Karunadu Technologies Private Limited

#17, ATK complex, 4th Floor, Acharya college main road, beside Karur Vysya bank, Guttebasaveshwaranagar Chikkabanvara, Bengaluru, Karnataka-560090

Internship Report

On

Python With Machine Learning

By

SUMANTHA KM
(4GK21EC045)

BACHELOR OF ENGINEERING
In

Department of Electronics & Communication Engineering
K R Pete Krishna Government College, K R Pet.Mandya-571426
2023-2024

List of Contents

List Of Figures

CHAPTER 1

Company Profile

It is pleasure in introducing "Karunadu Technologies Private Limited" as a leading IT software solutions and services industry focusing on quality standards and customer values. It is also a leading Skills and Talent Development company that is building a manpower pool for global industry requirements.

Profile

Fig 1.1 Company Logo

The company offers broad range of customized software applications powered by concrete technology and industry expertise. It also offers end to end embedded solutions and services. They deal with broad range of product development along with customized features ensuring at most customer satisfaction and also empower individual with knowledge, skills and competencies that assist them to escalate as integrated individuals with a sense of commitment and dedication.

Vision

To Empower Unskilled Individual with knowledge, skills and technical competencies in the field of Information Technology and Embedded engineering which assist them to escalate as integrated individuals contributing to company's and Nation's growth.

Mission

- Provide cost effective and reliable solutions to customers across various latest technologies.
- Offer scalable end-to-end application development and management solutions

- Provide cost effective highly scalable products for varied verticals.
- Focus on creating sustainable value growth through innovative solutions and unique partnerships.
- Create, design and deliver business solutions with high value and innovation by leveraging technology expertise and innovative business models to address long-term business objectives.
- Keep our products and services updated with the latest innovations in the respective requirement and technology.

Objectives

- To develop software and Embedded solutions and services focussing on quality standards and customer values.
- Offer end to end embedded solutions which ensure the best customer satisfaction.
- To build Skilled and Talented manpower pool for global industry requirements.
- To develop software and embedded products which are globally recognized.
- To become a global leader in Offering Scalable and cost-effective Software solutions and services across various domains like E-commerce, Banking, Finance, Healthcare and much more.
- To generate employment for skilled and highly talented youth of our Country INDIA.

Company Products and Services Offered

Products

- **KECMS – Karunadu Enterprise Content Management System**

Karunadu Enterprise Content Management System is a one stop solution for all our enterprise content management System relating to digital asset management, document imaging, workflow systems and records management systems. Increasing digitalization has led to an exponential growth in business content and managing this sea of unstructured data is tedious work. KECMS enables you to create, capture, manage, distribute, archive different forms of content and has much more features.

- **KEMS – Karunadu Education Management System**

Manage diversified data relating to education management on cloud. Educational data including students and staff is gathered over years which contain information from admission/appointment until leaving the Education. Statistical reports for the College/school can be generated along with admission Tracking and result analysis to keep track of progressive improvements of both student and staff.

- **KASS – Karunadu Advanced Security System**

A Complete one stop embedded solution for large apartments. Security system which monitors door breakage, window breakage, gas leakage, motion detection and various other features which can be operated and maintained by centralized monitored system. This Embedded solution enhances the security measures of apartment/building and enhances the security of individuals may be from unintended intervention or from unauthorized access.

Services

- **IT Solutions and Services**

Karunadu Technologies is a Bangalore based IT Training and Software Development center with an exclusive expertise in the area of IT Services and Solutions. Karunadu Technologies Pvt. Ltd. is also expertise in Web Designing and Consulting Services.

- **Embedded Design and Development**

Karunadu Technologies Pvt. Ltd. has expertise in Design and development of embedded products and offers solutions and services in field of Electronics.

- **Academic Projects**

Karunadu Technologies Pvt. Ltd. helps students in their academics by imparting industrial experience into projects to strive excellence of students. Karunadu Technologies Pvt. Ltd. encourages students to implement their own ideas to projects keeping in mind "A small seed sown upfront will be nourished to become a large tree one day", thereby focusing the future entrepreneurs. They have a wide range of IEEE projects for B.E, MTech, MCA, BCA, DIPLOMA students for all branches in each and every domain.

- **In plant Training**

Karunadu Technologies Pvt. Ltd. provides Implant training for students according to the interest of students keeping in mind the current technology and academic benefit one obtains after completing the training. Students will be nourished and will be trained throughout with practical experience. Students will be exposed to industrial standards which boost their carrier. Students will become Acquaint to various structural partitions such as labs, workshops, assembly units, stores, and administrative unit and machinery units. They help students to understand their functions, applications and maintenance. Students will be trained from initial stage that is from collection of Project Requirements, Project Planning, Designing, implementation, testing, deployment and maintenance there by helping to understand the business model of the industry. Entire project life cycle will be demonstrated with hands on experience. Students will also be trained about management skills and team building activities. They assure that by end of implant training students will Enhance communication skills and acquire technical skills, employability skills, start-up skills, and will be aware of risks in industry, management skills and many other skills which are helpful to professional engagement.

- **Software Courses**

Karunadu Technologies Pvt. Ltd. provides courses for students according to the interest of students keeping in mind the current technology and assist them for their further Employment. Company provides various courses such as C, C++, VB, DBMS, Dot Net, Core Java and J2EE along with live projects.

Contact Details

#17, ATK complex, 4th Floor, Acharya College Main Road, Beside KarurVysya Bank, Guttebasaveshwaranagar, Chikkabanvara, Bengaluru, Karnataka- 560090

support@karunadutechnologies.com

CHAPTER 2

Topics Learnt During the Course

The objective of the internship is to apply theoretical knowledge of "Machine Learning using Python" to solve real time complex problems, in order to achieve these following basic concepts were learnt:

- Python
- Machine Learning

Python

Python is a multiparadigm, general-purpose, interpreted, high-level programming language. Python allows programmers to use different programming styles to create simple or complex programs, get quicker results and write code almost as if speaking in a human language.

Fig 2.1 Python Logo

The topics learnt on Python are as follows:

- Installation of Python.
- Use of variables to store, retrieve and calculate information.
- Utilization of core programming operations such as functions and loops.
- Operation on strings, python supported libraries for Machine Learning.

Features of Python

- Extensive support libraries (NumPy for numerical calculations, Pandas for data analytics).
- Open source and community development.

- Dynamically typed language (No need to mention data type based on value assigned, it takes data type).
- Object-oriented language, Portable and Interactive across Operating systems.

Python libraries for Data Analytics

Machine Learning, as the name suggests, is the science of programming a computer by which they are able to learn from different kinds of data. A more general definition given by Arthur Samuel is – "Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed." They are typically used to solve various types of life problems.

In the older days, people used to perform Machine Learning tasks by manually coding all the algorithms and mathematical and statistical formula. This made the process time consuming, tedious and inefficient. But in the modern days, it is become very much easy and efficient compared to the olden days by various python libraries, frameworks, and modules. Python libraries that used in Machine Learning are:

- NumPy
- Pandas
- Matplotlib

Numpy

Numpy is basic package for scientific computing. It is the python language implementation which includes powerful N-dimensional array structure, sophisticated functions, Tools that can be integrated into C/C++ and Fortran code, Linear algebra, Fourier transform and Random number features. Besides its obvious scientific uses, numpy can also be used as an efficient multidimensional container of generic data.

The main aspect Numpy is the Numpy array, on which you can do various operations. The key is that a Numpy array isn't just a regular array you'd see in a language like Java or C++, but instead it is like a mathematical object as a vector or a matrix. That means you can do vector and matrix operations like addition, subtraction, and multiplication. The most important aspect of NumPy arrays is that they are optimized for speed.

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. Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data.

- **Basic Array Operations**

NumPy, arrays allow a wide range of operations which can be performed on a particular array or a combination of Arrays. These operations include some basic Mathematical operation as well as Unary and Binary operations.

```
# Python program to demonstrate
# basic operations on single array
import NumPy as np
a = np.array ([[1, 2], [3, 4]])  # Defining Array 1
b = np.array ([[4, 3],  [2, 1]]) # Defining Array 2
print ("Adding 1 to every element:", a + 1) # Adding 1 to every element
print ("\n Subtracting 2 from each element:", b - 2) # Subtracting 2 from each element
# sum of array elements
print ("\n Sum of all array "elements: ", a.sum()) # Performing Unary operations
print ("\n Array sum:\n", a + b) # Performing Binary operations
```

Pandas

Pandas Data Frame is two-dimensional size-mutable, potentially heterogeneous tabular data structure with labelled axes (rows and columns). A Data frame is a two-dimensional data structure, i.e., data is aligned in a tabular fashion in rows and columns. Pandas Data Frame consists of three principal components, the data, rows, and columns as shown in the fig 2.2. The basic operations which can be performed on Pandas Data Frame are:

- Creating a Data Frame
- Dealing with Rows and Columns
- Indexing and Selecting Data
- Working with Missing Data

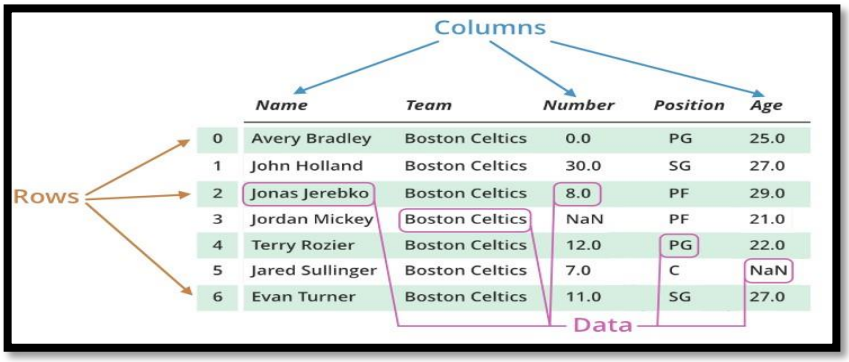


Fig 2.2 Pandas Data Frame components example

```
#Python program using Pandas for arranging a given set of data into a table
import pandas as pd
data = {"country": ["Brazil", "Russia", "India", "China", "South Africa"],
        "capital": ["Brasilia", "Moscow", "New Delhi", "Beijing", "Pretoria"],
        "area": [8.516, 17.10, 3.286, 9.597, 1.221],
        "population": [200.4, 143.5, 1252, 1357, 52.98] }
data table = pd. DataFrame(data)
print(data table)
```

Matplotlib

Matplotlib is a very popular Python library for data visualization. Like Pandas, it is not directly related to Machine Learning. It particularly comes in handy when a programmer wants to visualize the patterns in the data. It is a 2D plotting library used for creating 2D graphs and plots. A module named pyplot makes it easy for programmers for plotting as it provides features to control line styles, font properties, formatting axes, etc. It provides various kinds of graphs and plots for data visualization, viz., histogram, error charts, bar charts, etc,

```
#Python program using Matplotlib for forming a linear plot
import matplotlib. Pyplot as plt # importing the necessary packages and modules
import NumPy as np
x = np. linspace (0, 10, 100) # Prepare the data
plt. Plot (x, x, label ='linear') # Plot the data
plt. Legend() # Add a legend
plt. Show() # Show the plot
```

Software tool used

Anaconda is a free and open-source distribution of the Python programming language for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. Package versions are managed by the package management system conda. Anaconda distribution comes with more than 1,500 packages as well as the conda package and virtual environment manager. It also includes a GUI, Anaconda Navigator, as a graphical alternative to the command line interface (CLI). Anaconda Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. There are many applications available by default in navigator; among them is the Spyder.

