## A Project On

**Classification of Iris Flower Species using Machine Learning Techniques**

**For**

**Internship Program**

**on Python**

**At**

**CDAC(Bhubaneswar)**

**Submitted By:-**

**Amar Singh**

**Reshma Rani Sahoo**

**Abinash Mohapatra**

**Guided By:-**

**Mr. Ranjan Kumar Das**

****

**ABSTARACT**

**Machine learning** is an application of artificial intelligence (AI) that provides system the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves.

The aim is to predict the algorithm which gives the best result in our project, i.e, Classification of Iris datasets using Python.

The datasets for the Iris flower species are collected and loaded from UCI Repository. The datasets consist of 50 samples from each of three species (targets) of Iris (Iris setosa, Iris virginica and Iris versicolor). Four features (attributes) are measured from each sample: The length and width of sepals and length and width of petals.

The datasets are split for training and testing purpose. We train our algorithms (models) using this datasets and then use this training to predict species of a Iris flower with given measurements where we will use the packages like numpy, pandas, scipy, sklearn and matplotlib.

The models are:

1. Logistic Regression
2. Support Vector Machine(SVM)
3. Gaussian Naïve
4. Decision Tree Classification
5. Kth Nearest Neighbors

Finally, we will predict the best model on the basis of test score and accuracy.

CONTENTS Page No

1. INTRODUCTION 4 - 9
2. ABOUT THE PROJECT 10
3. REQUIREMENTS 11 - 14
4. ALGORITHMS APPLIED 15 - 21
5. STEPS FOLLOWED 22 - 27
6. PROJECT CODE 28 - 29
7. CONCLUSION 30

**INTRODUCTION:-**

Definition:

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves.

Tom M. Mitchell provided a widely quoted, more formal definition of the algorithms studied in the machine learning field: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E."

Why Machine learning?

There is one crucial reason why data scientists need machine learning, and that is: ‘High-value predictions that can guide better decisions and smart actions in real time without human intervention’.

Machine learning as a technology helps analyze large chunks of data, easing the tasks of data scientists in an automated process and is gaining a lot of prominence and recognition. Machine learning has changed the way data extraction and interpretation works by involving automatic sets of generic methods that have replaced the traditional statistical techniques.

Machine learning methods:

Supervised learning:

Supervised learning algorithms are trained using labeled examples, such as an input where the desired output is known. For example, a piece of equipment could have data points labeled either “F” (failed) or “R” (runs). The learning algorithm receives a set of inputs along with the corresponding correct outputs, and the algorithm learns by comparing its actual output with correct outputs to find errors. It then modifies the model accordingly. Through methods like classification, regression, prediction and gradient boosting, supervised learning uses patterns to predict the values of the label on additional unlabeled data. Supervised learning is commonly used in applications where historical data predicts likely future events.

Unsupervised learning:

Unsupervised learning is used against data that has no historical labels. The system is not told the "right answer." The algorithm must figure out what is being shown. The goal is to explore the data and find some structure within. Unsupervised learning works well on transactional data. For example, it can identify segments of customers with similar attributes who can then be treated similarly in marketing campaigns. Or it can find the main attributes that separate customer segments from each other. Popular techniques include self-organizing maps, nearest-neighbor mapping, k-means clustering and singular value decomposition. These algorithms are also used to segment text topics, recommend items and identify data outliers.

Semi supervised learning:

Semi-supervised learning is used for the same applications as supervised learning. But it uses both labeled and unlabeled data for training – typically a small amount of labeled data with a large amount of unlabeled data (because unlabeled data is less expensive and takes less effort to acquire). This type of learning can be used with methods such as classification, regression and prediction. Semi-supervised learning is useful when the cost associated with labeling is too high to allow for a fully labeled training process. Early examples of this include identifying a person's face on a web cam.

