**Iris Dataset Classification with Multiple ML Algorithms**

Iris Flower Classification is one of the most popular case studies among the data science community. Almost every data science newbie has solved this case study once in their life. Here, you are given the measurements associated with each species of iris flower and based on this data, you have to train a machine learning model for the task of classifying iris flowers. So if you are new to machine learning and have never tried to solve this case study, this article is for you. In this article, I will walk you through Iris Flower Classification with machine learning using Python.

**Iris Flower Classification**

Iris flower has three species; setosa, versicolor, and virginica, which differs according to their measurements. Now assume that you have the measurements of the iris flowers according to their species, and here your task is to train a machine learning model that can learn from the measurements of the iris species and classify them.

I hope you now have understood the case study of iris flower classification. Although the Scikit-learn library provides a dataset for iris flower classification, you can also download the same dataset from here for the task of iris flower classification with Machine Learning. Now in the section below, I will take you through how we can classify the iris flower species with machine learning using the Python programming language.

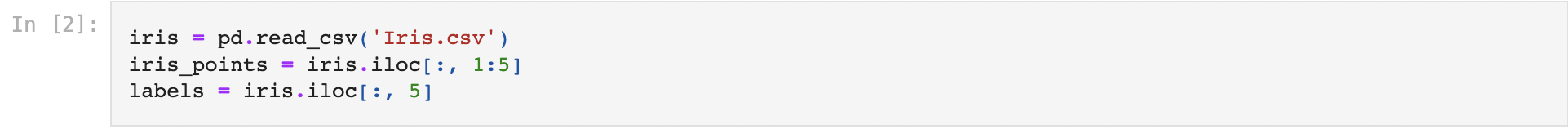
**Iris Flower Classification using Python**

I will start the task of Iris flower classification by importing the necessary Python libraries and the dataset that we need for this task:

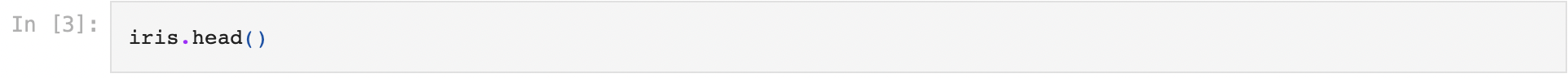
A white rectangular object with a black border

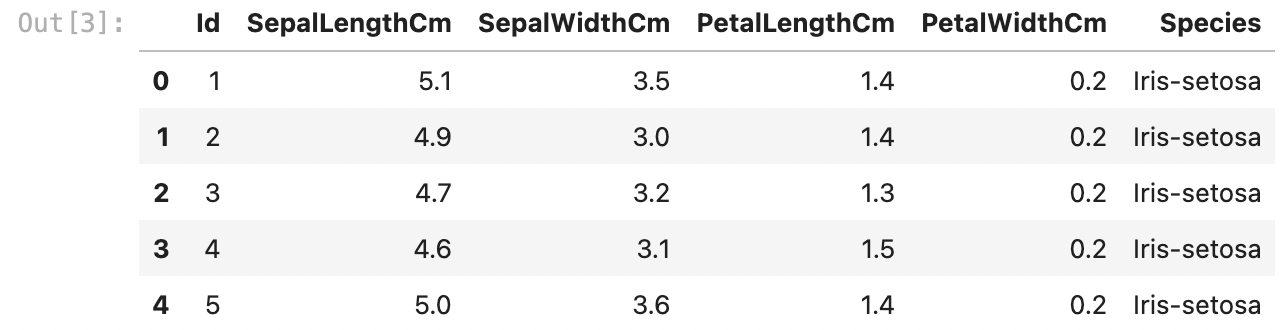
Description automatically generated

Import the dataset from your local desktop. Use pandas for it. Enter the path to the dataset file in the read\_csv method. It will import the iris dataset.

