MINI PROJECT REPORT

Submitted to the faculty of Engineering and Technology
VI Semester, B.Tech
(Autonomous Batch)

DIABETES PREDICTION USING LOGISTIC REGRESSION

BY
V GANESH
B18CS062

Under the Guidance of

Johnson Kolluri

Asst. Professor

Department of Computer Science & Engineering
Kakatiya Institute of Technology & Science
(An Autonomous Institute under Kakatiya University)
Warangal (Telangana State)

2020-21

CERTIFICATE

This is to certify that **V GANESH** bearing roll no: **B18CS062** of the VI Semester B.Tech. Computer Science and Engineering (Autonomous) has satisfactorily completed the Mini Project Report entitled **"Diabetes Prediction Using Logistic Regression".**

Supervisor
Johnson Kolluri
Asst. Professor

Cordinator
K. Srinivas
Asst. Professor

Convener
Dr. Kumar Dorthi
Asst. Professor

Head of the Department
Prof. V. Shankar

ACKNOWLEDGMENT

I extend my sincere and heartfelt thanks to our esteemed guide, **K. Johnson, Asst. Professor** for his exemplary guidance, monitoring and constant encouragement throughout the course at crucial junctures and for showing us the right way.

I am grateful to respected coordinator, **Dr. K. Srinivas, Asst. Professor** for guiding and permitting me to utilize all the necessary facilities of the Institute.

I sincere thanks to respected convener **Dr. Kumar Dorthi, Asst. Professor** for supporting me and to utilize all the necessary facilities of the Institute.

I would like to extend thanks to our respected head of the department, **Prof. V. Shankar**
for allowing us to use the facilities available. I would like to thank other faculty members also.

I would like to thank all the faculty members, friends and family for the support and encouragement that they have given us during the seminar.

V GANESH

B18CS062

ABSTRACT

Diabetes is one of the many major issues in medical field and lakhs of people are affected due to this diabetes. From many years many researches are going on this problem to detect this diabetes. Here we are mainly concerned towards women because during pregnancy they may get diabetes which is also termed as gestational diabetes and due to this there is a higher chance of getting diabetes called type2 in future and this occurs when our human body doesn't use the insulin hormone and it is unable to prepare it. Therefore many methods are there in literature that is used to classify whether a particular human being gets diabetes in future or not. Generally the dataset used for this purpose is Pima Indian diabetes dataset and it is mainly used by the researchers to classify whether an instance has diabetes or not. There are a lot of problems if this diabetes is not treated and it may leads to other organ related diseases. The main problems occur to kidneys, eyes and heart etc. the normal method that is used for this diabetes detection is to visit a hospital or any health care center and we have to reach doctor for treatment. Many researches in machine learning are going on for this purpose and many methods are proposed using the data of people of past and tries to develop models that is used to predict diabetes. In this we are going to propose a method using logistic regression which is technique that is used for detection of diabetes.

ACRONYMS

HTML : Hyper Text Markup Language

CSS : Cascading Style Sheets

ML : Machine Learning

TP : True Positives

TN : True Negatives

FP : False Positives

FN : False Negatives

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

1. INTRODUCTION

1.1. Introduction

What is Diabetes?

When the level of glucose is too high in our body there is a chance of getting diabetes. Glucose doesn't reach to the body cells when our body doesn't prepare insulin and due to this the level of glucose in our body gets increased and this leads to major health issues. There are some common types of diabetes that can occurs to human and they are type 1 diabetes which occurs when our body not prepares any kind of insulin and for this type of diabetes we have to consume insulin daily to live. Another kind of diabetes occurs when our body unable to prepare any insulin and also it cannot use that and this type 2 diabetes can be observed in aged or middle-aged people and this is the most common one.

The last type is gestational diabetes which occurs in pregnant women and it can be cured after when baby comes out. The main thing here is when there is gestational diabetes it can leads to type 2 diabetes in future and there is also a chance that the diabetes occurs during pregnancy may be type 2 diabetes. This diabetes can leads to major health problems like nerve damage, eye problems, kidney diseases, heart diseases etc. The main reasons for getting this type 2 diabetes is getting older, if there is diabetes in the family, overweight, high BP and not doing any activity physically. The main symptoms of type 2 diabetes are blurry vision, tiredness, increased hunger, increased urination etc. Due to hormonal changes there is high chance of getting gestational diabetes and when women are overweight and during pregnancy they become more weighted and there is high chance of getting diabetes in future.

Why we are using Machine Learning?

Extracting the knowledge from data is machine learning and it is also called as statistical learning because it is related to computer science, statistics and Artificial Intelligence. By using hardcoded rules there are many disadvantages to take decisions because it requires very deep understanding of data to make decisions and the knowledge in particular domain is very important. Normally the rules created by person with data by using software engineering which is helpful to answering a problem. Coming to machine learning it is finding new patterns or rules in the problem by using data given and solutions. To perform machine learning task there are many steps to perform like collecting the required data, data preparation, model training, model evaluation and to increase the performance of our built logistic regression copy.

Therefore machine learning mainly focuses on learning the data which we have provided and it develops computer programs. This all starts with by analyzing the given data and based on the training examples we have given it takes better decisions in the coming days. By experience and learning the machine learning model improve its performance and accuracy. Although machine learning expands its branches in many fields it may fails to get expected results in some

cases and many reasons for this are bias in data, resource lacking and many others. There are many applications of machine learning and it is used in various tasks like sentiment analysis, image recognition, medical field, speech recognition etc.

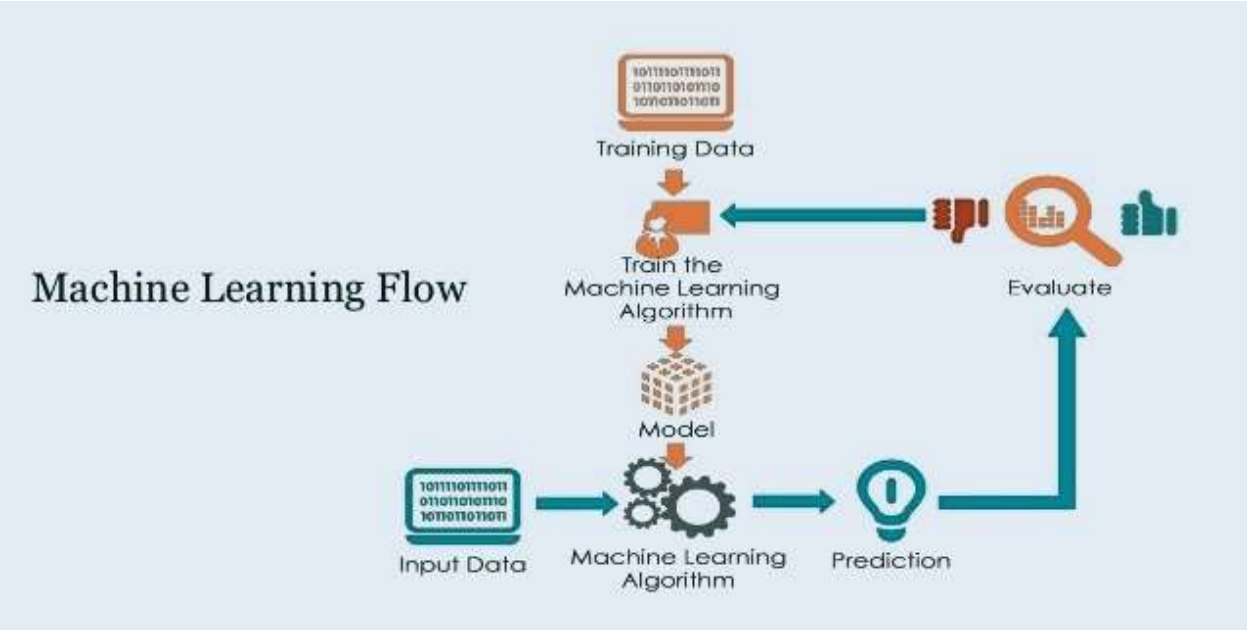


Fig 1-machine learning flow

Types of Machine Learning:

Let us discuss about the types that are in machine learning and related information to that:

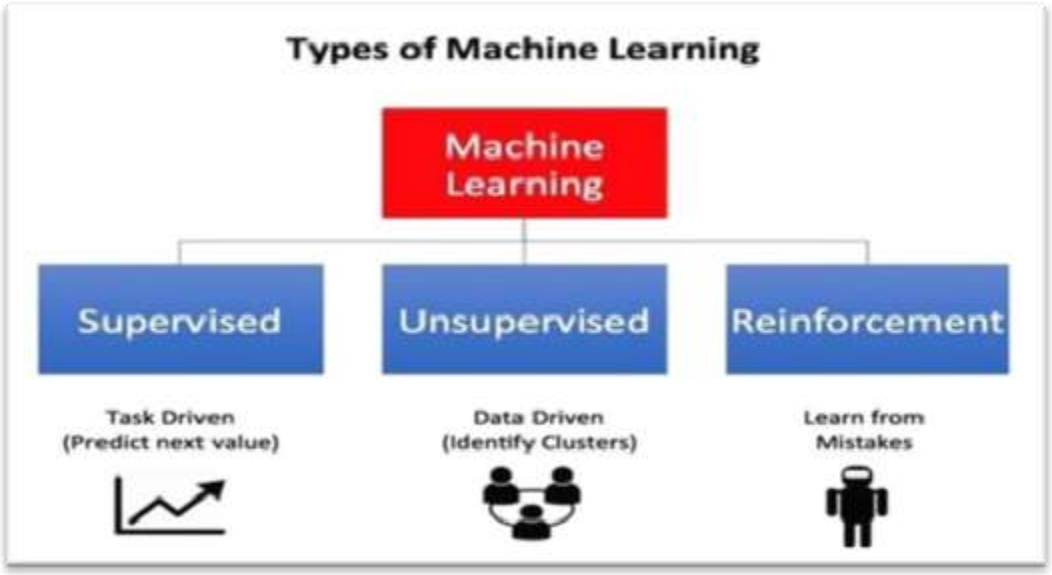


Fig 2-types of machine learning

1. **Supervised Learning**: In supervised machine learning we are dealing with mainly labeled data means the target is known. The implementation of algorithm is simple and it is easy to understand. The data given here is labeled means we know the output when particular instance is given. For example we are given with a fruit and it is labeled as apple and we train our machine learning model by feeding these apples and telling the model that they are apples and when we test the model by giving an apple it is in the position that it can correctly labeled it as apple.

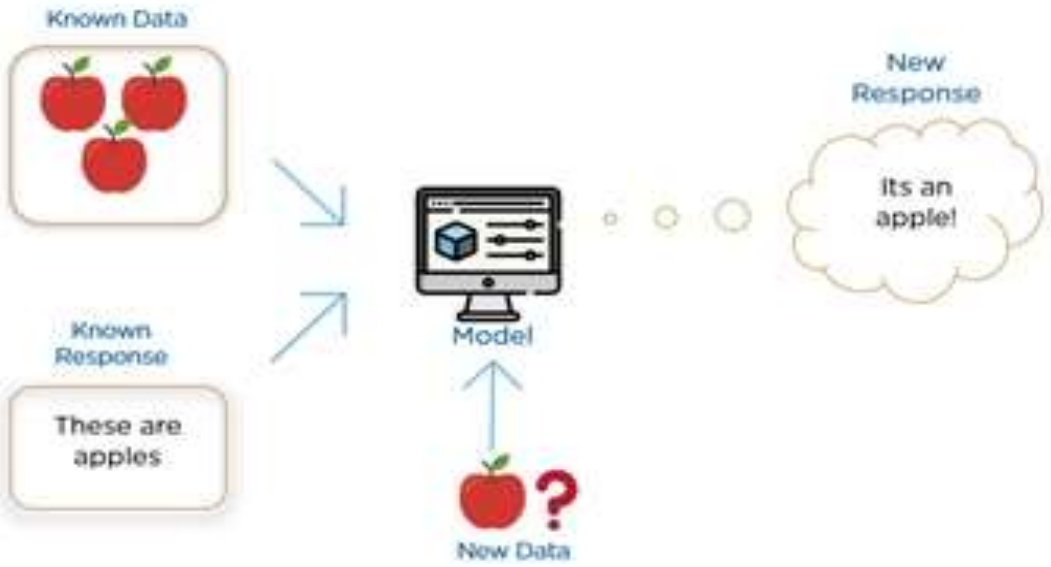


Fig 3-supervised learning

- *Regression*: we can define it as a relationship between things. In this algorithm we are dealing with continuous quantity and our model when given a particular input it should output a quantity which is continuous as at target. For example consider a task that we have to predict the price of land based on area and we are given with many instances of area with price whereas price is our target. In future after building the model when we want to predict price of land with given area the model should do it.

- *Classification*: Here the name indicates that we are going to classify something means the result here is categorical like 0/1, T/F, apple/mango/grapes. Consider we are given a data that to classify the given image whether it is donkey or monkey. First we train our model by feeding the monkey images which are labeled as M and donkey images which are labeled as D. Our model will learn those characteristics according to given labels. Now when we give a new image which is monkey it should classified it as M.

2. **Unsupervised Learning**: opposite to supervised learning here we are given with a data which is not labeled means we don't know any output regarding the given samples and we have to put the samples together based on similarity between them. In unsupervised

learning also there are two types of algorithms called clustering and association. In clustering based on similar features in objects we group them in one class. For example if we are given with apples, bananas, mangoes and they doesn't give any labels for them, based on their color, shape and other characteristics we have to group each other. In association we find the rules or patterns among variables and it is mainly used for marketing purposes.

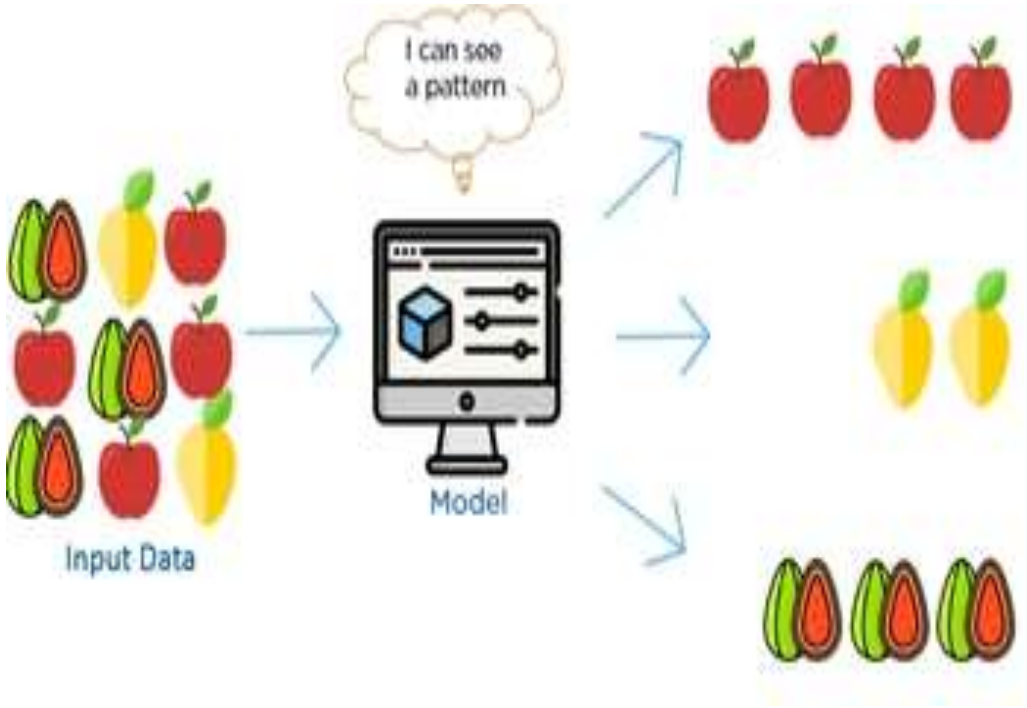


Fig 4-unsupervised machine learning

3. **Reinforcement Learning:** Here the model learns from its mistakes and getting better. For a given input it has to decide what action it should perform and based on its output it may get rewards or any kind of punishments based on its actions and it can learn continuously and tries to perform better than before. Therefore the output of current state depends on previous output also.

1.2 Objectives:

- The main objective here is to help doctors and practitioners by predicting whether the person (here women) will get diabetes or not in coming days.
- To build a classifier using logistic regression algorithm to predict whether a given instance has diabetes or not.
- We have to develop a user interface for predicting diabetes.
- The user interface will takes input as all diagnostic requirements for predicting diabetes and it outputs certain measures based on which the classifier predicts.
- The dataset for building a classifier or machine learning model is taken from National Institute of Diabetes and Digestive and Kidney Diseases.
- The main aim of this dataset is to early stage detection of diabetes and it will be very helpful to start treatment to cure the disease.
- The application here is related to medical field which is very useful and helpful for patients because it automatically detects diabetes based on old patient's information.
- Automated diagnosis in medical field is the key that should be from this diabetes prediction using Logistic Regression.

1.3 Methodology:

- Data Collection.
- Data Exploration and analysis.
- Data Preprocessing.
- Data Splitting.
- Training the model.
- Testing the model.
- Model Evaluation.
- Deployment.

CHAPTER 2

LITERATURE REVIEW

In literature there are many research conducted on diabetes prediction and some of them are descried here.

- There are many health problems that are occurring due to diabetes and according to WHO there are nearly 1.2 million deaths because of this diabetes and many deaths are occurred because of some other diseases like heart problems, kidney diseases etc.

