University of New Haven
TAGLIATELA COLLEGE OF ENGINEERING
Electrical & Computer Engineering and Computer Science

Distributed and Scalable Data Engineering Final project

Group4-Phoenix Attackers

TECHNICAL REPORT

CONTENTS

Project Title.....2
Executive Summary.....3
Highlights of Project.....4
Submitted on: 4
Abstract 5
Methodology 6
Problems or Challenges.....7
Tools Used.....8
Exploratory Data Analysis.....9
Analysis of Relationship Between Variables.....13
Data Deployment and Data Modeling.....16
Model Evaluation.....17
Results Section.....17
Building a Flask Web App.....18
Conclusion.....21
Contributions/References.....21

University of New Haven

The Early Prediction of Diabetes

- ARDIANA SULA
  PROFESSOR, ACADEMIC-ADVISOR

GROUP4-PHOENIX ATTACKERS

NAME	E-MAIL ID
RAHUL MUVVALA	rmuvv1@unh.newhaven.edu
KALPANA CHAMALA	kcham5@unh.newhaven.edu
RISHITHA VENKANNAGARI	rvenk3@unh.newhaven.edu
MASTANVALI SHAIK	mshai32@unh.newhaven.edu
FARHAN SHAIK	fshai2@unh.newhaven.edu

Executive Summary

Diabetes is one of the deadliest diseases in the world which is rapidly increasing and the cause of which vary depending on the genetic makeup family, history, ethnicity, health etc. It is an illness caused because of high glucose level in the body. Diabetes, untreated, may cause some major issues in like heart related problems, kidney problem, blood pressure, eye damage, it can also affect other organs of the body. Good thing is diabetes can be controlled if it is predicted earlier. So, the aim of this project is to design and implement diabetes prediction using machine learning methods and performance analysis of those methods. This helps a lot in saving the valuable time and money and is also a contribution for developing the technology in medical history.

Title of Project

The Early Prediction of Diabetes

Highlights of Project:

- **In this project, the objective is to predict whether the person has Diabetes or not based on various features such as**
  - ❑ Pregnancies
  - ❑ Age
  - ❑ Insulin Level
  - ❑ BMI
- **We have also predicted the best model among multiple Machine Learning Algorithms, to predict the Diabetes.**

Submitted on:

12 Dec 2022.

Abstract

The aim of our project is to design and implement diabetes prediction using machine learning methods and performance analysis of those methods. The techniques we are using are support vector machine-nearest Neighbor, decision Tree, logistic regression, random forest. Applying machine learning techniques to analyze the performance of these methods will help to find accuracy of the model. It will help to figure out the responsible/important feature which play a major role in prediction. Our result shows that random forest achieved higher accuracy compared to other machine learning techniques. Diabetes can be controlled if it is predicted earlier. Machine learning techniques will provide better result for prediction by constructing models from datasets collected from patients.

The main goal of the project is to decrease the diabetic related issues faced by the public. As diabetics are very common these days that most of the people are suffering from, the early prediction can be very helpful.

GitHub Link: https://github.com/You-Can-Do/DSCI-6007-02-Phonix-Attackers

Methodology

We imported the data sets into Jupyter notebook and performed the analysis. In this project, the objective is to predict whether the person has Diabetes or not based on various features such as:

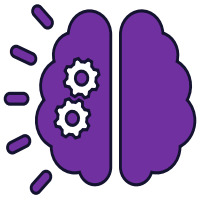
1. Pregnancies
2. Insulin Level
3. Age
4. BMI

In this project we have used CRISP-DM methodology.

Business Understanding

When it comes to health issues, there are many deadly diseases which effects a lot of people due to their inappropriate health habits and maintenance. Many people across the world are suffering a lot even some are losing their lives which is sad to know. There are many reasons for this like the feeling of getting their problems exposed, no proper guidance and knowledge, not enough money to consult a doctor and many more. So, in this project we developed a medical based application which helps at least a few in their medical life.

The Problem challenges



Problem

Cause of diabetes vary depending on the genetic makeup family, history, ethnicity, health etc.,

01

Problem

Diabetes is one of the deadliest diseases in the world which is rapidly increasing.

