GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

IRIS FLOWER CLASSIFICATION

A Project Report of Capstone Project - 2

Submitted by

JAYESH BANSAL

1613101309

in partial fulfilment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Under the Supervision of

Mr. DINESH KUMAR BAGHEL, B.Tech, M.Tech

Asst. Professor

APRIL / MAY- 2020

SCHOOL OF COMPUTING AND SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report "IRIS FLOWER CLASSIFICATION" is the bonafide work of "JAYESH BANSAL" who carried out the project work under my supervision.

SIGNATURE OF HEAD

Dr. MUNISH SHABARWAL,
PhD (Management), PhD (CS)
Professor & Dean,
School of Computing Science & Engineering

SIGNATURE OF SUPERVISOR

Mr. DINESH KUMAR BAGHEL,
B.Tech, M.Tech
Asst. Professor
School of Computing Science & Engineering

ABSTRACT

Classification is a supervised machine learning technique which is used to predict group membership for data instances. Neural networks are being introduced to simplify the problem of classification. This model focuses on Iris flower classification using Neural Network. For simplification of classification we will use scikit learn tool kit. This project mainly focuses on the classification of dataset using scikit learn. The problem concerns that the recognition of Iris flower species (setosa, versicolor and verginica) on the basis of the measurements of length and width of sepal and petal of the flower. We can generate classification model by using various machine learning algorithms through training the iris flower dataset and can choose the model with highest accuracy to predict the species of iris flower more precisely. Classification of Iris data set would be detecting patterns from examining sepal and petal size of the Iris flower and how the prediction was made from analyzing the pattern to form the class of Iris flower. By using this pattern and classification, in future upcoming years the unseen data can be predicted more precisely. Artificial neural networks have been successfully applied to problems in pattern classification, function approximations, optimization, and associative memories. The goal here is to model the probabilities of class membership, conditioned on the flower features. In this project we will train our model with data using machine learning to predict the species of iris flower by input of the unseen data using what it has learnt from the trained data.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1. Machine Learning

Machine learning is a process of feeding a machine enough data to train and predict a possible outcome using the algorithms. the more the processed or useful data is fed to the machine the more efficient the machine will become. When the data is complicated it learns the data and builds the prediction model. It is state that more the data, better the model, higher will be the accuracy. There are many ways for machine learning i.e. supervised learning, unsupervised learning and reinforcement learning.

1.2. Supervised Learning

In supervised learning machine learning model learns through the feature and labels of the object. Supervised learning uses labeled data to train the model here, the machine knew the features of the object and labels associated with those features or we can say that the supervised learning uses the set of data where the labels or the desired outcomes are already known. It is allowed to prediction about the unseen or future data.

1.3. Classification

Classification is one of the major data mining processes which maps data into predefined groups. It comes under supervised learning method as the classes are determined before examining the data. For applying all approaches to performing classification it is required to have some knowledge of the data. Usually, the knowledge of the data helps to find some unknown patterns. The aim of pattern classification is to building a function that provides output of two or more than two classes from the input feature.

The dataset for this project carried out from the UCI Machine Learning Repository. The Iris flower data set introduced by the British statistician and biologist Ronald Fisher that's why it

is also known by Fisher's Iris data set and it is a multivariate data set. The use of multiple measurements in taxonomic problems as an example of linear discriminant analysis.

The expectation from mining iris data set would be discovering patterns from examining sepal and petal size of the iris plant and how the prediction was made from analyzing the pattern to predict the class of iris plant. In upcoming years, using the classification and pattern recognition other flowers can be individually distinguish to each other. It is unmistakably expressed that the sort of relationship that being mined utilizing iris dataset would be a classification model.

1.4. HARDWARE REQUIREMENTS:

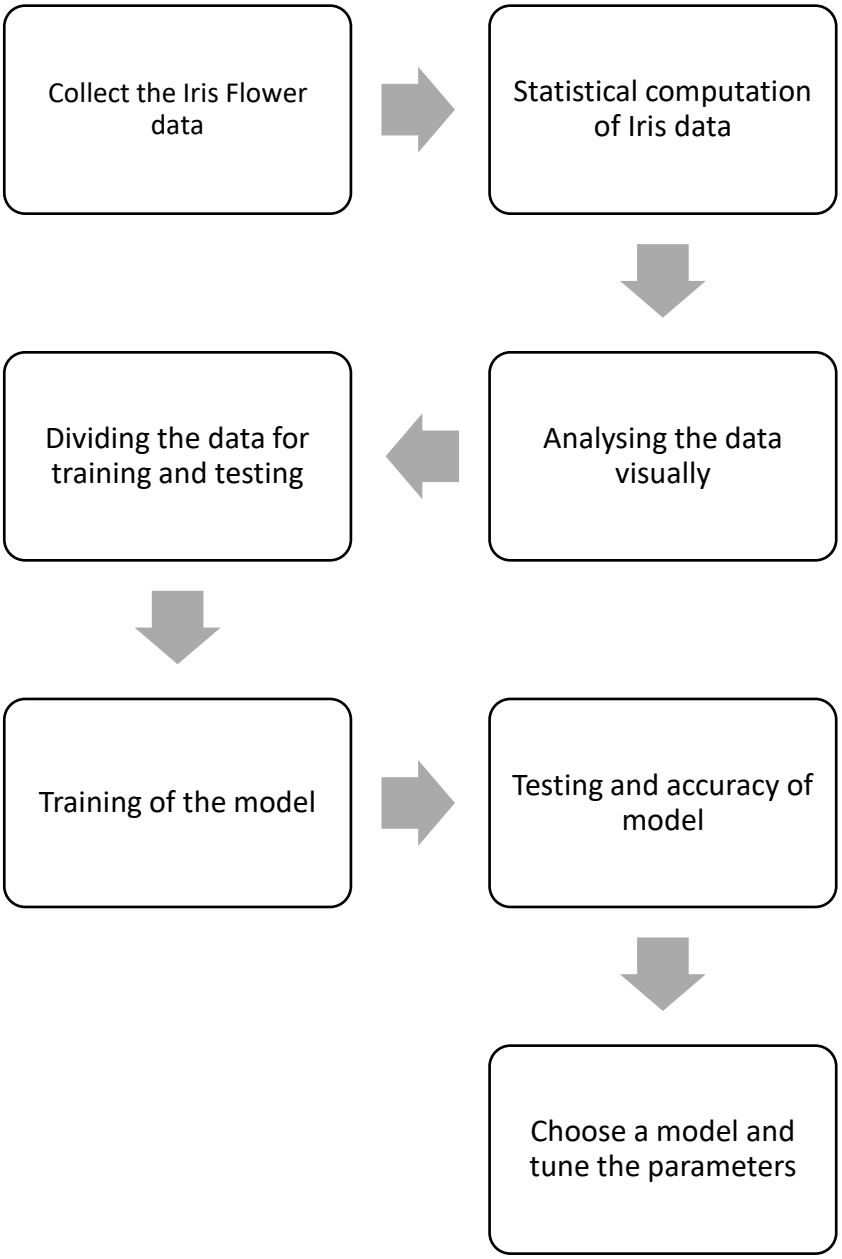
- Processor – i3
- 2 GB RAM
- Memory – 5 GB

1.5. SOFTWARE AND LIBRARIES REQUIREMENTS:

- python 3.7.2
- Jupyter Notebook
- sklearn
- csv
- numpy
- pandas
- matplotlib

2. PROPOSED MODEL

2.1. BLOCK DIAGRAM



Above block diagram is showing the step-by-step process to predict iris flower species.

Iris Data:

The dataset for this project originates from the UCI Machine Learning Repository. The Iris flower data set or Fisher's Iris data set is a multivariate data set. The data set consists of 50 samples from each of three species of Iris (Iris virginica, Iris versicolor and Iris setosa).

- Four features were measured from each sample (in centimeters):
    - Length of the petals
    - Width of the petals
    - Length of the sepals
    - Width of the sepals

Understanding the data:

Iris flower data set contains the observation data with 150 samples. Since the dataframe has four features (Sepal width, sepal length, petal width and petal length) with 150 samples belonging to either of the three target classes. In this step we going into the mathematics of the dataset to find out the standard deviation, mean, minimum value and the four-quartile percentile of the data. Since the dataframe has four features (Sepal width, sepal length, petal width and petal length) with 150 samples belonging to either of the three target classes. In this step we going into the mathematics of the dataset to find out the standard deviation, mean, minimum value and the four-quartile percentile of the data.

