

REPORT OF PROFESSIONAL TRAINING

at

Sathyabama Institute of Science and Technology

(DEEMED TO BE UNIVERSITY)

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering Degree in

INFORMATION TECHNOLOGY

By

SAYED SAJID ALI
(Reg. No. 37120065)



DEPARTMENT OF INFORMATION TECHNOLOGY
SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU.

DECEMBER 2020

**SATHYABAMA INSTITUTE OF SCIENCE
AND TECHNOLOGY**

(DEEMED TO BE UNIVERSITY)

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

www.sathyabama.ac.in



SCHOOL OF COMPUTING

BONAFIDE CERTIFICATE

This is to certify that this Professional Training Report is the bonafide work of **SAYED SAJID ALI (Reg.No.37120065)** who carried out the project entitled “**DIABETES DATASET ANALYSIS AND PREDICTION**” under our supervision from July 2020 to December 2020.

Internal Guide

Dr. Jabez.J M.Tech., Ph.D.,

Head of the Department

Dr. R. SUBHASHINI M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, **SAYED SAJID ALI** (Reg.No:37120065) hereby declare that the Project Report entitled “**DIABETES DATASET ANALYSIS AND PREDICTION**” done by me under the guidance of **Dr. Jabez.J M.Tech., Ph.D.**, Dept of I.T at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in INFORMATION TECHNOLOGY.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D., Dean**, School of Computing , **Dr. R.SUBHASHINI M.E., Ph.D.**, Head of the Department of Information and Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. Jabez.J M.E., Ph.D.**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

ABSTRACT

The goal for this project is to predict (classify) Diabetes in individual women based on several medical condition and clinical tests. Diabetes is a disease in which the body's ability to produce or respond to the hormone insulin is impaired, resulting in abnormal metabolism of carbohydrates and elevated levels of glucose in the blood and urine. Glucose comes from the foods you eat. Insulin is a hormone that helps the glucose get into your cells to give them energy. Over time, having too much glucose in your blood can cause serious problems. It can damage your eyes, kidneys, and nerves. Diabetes can also cause heart disease, stroke and even the need to remove a limb. Pregnant women can also get diabetes, called gestational diabetes. Models used in this data mining project would help in predicting (classifying) Diabetes in individual women. This modelling will help medical industry to diagnose diabetes with less efforts and more reliability.

TRAINING CERTIFICATE

Certificate of Completion

Type your text

This is to certify that **Sayed Sajid Ali** successfully completed 11 total hours of **Python A-Z™: Python For Data Science With Real Exercises!** online course on April 20, 2020

Kirill Eremenko *SuperDataScience Team*
Kirill Eremenko, Instructor SuperDataScience Team, Instructor

&



Certificate no: UC-044eb172-3e0c-4ace-bb85-52781e7917d0
Certificate url: [ude.my/UC-044eb172-3e0c-4ace-bb85-52781e7917d0](https://udemy.com/UC-044eb172-3e0c-4ace-bb85-52781e7917d0)

#BeAble

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	vii
1	INTRODUCTION	1
	1.1 ABOUT THE SYSTEM	1
	1.2 SYNOPSIS	1
2	SYSTEM REQUIREMENTS	2
	2.1 TOOLS REQUIRED	2
	2.2 SOFTWARE REQUIREMENTS	3
	2.3 HARDWARE REQUIREMENTS	3
3	SYSTEM ANALYSIS	
	3.1 EXISTING SYSTEM	4
	3.2 Data Analysis and Interpretation	4
	3.2.1 Data Explanation Variables Explanation PRG	5
	3.2.2 Data Selection	5
4	IMPLEMENTATION	6
5	ALGORITHM	10
6	CONCLUSION AND FUTURE WORK	18
	REFERENCES	19
	APPENDIX	20
	A. SCREENSHOTS	14
	B. SOURCE CODE	20

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE
4.1	DATASET	14
4.2	CODING PAGE	14
4.3	ACCURACY AND VARIABLE	15
4.4	X AND Y numpy ARRAY	15
4.5	X-Test AND Y-Test	16
4.6	X-Train, Y-Train AND Y- Pred	16
4.7	COMPARE Y-Train AND Y- Pred	17

LIST OF TABLES

TABLE No	TABLE NAME	PAGE NO
2.1	HARDWARE REQUIREMENTS	03

CHAPTER 1

INTRODUCTION

1.1 ABOUT THE PRODUCT:

Diabetes mellitus is one of the major noncommunicable diseases which have great impact on human life today. Many nations are now facing a swiftly rising growth of diabetes among their residents.

According to a study by the World Health Organization (WHO), this number will have raised to 552 million by 2030, denote that one in 10 grownups will have diabetes by 2030 if no serious act is taken. In 2014, the worldwide frequency of diabetes was projected to be 9 % among adults aged 18+ years. In developing nations, most publics with diabetes are aged between 35 and 64. WHO already made an alarm that Diabetes is the 7th leading cause of death in the world in 2030. In 2012, an estimated 1.5 million deaths were straight away triggered by diabetes. In this, more than 80 % of diabetes deaths occurred in low- and middle-income countries. Total deaths from diabetes are projected to rise by more than 50 % in the next 10 years.

It is apparent out that diabetes is a foremost cause of blindness, amputation and kidney failure. Lack of alertness about diabetes, combined with inadequate access to health services and vital medicines, can lead to many hitches. It is a universal problem with overwhelming human, social, and economic impact, affecting around 300 million people worldwide.

By applying computational analytics on clinical big data, the massive amount of data generated in the healthcare systems, will be used to create medical intelligence which will drive medical prediction and forecasting. Medical analysis is a new trend in medical science. Developing medical intelligence out of the clinical data available will create healthcare system to be patient-centred and will reduce medical cost and hospital readmission too.