Spyder is an open source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates with a number of prominent packages. Some of the features of Spyder are:

- An editor with syntax highlighting, introspection, code completion.
- Support for multiple I Python consoles.
- The ability to explore and edit variables from a GUI.
- A Help pane able to retrieve and render rich text documentation on functions, classes and methods automatically or on-demand.
- A debugger linked to IP dB, for step-by-step execution.
- A run-time Profiler, to benchmark code.
- Project support, allowing work on multiple development efforts simultaneously.
- A built-in file explorer, for interacting with the file system and managing projects.
- A "Find in Files" feature, allowing full regular expression search over a specified scope.
- An online help browser, allowing users to search and view Python and package documentation inside the IDE.
- A history log, recording every user command entered in each console.
- An internal console, allowing for introspection and control over Spyder's own operation.

Open CV

OpenCV is a huge open-source library for computer vision, machine learning, and image processing. OpenCV supports a wide variety of programming languages like Python, C++, Java, etc. It can process images and videos to identify objects, faces, or even the handwriting of a human. When it is integrated with various libraries, such as Numpy which is a highly optimized library for numerical operations, then the number of weapons increases in your Arsenal i.e., whatever operations one can do in Numpy can be combined with OpenCV.

Definition

OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

Topics Covered

- **Converting Image to Grey scale and HSV scale**

  Here we convert the given image into a grey scale image i.e., Black and white image. The function used here is cvtcolor and the code used for grey scale is 'COLOR_BGR2GRAY'. Similarly, we do for HSV scale and the function used is 'COLOR_BGR2HSV'. Following is an example for the above scales.

CODE:

```
import cv2
path=" C:\\Users\\Admin\\Desktop\\data\\Images\\cats.jpg"
img = cv2.imread(path)
cv2.imshow("Original Image", img)
img1= cv2.resize(img,(512,512))
cv2.imshow("Resized Image",img1)
grayimg = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
grayimg= cv2.resize(grayimg,(512,512))
cv2.imshow("Gray Image", grayimg)
havimg = cv2.cvtColor(img,cv2.COLOR_BGR2HSV)
```

```
havimg= cv2.resize(havimg,(512,512))
cv2.imshow(" HSV Image", havimg)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

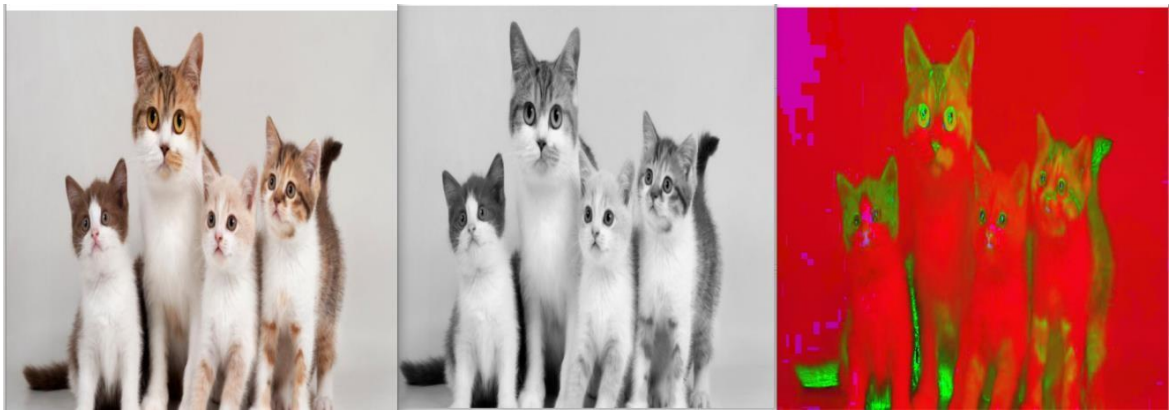


Fig 2.3 Example of Grey and HSV Image

- **Resizing Image**

Here we can resize the original image into whichever size required by us. The function used for resizing is ‘resize’. For further applications we require resizing of images in order to match them for further uses. This is particularly useful for cascading more than two images.

CODE:

```
import cv2
path = "C:\\Users\\Admin\\Desktop\\data\\Images\\Image.png"
img = cv2.imread(path)
cv2.imshow("Original  Image", img)
img1= cv2.resize(img,(512,512))
cv2.imshow("Resized Image",img1)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

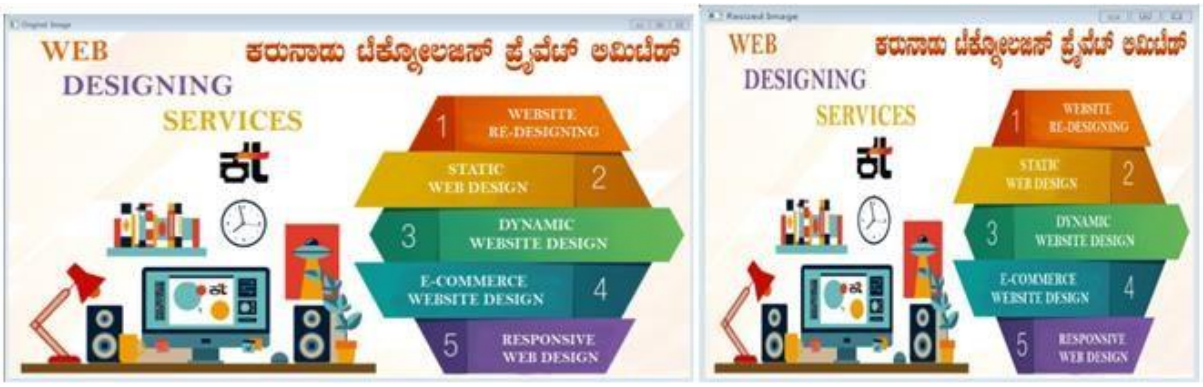


Fig 2.4 Example of Resized Image

- **Rotating Images**

Here we rotate the images in clockwise or anticlockwise direction in the required degree. The function we use ‘rotate’ in order to rotate the images.

CODE:

```
import cv2
    path="C:\\Users\\Admin\\Desktop\\data\\Images\\cats.jpg"
    img = cv2.imread(path)
    imgresize = cv2.resize(img, (300,300))
    img90 = cv2.rotate(imgresize, cv2.ROTATE_90_CLOCKWISE)
    img180 = cv2.rotate(imgresize, cv2.ROTATE_180)
    img270 = cv2.rotate(imgresize, cv2.ROTATE_90_COUNTERCLOCKWISE)
    cv2.imshow("Original Image”, img)
    cv2.imshow("90degree Image”, img90)
    cv2.imshow("180degree Image”, img180)
    cv2.imshow("270degree Image”, img270)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
```

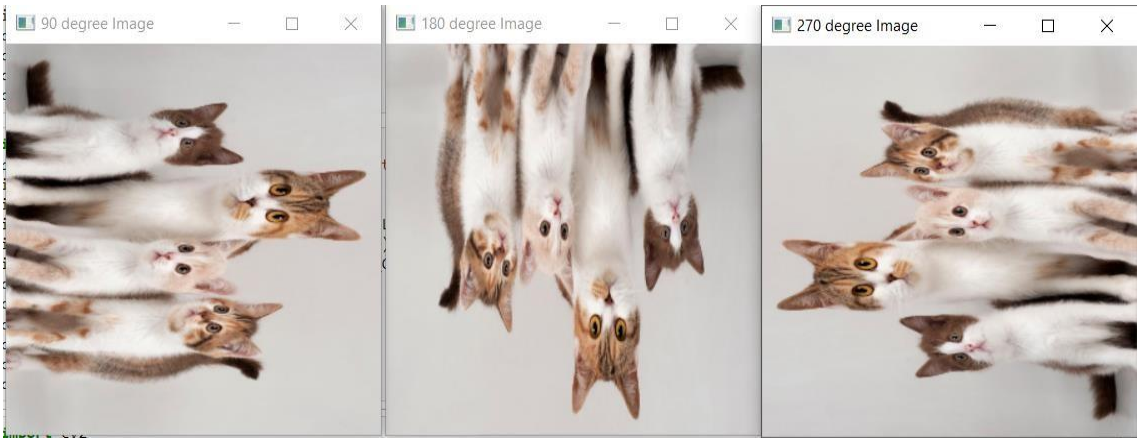


Fig 2.5 Example of Rotated Image

Machine Learning

Machine learning (ML) is a type of artificial intelligence (AI) that allows software applications to become more accurate at predicting outcomes without being explicitly programmed to do so. Machine learning algorithms use historical data as input to predict new output values.

Definition: Ability of a machine to improve its own performance through the use of software that employs artificial intelligence techniques to mimic the ways by which humans seem to learn, such as repetition and experience.

