

**PREDICTION OF DIABETES DISEASE USING DATA MINING WITH IT'S
APPLICATION**

BY

Faisal Ahmed

161412316

Md. Manwar

161412323

A Project Report submitted in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science in Computer Science and Engineering

Supervised By

Md. Ataullah Bhuiyan

Senior Lecturer

Department of Computer Science and Engineering (CSE)

City University, Dhaka, Bangladesh



CITY UNIVERSITY

DHAKA, BANGLADESH

AUGUST 2020

Certificate

This is to certify that the work presented in this project entitled “**Prediction of Diabetes Disease using Data mining with it’s Application**” is the outcome of the work done by Faisal Ahmed and Md Manwar under the supervision of Md. Ataullah Bhuiyan, Senior Lecturer, Department of Computer Science and Engineering, City University during December 2019 to August 2020. It is also declared that neither this project/report nor any part it has been submitted or is being currently submitted anywhere else for the award of any degree or diploma.

Approved By

Md. Ataullah Bhuiyan

Senior Lecturer

Department of Computer Science and Engineering (CSE)

City University, Dhaka, Bangladesh

DECLARATION

We hereby declare that, this project has been done by us under the supervision of **Md. Ataullah Bhuiyan**, Senior Lecturer, **Department of CSE**, City University. We also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma.

Submitted by:

Faisal Ahmed

ID: 161412316

Department of CSE

City University

DECLARATION

We hereby declare that, this project has been done by us under the supervision of **Md. Ataullah Bhuiyan**, Senior Lecturer, **Department of CSE**, City University. We also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma.

Submitted by:

Md. Manwar

ID: 161412323

Department of CSE

City University

ACKNOWLEDGEMENT

First we express our heartiest thanks and gratefulness to almighty Allah for His divine blessing makes us possible to complete this project successfully.

We fell grateful to and wish our profound our indebtedness to **Md. Ataullah Bhuiyan**, Senior Lecturer, Department of CSE, City University, Dhaka. Deep Knowledge & keen interest of our supervisor in the field of machine learning influenced us to carry out this project .His endless patience ,scholarly guidance ,continual encouragement , constant and energetic supervision, constructive criticism , valuable advice ,reading many inferior draft and correcting them at all stage have made it possible to complete this project.

We would like to express our heartiest gratitude to **Md. Safaet Hossain**, Associate professor and Head, Department of CSE, for his kind help to finish our project and special thanks go to the Honorable Dean of Faculty of Science, **Prof. Dr. Md. Shawkut Ali Khan** and also to other faculty member and the staff of CSE department of City University.

We would like to thank our entire course mate in City University, who took part in this discuss while completing the course work.

Finally, we must acknowledge with due respect the constant support and patients of our parents.

Faisal Ahmed

ID: 161412316

Department of CSE

City University

ACKNOWLEDGEMENT

First we express our heartiest thanks and gratefulness to almighty Allah for His divine blessing makes us possible to complete this project successfully.

We fell grateful to and wish our profound our indebtedness to **Md. Ataullah Bhuiyan**, Senior Lecturer, Department of CSE, City University, Dhaka. Deep Knowledge & keen interest of our supervisor in the field of machine learning influenced us to carry out this project .His endless patience ,scholarly guidance ,continual encouragement , constant and energetic supervision, constructive criticism , valuable advice ,reading many inferior draft and correcting them at all stage have made it possible to complete this project.

We would like to express our heartiest gratitude to **Md. Safaet Hossain**, Associate professor and Head, Department of CSE, for his kind help to finish our project and special thanks go to the Honorable Dean of Faculty of Science, **Prof. Dr. Md. Shawkut Ali Khan** and also to other faculty member and the staff of CSE department of City University.

We would like to thank our entire course mate in City University, who took part in this discuss while completing the course work.

Finally, we must acknowledge with due respect the constant support and patients of our parents.

Md. Manwar

ID: 161412323

Department of CSE

City University

ABSTRACT

Diabetes is one of the most deadly and incurable diseases in the world. Aim of this project is to develop a system which can perform early prediction of diabetes for a patient with a higher accuracy by combining the results of different machine learning techniques. It will generate results that make it closer to the real life situations. Much more than huge savings in costs in terms of medical expenses, loss of duty time and usage of critical medical facilities. It will be the important tool for supplementing the medical doctors in performing expert diagnosis. We use different supervised machine learning algorithm like Naive Bayes, Decision Tree, SVM etc. to find out higher accuracy model with larger dataset in global and local by using different machine learning library along with other tools.

Table of Contents

pages

CONTENTS	i-vi
Certificate	i
Declaration	ii
Acknowledgement	iv
Abstract	vi
CHAPTER	1-37
CHAPTER 1: INTRODUCTION	1-5
1.1 Introduction	1
1.2 Problem Statement	1
1.3 Objective	2
1.4 System Architecture	2
1.5 Work Procedure	2
1.6 Project Scope	3
1.7 Using Technology	3
1.7.1 Software	3
1.7.2 Hardware	3
1.8 Team Member Work Distribution	4
1.9 Gantt chart	4
1.10 Budget	5
1.11 Benefits	5
CHAPTER 2: BACKGROUND AND RELATED STUDY	6-8
2.1 Background and related study	6
2.2 Existing Project 1	7

2.3 Existing Project 2	7
2.4 Existing Project 3	8
2.5 Comparing with an Existing Work	8
CHAPTER 3: METHODOLOGY	9-23
3.1 Data Collection	9
3.1.1 Glucose concentration	9
3.1.2 Diastolic blood pressure	9
3.1.3 Skin Thickness (Triceps skin fold thickness (mm))	10
3.1.4 BMI	10
3.1.5 Age	10
3.1.6 Diabetes	10
3.2 Data Pre-processing	11
3.2.1 Data Cleaning	11
3.2.2 Missing Data Checking	12
3.2.3 Missing Data Filled up	13
3.2.4 Data Integration	14
3.2.5 Data Case Checking	14
3.2.6 Split Dataset	15
3.2.7 Training Data	15
3.2.8 Testing Data	15
3.3 Select the right machine learning algorithm	16
3.4 Classification of Machine Learning	16
3.4.1 Supervised learning	16
3.4.2 Unsupervised learning	17

3.4.3 Reinforcement learning	17
3.5 How to choose right algorithm	17
3.5.1 Size of the training data	18
3.5.2 Speed or training time	18
3.5.3 Number of features	18
3.6 Algorithm & Technique	18
3.6.1 Naive Bayes	18
3.6.2 KNN(K- nearest neighbor)	19
3.6.3 Random Forest	20
3.6.4 Logistic regression	21
3.6.5 Decision tree	22
CHAPTER 4: EXPERIMENTAL ANALYSIS	24-29
4.1 Build the model	24
4.2 Testing the model	26
4.3 Comparison of the Training & Testing performance	29
CHAPTER 5: RESULT	30-33
5.1 Case 1	31
5.2 Case 2	32
5.3 Case 3	33
5.4 Discussion	33
CHAPTER 6: CONCLUSION	34-37
6.1 Conclusion	34
6.2 Future Work	34
6.3 References	35

LIST OF FIGURES

FIGURES	pages
1.1 Proposed model diagram	2
3.1 Classification of Machine Learning	16
3.2 Formula of Posterior Probability	19
3.3 Data plotting on KNN	20
3.4 Working of the Random Forest algorithm	21
3.5 Logistic Regression function	22

LIST OF TABLES

TABLES	pages
1.8 Team Member Work Distribution	4
1.10 Budget	5
2.5 Comparing with Existing Work	8

CHAPTER 1

INTRODUCTION

1.1 Introduction

Diabetes is one of the most deadly diseases in the world. It is not only a disease but also a master of various types of diseases such as heart attack, blindness, kidney disease, etc. A common diagnostic procedure is that patients need to visit a diagnostic center, see their doctor, and stay for a day or more to get their reports. In addition, every time they want to get their diagnostic report, they should spend their money on it. Diabetes Mellitus (DM) is defined as a group of metabolic disorders caused by insulin resistance and / or action. Insulin deficiency leads to high levels of sugar (hyperglycemia) and damage to foods high in carbohydrates, fats and proteins. DM is one of the most common endocrine disorders, affecting more than 200 million people worldwide. The onset of diabetes is expected to increase dramatically in the coming years. DM can be divided into many different types. However, there are two major types of clinics, type 1 diabetes (T1D) and type 2 diabetes (T2D), according to the etiopathology of the problem. T2D appears to be the most common form of diabetes (90% of all patients with diabetes), especially those that show insulin resistance. The main causes of T2D include lifestyle, exercise, dietary habits and inheritance, while T1D is thought to be caused by the automatic destruction of the Langerhans islands that hold pancreatic- β cells. T1D affects about 10% of all patients with diabetes worldwide, and 10% of them eventually develop idiopathic diabetes. Other types of DM, classified on the basis of insulin secretion and / or onset, include Gestational Diabetes, endocrinopathies, MODY (Maturity Onset Diabetes in adolescents), neonatal, mitochondrial, and gestational diabetes. Symptoms of DM include polyuria, polydipsia, and excessive weight loss among others. Diagnosis is based on blood sugar levels (fast sugar = 7.0 mmol / L).