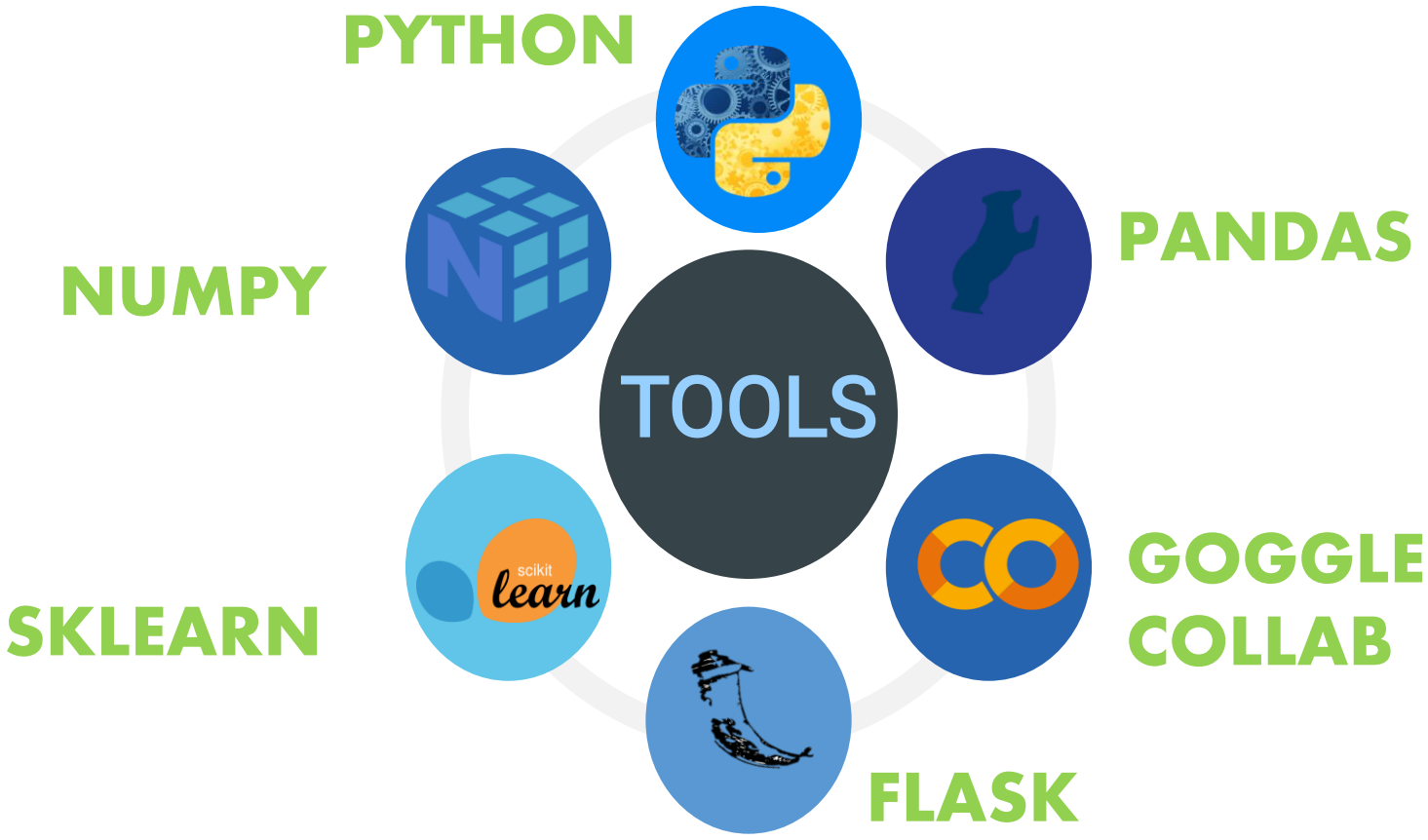
02

Problem

It is an illness caused because of high glucose; level in a human body.

03

TOOLS USED



IMPORTING LIBRARIES

Slide Type: Slide

Importing the Necessary Libraries

In [22]:

Slide Type: Sub-Slide

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier,GradientBoostingClassifier,AdaBoostClassifier,BaggingClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from xgboost import XGBClassifier
from collections.abc import Sequence
import warnings
warnings.filterwarnings('ignore')
```

EXPLORATORY DATA ANALYSIS

No	Steps
1	UnderStanding the Data
2	Clean The Data
3	Analysis of Relationship between variables

1. UNDERSTANDING THE DATA

The data set that has used in this project has taken from the Kaggle. "This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether a patient has diabetes or not, based on certain diagnostic measurements included in the dataset. Several constraints were placed on the selection of these instances from a larger database. All patients here are females at least 21 years old of Pima Indian heritage." and used a simple random forest classifier.

Slide Type Sub-Slide

```
# loading the diabetes dataset to a pandas DataFrame
diabetes_dataset = pd.read_csv('C:/Users/rahul/Downloads/DSCI 6007-02/Group4_Final_Project_Phoenix Attackers/diabetes.csv
```

Slide Type Sub-Slide

```
# printing the first 5 rows of the dataset
diabetes_dataset.head()
```

4]:

	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

Slide Type Sub-Slide

```
# number of rows and Columns in this dataset
diabetes_dataset.shape
```

```
(768, 9)
```

2. CLEANING THE DATA

Slide Type Sub-Slide

```
1 diabetes_dataset.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 [7]:

Slide Type Sub-Slide

```
1 diabetes_dataset.isnull().sum(axis=0)
```

Out[7]:

```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

Slide Type Sub-Slide

There is no empty data, so nothing is there to clean

In [8]:

Slide Type Sub-Slide

```
1 # getting the statistical measures of the data
2 diabetes_dataset.describe()
```

Out[8]:

	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 [9]:

Slide Type Sub-Slide