Reinforcement learning:

Reinforcement learning is often used for robotics, gaming and navigation. With reinforcement learning, the algorithm discovers through trial and error which actions yield the greatest rewards. This type of learning has three primary components: the agent (the learner or decision maker), the environment (everything the agent interacts with) and actions (what the agent can do). The objective is for the agent to choose actions that maximize the expected reward over a given amount of time. The agent will reach the goal much faster by following a good policy. So the goal in reinforcement learning is to learn the best policy.

Machine learning application:

▪ In classification, inputs are divided into two or more classes, and the learner must produce a model that assigns unseen inputs to one or more (multi-label classification) of these classes. This is typically tackled in a supervised manner. Spam filtering is an example of classification, where the inputs are email (or other) messages and the classes are "spam" and "not spam".

▪ In regression, also a supervised problem, the outputs are continuous rather than discrete.

▪ In clustering, a set of inputs is to be divided into groups. Unlike in classification, the groups are not known beforehand, making this typically an unsupervised task.

▪ Density estimation finds the distribution of inputs in some space.

▪ Dimensionality reduction simplifies inputs by mapping them into a lower-dimensional space. Topic modeling is a related problem, where a program is given a list of human language documents and is tasked with finding out which documents cover similar topics.

Application in daily life:

1. Virtual Personal Assistants

Ex- Smart Speakers: Amazon Echo and Google Home

Smart phones: Samsung Bixby on Samsung S8

Mobile Apps: Google Allo

2. Predictions while Commuting

Online Transportation Networks: When booking a cab, the app estimates the price of the ride.

3. Videos Surveillance

The video surveillance system track unusual behavior of people like standing motionless for a long time, stumbling, or napping on benches etc. And when such activities are reported and counted to be true, they help to improve the surveillance services. This happens with machine learning doing its job at the backend.

4. Online Fraud Detection

Machine learning is proving its potential to make cyberspace a secure place and tracking monetary frauds online is one of its examples. For example: PayPal is using ML for protection against money laundering.

5. Email spam and malware filtering

6. Social Media Services like face recognition, people you may know in Facebook.

7. Product Recommendations by online shopping sites.

8. Search Engine Result Refining

Google and other search engines use machine learning to improve the search results.

9. Health care industries

Machine learning is a fast-growing trend in the health care industry, the advent of wearable devices and sensors that can use data to assess a patient's health in real time.

Top companies that use machine learning:

1. Amazon.

2. Netflix.

3. Google.

4. Salesforce.

5. IBM.

6. Microsoft.

7. Intel.

8. Apple.

**About the Project:-**

**PROJECT ON MACHINE LEARNING**

**Name: -** Classification of Iris Flower Species

**Aim: -** To predict the algorithm which gives the best result.

**REQUIREMENTS:-**

Tools required for the project:

**Python Packages:-**

Python is a popular and powerful interpreted language. Python is a complete language and platform that is used for both research and development and developing production systems.

Here, we have used Python 3.6 version, which provides different packages, that contains a lot of modules and libraries to choose from, providing multiple ways to do each task.

The following are the most useful packages for machine learning in Python:

**NumPy:**

NumPy is a general-purpose array-processing package. It provides a

high-performance multidimensional array object and tools for working

with these arrays.

It is the fundamental package for scientific computing with Python. It

contains various features including these important ones:

* A powerful N-dimensional array object
* Sophisticated (broadcasting) functions
* Tools for integrating C/C++ and Fortran code
* Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data.  
Arbitrary data-types can be defined using Numpy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

Travis Oliphant created NumPy by incorporating features by incorporating features of the competing Numarray into Numeric, with extensive modification.

**SciPy:**

SciPy is a free and open-source library Python library used for scientific computing and technical computing. SciPy contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering. SciPy builds on the NumPy array object and is part of the NumPy stack which includes tools like Matplotlib, pandas and SymPy, and an expanding set of scientific computing libraries. This NumPy stack has similar users to other applications such as MATLAB, GNU Octave, and Scilab. The NumPy stack is also sometimes referred to as the SciPy stack. The basic data structure used by SciPy is a multidimensional array provided by the NumPy module.

**Pandas:**

Pandasis a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The library is highly optimized for performance.

