Diabetes Prediction

A PROJECT REPORT

Submitted by

BL.EN.U4AIE21121 Srinidhi Kannan
BL.EN.U4AIE21126 Suryamritha M
BL.EN.U4AIE21139 Varshini Balaji

BACHELOR OF TECHNOLOGY

IN

Computer Science (AIE)

AMRITA SCHOOL OF ENGINEERING, BANGALORE

AMRITA VISHWA VIDYAPEETHAM

BANGALORE 560 035

July-2022

AMRITA VISHWA VIDYAPEETHAM

AMRITA SCHOOL OF ENGINEERING, BANGALORE, 5600

ABSTRACT

Diabetes is a disease caused due to an increased level of glucose in the blood. Many people around the world suffer from this condition. A lot of research works have been published concerning diabetes, using a variety of classifiers and machine learning models, both for predictions and diagnosis. This paper outlines the effects of some risk factors of diabetes, specifically the effect of BMI, blood pressure and physical activity, using machine learning models. ML algorithms were applied to the dataset which consisted information of on Pregnancy, Glucose, blood pressure, SkinThikness, Insulin, BMI, DiabetesPedigreeFunction, Age, and Outcome.. The analysis is supported by logistic regression, SVC, RandomForestClassifier, GradientBoostingClassifier, and KNeighborsClassifier.The classifier that gives the highest accuracy is used for the prediction of diabetes.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Introduction

Diabetes is a disease that occurs when your blood glucose, also called blood sugar, is too high. Insulin, a hormone made by the pancreas, helps glucose from food get into your cells to be used for energy.

1.2 Causes

- Diabetes causes vary depending on your
- genetic makeup
- family history
- ethnicity, health
- environmental factors. There is no common diabetes cause that fits every type of diabetes as the causes of diabetes vary depending on the individual and the type.

1.3 Types of Diabetes

1.31 Type 1 Diabetes

- Pancreas produces little or no insulin.
- Including genetics and some viruses, may contribute to type 1 diabetes
- type 1 diabetes has no cure.

1.32 Type 2 diabetes

- Results in too much sugar circulating in the bloodstream.
- lead to disorders of the circulatory, nervous and immune systems
- In type 2 diabetes, there are primarily two interrelated problems:
- pancreas does not produce enough insulin
- cells respond poorly to insulin

- There's no cure for type 2 diabetes

1.33 Gestational diabetes

- diagnosed for the first time during pregnancy (gestation).
- affects how your cells use sugar (glucose). causes high blood
- affect pregnancy and baby's health.
- blood sugar returns to its usual level soon after delivery if first time .
- But if you've had gestational diabetes, you have a higher risk of getting type 2 diabetes.

1.4 Risk Factors

1.41 Type 1and Type 2 diabetes

- Weight
- Inactivity.
- Family history
- Race or ethnicity
- Age
- Gestational diabetes.
- Polycystic ovary syndrome
- High blood pressure.
- Abnormal cholesterol and triglyceride levels.

1.42 Gestational diabetes

- Complications in baby
- Excess growth
- Low blood sugar
- Type 2 diabetes later in life
- Death.
- Complications in the mother
- Preeclampsia.
- Subsequent gestational diabetes

CHAPTER 2

LITERATURE SURVEY

Paper[1],Diabetes is emerging as a predominant disease in the developing countries like India. The disease is becoming very serious and cause many other problems in the human body. Many factors are remaining as a cause for this disease in human body. The disease is not curable and can only be controlled. In this paper, support vector machine [SVM] is applied in diabetes prediction. The performance of SVM algorithm is analyzed for different available kernels. The best kernel is selected and used for prediction. The proposed work is implemented in python programming language and its performance is as good as other algorithms.

Paper[2], sklearn is used to create a model for the Pima Indians Diabetes Dataset, which is very popular diabetes study, and involves data from Pima women, who are very common to have diabetes. The cardinal factor of this dataset is that the features are physical factors rather than dependent on region of the women. To successfully predict and diagnose diabetes, we worked on finding the best suited algorithm for this purpose. The main goal is to compare the different algorithms to obtain the best accuracy.