1.2 SYNOPSIS:

This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective is to predict based on diagnostic measurements whether a patient has diabetes.

Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage.

- Pregnancies: Number of times pregnant
- Glucose: Plasma glucose concentration a 2 hours in an oral glucose tolerance test
- Blood Pressure: Diastolic blood pressure (mm Hg)
- Skin Thickness: Triceps skin fold thickness (mm)
- Insulin: 2-Hour serum insulin (mu U/ml)
- BMI: Body mass index (weight in kg/(height in m)²)

- Diabetes Pedigree Function: Diabetes pedigree function
- Age: Age (years)
- Outcome: Class variable (0 or 1)

CHAPTER 2

SYSTEM REQUIREMENTS

2.1 DATA REQUIREMENTS:

The data are necessary as inputs to the analysis, which is specified based upon the requirements of those directing the analysis or customers (who will use the finished product of the analysis). The general type of entity upon which the data will be collected is referred to as an experimental unit (e.g., a person or population of people). Specific variables regarding a population (e.g., age and income) may be specified and obtained. Data may be numerical or categorical (i.e., a text label for numbers).^[5]

Accessories:

- **Documentation Packages**

MS Word 2010

- **Operating System**

Any Operating System or windows
Family ahead of Windows 98, for
Desktop applications and Windows 98
Networking or Win NT/2K for network
Base multi-user system.

2.2 SOFTWARE REQUIREMENTS:

- **Operating System** Windows 7 Onwards.
- **Database** Any Database.

2.3 HARDWARE REQUIREMENTS:

A brief description of the hardware requirement to implement the “PCO Management System” is given below:

Table 2.1: Hardware Requirements

Sl. No.	Feature	Minimum	Recommended
01	Processor	P II	P III
02	RAM	2 GB RAM	4 GB RAM
03	Hard Disk Size	4.3 GB	40 GB for Server
04	Monitor (Color/SVGA)	Any Standard Monitor	Any Standard Monitor
05	Operating System	Win-7,8,10 Networking with Office 2010	Win-NT / 2K Server
06	Network/ Multi-user	Win-7,8,10 Networking	Win-NT/2K Server
07	System Printer	Any Good Printer	Any Good Printer
08	Network Interface Card	NE-2000 Ethernet	Card or Similar
09	Scanner	TWAIN compatible OCR software	Flat-bed Scanner

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Diabetes is a group of metabolic diseases in which there are high blood sugar levels over a prolonged period. Symptoms of high blood sugar include frequent urination, increased thirst, and increased hunger. To study the reason that leading to diabetes, a cluster of dataset about Pima Indian Diabetes was collected. It is consisted of 8 predict variables and 1 response variable. The variables are PRG, PLASMA, BP, THICK, INSULIN, BODY, PEDIGREE, AGE. After randomly selecting 700 observations from 768 patients, 9 variables were taken to fit a generalize linear model to predict the probability that individual females have diabetes. Then, using stepwise selection provided subgroups of characteristics with higher risk of diabetes.

3.2 Data Analysis and Interpretation:

3.2.1 Data Explanation Variables Explanation PRG:

Number of times pregnant PLASMA: Plasma glucose concentration in saliva BP: Diastolic blood pressure THICK: Triceps skin fold thickness INSULIN: Two Hours serum insulin BODY: Body mass index (Weight/Height) PEDIGREE: Diabetes pedigree function AGE: In years RESPONSE: 1: Diabetes, 0: Not For more details, please check Table 1 in Appendix 1.

3.2.2 Data Selection :

Randomly picking up 700 total observations from 768 patients was finished in R. The new dataset was named after PID. The R Code to do this was attached as Table 2 in Appendix 1.

3.2.3 Predicting

The probability to suffer from diabetes for individual females Generalize linear model

contributed to solve this problem. In the glm, the binomial family was chosen, since the response variable was only consisted of 1 and 0. The generated model is: $REP = -0.8532997 + PRG*0.0211334 + PLASMA*0.0059682 - BP*0.0021338 + THICK*0.0005073 - INSULIN*0.0002122 + BODY*0.0129676 + PEDIGREE*0.1361092 + AGE*0.0024004$ Here, doing Cross-Validation checked Generalized Linear Model. The MSE (mean square of error) of this model was 3.322. What's more, according to the Table 3, 5 variables (PRG, PLASMA, BP, BODY and PEDIGREE) had an important influence on REP, since their P-Value of coefficient < 0.05. Hence, the adjusted model was, $REP = -0.853 + PRG*0.0211 + PLASMA*0.00597 - BP*0.00213 + BODY*0.0130 + PEDIGREE*0.136$ The next, doing Cross-Validation checked Generalized Linear Model. The MSE (mean square of error) of this model was 2.952. The MSE of adjusted model, 2.952, is less than that of whole model, 3.322. Finally, the chosen model to predict the probability was: $REP = -0.853 + PRG*0.0211 + PLASMA*0.00597 - BP*0.00213 + BODY*0.0130 + PEDIGREE*0.136$ The value of REP was taken as index to return the probability of suffering from diabetes for individual females. When the value of REP is close to 1, it means the individual female has higher probability to have diabetes. When the value of REP is close to 0, it means the individual female has a healthy physical body. The details were attached in Table 3.