1.2 Problem Statement

We are going to build a model using machine learning algorithm which can predict a person's diabetes will have or not at home. We have to collect our dataset. The dataset should be modified. Our model will learn through our given dataset.

1.3 Objectives

The objective of this project is to predict the diabetic cases analyzing which factors responsible for diabetics using data mining methods using machine learning library for the Python with maximum accuracy using “**Pima Indian Dataset**”.

1.4 System Architecture

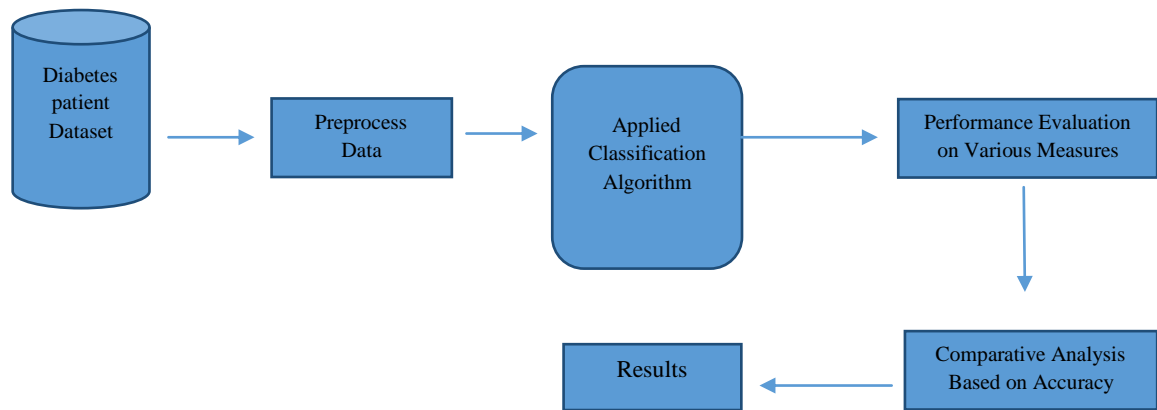


Figure 1.1: Proposed model diagram

1.5 Work Procedure

- **Data Collection:** Data collection is the process of gathering and measuring information on targeted variables in an established system.
- **Data Pre-processing:** Data preprocessing is a data mining technique which is used to transform the raw data in a useful and efficient format.
- **Select the right machine learning algorithm:** After proper evaluation of your problems, you can identify the applicable algorithms which are practical to implement using the available tools.

- **Build the model:** Build the model using suitable tools.
- **Testing the model:** Test the model with accuracy.
- **Model Deployment:** Think that which platform you have to deploy your model and work.

1.6 Project Scope

- The system will present at doctor assistant.
- The user can check his/her diagnose at home.
- It also helps in medical health research

1.7 Using Technology

1.7.1 Software:

- Spyder IDE
- Google Colab
- Python
- Flask
- Heroku As a PaaS
- Scikit-learn
- NumPy
- Pandas
- Matplotlib
- Seaborn etc.

1.7.2 Hardware:

We will deploy our trained model in core i5 2.40 GHz processor computer with 8 GB ram.

1.8 Team Member Work Distribution

Faisal Ahmed <ul style="list-style-type: none"> ➤ Data Collection ➤ Data Pre-Processing ➤ Model Testing 	Md. Manwar <ul style="list-style-type: none"> ➤ Model Analysis ➤ Model Building ➤ Model Testing
---	---

1.9 Gantt chart



1.10 Budget

1	Create dataset (Gathering Data, Data preparation, Data Wrangling, Analyses data)	10000/= tk
2	Train the model	50000/= tk
3	Test the model	40000/= tk
4	Computer for development	100000/=tk
Total=		200000/=tk

1.11 Benefits

- The number of test should be reduced.
- The reduced test plays an important role in time and cost.
- In early the ability to diagnose diabetes plays a vital role for the patient treatment process.

CHAPTER 2

BACKGROUND AND RELATED STUDY

2.1 Background and related study

Diabetes is a chronic disease that has the potential to challenge health care worldwide. According to the International Diabetes Federation 382 million people are living with diabetes worldwide. By 2035, this will double as much as 592 billion. Mellitus or diabetes mellitus is a disease caused by an increase in blood glucose. A variety of traditional methods, based on physical and chemical tests, are available for diabetes testing. However, early diagnosis of diabetes is a very challenging task for medical professionals due to the complex dependence on various factors as diabetes affects human organs such as the kidneys, eye, heart, nerves, foot etc. Data science methods have the potential to benefit other fields of science by shedding new light on common questions. One such function is to assist in making predictions based on medical data. Machine learning is a field of science that emerges in data science that works with the ways machines learn from information.

If diabetes is not controlled properly during pregnancy, the baby is exposed to high blood sugar levels. This can affect the baby and the mother during pregnancy, during birth and after childbirth.

Two types of diabetes can happen in pregnancy. These are:

Pregnancy Diabetes: In this case, you do not have diabetes before pregnancy. She develops it during pregnancy. This type of diabetes goes away after the birth of your baby.

Pre-gestational diabetes: In this case, you have pre-gestational diabetes. You can have type 1 or type 2 diabetes.

In 2011-2012, the average annual number of newly diagnosed cases of type 1 diabetes in the U.S. included 17,900 children and adolescents under 20 years of age.

The annual number of children and adolescents ranging in age from 10 to 19 was diagnosed with type 2 diabetes at 5,300.

2.2 Existing project 1:

Name: Prediction of Diabetes using Classification Algorithms

This paper explained the design of a model that could determine the chances of diabetes in patients with greater accuracy. So three machine learning algorithms namely Tree Decision, SVM and Naive Bayes were used in this trial to diagnose diabetes early. The study was conducted on the Pima Indians Diabetes Database (PIDD) obtained from the UCI machine learning repository. [1]

Accuracy: 76.36% *on-Pima Indians Diabetes Dataset (as they claimed).*

2.3 Existing project 2:

Name: Predicting Diabetes Mellitus With Machine Learning Techniques

This article focuses on your own data to predict diabetes. The data is data from a physical examination of a hospital in Luzhou, China. It has 14 symbols. They randomly selected 68994 healthy population data and details of diabetic patients, respectively as a training set. They used a randomized controlled trial (PCA) and a minimum height (mRMR) to reduce the size. [2]

Accuracy: 77.21% *on -Pima Indians Diabetes Dataset (as they claimed).*

80.84% *on –Luzhou Dataset (as they claimed).*

2.4 Existing project 3:

Name: Prediction of Diabetes by Employing a New Data Mining Approach Which Balances Fitting and Generalization

This paper has described a new approach, called the Homogeneity- Based Algorithm (or HBA) as developed by Pham and Triantaphyllou to optimally control the overfitting and overgeneralization behaviors of classification on the pima indian dataset. The HBA is used in conjunction with traditional classification approaches (such as Support Vector Machines (SVMs), Artificial Neural Networks (ANNs), or Decision Trees (DTs)) to enhance their classification accuracy.[3]

Accuracy: 77.6% on-Pima Indians Diabetes Dataset (as they claimed).

2.5 Comparing with Existing Work

	Existing System	Proposed System
Algorithms	[1]Decision Tree, SVM and Naive Bayes [2]Principal component analysis (PCA) [3]HBA,SVM,ANN,DT	Naïve Bayes
Technology	[1]WEKA tool,[2] WEKA, Matlab.	Python ,Flask, Scikit-learn, NumPy, Pandas, Matplotlib, seaborn etc.
Dataset	[1][3] <i>Pima Indians Diabetes</i> [2] Luzhou <i>Dataset, Pima Indians Diabetes</i>	<i>Pima Indians Diabetes Dataset</i>
Accuracy	[1]76.36%, [2] 77.21% , 80.84% [3]77.6%	75.65%
Deployment	NO	Web App

CHAPTER 3

METHODOLOGY

3.1 Data Collection: Data collection is one of the most important steps in solving any machine learning problem. However, it is also a critical roadblock for many researchers and data scientists.