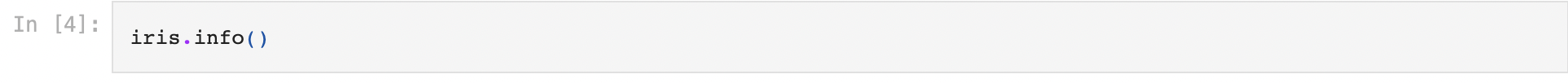


Now let’s have a look at the first five rows of this dataset:





View the info of the data frame that contains details like the count of non-null variables and the column’s datatype along with the column names. It will also show the memory usage.



<class 'pandas.core.frame.DataFrame'>

RangeIndex: 150 entries, 0 to 149

Data columns (total 6 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Id 150 non-null int64

1 SepalLengthCm 150 non-null float64

2 SepalWidthCm 150 non-null float64

3 PetalLengthCm 150 non-null float64

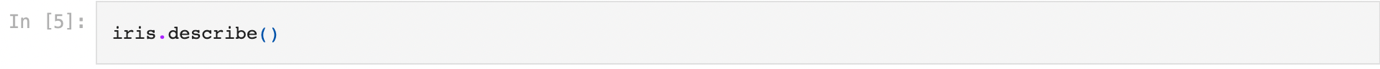
4 PetalWidthCm 150 non-null float64

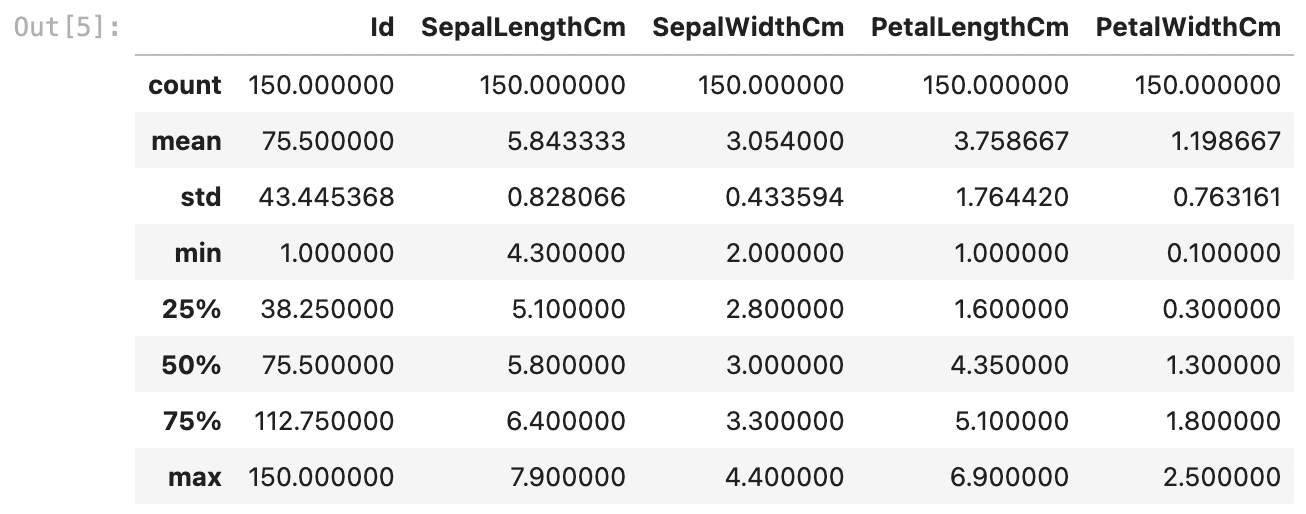
5 Species 150 non-null object

dtypes: float64(4), int64(1), object(1)

memory usage: 7.2+ KB

Now let’s have a look at the descriptive statistics of this dataset:





If there are any missing values, then modify them before using the dataset. For modifying you can use the fillna() method. It will fill null values.



Id 0

SepalLengthCm 0

SepalWidthCm 0

PetalLengthCm 0

PetalWidthCm 0

Species 0

dtype: int64

We can see that all values are 0. It means that there are no null values over the entire data frame.