The library contains different features, such as:

1. Data frame object for data manipulation with integrated indexing.
2. Data set joining and merging.
3. Reshaping and pivoting data set, and many more.

Pandas was developed by Wes McKinney and started working on pandas in 2008 while at AQR Capital Management out of the need for a high performance, flexible tool to perform quantitative analysis on financial data. Before leaving AQR he was able to convince management to allow him to open source the library.

**Scikit-learn:**

Scikit learn is free software machine learning library in Python.  It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, *k*-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy. Scikit-learn is largely written in Python, with some core algorithms written in Cython to achieve performance. Support vector machines are implemented by a Cython wrapper around LIBSVM; logistic regression and linear support vector machines by a similar wrapper around LIBLINEAR.

It was developed by David Cornapeau as Google Summer of code project in 2007.

**matplotlib:**

**Matplotlib** is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of matplotlib.

It was developed by John D Hunter, as an active development community, under a BSD style license.

Pyplot is a matplotlib module which provides a MATLAB-like interface. Matplotlib is designed to be as usable as MATLAB, with the ability to use Python, and the advantage of being free and open-source.

It plots different graph such as line plot, histogram, scatter plot, 3D plot, etc.

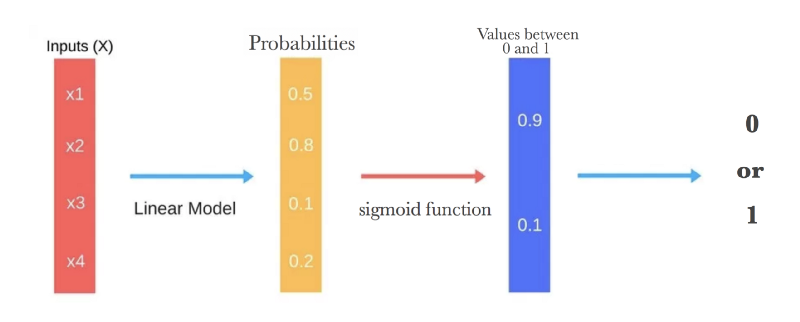
**ALGORITHMS APPLIED**:

Classification is one of the most important task in machine learning. For study purpose various algorithms are available for classification purpose, but in this paper we introduce five algorithms from them. They are:

1. Logistic Regression
2. Decision Tree Classification
3. Support Vector Machine (SVM)
4. Kth Nearest Neighbors
5. Gaussian Naïve

**Logistic Regression: Logistic Regression is one of the most used Machine Learning algorithms for binary classification. It is a simple Algorithm that you can use as a performance baseline, it is easy to implement and it will do well enough in many tasks.** Like many other machine learning techniques, it is borrowed from the field of statistics and despite its name, it is not an algorithm for regression problems, where you want to predict a continuous outcome. Instead, Logistic Regression is the go-to method for binary classification. It gives you a discrete binary outcome between 0 and 1. To say it in simpler words, it’s outcome is either one thing or another. A simple example of a Logistic Regression problem would be an algorithm used for cancer detection that takes screening picture as an input and should tell if a patient has cancer (1) or not (0).

**Working:** Logistic Regression measures the relationship between the dependent variable (our label, what we want to predict) and the one or more independent variables (our features), by estimating probabilities using it’s underlying logistic function. These probabilities must then be transformed into binary values in order to actually make a prediction. This is the task of the logistic function, also called the sigmoid function. The picture below illustrates the steps that logistic regression goes through to give you your desired output.

****

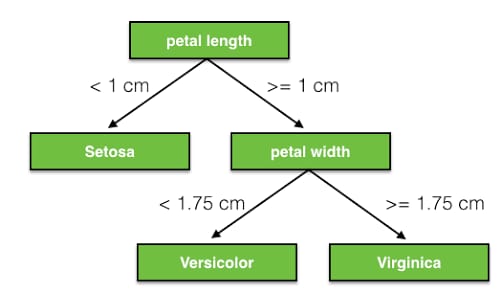
The LogisticRegression method is imported in Python as follows:

from sklearn.linear\_model import LogisticRegression

**Decision Tree Classification:** Decision tree learning uses a decision tree (as a predictive model) to go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves). It is one of the predictive modelling approaches used in statistics, data mining and machine learning. Tree models where the target variable can take a discrete set of values are called **classification trees**; in these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. Decision trees where the target variable can take continuous values (typically real numbers) are called **regression trees**. In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making.

