DIABETES PREDICTION USING MACHINE LEARNING ALGORITHM

BY

Dhanesh Ganesan

A DISSERTATION

Submitted to

The University of Liverpool

**in partial fulfillment of the requirements
for the degree of**

MASTER OF SCIENCE

07/10/2022

DIABETES PREDICTION USING MACHINE LEARNING ALGORITHM

BY
Dhanesh Ganesan

ABSTRACT

An interdisciplinary branch of study with roots in databases, statistics, machine learning, and data science for diabetic prediction, information gathering for healthcare is beneficial in assessing the efficacy of medical therapies. Even if there are various data science categorization approaches, machine learning is used to demonstrate the outcome in diagnosing diabetic disease. Diabetes is a long-term (chronic) illness affecting how your body converts food into energy. Most of the food you consume is converted to sugar, commonly known as glucose, and then discharged into your bloodstream. When the blood glucose volume increases, the pancreas releases insulin. Nowadays, machine learning is used in the public health system, and there is a possibility of early disease prediction. Data is the primary requirement for artificial intelligence. First, the historical dataset is gathered, and a computer vision model is created using that dataset. Next, the required pre-processing methods, such as univariate and bivariate analysis, are implemented. After visualizing the data, a categorization model is developed using machine learning to help with feature interpretation. Finally, algorithms are compared based on performance criteria, including accuracy, F1 score recall, etc.

- The project's primary goal is to employ real-time technology to forecast diabetes disease utilizing data science techniques built on a system's machine learning model.
- The aim is to create a machine-learning model to predict diabetes. Eventually, replace the supervised learning classification models that can be updated by expecting results with the highest degree of accuracy by contrasting supervised algorithms.
- Comparing different algorithms with Metrics such as accuracy, precision, recall, etc., can be used to verify the model's accuracy. The next step is to raise the organization (Hospitals) to new heights depending on the system's proposed method's accuracy.

DECLARATION

I hereby certify that this dissertation constitutes my own product, that where the language of others is set forth, quotation marks so indicate, and that appropriate credit is given where I have used the language, ideas, expressions, or writings of another.

I declare that the dissertation describes original work that has not previously been presented for the award of any other degree of any institution.

Signed,

Date

October 7, 2022

ACKNOWLEDGEMENTS

Thank Mr. Louwe Kuijer and Ullrich Hustadt for guiding me to his important publications and stimulating questions on artificial intelligence and automation. The discussions were essential in getting me to look at things from different angles and develop a well-rounded model.

TABLE OF CONTENT

Page

Chapter 1. INTRODUCTION

Neurons are the fundamental unit of the brain and nervous system. These cells are responsible for receiving electrical input and responding. In addition, neurons communicate with each other with synapses. Neural networks (NNs) are computer systems inspired by the biological brain. An artificial neural network (ANN) is constructed of artificial neurons, a group of interconnected units or nodes that resemble neurons in the human brain. Each link can communicate with neighboring neurons, like human synapses. An artificial neuron can communicate with interconnected neurons to it after processing signals that are sent to it. In a non-linear function, the "signal" at a connection is an actual number, and the inputs of a neuron influence its output. Edges are the terms for connections. As learning progresses, edges and neurons frequently change size. The weight modifies a connection's signal intensity by increasing or decreasing it. A threshold may exist in neurons that must be crossed before sending a signal. Neurons commonly form layers by grouping them. Various layers may alter their inputs in various ways. From the input layer at the top to the final layer, signals pass through the layers, maybe more than once (the output layer). Training data helps neural networks improve over time. Once adjusted for accuracy, these learning methods become helpful in computer science and AI for fast classifying and clustering data.

Artificial intelligence (AI) is the simulation of human intelligence in programmed machines to think and act like a human being. The word can also describe any mechanical system that demonstrates cognitive abilities typically associated with learning and problem-solving. Search engines, suggestion systems (used by Google, Amazon, YouTube, and Netflix), speech recognizing (such as google or Siri), and self-driving cars (like Tesla), AI is used in high-level strategic game systems (such as chess and Go) All of these are examples of AI applications.

1.1 Aims and Objectives

This project's scope is to investigate a diabetes dataset to find diabetes using machine learning techniques. Finding diabetes is based on people's actual health conditions. Therefore, it's not easy to predict, but we can build predictive models using the AI machine learning technique compared with different algorithms.

The systems will help reduce of cost of patient management by avoiding unnecessary investigations and patient follow-ups. These prediction systems will add accuracy and time management. In addition, computer-based patient support systems benefit patients by providing informational support that increases their participation in health care.

The model can be used by endocrinologists, dietitians, ophthalmologists, and podiatrists to predict if or if not the patient is likely to suffer from diabetes, if yes, how intense it could be.

1.2 Problem Statement

This project's primary objective is to determine whether a person is diabetic positive or negative.

1.3 Approach

To prepare the model. Data is an essential part of the model. We use machine learning algorithms to find the accuracy

1.4 Outcome

To calculate the model's performance, I'm using various metrics like Accuracy, Specificity, Sensitivity, Recall, and Precision and visualizing it with a Confusion matrix.

CHAPTER 2. BACKGROUND AND REVIEW OF THE LITERATURE

To develop hybrid treatment models that clinicians might utilize on diabetic patients. By considering the medical history, the Naïve Bayes and Randomized Forest algorithms are used to determine if a person has diabetes or not. As a result, this approach makes it simple for doctors to organize, classify, and categorize various disease types to make appropriate treatments. The data set is used as an input, and the NB Model is used to make the forecast. Here, feature selection is carried out using the Random Forest technique. It creates many uncorrelated decision trees using n inputs from the dataset during training. The class that represents the mode of each class output is then displayed individually (Geetha* and Prasad, 2020)

One of the leading causes of stroke, kidney failure, heart failure, amputations, blindness, and kidney failure is diabetes. Our bodies transform food into glucose when we eat., such as glucose. Our pancreas is then expected to release insulin. Insulin acts as a "key" to unlock our cells and let glucose in, allowing us to have glucose as fuel. However, this mechanism does not function in diabetes. The most prevalent forms of the disease are type 1 and type 2, but there are other varieties, including gestational diabetes, which develops during pregnancy. The recent advances in machine learning that have substantially impacted the diagnosis and classification of diabetes are the main topic of this presentation. (Indoria and Rathore, 2018)

Machine learning algorithms are applied to the dataset and classified using various algorithms. Two data sets have been used. Type 1 (Indian Pima data set) where the random forest gives 72% accuracy, Svm gives 68 % accuracy, decision tree retained 74%, and logistic regression has 76%of accuracy. But after switching the data set (type2) and applying the pipeline, we would obtain an average of 96 % accuracy. Using this dataset instead of the previous dataset allows the algorithm to predict diabetes more accurately and precisely. (Mujumdar and Vaidehi, n.d.)