We collected a dataset entitled "Pima Indian Diabetes" from kaggle and kaggle's largest data science community with powerful tools and resources to achieve data science goals.

```
[ ] data_frame.shape
```

```
↳ (768, 6)
```

The dataset has 768 records along with 6 columns.

```
[36] # iterating the columns
      for col in data_frame.columns:
          print(col)
```

```
↳ glucose_conc
    diastolic_bp
    skin_thickness
    bmi
    age
    diabetes
```

The dataset has 6 columns named glucose concentration, diastolic blood pressure, skin thickness, bmi, age and finally diabetes.

3.1.1 Glucose concentration: The blood sugar level is the concentration of glucose present in the blood of humans and other animals.

The international standard way of measuring blood glucose levels is calculated in milli moles per litre. In the United States, Germany and other countries mass concentration is calculated in milligrams per decilitre.

3.1.2 Diastolic blood pressure: Diastolic blood pressure is the pressure on the blood vessels when the heart muscle relaxes. The diastolic pressure is all time lower than the systolic pressure.

Blood pressure is calculated in units of millimeters of mercury (mmHg).

3.1.3 Skin Thickness (Triceps skin fold thickness (mm)):

A value used to estimate body fat, measured on the right arm halfway between the olecranon process of the elbow and the acromial process of the scapula.

Normal thickness in males is 12 mm and in females 23 mm.

A skinfold caliper named “Harpender calipers” is used to assess the skinfold thickness, so that a prediction of the total amount of body fat can be made. This method is based on the hypothesis that the body fat is equally distributed over the body and that the thickness of the skinfold is a measure for subcutaneous fat.

3.1.4 BMI: Bmi means Body Mass Index. It is a calculation of a person’s weight with respect to his or her height. It is more of an indicator than a direct calculation of a person’s total body fat.

BMI, more often than not, is associated with complete body fat. This is understandable that as BMI score goes up, so does a person's body fat.

Bmi calculation formula: $\text{weight (kg)} / [\text{height (m)}]^2$

3.1.5 Age: The length of time that a person has lived or a thing has existed.

3.1.6 Diabetes: Diabetes is a disease that occurs when your blood glucose, also called blood sugar, is too high. Blood glucose is your main source of energy and comes from the food you eat. Insulin is a hormone that made by the pancreas, helps glucose from food get into your cells to be used for energy. Many times, your body doesn’t make enough—or any—insulin or doesn’t use insulin well. Then glucose stays in your blood and doesn’t reach your cells.

After having sometime, having too much glucose in your blood can cause health problems. Although diabetes has no cure, you can take steps to manage your diabetes and stay healthy.

And this attribute is our outcome.

3.2 Data Pre-processing: Data preprocessing is a data mining technique that involves transforming raw data into an understandable format. In this present world, data is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors. Data preprocessing is a tested method of resolving such issues. Data preprocessing prepares raw data for further processing.

3.2.1 Data Cleaning: Data is cleaned through processes such as filling in missing values, smoothing the noisy data, or resolving the inconsistencies in the data.

glucose_conc	diastolic_bp	skin_thickness	bmi	age	diabetes
148	72	35		50	1
85	66	29		31	0
183	64		23.3		1
89	66	23		21	0
137	40	35	43.1	33	1
116	74		25.6	30	0
78	50	32	31		1
115	0	0	35.3	29	0
197	70	45	30.5	53	1
125	96		0	54	1
110	92	0	37.6	30	0
168	74	0	38	34	1

We see that there have many null values. And We check it out by the code.

```
print(data_frame.isnull().values.any())
```

```
True
```

The output true means the dataset contained with null values. And the null record put in NaN .

	glucose_conc	diastolic_bp	skin_thickness	bmi	age	diabetes
0	148.0	72.0	35.0	NaN	50.0	1
1	85.0	66.0	29.0	26.6	31.0	0
2	NaN	64.0	NaN	23.3	32.0	1
3	89.0	66.0	23.0	NaN	NaN	0
4	137.0	40.0	35.0	43.1	33.0	1
5	116.0	NaN	NaN	25.6	30.0	0
6	NaN	50.0	32.0	31.0	26.0	1
7	115.0	NaN	NaN	35.3	29.0	0
8	197.0	70.0	45.0	30.5	53.0	1
9	125.0	96.0	0.0	NaN	54.0	1

So we should remove all the null values for training the machine well.

3.2.2 Missing Data Checking:

```
[ ] print("# rows in dataframe {0}".format(len(data_frame)))
print("# rows missing glucose_conc: {0}".format(len(data_frame.loc[data_frame['glucose_conc'] == 0])))
print("# rows missing diastolic_bp: {0}".format(len(data_frame.loc[data_frame['diastolic_bp'] == 0])))
print("# rows missing thickness: {0}".format(len(data_frame.loc[data_frame['skin_thickness'] == 0])))
print("# rows missing bmi: {0}".format(len(data_frame.loc[data_frame['bmi'] == 0])))
print("# rows missing age: {0}".format(len(data_frame.loc[data_frame['age'] == 0])))
```

```
[ ] # rows in dataframe 768
# rows missing glucose_conc: 5
# rows missing diastolic_bp: 35
# rows missing thickness: 227
# rows missing bmi: 11
# rows missing age: 0
```

We have 768 rows but 5 fields of glucose concentration, 35 fields of diastolic bp , 227 fields of skin thickness and 11 fields of bmi are missing.

3.2.3 Missing Data Filled up: Datasets may have missing values, and this can cause problems for many machine learning algorithms.

As such, it is good practice to identify and replace missing values for each column in your input data before modeling your prediction task. This is called missing data imputation.

A popular method is to calculate a statistical value for data imputation in each column (such as a mean) and replace all missing values for that column by statistics. It is a popular method because the statistic is easy to calculate using the training dataset and because it often results in good performance.

```
#from sklearn.preprocessing import Imputer
#from sklearn.impute import SimpleImputer
#Impute with mean all 0 readings
fill_0 = SimpleImputer(missing_values=0, strategy="mean")
X_train = fill_0.fit_transform(X_train)
X_test = fill_0.fit_transform(X_test)
```

X_train[:5]

```
array([[ 94.      , 72.31838565, 28.73795181, 32.32571429,
        25.      ],
       [125.      , 96.      , 28.73795181, 32.32571429,
        54.      ],
       [111.      , 86.      , 19.      , 30.1      ,
        23.      ],
       [196.      , 90.      , 28.73795181, 39.8      ,
        41.      ],
       [158.      , 90.      , 28.73795181, 31.6      ,
        66.      ]])
```