**Working:** In this procedure all the features are considered and different split points are tried and tested using a cost function. The split with the best cost (or lowest cost) is selected.

The picture below illustrates the working of decision tree classifier:

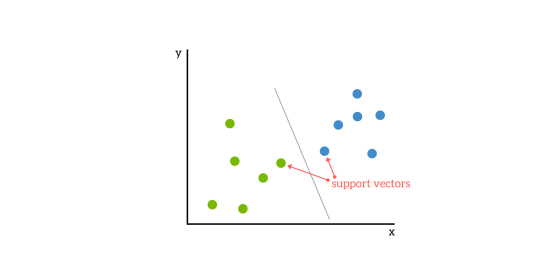


The DecisionTreeClassifier method is imported in Python as follows:

from sklearn.tree import DecisionTreeClassifier

**Support Vector Machine (SVM):** In machine learning, **support vector machines** (**SVMs**, also **support vector networks**) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

**Working:** SVMs are based on the idea of finding a hyperplane that best divides a dataset into two classes, as shown in the image below.



Support vectors are the data points nearest to the hyperplane, the points of a data set that, if removed, would alter the position of the dividing hyperplane.

The SVC method can be imported as follows:

from sklearn.svm import svc

**Kth Nearest Neighbors:**  In pattern recognition, the ***k*-nearest neighbors algorithm** (***k*-NN**) is a non-parametric method used for classification and regression. In both cases, the input consists of the *k* closest training examples in the feature space. The output depends on whether *k*-NN is used for classification or regression:

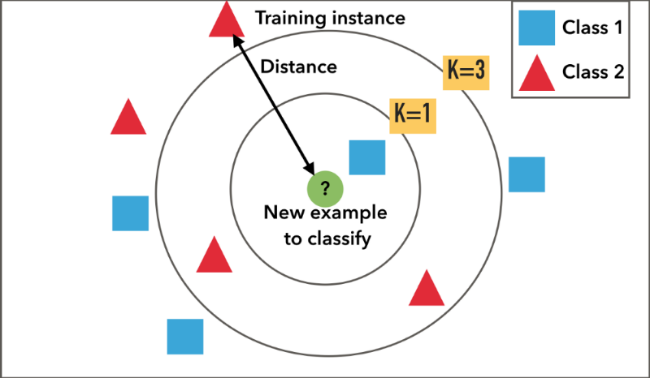
* In *k-NN classification*, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors (*k* is a positive integer, typically small). If *k* = 1, then the object is simply assigned to the class of that single nearest neighbor.
* In *k-NN regression*, the output is the property value for the object. This value is the average of the values of its *k* nearest neighbors.

*k*-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. The *k*-NN algorithm is among the simplest of all machine learning algorithms.

Both for classification and regression, a useful technique can be to assign weight to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones.

**Working:** The algorithmuse a database in which the data points are separated into several classes to predict the classification of a new sample point.

Example: The test sample (green circle) should be classified either to the first class of blue squares or to the second class of red triangles. If k = 3 (solid line circle) it is assigned to the second class because there are 2 triangles and only 1 square inside the inner circle. If k = 5 (dashed line circle) it is assigned to the first class (3 squares vs. 2 triangles inside the outer circle).



The KNeighborsClassifier method is imported as follows:

From sklearn.neighbors import KNeighborsClassifier

**Gaussian Naïve:** In machine learning, **naive Bayes classifiers** are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naive) independence assumptions between the features.

