# UNIT 5

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon **NumPy, SciPy** and **Matplotlib**.

## Origin of Scikit-Learn

It was originally called ***scikits.learn*** and was initially developed by David Cournapeau as a Google summer of code project in 2007. Later, in 2010, Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, and Vincent Michel, from FIRCA (French Institute for Research in Computer Science and Automation), took this project at another level and made the first public release (v0.1 beta) on 1st Feb. 2010.

Let's have a look at its version history −

May 2019: scikit-learn 0.21.0

March 2019: scikit-learn 0.20.3

December 2018: scikit-learn 0.20.2

November 2018: scikit-learn 0.20.1

September 2018: scikit-learn 0.20.0

July 2018: scikit-learn 0.19.2

July 2017: scikit-learn 0.19.0

September 2016. scikit-learn 0.18.0

November 2015. scikit-learn 0.17.0

March 2015. scikit-learn 0.16.0

July 2014. scikit-learn 0.15.0

August 2013. scikit-learn 0.14

## Prerequisites

Before we start using scikit-learn latest release, we require the following −

Python (|=3.5)

NumPy (|= 1.11.0)

Scipy (|= 0.17.0)li

Joblib (|= 0.11)

Matplotlib (|= 1.5.1) is required for Sklearn plotting capabilities.

Pandas (|= 0.18.0) is required for some of the scikit-learn examples using data structure and analysis.

# Installation

If you already installed NumPy and Scipy, following are the two easiest ways to install scikit-learn −

## Using pip

Following command can be used to install scikit-learn via pip −

```
pip install -U scikit-learn
```

## Using conda

Following command can be used to install scikit-learn via conda −

```
conda install scikit-learn
```

On the other hand, if NumPy and Scipy is not yet installed on your Python workstation then, you can install them by using either **pip** or **conda**.

Another option to use scikit-learn is to use Python distributions like ***Canopy*** and ***Anaconda*** because they both ship the latest version of scikit-learn.

## Features

Rather than focusing on loading, manipulating and summarising data, Scikit-learn library is focused on modeling the data. Some of the most popular groups of models provided by Sklearn are as follows −

**Supervised Learning algorithms** − Almost all the popular supervised learning algorithms, like Linear Regression, Support Vector Machine (SVM), Decision Tree etc., are the part of scikit-learn.

**Unsupervised Learning algorithms** − On the other hand, it also has all the popular unsupervised learning algorithms from clustering, factor analysis, PCA (Principal Component Analysis) to unsupervised neural networks.

**Clustering** − This model is used for grouping unlabeled data.

**Cross Validation** − It is used to check the accuracy of supervised models on unseen data.

**Dimensionality Reduction** − It is used for reducing the number of attributes in data which can be further used for summarisation, visualisation and feature selection.

**Ensemble methods** − As name suggest, it is used for combining the predictions of multiple supervised models.

**Feature extraction** − It is used to extract the features from data to define the attributes in image and text data.

**Feature selection** − It is used to identify useful attributes to create supervised models.

**Open Source** – It is open source library and also commercially usable under BSD license.

# Scikit Learn - Modelling Process

## Dataset Loading

A collection of data is called dataset. It is having the following two components –

**Features** – The variables of data are called its features. They are also known as predictors, inputs or attributes.

> **Feature matrix** – It is the collection of features, in case there are more than one.
>
> **Feature Names** – It is the list of all the names of the features.

**Response** – It is the output variable that basically depends upon the feature variables. They are also known as target, label or output.

> **Response Vector** – It is used to represent response column. Generally, we have just one response column.
>
> **Target Names** – It represent the possible values taken by a response vector.

Scikit-learn have few example datasets like **iris** and **digits** for classification and the **Boston house prices** for regression.

