1. **Frame the problem and look at the big picture: 1⁄2 page.**

Diabetes is a metabolic disease that increases the blood count of sugar in the blood [1]. This is caused either by the inability of the body to produce enough insulin that converts excess blood in the sugar to glycogen or when the body becomes resistant to insulin it produces thereby causing it not to be able to convert the excess sugar in the body to glycogen [2]. In both cases, there is excess sugar in the blood thereby resulting in the disease known as diabetes. The first scenario is known medically as “Type-1 Diabetes” while the second is regarded as “Type-2 Diabetes”. The third type of diabetes is known as gestational diabetes and it is witnessed by some pregnant women due to hormonal imbalances in the body during pregnancy [3].

Although the physiological effect of the various types of the disease is quite similar, it is worthy of note that Type-2 diabetes is the most common type. Common symptoms of the disease include frequent urination, burning thirsty, feeling famished, extreme fatigue, blurry vision, slow healing cuts/bruises, weight loss (type 1), tingling, pain, or numbness in the hands/feet (type 2) [4].

Complications that can result from the disease include skin complications, eye complications, neuropathy, foot complications, kidney disease (nephropathy), high blood pressure, stroke [4]. Therefore, early detection is very important in managing the disease.

Clinically, the detection of the disease is done by carrying out various medical and laboratory tests. Risk factors associated with the disease include high blood glucose, a history of gestational diabetes, high blood pressure, skin thickness, low insulin levels, disproportionate body mass index (BMI), and a family history of the disease [5].

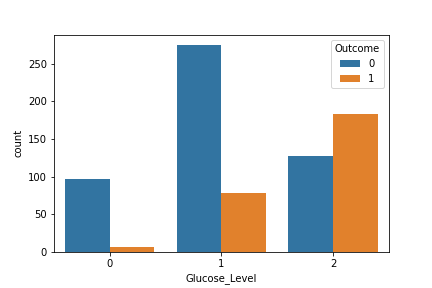
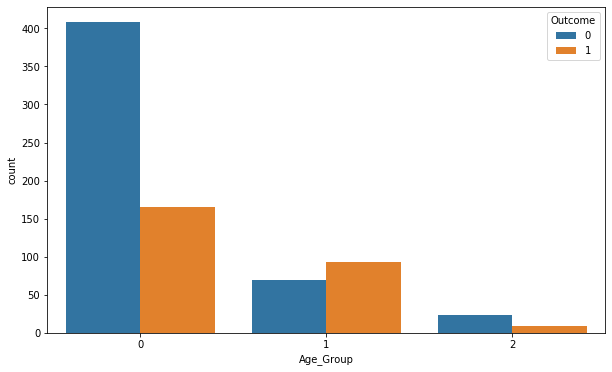
Furthermore, each of the specific risk factors that are associated with the disease has an associated threshold that is suggestive of a likely presence of the disease in an individual. However, the attainment of this value for any isolated risk factor may not necessarily be indicative of the presence of the disease. Therefore, having an effective predictive system based on the risk factors associated with the disease becomes of high importance to aid an effective diagnosis of the disease. This will, in turn, facilitate its early detection and management.

1. **Show 4+ graphs to explore the data and gain insights: 1 1⁄2 page.**



**Figure 1.1: A Count Plot of the Possible Outcomes**

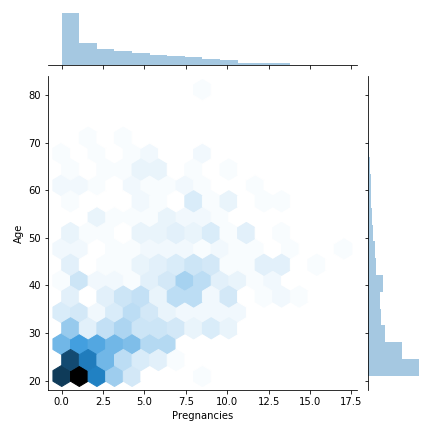
A count plot of the possible outcomes is shown above in figure 1.1. The blue represents the outcome “0” which are the people without the disease, while the red represents the outcome “1” which are people with the disease. From the figure, it can be seen that there are about 500 individuals without the disease while about 268 individuals have the disease.