CHAPTER 3

TESTS FOR DIABETES

3.1A1C Test

The A1C test—also known as the hemoglobin A1C or HbA1c test—is a simple blood test that measures your average blood sugar levels over the past 3 months. It's one of the commonly used tests to diagnose prediabetes and diabetes, and is also the main test to help you and your health care team manage your diabetes.

3.2 Fasting Blood Sugar Test

This measures your blood sugar after an overnight fast (not eating). A fasting blood sugar level of 99 mg/dL or lower is normal, 100 to 125 mg/dL indicates you have prediabetes, and 126 mg/dL or higher indicates you have diabetes.

3.3 Glucose Tolerance Test

This measures your blood sugar before and after you drink a liquid that contains glucose. You'll fast (not eat) overnight before the test and have your blood drawn to determine your fasting blood sugar level. Then you'll drink the liquid and have your blood sugar level checked 1 hour, 2 hours, and possibly 3 hours afterward. At 2 hours, a blood sugar level of 140 mg/dL or lower is considered normal, 140 to 199 mg/dL indicates you have prediabetes, and 200 mg/dL or higher indicates you have diabetes.

3.4 Random Blood Sugar Test

This measures your blood sugar at the time you're tested. You can take this test at any time and don't need to fast (not eat) first. A blood sugar level of 200 mg/dL or higher indicates you have diabetes. If your doctor thinks you have type 1 diabetes, your blood may also tested for autoantibodies (substances that indicate your body is attacking itself) that are often present in type 1 diabetes but not in type 2 diabetes. You may have your urine tested for ketones (produced when your body burns fat for energy), which also indicate type 1 diabetes instead of type 2 diabetes.

CHAPTER 4

AI IN DIABETES

4.1 AI in Diabetes

- Principles of machine learning have been used to build algorithms to support predictive models for the risk of developing diabetes.
- AI allows a continuous and burden-free remote monitoring of the patient's symptoms and biomarkers. Further, social media and online communities enhance patient engagement in diabetes care.
- AI will introduce a paradigm shift in diabetes care from conventional management strategies to building targeted data-driven precision care.

4.2 AI in Diabetes Care

- The continuous glucose monitor doesn’t require finger sticks. It is a small device under the skin that can simply be scanned using a smartphone app.
- use wearable technology such as a Fitbit or Samsung Health to track your stats to inform you of any prevention methods, such as measuring sugar content in your daily meals and advising how to eat healthily.
- Food apps such as Ascensia Diabetes Care have been designed to create personalized diet plans specifically for diabetics. In the app, users can create personalized food choices to set up a plan that suits them rather than being told what they need to eat.

CHAPTER 5

ISLET TRANSPLANTATION

5.1 Islet Transplantation

- Pancreatic islets, also called islets of Langerhans, are groups of cells in your pancreas.
- People with type 1 diabetes must take insulin because their bodies no longer make this hormone.
- Pancreatic islet transplantation is an experimental treatment for type 1 diabetes**.**

5.2 Procedure

- Doctors take islets with healthy beta cells from the pancreas of a deceased organ donor.
- Doctors then inject the healthy islet cells taken from the donor into a vein that carries blood to the liver of a person with type 1 diabetes.
- A person receiving a transplant is called a recipient. These islets begin to make and release insulin in the recipient's body.
- More than one injection of transplanted islet cells is often needed to stop using insulin**.**

