International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

DIABETES PREDICTION

Amina Afsa[1], Andela Vamshika[2], Mote Rohini[3]

Department of Computer Science and Engineering, Stanley College of Engineering and Technology for Women, Telangana, India

Abstract: Diabetes is among critical diseases and lots of people are suffering from this disease.A Diabetic patient suffers from a high level of blood sugar in the body.Undiagnosed diabetes may cause the nerve and kidney damage,heart and blood vessel disease,etc.Diabetes occurs when body does not make enough insulin.

Early detection of diabetes is very essential to have a healthy life . Data analysis plays an significant role in healthcare industries. Healthcare industries have large databases. One can study and analyze dataset and find hidden information, hidden patterns to discover knowledge from the data and predict outcomes accordingly.

The main objective of this study is to present a Machine Learning based solution which analyze the data and provide a predictive output which will be displayed on a front end page using html with an added feature ,chatbot created using IBM Watson integrated on the page for consultation.

Keywords: Machine learning, Logistic Regression, KNN, Random Forest, SVM, Decision Tree, Naïve Bayes, Diabetes Prediction.

1. INTRODUCTION

1.1 About Project

Diabetes occurs when the blood glucose/blood sugar level in the human body is very high. According to health experts, diabetes occurs when the human body's gland called the pancreas cannot produce enough insulin (Type 1 diabetes), and the produced insulin cannot be used by the cell of the body (Type 2 diabetes). When we eat food, after the digestion process, glucose gets released. Insulin is a blood hormone that moves from blood to cells and instructs cells to consume blood glucose and transform it into energy. When the pancreas cannot produce enough insulin, the cells cannot absorb glucose, and the glucose remains in the blood. Hence the blood glucose/blood sugar increases in the blood at a very unacceptable level. Due to high blood sugar, some symptoms arises in the human body.

Symptoms of Diabetes

- Frequent Urination
- Increased thirst
- Tired/Sleepiness
- Weight loss

• Blurred vision

• Mood swings

• Confusion and difficulty concentrating

• frequent infections

The usual range of glucose levels in the human body is **70 to 99mg per decilitre**. If the glucose level is more than **126 mg/dL**, it indicates diabetes. A person is considered to have prediabetes if body glucose concentration is **100 to 125 mg/dL**. If the human body's blood sugar level becomes too high, the impending complications can be heart disease, kidney failure, stroke, and nerve damage. There is no permanent cure for diabetes. Maintaining an effective fitness system and balanced diet can help to prevent diabetes.

Machine learning uses various algorithms to learn from the parsed data and make predictions.

1.2 Objectives of the Project

- Easy to use: The primary objective of this project is to broaden a platform so as to be simple and smooth to apply, as right here one have to provide the patient's scientific details and primarily based on the features extracted the algorithm will then discover diabetes. As right here set of rules does the task hence a well trained version is much less certain to make errors in predicting diabetes consequently, in short accuracy is advanced and thereby it additionally saves time and makes simpler for doctors in addition to sufferers to expect whether or not they may be vulnerable to diabetes or not, that is otherwise we difficult to do without health practitioner's involvement.
- No human intervention required: To predict diabetes one must provide scientific details which includes age, BMI, and so on. And right here the set of rules will offer the effects based on the capabilities extracted and consequently here probabilities of mistakes been made are very minimal given that there is no human intervention and it also saves lot of time for the sufferers or doctors and they could similarly continue for treatments or different tactics should quicker. This is in case whilst consequences are provided quicker to them. This can in-turn make the precaution/prevention for diabetes faster while it saves medical doctors and affected person the essential time, to be able to cross on to in addition treatments and precautions to be taken to minimize the effect of that diabetes.
- Not simplest hit upon the diabetes kind but additionally suggest precautions: In this mission our goal isn't only to find diabetes but pin point towards the precautions to be taken to minimize the impact of the diabetes. Getting hints on precautions to be taken will help the doctors and sufferers to progress without problems to similarly steps of their treatment.

1.3 Scope of the Project

The early intervention of diabetes can reduce the prevalence of diabetes and hence the economic burden due to it. There is no need of regional studies for diabetes prediction in India. Machine learning techniques play an important role in treatment plan workout, rehabilitation, chronic diseases management plan etc. Long term follow up plan may be easily guided and keen supervision is possible. The systems may definitely helpful in

reduction of cost of patient management by avoiding unnecessary investigations and patients follow up. These prediction systems will add accuracy and time management. Computer-based patient support systems benefit patients by providing informational support that increases their participation in health care.