### Example

Following is an example to load **iris** dataset –

```
from sklearn.datasets import load_iris
iris = load_iris()
X = iris.data
y = iris.target
feature_names = iris.feature_names
target_names = iris.target_namesprint(‘Feature names:‘,
feature_names)print(‘Target names:‘, target_names)print(‘\nFirst 10
rows of X:\n‘, X[:10])
```

## Output

```
Feature names: ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width
(cm)']
Target names: ['setosa' 'versicolor' 'virginica']
First 10 rows of X:
[
    [5.1 3.5 1.4 0.2]
    [4.9 3. 1.4 0.2]
    [4.7 3.2 1.3 0.2]
    [4.6 3.1 1.5 0.2]
    [5. 3.6 1.4 0.2]
    [5.4 3.9 1.7 0.4]
    [4.6 3.4 1.4 0.3]
    [5. 3.4 1.5 0.2]
    [4.4 2.9 1.4 0.2]
    [4.9 3.1 1.5 0.1]
]
```

# Splitting the dataset

To check the accuracy of our model, we can split the dataset into two pieces-**a training set** and **a testing set**. Use the training set to train the model and testing set to test the model. After that, we can evaluate how well our model did.

## Example

The following example will split the data into 70:30 ratio, i.e. 70% data will be used as training data and 30% will be used as testing data. The dataset is iris dataset as in above example.

```
from sklearn.datasets import load_iris
iris = load_iris()

X = iris.data
y = iris.target
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
   X, y, test_size = 0.3, random_state = 1)
print(X_train.shape)print(X_test.shape)
print(y_train.shape)print(y_test.shape)
```

## Output

```
(105, 4)
(45, 4)
(105,)
(45,)
```

As seen in the example above, it uses **train_test_split()** function of scikit-learn to split the dataset. This function has the following arguments –

**X, y** – Here, **X** is the **feature matrix** and y is the **response vector**, which need to be split.

**test_size** – This represents the ratio of test data to the total given data. As in the above example, we are setting **test_data** = **0.3** for 150 rows of X. It will produce test data of 150*0.3 = 45 rows.

**random_size** – It is used to guarantee that the split will always be the same. This is useful in the situations where you want reproducible results.

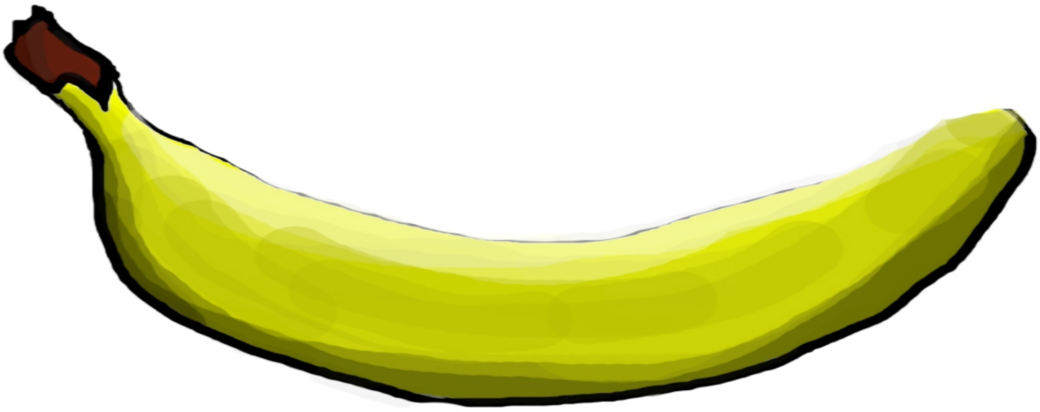
# Supervised and Unsupervised learning

Supervised learning: Supervised learning, as the name indicates, has the presence of a supervisor as a teacher. Basically supervised learning is when we teach or train the machine using data that is well-labelled. Which means some data is already tagged with the correct answer. After that, the machine is provided with a new set of examples(data) so that the supervised learning algorithm analyses the training data(set of training examples) and produces a correct outcome from labeled data.

**For instance**, suppose you are given a basket filled with different kinds of fruits. Now the first step is to train the machine with all the different fruits one by one like this:

- ➢ If the shape of the object is rounded and has a depression at the top, is red in color, then it will be labeled as **–Apple**.
- ➢ If the shape of the object is a long curving cylinder having Green-Yellow color, then it will be labeled as **–Banana**.

Now suppose after training the data, you have given a new separate fruit, say Banana from the basket, and asked to identify it.



Since the machine has already learned the things from previous data and this time has to use it wisely. It will first classify the fruit with its shape and color and would confirm the fruit name as BANANA and put it in the Banana category. Thus the machine learns the things from training data(basket containing fruits) and then applies the knowledge to test data(new fruit).