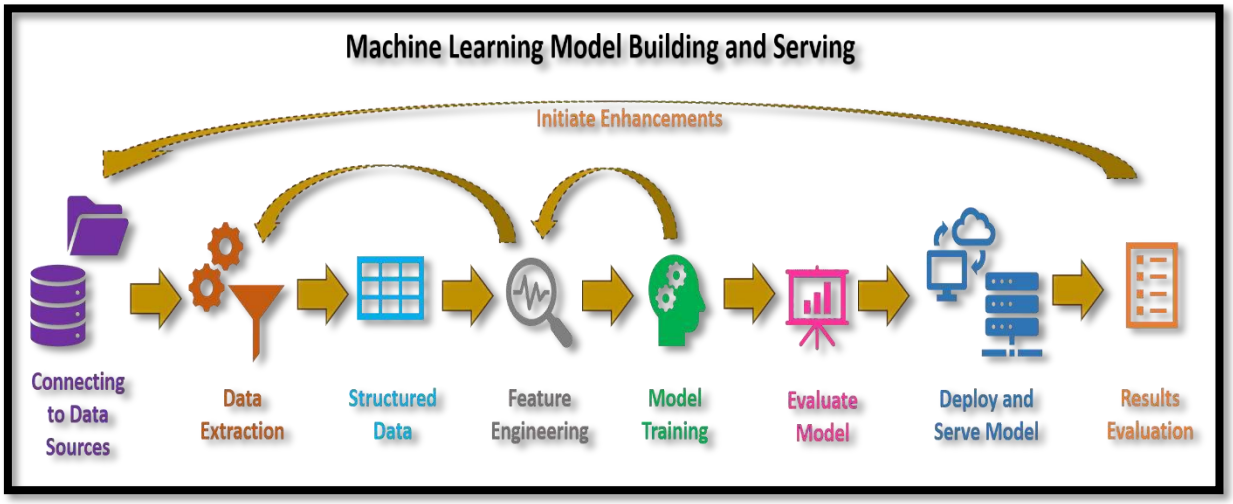


Fig 2.6 Machine Learning Infrastructure

Machine Learning (ML) is a sub-field of Artificial Intelligence (AI) which concerns with developing computational theories of learning and building learning machines. This sequence of learning and building is as shown in fig 2.9. The goal of machine learning, closely coupled with the goal of AI, is to achieve a thorough understanding about the nature of learning process (both human learning and other forms of learning), about the computational aspects of learning behaviours, and to implant the learning capability in computer systems.

The Goals of Machine Learning

The goal of ML, in simple words, is to understand the nature of human and other forms of learning, and to build learning capability in computers. To be more specific, there are three aspects of the goals of ML.

- To make the computers smarter, more intelligent. The more direct objective in this aspect is to develop systems (programs) for specific practical learning tasks in application domains.
- To develop computational models of human learning process and perform computer simulations. The study in this aspect is also called cognitive modelling.
- To explore new learning methods and develop general learning algorithms independent of applications.

Types of Machine Learning

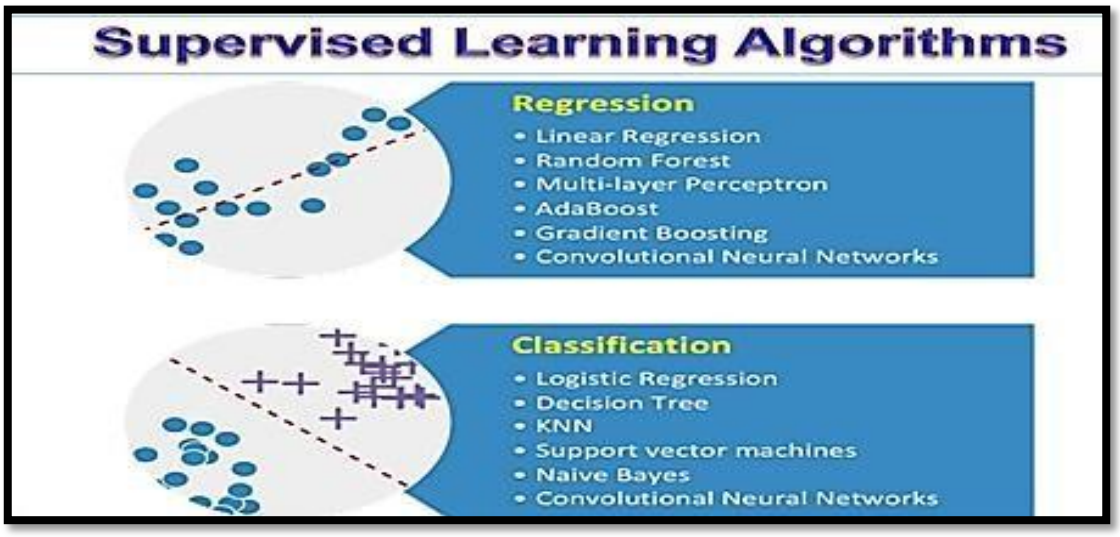
➢ Supervised learning

Supervised learning is a type of machine learning method in which we provide sample labelled data to the machine learning system in order to train it, and on that basis, it predicts the output. The system creates a model using labelled data to understand the datasets and learn about each data, once the training and processing are done then we test the model by providing a sample data to check whether it is predicting the exact output or not.

The goal of supervised learning is to map input data with the output data. The supervised learning is based on supervision, and it is the same as when a student learns things in the supervision of the teacher. The example of supervised learning is **spam filtering**.

Supervised learning can be grouped further in two categories of algorithms:

- **Classification:** A classification problem is when the output variable is a category, such as "Red" or "blue" or "disease" and "no disease".
- **Regression:** A regression problem is when the output variable is a real value, such as "dollars" or "weight".



2.7 Flow Model of Algorithms of Supervised Learning.

➢ Unsupervised Learning

Unsupervised learning is the training of machine using information that is neither classified nor labelled and allowing the algorithm to act on that information without guidance. Here the task of 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, machine is restricted to find the hidden structure in unlabelled data by our-self.

For instance, suppose it is given an image having both dogs and cats which have not seen ever. Thus, machine has no any idea about the features of dogs and cat so we can't categorize it in dogs and cats. But it can categorize them according to their similarities, patterns and differences i.e., one can easily categorize the above picture into two parts. First may contain all pics having dogs in it and second part may contain all pics having cats in it.

Unsupervised learning classified into two categories of algorithms:

- **Clustering:** A clustering problem is where one wants to discover the inherent groupings in the data as shown in fig 2.11, such as grouping customers by purchasing behaviour.
- **Association:** An association rule learning problem is where one wants to discover rules that describe large portions of the data, such as people that buy X also tend to buy Y.

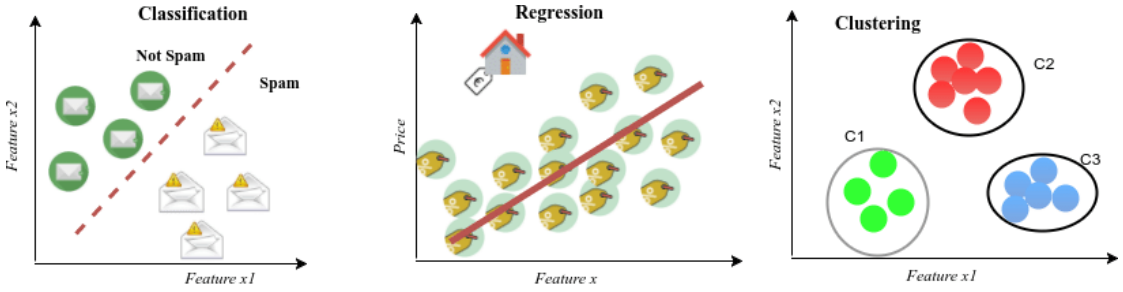
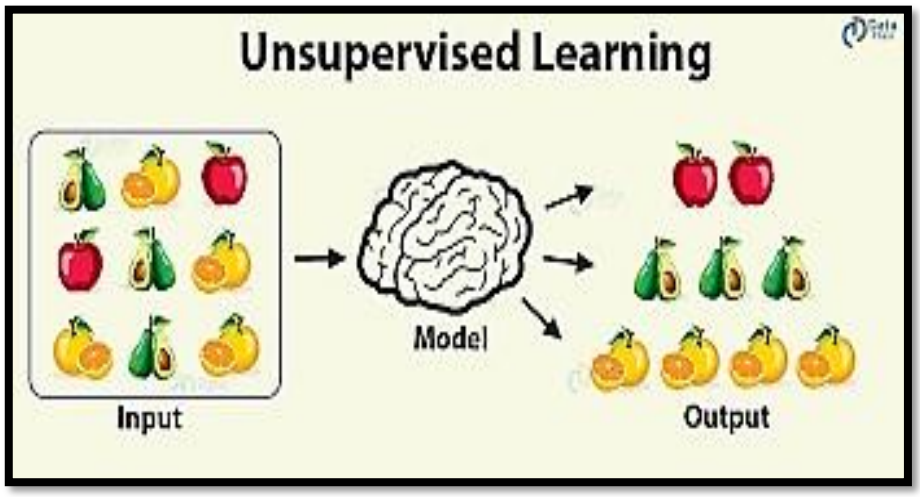


Fig 2.8 Clustering and Regression



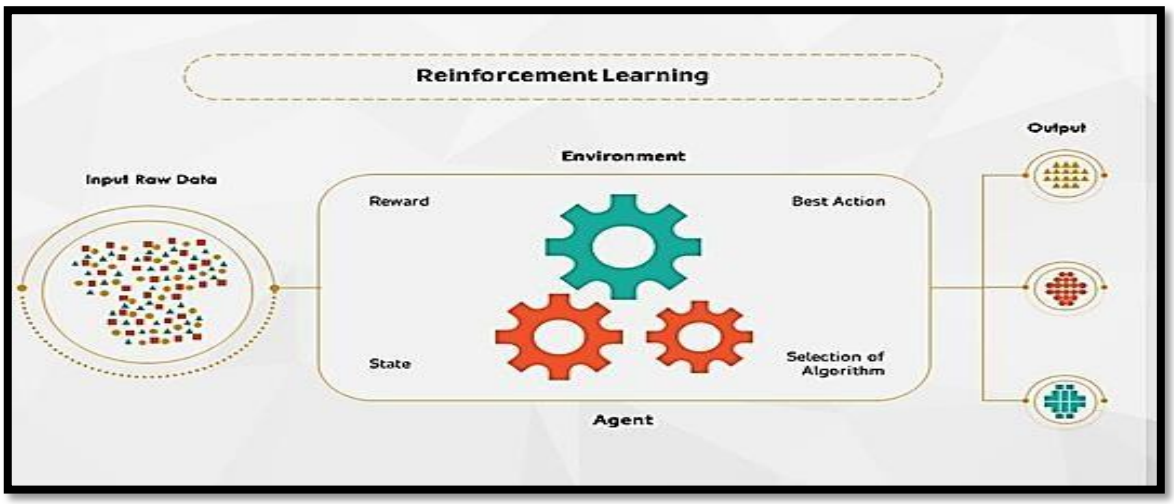
Flow Model of Algorithms of Unsupervised Learning.