```
1 diabetes_dataset.groupby('Outcome').mean()
```

Out[9]:

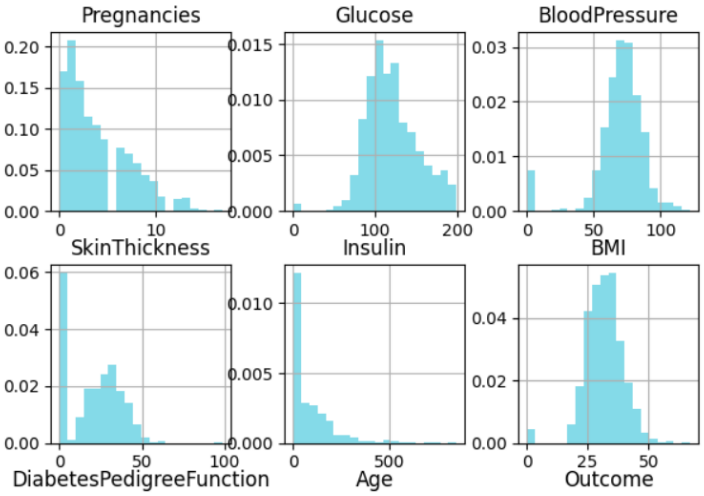
Outcome	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
0	3.298000	109.980000	68.184000	19.664000	68.792000	30.304200	0.429734	31.190000
1	4.865672	141.257463	70.824627	22.164179	100.335821	35.142537	0.550500	37.067164

3. DATA VISUALIZATION

In [10]:

Slide Type Sub-Slide

```
1 Data_isualizations = diabetes_dataset.hist(histtype="bar", bins=20, alpha=0.5, color="#0BB7D3", density=1,figsize = (7,7
```



4. ANALYSIS OF RELATIONALSHIP BETWEEN VARIABLES

In [11]:

Slide Type Sub-Slide

```
1 diabetes_dataset.corr()
```

Out[11]:

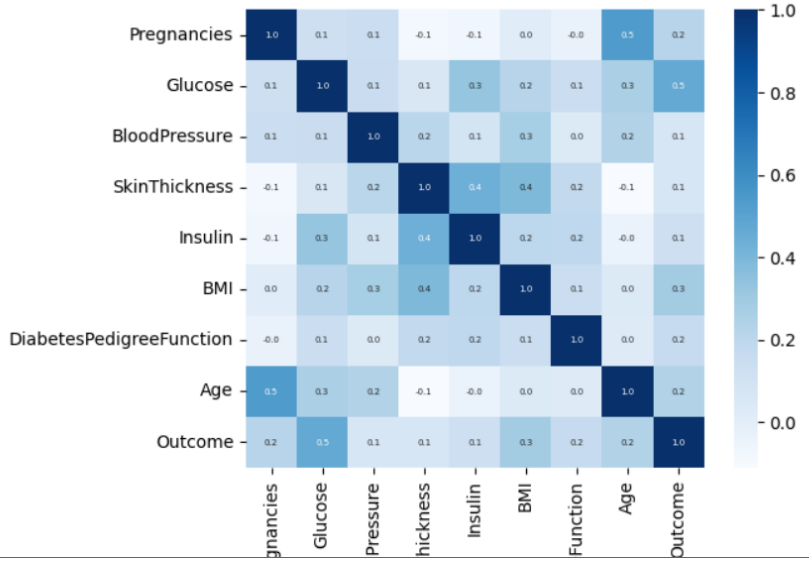
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
Pregnancies	1.000000	0.129459	0.141282	-0.081672	-0.073535	0.017683	-0.033523	0.544341	0.221898
Glucose	0.129459	1.000000	0.152590	0.057328	0.331357	0.221071	0.137337	0.263514	0.466581
BloodPressure	0.141282	0.152590	1.000000	0.207371	0.088933	0.281805	0.041265	0.239528	0.065068
SkinThickness	-0.081672	0.057328	0.207371	1.000000	0.436783	0.392573	0.183928	-0.113970	0.074752
Insulin	-0.073535	0.331357	0.088933	0.436783	1.000000	0.197859	0.185071	-0.042163	0.130548
BMI	0.017683	0.221071	0.281805	0.392573	0.197859	1.000000	0.140647	0.036242	0.292695
DiabetesPedigreeFunction	-0.033523	0.137337	0.041265	0.183928	0.185071	0.140647	1.000000	0.033561	0.173844
Age	0.544341	0.263514	0.239528	-0.113970	-0.042163	0.036242	0.033561	1.000000	0.238356
Outcome	0.221898	0.466581	0.065068	0.074752	0.130548	0.292695	0.173844	0.238356	1.000000

In [12]:

Slide Type Sub-Slide