Find and Replace

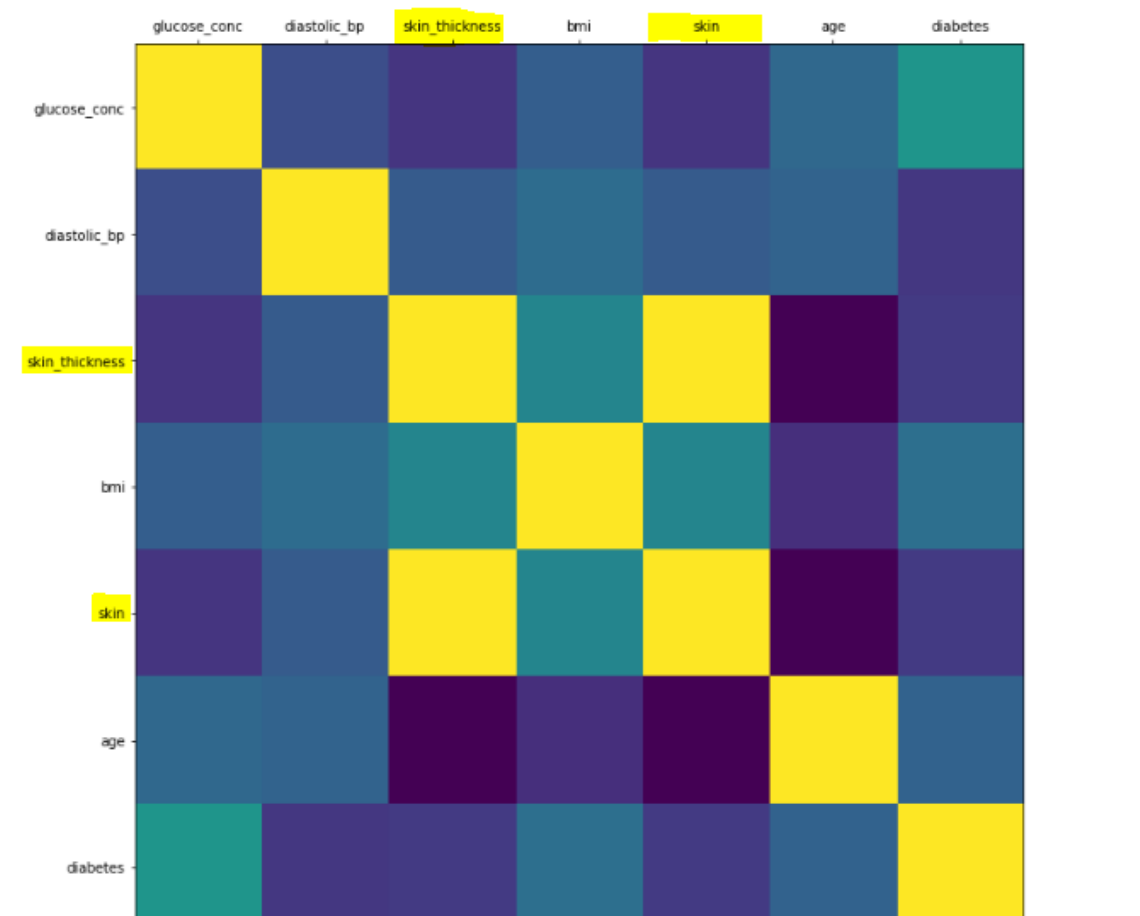
Find what: 125

	A	B	C	D	E	F
5	89	66	23	28.1	21	0
6	137	40	35	43.1	33	1
7	116	74	0	25.6	30	0
8	78	50	32	31	26	1
9	115	0	0	35.3	29	0
10	197	70	45	30.5	53	1
11	125	96	0	0	54	1
12	110	92	0	37.6	30	0
13	168	74	0	38	34	1
14	139	80	0	27.1	57	0
15	189	60	23	30.1	59	1
16	166	72	19	25.8	51	1
17	100	0	0	30	32	1

diabetes

We see that, the records are filled with mean values.

3.2.4 Data Integration: Data with different representations are put together and conflicts within the data are resolved.



3.2.5 Data Case Checking: Check the dataset what percentage of outcomes are true and false.

```
[54] num_true = len(data_frame.loc[data_frame['diabetes'] == True])
      num_false = len(data_frame.loc[data_frame['diabetes'] == False])
      print ("Number of True Cases: {0} ({1:2.2f}%)".format(num_true, (num_true / (num_true + num_false)) * 100))
      print ("Number of False Cases: {0} ({1:2.2f}%)".format(num_false, (num_false / (num_true + num_false)) * 100))
```

➡ Number of True Cases: 268 (34.90%)
 Number of False Cases: 500 (65.10%)

3.2.6 Split Dataset: Now our dataset is almost modified. And we should split our dataset into two important components like training and testing.

```
from sklearn.model_selection import train_test_split

feature_column_names = ['glucose_conc', 'diastolic_bp', 'skin_thickness', 'bmi', 'age']

predicted_class_name = ['diabetes']

# Getting feature variable values

X = data_frame[feature_column_names].values
y = data_frame[predicted_class_name].values

# Saving 40% for testing
split_test_size = 0.40

# Splitting using scikit-learn train_test_split function

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = split_test_size, random_state = 42)

[52] print("{0:0.2f}% in training set".format((len(X_train)/len(data_frame.index)) * 100))
      print("{0:0.2f}% in test set".format((len(X_test)/len(data_frame.index)) * 100))

59.90% in training set
40.10% in test set
```

3.2.7 Training Data: The examination of the training set create the experience that the algorithm used by learning algorithm.

For supervised learning problems, each examination consists of an observed output variable and one or more observed input variables.

We take 60% of the total data for training.

3.2.8 Testing Data: The test set is a set of examination used to calculate the models performance. It is important that no observations from the training set are included in the test set. If the test set contains examples from the training set, it will be difficult to test whether the algorithm has learned to perform normally from the training set or simply memorizes it.

A well-integrated system will be able to perform a task effectively with new data . In contrast, a system that memorizes the training data by learning an overly complex model could predict the values of the response variable for the training set accurately, but will fail to predict the value of the response variable for new examples. Remembering the training set is called over-fitting. A system that memorizes its observations may not perform its task well, as it could memorize relations and structures that are noise or coincidence. Balancing memorization and generalization, or over-fitting and under-fitting, is a problem common to many machine learning algorithms. We take 40% of the total data for testing.

3.3 Select the right machine learning algorithm: Machine learning is a growing technology that enables computers to learn automatically from past data. Machine learning uses various algorithms to build mathematical models and to make predictions using historical data or information.

3.4 Classification of Machine Learning

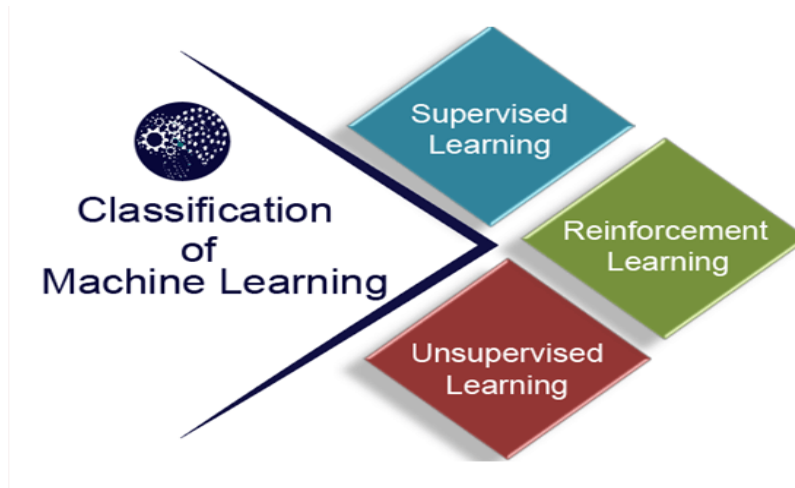


Figure 3.1: Classification of Machine Learning

3.4.1 Supervised learning: Supervised learning is a type of machine learning method in which we provide sample labeled data to the machine learning system in order to train it, and on that basis, it predicts the output.

The system creates a model using labeled data to understand the datasets and learn about each data, once the training and processing are done then we test the model by providing a sample data to check whether it is predicting the exact output or not.

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

- **Classification**
- **Regression**

3.4.2 Unsupervised learning: Unsupervised learning is a learning method in which a machine learns without any supervision.

The training is provided to the machine with the set of data that has not been labeled, classified, or categorized, and the algorithm needs to act on that data without any supervision. The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns.

It can be classified into two categories of algorithms:

- **Clustering**
- **Association**

3.4.3 Reinforcement learning: Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action. The agent learns automatically with these feedbacks and improves its performance. In reinforcement learning, the agent interacts with the environment and explores it. The goal of an agent is to get the most reward points, and hence, it improves its performance.

Our developed method is classification and it is one of the categories of supervised learning.

3.5 How to choose right algorithm: There are many classification algorithms.

Some of the valuable features while selecting an algorithm.

3.5.1 Size of the training data: It is usually recommended to gather a good amount of data to get reliable predictions. The availability of data is a constraint. So, the training dataset is smaller or if the dataset has a fewer number of features and a higher number of features like genetics or textual data, choose algorithms like Linear regression, Naïve Bayes, or Linear SVM.

If the training data is sufficiently large and the number of observations is higher as compared to the number of features, one can go for algorithms like KNN, Decision trees, or kernel SVM.

3.5.2 Speed or training time: Higher accuracy typically means higher training time. Also, algorithms takes huge time to train on large training data. In real-world applications, the selection of algorithm is driven by these two factors.

Some algorithms like Naïve Bayes and Linear and Logistic regression are easy to implement and quick to run. Some algorithms like SVM, which involve tuning of parameters, Neural networks with high convergence time, and random forests, need a lot of time to train the data.

3.5.3 Number of features: The dataset may have a large number of features that may not all be relevant and significant. For a certain type of data, such as genetics or textual, the number of features can be very large compared to the number of data points.

SVM is more suited in case of data with large feature space and lesser observations. PCA and feature selection techniques should be used to reduce dimensionality and select important features.