5.3 Benefits

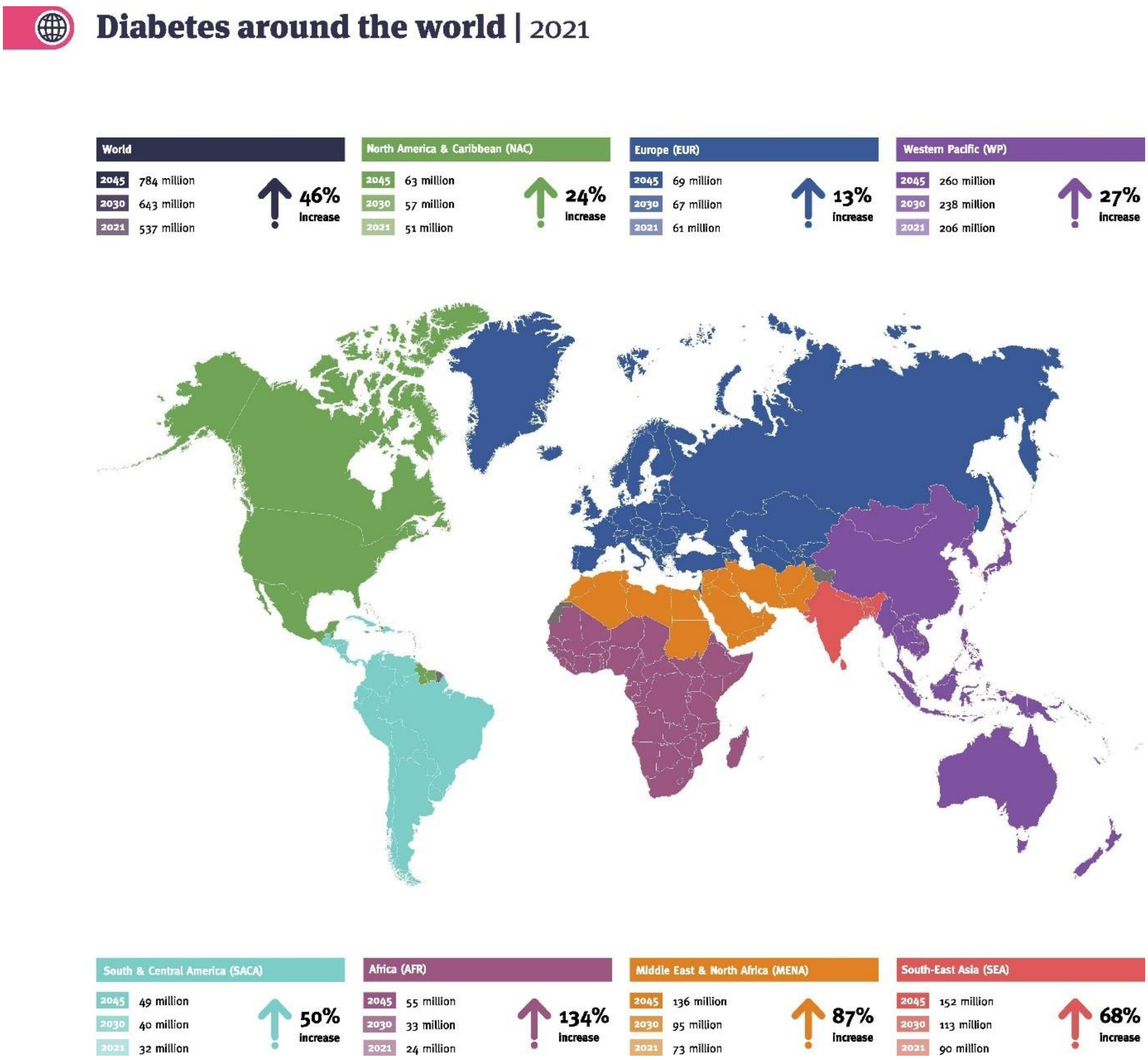
- Improve their blood glucose levels
- lower or remove the need for insulin injections
- better recognize symptoms of low blood glucose, also called hypoglycemia
- prevent severe hypoglycemia, which is when a person's blood glucose level becomes so low that he or she needs help from another person to treat the hypoglycemia

CHAPTER 6

DIABETES IN WORLD

6.1 Introduction

About 422 million people worldwide have diabetes, the majority living in low-and middle-income countries, and 1.5 million deaths are directly attributed to diabetes each year. Both the number of cases and the prevalence of diabetes have been steadily increasing over the past few decades.