```
1 correlation = diabetes_dataset.corr().round(2)
2 sns.heatmap(correlation, cbar=True, square=True, fmt='.1f', annot=True, annot_kws={'size':5}, cmap='Blues')
3 plt.show()
```

Out[12]: <AxesSubplot: >

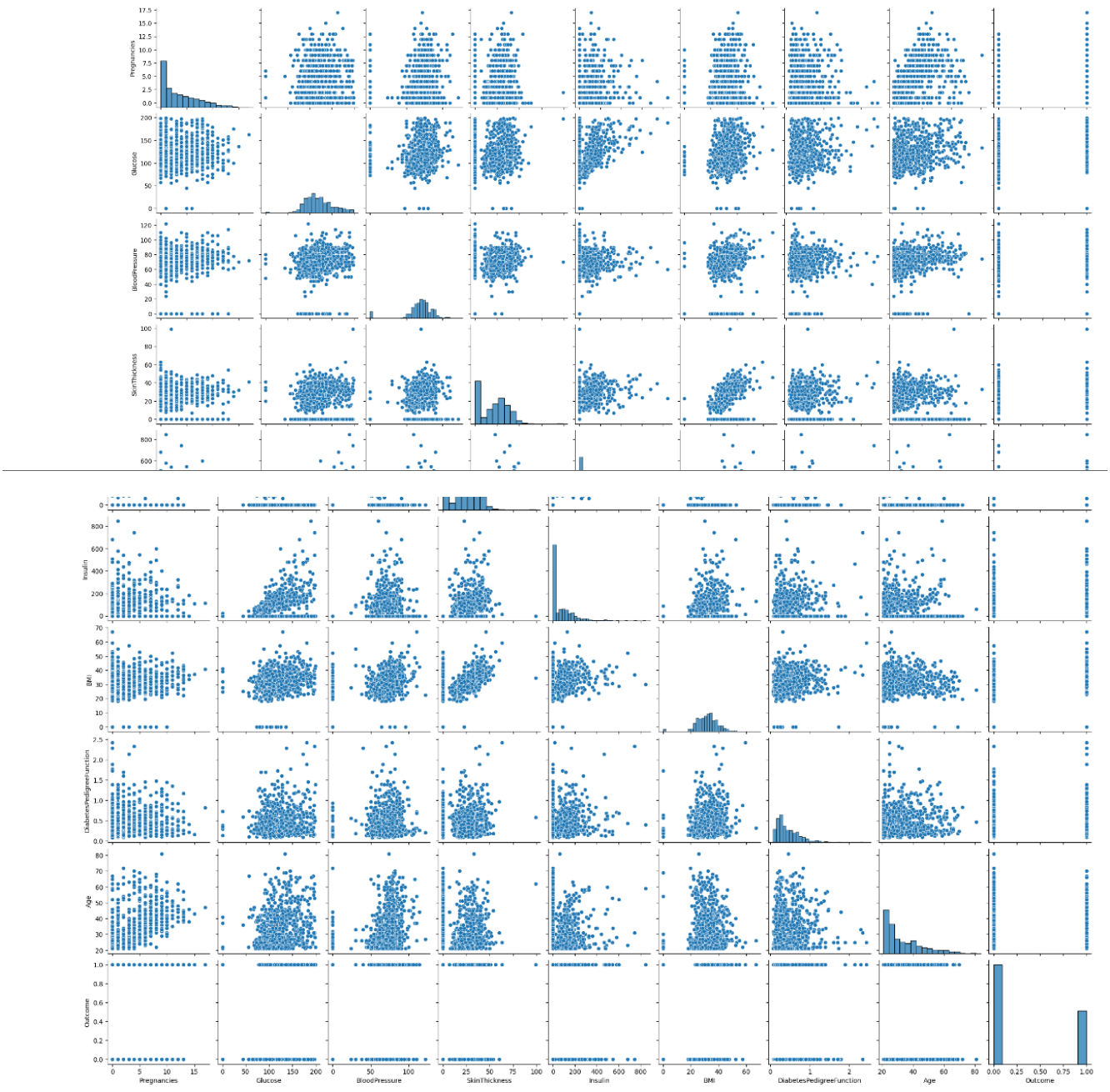


In [13]:

Slide Type Sub-Slide

```
1 sns.pairplot(diabetes_dataset)
```

Out[13]: <seaborn.axisgrid.PairGrid at 0x17e92d4ac50>

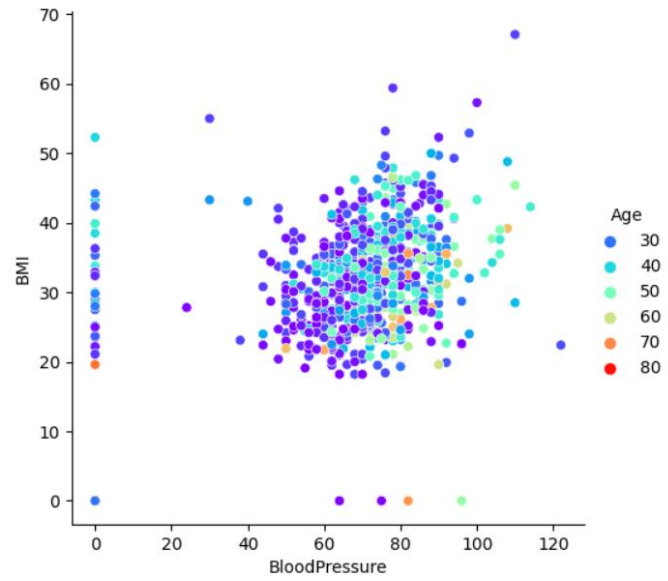


In [31]:

Slide Type Sub-Slide