- Faruque and H.Sarker also proposed that due to metabolic disorder our body tends to get the diabetes which is a common disorder. They have implemented various machine learning algorithms in order to provide the solution for the detection of diabetes.

- Miao, Zhao and Wei Sidong discussed that diabetes is a disorder that can occurs when the glucose levels in the human body is high. They use several techniques to get insights about disease like neural networks, support vector machine.

- Santhosh Kumar and Lakshmi thought hospital databases as a good source and resource for useful information which is required for diagnosis of diabetes. They used several algorithms from data mining and they used some natural language processing techniques for mainly to extract rules from the available data.

- P.Suresh Kumar and Teja used the techniques from data mining for detecting the diabetes mellitus like Decision trees, Naïve Bayes, Support Vector Machine.

- Goldberg, Elliot and Nilimi have proposed based on cloud and it is wireless which is a monitoring one for diabetes and it is one of the less cost system for diabetes which is very innovative and cloud based.

- Bhavya, Sanjay, Shiva proposed system design for prediction of diabetes where there is a admin and users can register into it. Data analysis is stored in storage server and the algorithm used is KNN. When the user enters for first time they have to register into that and after login they can enter their details for predicting the disease. There are many other facilities like discussion forum, updating profile etc.

- Vani and Deisy proposed technique based on data mining using a fuzzy set related classification and this is very useful in detecting the outliers in dataset and used factors like BMI, Glucose, Age which is used to predict the diabetes.

- Dalakleidi proposed various algorithm implementations like neural networks, logistic regression, decision trees and Bayesian-based one which is used to predict heart related diseases which are caused due to diabetes.

CHAPTER 3

IMPLEMENTATION

3.1 ALGORITHM:

3.1.1 Linear Regression:

We have already discussed that in supervised learning we have 2 types of algorithms. Our diabetes prediction is related to classification. Although first we discuss about regression algorithms and with help of this we can get better understanding about logistic regression.

Consider an example where we have given an area and we have to predict the price of area. Here the rental price is dependent variable means it depends on area and area is independent variable. Now if we try to fit a line using the given data points using linear regression the figure looks like as shown below. Here the process fitting a line for given data points is called training the model and after the training is finished we can find price for new areas using the trained model.

Size	Rental Price
350	1840
410	1682
430	1555
550	2609
480	1815
650	1900
900	4615
830	3512
730	2153
1200	6308

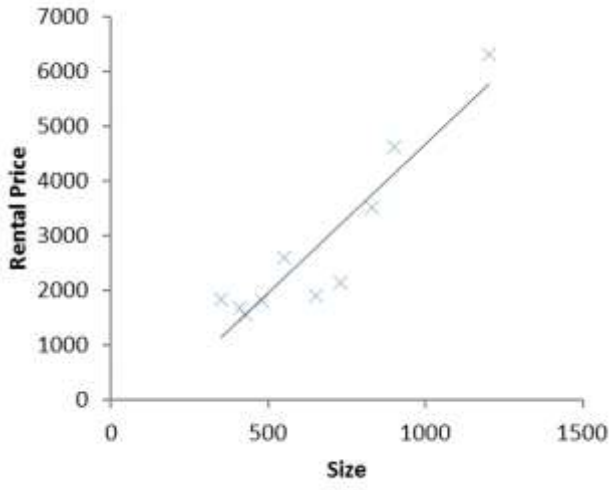


Fig 5-linear regression

Now let examine how we get these line fitted to given data points. Here we are using a line equation or hypothesis as $h_\theta(x) = \theta_{0+}\theta_1 x$ which is used for mapping x values to y values. Here $\theta_{0,}$ θ_1 are parameters and we have to choose the values of this parameters such that the line fits best to given points and our value of hypothesis should nearly match the actual value y for

given examples. Therefore our main aim is to decrease the difference between actual and predicted one and this is called as cost function J(θ). What we have to do here is we start the model by giving different values to $\theta_{0,}$ θ_1 and we have to find the errors which are squared and why squared is because we want to deal with positive values. Now the question arises in our mind how to make less the value of J(θ), here comes to rescue us that is nothing but gradient descent algorithm. The gradient descent method is useful to cut down the value of cost function. What we have to done here is simply start training our model by taking some random values for $\theta_{0,}$ θ_1 and we have to change the values of these parameters in the sense of reducing the cost function J($\theta_{0,}\theta_1$) and we have to obtain the minimum value for it. If we observe the graph below where f(x) is J(θ) and the parameters on x-axis.

There are some terms in gradient descent algorithms like $\alpha, \frac{\partial}{\partial\theta}J(\theta)$ where α is a learning rate and it is main parameter to model such that how fast it can learn and $\frac{\partial}{\partial\theta}J(\theta)$ is nothing but slope or gradient and how it is used is if the gradient is negative then the parameter moves towards right and if it is positive then the parameter is moving towards the left side as shown below. If we carefully observe the graph shown below for every iteration the value of cost is coming down and the value of cost comes to minimum after certain time and the convergence depends on parameters α such that if the value is very small then our algorithms is very slow to get the cost which is minimum and it is very bad even if our α is very big then it may be impossible to get minimum cost because it may overshoot it.

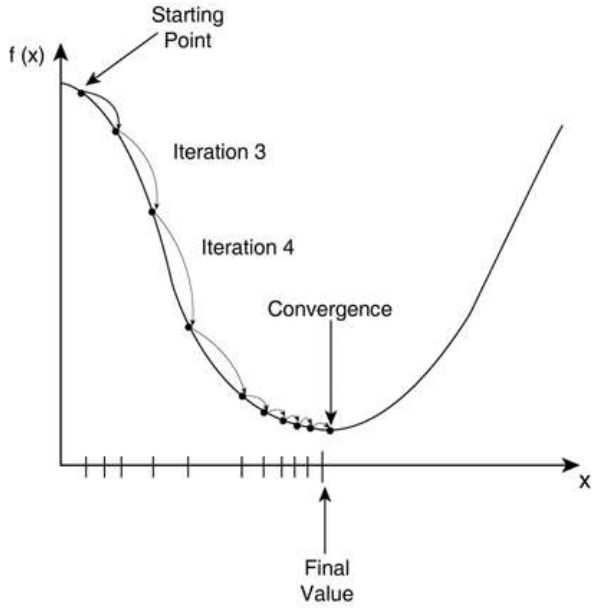


Fig 6-gradient descent

Let ‘m’ be the number of samples in training dataset then our hypothesis, parameters, cost function and gradient descent are written below.

Hypothesis: $h_\theta(x) = \theta_{0+}\theta_1 x$

Parameters: θ_0, θ_1

Cost function: $J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m} (h_\theta(x^i) - y^i)^2$

Gradient descent:

Repeat
{
$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$
}

3.1.2 Logistic Regression:

Now we are going to see other type of supervised learning algorithm that is related to classification. Consider an example will customer buy insurance or not and the answer here is yes/no and if we observe the below dataset here there are attributes age and have_insurance and if we carefully observe the data if the age increases he is more likely to buy an insurance and if the age is less then he is not interested in buying insurance and we can say 1-means interested and 0-means not interested. Now here we have to build a machine learning model that can do prediction whether they buy insurance or not.

age	have_insurance
22	0
25	0
47	1
52	0
46	1
56	1
55	0
60	1
62	1
61	1
18	0
28	0
27	0
29	0
49	1

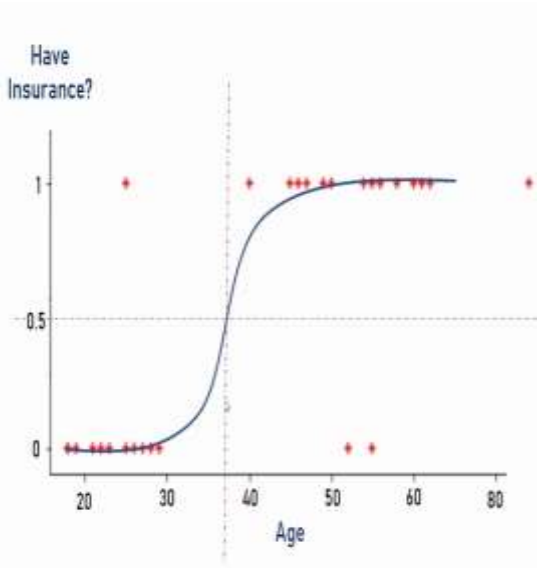


Fig 7-logistic regression

The first thing we do is scatter plot and starts using linear regression and we assign that if predicted value is >0.5 the person buy insurance and if it is <0.5 then he is not going to buy and if we assume a data point far on RHS such that customer age is more than 80and he buys insurance and now the problem arises that for many point the answer is yes but our model predicts as no and here it can be confirmed that we cannot use linear regression for this kind of classification problems whereas it is used to predict a continuous quantity. Now if we imagine a line as shown in below figure it is much better compared to previous one and we can clearly say that this model works better than previous because our problem here is related to classification and the predicted value is categorical. Logistic Regression is one of the technique that is used for classification. Now let's examine how to get this curve.