Naive Bayes has been studied extensively since the 1950s. It was introduced under a different name into the text retrieval community in the early 1960s, and remains a popular (baseline) method for text categorization, the problem of judging documents as belonging to one category or the other (such as spam or legitimate, sports or politics, etc.) with word frequencies as the features. With appropriate pre-processing, it is competitive in this domain with more advanced methods including support vector machines. It also finds application in automatic medical diagnosis.

Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.

**Working:** Bayes’ Theorem provides a way that we can calculate the probability of a hypothesis given our prior knowledge.

Bayes’ Theorem is stated as:

P(h|d) = (P(d|h) \* P(h)) / P(d)

Where

* **P(h|d)** is the probability of hypothesis h given the data d. This is called the posterior probability.
* **P(d|h)** is the probability of data d given that the hypothesis h was true.
* **P(h)** is the probability of hypothesis h being true (regardless of the data). This is called the prior probability of h.
* **P(d)** is the probability of the data (regardless of the hypothesis).

This extension of naive Bayes is called Gaussian Naive Bayes. Other functions can be used to estimate the distribution of the data, but the Gaussian (or Normal distribution) is the easiest to work with because you only need to estimate the mean and the standard deviation from your training data.

The GaussianNB method is imported as follows:

from sklearn.naive\_bayes import GaussianNB

**Steps Followed:-**

**Problem Definition:**

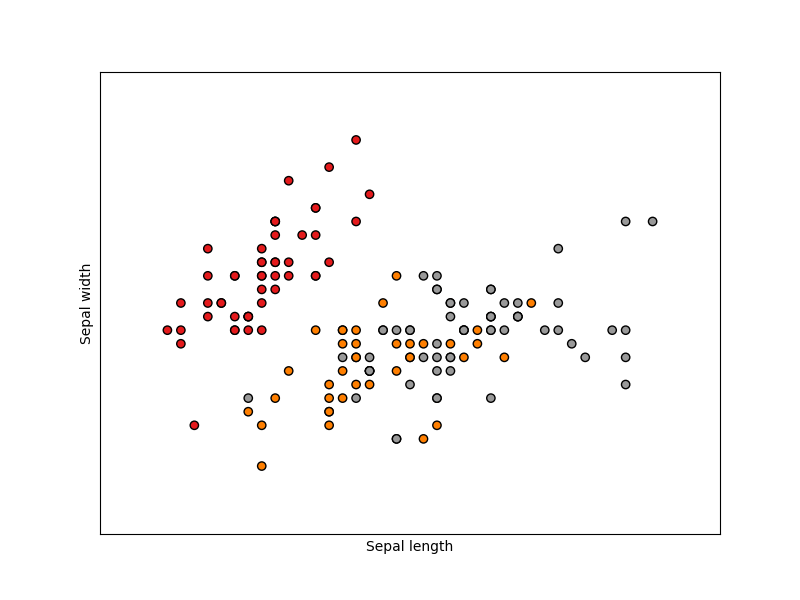
To predict the best algorithm (model) this gives the best result on the basis of test score and accuracy.

**Collect and Load Data:**

The datasets for the Iris flower species are collected and loaded from UCI Repository, using **scikit-learn** package, as follows:

from sklearn.datasets import load\_iris

The datasets consist of 50 samples from each of three species (targets) of Iris (Iris setosa, Iris virginica and Iris versicolor). Four features (attributes) are measured from each sample: The length and width of sepals and length and width of petals. It is stored in a **150x4 numpy.ndarray**. The rows being the samples and the columns being: Sepal Length, Sepal Width, Petal Length and Petal Width, as shown in the following figure:



**Visualize Data:**

The iris dataset is a classic and very easy multi-class classification dataset.

|  |  |
| --- | --- |
| Targets (classes) | 3 |
| Samples per class | 50 |
| Samples total | 150 |
| Dimensionality(attributes) | 4 |
| Features | real, positive |

We can get the target names as follows:

1.PNG

We can get the attributes of different species as follows:

>>>list(data.feature\_name)

**Split Data for Training and Testing:**

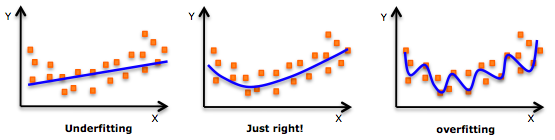
The datasets are split for training and testing purpose. Training and testing is performed by importing **train\_test\_split** method, as follows:

*from sklearn.model\_selection import train\_test\_split*

We train our algorithms (models) using this datasets and then use this training to predict species of a Iris flower with given measurements. 70% of datasets is for training purpose and the remaining 30% is for testing purpose.