Analysing the data visually:

It shows us the visual representation of how our data is scattered over the plane. This method is used in statistical analysis to understand various measures such as mean, median and deviation. To understand how each feature accounts for classification of the data, we plot the

graphs which shows us the correlation with respect to other features. This method helps just to figure out the important features which account the most for the classification in our model.

Dividing the data for training and testing:

The data we use will split into training data and test data. The training set contains a known output and the model learns on this data in order to be generalized to other data later on. We have the test dataset (or subset) in order to test our model's prediction on this subset.

Train the model:

Using some of the commonly used algorithms, we will be training our model to check which algorithm is best suitable for our model. The algorithms that will be using are:

- Logistic Regression
- K – Nearest Neighbor (KNN)
- Random forest
- Support Vector Machine (SVM)

Choose a model and tune the parameters:

From the above models we will choose a model which will give us the best accuracy. After that the parameters will be tuned to get the class of the IRIS flower.

3. IMPLEMENTATION

3.1. SETUP VIRTUAL ENVIRONMENT OR START JUPYTER NOTEBOOK

Virtual environment helps to keep dependencies between different projects. Its main purpose is to create an isolated environment for python projects. To setup virtual environment we have to follow some steps that are:

1. Open the terminal
2. Setup the pip package manager
3. Create the virtual environment
4. Activate the virtual environment
5. After that install the appropriate libraries (numpy, pandas, etc.) using pip

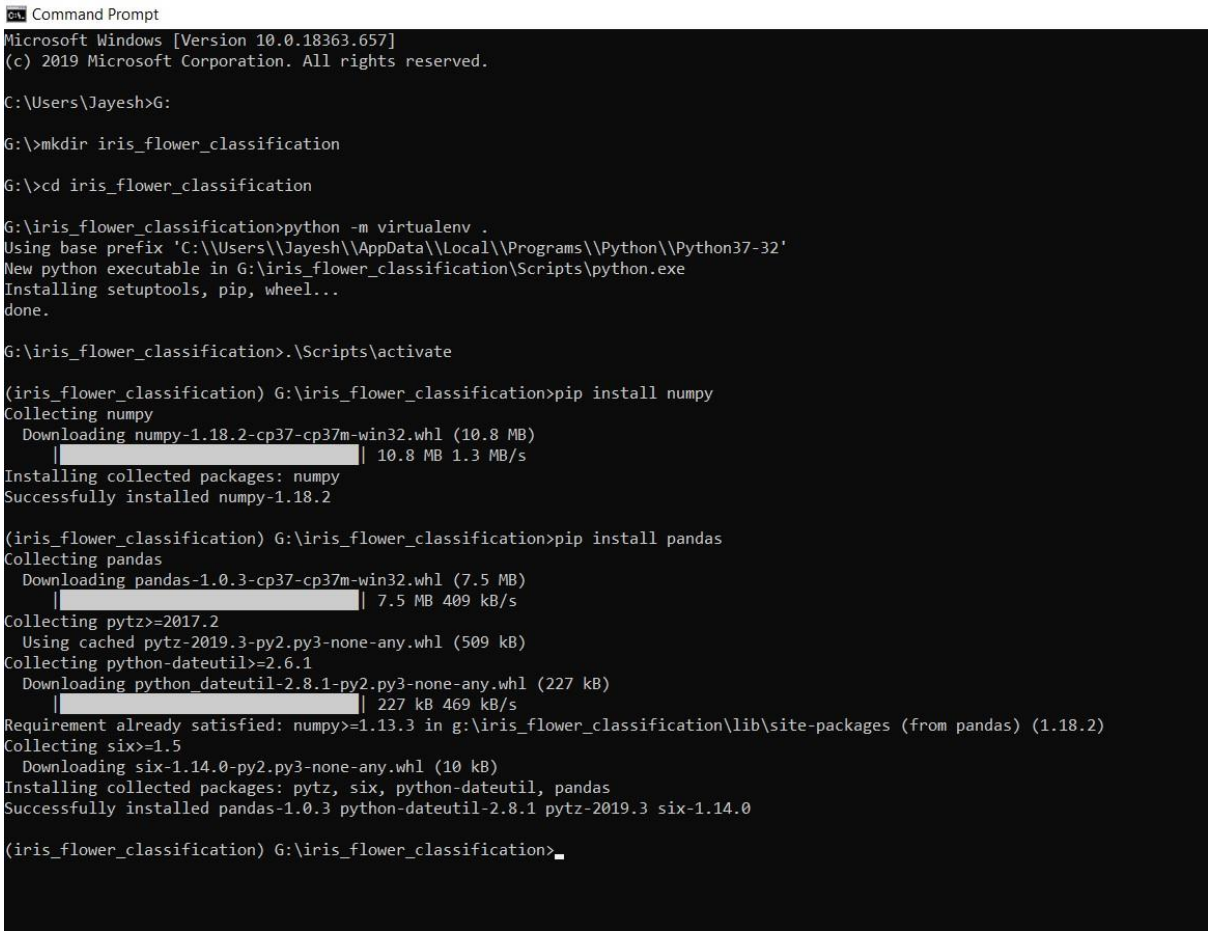


Fig.1 Process for setup virtual environment

3.2. IMPORTING LIBRARIES AND DOWNLOAD THE DATA

The following libraries are required for this project:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
sns.set(color_codes=True)
%matplotlib inline
```

Fig.2 Imported Libraries

Here, we are importing numpy, pandas, seaborn, matplotlib and sklearn libraries. Where, numpy is an array processing package which is used in scientific computing with arrays. Pandas is built on the Numpy package and its key data structure is called the DataFrame. DataFrames allow us to store and manipulate tabular data in rows of observations and columns of variables. Seaborn is a library for statistical graphical representation and data visualization which is based on matplotlib. Matplotlib is a visualization library or plotting library used to generate plot, histogram, bar-chart, pie-chart, etc. Scikit-learn provides a range of machine learning algorithms which contains both unsupervised and supervised learning algorithms via a consistent interface in Python.

The iris dataset can be downloaded from the UCI Machine Learning Repository. Characteristics of data set is multivariate. This data set contain four attributes i.e. sepal length, sepal width, petal length, petal width in cm and it also contain three classes i.e. iris setosa, iris versicolour and iris virginica. The dataset downloaded from the UCI Machine Learning Repository is in the form of CSV (Comma Separated Values) file and the file name is ‘iris.data’ and save the file in the same directory as our project contains.

3.3. DATA EXPLORATION

Now we are going to move into data exploration as well as analysis using the iris data. Let's import our data set using 'pandas' library, which will convert our data into the tabular format from the CSV format. The beauty of using pandas library is just that we can read the csv files.

For converted our data into the understandable format we have to add column to the imported dataset which contain the attributes (sepal length, sepal width, petal length, petal width), it gives heading for the imported data.

```
df=pd.read_csv("iris.data")
df=pd.read_csv("iris.data", header=-1)
column_name=["sepal length","sepal width","petal length","petal width","Iris Setosa"]
df.columns=column_name
df.head()
```

	sepal length	sepal width	petal length	petal width	Iris Setosa
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

Table 1: Showing Iris Dataset using pandas Library

Or we can use seaborn instead of pandas as:

```
iris=sns.load_dataset("iris")
print(iris.head())
```

```
   sepal_length  sepal_width  petal_length  petal_width species
0           5.1          3.5           1.4          0.2  setosa
1           4.9          3.0           1.4          0.2  setosa
2           4.7          3.2           1.3          0.2  setosa
3           4.6          3.1           1.5          0.2  setosa
4           5.0          3.6           1.4          0.2  setosa
```

Table 2: Showing Iris Dataset using seaborn Library

3.4. DATA ANALYSIS

This dataset contains 150 samples Since the dataframe has four features (Petal length, petal width, sepal length and sepal width) with 150 samples belonging to either of the three target classes, and each class has distributed equally.

```
print(iris.groupby("species").size())
```

```
species
setosa        50
versicolor    50
virginica     50
dtype: int64
```

Table 3: Species in Iris Dataset

By using 'df.describe()' we can see the mathematics of the dataset, which helps to find out the standard deviation, mean, minimum value and the four quartile percentile of the data.

```
df.describe()
```

	sepal length	sepal width	petal length	petal width
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

Table 4: Statistical description of iris dataset

We can analyze some more information about our dataset, that it contains four non-null columns and one object-based column. We can also see memory usage by the iris dataset.