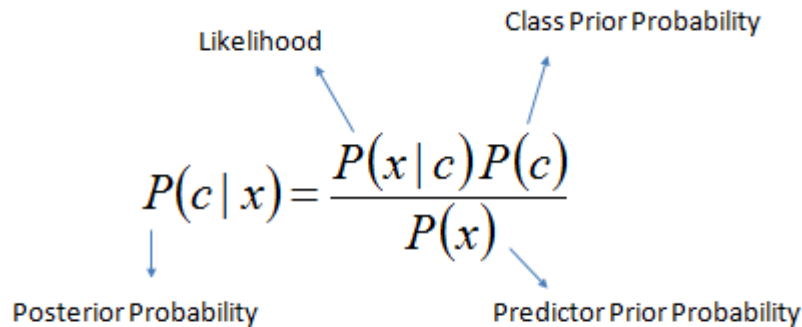
3.6 Algorithm & Technique:

3.6.1 Naive Bayes: This is a classification technique based on an assumption of independence between predictors or what's known as Bayes' theorem. In simple terms, a Naive Bayes classifier thinks that the presence of a particular feature in a class is unrelated to the presence of any other feature.

For example, a fruit can be considered an apple if it is red, round, and about 3 inches in diameter. And if these features depend on each other or in the presence of other features, Naive Bayes Classifier will consider that all of these properties will independently contribute to the probability that this fruit is an apple.

Building a Bayesian model is simple and particularly functional in case of enormous data sets. Along with simplicity, Naive Bayes is known to outperform sophisticated classification methods as well.

Bayes theorem provides a rules of calculating posterior probability $P(c|x)$ from $P(c)$, $P(x)$ and $P(x|c)$. The expression for Posterior Probability is as follows.

$$P(c | x) = \frac{P(x | c)P(c)}{P(x)}$$


$$P(c | X) = P(x_1 | c) \times P(x_2 | c) \times \dots \times P(x_n | c) \times P(c)$$

Figure 3.2: **Formula of Posterior Probability**

Here,

- $P(c|x)$ is the posterior probability of class that is given predictor (attribute).
- $P(c)$ is the prior probability of class.
- $P(x|c)$ is the probability of predictor given class.
- $P(x)$ is the prior probability of predictor.

3.6.2 KNN(K- nearest neighbor): K nearest neighbors is a simple algorithm used for not only classification problem and but also regression problems. It retains all available cases to separate new cases by a majority vote of its k neighbors. The case given to the class is most common among its K nearest neighbors measured by a distance function (Euclidean, Manhattan, Minkowski, and Hamming).

When the three former distance functions are used for continuous variables, Hamming distance function is used for categorical variables. If $K = 1$, So the case is nicely assigned to the class of its nearest neighbor. By times, choosing K turns out to be a challenge while performing KNN modeling.

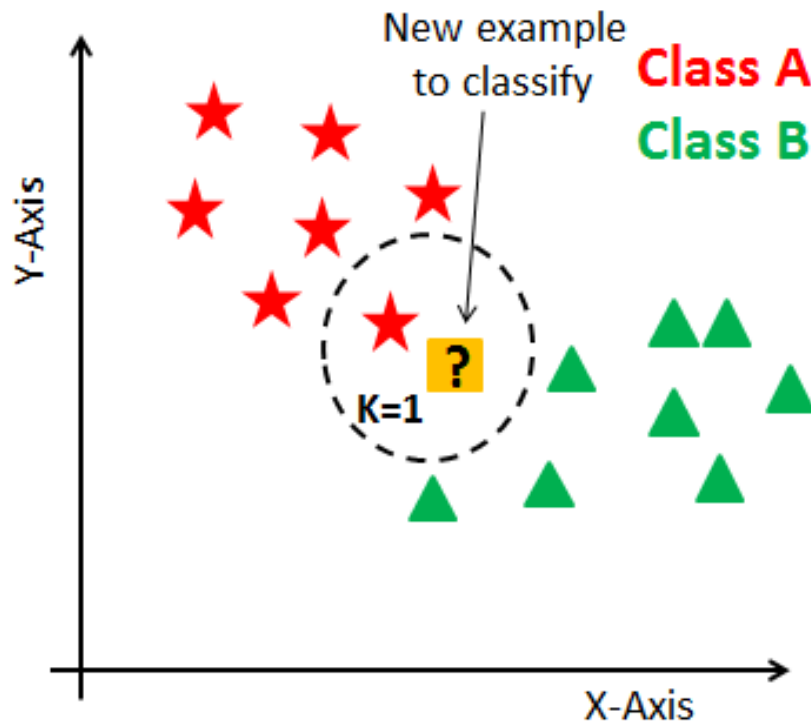


Fig 3.3: Data plotting on KNN

3.6.3 Random Forest: Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It could be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

The below diagram explains the working of the Random Forest algorithm:

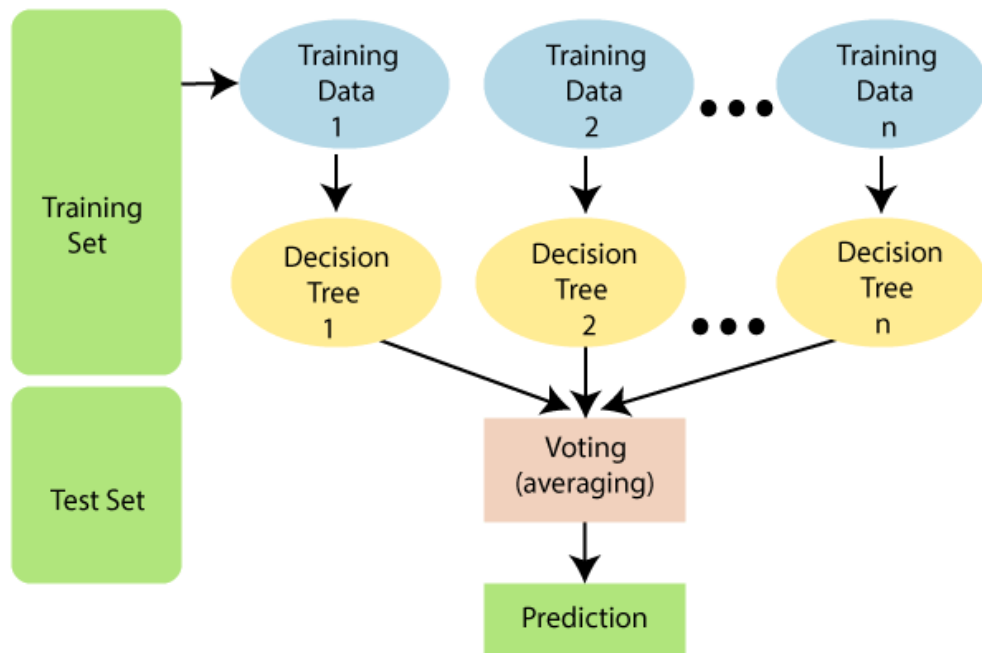


Fig 3.4: Working of the Random Forest algorithm

3.6.4 Logistic regression: Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.

- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.
- Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification.
- The below image is showing the logistic function:

The below image is showing the logistic function:

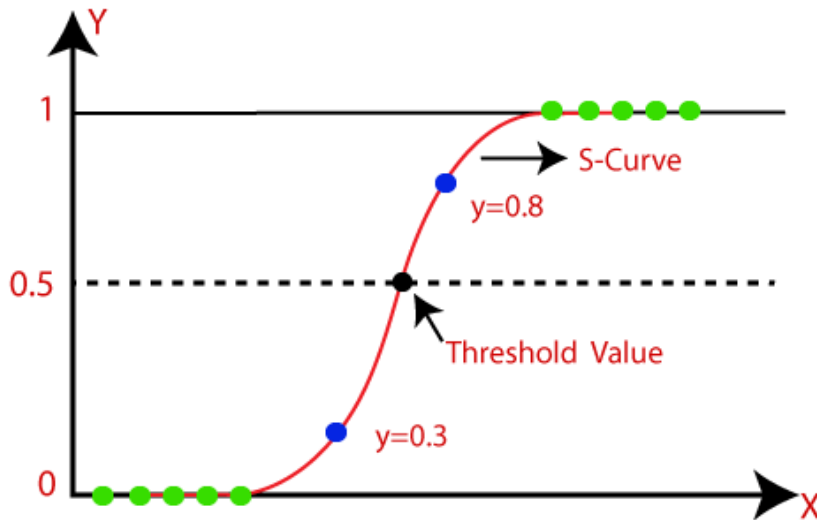
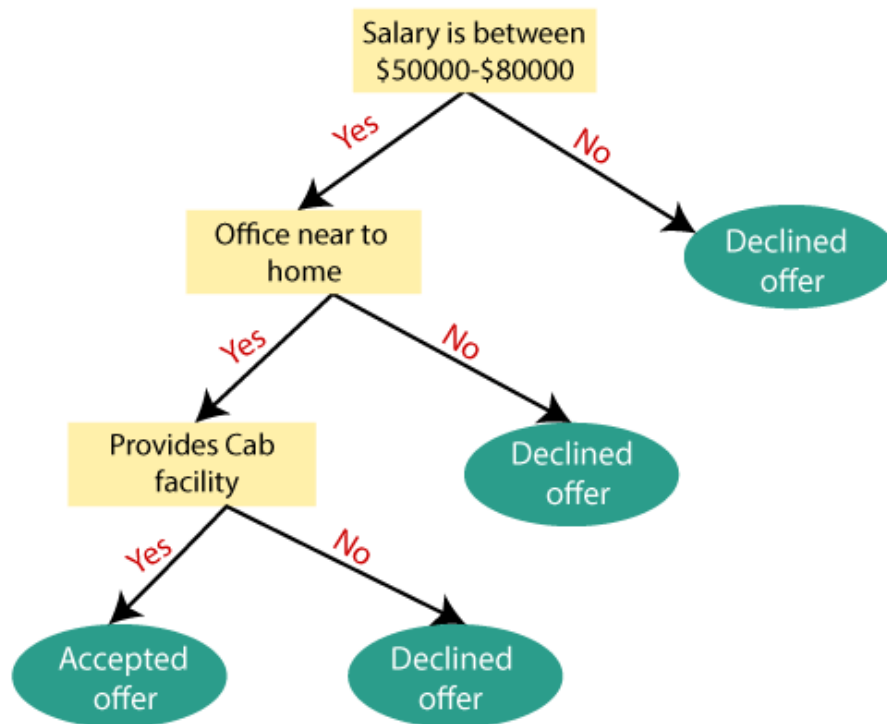