CHAPTER 7

DIABETES PREDICTION USING MACHINE LEARNING WITH PYTHON

7.1 Introduction

Diabetes prediction using machine learning with python is used in this work.A dataset with the information on Pregnancy, Glucose, blood pressure, SkinThikness, Insulin, BMI, DiabetesPedigreeFunction, Age, and Outcome. The outcome says if the person is diabetic or not diabetic.1 represents Diabetic and 0 represents non-diabetic.

We split the dataset to train data and test data . The first subset is known as the training data - it's a portion of our actual dataset that is fed into the machine learning model to discover and learn patterns. In this way, it trains our model. Once your machine learning model is built (with your training data), you need unseen data to test your model. This data is called testing data, and you can use it to evaluate the performance and progress of your algorithms' training and adjust or optimize it for improved results. Once your machine learning model is built (with your training data), you need unseen data to test your model. This data is called testing data, and you can use it to evaluate the performance and progress of your algorithms' training and adjust or optimize it for improved result

7.2 Classifiers Used

7.21 SVC

The objective of a Linear SVC (**Support Vector Classifier**) is to fit to the data you provide, returning a "best fit" hyperplane that divides, or categorizes, your data. From there, after getting the hyperplane, you can then feed some features to your classifier to see what the "predicted" class is.

7.22 DecesionTreeClassifier

It creates the classification model by building a decision tree. Each node in the tree specifies a test on an attribute, each branch descending from that node corresponds to one of the possible values for that attribute.

7.23 RandomForestClassifier

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

7.24 GradientBoostingClassifier

Gradient boosting classifiers are a group of machine learning algorithms that combine many weak learning models together to create a strong predictive model. Decision trees are usually used when doing gradient boosting

7.25 KNeighborsClassifier

KNeighborsClassifier looks for the 5 nearest neighbors. We must explicitly tell the classifier to use Euclidean distance for determining the proximity between neighboring points. Using our newly trained model, we predict whether a tumor is benign or not given its mean compactness and area.

7.26 Logistic regression

It is a classification technique borrowed by machine learning from the field of statistics.

The classifier with the highest accuracy is used for diabetes prediction.

7.3 Code and Output

Csv file

	A	B	C	D	E	F	G	H	I
1	Pregnanci	Glucose	BloodPres	SkinThickn	Insulin	BMI	DiabetesP	Age	Outcome
2	6	148	72	35	0	33.6	0.627	50	1
3	1	85	66	29	0	26.6	0.351	31	0
4	8	183	64	0	0	23.3	0.672	32	1
5	1	89	66	23	94	28.1	0.167	21	0
6	0	137	40	35	168	43.1	2.288	33	1
7	5	116	74	0	0	25.6	0.201	30	0
8	3	78	50	32	88	31	0.248	26	1
9	10	115	0	0	0	35.3	0.134	29	0
10	2	197	70	45	543	30.5	0.158	53	1
11	8	125	96	0	0	0	0.232	54	1
12	4	110	92	0	0	37.6	0.191	30	0
13	10	168	74	0	0	38	0.537	34	1
14	10	139	80	0	0	27.1	1.441	57	0
15	1	189	60	23	846	30.1	0.398	59	1
16	5	166	72	19	175	25.8	0.587	51	1
17	7	100	0	0	0	30	0.484	32	1
18	0	118	84	47	230	45.8	0.551	31	1
19	7	107	74	0	0	29.6	0.254	31	1
20	1	103	30	38	83	43.3	0.183	33	0
21	1	115	70	30	96	34.6	0.529	32	1
22	3	126	88	41	235	39.3	0.704	27	0
23	8	99	84	0	0	35.4	0.388	50	0
24	7	196	90	0	0	39.8	0.451	41	1
25	9	119	80	35	0	29	0.263	29	1
26	11	143	94	33	146	36.6	0.254	51	1
27	10	125	70	26	115	31.1	0.205	41	1

Code

```
In [1]: import pandas as pd
```

```
In [2]: data = pd.read_csv('Downloads/diabetesdataset.csv')
```

```
In [3]: data.head()
```

Out[3]:

	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

```
In [4]: data.tail()
```