```
print(iris.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
sepal_length    150 non-null float64
sepal_width     150 non-null float64
petal_length    150 non-null float64
petal_width     150 non-null float64
species         150 non-null object
dtypes: float64(4), object(1)
memory usage: 5.9+ KB
None
```

Fig.3 Information of iris dataset

3.5. DATA VISUALIZATION

In the previous section what we gone through is the exploration of all the data where we did some preliminary analysis of the data and get a few of it, but to progress further and to dive into the data a little bit more we are going to do some visualization. Visualization is a great way to develop a better understanding of your data and python and has a lot of great tools for specifically that purpose.

3.5.1. Pair-plot:

As we already import the seaborn so we just have to perform the pair-plot using the iris dataset we have.

To understand how each feature accounts for classification of the data, we can build a pair-plot which shows us the correlation with respect to other features.

In the below picture if we look carefully, we can see that all the attributes are plotted against each other and the three different colours shows the distribution of three individual species (setosa, versicolor and verginica). It shows the distinctive relationships between the attributes.

```
sns.pairplot(iris, hue='species', height=3, aspect=1);
plt.show();
```

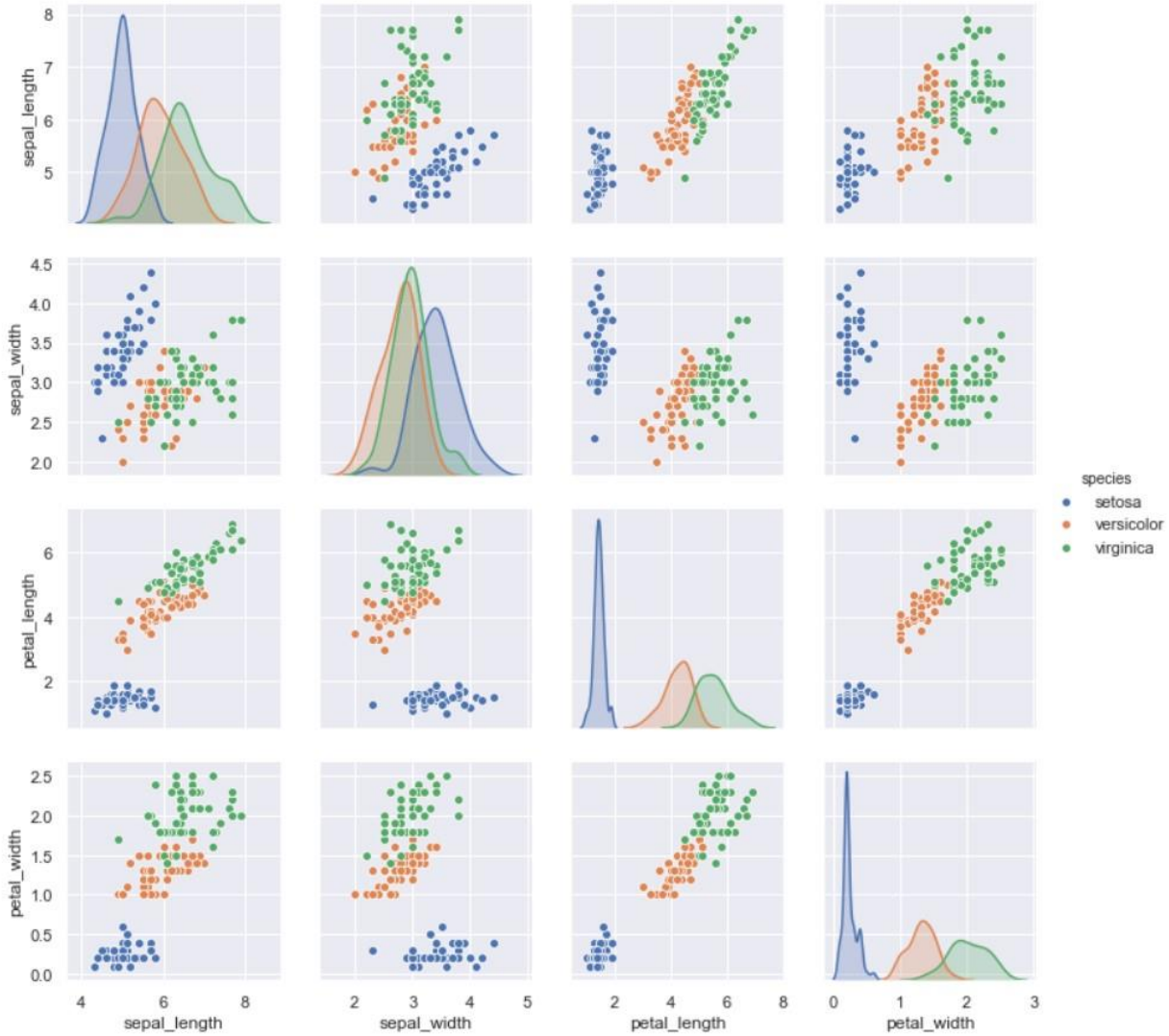


Fig.4 Pair-plot

3.5.2. Histogram

Historical representation is basically the pure distribution off all three combined species and from this it's not really all that informative because it just tells us overall distribution.

```
iris.hist(edgecolor='red', figsize=(12,8), linewidth=1.2)
plt.show()
```

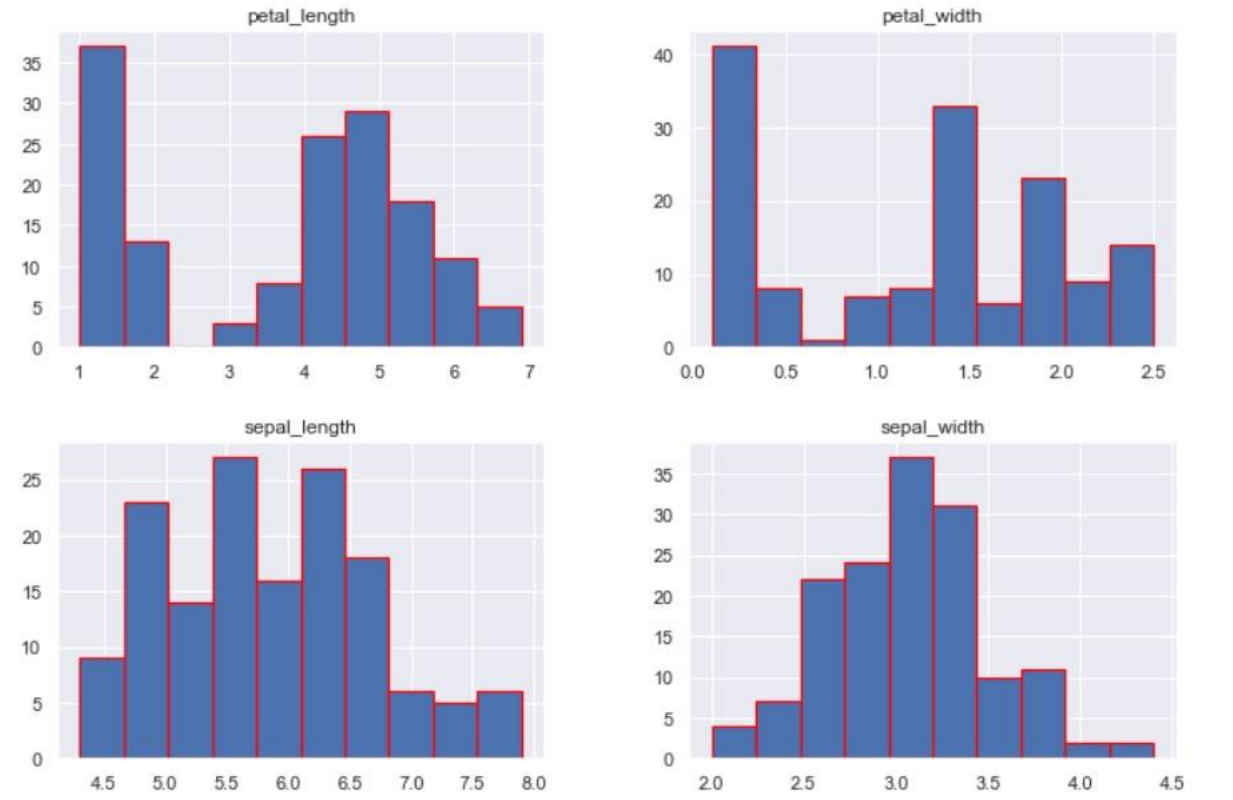


Fig.5 Histogram

3.5.3. Violin-plot:

Let us look at the violin-plot which is as similar to the box-plot, Violin plot shows us the visual representation of how our data is scattered over the plane. Here we can conclude from the below picture that in sepal length we can see this the distribution in setosa is much smaller than the versicolor and verginica. In sepal width we can examine that the distribution of setosa is

widest and also the longest sepal width and longest petal length in comparisons to the other attributes.