Fig 3.5: Logistic Regression function

3.6.5 Decision tree:

- Decision tree algorithm is under the supervised machine learning categories. In generally, decision tree algorithm is used to solve not only regression problems but also classification problems.
- Decision tree uses the tree representation to solve the problem where each leaf node corresponds to a class label and attributes are represented on the internal node of the tree.
- We can represent any boolean function in discrete attributes using the decision tree.

Example: Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or Not. So, to solve this problem, the decision tree starts with the root node (Salary attribute by Attribute Selection Measure). The root node splits further into the next decision node (distance from the office) and one leaf node based on the corresponding labels. After that, the next decision node further gets split into one decision node (Cab facility) and one leaf node. And last, the decision node splits into two leaf nodes (Accepted offers and Declined offer).



Entropy: Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

$$\text{Entropy}(s) = -P(\text{yes}) \log_2 P(\text{yes}) - P(\text{no}) \log_2 P(\text{no})$$

Where,

- **S= Total number of samples**
- **P(yes)= probability of yes**
- **P(no)= probability of no**

Information gain: Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.

$$\text{Information Gain} = \text{Entropy}(S) - [(\text{Weighted Avg}) * \text{Entropy}(\text{each feature})]$$

CHAPTER 4

EXPERIMENTAL ANALYSIS

4.1 Build the model:

Naïve Bayes:

```
[ ] from sklearn.naive_bayes import GaussianNB

# create Gaussian Naive Bayes model object and train it with the data
nb_model = GaussianNB()

nb_model.fit(X_train, y_train.ravel())

☞ GaussianNB(priors=None, var_smoothing=1e-09)

[ ] # This returns array of predicted results
prediction_from_trained_data = nb_model.predict(X_train)
```

Random Forest:

```
[ ] from sklearn.ensemble import RandomForestClassifier

# Create a RandomForestClassifier object
rf_model = RandomForestClassifier(random_state=42)

rf_model.fit(X_train, y_train.ravel())
```

Logistic Regression:

```
▶ from sklearn.linear_model import LogisticRegression

# Create a RandomForestClassifier object
lr_model = LogisticRegression(random_state=42)

lr_model.fit(X_train, y_train.ravel())
```

K- nearest neighbor:

```
[ ] # K nearest neighbors Algorithm
from sklearn.neighbors import KNeighborsClassifier
knn_model= KNeighborsClassifier(n_neighbors = 24, metric = 'minkowski', p = 2)
knn_model.fit(X_train, y_train.ravel())
```

Decision Tree:

```
[ ] # Decision tree Algorithm
from sklearn.tree import DecisionTreeClassifier
decisionTree_model = DecisionTreeClassifier(random_state=42)
decisionTree_model.fit(X_train, y_train.ravel())
```

4.2 Testing the model: We get various training and testing performance on different algorithm.

For Naïve Bayes:

```
[ ] # performance metrics library
    from sklearn import metrics

    # get current accuracy of the model

    accuracy = metrics.accuracy_score(y_train, prediction_from_trained_data)

    print ("Accuracy of our naive bayes model is : {0:.4f}".format(accuracy))
```

➞ Accuracy of our naive bayes model is : 0.7435

```
[ ] # this returns array of predicted results from test_data
    prediction_from_test_data = nb_model.predict(X_test)

    accuracy = metrics.accuracy_score(y_test, prediction_from_test_data)

    print ("Accuracy of our naive bayes model is: {0:.4f}".format(accuracy))
```

➞ Accuracy of our naive bayes model is: 0.7565

For Random Forest:

```
[ ] rf_predict_train = rf_model.predict(X_train)

    #get accuracy
    rf_accuracy = metrics.accuracy_score(y_train, rf_predict_train)

    #print accuracy
    print ("Accuracy: {0:.4f}".format(rf_accuracy))
```

➞ Accuracy: 1.0000

```
[ ] rf_predict_test = rf_model.predict(X_test)

    #get accuracy
    rf_accuracy_testdata = metrics.accuracy_score(y_test, rf_predict_test)

    #print accuracy
    print ("Accuracy: {0:.4f}".format(rf_accuracy_testdata))
```

➞ Accuracy: 0.7630

For Logistic Regression:

```
[28] lr_predict_train = lr_model.predict(X_train)

#get accuracy
lr_accuracy = metrics.accuracy_score(y_train, lr_predict_train)

#print accuracy
print ("Accuracy: {:.4f}".format(lr_accuracy))
```

➞ Accuracy: 0.7609

```
[29] lr_predict_test = lr_model.predict(X_test)

#get accuracy
lr_accuracy_testdata = metrics.accuracy_score(y_test, lr_predict_test)

#print accuracy
print ("Accuracy: {:.4f}".format(lr_accuracy_testdata))
```

➞ Accuracy: 0.7500

For K- nearest neighbor:

```
[31] knn_predict_train = knn_model.predict(X_train)

#get accuracy
knn_accuracy = metrics.accuracy_score(y_train, knn_predict_train)

#print accuracy
print ("Accuracy: {:.4f}".format(knn_accuracy))
```

➞ Accuracy: 0.7652

```
[32] knn_predict_test = knn_model.predict(X_test)

#get accuracy
knn_accuracy_testdata = metrics.accuracy_score(y_test, knn_predict_test)

#print accuracy
print ("Accuracy: {:.4f}".format(knn_accuracy_testdata))
```

➞ Accuracy: 0.7370

For Decision Tree:

```
[34] decisionTree_predict_train = decisionTree_model.predict(X_train)

#get accuracy
decisionTree_accuracy = metrics.accuracy_score(y_train, decisionTree_predict_train)

#print accuracy
print ("Accuracy: {:.4f}".format(decisionTree_accuracy))
```