Supervised learning is classified into two categories of algorithms:

**Classification**: A classification problem is when the output variable is a category, such as “Red” or “blue” , “disease” or “no disease”.

**Regression**: A regression problem is when the output variable is a real value, such as “dollars” or “weight”.

Supervised learning deals with or learns with “labeled” data. This implies that some data is already tagged with the correct answer.

**Types:-**

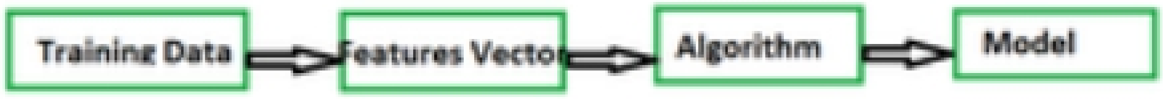
1. Regression
2. Logistic Regression
3. Classification
4. Naive Bayes Classifiers
5. K-NN (k nearest neighbors)
6. Decision Trees
7. Support Vector Machine

**Advantages:-**

- Supervised learning allows collecting data and produces data output from previous experiences.
- Helps to optimize performance criteria with the help of experience.
- Supervised machine learning helps to solve various types of real-world computation problems.
- It performs classification and regression tasks.
- It allows estimating or mapping the result to a new sample.
- We have complete control over choosing the number of classes we want in the training data.

**Disadvantages:-**

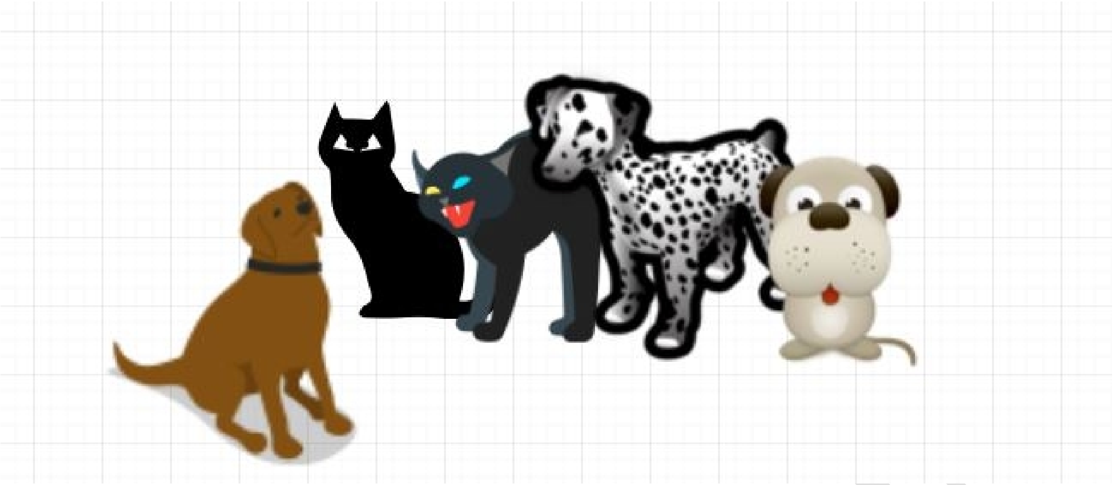
- Classifying big data can be challenging.
- Training for supervised learning needs a lot of computation time. So, it requires a lot of time.
- Supervised learning cannot handle all complex tasks in Machine Learning.
- Computation time is vast for supervised learning.
- It requires a labelled data set.
- It requires a training process.



## <u>Unsupervised learning</u>

Unsupervised learning is the training of a machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of the machine is to group unsorted information according to similarities, patterns, and differences without any prior training of data.

Unlike supervised learning, no teacher is provided that means no training will be given to the machine. Therefore the machine is restricted to find the hidden structure in unlabeled data by itself. **For instance**, suppose it is given an image having both dogs and cats which it has never seen.



Thus the machine has no idea about the features of dogs and cats so we can’t categorize it as ‘dogs and cats ‘. But it can categorize them according to their similarities, patterns, and differences, i.e., we can easily categorize the above picture into two parts. The first may contain all pics having **dogs** in them and the second part may contain all pics having **cats** in them. Here you didn’t learn anything before, which means no training data or examples.