```
plt.figure(figsize=(12,8));
plt.subplot(2,2,1)
sns.violinplot(x='species', y='sepal_length', data=iris)
plt.subplot(2,2,2)
sns.violinplot(x='species', y='sepal_width', data=iris)
plt.subplot(2,2,3)
sns.violinplot(x='species', y='petal_length', data=iris)
plt.subplot(2,2,4)
sns.violinplot(x='species', y='petal_width', data=iris)
plt.show()
```

Output:

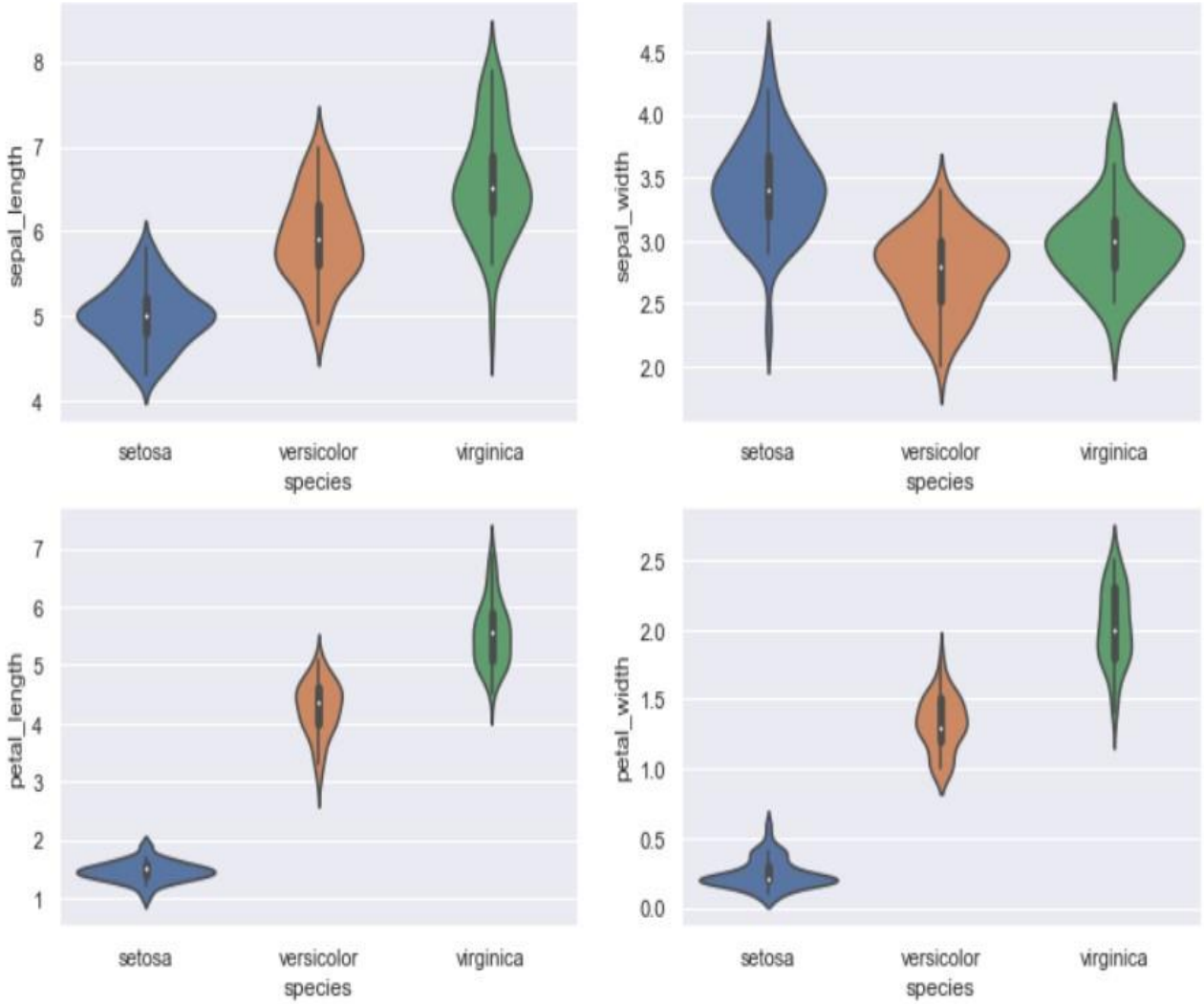


Fig.6 Violin-plot

3.5.4. Box-plot:

Box plot is a graph which is based on percentile, which divides the data into four quartiles of 25% each. This method is numerously used in statistical analysis to understand various measures such as max, min, mean, median and deviation.

```
iris.boxplot(by='species', figsize=(12,8))
plt.show()
```

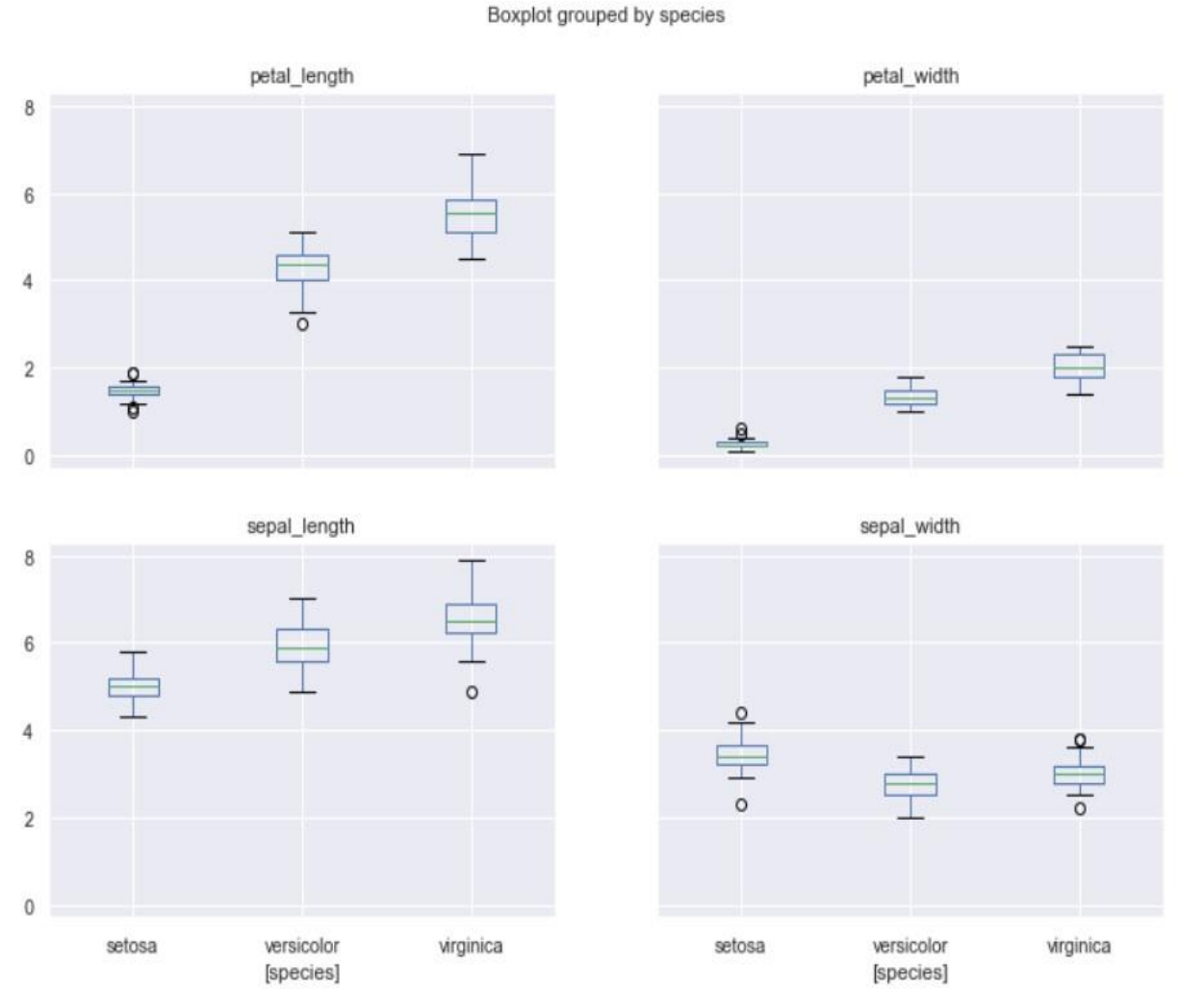


Fig.7 Box-plot

3.6. DIVIDING THE DATA FOR TRAINING AND TESTING

When we will understand what the dataset is about, we can start training our model based on the algorithms. First, we have to train our model with some of the samples. Here, we will be using scikit-learn library method called 'train_test_split' which divides our data set into a ratio of 80:20, in which 80% data will be using for training and 20% data will be using for testing. This process can be done by the following code:

```
X = df.iloc[:, :-1]
y = df.iloc[:, -1]
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=0)
```

Fig.8 Code for splitting dataset in training and testing dataset

Let's see our test data,

```
print(x_test.head())
```

	sepal length	sepal width	petal length	petal width
114	5.8	2.8	5.1	2.4
62	6.0	2.2	4.0	1.0
33	5.5	4.2	1.4	0.2
107	7.3	2.9	6.3	1.8
7	5.0	3.4	1.5	0.2

Table 5: Test Data

3.7. ALGORITHMS FOR TRAINING THE MODEL

3.7.1. LOGISTIC REGRESSION

Logistic Regression is a type of regression that predicts the probability of occurrence of an event by fitting the appropriate or cleaned data to a logistic function. Like several forms of regression analysis, it makes use of many predictor variables that will be either numerical or categorical. For instance, the probability that a email received is spam or not might be predicted

from knowledge of the type of data or the history of sender. This regression is quite used in several scenarios such as prediction of customer's propensity of purchasing a product or used in market analysis to improve the business and use to predict the unpredictable scenarios.

What is logistic regression?

Logistic Regression, also known as Logit Model or Logit Regression, is a mathematical model used in statistics to estimate (guess) the probability of an event occurring having been given some previous data. Logistic Regression works with binary data, where either the event happens (True or 1) or the event does not happen (False or 0). So, by giving some feature x it tries to find out whether some event y happens or not. So, y can either be 0 or 1. In the case if the event happens, y is given the value 1. If the event does not happen, then y is given the value of 0. For example, if y represents whether a coin gives head, then y will be 1 if after tossing the coin we get head or y will be 0 if we get tail. This is known as Binomial Logistic Regression. Logistic Regression can also be used when there is use of multiple values for the variable y. This form of Logistic Regression is known as Multinomial Logistic Regression.

Logistic regression is named for the function used at the core of the method, the logistic function. The logistic function, also called the Sigmoid function was developed by statisticians it’s an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1.

$$\frac{1}{1 + e^{-x}}$$

Eq. 1

Where,

e: base of the natural logarithms

x: value that you want to transform via the logistic function

The logistic regression equation has a very similar representation like linear regression. The difference is that the output value being modelled is binary in nature.

$$\hat{y} = \frac{e^{\beta_0+\beta_1 x_1}}{1 + \beta_0 + \beta_1 x_1}$$

or

$$\hat{y} = \frac{1.0}{1.0 + e^{-\beta_0-\beta_1 x_1}}$$

Eq. 2

$\beta 0$: intercept term

$\beta 1$: coefficient for $x1$

^y: predicted output with real value between 0 and 1