On a dataset obtained from the 44 Army Reference Hospital and the Yusuf Danstoho Memorial Medical Center Kaduna, which consists of nine attribute data, the method was created using a supervised learning algorithm, such as the Decision Tree algorithm, the K-Nearest Neighbor algorithm, and Neural Networks. The study was conducted on the incidence of diabetes among the people of Kaduna state using some preferred hospitals as a case study. According to the results, ANN provided the highest accuracy, at 97.40 percent, trailed by decision tree-based accuracy, at 96.10 percent, and K-NN algorithm accuracy, at 88.31%. (Evwiekpaefe and Abdulkadir, 2021)

High blood sugar levels are the most common symptom of diabetes. Type 2 diabetes or insulin sensitivity in periphery tissue cells may be the reason for uncontrolled diabetes. If diabetes is mistreated and undiagnosed, several issues arise. It is also the creator of several problems, such as coronary failure, eyesight, urinary organ diseases, etc. This study uses nine distinct machine-learning approaches to predict diabetes. Multiple machine-learning techniques are used to analyze a dataset of diabetes patients. The utilized algorithms, disease occurrence, Specificity, Precision, Recall, F1-Score, True Positive Rate, and false-positive rate are described and compared. Positive likelihood proportion, Negative chances ratio, Positive Predictive Value, and Negative Predictive Value. Diabetes requires ongoing monitoring because it is spreading globally at an increasing rate. To verify this, employed k-nearest neighbors (KNN), XGB classifier, Support Vector Machine, Randomized Forest, Logistic regression CV, Neural Networks (ANN), Decision Tree, and Logical Regression (Muthukumar , 2022)

2.1 Design:

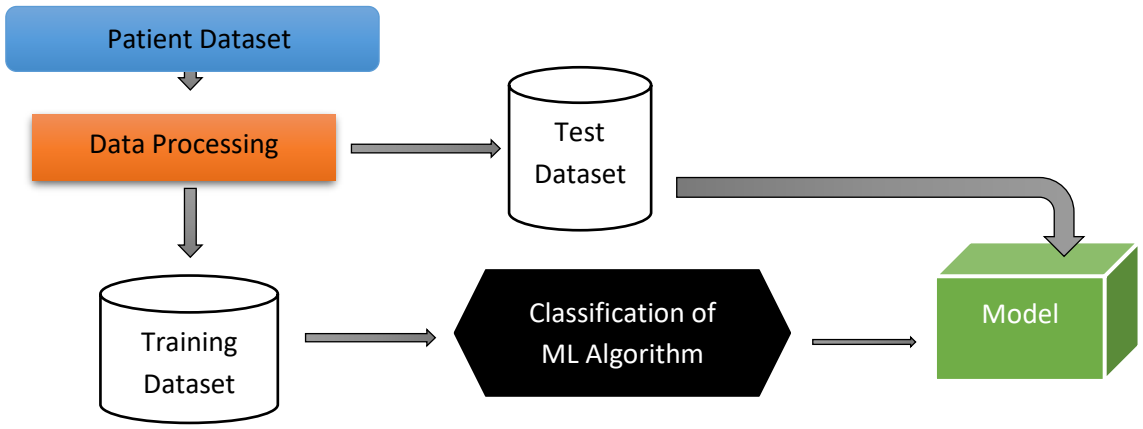


Figure 1: Architecture diagram of the proposed method

Explanation:

The architecture diagram for our suggested method is shown in Figure 1. Here, the outcome is generated using a few processes. Datasets depending on diabetic patient data are collected in the first step. Data pre-processing is then used to scale or normalize the data. Data analysis is carried out on the dataset with the correct variable identification, discovering both dependent and independent variables. The dataset where the data pattern is learned is then visualized to understand the data better. The training model is then used to learn data. Only after the training process is complete do we wish to test the data. We employ machine learning to classify the data during the testing phase. Finally, after trying different machine-learning techniques, a more effective algorithm is utilized to forecast the outcome. Accuracy terms are considered about the system's performance measures.

Block Diagram

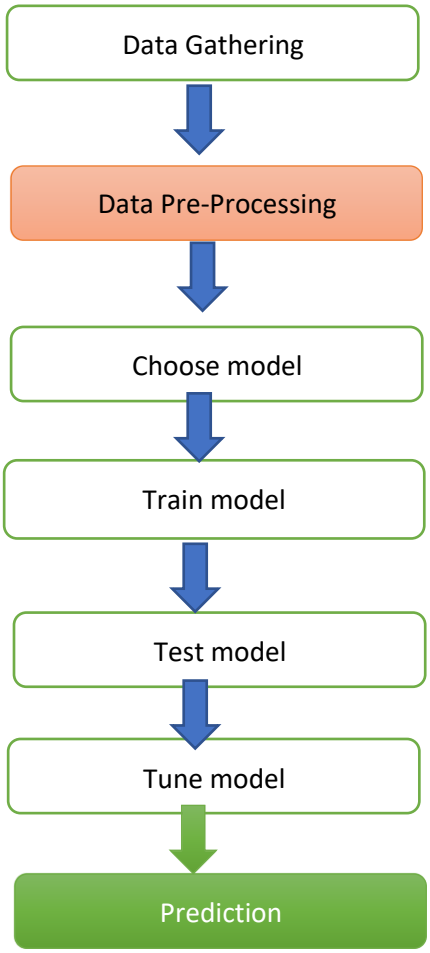


Figure 2: Block diagram of our system

There are mainly five steps involved in this project.

1: Dataset Collection

2: Data Pre-processing

3: Build a model

4: Classification

5: Deployment

2.2 Datasets Collection

Sets of data are referred to as datasets. The collecting of datasets is a crucial requirement for our endeavor. Because if you only select relevant data, the system will generate an accurate answer. Data collection is a vital system component in all data science, machine learning, and deep learning concepts. The data gathered was divided into a training set and a test set to forecast the given data. Typically, the Training data set and Test dataset are divided into 7:3 ratios. The training set is used to apply the data model that was produced using machine learning techniques, and the test set is predicted based on the accuracy of the test results.

Ethical use of Data

We will use an open data source. The National Institute of Diabetes and Kidney Diseases is the source of this dataset. The goal is to determine whether a patient has diabetes based on diagnostic parameters. All patients at this facility are Pima Indian women at least 21 years old. The datasets consist of one target variable, the Outcome, and several medical predictor variables. The patient's Bodyweight, insulin level, age, number of previous pregnancies, and other factors are predictor variables. Pregnancies, Glucose, Blood Pressure, Skin Density, Insulin, BMI, Diabetic, Pedigree Function, Maturity level, and Outcome are the dataset's only features.

- **Glucose**: Plasma glucose concentration 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)
- **Pregnancies:** Number of times pregnant
- **BMI**: Body mass index (weight in kg/(height in m)^2)
- **Diabetes Pedigree Function**: Diabetes pedigree function
- **Age**: Age (years)
- **Outcome**: Class variable (1 or 0)

Number of Units present: 769 (www.kaggle.com, n.d.)

2.3 Data Pre-Processing

Once the datasets have been collected, the system's Data Pre-Processing stage comes next. Preparing the data is a crucial step in predicting the system's outcome. In our dataset, the data pretreatment phase is utilized to remove the undesired data. Additionally, check the data's unique, duplicate, missing and tally values. Our data pre-processing step's primary goal is to employ data analysis to organize the information correctly. Utilizing Python's Pandas module, several distinct data cleaning tasks are carried out with an emphasis on data values, and perhaps the enormous data cleaning is done more quickly.