3.2.4 Detecting subgroups of characteristics that are at higher risk of diabetes. For the subgroup of characteristics, forward and backward stepwise selection (in AIC method) were used. AIC (Akaike information criterion): a measure of the relative quality of statistical models for a given set of data. Given a collection of models for the data, AIC estimates the quality of each model, relative to each of the other models. Hence, AIC provides a means for model selection. For forward stepwise selection, the result with the least AIC = 688.06 was $REP = -8.0308 + 0.1481*PRG + 0.0356*PLASMA + 0.0861*BODY + 0.9016*PEDIGREE - 0.0107*BP - 0.00132*INSULIN$ The details result was shown in Table 4. In the result, the P-value of INSULIN = 0.11091 > 0.05. Let the null hypothesis be the coefficient of INSULIN = 0, so the null hypothesis cannot be rejected since 0.11091 > 0.05. Hence the coefficient of INSULIN = 0. The final model was $REP = -8.0308 + 0.148*PRG + 0.0356*PLASMA + 0.0861*BODY + 0.9016*PEDIGREE - 0.0107*BP$ The MSE (mean square of error) of forward stepwise selection was 3.21068. For backward stepwise selection, the result with the least AIC = 727.67 was $REP = -8.0308 + 0.1481*PRG + 0.0356*PLASMA + 0.0861*BODY + 0.9016*PEDIGREE - 0.0107*BP - 0.00132*INSULIN$

The details result was shown in Table 5. In the result, the P-value of INSULIN = 0.11091 > 0.05. Let the null hypothesis be the coefficient of INSULIN = 0, so the null hypothesis cannot be rejected since 0.11091 > 0.05. Hence the coefficient of INSULIN = 0. The final model was $REP = -8.0308 + 0.148*PRG + 0.0356*PLASMA + 0.0861*BODY + 0.9016*PEDIGREE - 0.0107*BP$ The MSE (mean square of error) of forward stepwise selection was 3.21068. The forward stepwise selection method shared the same answer with backward stepwise selection method. And the coefficient of BP < 0, so BP will not be considered as dangerous variable to cause diabetes among the given variables. The conclusion was drawn that subgroups of characteristics that are at higher risk of diabetes are, PRG, PLASMA, BODY and PEDIGREE.

CHAPTER 4

IMPLEMENTATION

Let's divide the classification problem into below steps:

- Prerequisite and setting up the environment.
- Loading the data set in Spyder Notebook.
- Understanding the data.
- Data preprocessing.
- Running machine learning algorithm.

Step 1:

Prerequisite and setting up the environment

The prerequisites to follow this example are python version (3.6.4) and jupyter notebook. You can just install [anaconda](#) and it will get everything for you. Also, little bit of python and ML basics including text classification is required. We will be using scikit-learn (python) libraries for our example.

Step 2:

Loading the data set in Spyder notebook

After downloading the dataset from kaggle, I saved it to my working directory with the name mydata.csv. next I used `read_csv()` to read the dataset and save it to dataset variable. Before analysis, I just wanted to take a look at the data, so I used the `info()` method. I found there are no missing values so we don't need to take care of any null values. Next I used `describe()` method. The method revealed that the range of each variable is different. The maximum value of age is 77 but for chol it is 564.

Step 3:

Understanding the dataset

Correlation Matrix

To begin with, let's see the correlation matrix of features and try to analyse it. The figure size is defined to 12 x 8 by using rcParams. Then, I used pyplot to show the correlation matrix. Using xticks and yticks, I've added names to the correlation matrix. colorbar() shows the colorbar for the matrix.

It's easy to see that there is no single feature that has a very high correlation with our target value. Also, some of the features have a negative correlation with the target value and some have positive.

Histogram

The best part about this type of plot is that it just takes a single command to draw the plots and it provides so much information in return. Just use dataset.hist().

Bar Plot for Target Class

It's really essential that the dataset we are working on should be approximately balanced. An extremely imbalanced dataset can render the whole model training useless and thus, will be of no use. Let's understand it with an example.

Step 4.

Data processing

To work with categorical variables, we should break each categorical column into dummy columns with 1s and 0s.

Let's say we have a column Gender, with values 1 for Male and 0 for Female. It needs to be converted into two columns with the value 1 where the column would be true and 0 where it will be false. Take a look at the Gist below.

To get this done, we use the get_dummies() method from pandas. Next, we need to scale the dataset for which we will use the Standard Scaler. The fit transform() method of the scaler scales the data and we update the columns.

The dataset is now ready. We can begin with training our models.

Step 5.

Running machine learning algorithms

In this project, I took 4 algorithms and varied their various parameters and compared the final models. I split the dataset into 75% training data and 25% testing data.

Naive Bayes

There are various algorithms which can be used for text classification. We will start with the most simplest one 'Naive Bayes (NB).

You can easily build a NBclassifier in scikit using below 2 lines of code: (note - there are many variants of NB, but discussion about them is out of scope)

The accuracy we get is ~80%, which is not bad for start and for a naive classifier. Also, congrats!!! you have now written successfully a classification algorithm .

CHAPTER 5

Algorithms

What is the Naive Bayes Classifier?

The Naive Bayes classifier is a simple probabilistic classifier which is based on Bayes theorem with strong and naïve independence assumptions. It is one of the most basic text classification techniques with various applications in email spam detection, personal email sorting, document categorization, sexually explicit content detection, language detection and sentiment detection. Despite the naïve design and oversimplified assumptions that this technique uses, Naive Bayes performs well in many complex real-world problems.

Even though it is often outperformed by other techniques such as boosted trees, random forests, Max Entropy, Support Vector Machines etc, Naive Bayes classifier is very efficient since it is less computationally intensive (in both CPU and memory) and it requires a small amount of training data. Moreover, the training time with Naive Bayes is significantly smaller as opposed to alternative methods.

Naive Bayes classifier is superior in terms of CPU and memory consumption as shown by Huang, J. (2003), and in several cases its performance is very close to more complicated and slower techniques.

Which Naive Bayes Variation to use?

There are several Naive Bayes Variations. Here we will discuss about 3 of them: the Multinomial Naive Bayes, the Binarized Multinomial Naive Bayes and the Bernoulli Naive Bayes. Note that each can deliver completely different results since they use completely different models.