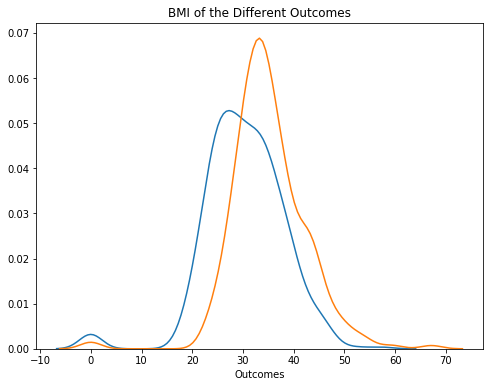


**Figure 1.2: A Count Plot of the Age Groups Figure 1.3: A Count Plot of the Glucose**

**of the Individuals Levels of the Individuals**

The individuals were separated into 3 different age groups. The first group are those who are 40 years old or less (group 0), the second group are those between 41 and 59 years old inclusively (group 1), while the last group are those that are 60 years old or older (group 2). Figure 1.2 shows that a greater percentage of people with the disease are those who are 40 years old or less. Also, the same was done for the glucose level in the blood of the individuals and this is shown in figure 1.3. Group 0 are those with glucose levels of less than 90, group 1 are those with glucose levels that is between 90 and 125 inclusively, while the last group are those with glucose levels that is more than 125. It can be seen than those with glucose levels of more than 125 are more likely to have the disease





**Figure 1.4: A Joint Plot of the Pregnancies Figure 1.5: A KDE Plot of the Glucose**

**Against the Ages Levels of the Individuals**

Figure 1.4 shows a plot of the “Pregnancies” against the age. The figure shows that those that re within the child bearing age more in the data in order to cater for gestational diabetes. Figure 1.5 shows that those with the disease (red) have a higher density than those without it (blue).

1. **Prepare the data to better expose the underlying data patterns to machine learning**

**algorithms: 1⁄2 page.**

In preparing the data for exposing the underlying data patterns to machine learning algorithms, the first step was to check for null values. Null values are values are empty cells or incomplete records that might be in the dataset. In very large data sets, such records might be dropped, but since the set is within the medium range, replacing empty cells with the column would have been done. However, there were no empty cells in the dataset.

Secondly, the ages of the individuals in the group were categorized into three groups. The first group are those that are 40 years old or less (group 0), the second group are those that are between 41 and 59 years old inclusively (group 1), while the last group are those that are above 50 years of age (group 2). This is done in order to be able to have specific categories of ages of the various individuals. This new column was then stored in the dataset and labelled as “Age Groups”.

Furthermore, a dummy variable was created for the newly created “Age Groups” column. This was done by creating 3 columns in the data set and assigning a “1” to the column which corresponds to the age group which a specific record belongs into. Correspondingly, a “0” is assigned to the two other columns which the specific record does not belong into.

Finally, the first column out of the 3 was dropped. This is done so as to have uniqueness among the 3 possible age groups. Also, the initial “Age” column and the created “Age Group” column were dropped in order to avoid diffusion and overfitting. The head rows (2 rows) of the prepared dataset used for the supervised learning for diabetes prediction is given in Table 1.1.

**Table 1.1: The Head Section (First 2 Records) of the Prepared Dataset**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Pregnancies** | **Glucose** | **Blood**  **Pressure** | **Skin**  **Thickness** | **Insulin** | **BMI** | **Diabetes**  **Pedigree Function** | **Outcome** | **1** | **2** |
| **0** | 6 | 148 | 72 | 35 | 0 | 33.6 | 0.627 | 1 | 1 | 0 |
| **1** | 1 | 85 | 66 | 29 | 0 | 26.6 | 0.351 | 0 | 0 | 0 |

1. **Use three machine learning algorithms (one of which must be a deep neural network), discuss their performance, and show a comparative performance table: 1 1⁄2 page.**

The three machine learning algorithms that were used in training the diabetes prediction system are the support vector machine (SVM), the random forest classifier and deep neural network (using TensorFlow). In all cases, the data was randomly divided 2; 70% for training and 30% for testing. Also, as earlier stated, outcome “0” are individuals without the disease while outcome “1” are individuals suffering from the disease.

