.00 | 13 |
| Iris-virginica | 1.00 | 1.00 | 1.00 | 6 |
| accuracy | | | 1.00 | 30 |
| macro avg | 1.00 | 1.00 | 1.00 | 30 |
| weighted avg | 1.00 | 1.00 | 1.00 | 30 |

K = 11: Accuracy of KNN model = 96.67%

Classification Report:

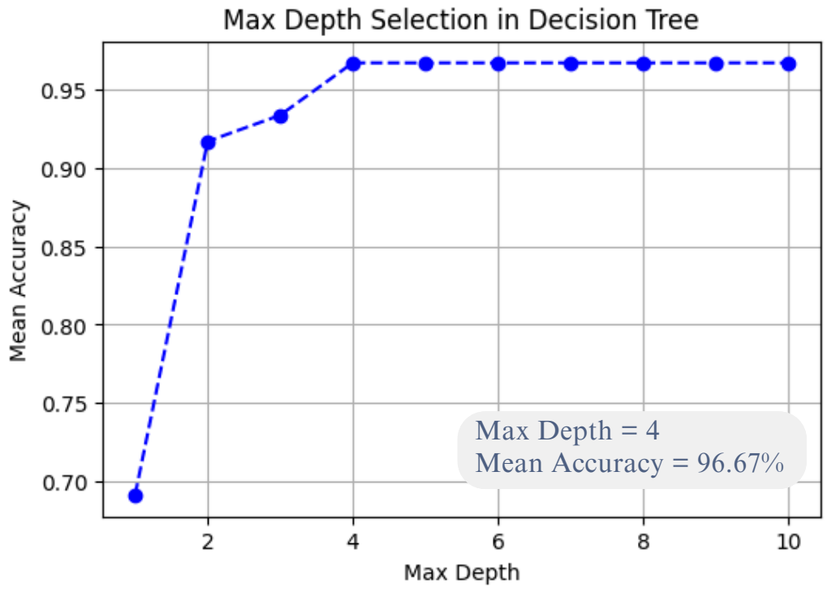
| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Iris-setosa | 1.00 | 1.00 | 1.00 | 11 |
| Iris-versicolor | 1.00 | 0.92 | 0.96 | 13 |
| Iris-virginica | 0.86 | 1.00 | 0.92 | 6 |
| accuracy | | | 0.97 | 30 |
| macro avg | 0.95 | 0.97 | 0.96 | 30 |
| weighted avg | 0.97 | 0.97 | 0.97 | 30 |

K = 3 (100% accuracy): Since the Iris dataset is small, the model may overfit, leading to poor performance on new data. It might only work well on small datasets.

K = 11 (96.67% accuracy): Larger k value tends to be more stable and generalizable, reducing overfitting and better suited for real-world, diverse data.