➢ Reinforcement learning

Reinforcement learning is an area of Machine Learning. It is about taking suitable action to maximize reward in a particular situation. It is employed by various software and machines to find the best possible behaviour or path it should take in a specific situation. Reinforcement learning differs from the supervised learning in a way that in supervised learning the training data has the answer key with it so the model is trained with the correct answer itself whereas in reinforcement learning, there is no answer but the reinforcement agent decides what to do to perform the given task. In the absence of training dataset, it is bound to learn from its experience.

There are two types of Reinforcement learning:

- **Positive:** Positive Reinforcement is defined as when an event, occurs due to a particular behaviour, increases the strength and the frequency of the behaviour. In other words, it has a positive effect on the behaviour. It maximizes performance and sustains change for a long period of time. Too much Reinforcement can lead to overload of states which can diminish the results.
- **Negative:** Negative Reinforcement is defined as strengthening of behaviour because a negative condition is stopped or avoided. It increases behaviour and provide defiance to minimum standard of performance. It only provides enough to meet up the minimum behaviour.



Flow Model of Algorithms of Reinforcement Learning.

Applications of Machine Learning

Machine learning has been recognized as central to the success of Artificial Intelligence, and it has applications in various areas of science, engineering and society. Some of them are:

- Product recommendations (e.g., Amazon etc.)
- Refining the search engine results (e.g., Google)
- Fighting the web spam (e.g., Gmail)
- Video surveillance (e.g., crime alerts)
- Face recognition and many more.

CHAPTER 3

Machine Learning Algorithms

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.

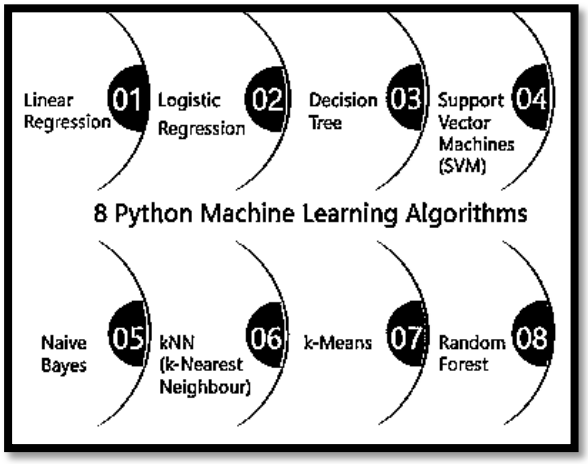


Fig 3.1 Types of Machine Learning Algorithms

Linear Regression

It is one of the most widely known modeling technique. Linear regression is usually among the first few topics which people pick while learning predictive modelling. In this technique, the dependent variable is continuous, independent variable(s) can be continuous or discrete, and nature of regression line is linear. Linear Regression establishes a relationship between dependent variable (Y) and one or more independent variables(X) using a best fit straight line (also known as regression line) as shown in fig 3.2.It is represented by an equation Y=a+b*X + e, where a is intercept, b is slope of the line and e is error term. This equation can be used to predict the value of target variable based on given predictor variable.

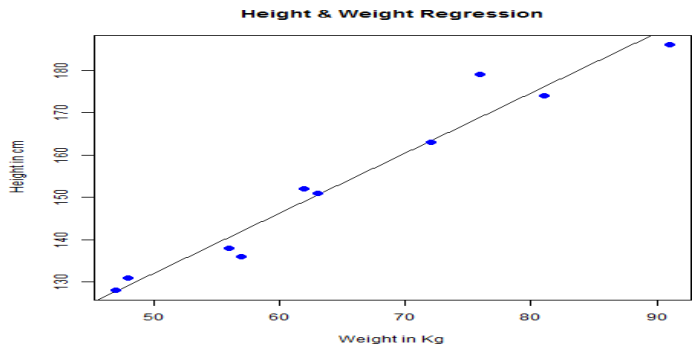
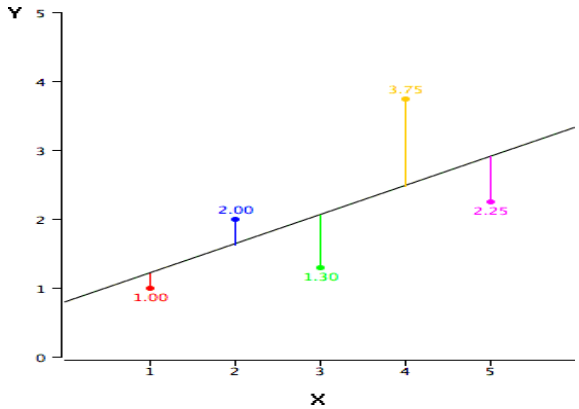


Fig 3.2 Linear Regression

Obtaining best fit line (Value of a and b)

This task can be easily accomplished by Least Square Method. It is the most common method used for fitting a regression line. It calculates the best-fit line for the observed data by minimizing the sum of the squares of the vertical deviations from each data point to the line. Because the deviations are first squared, when added, there is no cancelling out between positive and negative values.



$$\min_{w} ||Xw - y||_2^2$$

Fig 3.3 Plot of Input Graph

There must be linear relationship between independent and dependent variables. Linear Regression is very sensitive to Outliers. It can terribly affect the regression line and eventually the forecasted values. Simple linear regression is used for finding the relationship between the dependent variable Y and the independent or predictor variable X. Both of these variables are continuous in nature. While performing simple linear regression, we assume that the values of predictor variable X are controlled. Furthermore, they are not subject to the measurement error from which the corresponding value of Y is observed.

The equation of a simple linear regression model to calculate the value of the dependent variable, Y based on the predictor X is as follows:

$$y_i = b_0 + b_1 x + e_i$$

Where the value of y_i is calculated with the input variable xi for every ith data point. The coefficients of regressions are denoted by b_0 and b_1 ; the ith value of x has $e_i$as its error in the measurement.

Regression analysis is implemented to do the following:

- With it, we can establish a linear relationship between the independent and the dependent variables.
- The input variables x1, x2…. xn is responsible for predicting the value of y.
- In order to explain the dependent variable precisely, we need to identify the independent variables carefully. This will allow us to establish a more accurate causal relationship between these two variables.

Advantages and Disadvantages

Advantages

Linear regression is an extremely simple method. It is very easy and intuitive to use and understand. A person with only the knowledge of high school mathematics can understand and use it. In addition, it works in most of the cases. Even when it doesn't fit the data exactly, we can use it to find the nature of the relationship between the two variables.

Disadvantages

- By its definition, linear regression only models relationships between dependent and independent variables that are linear. It assumes there is a straight-line relationship between them which is incorrect sometimes. Linear regression is very sensitive to the anomalies in the data (or outliers).
- Take for example most of your data lies in the range 0-10. If due to any reason only one of the data items comes out of the range, say for example 15, this significantly influences the regression coefficients.
- Another disadvantage is that if we have a number of parameters than the number of samples available then the model starts to model the noise rather than the relationship between the variables.

```
def LINEAR(self):
    Area=self.area.text()
    import pandas as pd
    path="C:\\Users\\User\\Desktop\\DATA\\homeprices.csv"
    data=pd.read_csv(path)
    print(data)
    print(data.info())

    inputs=data.drop('price','columns')
    output=data.drop(['area'],'columns')
    print(inputs)
    print(output)
    import sklearn
    from sklearn import linear_model
    model=linear_model.LinearRegression()
    model.fit(inputs,output)
    res=model.predict([[Area]])
    print(res)
    self.result.setText(str(res))
```

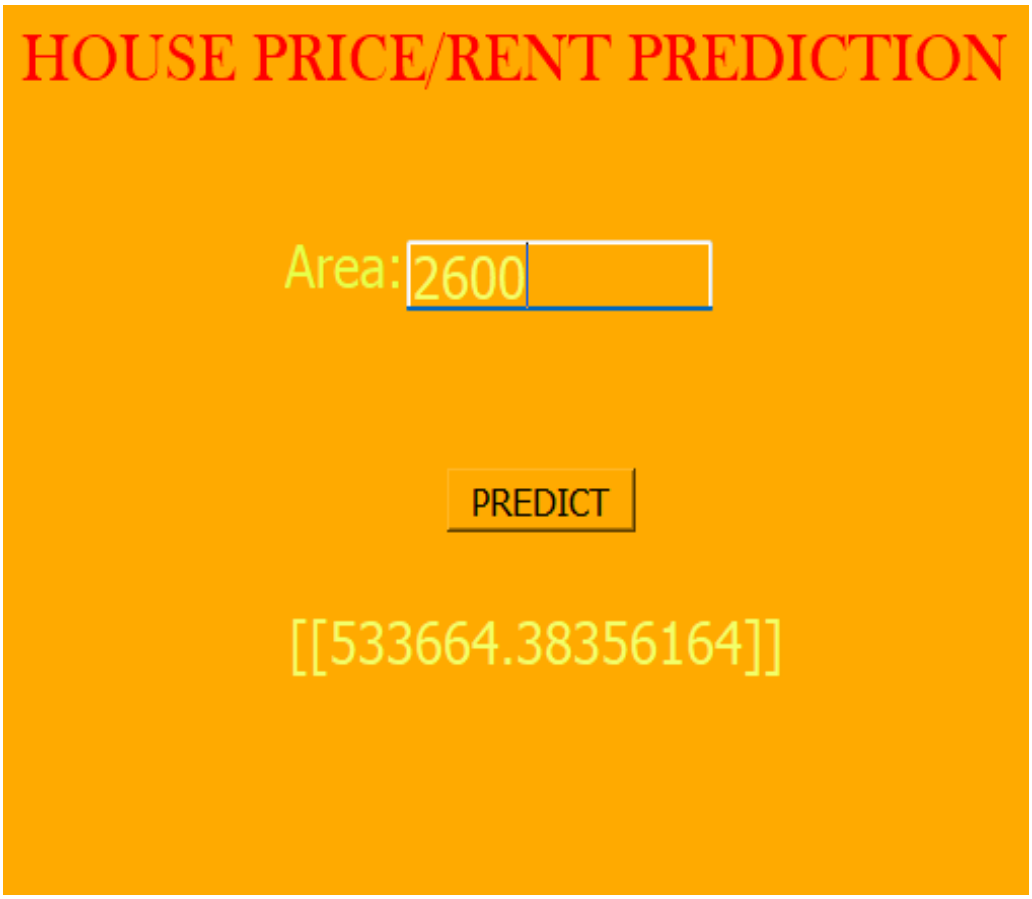


Fig 3.4 Example of Linear Regression

Multiple Linear Regression

In many cases, there may be possibilities of dealing with more than one predictor variable for finding out the value of the response variable. Therefore, the simple linear models cannot be utilized as there is a need for undertaking multiple linear regression for analyzing the predictor variables. The difference between simple linear regression and multiple linear regression is that, multiple linear regression has more than 1 independent variables, whereas simple linear regression has only 1 independent variable. Using the two explanatory variables, we can delineate the equation of multiple linear regression as follows:

$$y_i = b_0 + b_1 x_1 + b x_2 + e_i$$

The two explanatory variables x_1 and x_2 , determine y_i , for the ith data point. Furthermore, the predictor variables are also determined by the three parametersb_0, b_1, and b_2 of the model, and by the residual e_i of the point i from the fitted surface. General Multiple regression models can be represented as:

$$y_i = \sum b_i x_i + e_i$$

Multiple regression suffers from multicollinearity, autocorrelation, heteroskedasticity**.** Multicollinearity can increase the variance of the coefficient estimates and make the estimates very sensitive to minor changes in the model. The result is that the coefficient estimates are unstable. In case of multiple independent variables, we can go with forward selection**,** backward elimination and step wise approach for selection of most significant independent variables.