Usually Multinomial Naive Bayes is used when the multiple occurrences of the words matter a lot in the classification problem. Such an example is when we try to perform Topic Classification. The Binarized Multinomial Naive Bayes is used when the frequencies of the words don't play a key role in our classification. Such an example is Sentiment Analysis, where it does not really matter how many times someone mentions the word "bad" but rather only the fact that he does. Finally the Bernoulli Naive Bayes can be used when in our problem the absence of a particular word matters. For example Bernoulli is commonly used in Spam or Adult Content Detection with very good results.

The Theoretical Background of Naive Bayes

As stated earlier, the Naive Bayes classifier assumes that the features used in the classification are independent. Despite the fact that this assumption is usually false, analysis of the Bayesian classification problem has shown that there are some theoretical reasons for the apparently unreasonable efficacy of Naive Bayes classifiers as [Zhang \(2004\)](#) shown. It can be proven ([Manning et al, 2008](#)) that even though the probability estimates of Naive Bayes are of low quality, its classification decisions are quite good. Thus, despite the fact that Naive Bayes usually over estimates the probability of the selected class, given that we use it only to make the decision and not to accurately predict the actual probabilities, the decision making is correct and thus the model is accurate.

In a text classification problem, we will use the words (or terms/tokens) of the document in order to classify it on the appropriate class. By using the “maximum a posteriori (MAP)” decision rule, we come up with the following classifier:

$$c_{map} = \arg \max_{c \in C} (P(c | d)) = \arg \max_{c \in C} \left(P(c) \prod_{1 \leq k \leq n_d} P(t_k | c) \right)$$

Where t_k are the tokens (terms/words) of the document, C is the set of classes that is used in the classification, $P(c | d)$ the conditional probability of class c given document d , $P(c)$ the prior probability of class c and $P(t_k | c)$ the conditional probability of token t_k given class c .

This means that in order to find in which class we should classify a new document, we must estimate the product of the probability of each word of the document given a particular class (likelihood), multiplied by the probability of the particular class (prior). After calculating the above for all the classes of set C , we select the one with the highest probability.

Due to the fact that computers can handle numbers with specific decimal point accuracy, calculating the product of the above probabilities will lead to float point underflow. This means that we will end up with a number so small, that will not be able to fit in memory and thus it will be rounded to zero, rendering our analysis useless. To avoid this instead of maximizing the product of the probabilities we will maximize the sum of their logarithms:

$$c_{map} = \arg \max_{c \in C} \left(\log P(c) + \sum_{1 \leq k \leq n_d} \log P(t_k | c) \right)$$

Thus instead of choosing the class with the highest probability we choose the one with the highest log score. Given that the logarithm function is monotonic, the decision of MAP remains the same.

The last problem that we should address is that if a particular feature/word does not appear in a particular class, then its conditional probability is equal to 0. If we use the first decision method (product of probabilities) the product becomes 0, while if we use the second (sum

of their logarithms) the $\log(0)$ is undefined. To avoid this, we will use add-one or Laplace smoothing by adding 1 to each count:

$$P(t|c) = \frac{T_{ct} + 1}{\sum_{t \in V} (T_{ct'} + 1)} = \frac{T_{ct} + 1}{\sum_{t \in V} (T_{ct'}) + B'}$$

Where B' is equal to the number of the terms contained in the vocabulary V .

Naive Bayes Variations

Let's examine three common Naive Bayes variations which differ on the way that they calculate the conditional probabilities of each feature and on the scoring of each category.

Multinomial Naive Bayes model

This variation, as described by [Manning et al \(2008\)](#), estimates the conditional probability of a particular word/term/token given a class as the relative frequency of term t in documents belonging to class c :

$$P(t|c) = \frac{T_{ct}}{\sum_{t \in V} T_{ct'}}$$

Thus this variation takes into account the number of occurrences of term t in training documents from class c , including multiple occurrences.

Both the training and the testing algorithms are presented below in the form of pseudo code:

```

TRAINMULTINOMIALNB( $\mathbb{C}, \mathbb{D}$ )
1   $V \leftarrow \text{EXTRACTVOCABULARY}(\mathbb{D})$ 
2   $N \leftarrow \text{COUNTDOCS}(\mathbb{D})$ 
3  for each  $c \in \mathbb{C}$ 
4  do  $N_c \leftarrow \text{COUNTDOCSINCLASS}(\mathbb{D}, c)$ 
5      $\text{prior}[c] \leftarrow N_c / N$ 
6      $\text{text}_c \leftarrow \text{CONCATENATETEXTOFALLDOCSINCLASS}(\mathbb{D}, c)$ 
7     for each  $t \in V$ 
8     do  $T_{ct} \leftarrow \text{COUNTTOKENSOFTERM}(\text{text}_c, t)$ 
9     for each  $t \in V$ 
10    do  $\text{condprob}[t][c] \leftarrow \frac{T_{ct}+1}{\sum_{c'} (T_{ct'}+1)}$ 
11 return  $V, \text{prior}, \text{condprob}$ 

```

```

APPLYMULTINOMIALNB( $\mathbb{C}, V, \text{prior}, \text{condprob}, d$ )
1   $W \leftarrow \text{EXTRACTTOKENSFROMDOC}(V, d)$ 
2  for each  $c \in \mathbb{C}$ 
3  do  $\text{score}[c] \leftarrow \log \text{prior}[c]$ 
4     for each  $t \in W$ 
5     do  $\text{score}[c] += \log \text{condprob}[t][c]$ 
6 return  $\arg \max_{c \in \mathbb{C}} \text{score}[c]$ 

```

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunc	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						
28	7	147	76	0	0	39.4	0.257	43	1						
29	1	97	66	15	140	23.2	0.487	22	0						
30	13	145	82	19	110	22.2	0.245	57	0						
31	5	117	92	0	0	34.1	0.337	38	0						
32	5	100	75	26	0	36	0.546	60	0						