## 4.3 Decision Tree

### 1. Search the best depth




### 2. Search accuracy of Decision Tree after applying them to test data

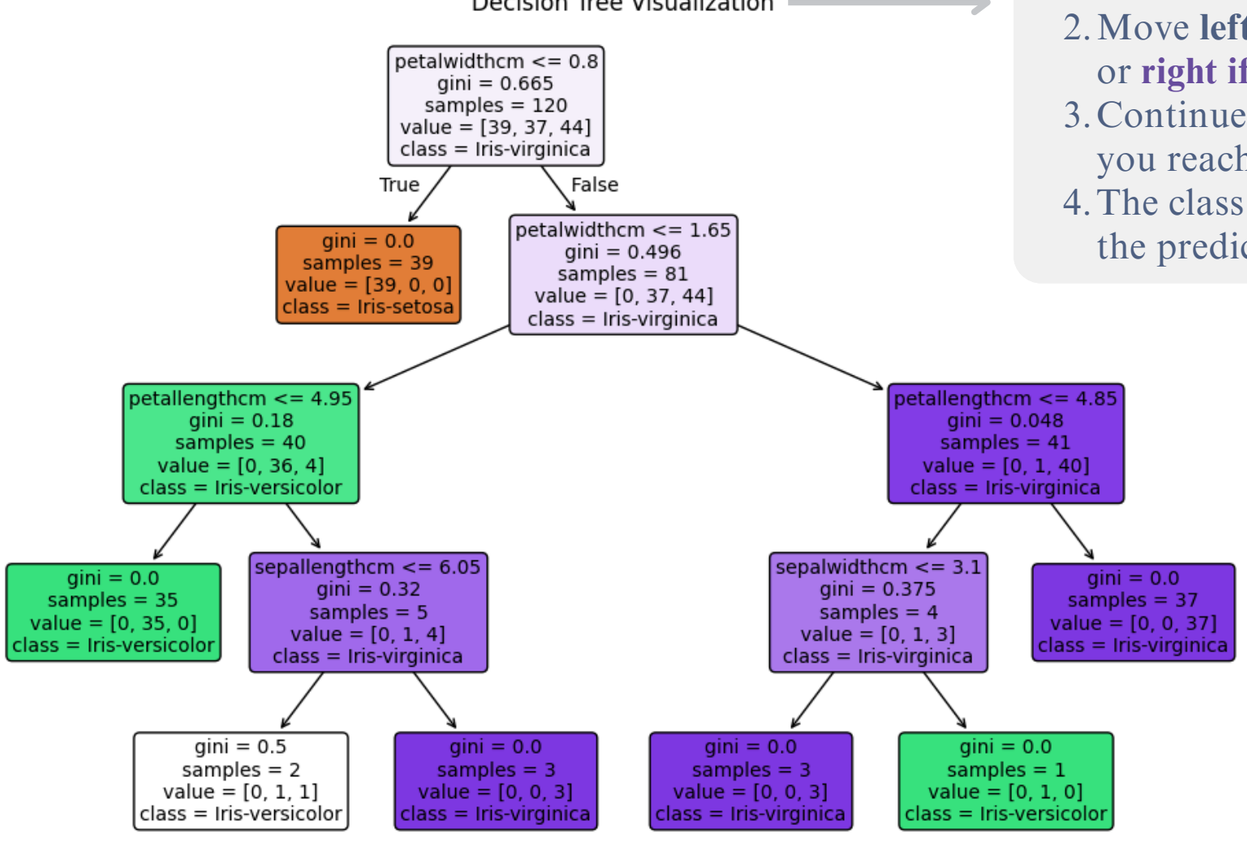
Accuracy = 96.67%

Accuracy of Decision Tree Model: 96.67%

Classification Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Iris-setosa | 1.00 | 1.00 | 1.00 | 11 |
| Iris-versicolor | 1.00 | 0.92 | 0.96 | 13 |
| Iris-virginica | 0.86 | 1.00 | 0.92 | 6 |
| accuracy | | | 0.97 | 30 |
| macro avg | 0.95 | 0.97 | 0.96 | 30 |
| weighted avg | 0.97 | 0.97 | 0.97 | 30 |

Decision Tree Visualization

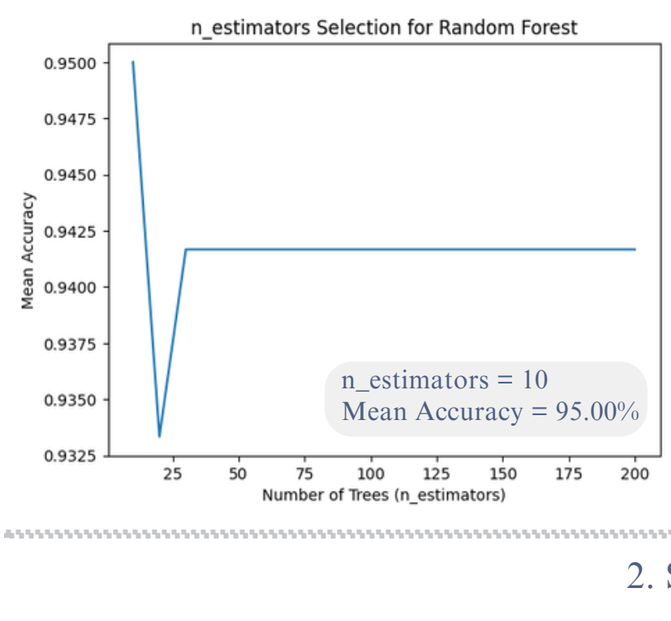



How to Read Decision Tree :

1. Start at the root node (the top).
2. Move **left** if the condition is **True**, or **right if False**.
3. Continue down the branches until you reach a leaf node.
4. The class label at the leaf node is the predicted class

## 4.4 Random Forest

### 1. Search the best n_estimators




### 2. Search accuracy of Random Forest after applying it to test data

Accuracy = 96.67%

Accuracy of Random Forest Model: 96.67%

Classification Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Iris-setosa | 1.00 | 1.00 | 1.00 | 11 |
| Iris-versicolor | 1.00 | 0.92 | 0.96 | 13 |
| Iris-virginica | 0.86 | 1.00 | 0.92 | 6 |
| accuracy | | | 0.97 | 30 |
| macro avg | 0.95 | 0.97 | 0.96 | 30 |
| weighted avg | 0.97 | 0.97 | 0.97 | 30 |

## 4.5 Support Vector Machine

### 1. Search the best kernel

Kernel = linear: Mean Accuracy = 0.9833
Kernel = poly: Mean Accuracy = 0.9750
Kernel = rbf: Mean Accuracy = 0.9750
Kernel = sigmoid: Mean Accuracy = 0.3667

### 2. Search accuracy of SVM after applying it to test data

With kernel: linear, the accuracy of SVM Model: 100.00%

Accuracy = 100%

Classification Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Iris-setosa | 1.00 | 1.00 | 1.00 | 11 |
| Iris-versicolor | 1.00 | 1.00 | 1.00 | 13 |
| Iris-virginica | 1.00 | 1.00 | 1.00 | 6 |
| accuracy | | | 1.00 | 30 |
| macro avg | 1.00 | 1.00 | 1.00 | 30 |
| weighted avg | 1.00 | 1.00 | 1.00 | 30 |

## CONCLUSION

Accuracy of =

- Logistic Regression → 96.67%
- K-NN → 96.67%
- Decision Tree → 96.67%
- Random Forest → 96.67%
- SVM → 100%

Support Vector Machine (SVM) is the best machine learning model for Iris datasset

## CODE IN COLAB

Google Colab
co google.com
https://colab.research.google.com/drive/1c2TE-WVX2AMGtQK6HlsRuCYO2I1zwGFX?usp=sharing

## REFERENCE

Albon, C. (2018). Machine Learning with Python Cookbook: Practical Solutions from Preprocessing to Deep Learning.
Géron, A. (2017). Hands-On Machine Learning with Scikit-Learn and TensorFlow: Techniques and Tools to Build Learning Machines. O'Reilly Media.
Marsland, S. (2014). Machine learning: An Algorithmic Perspective, Second Edition. CRC Press.
Mijwil, M. M., & Abttan, R. A. (2021). Utilizing the genetic algorithm to pruning the C4.5 decision tree algorithm. Asian Journal of Applied Sciences, 9(1). https://doi.org/10.24203/ajas.v9i1.6503
Müller, A. C., & Guido, S. (2016). Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media.
Sarkar, D., Bali, R., & Sharma, T. (2017). Practical Machine Learning with Python: A Problem-Solver's Guide to Building Real-World Intelligent Systems. Apress.