Advantages and disadvantages

Advantages

- The ability to determine the relative influence of one or more predictor variables to the criterion value.
- The ability to identify outliers, or anomalies.

Disadvantage

- Any disadvantage of using a multiple regression model usually comes down to the data being used. Two examples of this are using incomplete data and falsely concluding that a correlation is causation.

```
def MULTILIN(self):
    import pandas as pd
    path="C:\\Users\\User\\Desktop\\DATA\\homeprices_Mul.csv"
    data=pd.read_csv(path)
    print(data,data.info())
    medianval=data.bedrooms.median()
    print(medianval)
    medianval=data.bedrooms.median()
    data.bedrooms=data.bedrooms.fillna(medianval)
    print(data)
    inputs=data.drop('price','columns')
    output=data.drop(['area','bedrooms','age'],'columns')
    print(inputs,output)
    import sklearn
    from sklearn import linear_model
    model=linear_model.LinearRegression()
    model.fit(inputs,output)
    Area=int(input())
    Bedrooms=float(input())
    Age=int(input())
    res=model.predict([[Area,Bedrooms,Age]])
    print(res)
    m=model.coef_        #y=m1*x1+m2*+m2m3*+m3*x3+b
    b=model.intercept_
    print(m,b)
    y=(112.06244194*Area) +(23388.88007794*Bedrooms)+ (-3231.71790863*Age)+221323.0018654
    print(y)
    self.result.setText(float(res))
```

Fig 3.5 Example of Multiple Linear Regression

Logistic Regression

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression is estimating the parameters of a logistic model (a form of binary regression). Mathematically, a binary logistic model has a dependent variable with two possible values, such as pass/fail which is represented by an indicator variable, where the two values are labelled "0" and "1". In the logistic model, the log-odds (the logarithm of the odds) for the value labelled "1" is a linear combination of one or more independent variables ("predictors"); the independent variables can each be a binary variable (two classes, coded by an indicator variable) or a continuous variable (any real value). The corresponding probability of the value labelled "1" can vary between 0 (certainly the value "0") and 1 (certainly the value "1"), hence the labelling, the function that converts log-odds to probability is the logistic function, hence the name. The unit of measurement for the log-odds scale is called a logit, from logistic unit, hence the alternative names. Analogous models with a different sigmoid function instead of the logistic function can also be used, such as the profit model; the defining characteristic of the logistic model is that increasing one of the independent variables multiplicatively scales the odds of the given outcome at a constant rate, with each independent variable having its own parameter; for a binary dependent variable this generalizes the odds ratio.

Key Features

- Logistic regression predicts whether something is True (1) or False (0) instead, predicting something that is continuous like size.
- It has an S-shaped line.
- We can take our Linear Regression Model and convert it into Logistic Regression model with the help of Sigmoid Function.
- Logistic Regression’s ability to provide probabilities and classify new samples using continuous and discrete measurements makes ita popular machine learning method.

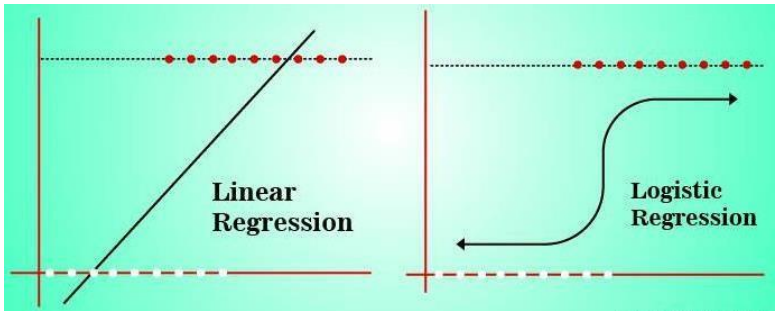


Fig 3.6 Linear Regression v/s Logistic Regression

This is where logistic regression comes into play. In logistic regression, you get a probability score that reflects the probability of the occurrence of the event. An event in this case is each row of the training dataset. It could be something like classifying if a given email is spam, or mass of cell is malignant or a user will buy a product and so on.

Advantages and Disadvantages

Advantages

- It doesn't require high computational power.
- Is easily interpretable.
- Is used widely by the data analyst and data scientists.
- Is very easy to implement.
- It doesn't require scaling of features.
- It provides a probability score for observations.

Disadvantages

- While working with Logistic regression you are not able to handle a large number of categorical features/variables.
- It is vulnerable to over fitting.
- It can't solve the non-linear problem with the logistic regression model that is why it requires a transformation of non-linear features.
- Logistic regression will not perform well with independent(X) variables that are not correlated to the target(Y) variable.

```
def magzine(self):
    import pandas as pd
    import sklearn
    from sklearn.model_selection import train_test_split
    from sklearn.metrics import confusion_matrix
    from sklearn.svm import SVC
    from sklearn.preprocessing import StandardScaler

    path="C:\\Users\\User\\Desktop\\ DATA\Logistic\\Magzine\\Kid.csv"
    data=pd.read_csv(path)
    print(data)

    inputs=data.drop(['Buy','Obs No.'],'columns')
    output=data['Buy']
    print(inputs,output)

    x_train,x_test,y_train,y_test=train_test_split(inputs,output,test_size=0.2)

    model=SVC()  #initialize my model

    model.fit(x_train,y_train) #train the model

    y_pred=model.predict(x_test)
    print(y_pred)

    ans=model.predict(([[Income,female,married,college,prof,Ret,Unemp,res,dual,mine,own,house,white,eng,child,perent]]))
    print(ans)
    if ans==0:
        print("YES")
    elif ans=="1":
        print("NO")

    cm=confusion_matrix(y_test,y_pred)
    print(cm)

    self.result.setText(str(ans))
```

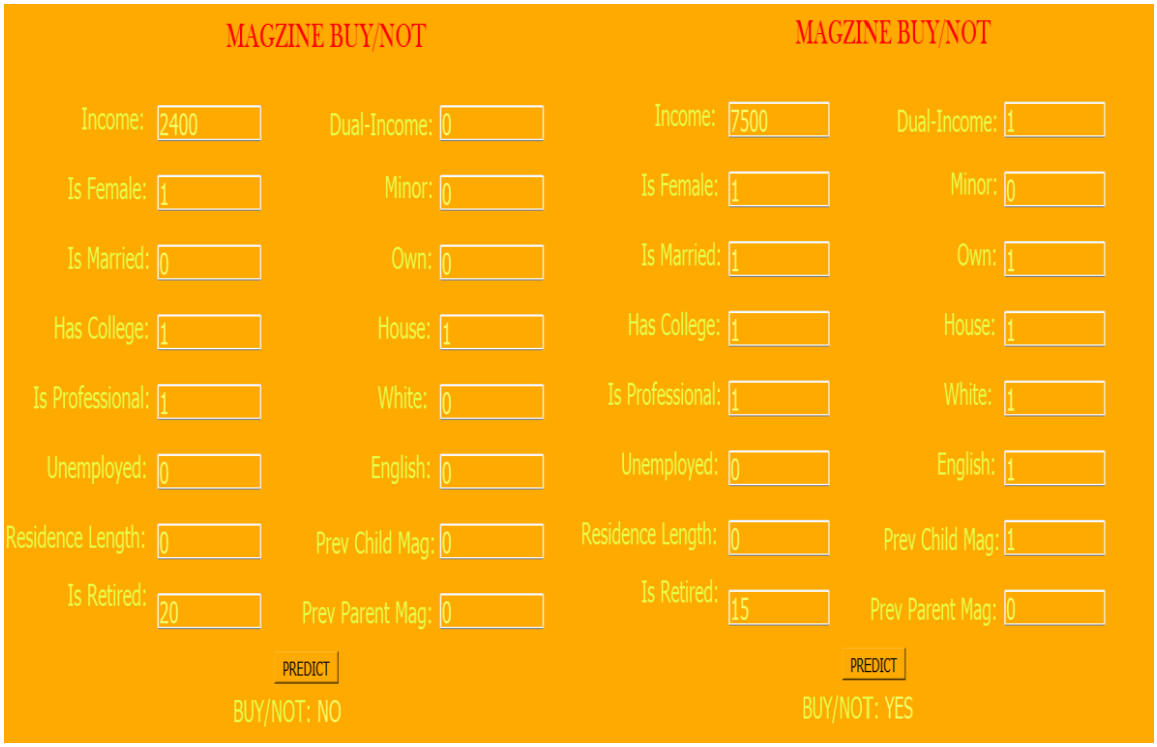


Fig 3.7 Example of Logistic Regression

KNN

K nearest neighbours or KNN Algorithm is a simple algorithm which uses the entire dataset in its training phase. Whenever a prediction is required for an unseen data instance, it searches through the entire training dataset for k-most similar instances and the data with the most similar instance is finally returned as the prediction. KNN is often used in search applications where you are looking for similar items, like find items similar to this one.

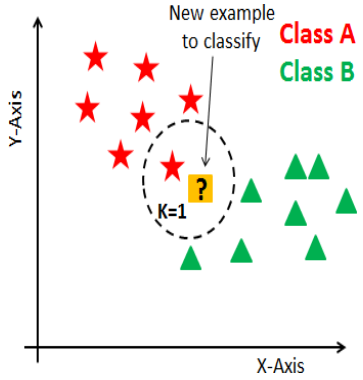
Features of KNN Algorithm

- KNN is a Supervised Learning algorithm that uses labelled input data set to predict the output of the data points.
- It is one of the simplest Machine learning algorithms and it can be easily implemented for a varied set of problems.
- It is mainly based on feature similarity. KNN checks how similar a data point is to its neighbours and classifies the data point into the class it is most similar to.
- Unlike most algorithms, KNN is a non-parametric model which means that it does not make any assumptions about the data set. This makes the algorithm more effective since it can handle realistic data.
- KNN is a lazy algorithm; this means that it memorizes the training data set instead of learning a discriminative function from the training data.
- KNN can be used for solving both classification and regression problems.