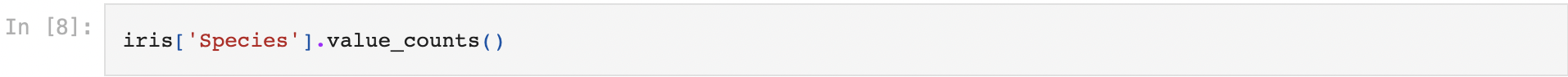
The target labels of this dataset are present in the species column, let’s have a quick look at the target labels:



Target Labels ['Iris-setosa' 'Iris-versicolor' 'Iris-virginica']

# Visualizations

View the count plot of species feature using seaborn.

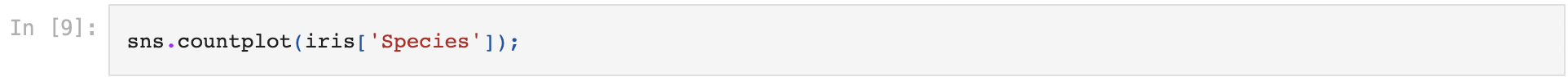


Iris-setosa 50

Iris-versicolor 50

Iris-virginica 50

Name: Species, dtype: int64



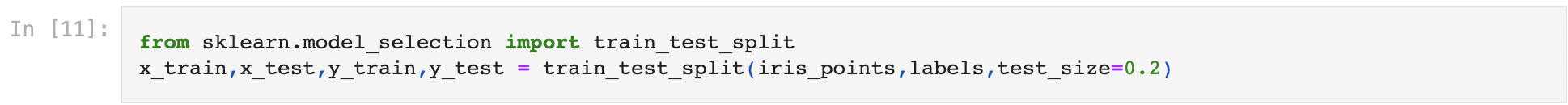
A chart of different colors

Description automatically generated



# Split Data Into Testing and Training Data

Before training any kind of ML model, we first need to split data into testing and training data using the train\_test\_split function from sklearn.



# Normalization/Standardization of Data

Before we work on the ML modeling and the data processing, we need to normalize the data for which the code is mentioned below.

A screenshot of a computer

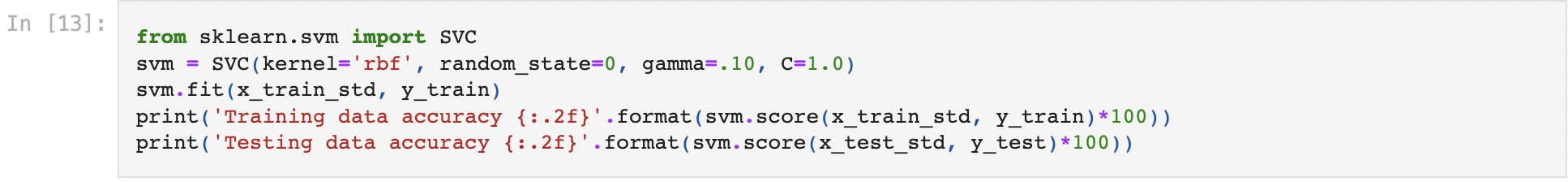
Description automatically generated

# Applying Classification ML model

Now that our data is prepared and is ready to go into the various ML models we will be testing and comparing the efficiency of various classification models

# SVM (Support Vector Machine)

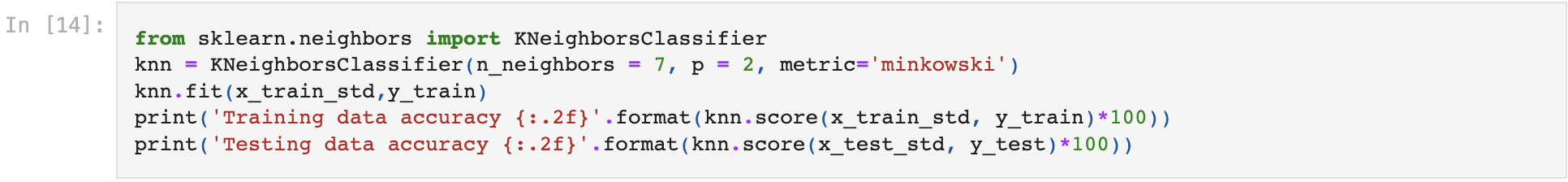
The first model we are going to test the SVM Classifier. The code for the same is mentioned below.