```
x = np.linspace(-6, 6, num = 1000)
plt.figure(figsize = (12,8))
plt.plot(x, 1 / (1 + np.exp(-x))); # Sigmoid Function
plt.title("Sigmoid Function");
```

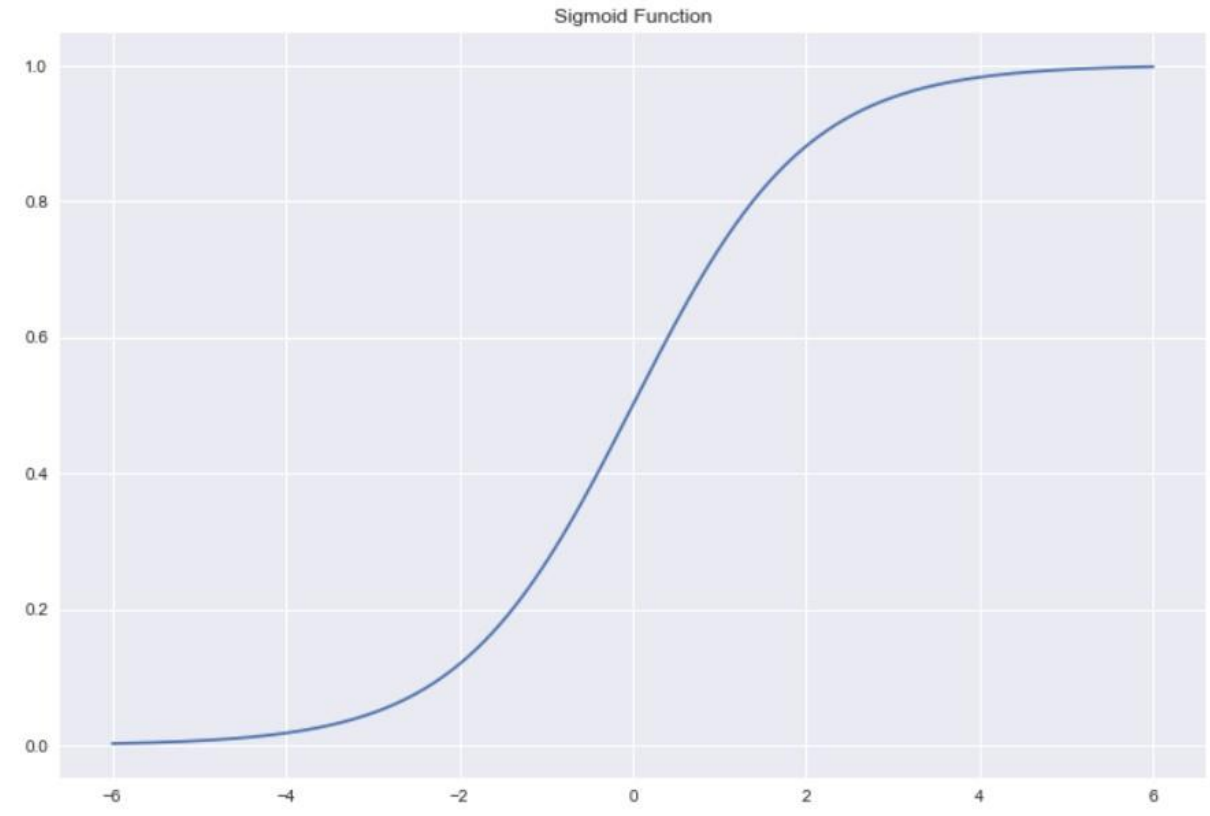


Fig.9 Graph of Sigmoid Function

3.7.2. K-NEAREST NEIGHBOR ALGORITHM

K-Nearest Neighbor (KNN) is one of the simplest supervised machine learning algorithms mostly used for classification which uses an entire dataset in extreme phase. When prediction is required for unknown data it searches through the entire training dataset for k more similar instances and the data with the most similar instance is finally returned as the prediction.

What is KNN Algorithm?

K-Nearest Neighbor is an algorithm which stores every accessible case and characterizes the new cases dependent on the component similarity measure or similarity features. We can say that if we are similar to our neighbors then we are one of them. For example, if apple looks most similar to banana, orange or melon rather than a monkey, rat or a cat then most likely apple belongs to the group of fruits. But in-general KNN is using such application where you're looking for similar items i.e. when the task is some form of fine items similar to this one then you call the search as a KNN search.

Let's see the example of scenarios that is used in the industry:

So let's see the industrial application of KNN algorithm starting with recommender system, the biggest use case of KNN search is a recommender system, this recommender system is like an automated form of a shop counter person, today when you're asking for a product not only shows you the product but also suggest you or displays you relevance set of products which are related to the item you're already interested in buying this KNN algorithm is applied to recommending products like an Amazon or a recommending media like in case of Netflix or Today's almost all of us must have used Amazon for shopping so just for a knowledge, more than thirty five percent of Amazon dot com's revenue is generated by its recommendation engine and so what's the strategy the Amazon uses recommendation as a targeted marketing

tool in both the email campaigns and on most of its website pages. Amazon will recommend many products from different categories based on what we are browsing and it will pull those products in front of us which are likely to buy like the frequently bought together option that comes at the bottom of the product page to tempt us into buying the combo well this recommendation has just one main goal that is increase average order value by providing product suggestions based on items in the shopping cart well based on the product they're currently looking out on site.

What is K in KNN Algorithm?

In K-Nearest Neighbor, K denotes the number of nearest neighbors which are voting for the class of the new data or the testing data. For example, if K=1 then the testing data are given the same label as a close to this example in the training set, similarly if K=3 or more the labels of the three closest classes are checked and most common label are assigned to the testing data.

As we can see in figure the value of K increases decision region gets smoother. If K=1 then there will be the case of overfitting as it takes only one neighbor. Most of the value of K lies between 3-10 to generate accurate result.

K = Number of Nearest Neighbors

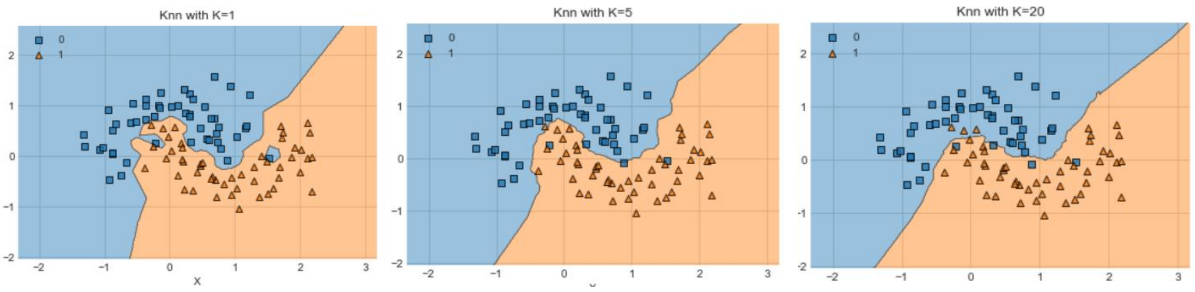


Fig.10 Decision Region graph

KNN algorithm is based on feature similarity choosing the right value of K is a process called parameter tuning and is important for better accuracy. There is no particular method for finding the value of K but for finding k we could keep some points in our mind like K shouldn't be too big, if it's too big then it's going to take forever to process so this going to run into processing issues and resources issues. To prevent from these issues there are some methods to choose a value of K:

1) Sqrt(n), where n is the total number of data points,
2) Odd value of K is selected to avoid confusion between two classes of data,
3) There is another way to choose the value of K i.e. cross-validation, in this way take the small portion of training dataset and call it a validation dataset and then use the different values of K to choose the best performance on the validation set and then choose that value of K to minimizing the validation error.

How does KNN algorithm Work?

In the given below fig. there is blue and orange point on a graph so these blue points belong to class A and orange ones belong to class B. Now we can see a new point in the form of star and our task is to predict whether this new point belongs to class A or it belongs to class B. So to start the prediction the very first thing that we have to do is to select the value of K, as we know K in KNN algorithm refers to the number of nearest neighbors that you want to set, for example in this case take K equal to three so what does that mean it means that select three points which are the least distance to the new point or we can say that select three different points which is closest to the star. KNN uses least distance measures, So when we calculate the distance we'll get one blue and two orange points which are closest to the star now since in this case as we have a majority of orange points so we can say that for K equal to three the star belongs to class B. all we can see that the star is more similar to the orange points.

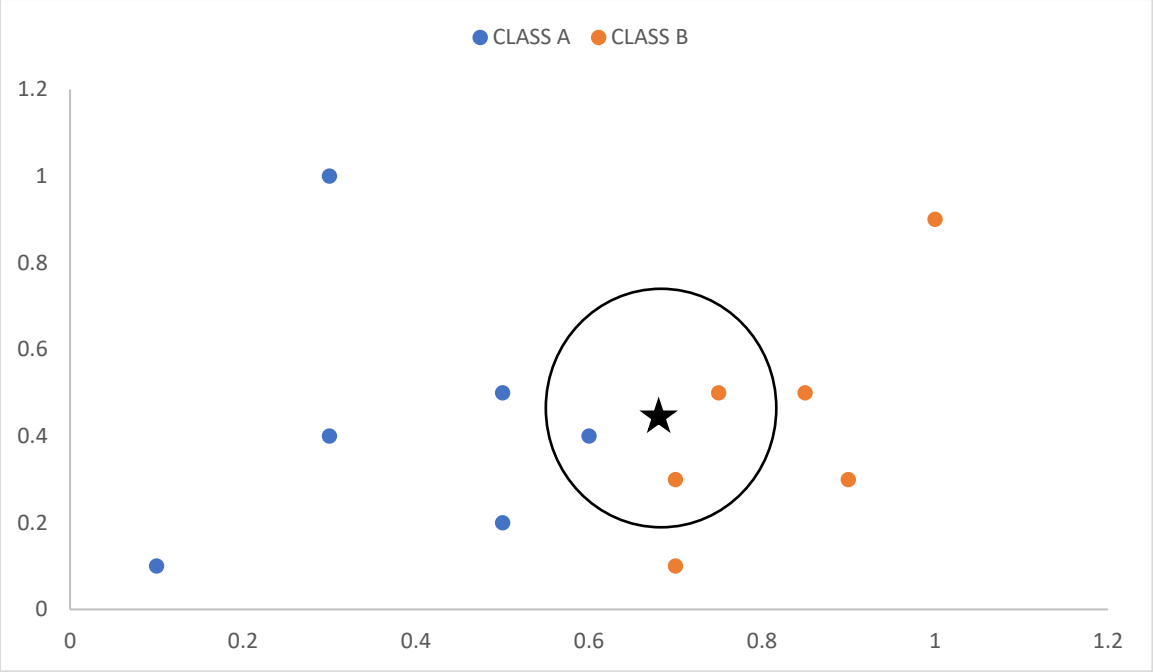


Fig.11 Graph of 3- Nearest Neighbor

Mathematics of KNN algorithm

There are different types of data and similarity measures are dependent on that. The Euclidean distance can be used for real-valued data. Hamming distance can be used for other types of data such as binary or categorical data. The average of the predicted attribute may be returned in the case of regression problems. The most prevalent class may be returned in the case of classification.

Euclidean Distance

Euclidean distance is defined as the square root of the sum of difference between the new point x and existing point y. KNN algorithm use the Euclidean Distance to find the K number of nearest neighbors.

$$\text{Euclidean distance} = \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

Eq. 3

Manhattan Distance

Manhattan distance has used to calculate the distance between real vector using sum of their absolute difference.

$$\text{Manhattan distance} = \sqrt{\sum_{i=1}^{k} |x_i - y_i|}$$

Eq. 4