Working

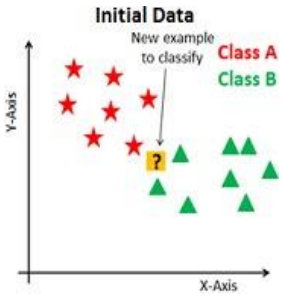
In KNN, K is the number of nearest neighbours. The number of neighbours is the core deciding factor. K is generally an odd number if the number of classes is 2. When K=1, then the algorithm is known as the nearest neighbour algorithm. This is the simplest case. However, the number of neighbours (K) is a hyper parameter that needs to be chosen at the time of model building. Research has shown that no optimal number of neighbours suits all kind of data sets. Each dataset has its own requirements. Generally, Data scientists choose as an odd number if the number of classes is even. We can also check by generating the model on different values of k and check their performance.

Suppose '?' is the point, for which label needs to predict. First, you find the k closest point to P1 and then classify points by majority vote of its k neighbours. Each object votes for their class and the class with the most votes are taken as the prediction. For finding closest similar points, you find the distance between points using distance measures such as Euclidean distance, Hamming distance, Manhattan distance and Minkowski distance. Then we find the one closest point to '?' and then the label of the nearest point is assigned to '?'.

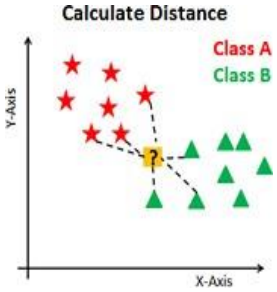


KNN has the following basic steps:

1. Calculate distance



2. Find closest neighbours



3. Vote for labels

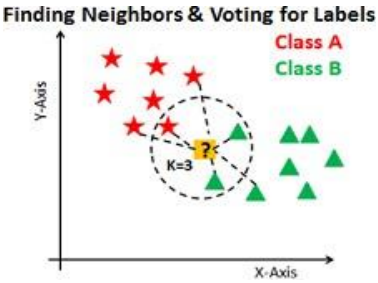


Fig 3.8 Plot of ideal KNN Algorithm

Advantages and Disadvantages

Advantages

- The algorithm is simple and easy to implement.
- There's no need to build a model, tune several parameters, or make additional assumptions.
- The algorithm is versatile. It can be used for classification, regression, and search
- The training phase of K-nearest neighbour classification is much faster compared to other classification algorithms. There is no need to train a model for generalization that is why KNN is known as the simple and instance-based learning algorithm.
- KNN can be useful in case of nonlinear data. It can be used with the regression problem. Output value for the object is computed by the average of k closest neighbours value.

Disadvantages

- The algorithm gets significantly slower as the number of examples and/or predictors/independent variables increase.
- The testing phase of K-nearest neighbour classification is slower and costlier in terms of time and memory. It requires large memory for storing the entire training dataset for prediction.
- KNN requires scaling of data because KNN uses the Euclidean distance between two data points to find nearest neighbours. Euclidean distance is sensitive to magnitudes. The features with high magnitudes will weigh more than features with low magnitudes.
- KNN also not suitable for large dimensional data.

```
def flower(self):
    import pandas as pd
    path="C:\\Users\\User\\Desktop\\DATA\\Iris.csv"
    data=pd.read_csv(path)
    print(data)
    print(data.info())


    inputs=data.drop(['Id','Species'],'columns')
    output=data.drop(['Id','SepalLengthCm','SepalWidthCm', 'PetalLengthCm','PetalWidthCm'],'columns')
    print(inputs,output)


    import sklearn
    from sklearn.model_selection import train_test_split
    x_train,x_test,y_train,y_test=train_test_split(inputs,output,train_size=0.8)
    print(x_train,x_test,y_train,y_test)

    from sklearn.neighbors import KNeighborsClassifier
    model=KNeighborsClassifier(n_neighbors=13)
    model.fit(x_train,y_train)
    y_pred=model.predict(x_test)
    print(y_pred,y_test)

    from sklearn.metrics import confusion_matrix
    cm=confusion_matrix(y_test,y_pred)
    print(cm)
    res=model.predict([[6.3,3.3,6.0,2.5]])
    print(res)

    self.result.setText(str(res))
```

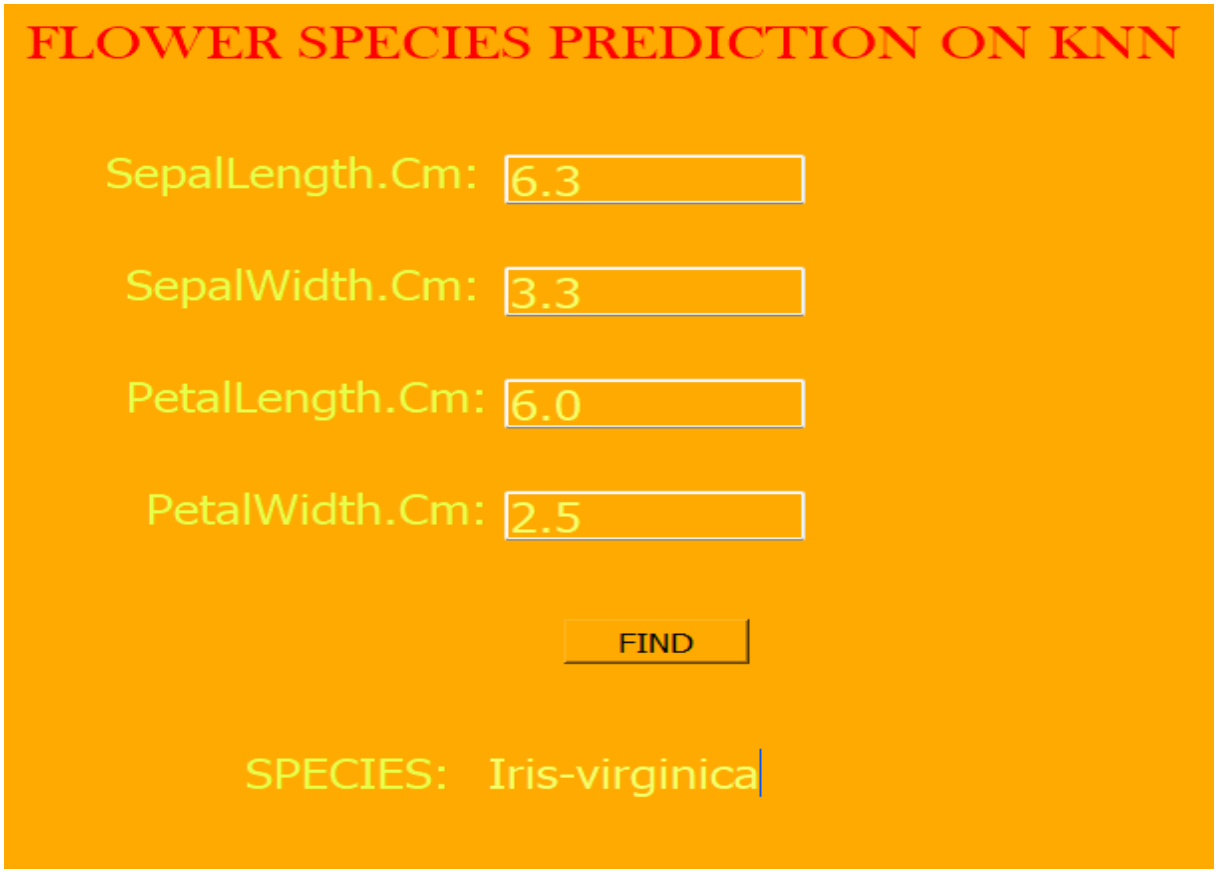


Fig 3.9 Example of KNN Algorithm

SVM

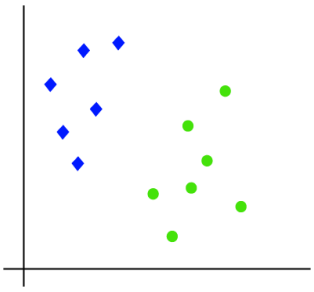
"Support Vector Machine" (SVM) is a supervised machine learning algorithm that can be used for both classification and regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is a number of features you have) with the value of each feature being the value of a particular coordinate.

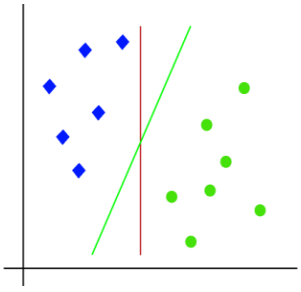
Features of SVM

- SVM is a Supervised Learning algorithm that uses labelled input data set to predict the output of the data points.
- It is one of the simplest Machine learning algorithms and it can be easily implemented for a varied set of problems.
- SVM can be used for solving both classification and regression problems.

Working

The working of the SVM algorithm can be understood by using an example. Suppose we have a dataset that has two tags (green and blue), and the dataset has two features x1 and x2. We want a classifier that can classify the pair(x1, x2) of coordinates in either green or blue. So, as it is 2-d space so by just using a straight line, we can easily separate these two classes. But there can be multiple lines that can separate these classes. Consider the below image:





Hence, the SVM algorithm helps to find the best line or decision boundary; this best boundary or region is called as a **hyperplane**. SVM algorithm finds the closest point of the lines from both the classes. These points are called support vectors. The distance between the vectors and the hyperplane is called as **margin**. And the goal of SVM is to maximize this margin. The **hyperplane** with maximum margin is called the **optimal hyperplane**.

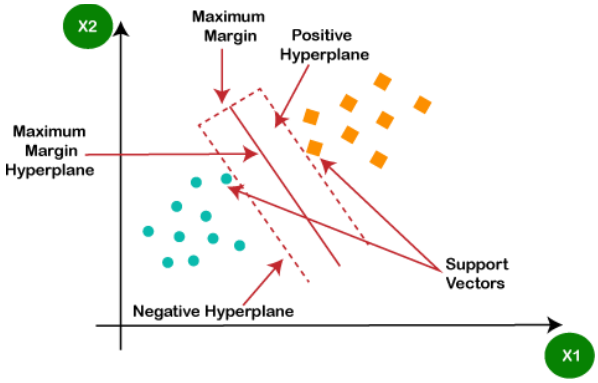


Fig 3.10 Plot of ideal SVM Algorithm

Advantagesand Disadvantages

Advantages

- SVM works relatively well when there is a clear margin of separation between classes.
- SVM is more effective in high dimensional spaces.
- SVM is effective in cases where the number of dimensions is greater than the number of samples.
- SVM is relatively memory efficient.