1.4 Advantages

- Easily identifies trends and patterns: Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans. For instance, for an e-commerce website like Amazon, it serves to understand the browsing behaviors and purchase histories of its users to help cater to the right products, deals, and reminders relevant to them. It uses the results to reveal relevant advertisements to them.
- No human intervention needed (automation): With ML, you don't need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own. A common example of this is anti-virus softwares; they learn to filter new threats as they are recognized. ML is also good at recognizing spam.
- Continuous Improvement: As ML algorithms gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.
- Handling multi-dimensional and multi-variety data: Machine Learning algorithms are good at handling data that are multi-dimensional and multi-variety, and they can do this in dynamic or uncertain environments.
- Wide Applications: You could be an e-tailer or a healthcare provider and make ML work for you. Where it does apply, it holds the capability to help deliver a much more personal experience to customers while also targeting the right customers.

1.5 Disadvantages

- Data Acquisition: Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times where they must wait for new data to be generated.
- Time and Resources: ML needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.
- Interpretation of Results: Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose
- High error-susceptibility: Machine Learning is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions coming from a biased training set. This leads to irrelevant advertisements being displayed to customers. In the case of ML, such blunders can set off a chain of errors that can go undetected for long periods of time. And when they do get noticed, it takes quite some time to recognize the source of the issue, and even longer to correct it.

1.6 Applications

- Social Media Features: Social media platforms use machine learning algorithms and approaches to create some attractive and excellent features. For instance, Facebook notices and records your activities, chats, likes, and comments, and the time you spend on specific kinds of posts. Machine learning learns from your own experience and makes friends and page suggestions for your profile.

- Product Recommendations: Product recommendation is one of the most popular and known applications of machine learning. Product recommendation is one of the stark features of almost every e-commerce website today, which is an advanced application of machine learning techniques. Using machine learning and AI, websites track your behavior based on your previous purchases, searching patterns, and cart history, and then make product recommendations.

- Image Recognition: Image recognition, which is an approach for cataloging and detecting a feature or an object in the digital image, is one of the most significant and notable machine learning and AI techniques. This technique is being adopted for further analysis, such as pattern recognition, face detection, and face recognition.

- Sentiment Analysis: Sentiment analysis is one of the most necessary applications of machine learning. Sentiment analysis is a real-time machine learning application that determines the emotion or opinion of the speaker or the writer. For instance, if someone has written a review or email (or any form of a document), a sentiment analyzer will instantly find out the actual thought and tone of the text. This sentiment analysis application can be used to analyze a review based website, decision-making applications, etc.

- Automating Employee Access Control: Organizations are actively implementing machine learning algorithms to determine the level of access employees would need in various areas, depending on their job profiles. This is one of the coolest applications of machine learning.

- Marine Wildlife Preservation: Machine learning algorithms are used to develop behavior models for endangered cetaceans and other marine species, helping scientists regulate and monitor their populations.

- Regulating Healthcare Efficiency and Medical Services: Significant healthcare sectors are actively looking at using machine learning algorithms to manage better. They predict the waiting times of patients in the emergency waiting rooms across various departments of hospitals. The models use vital factors that help define the algorithm, details of staff at various times of day, records of patients, and complete logs of department chats and the layout of emergency rooms. Machine learning algorithms

also come to play when detecting a disease, therapy planning, and prediction of the disease situation. This is one of the most necessary machine learning applications.

- Predict Potential Heart Failure: An algorithm designed to scan a doctor's free-form e-notes and identify patterns in a patient's cardiovascular history is making waves in medicine. Instead of a physician digging through multiple health records to arrive at a sound diagnosis, redundancy is now reduced with computers making an analysis based on available information.

- Banking Domain: Banks are now using the latest advanced technology machine learning has to offer to help prevent fraud and protect accounts from hackers. The algorithms determine what factors to consider to create a filter to keep harm at bay. Various sites that are unauthentic will be automatically filtered out and restricted from initiating transactions.

- Language Translation: One of the most common machine learning applications is language translation. Machine learning plays a significant role in the translation of one language to another. We are amazed at how websites can translate from one language to another effortlessly and give contextual meaning as well. The technology behind the translation tool is called 'machine translation.' It has enabled people to interact with others from all around the world; without it, life would not be as easy as it is now. It has provided confidence to travelers and business associates to safely venture into foreign lands with the conviction that language will no longer be a barrier. Your model will need to be taught what you want it to learn. Feeding relevant back data will help the machine draw patterns and act accordingly. It is imperative to provide relevant data and feed files to help the machine learn what is expected. In this case, with machine learning, the results you strive for depend on the contents of the files that are being recorded.

1.7 Hardware and Software Requirements

Hardware Requirements

System Processor: Intel i5core

Hard Disk: 100GB

Ram: 4GB

Software Requirements

Operating System: Window 10

Programming Language: Python

IDE: Jupyter/Spyder/Anaconda

2. LITERATURE SURVEY

2.1 Existing System

Data mining approaches were studied in existing system. Diabetes prediction using algorithms such as branch and bound algorithm was proposed. A basic diabetic dataset is choosen for carrying out the comparative analysis.