Out[4]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

```
In [5]: data.shape

Out[5]: (768, 9)
```

```
In [6]: print("Number of Rows",data.shape[0])
        print("Number of Columns",data.shape[1])

Number of Rows 768
Number of Columns 9
```

```
In [7]: data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

```
In [8]: data.isnull().sum()
```

```
Out[8]: Pregnancies                 0
        Glucose                     0
        BloodPressure               0
        SkinThickness               0
        Insulin                     0
        BMI                         0
        DiabetesPedigreeFunction    0
        Age                         0
        Outcome                     0
        dtype: int64
```

```
In [9]: data.describe()
```

Out[9]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

```
In [10]: import numpy as np
         data_copy = data.copy(deep=True)
         data.columns
```

```
Out[10]: Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
                'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
               dtype='object')
```

```
In [11]: data_copy[['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
                'BMI']] = data_copy[['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
                'BMI']].replace(0,np.nan)
```

```
In [12]: data_copy.isnull().sum()
```

```
Out[12]: Pregnancies                   0
         Glucose                       5
         BloodPressure                35
         SkinThickness               227
         Insulin                     374
         BMI                          11
         DiabetesPedigreeFunction      0
         Age                           0
         Outcome                       0
         dtype: int64
```

```
In [13]: data['Glucose'] = data['Glucose'].replace(0,data['Glucose'].mean())
         data['BloodPressure'] = data['BloodPressure'].replace(0,data['BloodPressure'].mean())
         data['SkinThickness'] = data['SkinThickness'].replace(0,data['SkinThickness'].mean())
         data['Insulin'] = data['Insulin'].replace(0,data['Insulin'].mean())
         data['BMI'] = data['BMI'].replace(0,data['BMI'].mean())
```

```
In [14]: X = data.drop('Outcome',axis=1)
         y = data['Outcome']
```

```
In [15]: from sklearn.model_selection import train_test_split
         X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.20,
                                                        random_state=42)
```

```
In [16]: from sklearn.preprocessing import StandardScaler
         from sklearn.linear_model import LogisticRegression
         from sklearn.neighbors import KNeighborsClassifier
         from sklearn.svm import SVC

         from sklearn.tree import DecisionTreeClassifier
         from sklearn.ensemble import RandomForestClassifier
         from sklearn.ensemble import GradientBoostingClassifier

         from sklearn.pipeline import Pipeline
```

In [17]:
```
pipeline_lr  = Pipeline([('scalar1',StandardScaler()),
                         ('lr_classifier',LogisticRegression())])

pipeline_knn = Pipeline([('scalar2',StandardScaler()),
                          ('knn_classifier',KNeighborsClassifier())])

pipeline_svc = Pipeline([('scalar3',StandardScaler()),
                         ('svc_classifier',SVC())])

pipeline_dt = Pipeline([('dt_classifier',DecisionTreeClassifier())])
pipeline_rf = Pipeline([('rf_classifier',RandomForestClassifier(max_depth=3))])
pipeline_gbc = Pipeline([('gbc_classifier',GradientBoostingClassifier())])
pipelines = [pipeline_lr,
            pipeline_knn,
            pipeline_svc,
            pipeline_dt,
            pipeline_rf,
            pipeline_gbc]
pipelines
```

Out[17]:
```
[Pipeline(steps=[('scalar1', StandardScaler()),
                 ('lr_classifier', LogisticRegression())]),
 Pipeline(steps=[('scalar2', StandardScaler()),
                 ('knn_classifier', KNeighborsClassifier())]),
 Pipeline(steps=[('scalar3', StandardScaler()), ('svc_classifier', SVC())]),
 Pipeline(steps=[('dt_classifier', DecisionTreeClassifier())]),
 Pipeline(steps=[('rf_classifier', RandomForestClassifier(max_depth=3))]),
 Pipeline(steps=[('gbc_classifier', GradientBoostingClassifier())])]
```

In [18]:
```
for pipe in pipelines:
    pipe.fit(X_train,y_train)
pipe_dict = {0:'LR',
             1:'KNN',
             2:'SVC',
             3:'DT',
             4: 'RF',
             5: 'GBC'}
pipe_dict
```

Out[18]:
```
{0: 'LR', 1: 'KNN', 2: 'SVC', 3: 'DT', 4: 'RF', 5: 'GBC'}
```