	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

Figure 3: shows the mean, median, mode, min, and max in the data

The error rate of an ML model can be obtained using validation methods and is thought to be near the actual error rate of the dataset. However, validation methods may be unnecessary if the data volume sufficiently represents the population. However, in practical situations, you'll have to work with data samples that might not accurately reflect the entire dataset. As well as identifying and eliminating duplicates, this feature may also provide a breakdown of the data type, be it a float variable or an integer, and locate any missing values. The subset of the training dataset is used for objectively assessing the model's fit while tuning its hyperparameters.

	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

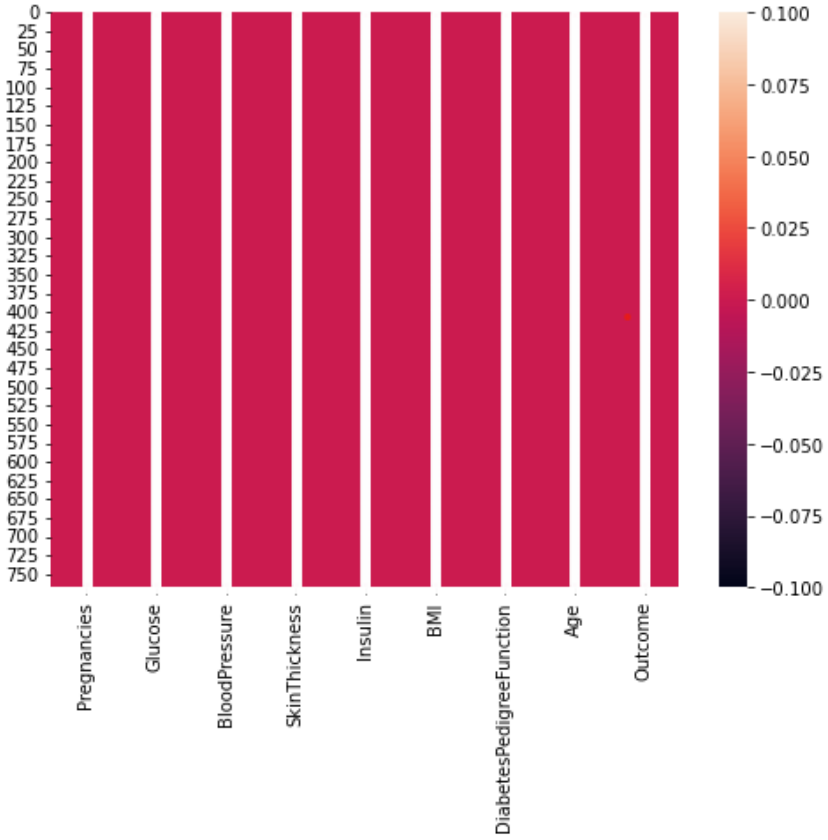
Variable identification with Uni-variate, Bi-variate and Multi-variate analysis:

	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

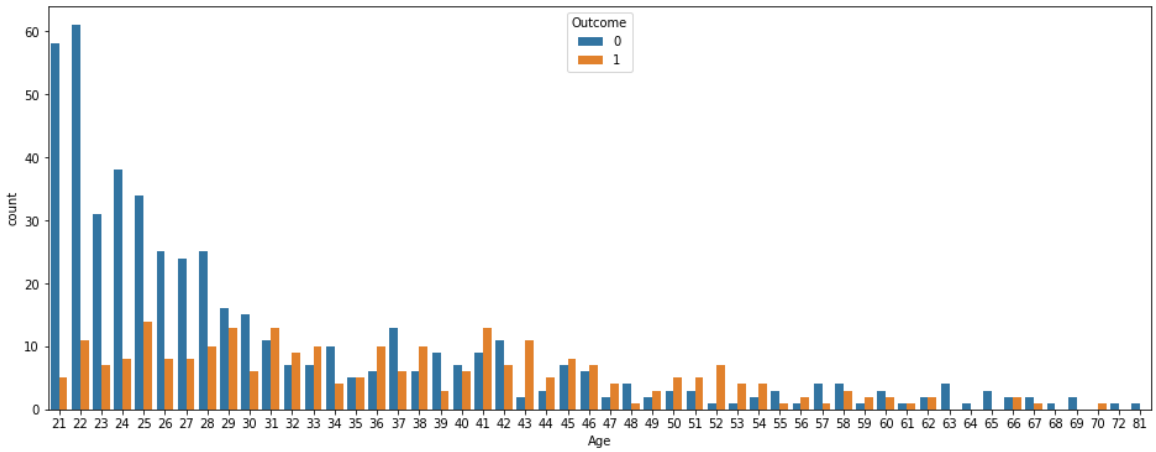
Data Validation

In introductory statistics and machine learning, data visualization is a crucial ability. The focus of statistics is on statistical estimates and representations of data. An essential set of tools for obtaining a qualitative understanding is provided by data visualization. It helps find patterns, data corruption, misfits, and much more when examining and trying to get to know a dataset. Data representations express and display meaningful relationships in graphs and charts that are more vivid and fascinating to clients than assessments of association or importance with some domain knowledge.

2.4 Data analysis and visualization

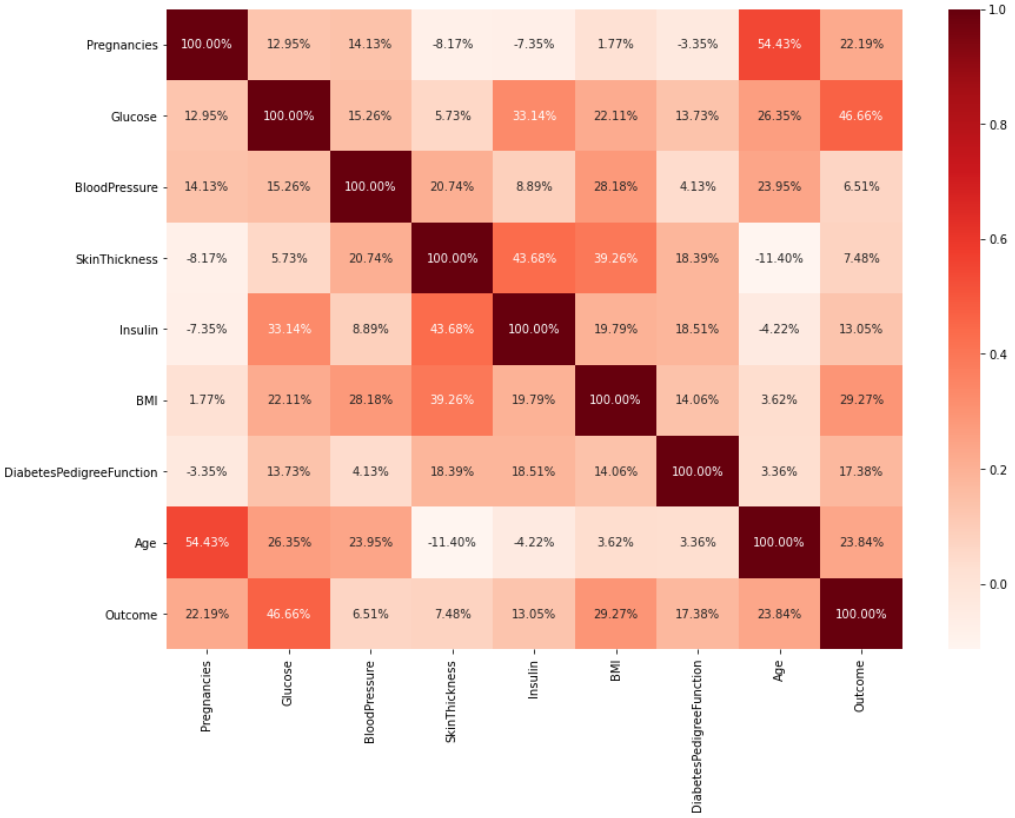


Data will not always make sense unless presented visually, such as with charts and graphs. However, statistics and machine learning appreciate the fast visualization of data samples and other objects. It will demonstrate how to utilize various plot types to analyze your data and the many plot types you'll need to be familiar with when data is visualized in Python.