It allows the model to work on its own to discover patterns and information that was previously undetected. It mainly deals with unlabelled data.

Unsupervised learning is classified into two categories of algorithms:

**Clustering**: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.

**Association**: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

Types of Unsupervised Learning:-

**Clustering**

1. Exclusive (partitioning)
2. Agglomerative

3. Overlapping
4. Probabilistic

**Clustering Types:-**

1. Hierarchical clustering
2. K-means clustering
3. Principal Component Analysis
4. Singular Value Decomposition
5. Independent Component Analysis

**Supervised vs. Unsupervised Machine Learning:**

| **Parameters** | **Supervised machine learning** | **Unsupervised machine learning** |
|---|---|---|
| Input Data | Algorithms are trained using labeled data. | Algorithms are used against data that is not labeled |
| Computational Complexity | Simpler method | Computationally complex |
| Accuracy | Highly accurate | Less accurate |
| No. of classes | No. of classes is known | No. of classes is not known |
| Data Analysis | Uses offline analysis | Uses real-time analysis of data |
| Algorithms used | Linear and Logistics regression, Random forest,<br>Support Vector Machine, Neural Network, etc. | K-Means clustering, Hierarchical clustering,<br>Apriori algorithm, etc. |
| Output | Desired output is given. | Desired output is not given. |
| Training data | Use training data to infer model. | No training data is used. |

| | | |
|---|---|---|
| Complex model | It is not possible to learn larger and more complex models than with supervised learning. | It is possible to learn larger and more complex models with unsupervised learning. |
| Model | We can test our model. | We can not test our model. |
| Called as | Supervised learning is also called classification. | Unsupervised learning is also called clustering. |
| Example | Example: Optical character recognition. | Example: Find a face in an image. |

**Advantages of unsupervised learning**:

- It does not require training data to be labeled.
- Dimensionality reduction can be easily accomplished using unsupervised learning.
- Capable of finding previously unknown patterns in data.
- **Flexibility**: Unsupervised learning is flexible in that it can be applied to a wide variety of problems, including clustering, anomaly detection, and association rule mining.
- **Exploration**: Unsupervised learning allows for the exploration of data and the discovery of novel and potentially useful patterns that may not be apparent from the outset.
- **Low cost**: Unsupervised learning is often less expensive than supervised learning because it doesn't require labeled data, which can be time-consuming and costly to obtain.

**Disadvantages of unsupervised learning** :

- Difficult to measure accuracy or effectiveness due to lack of predefined answers during training.

- The results often have lesser accuracy.
- The user needs to spend time interpreting and label the classes which follow that classification.
- **Lack of guidance**: Unsupervised learning lacks the guidance and feedback provided by labeled data, which can make it difficult to know whether the discovered patterns are relevant or useful.
- **Sensitivity to data quality**: Unsupervised learning can be sensitive to data quality, including missing values, outliers, and noisy data.
- **Scalability**: Unsupervised learning can be computationally expensive, particularly for large datasets or complex algorithms, which can limit its scalability.

# Regression and Classification

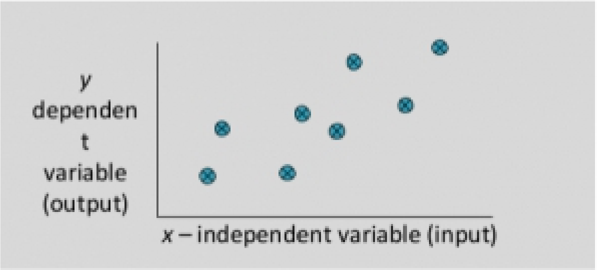
Supervised Machine Learning: The majority of practical machine learning uses supervised learning. Supervised learning is where you have input variables (x) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output **Y = f(X)** . The goal is to approximate the mapping function so well that when you have new input data (x) that you can predict the output variables (Y) for that data.

Techniques of Supervised Machine Learning algorithms include **linear** and **logistic regression**, **multi-class classification**, **Decision Trees** and **support vector machines**.