Training data accuracy 96.67

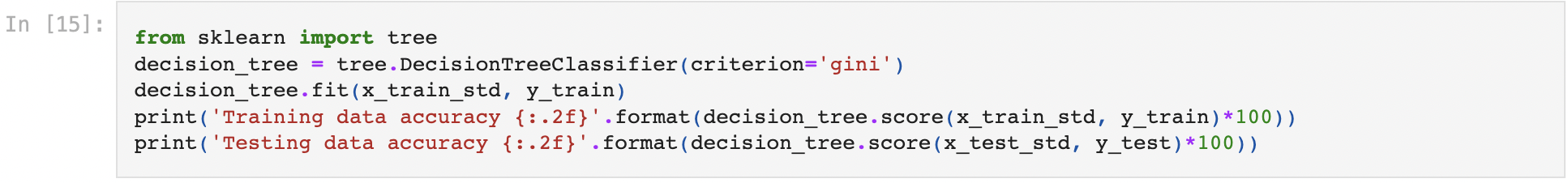
Testing data accuracy 100.00

**KNN (K-Nearest Neighbors)**



Training data accuracy 96.67

Testing data accuracy 100.00

**Decision Tree**

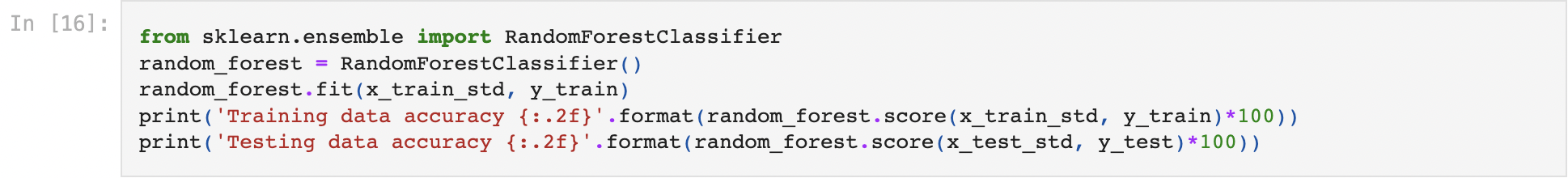
Training data accuracy 100.00

Testing data accuracy 100.00

The testing accuray in this model as well is still around 80%, hence so far SVM gives the best results.

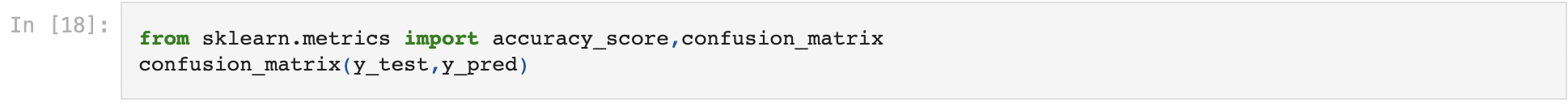
# Random Forest

Random Forest is a more complex and better decision tree in Machine Learning. The implementation of same is shown below.



Training data accuracy 100.00

Testing data accuracy 100.00





array([[10, 0, 0],

[ 0, 14, 0],

[ 0, 0, 6]], dtype=int64)



Accuracy of the model is 100.00

We can see that accuracy of the model is 96.67 percent which is very accurate.

# Summary

So this is how you can train a machine learning model for the task of Iris classification using Python. Iris Classification is one of the most popular case studies among the data science community. Almost every data science newbie has solved this case study once in their life. I hope you liked this article on the task of classifying Iris species with machine learning using Python. Feel free to ask your valuable questions in the comments section below.