The chart shows people count with the corresponding age

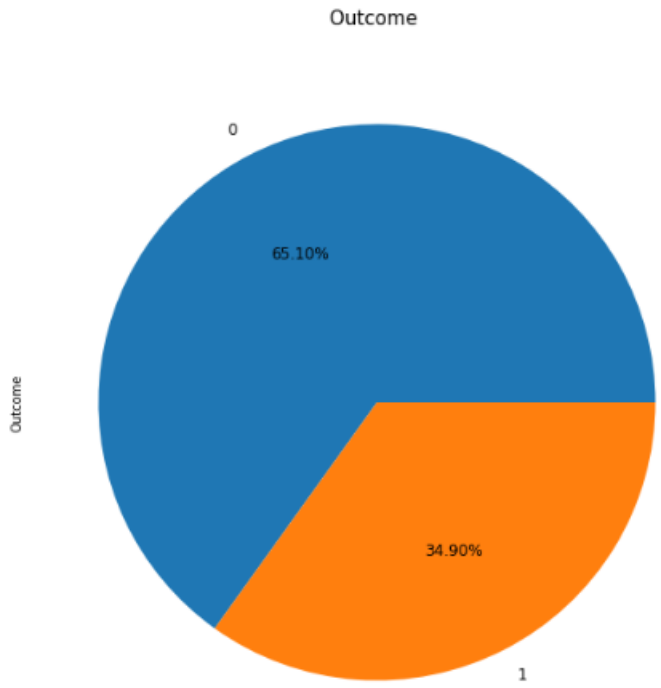
Data preprocessing is a method for cleaning up unstructured data. In other words, the data is continuously acquired in a raw format that is impractical for analysis whenever it is gathered from several sources. When using a model using the Machine Learning technique, having clean, well-organized data is essential for good outcomes.



It is crucial to compare the performance of various machine learning algorithms, and this tutorial will show how to set up a test harness in Python using scikit-learn to do just that. It can use this test harness as a template for your machine learning problems and add more and different algorithms to compare. It's expected that the performance of the various models would vary. Using resampling techniques like cross-validation, you may gain a rough idea of each model's potential accuracy on unseen data. It must utilize these estimations to select the top one or two models from the set of models you have developed. If you have access to a new dataset, it's a smart option to visualize it in various ways so that you may analyze it from many aspects. In the same manner, the model selection makes use of the same concept. To fairly compare machine learning algorithms, they all must be assessed in the same way using the same data. It can be accomplished by mandating that all algorithms be evaluated using the same test harness.

```
#Propagation by variable
def PropByVar(df, variable):
    dataframe_pie = df[variable].value_counts()
    ax = dataframe_pie.plot.pie(figsize=(10,10), autopct='%1.2f%%', fontsize = 12)
    ax.set_title(variable + ' \n', fontsize = 15)
    return n.round(dataframe_pie/df.shape[0]*100,2)

PropByVar(data, 'Outcome')
```

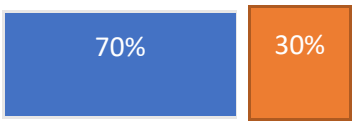
```
0    65.1
1    34.9
Name: Outcome, dtype: float64
```



Sixty-five percent of the patient are diabetic negative, and thirty-four percent of the present patient are diabetic patients.

```
#We'll use a test size of 30%. We also stratify the split on the response variable, which is very important
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1, stratify=y)
```

Here data is split into thirty percent as test data, and seventy percent will be treated as train data.



3. Implementation

Build a model

- ✓ Logistic Regression
- ✓ Support Vector Machine
- ✓ Decision Tree
- ✓ Random Forest

3.1 Logistic Regression

Logistic regression is a statistical way used to calculate which model best describes the association between a set of independent (predictor or explanatory) factors and a dichotomous feature of interest (dependent variable = response or outcome variable). Outcomes of categorical dependent variables can be predicted using logistic Regression. This implies that the result must take the form of a categorical or discrete number. Yes or no, zero or one, true or false, etc., but instead of presenting the exact answer as 0 or 1, it provides probabilistic values that range from 0 to 1. Logistic Regression is comparable to Linear Regression but applied differently. Linear Regression solves regression problems, while logistic Regression solves classification difficulties.

Sigmoid Function

The sigmoid function is used to route the predicted values to probabilities. Any actual number is converted to a number between zero and one. The logistic regression's result must fall within the range of 0 and 1, and because it cannot go beyond this value, it has the shape of an "S" curve. The sigmoid function or logistic function is another name for the S-form curve.

Measurements:

	Predicted (Class 1)	Predicted (Class 2)
(Class 1) Actual	True positive (TP)	False Negative (FN)
(Class 2) Actual	False Positive (FP)	True Negative (TN)

True Positive

It means that the observation is positive and is predicted to be positive. For example- A person has diabetes, and our model also predicted a diabetic

False Negative

It means that the observation is positive, and the prediction is negative. So, for example, a person has diabetes, but our model predicts they do not have diabetes.

True Negative

True negative means that both the observation and the prediction are wrong. For example, if a person doesn't have diabetes, our model also predicts that they don't have diabetes.

False Positive

When an observation is negative, but the prediction is positive, that is a false positive. So, for example, a person does not have diabetes, but our model said they did.

Accuracy calculation:

Accuracy = (TP + TN) / (TP + TN + FP + FN)

Accuracy is the most obvious way to measure performance, and it's the number of correct guesses compared to the total number of observations. Accuracy is a great measure, but only when the number of false positives and false negatives are almost the same in the data set.

Recall:

The proportion of positive observed values correctly predicted. (The ratio of actual defaulters that the model will correctly predict)

Recall = TP / (TP + FN)

Precision:

Precision refers to the positive predictive value, often the proportion of correct positive predictions.

Precision = TP / (TP + FP)

Specificity:

Specificity is the rate of true negatives or the number of correctly identified negatives.

Specificity = TN / (TN + FP)

F1 Score:

The F1 Score is the average of the weighted scores for Precision and Recall. So, both false negatives and positives are considered in this score. F1 is usually better than accuracy, especially if your classes are not all the same size. The best way for accuracy to work is if both false positives and false negatives cost the same. So, it's good to look at both Precision and Recall if the costs of false positives and negatives differ.