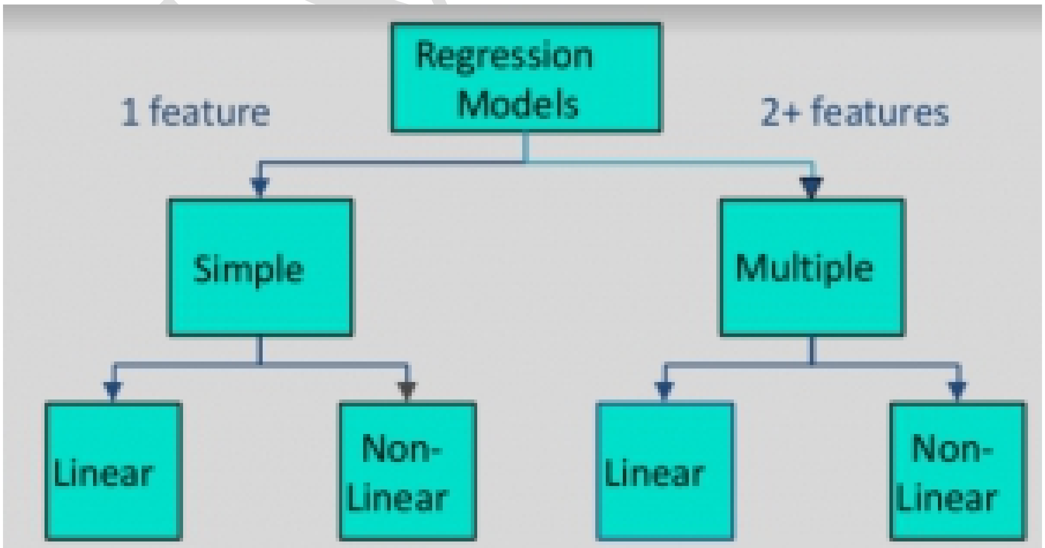
Supervised learning requires that the data used to train the algorithm is already labelled with correct answers. For example, a classification algorithm will learn to identify animals after being trained on a dataset of images that are properly labelled with the species of the animal and some identifying characteristics.

Supervised learning problems can be further grouped into **Regression** and **Classification** problems. Both problems have a goal of the construction of a succinct model that can predict the value of the dependent attribute from the attribute variables. The difference between the two tasks is the fact that the dependent attribute is numerical for regression and categorical for classification.

A regression problem is when the output variable is a real or continuous value, such as “salary” or “weight”. Many different models can be used, the simplest is the linear regression. It tries to fit data with the best hyper-plane which goes through the points.



**Types of Regression Models:**



**For Examples:**

**Which of the following is a regression task?**

Predicting age of a person
Predicting nationality of a person
Predicting whether stock price of a company will increase tomorrow
Predicting whether a document is related to sighting of UFOs?

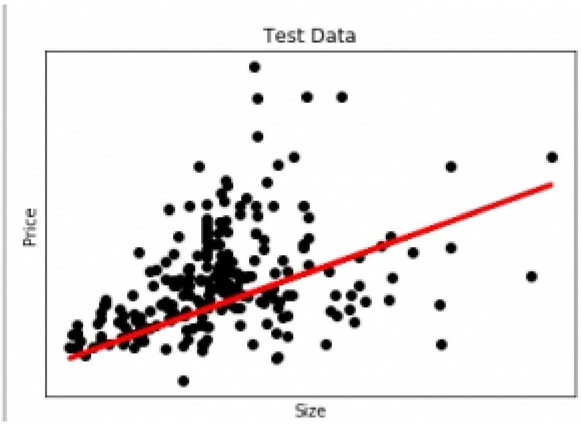
**Solution :** Predicting age of a person (because it is a real value, predicting nationality is categorical, whether stock price will increase is discrete-yes/no answer, predicting whether a document is related to UFO is again discrete- a yes/no answer).