```
In [19]: for i,model in enumerate(pipelines):
             print("{} Test Accuracy:{}".format(pipe_dict[i],model.score(X_test,y_test)*100))

LR Test Accuracy:76.62337662337663
KNN Test Accuracy:76.62337662337663
SVC Test Accuracy:73.37662337662337
DT Test Accuracy:72.07792207792207
RF Test Accuracy:77.92207792207793
GBC Test Accuracy:75.32467532467533
```

```
In [20]: from sklearn.ensemble import RandomForestClassifier
```

```
In [21]: X = data.drop('Outcome',axis=1)
         y = data['Outcome']
```

```
In [22]: rf =RandomForestClassifier(max_depth=3)
```

```
In [23]: rf.fit(X,y)

Out[23]: RandomForestClassifier(max_depth=3)
```

```
In [24]: new_data = pd.DataFrame({
             'Pregnancies':6,
             'Glucose':148.0,
             'BloodPressure':72.0,
             'SkinThickness':35.0,
             'Insulin':79.799479,
             'BMI':33.6,
             'DiabetesPedigreeFunction':0.627,
             'Age':50,

         },index=[0])
```

```
In [25]: p = rf.predict(new_data)
```

```
In [26]: if p[0] == 0:
             print('non-diabetic')
         else:
             print('diabetic')

diabetic
```

```
In [27]: import joblib
```

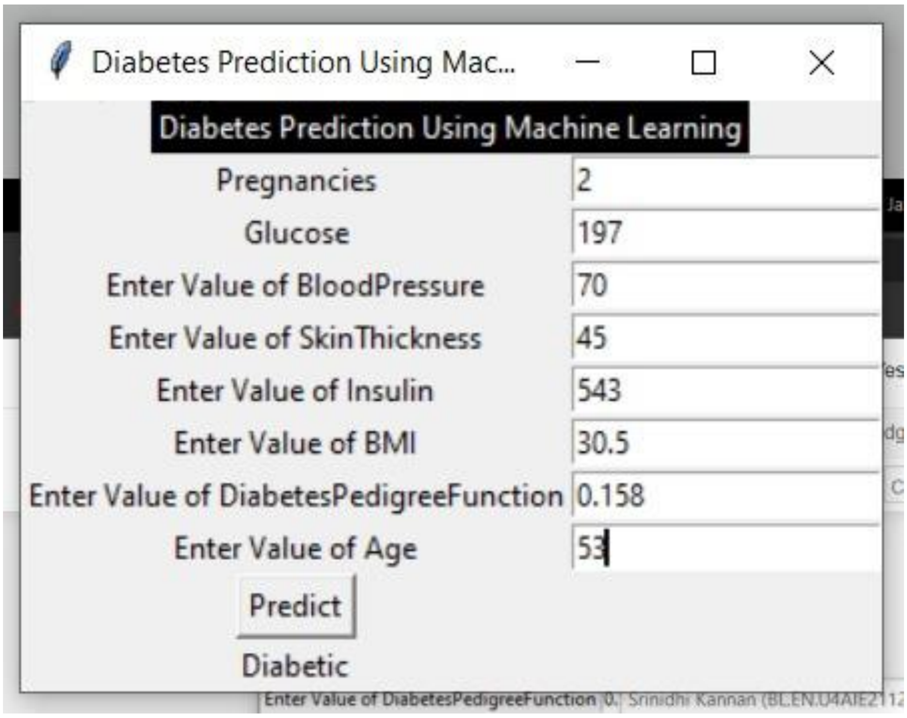
```
In [28]: joblib.dump(rf,'model_joblib_diabetes')

Out[28]: ['model_joblib_diabetes']
```

```
In [29]: model = joblib.load('model_joblib_diabetes')
```

```
In [30]: model.predict(new_data)