General Formula:

F- Measure = 2TP / (2TP + FP + FN)

F1-Score Formula:

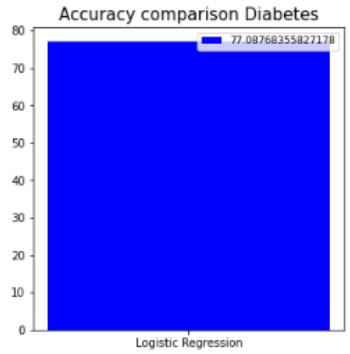
F1 Score = 2*(Recall * Precision) / (Recall + Precision)

Logistic Regression :

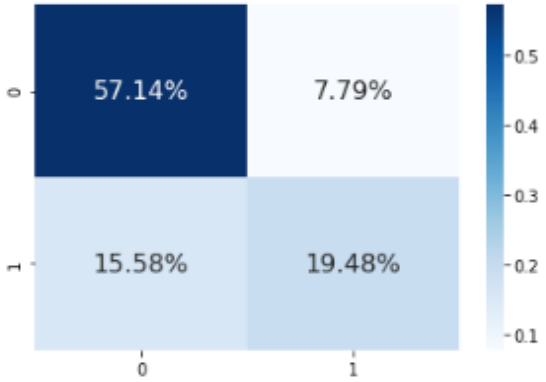
```
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_val_score

logR= LogisticRegression()

logR.fit(X_train,y_train)

predictLR = logR.predict(X_test)

print("")
print('Classification report of Logistic Regression Results:')
print("")
print(classification_report(y_test,predictLR))
x = (accuracy_score(y_test,predictLR)*100)

print('Accuracy result of Logisticregression is:', x)

print("")

cm2=confusion_matrix(y_test,predictLR)
print('Confusion Matrix result of Logistic Regression is:\n',cm2)
print("")
sensitivity2 = cm2[0,0]/(cm2[0,0]+cm2[0,1])
print('Sensitivity : ', sensitivity2 )
print("")
specificity2 = cm2[1,1]/(cm2[1,0]+cm2[1,1])
print('Specificity : ', specificity2)
print("")

accuracy = cross_val_score(logR, X, y, scoring='accuracy')
print('Cross validation test results of accuracy:')
print(accuracy)
#get the mean of each fold
print("")
print("Accuracy result of Logistic Regression is:",accuracy.mean() * 100)
LR=accuracy.mean() * 100



def graph():
    import matplotlib.pyplot as plt
    data=[LR]
    alg="Logistic Regression"
    plt.figure(figsize=(5,5))
    b=plt.bar(alg,data,color=("b"))
    plt.title("Accuracy comparison Diabetes",fontsize=15)
    plt.legend(b,data,fontsize=9)



graph()
```

Results

```
Classification report of Logistic Regression Results:

              precision    recall  f1-score   support

           0       0.79      0.88      0.83       150
           1       0.71      0.56      0.63        81

    accuracy                           0.77       231
   macro avg       0.75      0.72      0.73       231
weighted avg       0.76      0.77      0.76       231

Accuracy result of Logisticregression is: 76.62337662337663

Confusion Matrix result of Logistic Regression is:
 [[132  18]
 [ 36  45]]

Sensitivity :  0.88

Specificity :  0.5555555555555556

Cross validation test results of accuracy:
[0.77272727 0.74675325 0.75974026 0.81699346 0.75816993]

Accuracy result of Logistic Regression is: 77.08768355827178
```




```
True Positive : 45
True Negative : 36
False Positive : 18
False Negative : 132

True Positive Rate : 0.2542372881355932
True Negative Rate : 0.6666666666666666
False Positive Rate : 0.3333333333333333
False Negative Rate : 0.7457627118644068

Positive Predictive Value : 0.7142857142857143
Negative predictive value : 0.21428571428571427
[[132  18]
 [ 36  45]]
```



We were able to obtain max 77 % of accuracy for logistic regression

3.2 Decision Tree

As a form of predictive modelling, decision tree analysis has numerous potential uses. An algorithmic method can divide the dataset in several ways depending on the conditions, allowing for the construction of decision trees. In the class of algorithms known as supervised algorithms, decision trees stand out as the most effective. They can be applied to tasks requiring classification and regression. A tree's decision nodes, where the data is divided, and leaves, where the results are found, are its two key components.

Why Decision tree :

- ✓ The best thing about decision trees is that they are easy to understand. Other machine learning models are almost like black boxes, but decision trees make it easy to comprehend what our algorithm is doing
- ✓ Decision Trees can be trained with less data than many other Machine Learning techniques.
- ✓ Classification and regression are both possible by utilizing them.
- ✓ They are easy to use and forgiving of missing data.

Drawback

They can be sensitive to outliers and tend to overfit the training data.

They are learners:

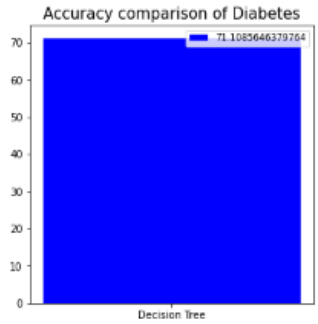
A single decision tree often doesn't make perfect predictions, so many trees are often put together to create "forests" to make more robust ensemble models.

Decision Tree Classifier :

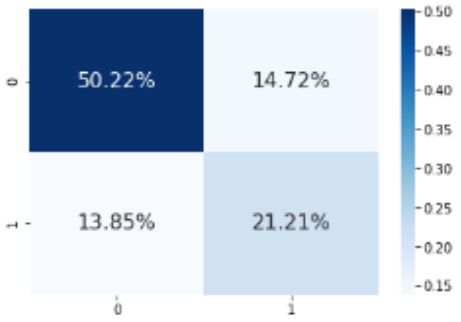
```
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score

DT=DecisionTreeClassifier()

DT.fit(X_train,y_train)

predictDT = DT.predict(X_test)

print("")
print('Classification report DecisionTree classifier Results:')
print("")
print(classification_report(y_test,predictDT))

print("")
x = (accuracy_score(y_test,predictDT)*100)

print('Accuracy result of DecisionTree  is:', x)
print("")

cm2=confusion_matrix(y_test,predictDT)
print('Confusion Matrix result of DecissionTree Classifier is:\n',cm2)
print("")
sensitivity2 = cm2[0,0]/(cm2[0,0]+cm2[0,1])
print('Sensitivity : ', sensitivity2 )
print("")
specificity2 = cm2[1,1]/(cm2[1,0]+cm2[1,1])
print('Specificity : ', specificity2)
print("")

accuracy = cross_val_score(DT, X, y, scoring='accuracy')
print('Cross validation test results of accuracy:')
print(accuracy)
#get the mean of each fold
print("")
print("Accuracy result of DecisionTree Classifier is:",accuracy.mean() * 100)
dt=accuracy.mean() * 100



def graph():
    import matplotlib.pyplot as plt
    data=[dt]
    alg="Decision Tree"
    plt.figure(figsize=(5,5))
    b=plt.bar(alg,data,color=("b"))
    plt.title("Accuracy comparison of Diabetes",fontsize=15)
    plt.legend(b,data,fontsize=9)



graph()
```