```
1 sns.relplot (y='BMI',x='BloodPressure',hue='Age',data=diabetes_dataset,palette = 'rainbow')
```

Out[31]: <seaborn.axisgrid.FacetGrid at 0x17ea020faf0>

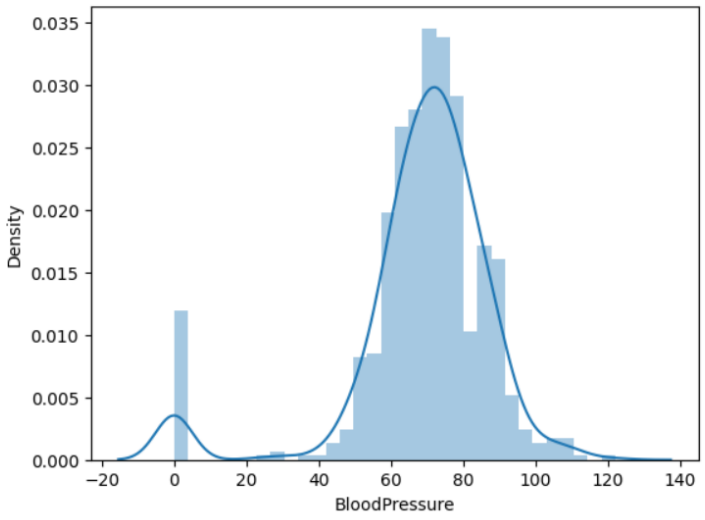


In [23]:

Slide Type Sub-Slide

```
1 sns.distplot(diabetes_dataset['BloodPressure'] )
```

Out[23]: <AxesSubplot: xlabel='BloodPressure', ylabel='Density'>



5. DATA DEPLOYMENT

In [16]:

Slide Type Sub-Slide

```
# Label Encoding

for column in diabetes_dataset.columns:
    if diabetes_dataset[column].dtype == type(object):
        le = sklearn.preprocessing.LabelEncoder()
        diabetes_dataset[column] = le.fit_transform(diabetes_dataset[column])
```

In [17]:

Slide Type Sub-Slide

```
# Splting data into Train and test data

X = diabetes_dataset.drop(columns = 'Outcome', axis=1)
Y = diabetes_dataset['Outcome']
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.2, stratify=Y, random_state=2)
```

6. DATA MODELING

In [25]:

Slide Type Sub-Slide

```
# RandomForestClassifier
rndf = RandomForestClassifier()
rndf.fit(X_train, Y_train)
# LogisticRegression
log=LogisticRegression()
log = LogisticRegression(solver='lbfgs', max_iter=1000)
log.fit(X_train,Y_train)
# Support Vector Machine
s=SVC()
s.fit(X_train,Y_train)
# XgBoost
xg=XGBClassifier()
xg.fit(X_train,Y_train)
# k-Nearest-Neighbor
k=KNeighborsClassifier()
k.fit(X_train,Y_train)
```

Out[25]: RandomForestClassifier()

Out[25]: LogisticRegression(max_iter=1000)

Out[25]: SVC()

7. MODEL EVALUATION

In [19]:

Slide Type Sub-Slide

```
# RandomForestClassifier
pred=rndf.predict(X_test)
R = round(accuracy_score(Y_test, pred),2)
# LogisticRegression
pred1=log.predict(X_test)
R1 = round(accuracy_score(Y_test, pred1),2)
# Support Vector Machine
pred2=s.predict(X_test)
R2 = round(accuracy_score(Y_test, pred2),2)
# XgBoost
pred3=xg.predict(X_test)
R3 = round(accuracy_score(Y_test, pred3),2)
# k-Nearest-Neighbor
pred4=k.predict(X_test)
R4 = round(accuracy_score(Y_test, pred4),2)
```

In [20]:

Slide Type Sub-Slide

```
scores=[]
scores.append(R)
scores.append(R1)
scores.append(R2)
scores.append(R3)
scores.append(R4)
Accuracy=pd.DataFrame(scores)

```

Results Section

In [21]:

Slide Type Sub-Slide