KNN is called as a lazy learner. KNN as a classification is a very simple algorithm but that's not the reason why it is called lazy learner. KNN is a lazy learner because it doesn't have a discriminative function from the training data but what it does, it memorizes the training data. There is no learning phase of the model and all of the work happens at the time of prediction is requested that's why KNN is referred as a lazy learner algorithm.

3.7.3. RANDOM FOREST ALGORITHM

Random forest or random decision forest is an ensemble classifier that operates by constructing multiple decision trees models during the training phase. Ensemble models combines the decision from multiple models to choose the result as the final decision.

What is Random Forest?

Random forest is a supervised machine learning algorithm used for classification. In this the trees are trained on subsets which are being selected as random therefore this is called random forest. So, random forest is the collection or an ensemble of decision trees. The whole dataset is used in the decision tress with considering all features but in random forest only the subset of dataset is selected at random and the particular number of features are selected at random, that is how the random forest is built upon. Number of decision trees will be grown and each

decision tree will result into certain final outcome and random forest just collect results of all the decision trees and the decision of the majority of the trees will be the final result.

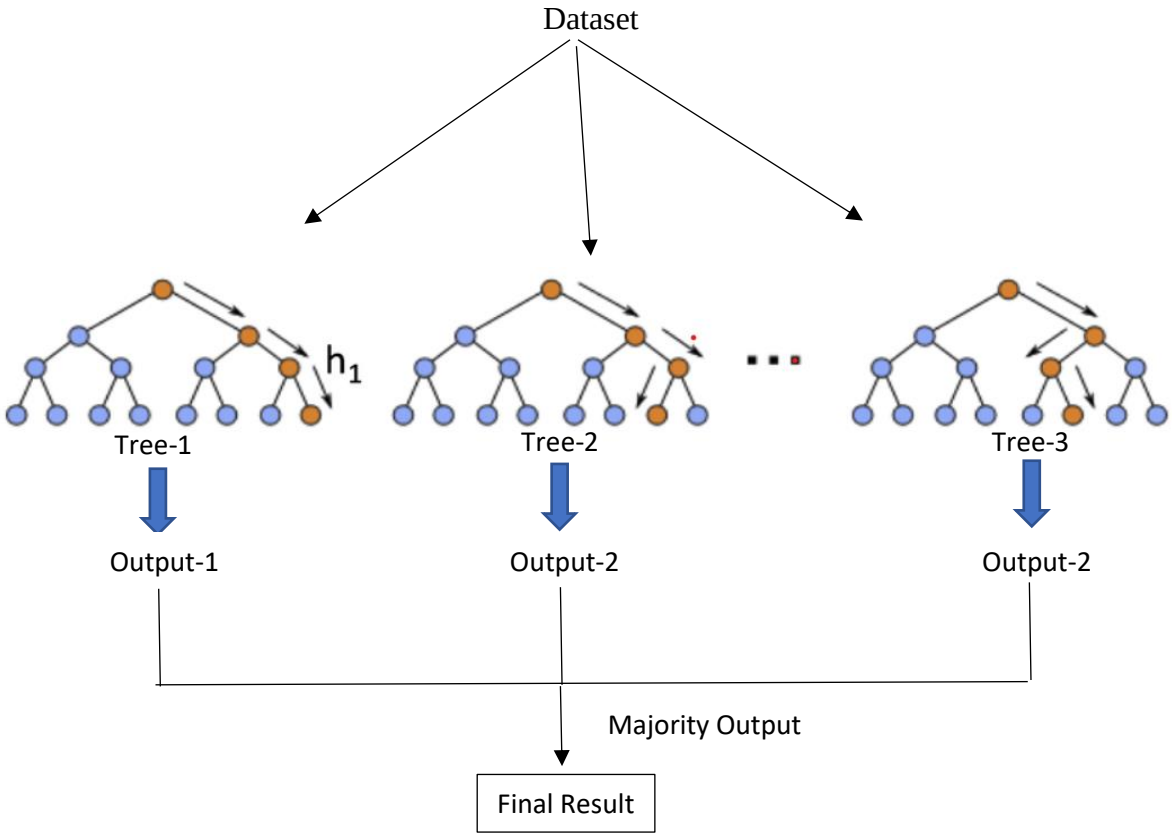


Fig.12 Working Diagram of Random Forest

Random Forest is a versatile algorithm capable of performing both regression and classification. Random Forest has many benefits like

- It uses the multiple trees which reduce the risk of overfitting
- Random Forest has high accuracy that runs efficiently on large database.
- Random Forest algorithm can also estimate missing data. Random Forest can maintain accuracy when a large proportion of data is missing.

What is decision tree?

Decision tree builds classification models in the form of a tree structure. Decision tree splits the entire dataset in the structure of a tree and it makes decision at every node hence called decision tree. Decision Tree has some important terms which are:

Entropy: Entropy is a measure of randomness or unpredictability in the data set.

Entropy for one attribute-

$$E(T) = \sum_{i=1}^{c} -p_i \log_2 p_i$$

Eq.5

Entropy for two attributes-

$$E(T, X) = \sum_{c \in X} P(c)E(c)$$

Eq.6

Information gain: Information gain is the measure of the decrease in entropy after a dataset is split.

$$Gain(T, X) = Entropy(T) - Entropy(T, X)$$

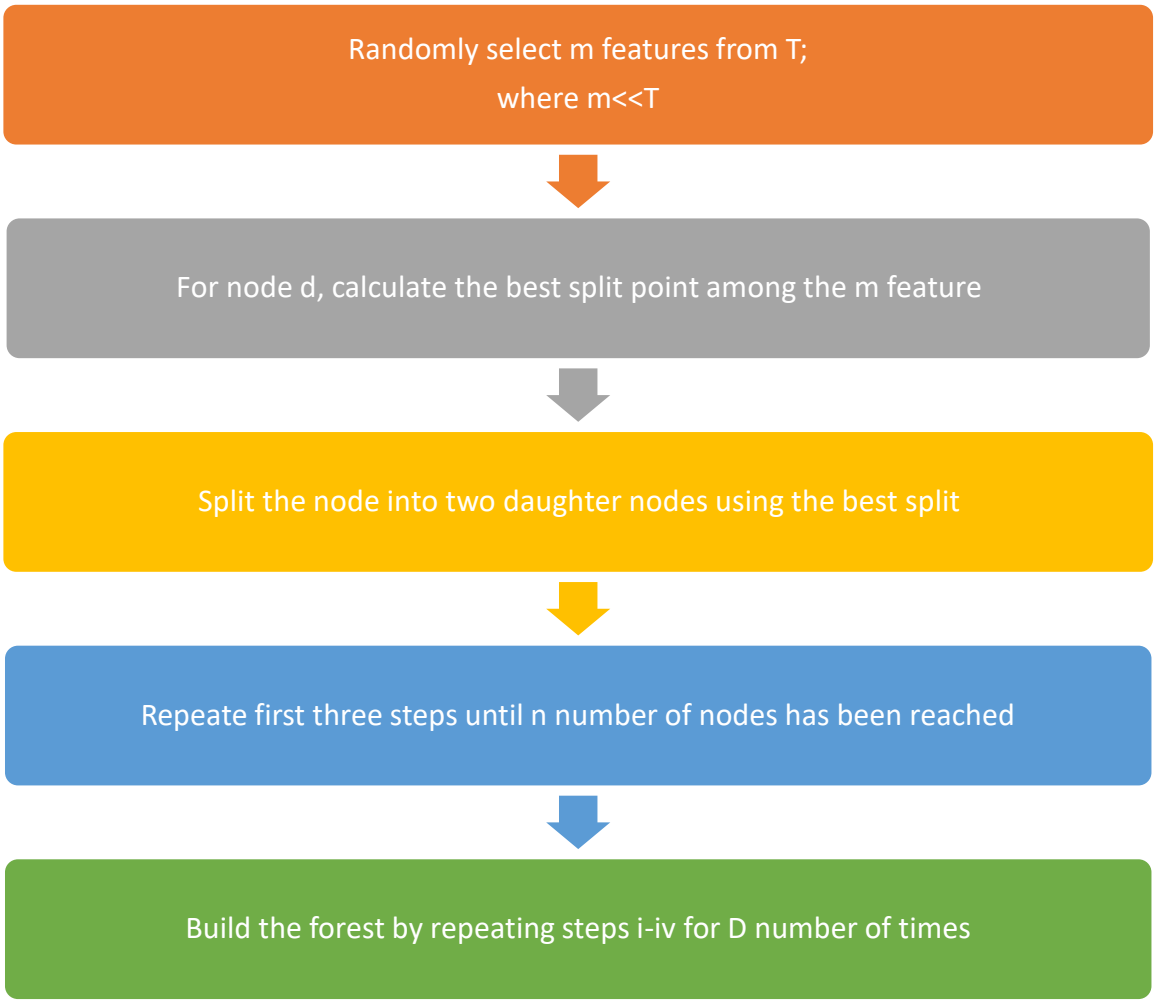
Eq.7

Leaf Node: Final decisions or classification results of a tree are known as leaf node.

Decision Node: Node which has two or more branches in decision tree.

Root Node: The top most node in decision tree is the root node.

How Random Forest Algorithm works?



T: number of features

D: number of trees to be constructed

V: output; the class with highest vote

3.7.4. SUPPORT VECTOR MACHINE ALGORITHM

A support vector machine abbreviated as SVM was first introduced in the 1960's and late an improvised in the 1990's. SVM is supervised learning machine learning classification algorithm that has become extremely popular nowadays owing to its extremely efficient results so SVM is implemented in a slightly differently than other machine learning algorithms it is capable of performing classification and regression and outlier detection as well.

What is Support Vector Machine?

Support vector machine is a discriminative classifier that is formally designed by a separative hyperplane. It is a representation of examples as points in space that are mapped so that the points of different categories are separated by a gap as wide as possible. SVM does not directly provide probability estimates these are calculated using five-fold cross validation. Five-fold cross validation means the dataset will divided randomly into 5 subsets and then take a subset for use as a test set and use remaining subgroups as a training set, then fit a model on the training set and evaluate the score, this will happen until each subset one-by-one considered as a test-set.

The main objective of support vector machine is to separate the given data of different classes by a line (hyperplane) in the best possible way. The nearest points of classes from the hyperplane is known by support vectors. There can be many hyperplanes that will separate the classes, so to choose the appropriate hyperplane SVM algorithm find the nearest points to the hyperplane of both the classes and check the distance between the hyperplane and support vectors. Here, this distance is known by margin. SVM algorithm selects the hyperplane which gives the maximum margin.

In the below figure in Graph-A we can see that there are two lines blue and green. It is clearly seen that blue is placed in the space in which the support vectors give maximum margin from the blue line therefor blue line will selected as a hyperplane.

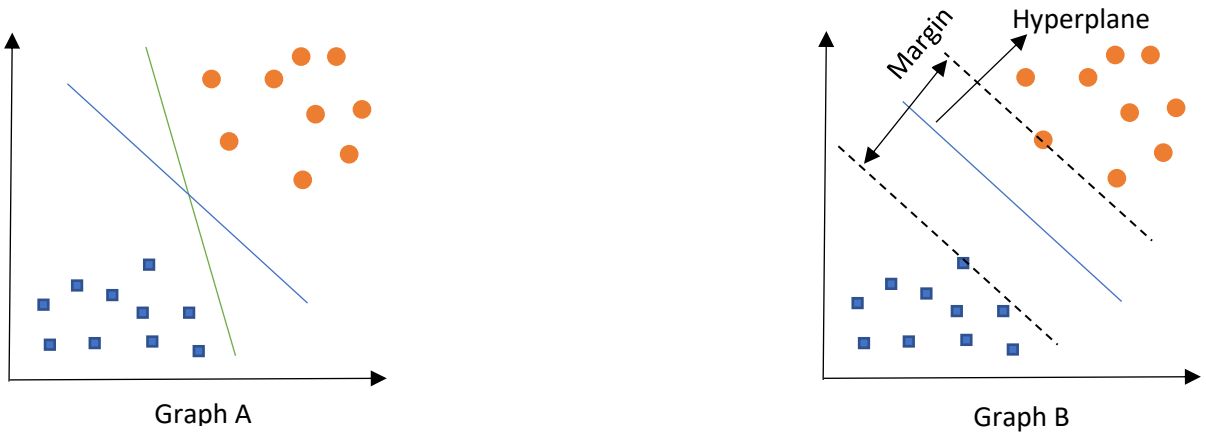


Fig.13 Select hyperplane

The distribution of the data can be critical means data will not always be separated in linear form; it can be inseparable in linear plane. So, to separate this type of data there is a trick in SVM i.e. kernel. Kernel is used to transform the input in higher dimensional space so, the separation between the classes can be easy.

What is SVM kernel?

SVM Kernels is basically used to add more dimensions to a lower dimensional space to make it is easier to separate the data. It converts the linearly inseparable problem in linearly separable problem by adding more dimensions using the kernel. a support vector machine is implemented in practice by kernel, the kernel helps to make a more accurate classifier. In the below figure it is shown that to separate the classes it is mandatory to add one more dimension. So, the z-axis is added to the graph to separate the classes.

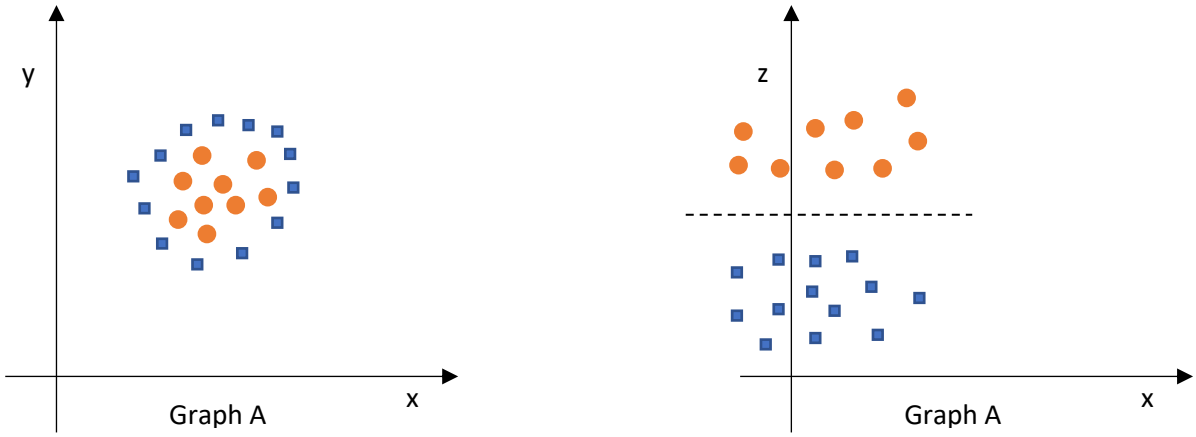


Fig.14 Add dimension to separate the classes

There are three types of kernels:

1. Linear Kernels: A linear kernel can be used as a normal dot product between any two given observations.

2. Polynomial Kernels: It is a generalized form of linear kernels. It can distinguish curved and non-linear input space as well.

3. Radial Basis Function Kernel: It is commonly used in SVM classification, it can map the space in infinite dimensions.

3.8. TRAINING OF THE MODEL

For training of the data fit the 80% trained data into the model. By the below codes we can train our data.

By Logistic regression

