MACHINE LEARNING MAJOR PROJECT

DIABETES CLASSIFICATION

Problem Statement

The dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. 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.

Objectives

1.The first is to analyze the data, and see if it is possible to gleam any further information from the data to determine correlation between parameters and diabetes.

2.The second is to attempt to get the best accuracy score using various supervised learning machine learning algorithms. This means to find out which algorithm is able to best predict whether a Pima Indian has diabetes or not based on this dataset.

LIBRARIES AND TOOLS USED

1.JUPYTER NOTEBOOK: The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text.

2.PYTHON: Python is an interpreted, object-oriented, high-level programming language. Python is one of the most versatile programming languages in the world which can easily fit any project.

3.PANDAS: Pandas offers data structure and tools for effective data manipulation and analysis. It provides facts, access to structured data. The primary instrument of Pandas is the two dimensional table consisting of column and row labels, which are called a data frame.

4.NUMPY: The NumPy library uses arrays for its inputs and outputs. It can be extended to objects for matrices.

5.MATPLOTLIB: The Matplotlib package is the most well-known library for data visualization. It is great for making graphs and plots. The graphs are also highly customizable.

6.SEABORN: It is based on Matplotlib. It's very easy to generate various plots such as heat maps, time series and box plots.

ALGORITHMS:

1.K-Nearest Neighbour: It is an algorithm for supervised learning. Where the data is 'trained' with data points corresponding to their classification. Once a point is to be predicted, it takes into account the 'K' nearest points to it to determine its classification.

K Nearest Neighbor(KNN)

```
In [19]: from sklearn.neighbors import KNeighborsClassifier

In [20]: k=6
         neigh=KNeighborsClassifier(n_neighbors=k).fit(x_train,y_train)
         print(neigh)

         KNeighborsClassifier(n_neighbors=6)

In [21]: yhat=neigh.predict(x_test)
         yhat[0:5]

Out[21]: array([0, 0, 0, 1, 0], dtype=int64)

In [22]: from sklearn import metrics
         print("Test set accuracy:",metrics.accuracy_score(y_test,yhat))

         Test set accuracy: 0.7662337662337663
```

2.Decision Tree: They are build using recursive partitioning to classify data, by decreasing impurity.

Decision Tree

```
In [26]: from sklearn.tree import DecisionTreeClassifier

In [27]: outcomeTree=DecisionTreeClassifier(criterion="entropy",max_depth=4).fit(x_train,y_train)
         print(outcomeTree)

         DecisionTreeClassifier(criterion='entropy', max_depth=4)

In [28]: yhat=outcomeTree.predict(x_test)
         yhat[0:5]

Out[28]: array([0, 0, 0, 0, 0], dtype=int64)

In [29]: from sklearn import metrics
         print("Test set accuracy:",metrics.accuracy_score(y_test,yhat))

         Test set accuracy: 0.7792207792207793
```

3.Support Vector Machine: It is a supervised algorithm that can classify cases by finding a separator. SVM works by first mapping data to a high dimensional feature space so that data points can be categorized, even when the data are not otherwise linearly separable.

Support Vector Machine

```
In [31]: from sklearn import svm

In [32]: clf=svm.SVC(kernel='rbf').fit(x_train,y_train)

In [33]: yhat=clf.predict(x_test)
         yhat[0:5]

Out[33]: array([0, 0, 0, 0, 0], dtype=int64)

In [34]: from sklearn import metrics
         print("Test set accuracy:",metrics.accuracy_score(y_test,yhat))

         Test set accuracy: 0.8181818181818182
```

4.Logistic Regression: Logistic regression is a statistical and machine learning technique for classifying records of a dataset based on the values of the input fields.

Logistic Regression

```
In [35]: from sklearn.linear_model import LogisticRegression

In [36]: LR= LogisticRegression(C=0.01,solver='liblinear').fit(x_train,y_train)
         LR

Out[36]: LogisticRegression(C=0.01, solver='liblinear')

In [37]: yhat = LR.predict(x_test)
         yhat

Out[37]: array([0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1,
                0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1,
                0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1,
                0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1,
                0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
                0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
                0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0],
               dtype=int64)

In [38]: yhat_prob = LR.predict_proba(x_test)
         yhat_prob

Out[38]: array([[0.70831592, 0.29168408],
                [0.71859942, 0.28140058],
                [0.76092242, 0.23907758],
                [0.75296754, 0.24613246],
```

```
In [39]: from sklearn import metrics
         print("Test set accuracy:",metrics.accuracy_score(y_test,yhat))

         Test set accuracy: 0.7207792207792207
```

CONCLUSION:

India is considered to be the diabetes capital of world. Diabetes is one of the primary causes of mortality in India. It becomes very important to spot the disease early as many people might develop it silently especially in the young workforce. In this regard, Machine Learning may play a stellar role in preventing diabetes among those who are at moderate to high risk by carefully predicting patterns based upon several variables.

Using K-Nearest Neighbour the accuracy is **76.62%**, Using Decision Tree accuracy is **77.92% ,** Using Logistic Regression accuracy is **72.07%.**Using Support Vector Machine accuracy is **81.88%,** this is a high level of accuracy and it means our model may be trustworthy enough to use. From the above models, **Support Vector Machine** gives optimal accuracy compared to other algorithms.