```
Accuracy.index= ["RandomForest","LogisticRegression ","SVM","XgBoost","KNN"]
Accuracy.columns=["Accuracy"]
Accuracy.index.name='Methods'
Accuracy
```

Out[21]:

Methods	Accuracy
RandomForest	0.76
LogisticRegression	0.76
SVM	0.79
XgBoost	0.75
KNN	0.73

Building Flask Web App

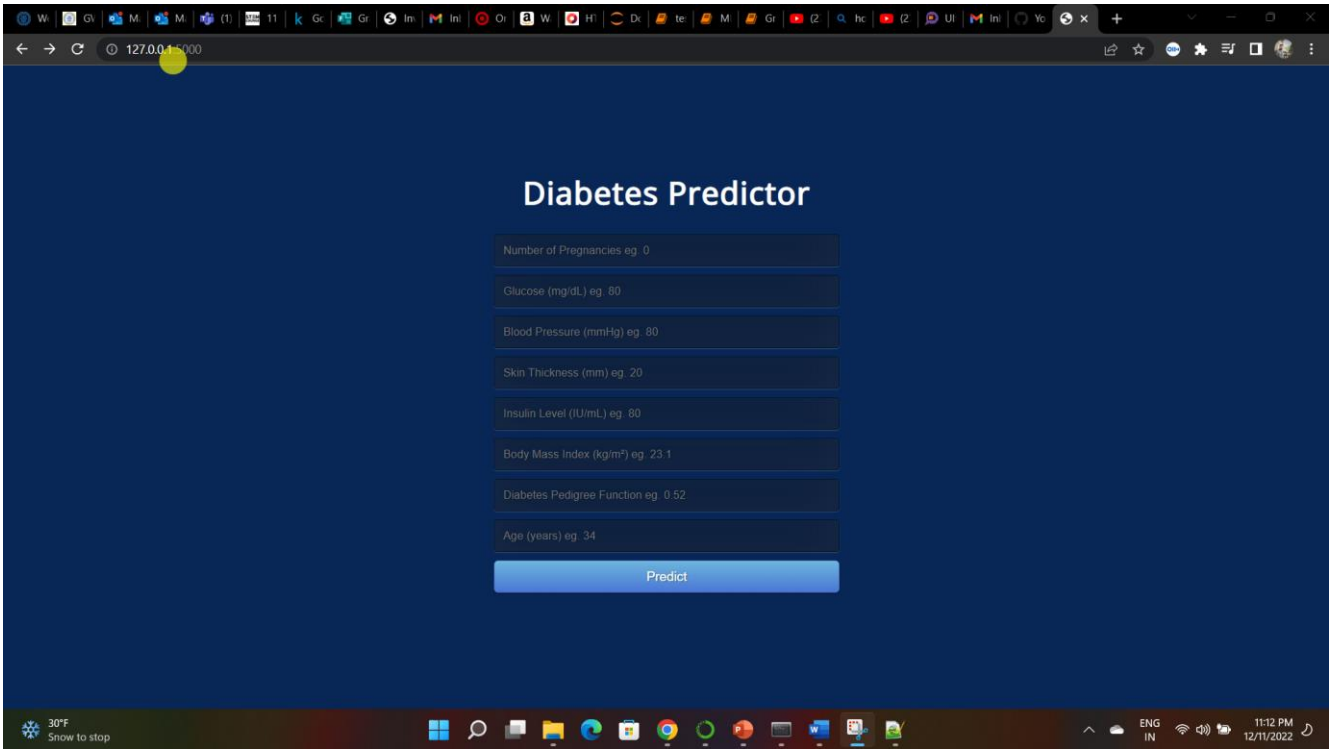
For this we developed

1. App.py file
2. Pickle file
3. Static files
4. Templates
    a. Index.html
    b. Result.html

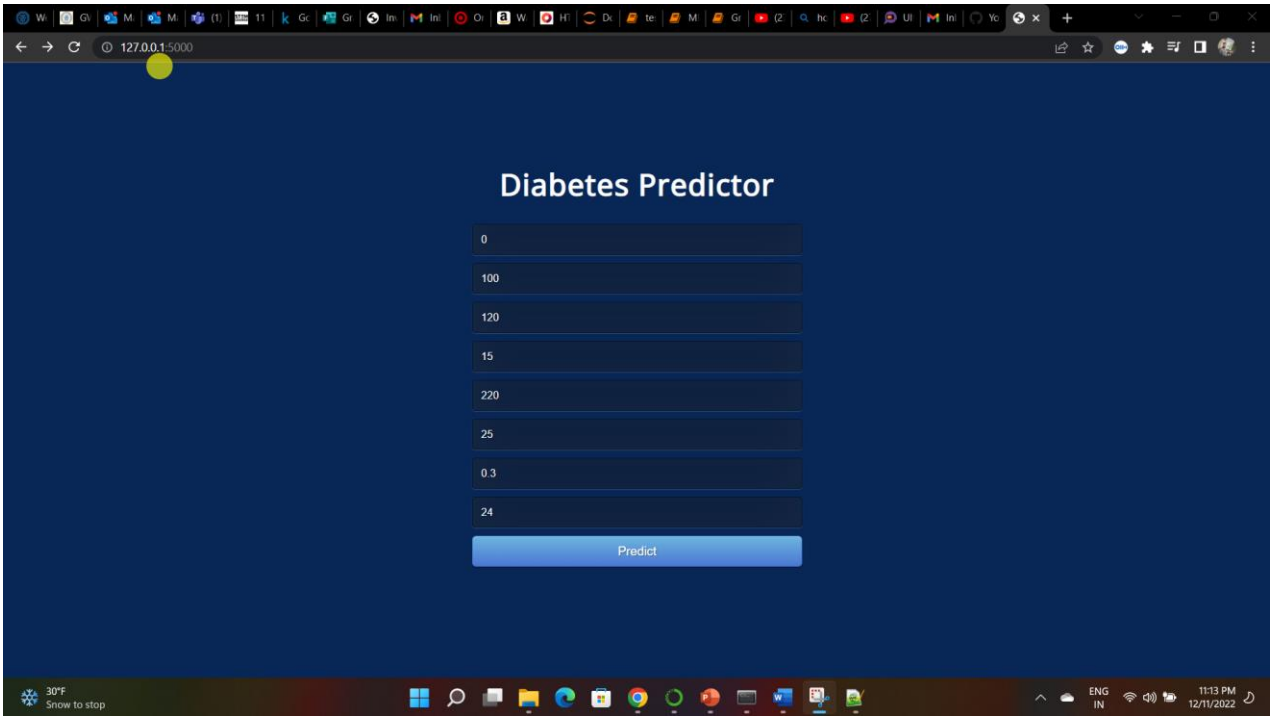