Issues in Existing System

- The accuracy of detection is less.
- High false positives.
- Increased False Positive rate resulted in low accuracy.
- False Negative rate was low but it affects the accuracy.
- There is no interactive tool for users to predict diabetes.

2.2 Proposed System

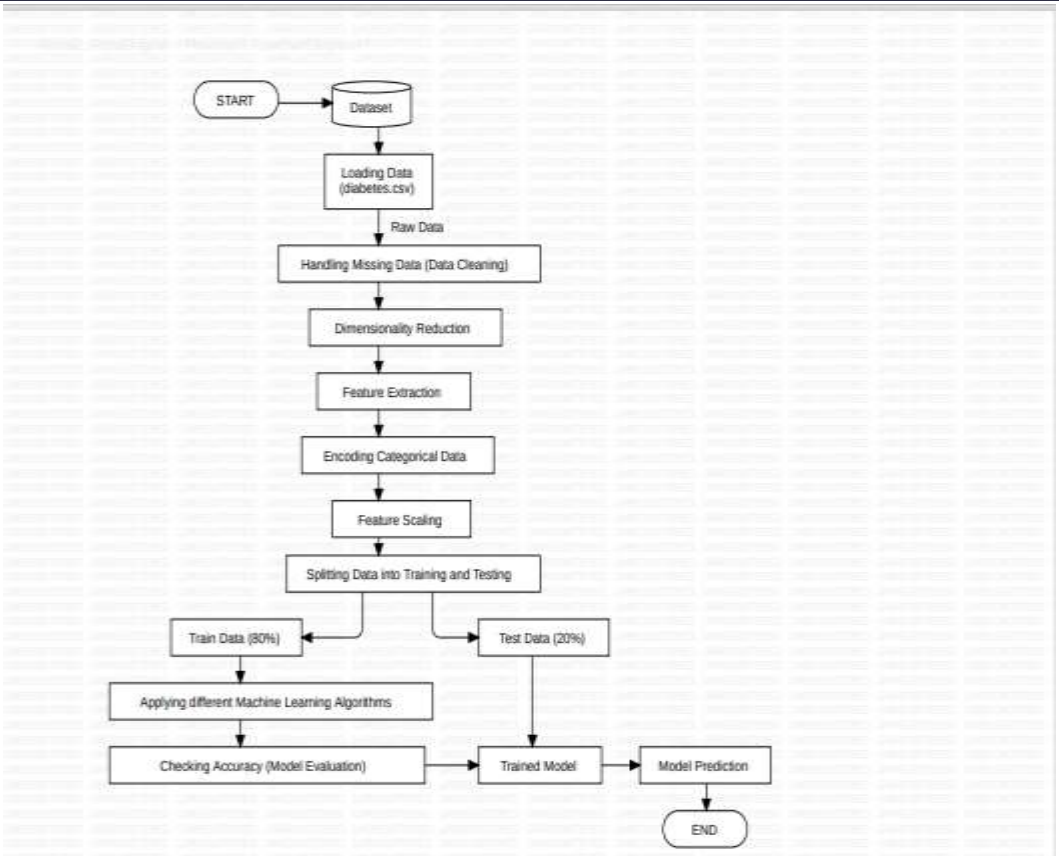
The proposed system is a machine learning application to classify diabetic or non-diabetic. In many cases, the performance of algorithm is high in the context of speed. The main objective of our model is to achieve high accuracy.

Advantages

- One of the biggest advantages of these algorithms is its versatility. It can be used for both regression and classification tasks.
- Random forest algorithm builds multiple decision trees and merges them together to get a more accurate and stable prediction.
- The random forest algorithm also works well when data possess missing values.

3. PROPOSED ARCHITECTURE

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org



4. IMPLEMENTATION

4.1 Algorithm

4.1.1 Logistic Regression

Logistic regression is also a supervised learning classification algorithm. It is used to estimate the probability of a binary response based on one or more predictors. They can be continuous or discrete. Logistic regression used when we want to classify or distinguish some data items into categories. It classify the data in binary form means only in 0 and 1 which refer case to classify patient that is positive or negative for diabetes.

Main aim of logistic regression is to best fit which is responsible for describing the relationship between target and predictor variable. Logistic regression is a based on Linear regression model. Logistic regression model uses sigmoid function to predict probability of positive and negative class.

Sigmoid function P = 1/1+e – (a+bx) Here P = probability, a and b = parameter of Model.

4.1.2 K-Nearest Neighbor

KNN is also a supervised machine learning algorithm. KNN helps to solve both the classification and regression problems. KNN is lazy prediction technique. KNN assumes that similar things are near to each other. Many times data points which are similar are very near to each other. KNN helps to group new work based on similarity measure. KNN algorithm record all the records and classify them according to their similarity measure. For finding the distance between the points uses tree like structure. To make a prediction for a new data point, the algorithm finds the closest data points in the training data set its nearest neighbors. Here K= Number of nearby neighbors, it's always a positive integer. Neighbors value is chosen from set of class. Closeness is mainly defined in terms of Euclidean distance.