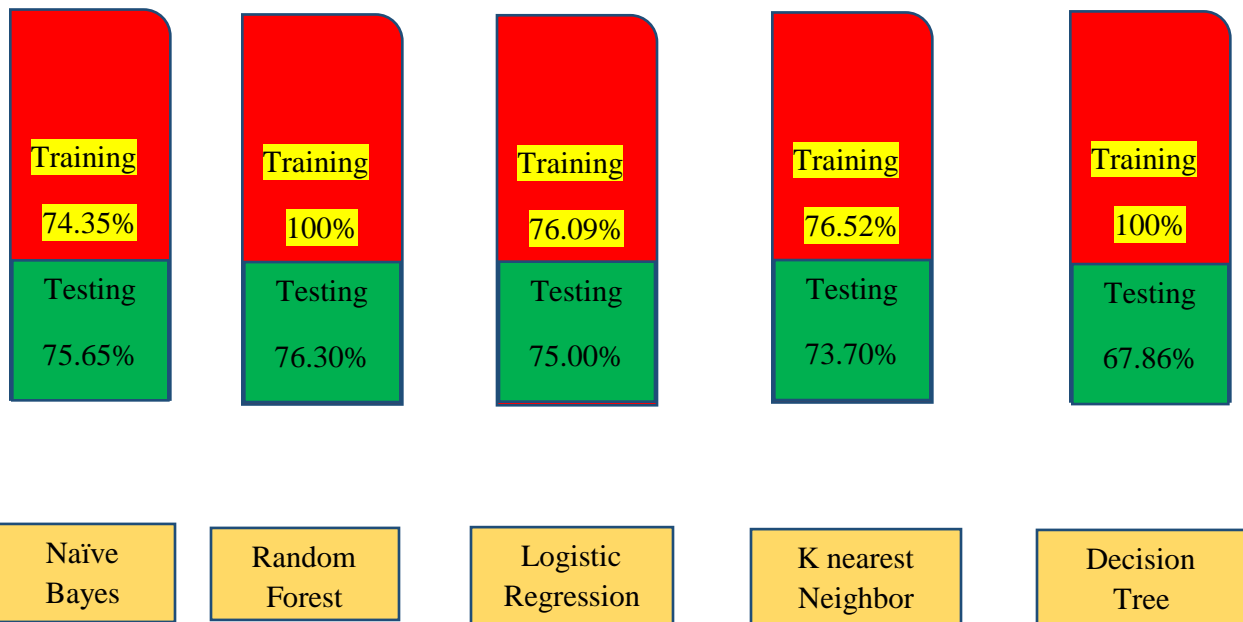
Accuracy: 1.0000

```
[35] decisionTree_predict_test = decisionTree_model.predict(X_test)

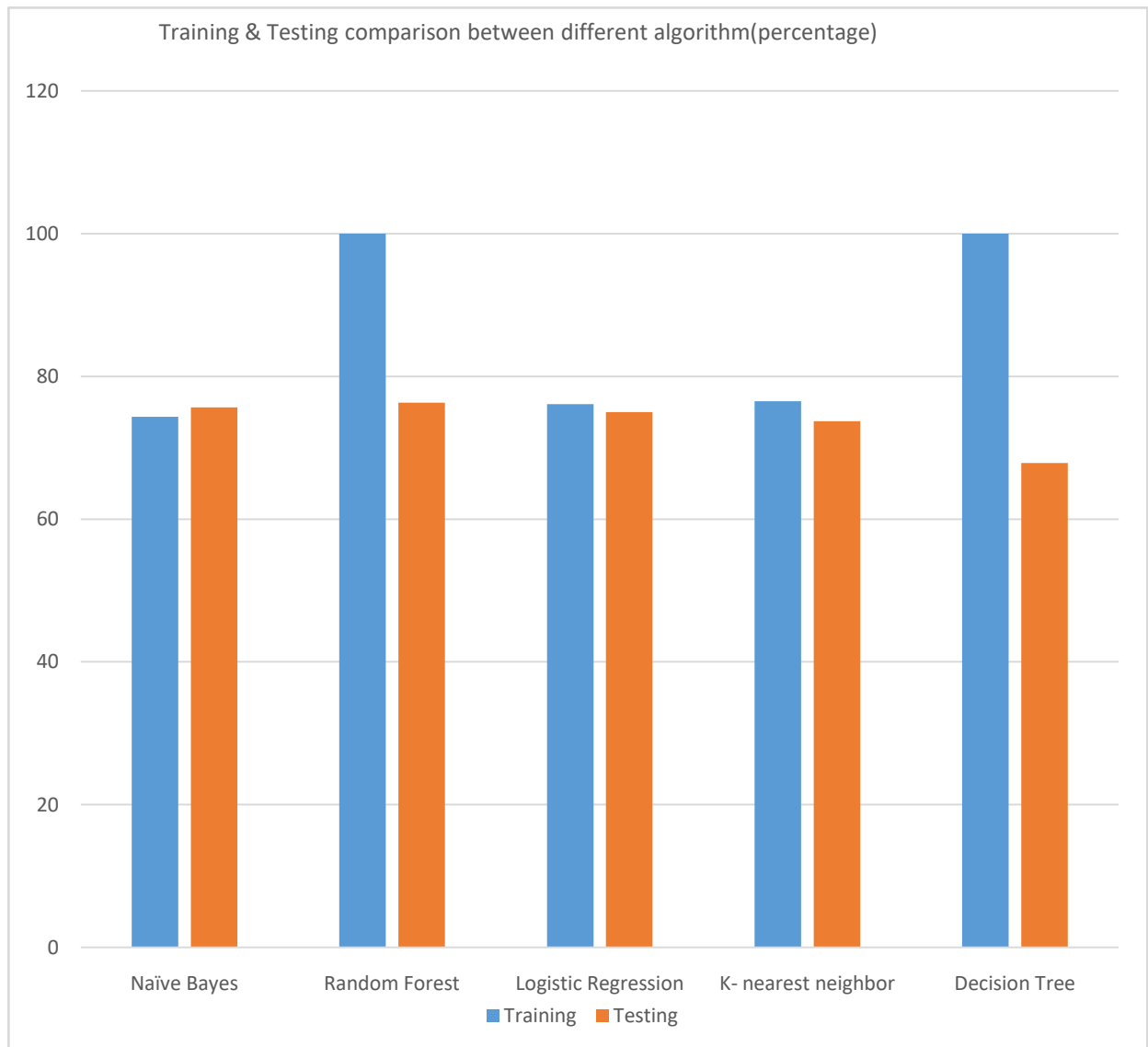
#get accuracy
decisionTree_accuracy_testdata = metrics.accuracy_score(y_test, decisionTree_predict_test)

#print accuracy
print ("Accuracy: {:.4f}".format(decisionTree_accuracy_testdata))
```

Accuracy: 0.6786



4.3 Comparison of the Training & Testing performance:



We use five different algorithms like Naïve bayes, Random forest, Logistic regression, K-nearest neighbor and decision tree. We get different training and testing performance among different algorithm. We get 100 percent training performance of Random forest and Decision tree algorithm but choose Naïve bayes algorithm in our project. Because Random forest and Decision tree algorithm fall into overfitting problem.

CHAPTER 5

RESULT

Result: Although the testing performance of Random forest algorithm is higher than naïve bayes but we choose naïve bayes algorithm for our prediction. Because random forest trained data 100% that means it faces overfitting problem.

Overfitting refers that the model's training data is too well.

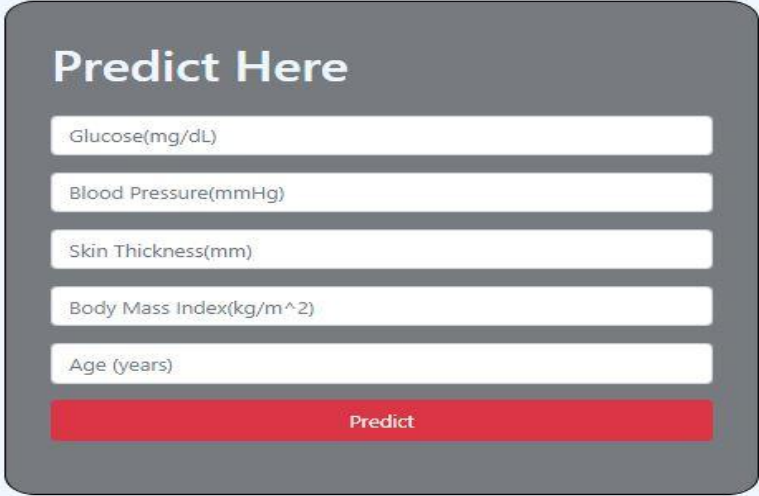
Overfitting occurs when a model reads the details and noise in training data to the detriment of the model's performance on new data.

This means that noise or random variability in training data is taken and read as ideas by model.

The problem is that these concepts do not apply to new data and have a negative impact in the ability of models to perform integration.

Anyone can be able to verify a person's chances of getting diabetes through in this Web App.

Web App link: <https://p2diabetes.herokuapp.com/>



5.1 Case1: (Compare prediction with dataset)

1	glucose conc	diastolic_bp	skin_thickness	bmi	age	diabetes
2	148	72	35	33.6	50	1
3	85	66	29	26.6	31	0
4	183	64	0	23.3	32	1
5	89	66	23	28.1	21	0
6	137	40	35	43.1	33	1
7	116	74	0	25.6	30	0
8	78	50	32	31	26	1
9	115	0	0	35.3	29	0
10	197	70	45	30.5	53	1

Predict Here

Predict

Prediction: You may have a chance of diabetes.

While diabetes is incurable, a person can stay in remission for a long time. No cure for diabetes currently exists, but the disease can go into remission.