Results

```
Classification report DecisionTree classifier Results:

              precision    recall  f1-score   support

           0       0.78      0.77      0.78       150
           1       0.59      0.60      0.60        81

    accuracy                           0.71       231
   macro avg       0.69      0.69      0.69       231
weighted avg       0.72      0.71      0.72       231


Accuracy result of DecisionTree  is: 71.42857142857143

Confusion Matrix result of DecissionTree Classifier is:
 [[116  34]
 [ 32  49]]

Sensitivity :  0.7733333333333333

Specificity :  0.6049382716049383

Cross validation test results of accuracy:
[0.67532468 0.65584416 0.68831169 0.81045752 0.7254902 ]

Accuracy result of DecisionTree Classifier is: 71.1085646379764
```




```
True Positive : 49
True Negative : 32
False Positive : 34
False Negative : 116

True Positive Rate : 0.296969696969697
True Negative Rate : 0.48484848484848486
False Positive Rate : 0.5151515151515151
False Negative Rate : 0.703030303030303

Positive Predictive Value : 0.5903614457831325
Negative predictive value : 0.21621621621621623
[[116  34]
 [ 32  49]]
```



We were able to obtain 71 % of accuracy for Decision tree

3.3 Random forest Algorithm

A popular machine learning method that belongs to the supervised learning technique is Random Forest. It applies to both Classification and Regression problems in Machine Learning. Furthermore, it is based on ensemble learning, which integrates multiple classifiers to solve a complex issue and improve the model's performance.

Random Forest is a classifier that, as its name suggests, Uses the average of several decision trees on subsets of a dataset to increase its forecast accuracy. Random Forest is a technique that takes the standard to improve the predictive accuracy of that dataset. The random forest model does not rely on a single decision tree. Instead, it considers the prediction from each tree in the forest. It determines the final output based on which tree's forecast received the most votes. The more trees in the forest, the better the accuracy and the prevention of the issue of overfitting.

Since the random forest uses more than one tree to predict the class of the dataset, some decision trees may give the correct answer while others may not. However, when all the trees are put together, they provide the right answer. So, here are two ideas about how to make a better Random Forest classifier:

- ✓ In the feature variable of the dataset, there should be some real numbers, not just guesses, so that the classifier can make accurate predictions instead of just guesses.
- ✓ Each tree's predictions must be very different from the others.

Why use Random Forest?

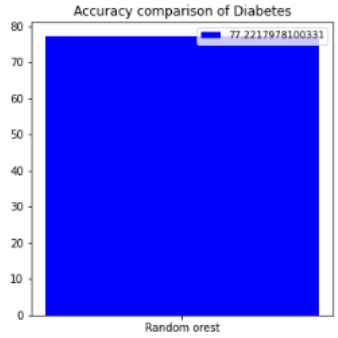
- ✓ When compared to other algorithms, it requires significantly less time to train.
- ✓ The predicted production is exact. Despite the massive size of the dataset, it still performs well.
- ✓ Moreover, it can still be precise even if much of the data is absent.

Random Forest:

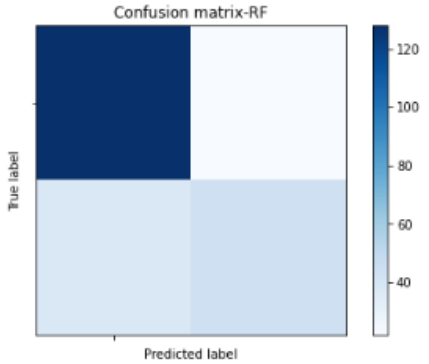
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.model_selection import cross_val_score

rfc = RandomForestClassifier()

rfc.fit(X_train,y_train)

predictR = rfc.predict(X_test)

print("")
print('Classification report of Random Forest Results:')
print("")

print(classification_report(y_test,predictR))
x = (accuracy_score(y_test,predictR)*100)

print('Accuracy result of Random Forest is:', x)
print("")
cm1=confusion_matrix(y_test,predictR)
print('Confusion Matrix result of Random Forest is:\n',cm1)
print("")
sensitivity1 = cm1[0,0]/(cm1[0,0]+cm1[0,1])
print('Sensitivity : ', sensitivity1 )
print("")
specificity1 = cm1[1,1]/(cm1[1,0]+cm1[1,1])
print('Specificity : ', specificity1)
print("")

accuracy = cross_val_score(rfc, X, y, scoring='accuracy')
print('Cross validation test results of accuracy:')
print(accuracy)
#get the mean of each fold
print("")
print("Accuracy result of Random Forest is:",accuracy.mean() * 100)
RFC=accuracy.mean() * 100




def graph():
    import matplotlib.pyplot as plt
    data=[RFC]
    alg="Random orest"
    plt.figure(figsize=(5,5))
    b=plt.bar(alg,data,color=("b"))
    plt.title("Accuracy comparison of Diabetes")
    plt.legend(b,data,fontsize=9)



graph()
```

Results

```
Classification report of Random Forest Results:

              precision    recall  f1-score   support

           0       0.78      0.85      0.81       150
           1       0.67      0.54      0.60        81

    accuracy                           0.74       231
   macro avg       0.72      0.70      0.71       231
weighted avg       0.74      0.74      0.74       231

Accuracy result of Random Forest is: 74.45887445887446

Confusion Matrix result of Random Forest is:
 [[128  22]
 [ 37  44]]

Sensitivity :  0.8533333333333334

Specificity :  0.5432098765432098

Cross validation test results of accuracy:
[0.74675325 0.73376623 0.77272727 0.83006536 0.77777778]

Accuracy result of Random Forest is: 77.2217978100331
```




```
True Positive : 44
True Negative : 37
False Positive : 22
False Negative : 128

True Positive Rate : 0.2558139534883721
True Negative Rate : 0.6271186440677966
False Positive Rate : 0.3728813559322034
False Negative Rate : 0.7441860465116279

Positive Predictive Value : 0.6666666666666666
Negative predictive value : 0.22424242424242424
Confusion matrix-RF:
[[128  22]
 [ 37  44]]
```



We were able to obtain a max 77 % of accuracy for Random forest.

3.4 Support Vector Machines

In supervised machine learning, SVM is used as a classification method. For example, when working with two-dimensional data that can be separated linearly, a typical machine learning algorithm looks for boundaries that divide the data in a way that makes misclassification errors as minor as possible.

Using SVMs, we may better partition our space into classes by identifying the optimal line in two dimensions or the optimal hyperplane in more than two dimensions. It is possible to locate the hyperplane (line) by determining the most significant margin or the most significant possible separation between data points belonging to different classes.

Why use SVM

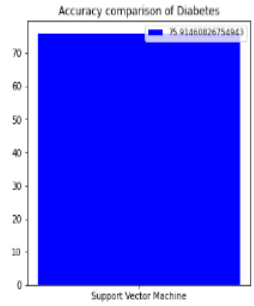
- ✓ SVM works well when there is a clear difference between the classes.
- ✓ SVM gets effectively in spaces with many dimensions.
- ✓ In situations when there are more dimensions than samples, SVM shines.
- ✓ SVM has a low memory consumption.

Support Vector Machines:

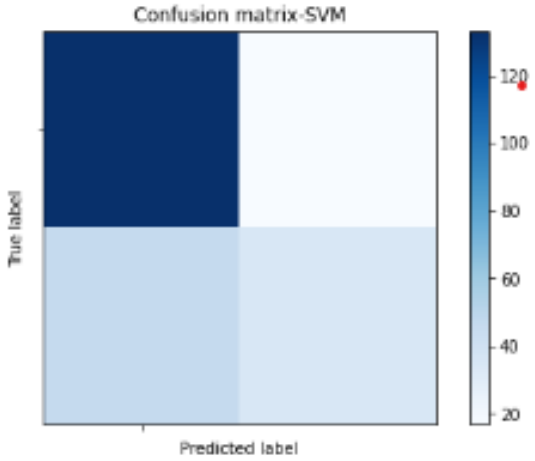
```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.model_selection import cross_val_score


