**Supervised Learning Hands on Assignment**

**Diabetes.csv dataset**

The dataset used for the given problem, that is “diabetes.csv” has nine characteristics or features, so the dimension of this dataset is nine. The first eight of these are the prospective physical predictors for the condition of diabetes in a patient and are independent of each other. The characteristics of Pregnancies, Glucose, Blood Pressure, Skin Thickness, Insulin and Age have discrete values while the two which are BMI and Diabetes Pedigree Function have continuous values. The last characteristic which is named “outcome”, is of binary type with a value of either 0 or 1 and is the class label. It is dependent on some or all, of the first eight characteristics and it signifies whether a patient suffers from diabetes or not. The total number of instances or patient observations in this dataset is 768.

The data in this dataset has been collected by the National Institute of Diabetes and Digestive and Kidney Diseases. The constraints on this data are that all the observations are for females who are older than twenty years and who are of Pima Indian heritage [1].

The Steps for the given assignment are the following.

First, we load the data and import the libraries which would be used in all the models.

Then, we perform some Exploratory Data Analysis (EDA) to get the summary statistics of the data and correlation between each pair of features and between the features and the class label. We use “describe” and “corr” function for this and get the following outputs.

Table

Description automatically generated

Table

Description automatically generated

After this we create and run the first three models which are K Nearest Neighbor [2], Decision Tree [3] and the Naïve Bayes [4] models in the same manner as we did in the first three assignments to get different accuracy scores after some hyperparameter tuning.

We use the same number of splits (10) in both K nearest neighbor and Decision Tree so that we can properly compare the performances.

KNN code:

from sklearn.neighbors import KNeighborsClassifier

array = df.values

X = array[:,1:8]

Y = array[:,8]

maxScore = 0

maxNeighbor = 0

for num2 in range(2,50):

kfold = KFold(n\_splits=10, random\_state=2, shuffle=True)

model = KNeighborsClassifier(n\_neighbors=num2)

results = cross\_val\_score(model, X, Y, cv=kfold)

num3 = results.mean()

if num3>maxScore:

maxScore=num3

maxNeighbor = num2

print(maxNeighbor)

print(maxScore)

Here, we get the best accuracy score of 0.7617 for 23 neighbors.

We drop the pregnancies feature in all the models as, from trying the various combinations of features, we concluded that the performance improves in all the models on dropping it.

DT code:

from sklearn.tree import DecisionTreeClassifier

SEED = 5

SPLITS = 10

X = df.drop('Outcome', axis=1).values

y = df['Outcome'].values

maxSPEED = 0

maxScore = 0

for num1 in range(2,20):

kfold = KFold(n\_splits=10, random\_state=num1, shuffle=True)

model = DecisionTreeClassifier()

results = cross\_val\_score(model, X, y, cv=kfold)

num3 = results.mean()

if num3>maxScore:

maxScore=num3

maxSPEED = num1

print(maxSPEED)

print(maxScore)

We get the best accuracy score of 0.7267 in this model.

NB code:

from sklearn.naive\_bayes import GaussianNB

X = df.iloc[:, [0,1,2,3,4,5,6]].values

y = df.iloc[:, -1].values

from sklearn.preprocessing import StandardScaler

std\_scaler = StandardScaler()

scaled\_X = std\_scaler.fit\_transform(X)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(scaled\_X, y, test\_size=0.2, random\_state = 0, stratify = y)

model = GaussianNB()

model.fit(X\_train, y\_train)

model.score(X\_test, y\_test)

Here, we get an accuracy score of .7857.

The best performing in the first three models as we already know from the earlier assignments, is the Naïve Bayes with an accuracy score of 0.7857.

After this we build and run the SVM model [5] on the given data excluding the pregnancies feature to get an accuracy score of 0.7987 which is slightly better than Naïve Bayes. We use twenty percent of data for testing and radial basis function kernel with our SVM model.

SVM code:

from sklearn import svm

x = df.iloc[:, :-2]

y = df.iloc[:, -1]

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, random\_state = 0, test\_size = 0.2)

clf = svm.SVC(kernel='rbf')

clf.fit(x\_train,y\_train)

y\_pred = clf.predict(x\_test)

clf.score(x\_test, y\_test)

We get an accuracy score of 0.7987.

Finally, we build and run the adaboost machine learning model [6][7] on the diabetes data after removing the pregnancies feature.

Adaboost code:

from sklearn.ensemble import AdaBoostClassifier

from sklearn.datasets import make\_classification

x = df.iloc[:, :-2]

y = df.iloc[:, -1]

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, random\_state = 0, test\_size = 0.2)

clf = AdaBoostClassifier(n\_estimators=100, random\_state=0)

clf.fit(x\_train,y\_train)

clf.score(x\_test,y\_test)

With the adaboost model we get an accuracy score of 0.8442. This is the best score we have got so far and substantially better than the rest of the models.

## Resources

1. <https://www.kaggle.com/uciml/pima-indians-diabetes-database>
2. <https://towardsdatascience.com/building-a-k-nearest-neighbors-k-nn-model-with-scikit-learn-51209555453a>
3. <https://github.com/LaxmiChaudhary/Building-Naive-Bayes-Classifier-on-Pima-Diabetic-Dataset>
4. <https://towardsdatascience.com/how-to-tune-a-decision-tree-f03721801680>
5. <https://www.kaggle.com/rishidamarla/svm-tutorial-with-diabetes-dataset>
6. https://www.kaggle.com/shihabshahriar/adaboost-simple-python-implementation
7. <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html>