Algorithm

- Take a sample dataset of columns and rows named as Pima Indian Diabetes data set.
- Take a test dataset of attributes and rows.
- Find the Euclidean distance by the help of formula
- Then, Decide a random value of K. is the no. of nearest neighbors
- Then with the help of these minimum distance and Euclidean distance find out the nth column of each.
- Find out the same output values.
- If the values are same, then the patient is diabetic, other- wise not.

4.1.3 Decision Tree

Decision tree is a basic classification method. It is supervised learning method. Decision tree used when response variable is categorical. Decision tree has tree like structure based model which describes classification process based on input feature. Input variables are any types like graph, text, discrete, continuous etc.

Steps for Decision Tree Algorithm

- Construct tree with nodes as input feature.
- Select feature to predict the output from input feature whose information gain is highest.
- The highest information gain is calculated for each attribute in each node of tree.
- Repeat step 2 to form a subtree using the feature which is not used in above node.

4.1.4 Random Forest

It is type of ensemble learning method and also used for classification and regression tasks. The accuracy it gives is grater then compared to other models. This method can easily handle large datasets. Random Forest is developed by Leo Bremen. It is popular ensemble Learning Method. Random Forest Improve Performance of Decision Tree by reducing variance. It operates by constructing a multitude of decision trees at training time and outputs the class that is the mode of the classes or classification or mean prediction (regression) of the individual trees.

Algorithm

- The first step is to select the R features from the total features m where R<<M.
- Among the R features, the node using the best split point.
- Split the node into sub nodes using the best split.
- Repeat a to c steps until l number of nodes has been reached.
- Built forest by repeating steps a to d for a number of times to create n number of trees.

4.1.5 Support Vector Machine

Support Vector Machine also known as SVM is a supervised machine learning algorithm.

SVM is most popular classification technique. SVM creates a hyperplane that separate two classes. It can create a hyperplane or set of hyperplane in high dimensional space. This hyper plane can be used for classification or regression also. SVM differentiates instances in specific classes and can also classify the entities which are not sup- ported by data.

Separation is done by through hyperplane performs the separation to the closest training point of any class.

Algorithm

- Select the hyper plane which divides the class bet- ter.
- To find the better hyper plane you have to calcu- late the distance between the planes and the data which is called Margin.
- If the distance between the classes is low then the chance of miss conception is high and vice versa. So we need to
- Select the class which has the high margin. Margin = distance to positive point + Distance to negative point.

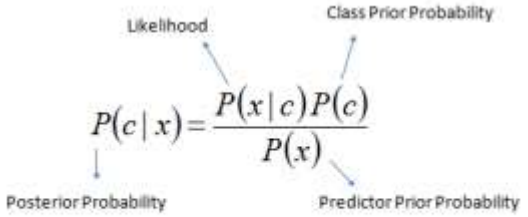
4.1.6 Naïve Bayes

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability that this fruit is an apple and that is why it is known as 'Naive'.

Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:



$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Above,

- $P(c|x)$ is the posterior probability of *class* (c, *target*) given *predictor* (x, *attributes*).
- $P(c)$ is the prior probability of *class*.
- $P(x|c)$ is the likelihood which is the probability of *predictor* given *class*.
- $P(x)$ is the prior probability of *predictor*.

Let's understand it using an example. Below I have a training data set of weather and corresponding target variable 'Play' (suggesting possibilities of playing). Now, we need to classify whether players will play or not based on weather condition. Let's follow the below steps to perform it.

Step 1: Convert the data set into a frequency table

Step 2: Create Likelihood table by finding the probabilities like Overcast probability = 0.29 and probability of playing is 0.64.

Step 3: Now, use Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction

4.2 Code Implementation

IMPORT LIBRARIES

```
import numpy as np
import pandas as pd
```

LOADING DATASET

```
dataset=pd.read_csv("https://raw.githubusercontent.com/anujvyas/Diabetes-Prediction-Deployment/master/kaggle_diabetes.csv")
dataset.head()
```

DATA PREPROCESING

```
dataset.isnull().any()
dataset["Outcome"].unique()
```

LABEL ENCODING

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
dataset["Outcome"]=le.fit_transform(dataset["Outcome"])
dataset
dataset.head(1)
```

SPLITTING DATASET INTO TRAIN & TEST

```
x=dataset.iloc[:,0:8].values
y=dataset.iloc[:,8:9].values
x[0]
x.shape
y.shape
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