s = SVC()

s.fit(X_train,y_train)

predicts = s.predict(X_test)

print("")
print('Classification report of Support Vector Machines Results:')
print("")

print(classification_report(y_test,predicts))
x = (accuracy_score(y_test,predicts)*100)

print('Accuracy result of Support Vector Machines is:', x)
print("")
cm2=confusion_matrix(y_test,predicts)
print('Confusion Matrix result of Support Vector Machines is:\n',cm2)
print("")
sensitivity1 = cm2[0,0]/(cm2[0,0]+cm2[0,1])
print('Sensitivity : ', sensitivity1 )
print("")
specificity1 = cm2[1,1]/(cm2[1,0]+cm2[1,1])
print('Specificity : ', specificity1)
print("")

accuracy = cross_val_score(s, X, y, scoring='accuracy')
print('Cross validation test results of accuracy:')
print(accuracy)
#get the mean of each fold
print("")
print("Accuracy result of Support Vector Machine is:",accuracy.mean() * 100)
S=accuracy.mean() * 100




def graph():
    import matplotlib.pyplot as plt
    data=[S]
    alg="Support Vector Machine"
    plt.figure(figsize=(5,5))
    b=plt.bar(alg,data,color=("b"))
    plt.title("Accuracy comparison of Diabetes")
    plt.legend(b,data,fontsize=9)



graph()
```

Results

```
Classification report of Support Vector Machines Results:

              precision    recall  f1-score   support

           0       0.74      0.89      0.81       150
           1       0.67      0.43      0.53        81

    accuracy                           0.73       231
   macro avg       0.71      0.66      0.67       231
weighted avg       0.72      0.73      0.71       231

Accuracy result of Support Vector Machines is: 72.72727272727273

Confusion Matrix result of Support Vector Machines is:
 [[133  17]
 [ 46  35]]

Sensitivity :  0.8866666666666667

Specificity :  0.43209876543209874

Cross validation test results of accuracy:
[0.74675325 0.73376623 0.77272727 0.79084967 0.75163399]

Accuracy result of Support Vector Machine is: 75.91460826754943
```




```
True Positive : 35
True Negative : 46
False Positive : 17
False Negative : 133

True Positive Rate : 0.20833333333333334
True Negative Rate : 0.7301587301587301
False Positive Rate : 0.2698412698412698
False Negative Rate : 0.7916666666666666