Fig 4.1: Dataset :

```

1 import numpy as np
2 import matplotlib.pyplot as plt
3 import pandas as pd
4
5 #importing the dataset
6 dataset=pd.read_csv('mydata.csv')
7 X=dataset.iloc[:, :-1].values
8
9 Y=dataset.iloc[:, 8].values
10
11
12 #Encoding categorical Data (this is only when you have categorical attributes)
13 """from sklearn.preprocessing import LabelEncoder, OneHotEncoder
14 lb_x=LabelEncoder()
15 X[:,1]=lb_x.fit_transform(X[:,1])
16 X[:,2]=lb_x.fit_transform(X[:,2])
17 X[:,5]=lb_x.fit_transform(X[:,5])
18 OneHotEncoder=OneHotEncoder(categorical_features=[1])
19 OneHotEncoder=OneHotEncoder(categorical_features=[2])
20 OneHotEncoder=OneHotEncoder(categorical_features=[5])
21 X=OneHotEncoder.fit_transform(X).toarray()"""
22
23
24
25 #splitting the dataset into training and testing set
26 from sklearn.cross_validation import train_test_split
27 X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.25,random_state=0)
28
29 #feature scaling/normalization/transformation
30 """from sklearn.preprocessing import StandardScaler
31 sc_X=StandardScaler()
32 X_train=sc_X.fit_transform(X_train)
33 X_test=sc_X.fit_transform(X_test)"""
34
35 #fitting naive bayes model
36 from sklearn.naive_bayes import GaussianNB
37 classifier=GaussianNB()
38 classifier.fit(X_train,Y_train)
39
40 #prediction
41 Y_pred=classifier.predict(X_test)
42
43 #making the confusion matrix

```

Fig 4.2: Coding page:

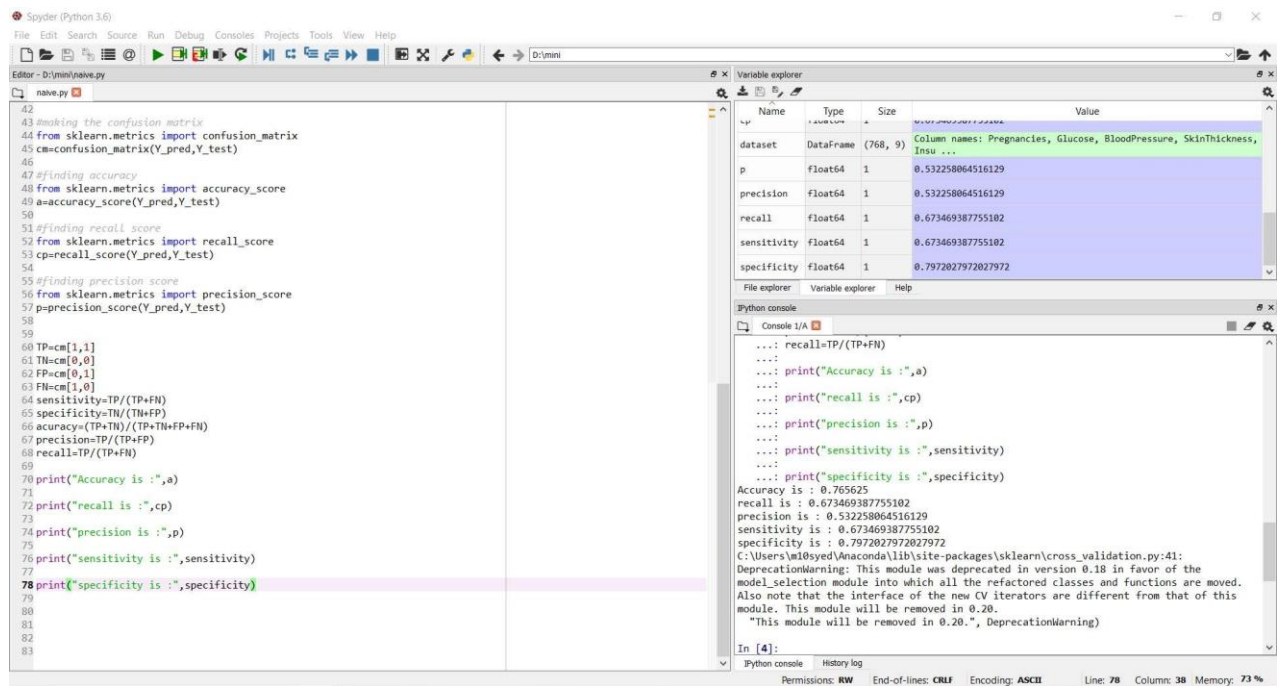


Fig 4.3: Accuracy and Variable Exploration:

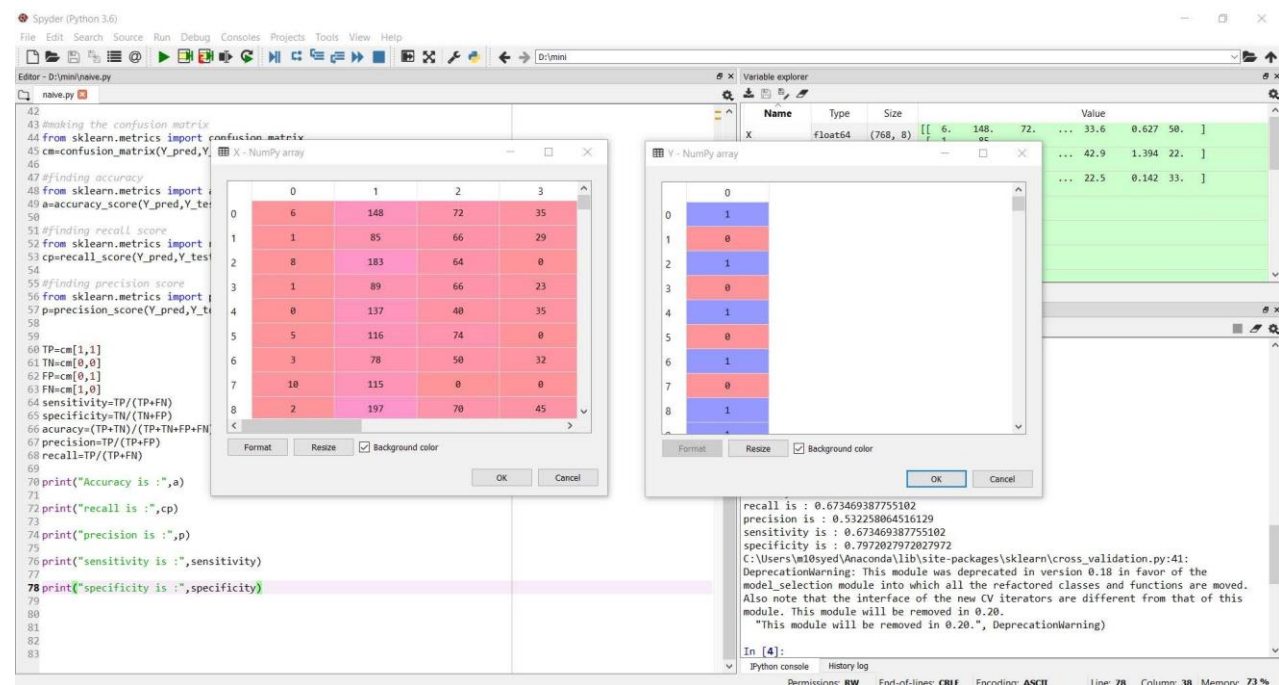


Fig 4.4: X and Y numpy array:

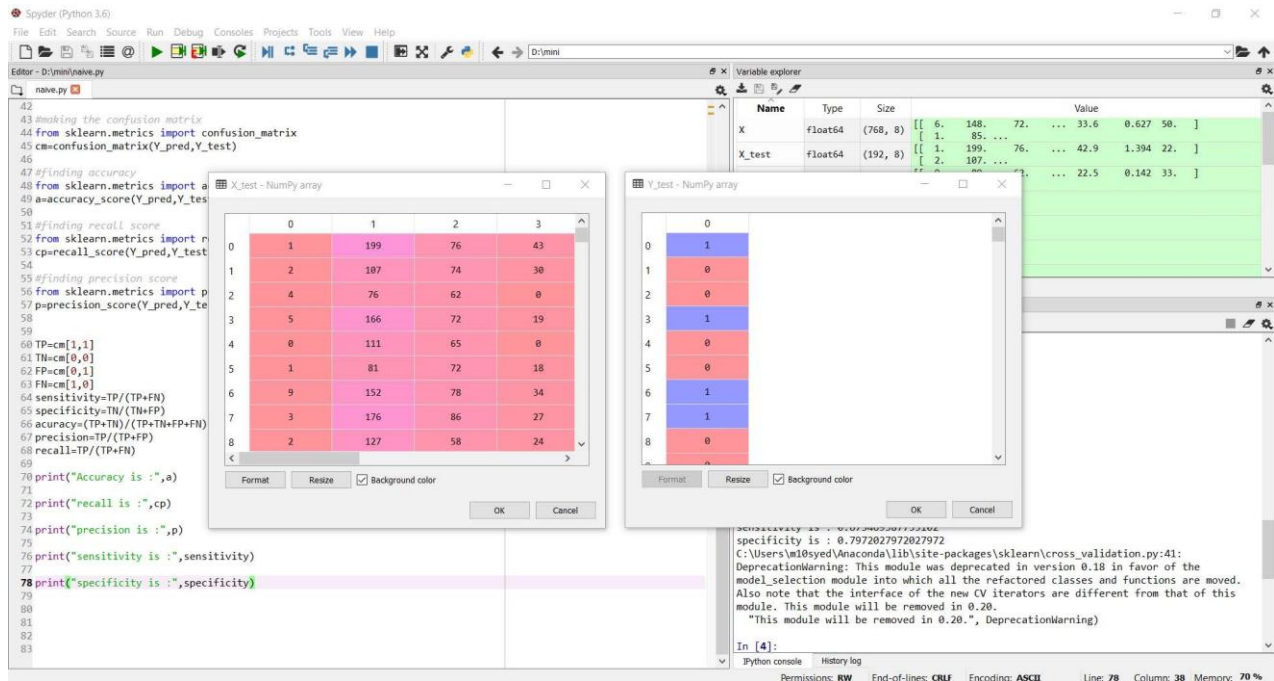


Fig 4.5:X-Test and Y-Test:

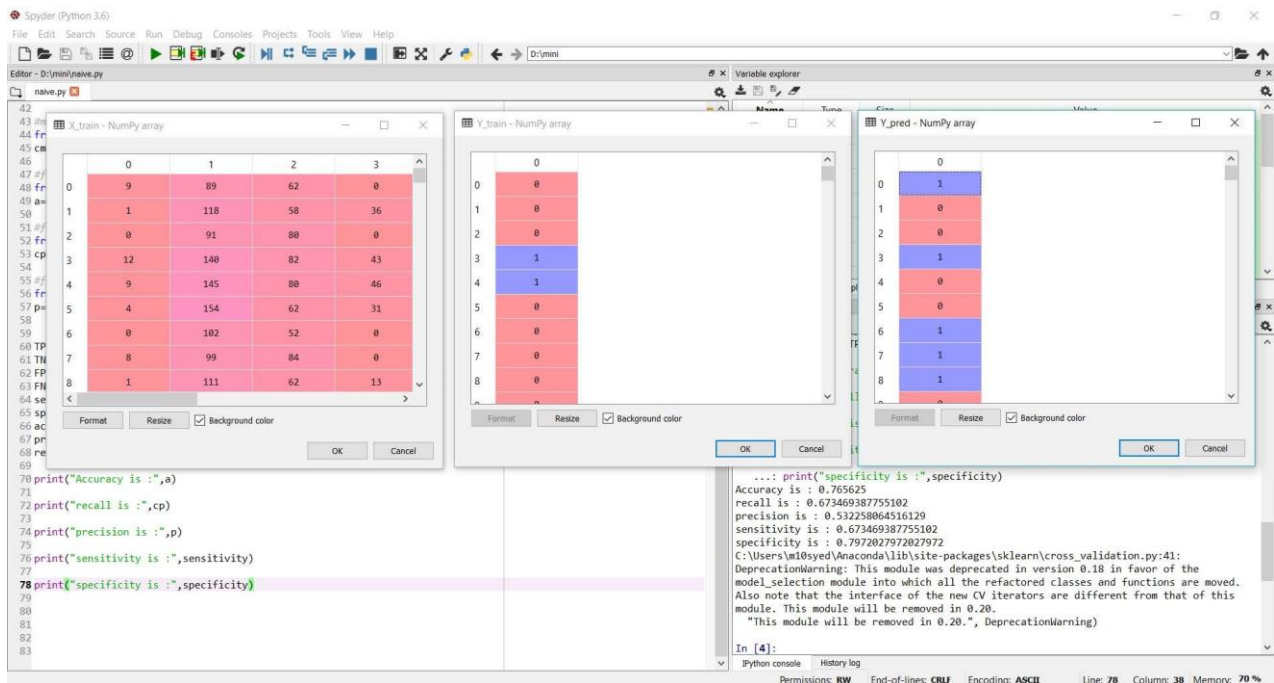


Fig 4.6:X-Train,Y-Train and Y-Prediction:

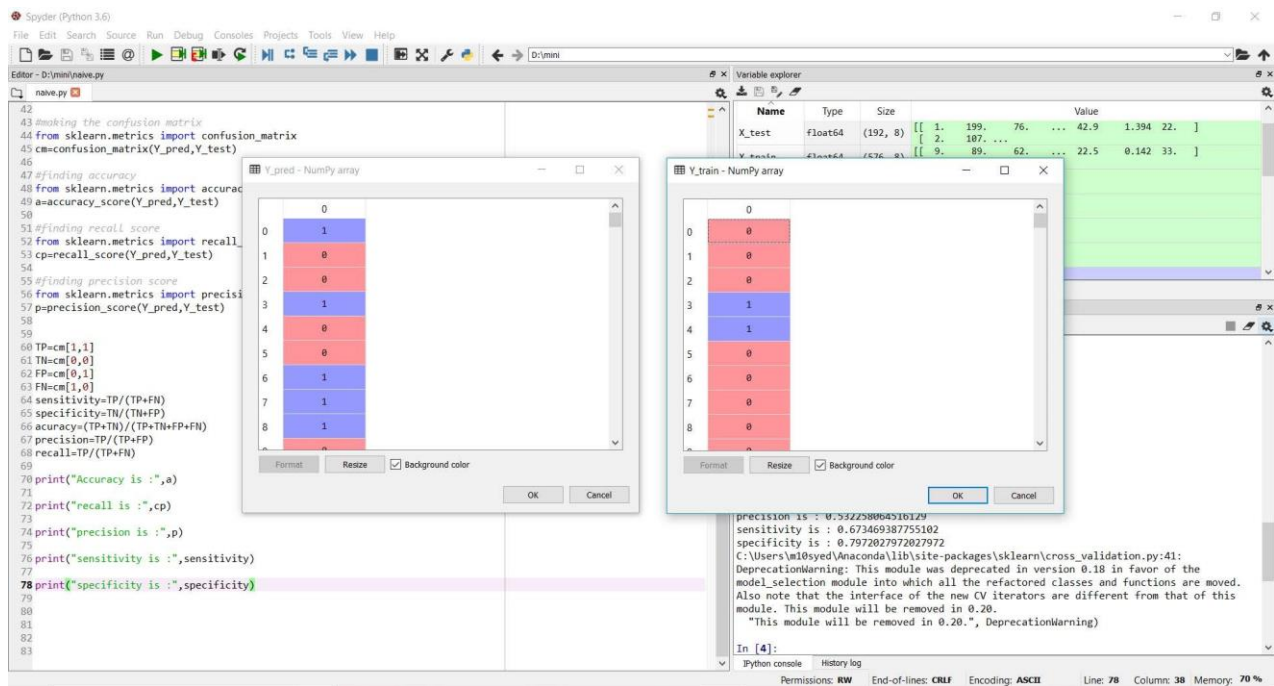


Fig 4.7: Comparing the accuracy of Y-Pred and Y-Train:

CHAPTER 7

CONCLUSION AND FUTURE WORK

In this report, initially, the dataset, 700 total observations, was randomly selected via the help of R. Then, according to the picked dataset, the probability that individual females have diabetes was predicted. The result was $REP = -0.8533 + PRG*0.0211 + PLASMA*0.00597 - BP*0.00213 + BODY*0.01297 + PEDIGREE*0.1361$ When the value of REP is close to 1, it means the individual female has higher probability to have diabetes. When the value of REP is close to 0, it means the individual female has a healthy physical body. The lower the REP is, the healthier the individual female is. The mean square of error is 2.952. The next, the higher risk subgroups of characteristics that lead to diabetes were gotten. The unmodified model is PRG, PLASMA, BP, BODY and PEDIGREE. According to their coefficient, we get that PRG (Number of times pregnant), PLASMA (Plasma glucose concentration in saliva), BODY (Body mass index) and PEDIGREE (Diabetes pedigree function) have positive influence on having diabetes. This means that the larger those variables are, the higher probability the individual female has diabetes. The mean square of error is 3.21068. Hence, the final answer is 4 variables PRG, PLASMA, BODY and PEDIGREE. Some suggestion to prevent diabetes are diet control and doing exercise. Diet control and doing exercise can help people lower their weight. If weight is reduced, so is Body mass index, finally the REP decreases.