The results obtained from the various algorithms are discussed as follows;

**Support Vector Machine (SVM)**

Support vector machine classifies data by attempting to estimate support vectors (borders) in the data set. The performance of this algorithm is shown in Table 1.2 below;

**Table 1.2: Performance Analysis of Support Vector Machine**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Outcome** | **Precision** | **Recall** | **F1-score** | **Support** |
| **0** | 0.76 | 0.93 | 0.83 | 150 |
| **1** | 0.77 | 0.46 | 0.57 | 81 |
| **Accuracy** |  |  | 0.76 | 231 |

From the table, it can be seen that SVM recorded a negative predictive value (NPV) of 0.76 and a positive predictive value (PPV) of 0.77. It recorded a true negative rate (TNR) of 0.93 and a true positive rate (TPR) of 0.46. The harmonic mean of precision and recall (F1-score) recorded are 0.83 and 0.57 for the “0” and “1” outcomes respectively. The measure of correct prediction (accuracy) of the algorithm is 0.76.

**The Random Forest Classifier (RFC)**

Random forest classifier fits decision trees on a sample set of data from the dataset. The method improves accuracy by averaging. 200 estimators were used in building the classifier. The performance of the algorithm is show in Table 1.3 below;

**Table 1.3: Performance Analysis of Random Forest Classifier**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Outcome** | **Precision** | **Recall** | **F1-score** | **Support** |
| **0** | 0.81 | 0.84 | 0.83 | 150 |
| **1** | 0.68 | 0.64 | 0.66 | 81 |
| **Accuracy** |  |  | 0.77 | 231 |

From the table, it can be seen that RFC recorded a negative predictive value (NPV) of 0.81 and a positive predictive value (PPV) of 0.68. It recorded a true negative rate (TNR) of 0.84 and a true positive rate (TPR) of 0.64. The harmonic mean of precision and recall (F1-score) recorded are 0.83 and 0.66 for the “0” and “1” outcomes respectively. The measure of correct prediction (accuracy) of the algorithm is 0.77.

**Deep Neural Network (DNN)**

Deep neural network creates multi-directional dense layers of artificial neural network (ANN) between input and output layers. Before using the DNN, the values of the dataset were initially scaled in order to normalize the data. 3 layers of DNN were then applied in training the data set. The performance of the algorithm is show in Table 1.4 below;

**Table 1.4: Performance Analysis of Deep Neural Network**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Outcome** | **Precision** | **Recall** | **F1-score** | **Support** |
| **0** | 0.83 | 0.85 | 0.84 | 150 |
| **1** | 0.71 | 0.68 | 0.69 | 81 |
| **Accuracy** |  |  | 0.79 | 231 |

From the table, it can be seen that DNN recorded a negative predictive value (NPV) of 0.83 and a positive predictive value (PPV) of 0.71. It recorded a true negative rate (TNR) of 0.85 and a true positive rate (TPR) of 0.68. The harmonic mean of precision and recall (F1-score) recorded are 0.84 and 0.69 for the “0” and “1” outcomes respectively. The measure of correct prediction (accuracy) of the algorithm is 0.79.