Sigmoid function:

The curve is obtained by passing all values through sigmoid function before plotting. The sigmoid function maps any value between 0 and 1 and here we are dealing with classification and expects output as 0 or 1. Therefore our predicted value turns into probability after passing through sigmoid function.

$$f(z) = \frac{1}{1 + e^{-z}}$$

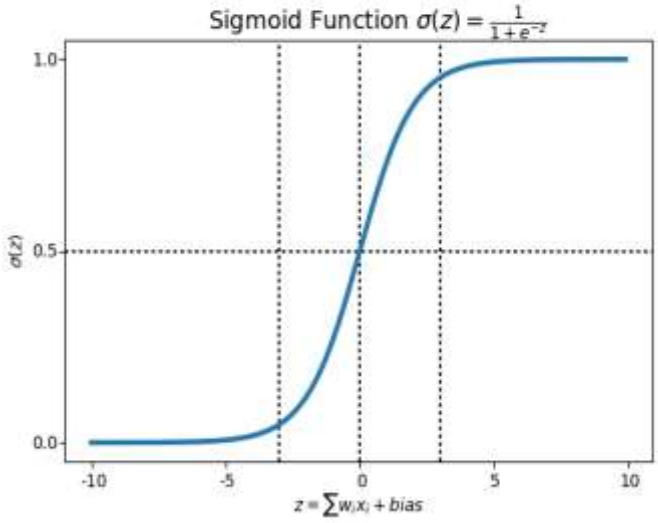


Fig 8-sigmoid function

Hypothesis:

In linear regression we have used hypothesis as $h_\theta(x) = \theta_{0+}\theta_1 x$ and here also we are using same hypothesis but there is a small change here. Instead of plotting the result from hypothesis we are plotting the values after passing through sigmoid function. Therefore our hypothesis becomes

$$h_\theta(x) = \frac{1}{1 + e^{-(\theta_0 + \theta_1 x)}}$$

Decision boundary:

Here our model predicts a value between 0 and 1beacause our value is passing through sigmoid function but our aim is to predict 0 or 1. So we have to set a threshold such that if we get a value above threshold we mark it as one class (1) and if we get a value below threshold then we put them in to second class (0). This threshold is nothing but a boundary that helps to take decisions.

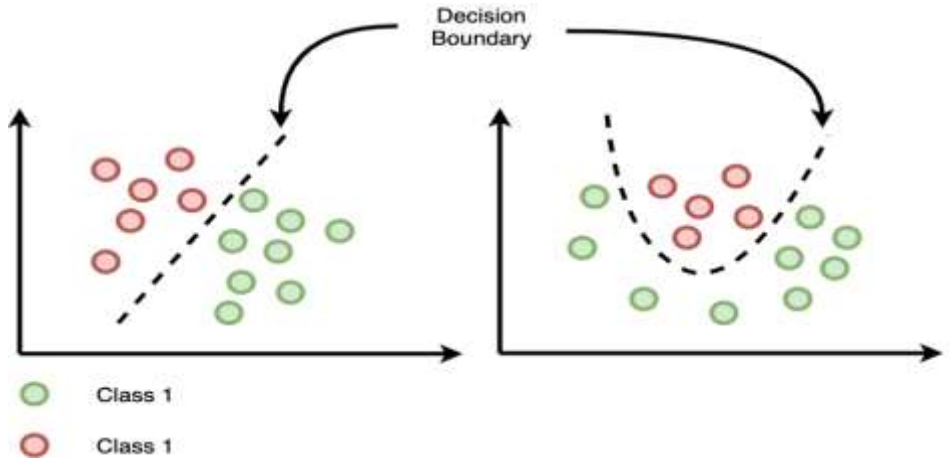


Fig 9-decision boundary

Cost function:

If we use the cost function which we have discussed in linear regression then it will not get good results because due to intervention of sigmoid function here our graph of cost function may turns into non-convex and it has many local minima as shown in the figure below. So it is very hard to get global minimum and we can end up in any local minima and it is not we are expecting.

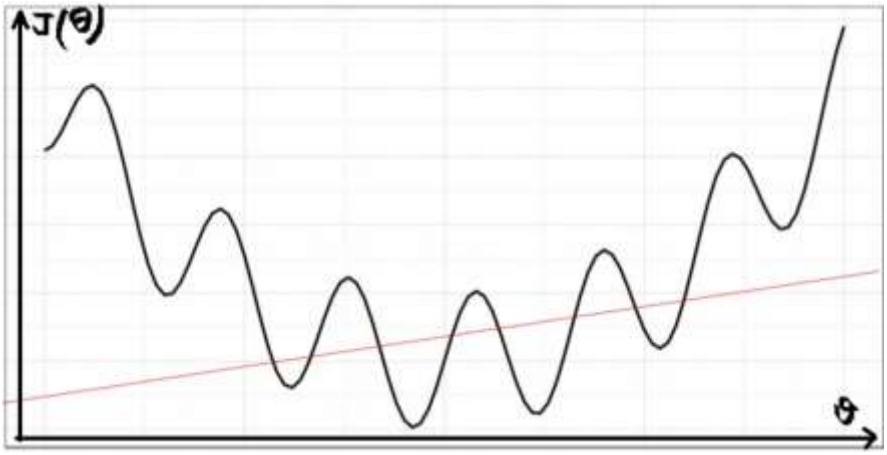


Fig 10-nonconvex cost function

Therefore for logistic regression the cost function can be written as

$$\text{Cost}\,(h_\theta(x),y) \quad = \quad \begin{matrix} -\log\big(h_\theta(x)\big) \;\; if\; y=1 \\ -\log\big(1-h_\theta(x)\big) \;\; if\; y=0 \end{matrix}$$

We can merge the above two equations and cost function can be written as

$$J(\theta) = -\frac{1}{m}\sum[\,y^{(i)}\log\Big(h_\theta\big(x^{(i)}\big)\Big) + (1-y^{(i)})\log\Big(1-h_\theta\big(x^{(i)}\big)\Big)]$$

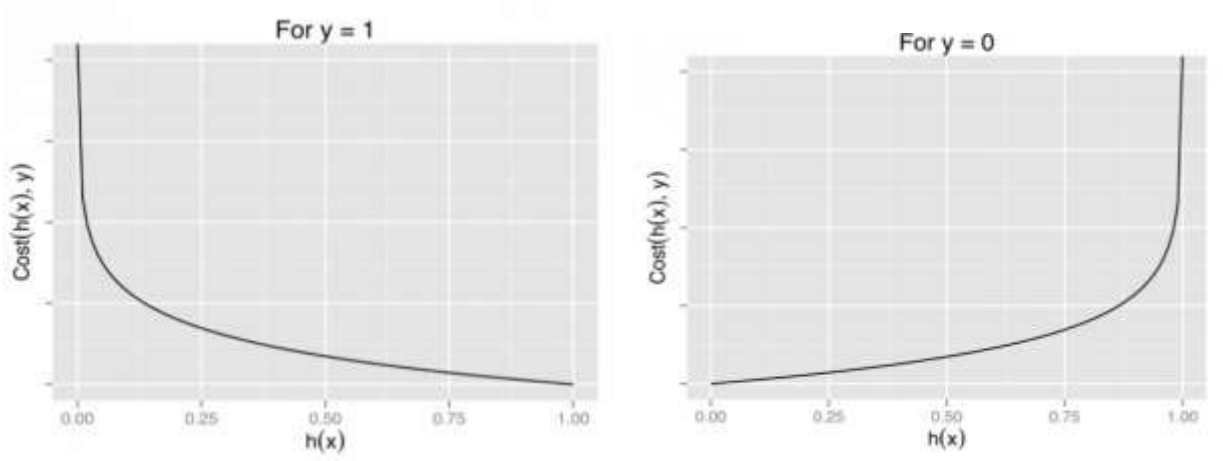


Fig 11-cost function

Gradient Descent:

We have already discussed about gradient descent algorithm that is used to obtain minimum cost and we have to implement this on every parameter.

Repeat
{

$$\theta_j := \theta_j - \alpha \sum_{i=1}^{m} \big(h_\theta\big(x^{(i)}\big) - y^{(i)}\big)x_j^{(i)}$$

}
We have to update θ_j for j=0, 1,……..n at a time.

3.1.3 Steps to build Machine Learning model:

Dataset:

The dataset is actually acquired from National Institute of Diabetes and digestive and Kidney diseases. In this there are all females who are having age above twenty one and they are Native Americans.

Data description:

In the dataset given there are nearly seven hundred and sixty eight tuples or instances and there are nine attributes. Using these attributes we have to find whether a given instance has diabetes or not. The attributes and its related information are shown below.