REFERENCES

1. The reference for the project has been taken from
 - Sathyabama institute of science and technology
 - Kaggle.com
 - Udemy
 - YouTube

APPENDIX

A. SOURCE CODE

1. SOURCE CODE (KNN):

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

#importing the dataset

```
dataset=pd.read_csv('mydata.csv')
X=dataset.iloc[:, :-1].values
Y=dataset.iloc[:, 8].values
```

#Encoding categorical Data (this is only when you have categorical attributes)

```
'''from sklearn.preprocessing import LabelEncoder, OneHotEncoder
lb_x=LabelEncoder()
X[:,1]=lb_x.fit_transform(X[:,1])
X[:,2]=lb_x.fit_transform(X[:,2])
X[:,5]=lb_x.fit_transform(X[:,5])
OneHotencoder=OneHotEncoder(categorical_features=[1])
OneHotencoder=OneHotEncoder(categorical_features=[2])
OneHotencoder=OneHotEncoder(categorical_features=[5])
X=OneHotencoder.fit_transform(X).toarray()'''
```

#splitting the dataset into training and testing set

```
from sklearn.cross_validation import train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.25,random
_state=0)
```

#feature scaling/normlization/transformation

```
"""from sklearn.preprocessing import StandardScaler
sc_X=StandardScaler()
X_train=sc_X.fit_transform(X_train)
X_test=sc_X.fit_transform(X_test)"""
```

#fitting KNN model

```
from sklearn.neighbors import KNeighborsClassifier
classifier=KNeighborsClassifier()
classifier.fit(X_train,Y_train)
```

#prediction

```
Y_pred=classifier.predict(X_test)
```

#making the confusion matrix

```
from sklearn.metrics import confusion_matrix
cm=confusion_matrix(Y_pred,Y_test)
```

#finding accuracy

```
from sklearn.metrics import accuracy_score
a=accuracy_score(Y_pred,Y_test)
```

#finding recall score

```
from sklearn.metrics import recall_score  
cp=recall_score(Y_pred,Y_test)
```

#finding precision score

```
from sklearn.metrics import precision_score  
p=precision_score(Y_pred,Y_test)
```

```
TP=cm[1,1]  
TN=cm[0,0]  
FP=cm[0,1]  
FN=cm[1,0]  
sensitivity=TP/(TP+FN)  
specificity=TN/(TN+FP)  
accuracy=(TP+TN)/(TP+TN+FP+FN)  
precision=TP/(TP+FP)  
recall=TP/(TP+FN)
```

```
print("Accuracy is :",a)  
print("recall is :",cp)  
print("precision is :",p)  
print("sensitivity is :",sensitivity)  
print("specificity is :",specificity)
```

2.SOURCE CODE(NAÏVE BAYES)

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

#importing the dataset

```
dataset=pd.read_csv('mydata.csv')
X=dataset.iloc[:, :-1].values
Y=dataset.iloc[:, 8].values
```

#Encoding categorical Data (this is only when you you categorical attributes)

```
"""from sklearn.preprocessing import LabelEncoder, OneHotEncoder
lb_x=LabelEncoder()
X[:,1]=lb_x.fit_transform(X[:,1])
X[:,2]=lb_x.fit_transform(X[:,2])
X[:,5]=lb_x.fit_transform(X[:,5])
OneHotencoder=OneHotEncoder(categorical_features=[1])
OneHotencoder=OneHotEncoder(categorical_features=[2])
OneHotencoder=OneHotEncoder(categorical_features=[5])
X=OneHotencoder.fit_transform(X).toarray()"""
```

#splitting the dataset into training and testing set

```
from sklearn.cross_validation import train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.25,random
_state=0)
```

#feature scaling/normlization/transformation

```
from sklearn.preprocessing import StandardScaler  
sc_X=StandardScaler()  
X_train=sc_X.fit_transform(X_train)  
X_test=sc_X.fit_transform(X_test)"
```

#fitting naive bayes model

```
from sklearn.naive_bayes import GaussianNB  
classifier=GaussianNB()  
classifier.fit(X_train,Y_train)
```

#prediction

```
Y_pred=classifier.predict(X_test)
```

#making the confusion matrix

```
from sklearn.metrics import confusion_matrix  
cm=confusion_matrix(Y_pred,Y_test)
```

#finding accuracy

```
from sklearn.metrics import accuracy_score  
a=accuracy_score(Y_pred,Y_test)
```

#finding recall score

```
from sklearn.metrics import recall_score cp=recall_score(Y_pred,Y_test)
```

#finding precision score

```
from sklearn.metrics import precision_score
```

```
p=precision_score(Y_pred,Y_test)
TP=cm[1,1]
TN=cm[0,0]
FP=cm[0,1]
FN=cm[1,0]
sensitivity=TP/(TP+FN)
specificity=TN/(TN+FP)
accuracy=(TP+TN)/(TP+TN+FP+FN)
precision=TP/(TP+FP)
recall=TP/(TP+FN)
print("Accuracy is:",a)
print("recall is :",cp)
print("precision is :",p)
print("sensitivity is :",sensitivity)
print("specificity is :",specificity)
```