**Overfitting:** Overfitting means that model we trained has trained “too well” and is now, well, fit too closely to the training dataset. This usually happens when the model is too complex (i.e. too many features/variables compared to the number of observations). This model will be very accurate on the training data but will probably be very not accurate on untrained or new data. The trained datasets is stored in a numpy array. Basically, the model learns the training data instead of the actual relationships between the variables in the data.

**Underfitting:** This usually happens when the model is too complex (i.e. too many features/variables compared to the number of observations). This model will be very accurate on the training data but will probably be very not accurate on untrained or new data.



An example of overfitting, underfitting and a model that’s “just right!”

**Train the data by using different algorithms:**

The datasets collected and loaded from UCI Repository is spitted, i.e, 70% for training purpose and 30% for testing purpose. 70% of datasets is trained by using different algorithms (models), such as Logistic Regression, Decision Tree Classification, Support Vector Machine (SVM), Kth Nearest Neighbors and Gaussian Naïve, which are described briefly in the above section.

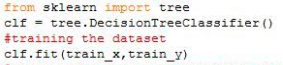
Logistic Regression:

In logistic regression, the datasets are trained as follows:

LR.PNG

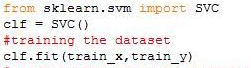
Decision Tree Classification:

In decision tree classification, the datasets are trained as follows:

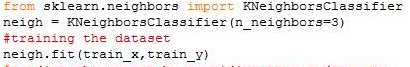


SVM:

In SVM, the datasets are trained as follows:



KNN: In KNN, the datasets are trained as follows:



Gaussian Naïve: In Gaussian Naïve, the datasets are trained as follows:

GN.PNG

**Test Data:**

The test data is stored in a numpy array and we want to predict the species of the new flower. We do this using the **predict** method which takes this array as input and spits out predicted target value as output.

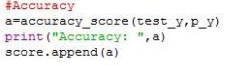
This is shown as follows:

predict.PNG

**Accuracy:**

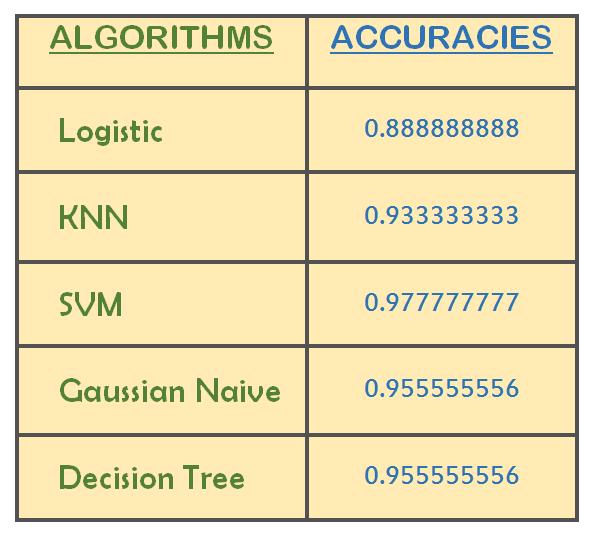
Finally we find the accuracy by using the **accuracy\_score** method.