STANDARD SCALING

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

DECISION TREE CLASSIFIER

```
from sklearn.tree import DecisionTreeClassifier
dec=DecisionTreeClassifier(random_state=0,criterion="entropy")
```

```
dec.fit(x_train,y_train)
dec_diab_pred=dec.predict(x_test)
dec_diab_pred
# from sklearn.metrics import accuracy_score

dec_acc=accuracy_score(y_test,dec_diab_pred)
```

DECISION TREE ACCURACY

```
dec_acc
```

CONFUSION MATRIX

```
from sklearn.metrics import confusion_matrix
dec_cm=confusion_matrix(y_test,dec_diab_pred)


dec_cm
```

ROC-AUC CURVE

```
import sklearn.metrics as metrics
fpr,tpr,threshold=metrics.roc_curve(y_test,dec_diab_pred)
roc_auc=metrics.auc(fpr,tpr)
import matplotlib.pyplot as plt
plt.plot(fpr,tpr,"b",label="auc=%0.2f"%roc_auc)
plt.legend(loc='lower right')
plt.title("ROC")
plt.xlabel("FPR")
plt.ylabel("TPR")
```

RANDOM FOREST

```
from sklearn.ensemble import RandomForestClassifier
ran= RandomForestClassifier(n_estimators=10,random_state=0,criterion="entropy")
ran.fit(x_train,y_train)
ran.fit(x_train,y_train)
ran_diab_pred=ran.predict(x_test)
y_test
ran_diab_pred
```

```
ran_acc=accuracy_score(y_test,ran_diab_pred)
```

RANDOM FOREST ACCURACY

```
ran_acc
ran_cm=confusion_matrix(y_test,ran_diab_pred)
```

CONFUSION MATRIX

```
ran_cm
```

ROC-AUC CURVE

```
import sklearn.metrics as metrics
ranfpr,rantpr,threshold=metrics.roc_curve(y_test,ran_diab_pred)
ran_roc_auc=metrics.auc(ranfpr,rantpr)
import matplotlib.pyplot as plt
plt.plot(ranfpr,rantpr,"b",label="auc=%0.2f"%ran_roc_auc)
plt.legend(loc='lower right')
plt.title("ROC")
plt.xlabel("FPR")
plt.ylabel("TPR")
```

LOGISTIC REGRESSION

```
from sklearn.linear_model import LogisticRegression
log=LogisticRegression()
log.fit(x_train,y_train)
log_diab_pred=log.predict(x_test)
log_acc=accuracy_score(y_test,log_diab_pred)
```

LOGISTIC REGRESSION ACCURACY

```
log_cm=confusion_matrix(y_test,log_diab_pred)
```

CONFUSION MATRIX

```
log_cm
```

ROC-AUC CURVE

```
import sklearn.metrics as metrics
logfpr,logtpr,threshold=metrics.roc_curve(y_test,log_diab_pred)
```

```
log_roc_auc=metrics.auc(logfpr,logtpr)
import matplotlib.pyplot as plt
plt.plot(logfpr,logtpr,"b",label="auc=%0.2f"%log_roc_auc)
plt.legend(loc='lower right')
plt.title("ROC")
plt.xlabel("FPR")
plt.ylabel("TPR")
```

K-NEAREST NEIGHBOR

```
from sklearn.neighbors import KNeighborsClassifier
knn=KNeighborsClassifier(n_neighbors=5,metric="euclidean")
knn.fit(x_train,y_train)
knn_diab_pred=knn.predict(x_test)
knn_diab_pred
knn_acc=accuracy_score(y_test,knn_diab_pred)
```

KNN ACCURACY

```
knn_acc
knn_cm=confusion_matrix(y_test,knn_diab_pred)
```

CONFUSION MATRIX

```
knn_cm
import sklearn.metrics as metrics
knnfpr,knntpr,threshold=metrics.roc_curve(y_test,knn_diab_pred)
knn_roc_auc=metrics.auc(knnfpr,knntpr)
```

ROC-AUC CURVE

```
import matplotlib.pyplot as plt
plt.plot(knnfpr,knntpr,"b",label="auc=%0.2f"%knn_roc_auc)
plt.legend(loc='lower right')
plt.title("ROC")
plt.xlabel("FPR")
plt.ylabel("TPR")
```

NAIVE BAYES

```
from sklearn.naive_bayes import GaussianNB
```

```
nb=GaussianNB()
nb.fit(x_train,y_train)
nbpred=nb.predict(x_test)
nbpred
from sklearn.metrics import accuracy_score
nbacc=accuracy_score(y_test,nbpred)
```

NAIVE BAYES ACCURACY

```
nbacc
from sklearn.metrics import confusion_matrix
nbcm=confusion_matrix(y_test,nbpred)
```

CONFUSION MATRIX

```
nbcm
import sklearn.metrics as metrics
nbfpr,nbtpr,threshold=metrics.roc_curve(y_test,nbpred)
nbroc_auc=metrics.auc(nbfpr,nbtpr)
```