- *Pregnancies*: it gives information about how many times the person gets pregnancy.
- *Blood Pressure*: it gives the diastolic blood pressure and is measured in mm Hg.
- *Glucose*: it gives the sugar levels after two hours of glucose load in the body.
- *Skin thickness*: it gives thickness of skin and it is measured in mm.
- *BMI*: it gives body mass index and it is measured in kg/m2.
- *Insulin*: it gives us the level of insulin in our body after two hours of glucose load.
- *Diabetes Pedigree Function:* it gives us the measure that how likely the chances of getting diabetes based on the family history.
- *Age*: it gives us the age in years.
- *Outcome*: it gives us the classification result that means if we get result as '0' means there is no diabetes and if we get '1' as result then there is a chance of getting diabetes.

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

Fig-12 data description

Data Exploration and Data Visualization:

Now we can analyze the dataset by exploring it and knowing about features and we can get some details from it. We can see the relation between each of them using a heat map and we can observe that how every attribute is correlated with the output. If we observe below figure we can say that the attributes BMI, age, glucose are highly correlated with the output class.

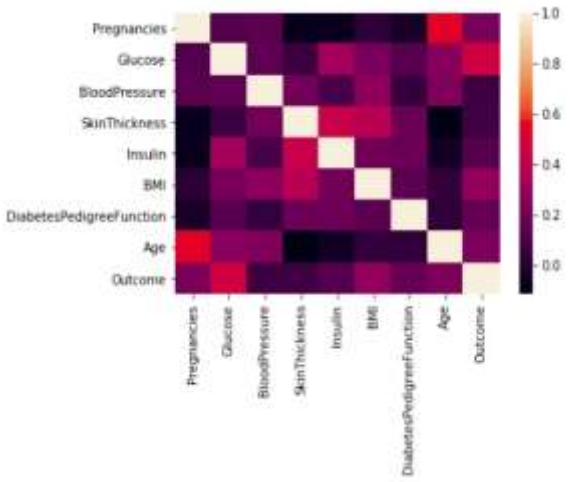


Fig-13 correlation map

Now we can visualize the dataset and there are nearly 500 instances who has output as '0' means they are not diabetic and there are 268 tuples which outputs our class variable as '1'.

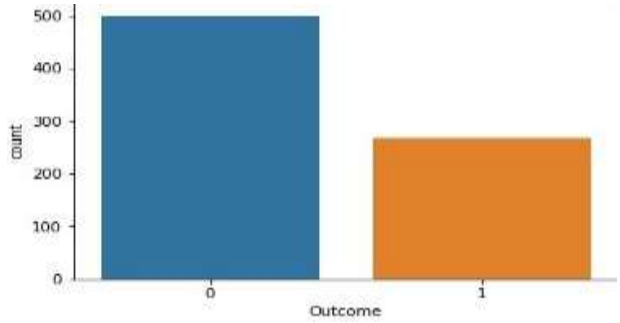


Fig-14 bar graph

We can even get some measures like minimum value, maximum value and some statistic measures like mean, median and standard deviation etc. These are very helpful to understand the data and they are for each attribute or feature in given dataset.

	count	mean	std	min	25%	50%	75%	max
Pregnancies	768.0	3.845052	3.369578	0.000	1.00000	3.0000	6.00000	17.00
Glucose	768.0	120.894531	31.972618	0.000	99.00000	117.0000	140.25000	199.00
BloodPressure	768.0	69.105469	19.355807	0.000	62.00000	72.0000	80.00000	122.00
SkinThickness	768.0	20.536458	15.952218	0.000	0.00000	23.0000	32.00000	99.00
Insulin	768.0	79.799479	115.244002	0.000	0.00000	30.5000	127.25000	846.00
BMI	768.0	31.992578	7.884160	0.000	27.30000	32.0000	36.60000	67.10
DiabetesPedigreeFunction	768.0	0.471876	0.331329	0.078	0.24375	0.3725	0.62625	2.42
Age	768.0	33.240885	11.760232	21.000	24.00000	29.0000	41.00000	81.00
Outcome	768.0	0.348958	0.476951	0.000	0.00000	0.0000	1.00000	1.00

Fig-15 statistics

We can individually examine each feature to know about it and we can even draw a histogram that tells about attribute. For example if we observe the histogram of insulin feature we can notice that there is large number of rows which has insulin value as zero. Similarly for all other columns like glucose, skin thickness, BMI etc.

Data Preprocessing:

There are many preprocessing techniques are there to build a good model.

- *Balancing:* In real world many datasets are Imbalanced and due to this when we train our model using these kinds of datasets our model may show bias. Consider in real life the persons who got cancer are very less compared to normal people. There are some techniques to balance the dataset like oversampling which increases the instances that are present in smallest class. Under sampling decreases the instances that are present in largest class. Another manual technique is to adjusting the threshold.

- *Missing Values*: There are many samples in our data that has missing values in columns. We have to check in the dataset that where the missing values are there. There are many techniques to handle this situation. We can delete the rows when we encountered missing values and we can even replace those missing values using some statistics. For example if we consider numeric data we can replace missing values with mean and if we take categorical data we can replace the missing values with most frequently occurring value that is mode.

- *Normalization*: normalization is very useful to convert every attribute which is measured on different scales to single common scale. Z-score is one of the normalization techniques that are performed like this every value in the column is subtracted with average of that column and divided by standard deviation. Another technique in normalization is dividing each value in column by maximum value.

- *Outlier detection*: In any dataset it is common to have outliers who are acting different from the other instances and having these outliers in our dataset may decrease our model accuracy and to get more accuracy what we have to do is to remove those abnormal behavioral instances. There are many techniques like extreme value analysis, statistical and probabilistic models etc.

- *Standardization*: There is difference between normalization and standardization where standardization is used to change the data such that the mean of data is zero and root of variance is one and we already seen about normalization.

Splitting the dataset:

After the data preprocessing is done now we have to divide the dataset into training and testing datasets whereas training dataset is used to make the model learn and testing dataset is to verify our model such that how good it learns. Normally 70-80% of data is used for training purpose and 20-30% of data is used for testing the model.

Training the model:

After splitting of data the training data is used for training the model and it is used to make the model learn. We have already discussed that how to make the model learn data using gradient descent algorithm such that by minimizing the error. For each epoch the whole training dataset is passed through the model and updating the weights thereby decreasing the error. After training we get a model that is best fitted to the data that is used for training and now our model is ready.

Testing the model:

After training is completed what we have to do is with the help of testing data we have to test the trained model whether it is classifying our instances correctly or not.

Model Evaluation:

After testing phase we can evaluate our machine learning model based on its performance. Here there is a matrix called confusion matrix which is very helpful for this kind of job. Before moving to the performance measures we have to know about some terms. Consider covid-19 disease and machine learning model to know about the terms given below:

- *True Positive*: let a person have covid-19 disease it is considered as positive and our model predicts that it is true and this is called true positive.

- *True Negative*: let a person have no covid-19 disease it is considered as negative and our model predicts that it is true and this is called true negative.

- *False Positive*: let a person have no covid-19 disease it is considered as negative and our model predicts that it is false means our model wrongly tells that a person has covid-19 although it is negative sample and this is called false positive.

- *False Negative*: let a person have covid-19 disease it is considered as positive and our model predicts that it is false means our model wrongly tells that a person has no covid-19 although it is positive sample and this is called false negative.

Now observe the below figure which is confusion matrix and there are some evaluation measures like Accuracy, Specificity etc. let's see about them.

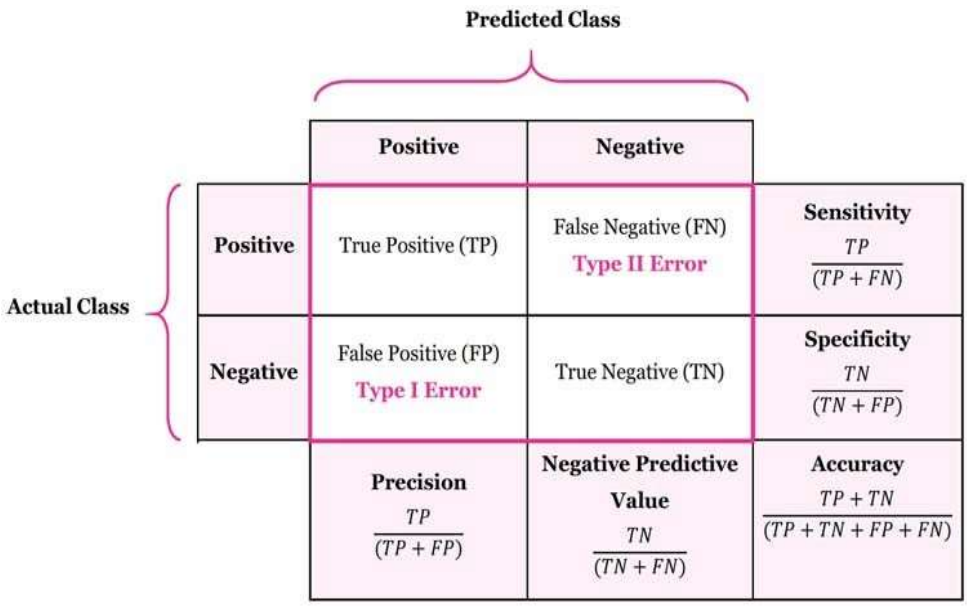


Fig-16 confusion matrix

- *Accuracy*: it can be also called as classifier recognition rate means how much percentage of tuples that are correctly classified by our model.