Anaconda Prompt:

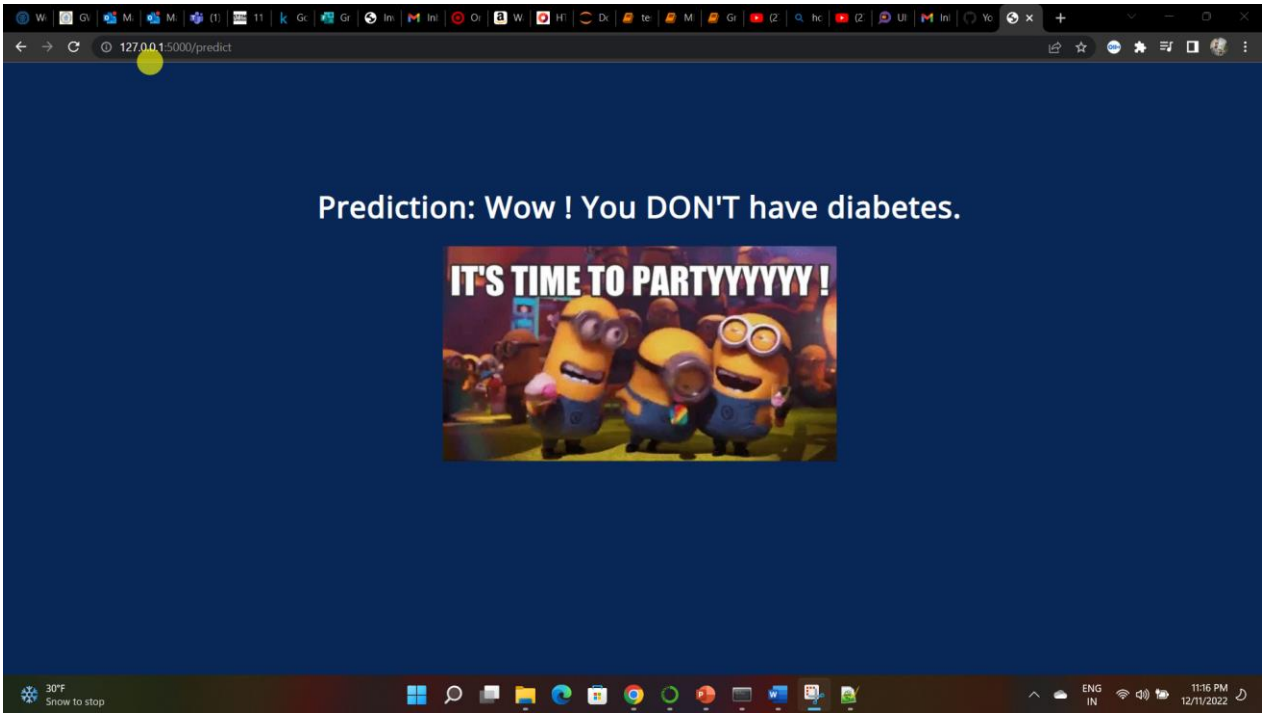
```
(rahul) C:\Users\rahul\Downloads\DSCI 6007-02\Group4_Final_Project_Phoenix Attackers>python app.py
 * Serving Flask app 'app' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 733-395-232
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [11/Dec/2022 23:08:32] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [11/Dec/2022 23:08:32] "GET /static/css/style.css HTTP/1.1" 404 -
```

HOME PAGE:

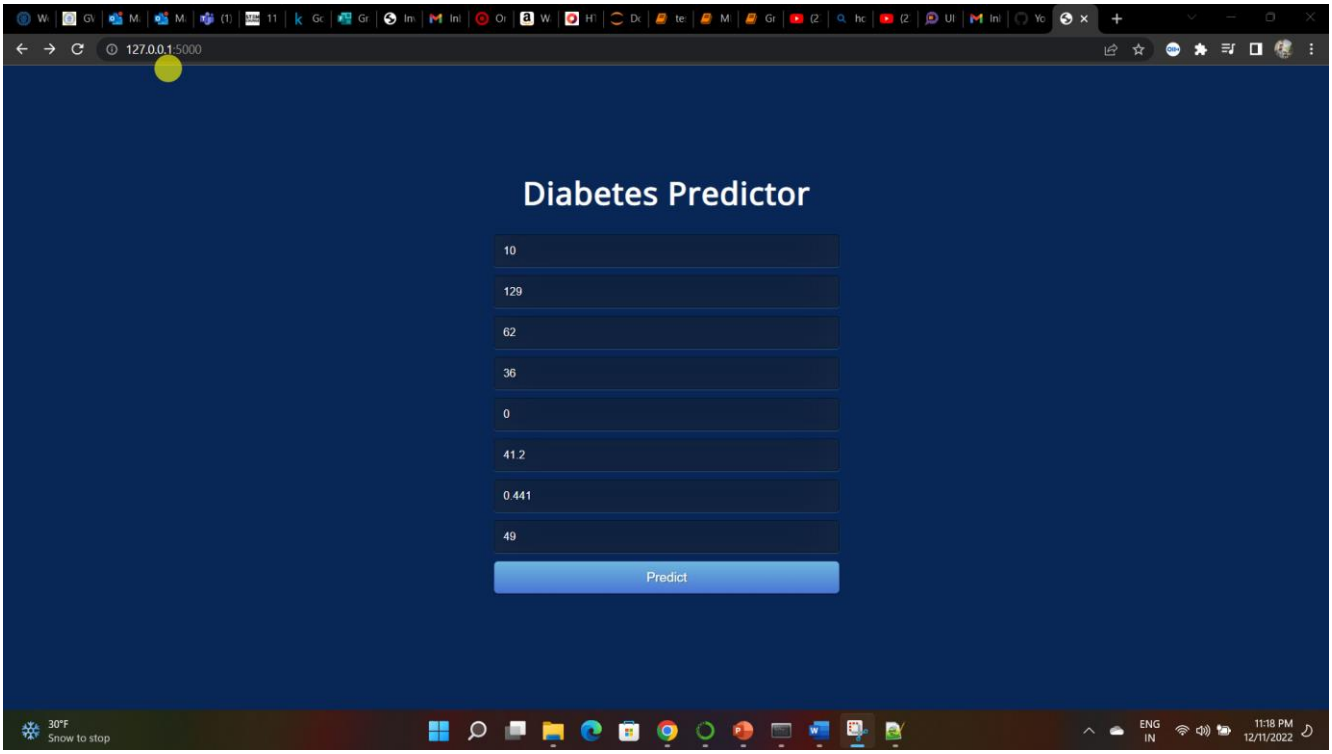