Disadvantages

- SVM algorithm is not suitable for large data sets.
- SVM does not perform very well when the data set has more noise i.e. target classes are overlapping.
- In cases where the number of features for each data point exceeds the number of training data samples, the SVM will underperform.
- As the support vector classifier works by putting data points, above and below the classifying hyperplane there is no probabilistic explanation for the classification.

```
def ads(self):
    Gender=self.gender.text()
    Age=self.age.text()
    salary=self.salary.text()
    import pandas as pd
    path="C:\\Users\\User\\Desktop\\DATA\\Social_Network_Ads.csv"
    data=pd.read_csv(path)
    print(data,data.info())

    print(data['Gender'].unique())
    print(data['Gender'].nunique())
    data['Gender']=data['Gender'].map(('Male':1,'Female':0))
    print(data)

    inputs=data.drop(['User ID','purchased'],'columns')
    output=data.drop(['User ID','Gender','Age','EstimatedSalary'],'columns')
    print(inputs,output)

    import sklearn
    from sklearn.model_selection import train-test_split
    x_train,x_test,y_train,y_test=train_test_split(inputs,output,train_size=0.8)
    print(x_test,x_train)
    print(y_test,y_traint)

    from sklearn.preprocessing import standardScaler
    sc=StandardScaler()
    x_train=sc.fit_transform(x_train)
    x_test=sc.fit_transform(x_test)

    from sklearn.svc import svc
    model=SVC()
    model.fit(x_train,x_train)
    y_pred=model.predict(x_test)
    print(y_pred)
    print(y_test)

    from skearn.matrics import confusion_matrics
    cm=confusion_matrix(y_test,y_pred)
    print(cm)

    import numpy as np
    newdata=np.array([[Gender,Age,Salary]])
    newdata=sc.transform(newdata)
    res=model.predict(newdata)
    print(res)
    if res==0:
        self.resut.setText("NO")
    else:
        self.result.setText("YES")
```

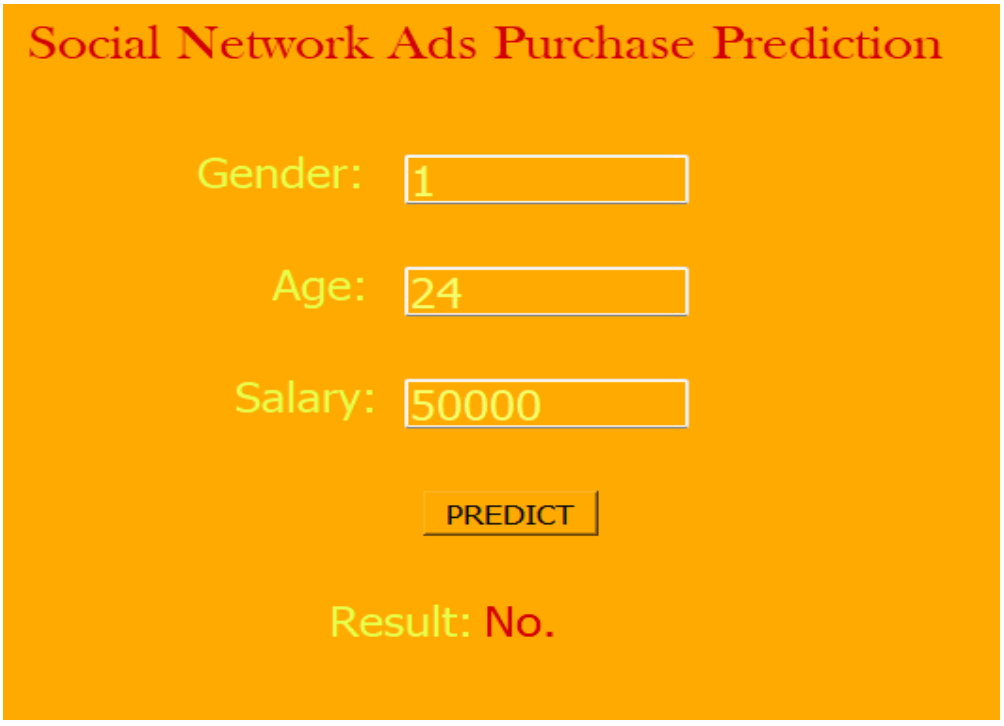


Fig 3.11 Example of SVM Algorithm

Decision Tree

Decision Tree is a **supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.** It is a graphical representation for getting all the possible solutions to a problem/decision on based on given conditions. In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

Features of Decision tree

- Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
- The logic behind the decision tree can be easily understood because it shows a tree-like structure.
- It is very easy to understand and implement.

Working

In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node. For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

Step-1: Begin the tree with the root node, says S, which contains the complete dataset.

Step-2: Find the best attribute in the dataset using **Attribute Selection Measure (ASM).**

Step-3: Divide the S into subsets that contains possible values for the best attributes.

Step-4: Generate the decision tree node, which contains the best attribute.

Step-5: Recursively make new decision trees using the subsets of the dataset created in step3.

Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

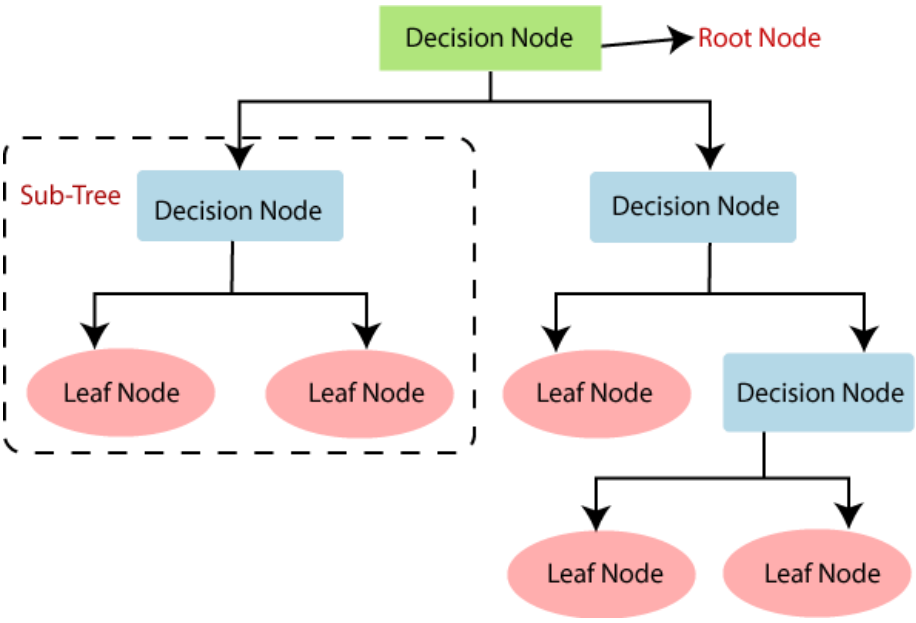


Fig 3.12Ideal diagram of a Decision Tree

Advantages and Disadvantages

Advantages

- It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
- It can be very useful for solving decision-related problems.
- It helps to think about all the possible outcomes for a problem.
- There is less requirement of data cleaning compared to other algorithms.

Disadvantages

- The decision tree contains lots of layers, which makes it complex.
- It may have an over fitting issue, which can be resolved using the **Random Forest algorithm.**
- For more class labels, the computational complexity of the decision tree may increase.

```
def morethan(self):
    Comapy=self.comany.text()
    Job=self.job.text()
    Degree=self.degree.text()
    import pandas as pd
    path="C:\\Users\\User\\Desktop\\DATA\\salaries.csv"
    data=pd.read_csv(path)
    print(data,data.info())

    import sklearn
    from sklearn.preprocessing import LabelEncoder
    le_compony=LabelEncoder()
    le_job=LabelEncoder()
    le_degree=LabelEncoder()
    print(data['company'].unique())
    data['compony']=le_company.fit_transform(data['company'])
    data['job']=le_job.fit_transform(data['job'])
    data['degree']=le_degree.fit_transform(data['degree'])
    print(data)

    inputs=data.drop('salary_more-than_100k','columns')
    output=data.drop(['company','job','degree'],'columns')
    print(inputs)
    print(output)

    from sklearn import tree
    model=tree.DecisionTreeClissifier()
    model.fit(inputs,output)

    res=model.predict([[Company,Job,Degree]])
    print(res)

    if res==0:
        self.result.setText(str("salary is more than 100k"))
    else:
        self.result.setText(str("salary is less than 100k"))
```

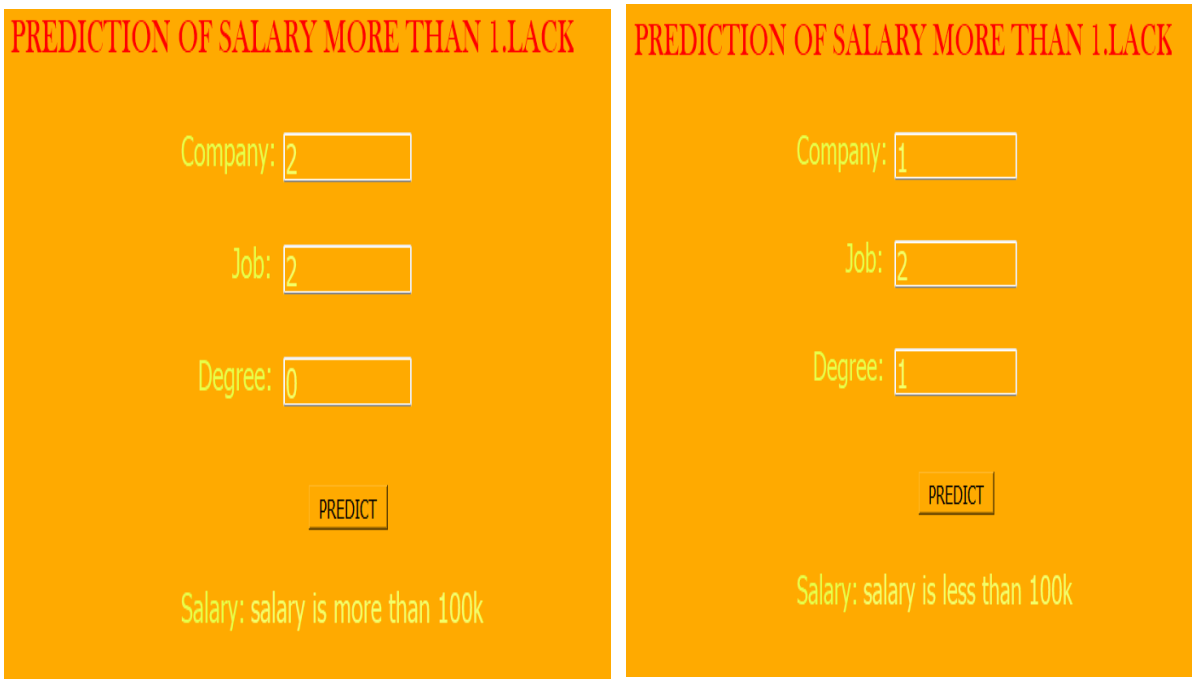


Fig 3.13 Example of Decision Tree

CHAPTER 4

Project - Description

Prediction Of Heart Attack Using KNN

In this project, we will work with csv dataset to develop a machine learning algorithm that predicts the person whether get a chance of heart attack or not . A model like this would be very valuable to predict person is get a chance of heart attack or not using result status.