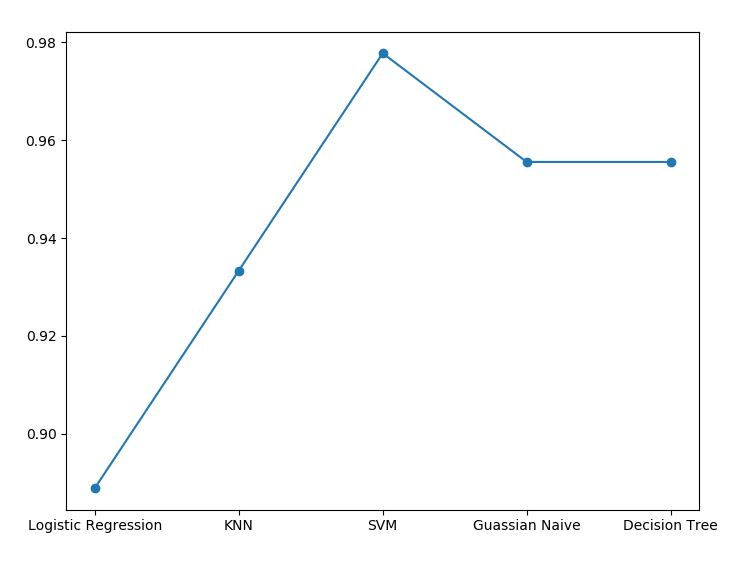
This is shown in the following figure:



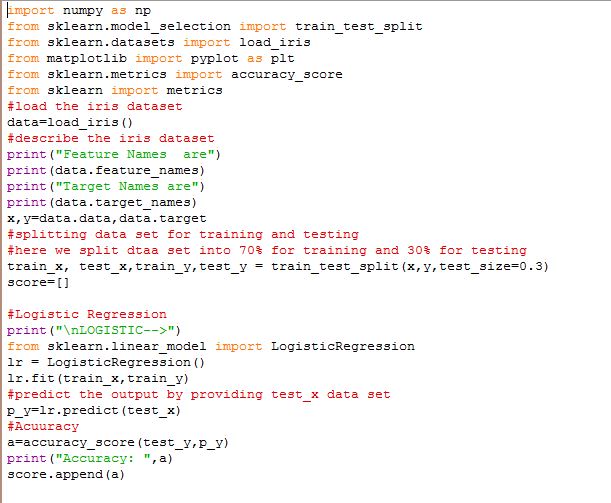
**Predict the best model:**

According to the following graph and table, we found that **SVM** model is the best classification model with an accuracy of **0.9777777.**

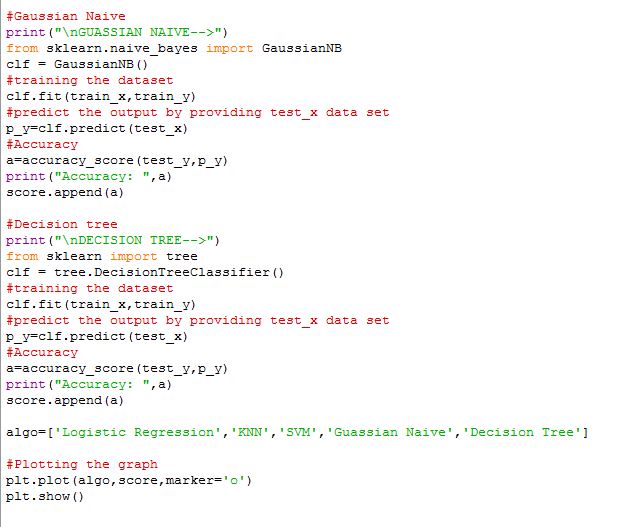




**PROJECT CODES:**

****

****

****

**CONCLUSION:**

Finally, we concluded that **SVM** model is the best model for the classification of Iris flower species with an accuracy of **0.9777777.**

**References:**

The datasets for the Iris flower species are collected and loaded from UCI Repository, available on the site [uci.edu](Report.docx)

Other websites:

* + - [wikipedia.org](Report.docx)
    - [stackoverflow.com](Report.docx)
    - [machinelearningmastery.com](Report.docx)
    - [scikit-learn.org](Report.docx)
    - [matplotlib.org](Report.docx)
    - [medium.com](Report.docx)
    - [kdnuggets.com](Report.docx)
    - [dataspirant.com](Report.docx)
    - [kaggle.com](Report.docx)