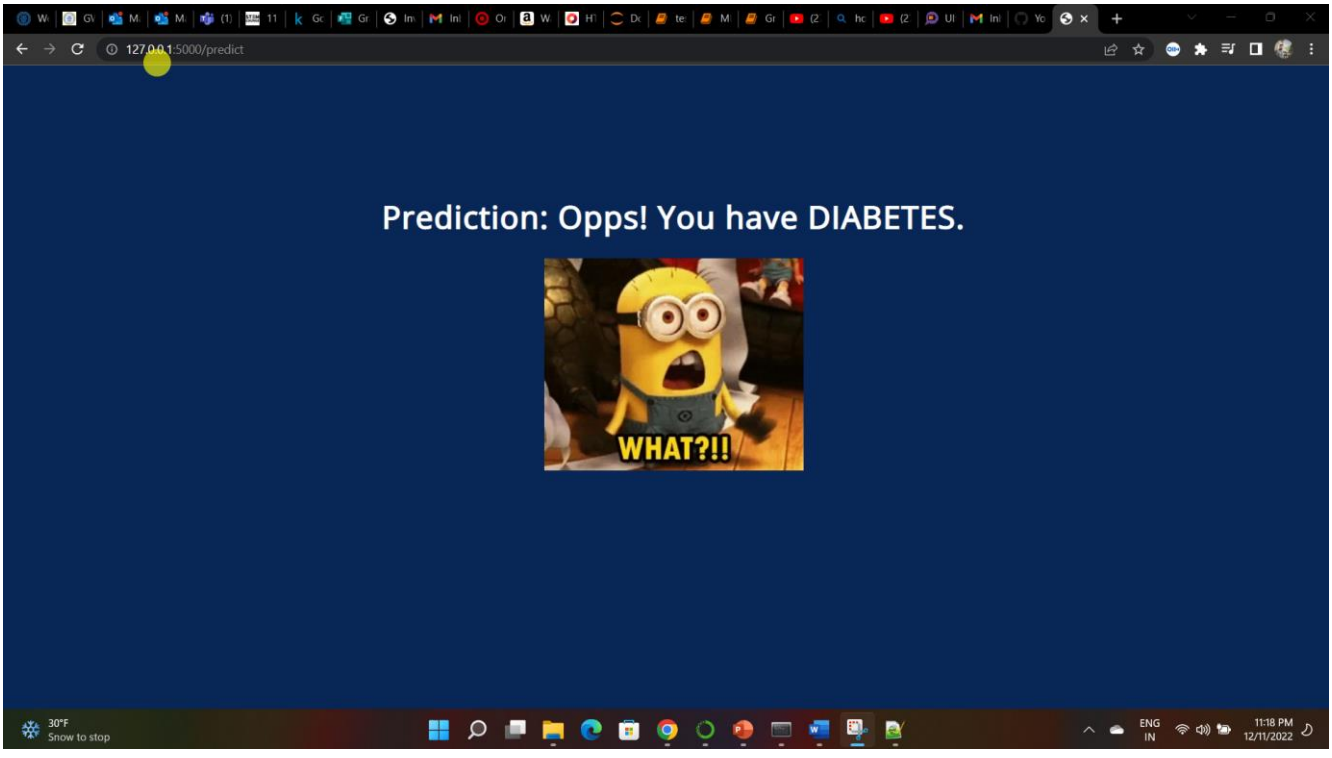
Prediction 1:





Prediction 2:





Conclusion

Based on the above analysis, we can conclude that "Support Vector Machine (SVM)" is the best model to predict the diabetes with most accurate value compared to other machine learning algorithms.

Contributions/References:

1. Data sets We have taken from the Kaggle. **This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases.**
2. Some Information is taken from Internet.