ROC-AUC CURVE

```
import matplotlib.pyplot as plt
plt.plot(nbfpr,nbtpr,label="auc=%0.2f"%nbroc_auc)
plt.legend(loc="lower right")
```

SUPPORT VECTOR MACHINE

```
from sklearn.svm import SVC
svm=SVC(kernel="linear",random_state=0)
svm.fit(x_train,y_train)
svmpred=svm.predict(x_test)
svm_acc=accuracy_score(y_test,svmpred)
```

SVM ACCURACY

```
svm_acc
```

CONFUSION MATRIX

```
from sklearn.metrics import confusion_matrix
svm_cm=confusion_matrix(y_test,nbpred)
```

```
svm_cm
import sklearn.metrics as metrics
svmfpr,svmtpr,threshold=metrics.roc_curve(y_test,svmpred)
svmroc_auc=metrics.auc(svmfpr,svmtpr)
```

ROC_AUC CURVE

```
import matplotlib.pyplot as plt
plt.plot(svmfpr,svmtpr,label="auc=%0.2f"%svmroc_auc)
plt.legend(loc="lower right")
dec_y=dec.predict(sc.transform([[4,128,90,28,0,33,0.354,60]]))
dec_y
ran_y=ran.predict(sc.transform([[4,128,90,28,0,33,0.354,60]]))
ran_y
log_y=log.predict(sc.transform([[4,128,90,28,0,33,0.354,60]]))
log_y
knn_y=knn.predict(sc.transform([[4,128,90,28,0,33,0.354,60]]))
knn_y
nb_y=nb.predict(sc.transform([[4,128,90,28,0,33,0.354,60]]))
nb_y
svm_y=svm.predict(sc.transform([[4,128,90,28,0,33,0.354,60]]))
svm_y
```

INDEX

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
  <meta charset="utf-8">
  <title>Diabetes Predictor</title>
<link rel="shortcut icon" href="{{ url_for('static', filename='diabetes-favicon.ico') }}"
```

```
<link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='styles.css') }}">

<script src="https://kit.fontawesome.com/5f3f547070.js"
crossorigin="anonymous"></script>

<link href="https://fonts.googleapis.com/css2?family=Pacifico&display=swap"
rel="stylesheet">

 </head>

   <body>

 <!-- Website Title -->

<div class="container">

<h2 class='container-heading'><span class="heading_font">Diabetes Predictor</span></h2>

<div class='description'>

<p>A Machine Learning Web App, Built with Flask, Deployed using Heroku.</p>

</div> </div>

 <!-- Text Area -->

<div class="ml-container">

<form action="{{ url_for('predict') }}" method="POST">

 <input class="form-input" type="text" name="pregnancies" placeholder="Number of
Pregnancies eg. 0"><br>

 <input class="form-input" type="text" name="glucose" placeholder="Glucose (mg/dL) eg.
80"><br>

 <input class="form-input" type="text" name="bloodpressure" placeholder="Blood Pressure
(mmHg) eg. 80"><br>

 <input class="form-input" type="text" name="skinthickness" placeholder="Skin Thickness
(mm) eg. 20"><br>
```

<input class="form-input" type="text" name="insulin" placeholder="Insulin Level (IU/mL) eg. 80">

<input class="form-input" type="text" name="bmi" placeholder="Body Mass Index (kg/m²) eg. 23.1">

<input class="form-input" type="text" name="dpf" placeholder="Diabetes Pedigree Function eg. 0.52">

<input class="form-input" type="text" name="age" placeholder="Age (years) eg. 34">

<input type="submit" class="my-cta-button" value="Predict">

</form>

</body>

</html>

RESULT

<!DOCTYPE html>

<html lang="en" dir="ltr"> <head>

<meta charset="utf-8">

<title>Diabetes Predictor</title>

<link rel="shortcut icon" href="{{ url_for('static', filename='diabetes-favicon.ico') }}">

<link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='styles.css') }}">

<script src="https://kit.fontawesome.com/5f3f547070.js" crossorigin="anonymous"></script>

<linkhref="https://fonts.googleapis.com/css2?family=Pacifico&display=swap"rel="stylesheet">

</head> <body>

```
<!-- Website Title -->

<div class="container">

<h2 class='container-heading'>

<span class="heading_font">Diabetes Predictor</span></h2>

<div class='description'>

<p>A Machine Learning Web App, Built with Flask, Deployed using Heroku.</p>

</div> </div>

<!-- Result --> <div class="results"> {% if prediction==1 %}

<h1>Prediction: <span class='danger'>Oops! You have DIABETES.</span></h1>

<img class="gif" src="{{ url_for('static', filename='diabetes.webp') }}" alt="Diabetes Image">