You have to maintain your BMI (18.5 to 24.9)

You blood sugar level should keep below 5.5 mmol/L

You should have a blood pressure of no more than 130/80

Home

Here, we see that, the model can predict accurately according to the dataset for true cases.

5.2 Case2: (Compare prediction with dataset)

1	glucose_conc	diastolic_bp	skin_thickness	bmi	age	diabetes
2	148	72	35	33.6	50	1
3	85	66	29	26.6	31	0
4	183	64	0	23.3	32	1
5	89	66	23	28.1	21	0
6	137	40	35	43.1	33	1
7	116	74	0	25.6	30	0
8	78	50	32	31	26	1
9	115	0	0	35.3	29	0
10	197	70	45	30.5	53	1

Predict Here

Predict

Prediction: Don't Worry.
You may have no chance of
diabetes.

Home

Here, we see that, the model can predict accurately according to the dataset for false cases.

5.3 Case3: (Compare prediction with dataset)

418	97	68	21	27.2	22	0
419	144	82	32	38.5	37	1
420	83	68	0	18.2	27	0
421	129	64	29	26.4	28	1
422	119	88	41	45.3	26	0
423	94	68	18	26	21	0
424	102	64	46	40.6	21	0
425	115	64	22	30.8	21	0
426	151	78	32	42.9	36	1
427	184	78	39	37	31	1
428	94	0	0	0	25	0
429	181	64	30	34.1	38	1
430	135	94	46	40.6	26	0
431	95	82	25	35	43	1
432	99	0	0	22.2	23	0
433	89	74	16	30.4	38	0
434	80	74	11	30	22	0
435	139	75	0	25.6	29	0
436	90	68	8	24.5	36	0
437	141	0	0	42.4	29	1

```
[20] X_train[:5]
```

```
array([[ 94.      , 72.31838565, 28.73795181, 32.32571429,
        25.      ],
       [125.      , 96.      , 28.73795181, 32.32571429,
        54.      ],
       [111.      , 86.      , 19.      , 30.1      ,
        23.      ],
       [196.      , 90.      , 28.73795181, 39.8      ,
        41.      ],
       [158.      , 90.      , 28.73795181, 31.6      ,
        66.      ]])
```

Predict Here

Prediction: Don't Worry.
You may have no chance of
diabetes.

Here, we see that, the model can predict accurately according to the modified dataset. Finally, we can say that, our model can predict accurately in maximum times.

5.4 Discussion:

In our dataset, we see that when the outcome is 0 that means there have no chance of diabetes and when the outcome is 1 that means there have a chance of diabetes. And our web app is inform you when you have no diabetes and our web app gives suggestion when you have a chance of diabetes.

CHAPTER 6

CONCLUSION

6.1 Conclusion

The result of the project show the predictions system that is capable of predicting diabetes effectively, efficiently and most importantly, timely. That means the project will capable of making decisions towards patient health risks. It generates results that make it closer to the real life situations. By this project, huge savings in costs of medical expenses, loss of duty time and usage of critical medical facilities. The system performs good prediction with less error and this technique is ready to be an important tool for supplementing the medical doctors in performing expert diagnosis.

6.2 Future work

- Improving Performance
- Mobile base Application
- User Account System
- Automatic Treatment or Recommendation

6.3 References

- [1] D. Sisodia and D. S. Sisodia, "Prediction of Diabetes using Classification Algorithms," *Procedia Computer Science*, vol. 132, pp. 1578–1585, 2018.
- [2] Q. Zou, K. Qu, Y. Luo, D. Yin, Y. Ju, and H. Tang, "Predicting Diabetes Mellitus With Machine Learning Techniques," *Frontiers in Genetics*, vol. 9, Nov. 2018.
- [3] H. N. A. Pham and E. Triantaphyllou, "Prediction of Diabetes by Employing a New Data Mining Approach Which Balances Fitting and Generalization," *Studies in Computational Intelligence*, pp. 11–26.
- [4] Diabetesresearch.org. n.d. *Diabetes Statistics*. [online] Available at: <<https://www.diabetesresearch.org/diabetes-statistics>>
- [5] Urmc.rochester.edu. n.d. *Content - Health Encyclopedia - University Of Rochester Medical Center*. [online] Available at: <<https://www.urmc.rochester.edu/encyclopedia/content.aspx?ContentTypeID=90&ContentID=P02354>>
- [6] Lionbridge AI. n.d. *The Best Data Collection Tools For Machine Learning / Lionbridge AI*. [online] Available at: <<https://lionbridge.ai/articles/best-data-collection-tools-for-machine-learning/>>
- [7] Information, H., Overview, D., Diabetes?, W., Diabetes?, W., Center, T. and Health, N., n.d. *What Is Diabetes? / NIDDK*. [online] National Institute of Diabetes and Digestive and Kidney Diseases. Available at: <<https://www.niddk.nih.gov/health-information/diabetes/overview/what-is-diabetes>>
- [8] TheFreeDictionary.com. n.d. *Triceps Skin-Fold Thickness*. [online] Available at: <<https://medical-dictionary.thefreedictionary.com/triceps+skin-fold+thickness>>
- [9] Techopedia.com. n.d. *What Is Data Preprocessing? - Definition From Techopedia*. [online] Available at: <<https://www.techopedia.com/definition/14650/data-preprocessing>>
- [10] Brownlee, J., n.d. *Statistical Imputation For Missing Values In Machine Learning*. [online] Machine Learning Mastery. Available at: <<https://machinelearningmastery.com/statistical-imputation-for-missing-values-in-machine-learning/>>

- [11] Tutorialspoint.com. n.d. *Training Data And Test Data - Tutorialspoint*. [online] Available at: <https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_training_test_data.htm>
- [12] www.javatpoint.com. n.d. *Machine Learning Tutorial / Machine Learning With Python - Javatpoint*. [online] Available at: <https://www.javatpoint.com/machine-learning>
- [13] GeeksforGeeks. n.d. *ML / Types Of Learning – Supervised Learning - Geeksforgeeks*. [online] Available at: <https://www.geeksforgeeks.org/ml-types-learning-supervised-learning/>
- [14] KDnuggets. n.d. *An Easy Guide To Choose The Right Machine Learning Algorithm - Kdnuggets*. [online] Available at: <<https://www.kdnuggets.com/2020/05/guide-choose-right-machine-learning-algorithm.html>>
- [15] R, 6. and Ray, S., n.d. *Learn Naive Bayes Algorithm / Naive Bayes Classifier Examples*. [online] Analytics Vidhya. Available at: <<https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/>>
- [16] Medium. n.d. *Machine Learning Basics With The K-Nearest Neighbors Algorithm*. [online] Available at: <[https://medium.com/m/global-identity?redirectUrl=https%3A%2F%2Ftowardsdatascience.com%2Fmachine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761#:~:text=Summary-,The%20k%2Dnearest%20neighbors%20\(KNN\)%20algorithm%20is%20a%20simple,that%20data%20in%20use%20grows.>](https://medium.com/m/global-identity?redirectUrl=https%3A%2F%2Ftowardsdatascience.com%2Fmachine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761#:~:text=Summary-,The%20k%2Dnearest%20neighbors%20(KNN)%20algorithm%20is%20a%20simple,that%20data%20in%20use%20grows.>)>
- [17] DataCamp Community. n.d. *KNN Classification Using Scikit-Learn*. [online] Available at: <https://www.datacamp.com/community/tutorials/k-nearest-neighbor-classification-scikit-learn?fbclid=IwAR0KPHVElv--g6r_UTg5whV92RW0BjKURly6H4Hk3Dp3wwgrpX5jZyrjE1g>
- [18] www.javatpoint.com. n.d. *Machine Learning Random Forest Algorithm - Javatpoint*. [online] Available at: <<https://www.javatpoint.com/machine-learning-random-forest-algorithm>>

[19] www.javatpoint.com. n.d. *Logistic Regression In Machine Learning - Javatpoint*. [online] Available at: <<https://www.javatpoint.com/logistic-regression-in-machine-learning>>

[20] GeeksforGeeks. n.d. *Decision Tree Introduction With Example - Geeksforgeeks*. [online] Available at: <<https://www.geeksforgeeks.org/decision-tree-introduction-example/>>

[21] www.javatpoint.com. n.d. *Machine Learning Decision Tree Classification Algorithm - Javatpoint*. [online] Available at: <<https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm>>

[22] Brownlee, J., n.d. *Overfitting And Underfitting With Machine Learning Algorithms*. [online] Machine Learning Mastery. Available at: <<https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/#:~:text=Overfitting%20in%20Machine%20Learning,the%20model%20on%20new%20data.>>