Problem Statement

Develop a model that has the capacity of predicting Heart Attack by making use of information provided in Heart Attack data set.

Dataset:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
2	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
3	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
4	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
5	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
6	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
7	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1
8	56	0	1	140	294	0	0	153	0	1.3	1	0	2	1
9	44	1	1	120	263	0	1	173	0	0	2	0	3	1
10	52	1	2	172	199	1	1	162	0	0.5	2	0	3	1
11	57	1	2	150	168	0	1	174	0	1.6	2	0	2	1
12	54	1	0	140	239	0	1	160	0	1.2	2	0	2	1
13	48	0	2	130	275	0	1	139	0	0.2	2	0	2	1
14	49	1	1	130	266	0	1	171	0	0.6	2	0	2	1
15	64	1	3	110	211	0	0	144	1	1.8	1	0	2	1
16	58	0	3	150	283	1	0	162	0	1	2	0	2	1
17	50	0	2	120	219	0	1	158	0	1.6	1	0	2	1
18	58	0	2	120	340	0	1	172	0	0	2	0	2	1
19	66	0	3	150	226	0	1	114	0	2.6	0	0	2	1

Fig 4.1 Overview of Dataset

Algorithm –KNN(K nearest neighbor)

- K nearest neighbors or KNN Algorithm is a simple algorithm which uses the entire dataset in its training phase .
- Whenever a prediction is required for an unseen data instance, it searches through the entire training dataset for K-most similar instances and the data with the most similar instance is finally returned as the prediction.
- KNN is often used in search applications where we are looking for similar items, like find items similar to the one.

Programming Steps

- This project requires us to predict the Person is hepatitis or not is based on the given input dataset.
- First, we read the given dataset using pandas function.
- Then we print the inputs and output from csv file.
- Label encoding is used for

  Age,Sex,ALB,ALP,ALT,AST,BIL,CHE,CHOL,CREA,GGT,PROT

- column.
- We initialize the model i.e., **K nearest neighbors**.
- We further implement this using Pyqt in order for better representation.

Code:

```
import pandas as pd
path="C:\\Users\\hp\\Desktop\\Project\\2023_Geckrojects\\KNN\\Heart attack prediction\\heart.csv"
data=pd.read_csv(path)
print(data)
print(data.info())

inputs=data.drop(['output'],'columns')
output=data.drop(['age','sex','cp','trtbps','chol','fbs','restecg','thalachh','exng','oldpeak','slp','caa','thall'],'columns')
print(inputs)
print(output)
```

```
import sklearn
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(inputs,output,train_size=0.8)
print(x_train)
print(x_test)
print(y_train)
print(y_test)

from sklearn.neighbors import KNeighborsClassifier
model=KNeighborsClassifier(n_neighbors=18)
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
print(y_pred)
print(y_test)

from sklearn.metrics import confusion_matrix
cm=confusion_matrix(y_test,y_pred)
print(cm)

res=model.predict([[57,0,0,140,241,0,1,123,1,0.2,1,0,3]])
print(res)

acc=model.score(inputs,output)*100
print(acc,"%")
```

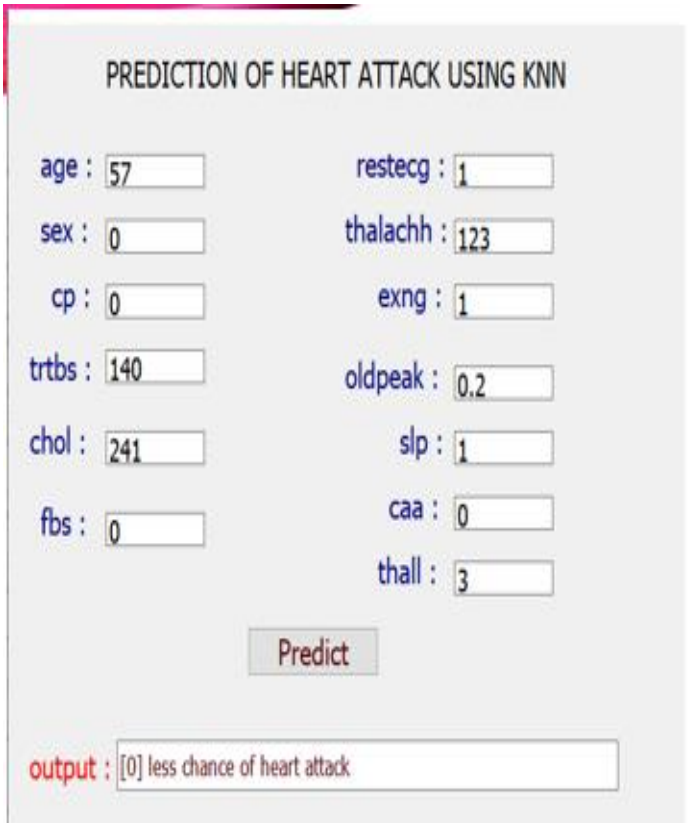
OUTPUT

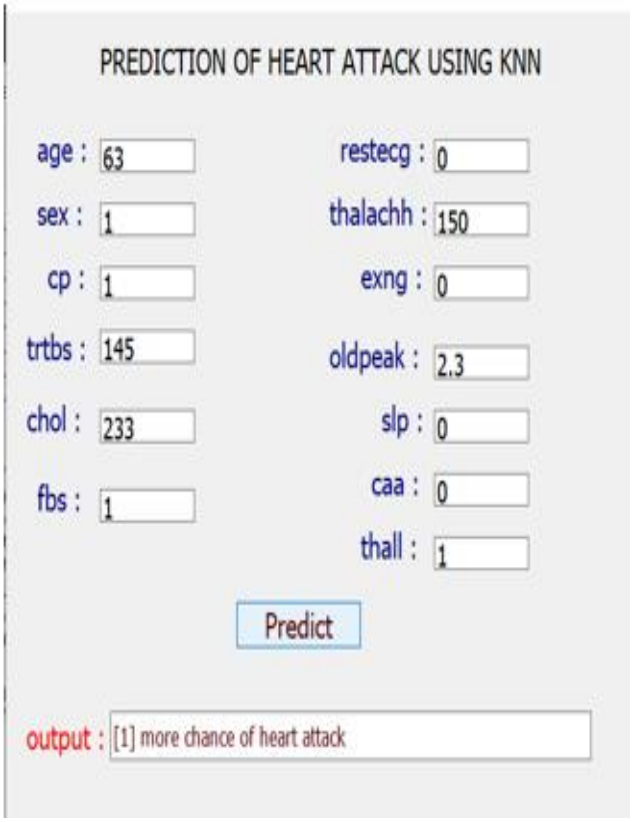
```
     output
8         1
256       0
29        1
274       0
120       1
..      ...
173       0
251       0
145       1
46        1
281       0

[61 rows x 1 columns]
[[14 10]
 [12 25]]
[0]
69.63696369636963 %
```

Implementing Code in PYQT5:

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'sumanth.ui'
#
# Created by: PyQt5 UI code generator 5.15.9
#
# WARNING: Any manual changes made to this file will be lost when pyuic5 is
# run again.  Do not edit this file unless you know what you are doing.


from PyQt5 import QtCore, QtGui, QtWidgets


class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(633, 536)
        Form.setStyleSheet("font: 16pt \"MS Shell Dlg 2\";\n"
"color:rgb(0, 0, 127)")
        self.label = QtWidgets.QLabel(Form)
        self.label.setGeometry(QtCore.QRect(90, 10, 441, 41))
        self.label.setStyleSheet("font: 75 16pt \"MS Shell Dlg 2\";\n"
"color:rgb(0, 0, 0)")
        self.label.setObjectName("label")
        self.label_2 = QtWidgets.QLabel(Form)
        self.label_2.setGeometry(QtCore.QRect(30, 70, 51, 31))
        self.label_2.setStyleSheet("font: 16pt \"MS Shell Dlg 2\";")
        self.label_2.setObjectName("label_2")
        self.label_3 = QtWidgets.QLabel(Form)
        self.label_3.setGeometry(QtCore.QRect(30, 110, 51, 31))
        self.label_3.setObjectName("label_3")
        self.label_4 = QtWidgets.QLabel(Form)
        self.label_4.setGeometry(QtCore.QRect(40, 150, 41, 31))
        self.label_4.setObjectName("label_4")
        self.label_5 = QtWidgets.QLabel(Form)
        self.label_5.setGeometry(QtCore.QRect(20, 200, 81, 21))
        self.label_5.setObjectName("label_5")
```

PYQT5 Output:





CONCLUSION

- The internship aims to use Python programming language for Machine Learning so as to apply the theoretical knowledge to solve real-time and complex problems. The internship helped to find appropriate prediction model to the problems by applying suitable learning algorithm which can be used in future. The internship project assigned by the company helped to improve programming skills and to implement basic knowledge for solving real world problem. In this project **KNN Algorithm** is used to predict the **Heart Attack. After** having a basic understanding of Supervised learning, we explored the KNN Algorithm which is used to solve machine learning problem. The Income varies based on the inputs given in the salary dataset.

REFERENCES

[1] www.karunadutechnologies.com

[2] https://www.simplilearn.com/10-algorithms-machine-learning-engineers-need-to-know- article

[3] https://www.edureka.co/blog/machine-learning-algorithms/

[4] https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/