{% elif prediction==0 %}

<h1>Prediction: <span class='safe'>Great! You DON'T have diabetes.</span></h1>

<img class="gif1" src="{{ url_for('static', filename='no-diabetes.webp') }}" alt="Not Diabetes Image">

{% endif %}  </div>   </body>  </html>
```

5. RESULTS

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

Fig 5.1 Diabetes Predictor

Fig 5.2 Taking User Input

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

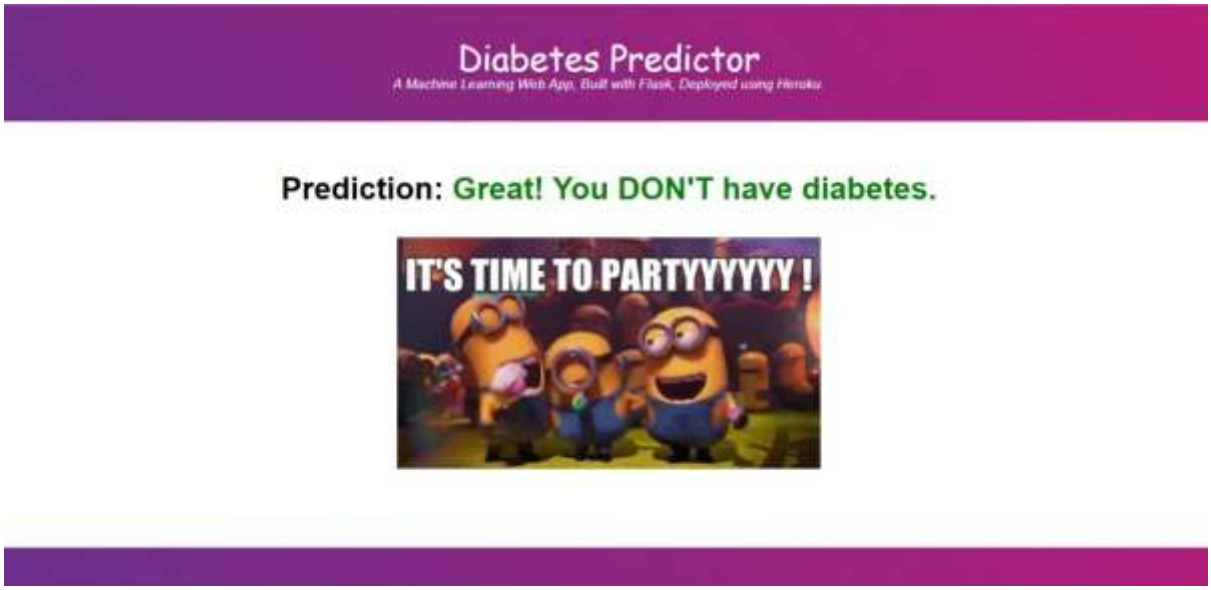


Fig 5.3 User Predicted as Non-Diabetic

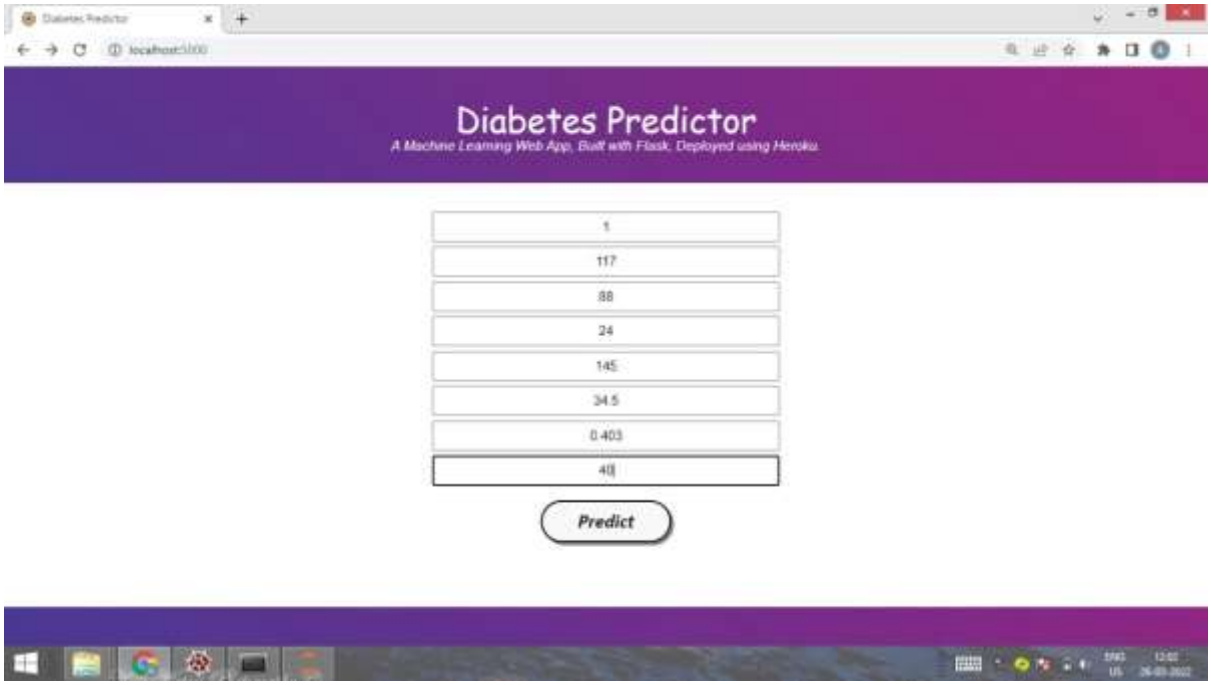


Fig 5.4 Taking User Input

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

Fig 5.5 User Predicted as Diabetic

Fig 5.6 Diabetic Chat Bot Assistant & Online Consultancy