```
model = LogisticRegression()
model.fit(x_train, y_train)
```

By K-Nearest Neighbor Algorithm

```
from sklearn.neighbors import KNeighborsClassifier
model_1 = KNeighborsClassifier(n_neighbors=5)
model_1.fit(x_train, y_train)
```

By Random Forest Algorithm

```
from sklearn.ensemble import RandomForestClassifier
model_2 = RandomForestClassifier(n_estimators=5)
model_2.fit(x_train, y_train)
```

By Support Vector Machine Algorithm

```
from sklearn.svm import SVC
model_3 = SVC(kernel='linear')
model_3.fit(x_train, y_train)
```

Fig.15 Code for train the model

3.9. PREDICT THE DATA AND ACCURACY OF THE MODELS

In this step we will predict iris species of our test data and also find the iris species of unknown data i.e. data out of the box or can say that, data outside the taken dataset based on what it has learnt in previous step and get the result.

For test data:

```
predictions = model.predict(x_test)
print(predictions)
```

```
['Iris-virginica' 'Iris-versicolor' 'Iris-setosa' 'Iris-virginica'
 'Iris-setosa' 'Iris-virginica' 'Iris-setosa' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-setosa'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-virginica' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-setosa' 'Iris-setosa' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-setosa']
```

```
KNN_predictions = model_1.predict(x_test)
print(KNN_predictions)
```

```
['Iris-virginica' 'Iris-versicolor' 'Iris-setosa' 'Iris-virginica'
 'Iris-setosa' 'Iris-virginica' 'Iris-setosa' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica' 'Iris-setosa'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-setosa' 'Iris-setosa' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-setosa']
```

```
RF_predictions = model_2.predict(x_test)
print(RF_predictions)
```

```
['Iris-virginica' 'Iris-versicolor' 'Iris-setosa' 'Iris-virginica'
 'Iris-setosa' 'Iris-virginica' 'Iris-setosa' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-setosa'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-setosa' 'Iris-setosa' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-setosa']
```

```
SVM_predictions = model_3.predict(x_test)
print(SVM_predictions)
```

```
['Iris-virginica' 'Iris-versicolor' 'Iris-setosa' 'Iris-virginica'
 'Iris-setosa' 'Iris-virginica' 'Iris-setosa' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-setosa'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-setosa' 'Iris-setosa' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-setosa']
```

Fig.16 Code for predictions of test data

For unknown data:

```
p = pd. DataFrame([[1,3,5.1,2.4]], columns = ["a", "b","c","d"])
predictions = model.predict(p)
print(predictions)
```

```
['Iris-virginica']
```

Fig.17 Prediction of unseen data

Now, for finding accuracy of the model we will use 20% test data predictions and y_test. The code for the following is:

```
print("Accuracy of logistic regression algorithm:", accuracy_score(y_test, predictions))
print("Accuracy of KNN algorithm:", accuracy_score(y_test, KNN_predictions))
print("Accuracy of Random Forest algorithm:", accuracy_score(y_test, RF_predictions))
print("Accuracy of SVM algorithm:", accuracy_score(y_test, SVM_predictions))
```

```
Accuracy of logistic regression algorithm: 0.9666666666666667
Accuracy of KNN algorithm: 0.9666666666666667
Accuracy of Random Forest algorithm: 1.0
Accuracy of SVM algorithm: 1.0
```

Fig.18 Accuracy of trained models

There are two models which shows the highest accuracy i.e. Random Forest and SVM with accuracy score 1.0.

4. CONCLUSION

In this project, we used the various powerful algorithms to train our data. Processing of data is also important to acquire the best result and as we can see the above results, they are very satisfactory. The accuracy score of above four models are very good and they can be used to predict the species of iris flower and in four of the above models two models shows the 100% accuracy. As we can conclude that in future with appropriate data of features of any flower it is possible to classify the species of any flower.

5. REFERENCES

- International Journal on Soft Computing (IJSC) Vol.3, No.1, February 2012
- http://lab.fs.uni-lj.si/lasin/wp/IMIT_files/neural/doc/seminar8.pdf
- https://archive.ics.uci.edu/ml/datasets/iris
- https://www.neuraldesigner.com/learning/examples/iris-flowers-classification