$$Accuracy=\frac{TN+TP}{FP+TN+TP+FN}$$

- *Error Rate*: It can be called as opposite to accuracy.

$$Error\ rate=\frac{FN+FP}{FP+TN+TP+FN}$$

- *Sensitivity or Recall:* It is also called as rate of true positives or recognition rate means how many tuples that are positive are truly classified.

$$Sensitivity=\frac{TP}{TP+FN}$$

- *Specificity*: It is also called as rate of true negatives means that how many tuples that belongs to negative class are truly classified as negative.

$$Specificity=\frac{TN}{FP+TN}$$

- *Precision:* It can be also called as measure of exactness and it tells about that how many instances that belong to class positive are classified exactly as positive.

$$Precision=\frac{TP}{FP+TP}$$

- *F1-score*: The evaluation measures recall and precision can be termed as single evaluation measure and that is called F-score or F1-score.

$$F=\frac{2*precision*recall}{precision+recall}$$

Saving the model:

We can save our machine learning model which is already trained for further use and when we want to classify new examples we can use this saved model by avoiding training again and again when we want to classify new tuples. This process of saving the model is called serialization and when we want to classify new instances we can perform deserialization.

Deploying the Model:

We can build a user interface which takes input as features in the dataset and outputs result after passing features through classifier. This interface can useful to all and it can be used for model validation in future. We can even get feedbacks from user that our model can correctly predicting or not.

3.2 SOFTWARE REQUIREMENT SPECIFICATIONS:

A document that specifies the nature of a software model or project is commonly referred to as Software Requirement Specifications (SRS). This can be stated as manual of project and is prepared before proceeding onto the project. We have to follow some important guidelines in preparing an efficient SRS document which can be understood easily. This consists of scope, purpose, functional and non-functional requirements of software application as well as requirements of software and hardware. This also consists of details regarding conditions of environment, safety, security, quality attributes etc.

Hardware Requirements:

Processor : minimum core i3 processor
RAM : minimum 2GB or 8GB maximum
Hard Disk Space : more than 50GB

Software Requirements:

Operating System : Windows 8 or later versions of Windows
Presentation Tier : HTML, CSS
Anaconda Software

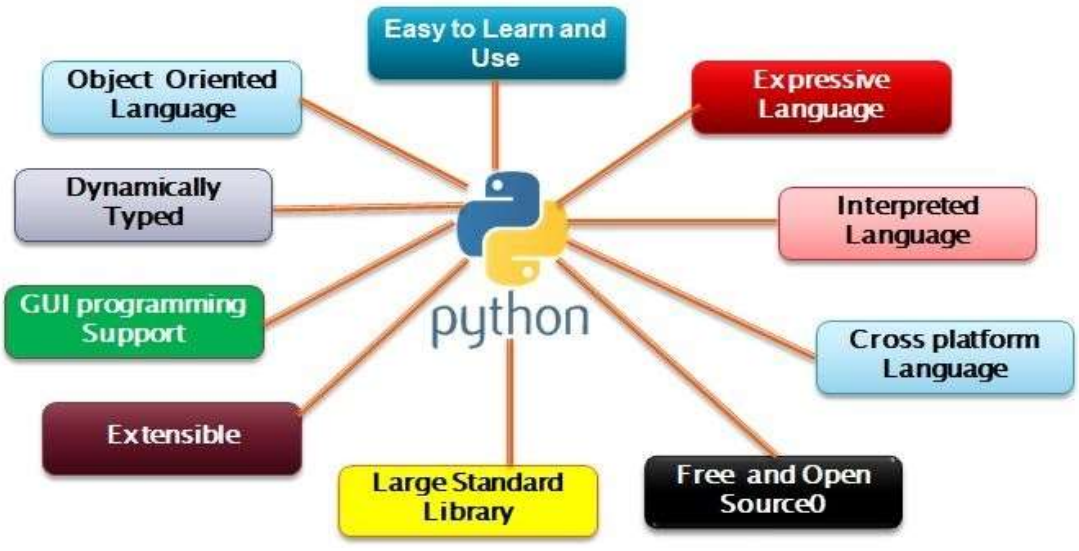
3.2.1 TECHNOLOGY DESCRIPTION:

Python:

- Python is an oop language and everyone can understand it easily.
- It is interpreted, high-level, interactive language which is used worldwide in many applications.
- The indentation feature of python improves its code readability.

Syntax and Semantics:

- Python doesn't use any curly braces like other languages and it doesn't uses semicolon at end.
- Python statements includes assignment statement which is like this '='.
- If statement can be used to execute conditional statements and we can even use elif or else along with that.
- Using while we can execute statements in the loop until a specified condition is true.
- for is also used to execute a statement required number of times.
- Exceptional handling in python can be done using the statements like try and except and we can even use finally block which executes always.
- def statement in python is used to define methods or functions.
- Return statement is used to return something from a method or function.
- class statement is used to define a blueprint to create objects.
- import statement is useful while importing the modules, functions or classes which are already there.



Flask web framework:

In python there is micro web framework which is flask and the template engine used by this framework is jinja.it doesn't want any particular libraries and it supports extensions which we can add additional features of application.

Features:

- Extensive documentation.
- Development server and debugger.
- Unicode based.
- Uses jinja templating.
- Feature enhancement by extensions.

Example program:

```
#first import flask
from flask import Flask
#next instantiate flask applications
app=Flask(__name__)
#using route decorator we can traverse to web pages
@app.route("/")
#here it is the root page and the content to be displayed is written in python function
def welcome():
   return "welcome to flask"
```

Execution:

```
>>set FLASK_APP=filename.py
>>flask run
```

HTML:

Hypertext markup language is used as markup language for documents. It is a front-end technology and the web browser receives documents of HTML from server and it defines the structure of a web page. The elements in html are considered as building blocks for web pages.

CSS:

Cascading style sheets gives the presentation for the page that is written using HTML. CSS gives presentation, layout, fonts and colors etc. It has many properties regarding styles and the specifications of CSS are maintained by W3C.

3.3 FLOWCHART:

The steps to check the result using user interface:

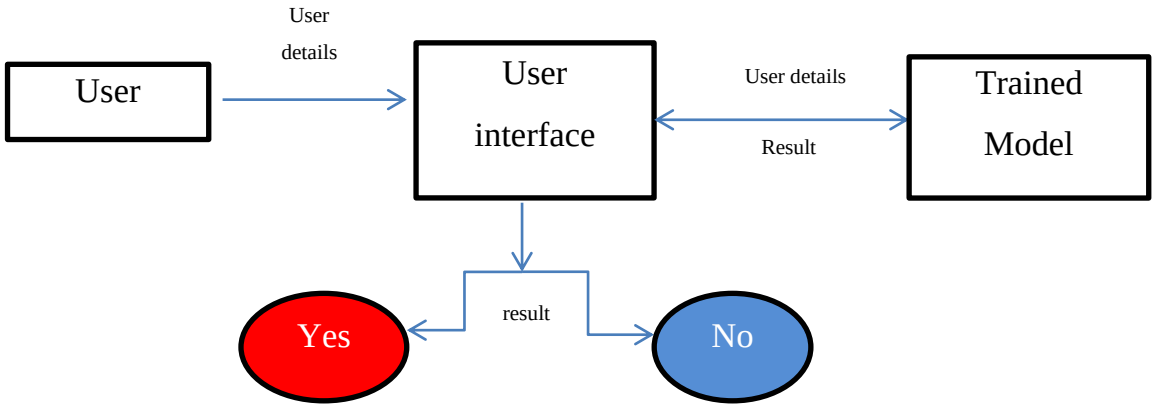


Fig 17-user interface flow

The steps to be followed while building a model is shown below in the form of flow chart:

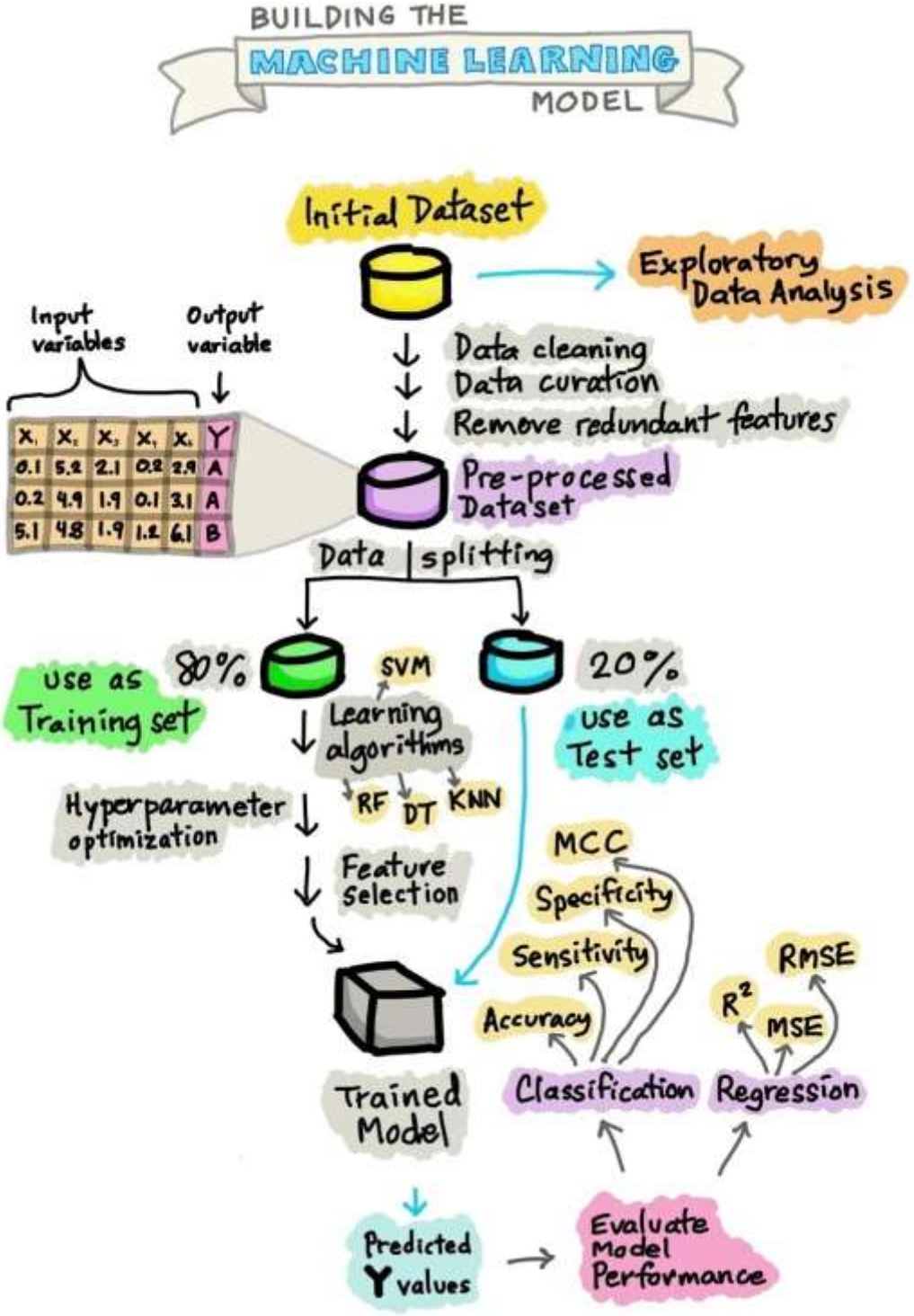


Fig 18-building ML model

CHAPTER 4

EXPERIMENTATION AND RESULTS:

4.1 Experimental work:

4.1.1 Implementation of Logistic Regression for diabetes prediction:

```
import numpy as np
from sklearn.model_selection import train_test_split
from random import seed
from random import gauss
from matplotlib import pyplot as plt
from sklearn import metrics
import pandas as pd



class LR:

    def __init__(self,the_rate=0.001,loop_rotations=100):
        self.n_iters=loop_rotations
        self.weights=None
        self.lr=the_rate
        self.bias=None

    def predict(self,features):

        req_ans = np.dot(features, self.weights)+self.bias
        the_predicted_val=self._sigmoid(req_ans)
        req_cls=[1 if  the_required>0.5 else 0 for the_required in the_predicted_val]
        return req_cls,the_predicted_val

    def calc_the_cost(self,features, labels,n_samples):

        the_val_pred= self._sigmoid(np.dot(features,self.weights)+self.bias)
        ol=list(self.weights)
        theta=0
        for i in ol:
            theta+=i*i

        cost=((-labels*np.log(the_val_pred))-((1-labels)*np.log(1-the_val_pred)))
        total_c_find = cost.sum() / n_samples

        return total_c_find
```

```
def  changing_w(self,features,labels,n_samples):

    ans_req=np.dot(features,self.weights)+self.bias
    y_predicted=self._sigmoid(ans_req)
    ol=list(self.weights)
    theta=0
    for i in ol:
        theta+=i

    gradient=(1 / n_samples)*(np.dot(features.T,(y_predicted-labels)))
    db=(1 / n_samples)*(np.sum(y_predicted-labels))
    self.weights-=self.lr*gradient
    self.bias-=self.lr*db


def train(self,features, labels):
    n_samples,n_features=features.shape
    v=[]
    seed(2)
    for _ in range(n_features):
        v.append(gauss(0,1))
    self.weights=np.array(v)
    self.bias=0
    all_c_val = []

    for i in range(self.n_iters):
        self.changing_w(features, labels,n_samples)
        found_c= self.calc_the_cost(features, labels,n_samples)
        all_c_val.append(found_c)

    return all_c_val

def _sigmoid(self,por):
    return 1.0 / (1 + np.exp(-por))

def accuracy(self,dep_x,actu_y):
    non_y,y_prob=self.predict(dep_x)
    plt.figure(figsize=(10,3))
    yp=list(y_prob)
    yt=list(actu_y)

    ar_true_0=[reqvar for reqvar in range(len(yt)) if yt[reqvar]==0]
    ar_true_1=[reqvar for reqvar in range(len(yt)) if yt[reqvar]==1]
    ar_pred_0=[yp[j] for j in ar_true_0]
    ar_pred_1=[yp[j] for j in ar_true_1]
```

```
        plt.hist(ar_pred_0,bins=50,label='Negatives',color='r')
        plt.hist(ar_pred_1,bins=50,label='Positiives',alpha=0.5,color='b')
        plt.xlabel('prob of eing pos')
        plt.ylabel('num of records')
        plt.legend(fontsize=10)
        plt.tick_params(axis='both',labelsize=10)
        plt.show()

        conf_2darray=metrics.confusion_matrix(actu_y,non_y)
        print(conf_2darray)
        accuracy=np.sum(actu_y==non_y)/len(actu_y)
        return accuracy,conf_2darray

df=pd.read_csv('C:\\Users\\Ayyappa\\OneDrive\\Documents\\ganesh\\diabetes_project.csv')


df.reset_index(inplace=True, drop=True)

filt=(df['Outcome']==0) & (df['Insulin']!=0)
ins_med_0=df.loc[filt].median()
df.loc[((df.Insulin==0) & (df.Outcome==0)) ,'Insulin']=130.2


filt=(df['Outcome']==1) & (df['Insulin']!=0)
ins_med_1=df.loc[filt].median()
print('is',ins_med_1['Insulin'])
df.loc[((df.Insulin==0) & (df.Outcome==1)) ,'Insulin']=206.8


filt=(df['Outcome']==0) & (df['Glucose']!=0)
glu_med_0=df.loc[filt].mean()
df.loc[((df.Glucose==0) & (df.Outcome==0)) ,'Glucose']=110.0

filt=(df['Outcome']==1) & (df['Glucose']!=0)
glu_med_1=df.loc[filt].mean()
df.loc[((df.Glucose==0) & (df.Outcome==1)) ,'Glucose']=142.3


filt=(df['Outcome']==0) & (df['SkinThickness']!=0)
sk_med_0=df.loc[filt].mean()
df.loc[((df.SkinThickness==0) & (df.Outcome==0)) ,'SkinThickness']=27.0


filt=(df['Outcome']==1) & (df['SkinThickness']!=0)
sk_med_1=df.loc[filt].mean()
```

```
df.loc[((df.SkinThickness==0) & (df.Outcome==1)) ,'SkinThickness']=33.0

filt=(df['Outcome']==0) & (df['BloodPressure']!=0)
bp_med_0=df.loc[filt].mean()
df.loc[((df.BloodPressure==0) & (df.Outcome==0)) ,'BloodPressure']=70.0

filt=(df['Outcome']==1) & (df['BloodPressure']!=0)
bp_med_1=df.loc[filt].mean()
df.loc[((df.BloodPressure==0) & (df.Outcome==1)) ,'BloodPressure']=74.5

bc=df.loc[:,['Pregnancies','Glucose','BloodPressure','SkinThickness','Insulin','BMI','DiabetesPedi
greeFunction','Age']]
dc=df.loc[:,['Outcome']]

X=bc.values.tolist()
X=np.array(X)
y=dc.values.tolist()
y=np.array(y).flatten()
mean=np.mean(X,axis=0)
std=np.std(X,axis=0)
X_norm=(X-mean)/std

X_train,X_test,y_train,y_test=train_test_split(X_norm,y,test_size=0.2,random_state=6)

reg=LR(lr=0.01,n_iters=20000)
cost=reg.train(X_train,y_train)

accuracy,con_2d=reg.accuracy(X_test,y_test)
tr_pe=con_2d[1][1]
tr_ne=con_2d[0][0]
fs_pe=con_2d[0][1]
fs_ne=con_2d[1][0]

print(tr_pe,tr_ne,fs_pe,fs_ne)
print('actual pos tuples',tr_pe+fs_ne)
print('actual neg tuples',tr_ne+fs_pe)
error_rate=(fs_pe+fs_ne)/(tr_pe+tr_ne+fs_pe+fs_ne)
acc=(tr_pe+tr_ne)/(tr_pe+tr_ne+fs_pe+fs_ne)
sensitivity_recall=(tr_pe)/(tr_pe+fs_ne)
specificity=(tr_ne)/(fs_pe+tr_ne)
precision=(tr_pe)/(tr_pe+fs_pe)
F1score=(2*precision*sensitivity_recall)/(precision+sensitivity_recall)

print('accuracy=',np.round(acc*100,2))
print('error_rate=',np.round(error_rate*100,2))
print('sensitivity_recall=',np.round(sensitivity_recall*100,2))
```

```
print('specificity=',np.round(specificity*100,2))
print('precision=',np.round(precision*100,2))
print('F1score=',np.round(F1score*100,2))
#cost function graph

xaxis=list(range(reg.n_iters))
yaxis=cost
print(xaxis,yaxis)

plt.plot(xaxis,yaxis,color='red')
plt.xlabel('iterations')
plt.tight_layout()
plt.style.use('seaborn')
plt.ylabel('cost value')
plt.title('COST FUNCTION')
plt.show()
```