```
ALL ALGORITHM'S ACCURACY :
                          DECISION TREE       : 0.88
                          RANDOM FOREST       : 0.95
                          LOGISTIC REGRESSION : 0.78
                          K NEAREST NEIGHBOR  : 0.79
                          NAIVE BAYES         : 0.77
                          SUPPORT VECTOR      : 0.77
```

Fig 5.7 Accuracy

6. CONCLUSION

In this project we have used different types of machine learning algorithm for detection of diabetes. We implemented machine learning algorithms on the dataset and performed classification to signify the best machine learning algorithm for diabetes prediction on the bases of old data available. The higher accuracy the better prediction rate we will achieve. The random forest algorithm obtained the best accuracy. The overall experimentation displayed that random forest is better than other algorithms in diabetes prediction that is the accuracy of random forest is 95%.whereas the decision tree has got 88%, logistic regression got 78%,k nearest neighbor go 79%, naive bayes got 77%, support vector got 77%.

7. FUTURE SCOPE

Healthcare professions found it hard to find healthcare data and perform analysis on them due to lack of tools, resources. But using ML, we can overcome this and can perform analysis on real-time data leading to better modeling , predictions. This enhances and improves overall healthcare services. Now, IoTs being integrated with ML in order to make smart healthcare devices that sense if there is any change in the person’s body, health data that he uses the device(Pacemaker, Stethoscope, etc.) and this will notify the person regarding this through app. This helps in monitoring, advanced prediction and analysis thereby reducing errors, saving time and life of people.

8. REFERENCES

1. [1] Aiswarya Iyer, S. Jeyalatha and Ronak Sumbaly," Diagnosis of Diabetes Using Classification Mining Techniques", International Journal of Data Mining & Knowledge Management Process (IJDKP) Vol.5, No.1, January 2015. "Diabetes Care Decision Support System" 2nd International Conference on Industrial and Information Systems IEEE 2010
2. [2] A.K., Dewangan, and P., Agrawal, "Classification of Diabetes Mellitus Using Machine Learning Techniques," International Journal of Engineering and Applied Sciences, vol. 2, 2015.
3. [3] Ayush Anand and Divya Shakti," Prediction of Diabetes Based on Personal Lifestyle Indicators", 1st International Conference on Next Generation Computing Technologies, 978-1-4673-6809-4, September 2015.
4. [4] B. Nithya and Dr. V. Ilango," Predictive Analytics in Health Care Using Machine Learning Tools and Techniques", International Conference on Intelligent Computing and Control Systems, 978-1-5386-2745-7,2017.
5. [5] Dr Saravana kumar N M, Eswari T, Sampath P and Lavanya S," Predictive Methodology for Diabetic Data Analysis in Big Data", 2nd International Symposium on Big Data and Cloud Computing,2015.
6. [6] K.VijiyaKumar, B.Lavanya, I.Nirmala, S.Sofia Caroline, "Random Forest Algorithm for the Prediction of Diabetes ".Proceeding of International Conference on Systems Computation Automation and Networking, 2019.
7. [7] P. Suresh Kumar and S. Pranavi "Performance Analysis of Machine Learning Algorithms on Diabetes Dataset using Big Data Analytics", International Conference on Infocom Technologies and Unmanned Systems, 978-1-5386-0514-1, Dec. 18-20, 2017.
8. [8] Rani, A. S., & Jyothi, S. (2016, March). Performance analysis of classification algorithms under different datasets. In Computing for Sustainable Global Development (INDIACom), 2016 3rd International Conference on (pp. 1584- 1589). IEEE.
9. [9] Nahla B., Andrew et al,"Intelligible support vector machines for diagnosis of diabetes mellitus. Information Technology in Biomedicine", IEEE Transactions. 14, (July. 2010), 1114-20.