Positive Predictive Value : 0.6730769230769231
Negative predictive value : 0.2569832402234637
Confusion matrix-SVM:
[[133  17]
 [ 46  35]]
```



We were able to obtain max 75 % of accuracy for SVM

CHAPTER 4. CONCLUSION

Diabetes mellitus is a disease that can cause a variety of difficulties. Therefore, using machine learning to precisely predict and diagnose this disease is worth studying. Based on all of these experiments, we found that the accuracy of using a decision tree is not good, and the rest of the algorithms had better results. Glucose is the most important index for the prediction, but we can't get the best results by only using glucose. If we want to make accurate predictions, we need more indexes. By comparing the results of three different classifications, we can find there is not much difference among the random forest, SVM, and logistic regression. The dataset which was obtained from the 44 Army Reference Hospital, Yusuf Danstoho Memorial Medical Center Kaduna, which consists of nine attribute data. The best result was that ANN provided the highest accuracy, at 97.40 percent, trailed by decision tree-based accuracy at 96.10 percent and K-NN algorithm accuracy at 88.31%. (Evwiekpaefe and Abdulkadir, 2021). We got the best performance for the Pima Indian data set is 77%, which suggests that machine learning can be used to predict diabetes. However, finding the right attributes, classifier, and data mining method is vital. Due to the data, we can't tell what kind of diabetes someone has. In the future, we hope to be able to tell what type of diabetes someone has and look into the proportion of each indicator, which may help us predict diabetes more accurately.

4.1 Evaluation

- ✓ In design and specification (Majumdar and vaidehi), which was used for evaluation. In random forest, which had an accuracy of 72 %. The new model built has an accuracy of 77%
- ✓ In logistic regression, which had an accuracy of 76% (Majumdar and vaidehi). The new model built has an accuracy of 77%
- ✓ In the decision tree retained an accuracy of 74% (Majumdar and vaidehi). The model built has an accuracy of 71%
- ✓ In the support vector machine held accuracy of 68% (Majumdar and vaidehi). The new model built has an accuracy of 75%

There are three possible types of inaccuracy in the current system of medical diagnosis:

First, there's the false-negative variety, which occurs when a person with diabetes is given negative test results.

The second is a case of the false-positive variety. The patient in question does not have diabetes, despite diagnostic tests indicating otherwise.

Third, there's the indeterminate category, which describes situations where a computerized diagnostic tool can't decide what's wrong. For example, a specific patient may be projected in an unclassified type if insufficient knowledge is extracted from historical data. But, the patient must forecast whether or not they fall into the diabetic category. Possible outcomes of incorrect diagnosis include administering unneeded treatment or failing to administer treatment altogether. Therefore, a system that uses machine learning algorithms and data mining approaches to give accurate results with minimal human input is required to avoid or reduce the impact.

4.2 Learning Points

Classifiers must adapt their methods to the specific characteristics of each data type. There is a tradeoff between the amount of time and energy required for calculations and the quality of predictions made on test data. Overfitting can occur when simpler models try to generalize every data point in the data, despite their low variance and substantial bias. Complex models can achieve high levels of prediction performance and accuracy. It was hard to understand the data, but after visualizing it, it was beneficial. I tried many algorithms and figured out which gives more accuracy eventually figured out that random forest and Logistic regression had decent accuracy (77 %) compared to others.

4.3 Future Activity

The Python programming language provides the Flask web framework. Without having to start from scratch, Flask allows developers to create websites with a library and collection of programs. Flask displays the results on a webpage where users can input the required parameters to show corresponding results.

- ✓ To optimize, we will employ Stochastic Gradient Descent (SGD).
- ✓ Automate this procedure by displaying the prediction outcome in a web application.
- ✓ To maximize work for implementation in an artificial intelligence environment.

REFERENCE:

1. Mujumdar, A. and DR. Vaidehi v (2019). ScienceDirect-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/) Peer-review under responsibility of the scientific committee of the INTERNATIONAL CONFERENCE ON RECENT TRENDS IN Diabetes Prediction using Machine Learning Algorithms-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/) Peer-review under responsibility of the scientific committee of the INTERNATIONAL CONFERENCE ON RECENT TRENDS IN ADVANCED COMPUTING 2019. In: *INTERNATIONAL CONFERENCE ON RECENT TRENDS IN ADVANCED COMPUTING*. pp.292–299. doi:10.1016/j.procs.2020.01.047.

2. www.kaggle.com. (n.d.). *Diabetes Dataset*. [online] Available at: https://www.kaggle.com/datasets/mathchi/diabetes-data-set?resource=download [Accessed 8 Jul. 2022].

3. Geetha*, G. and Prasad, Dr.K.Mohana. (2020). Prediction of Diabetics using Machine Learning. *International Journal of Recent Technology and Engineering (IJRTE)*, 8(5), pp.1119–1124. doi:10.35940/ijrte.e6290.018520.

4. Indoria, P. and Rathore, Y.K. (2018). *IJERT-A Survey: Detection and Prediction of Diabetes Using Machine Learning Techniques A Survey: Detection and Prediction of Diabetes Using Machine Learning Techniques*. *IJERT Journal International Journal of Engineering Research and Technology*. IJERT.

5. M, S., J, N., B, H. and Muthukumar, Mrs.S. (2022). Prediction of Diabetes Using Logistics Regression Algorithms with Flask. *International Journal for Research in Applied Science and Engineering Technology*, 10(5), pp.26–33. doi:10.22214/ijraset.2022.41968.

6. Evwiekpaefe, a. (2021). *A predictive model for diabetes using machine a predictive model for diabetes using machine learning techniques (a case studyof some selected learning techniques (a case studyof some selected hospitals in kaduna metropolis) hospitals in kaduna metropolis)*.

APPENDIX

Project Requirements

General:

The requirements for a system are the fundamental limitations within which its development must proceed. During the system's design process, requirements are gathered. The requirements to be discussed are as follows.

1. Functional requirement

The following libraries are used: SK-learn, pandas, NumPy, matplotlib, and seaborn.

SK- learn:

Through a Python programming interface, it offers a variety of practical tools for statistical modeling and machine learning, including classification, regression, clustering, and dimensionality reduction.

NumPy:

It provides multidimensional array objects and variants like masks and matrices that can be applied to different mathematical operations.

Pandas:

It significantly helps in handling data frames. Data representation, with less amount of code and more work, is done. Efficiently takes large data sets.

Matplotlib:

Matplotlib is powerful because it lets users make many different plots. As a result, it is used in many user interfaces, such as IPhython shells, Python scripts, Jupyter notebooks, web applications, and GUI toolkits.

2 . Non-Functional Requirements:

1. Defining the Problem
2. Collection and preparing of data
3. Finding the best-fit algorithm
4. Evaluating algorithms
5. Prediction of result

3. Environmental Requirements:

Software Requirements:

Operating System: Windows, macOS, or Linux

Tool: Anaconda with Jupyter Notebook

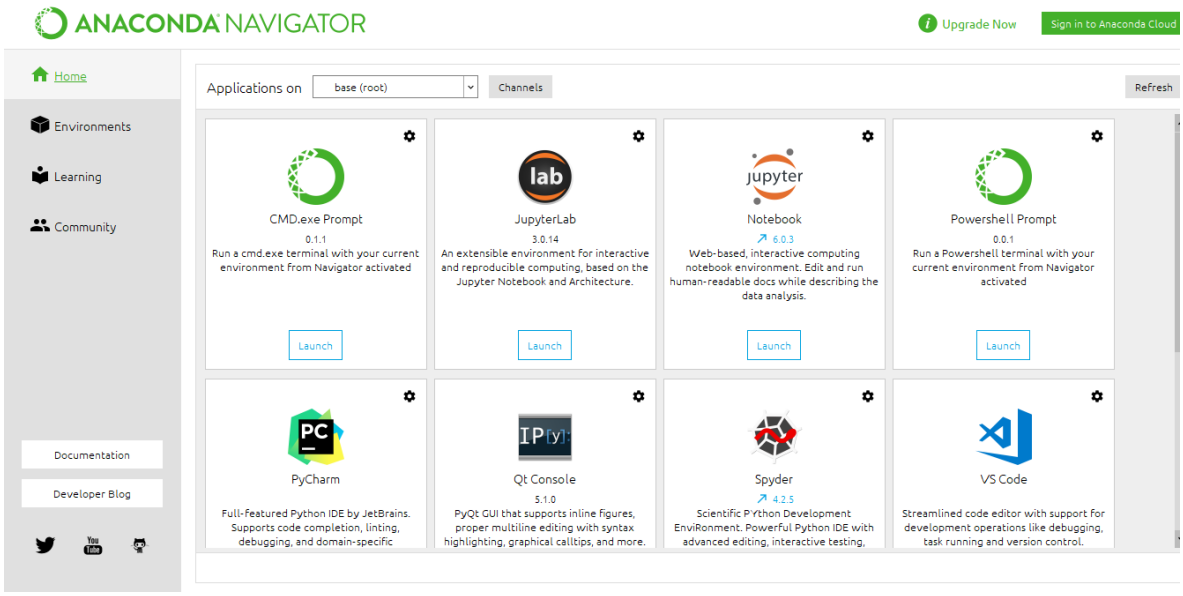
Hardware requirements:

Processor: Pentium IV/III

Hard disk: minimum 80 GB **(Minimum 3 GB disk space)**

RAM: minimum 2 GB

The Anaconda is an unchained and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.). Its goal is to make package management and deployment easier. It comes with a GUI tool called Anaconda Navigator. This tool makes it easy to set up, install, and run tools like Jupyter Notebooks. A Conda Python environment is its particular place. It lets you install packages without changing how Python is set up on your system.



Sample data:

Pregnanci	Glucose	BloodPres	SkinThick	Insulin	BMI	DiabetesF	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1
5	116	74	0	0	25.6	0.201	30	0
3	78	50	32	88	31	0.248	26	1
10	115	0	0	0	35.3	0.134	29	0
2	197	70	45	543	30.5	0.158	53	1
8	125	96	0	0	0	0.232	54	1
4	110	92	0	0	37.6	0.191	30	0
10	168	74	0	0	38	0.537	34	1
10	139	80	0	0	27.1	1.441	57	0
1	189	60	23	846	30.1	0.398	59	1
5	166	72	19	175	25.8	0.587	51	1
7	100	0	0	0	30	0.484	32	1
0	118	84	47	230	45.8	0.551	31	1
7	107	74	0	0	29.6	0.254	31	1
1	103	30	38	83	43.3	0.183	33	0
1	115	70	30	96	34.6	0.529	32	1
3	126	88	41	235	39.3	0.704	27	0
8	99	84	0	0	35.4	0.388	50	0
7	196	90	0	0	39.8	0.451	41	1
9	119	80	35	0	29	0.263	29	1
11	143	94	33	146	36.6	0.254	51	1
10	125	70	26	115	31.1	0.205	41	1
7	147	76	0	0	39.4	0.257	43	1
1	97	66	15	140	23.2	0.487	22	0
13	145	82	19	110	22.2	0.245	57	0
5	117	92	0	0	34.1	0.337	38	0
5	109	75	26	0	36	0.546	60	0
3	158	76	36	245	31.6	0.851	28	1
3	88	58	11	54	24.8	0.267	22	0