4.1.2 Graphical User Interface:

Layout.html

```
<html lang="en">
<head>
   <title>Diabetic Diagnosis</title>
   <link href="{{ url_for('static', filename = 'css/main.css') }}" rel="stylesheet" media="all">

   <style>
      body {
        background-attachment: fixed;
        background-image: url('{{ url_for('static', filename = 'images/diabetes.jpg') }}');
        background-size: 100% 100%;
        background-repeat: no-repeat;
        background-size: cover;

      }
      </style>
</head>

<body >
   <div class="page-wrapper p-t-100 p-b-100 font-robo">
      <div class="wrapper wrapper--w680">
         <div class="card card-1">
            <div class="card-heading"></div>
            <div class="card-body">
```

```
            {% block content %} {% endblock %}

          </div>
        </div>
      </div>
   </div>
</body>
</html>
```

home.html

```
{% extends "layout.html" %}
{% block content %}
                <h3 class="title">Required Info</h3>
                <form action="/detection" method="POST" >
                   <div class="gpr">
                      <input class="ini" placeholder="Enter your name" type="text"
name="person">
                   </div>


                   <div class="gpr">
                      <input class="ini "  placeholder="number of times pregnant" type="text"
name="preg">
                   </div>

                   <div class="gpr">
                      <input class="ini"  placeholder="enter glucose level" type="text"
name="glucose">
                   </div>

                    <div class="gpr">
                      <input class="ini"  placeholder="enter the bp level" type="text"  name="bp">
                   </div>

                    <div class="gpr">
                      <input class="ini" type="text"  placeholder="enter your skinthicknes"
name="skinthick">
                   </div>



                   <div class="gpr">
                      <input class="ini"  placeholder="enter the insulin level"  type="text"
name="insulin">
```

```
                </div>

                <div class="gpr">
                   <input class="ini"  placeholder="enter body mass index" type="text"
name="bmi">
                </div>

                <div class="gpr">
                   <input class="ini"  placeholder="enter the pedigree level" type="text"
name="pedigree">
                </div>

                <div class="gpr">
                   <input class="ini"   placeholder="enter your age" type="text" name="age">
                </div>

                <div class="but">
                   <button class="grn "  type="submit">submit</button>
                </div>
              </form>
 {% endblock content %}
```

diabetes.html

```
{% extends "layout.html" %}
{% block content %}
<h2 class="title">Outcome: Positive</h2>
<p>{{name }} meet the doctor immediately.</p>
{% endblock content %}
```

nodiabetes.html

```
{% extends "layout.html" %}
{% block content %}
<h2 class="title">Outcome: Negetive</h2>
{% endblock content %}
```

4.2 Results and Discussion:

By using logistic regression for predicting diabetes we got accuracy of 86% and some of the evaluation measures are listed in the table below.

Evaluation Metric	Result (%)
Accuracy	86
Recall	78
Precision	80
Specificity	90
F1-Score	77

Table 1-results

The cost function value vs. number of iterations is shown graphically below and it has to minimize to get accurate results.

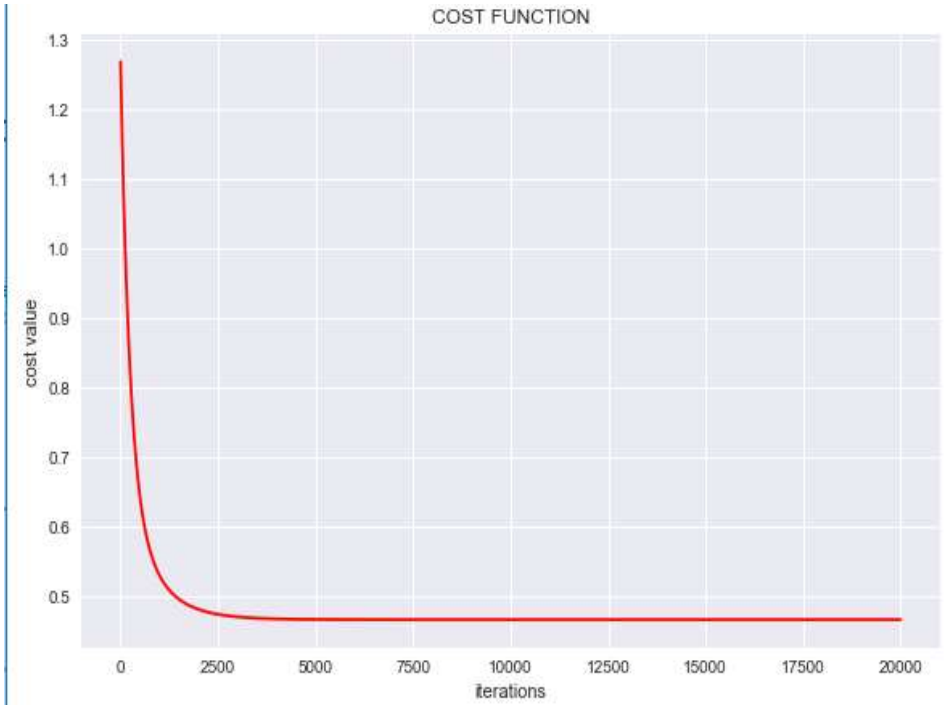


Fig-19 cost function graph

CHAPTER 5

CONCLUSION AND FUTURE SCOPE:

5.1 Conclusion:

In this we have seen how to predict diabetes using Logistic Regression algorithm. We build a model by following the steps required to build a machine learning model and we train the model by using the dataset which is separated from original for training and finally we evaluate our model by using testing dataset. This will be very useful for doctors and practitioners to predict diabetes in advance. The user interface is very helpful and it is easy to check whether there is a chance of getting diabetes by giving the required details and we get accuracy of 86%. Automated classification is the key that will be benefitted from this project.

5.2 Future Scope:

In this project we have used logistic regression algorithm for training our model to predict the diabetes and in future we can even build classifiers using some advanced algorithms like Decision Trees, Naïve Bayes and we can even improve our classification accuracy by using Artificial Neural Networks for even better results.

REFERENCES

1. Thank Daghistani and Riyad Alshammari(2020), Comparison of Statistical Logistic Regression and RandomForest Machine Learning Techniques in Predicting Diabetes, Journal of Advances in Information Technology Vol. 11, No.2

2. Bhavya, Sanjay, Suraj and Shivshankar Rao(2020), Diabetes Prediction using Machine Learning,International Journal of Advanced Research in Computer and Communication Engineering, Vol. 9.

3. Iqbal, Faruque and Asra Kalim(2020), K-Nearest Neighbor Learning based Diabetes Mellitus Prediction and Analysis for eHealth Services, Journal of Endorsed Transactions on Scalable Information Systems, Vol. 7.

4. K.VijayaKumar, Lavanya and Nirmala(2019), Random Forest Algorithm for the Prediction of Diabetes, Journal of Proceeding of International Conference on Systems computation Automation and Networking.