Let’s take an example of linear regression. We have a Housing data set and we want to predict the price of the house. Following is the python code for it.

```
# Python code to illustrate
# regression using data set
import matplotlib
matplotlib.use(’GTKAgg’)

import matplotlib.pyplot as plt
import numpy as np
from sklearn import datasets, linear_model
import pandas as pd

# Load CSV and columns
df = pd.read_csv(‘Housing.csv‘ )

Y = df[’price’]
X = df[’lotsize’]

X=X.values.reshape(len(X),1)
Y=Y.values.reshape(len(Y),1)

# Split the data into training/testing sets
```

```
X_train = X[:-250]
X_test = X[-250:]

# Split the targets into training/testing sets
Y_train = Y[:-250]
Y_test = Y[-250:]

# Plot outputs
plt.scatter(X_test, Y_test,  color=’black’)
plt.title( ’Test Data’)
plt.xlabel( ’Size’)
plt.ylabel( ’Price’)
plt.xticks(())
plt.yticks(())


# Create linear regression object
regr = linear_model.LinearRegression()

# Train the model using the training sets
regr.fit(X_train, Y_train)

# Plot outputs
plt.plot(X_test, regr.predict(X_test), color=’red’,linewidth=3)
plt.show()
```

The output of the above code will be:



Here in this graph, we plot the test data. The red line indicates the best fit line for predicting the price. To make an individual prediction using the linear regression model:

```
print( str(round(regr.predict(5000))) )
```

## Classification

A classification problem is when the output variable is a category, such as “red” or “blue” or “disease” and “no disease”. A classification model attempts to draw some conclusion from observed values. Given one or more inputs a classification model will try to predict the value of one or more outcomes.

For example, when filtering emails “spam” or “not spam”, when looking at transaction data, “fraudulent”, or “authorized”. In short Classification either predicts categorical class labels or classifies data (construct a model) based on the training set and the values (class labels) in classifying attributes and uses it in classifying new data. There are a number of classification models. Classification models include logistic regression, decision tree, random forest, gradient-boosted tree, multilayer perceptron, one-vs-rest, and Naive Bayes.

**For example :**

**Which of the following is/are classification problem(s)?**

- Predicting the gender of a person by his/her handwriting style
- Predicting house price based on area
- Predicting whether monsoon will be normal next year
- Predict the number of copies a music album will be sold next month

Solution : Predicting the gender of a person Predicting whether monsoon will be normal next year. The other two are regression.

## Dataset Description

Title: Iris Plants Database

Attribute Information:

1. sepal length in cm
2. sepal width in cm
3. petal length in cm
4. petal width in cm
5. class:

   -- Iris Setosa

   -- Iris Versicolour

   -- Iris Virginica

Missing Attribute Values: None

Class Distribution: 33.3% for each of 3 classes

```
# Python code to illustrate
# classification using data set
#Importing the required library
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report

#Importing the dataset
dataset = pd.read_csv(
            'https://archive.ics.uci.edu/ml/machine-learning-'+
            ’databases/iris/iris.data’,sep= ’,’, header= None)
data = dataset.iloc[:, :]

#checking for null values
print(‘Sum of NULL values in each column. ‘)
print(data.isnull().sum())

#separating the predicting column from the whole dataset
X = data.iloc[:, :-1].values
y = dataset.iloc[:, 4].values

#Encoding the predicting variable
labelencoder_y = LabelEncoder()
y = labelencoder_y.fit_transform(y)

#Splitting the data into test and train dataset
X_train, X_test, y_train, y_test = train_test_split(
                    X, y, test_size = 0.3, random_state = 0)

#Using the random forest classifier for the prediction
classifier=RandomForestClassifier()
classifier=classifier.fit(X_train,y_train)
predicted=classifier.predict(X_test)

#printing the results
print ('Confusion Matrix :')
print(confusion_matrix(y_test, predicted))
print ('Accuracy Score :',accuracy_score(y_test, predicted))
print ('Report : ')
print (classification_report(y_test, predicted))
```

Output:

```
Sum of NULL values in each column.

            0       0

            1       0

            2       0
```

```
        3     0

        4     0


Confusion Matrix :

                    [[16   0   0]

                     [ 0 17   1]

                     [ 0   0 11]]


Accuracy Score : 97.7


Report :

             precision     recall   f1-score    support

      0         1.00        1.00       1.00          16

      1         1.00        0.94       0.97          18

      2         0.92        1.00       0.96          11

avg/total     0.98       0.98       0.98          45
```