Out[30]: array([1], dtype=int64)
```

In [31]:
```
from tkinter import *
import joblib
import numpy as np
from sklearn import *
def show_entry_fields():
    p1=float(e1.get())
    p2=float(e2.get())
    p3=float(e3.get())
    p4=float(e4.get())
    p5=float(e5.get())
    p6=float(e6.get())
    p7=float(e7.get())
    p8=float(e8.get())

    model = joblib.load('model_joblib_diabetes')
    result=model.predict([[p1,p2,p3,p4,p5,p6,p7,p8]])

    if result == 0:
        Label(master, text="Non-Diabetic").grid(row=31)
    else:
        Label(master, text="Diabetic").grid(row=31)


master = Tk()
master.title("Diabetes Prediction Using Machine Learning")


label = Label(master, text = "Diabetes Prediction Using Machine Learning"
                          , bg = "black", fg = "white"). \
                               grid(row=0,columnspan=2)


Label(master, text="Pregnancies").grid(row=1)
Label(master, text="Glucose").grid(row=2)
Label(master, text="Enter Value of BloodPressure").grid(row=3)
Label(master, text="Enter Value of SkinThickness").grid(row=4)
Label(master, text="Enter Value of Insulin").grid(row=5)
Label(master, text="Enter Value of BMI").grid(row=6)
Label(master, text="Enter Value of DiabetesPedigreeFunction").grid(row=7)
Label(master, text="Enter Value of Age").grid(row=8)
```

```
e1 = Entry(master)
e2 = Entry(master)
e3 = Entry(master)
e4 = Entry(master)
e5 = Entry(master)
e6 = Entry(master)
e7 = Entry(master)
e8 = Entry(master)


e1.grid(row=1, column=1)
e2.grid(row=2, column=1)
e3.grid(row=3, column=1)
e4.grid(row=4, column=1)
e5.grid(row=5, column=1)
e6.grid(row=6, column=1)
e7.grid(row=7, column=1)
e8.grid(row=8, column=1)


Button(master, text='Predict', command=show_entry_fields).grid()

mainloop()
```



7.4 Conclusion

RandomForestClassifier gives us the highest accuracy of 77.92207792207793
And hence we use this to predict diabetes.We have also used tinkinter, a GUI as the frontend to display the result.

CHAPTER 8

CONCLUSION

8.1 Complications

- Cardiovascular diseases
- Nerve Damage(nephropathy)
- Kidney damage
- Eye damage (retinopathy)
- Foot damage.
- Skin conditions.
- Hearing impairment.
- Alzheimer's disease
- Deppression

8.2 Symptoms

- Urinating often
- Feeling very thirsty
- Feeling very hungry—even though you are eating
- Extreme fatigue
- Blurry vision
- Cuts/bruises that are slow to heal
- Weight loss—even though you are eating more (type 1)
- Tingling, pain, or numbness in the hands/feet (type 2)

8.3 Prevention

- Reduce your total carb intake.
- Exercise regularly.
- Drink water as your primary beverage
- Try to lose excess weightQuit smoking. ...
- Reduce your portion sizes

- Cut back on sedentary behaviors
- Follow a high fiber diet.

Chapter 9

FUTURE SCOPE

In future, the designed system with the used machine learning classification algorithms can be used to predict or diagnose other diseases. We can plot a graph from the information and analyze the pattern for better comparison .The work can also be extended and improved for the automation of diabetes analysis including some other machine learning algorithms.

REFERENCES

1. Performance Analysis of Support Vector Machine in Diabetes Prediction Narendra Mohan Assistant Professor, Department of Computer Engineering Applications GLA University, Mathura, India narendra.mohan@gla.ac.in (ORCHID-0000-0002-7037-3318) Vinod Jain Assistant Professor, Department of Computer Engineering Applications GLA University, Mathura, India Vinod.jain@gla.ac.in (ORCHID- 0000-0003-0260-7319)

2. Analysis of Machine Learning Algorithms and Obtaining Highest Accuracy for Prediction of Diabetes in Women Arushi Agarwal Amity University Uttar Pradesh Noida, India arushiagarwal14@gmail.com Ankur Saxena Amity University Uttar Pradesh Noida, India asaxena1@amity.edu

3. https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database