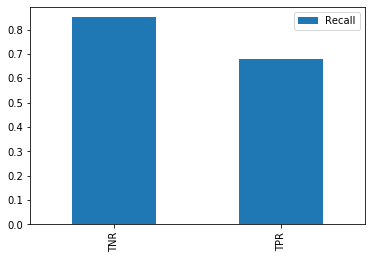
A comparative performance table for the 3 algorithms used for the prediction of the disease is shown if Table 1.5 below;

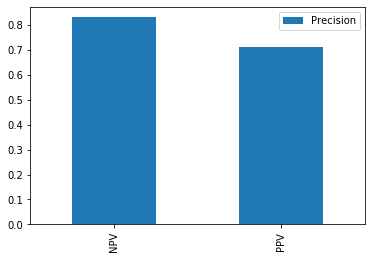
**Table 1.4: Comparative Performance Table of the Algorithms used in the Prediction of Diabetes**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Algorithm** | **NPV** | **PPV** | **TNR**  **(Specificity)** | **TPR**  **(Sensitivity)** | **F-Measure**  **(Negative)** | **F-Measure**  **(Positive)** | **Accuracy** |
| **SVM** | 0.76 | 0.77 | 0.93 | 0.46 | 0.83 | 0.57 | 0.76 |
| **RFC** | 0.81 | 0.68 | 0.84 | 0.64 | 0.83 | 0.66 | 0.77 |
| **DNN** | 0.83 | 0.71 | 0.85 | 0.68 | 0.84 | 0.69 | 0.79 |

**E. For the best performing algorithm, show 4+ graphs: 1 page.**

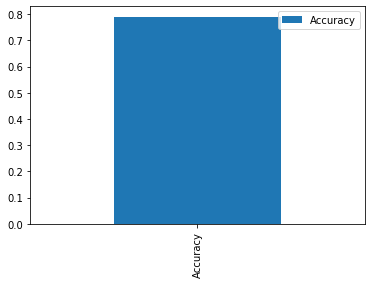
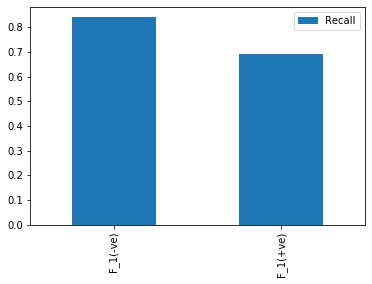
Out of the three algorithms, DNN can be adjudged the best performing algorithm. Graphs showing the performance of the algorithm are shown in figures 1.6 to 1.9 below;





**Figure 1.6: Graph Showing the NPV and Figure 1.7: Graph Showing the TNR and TPR**

**PPV of the DNN Algorithm of the DNN Algorithm**

****

**Figure 1.8: Graph showing the F-Measure of Figure 1.9: Graph showing the Accuracy**

**Scores of the Algorithm of the Algorithm**

**References**

[1] American Diabetes Association, “Diagnosis and classification of diabetes mellitus,” *Diabetes Care*, vol. 32 Suppl 1, no. Suppl 1, pp. S62–S67, Jan. 2009, doi: 10.2337/dc09-S062.

[2] J. R. Zierath, “Major Advances and Discoveries in Diabetes - 2019 in Review,” *Curr. Diab. Rep.*, vol. 19, no. 11, p. 118, Nov. 2019, doi: 10.1007/s11892-019-1255-x.

[3] H. M. Kharroubi, A. T., & Darwish, “Diabetes mellitus: The epidemic of the century. World journal of diabetes.” pp. 850–867, 2015, doi: https://doi.org/10.4239/wjd.v6.i6.850.

[4] American Diabetes Association, “Diabetes - Symptoms.” https://www.diabetes.org/diabetes/type-1/symptoms (accessed Nov. 06, 2020).

[5] National Institute of Diabetes and Digestive and Kidney Diseases, “Risk Factors for Type 2 Diabetes.” https://www.niddk.nih.gov/health-information/diabetes/overview/risk-factors-type-2-diabetes (accessed Nov. 06, 2020).

**Appendix**

import numpy as np #import the NumPy library

import pandas as pd #import the pandas library

import matplotlib.pyplot as plt #import the matplot library

import seaborn as sns #import seaborn library

dataframe = pd.read\_csv('diabetes2.csv') #read the dataset

dataframe.info() getting information on the data types and number of records in each column

dataframe.describe() #getting a quick description of the data set columns like mean, standard #deviation

sns.countplot(dataframe['Outcome']) #Checking the various possible outcomes and their value #counts

sns.jointplot(x='Pregnancies',y='Age',data=dataframe,kind='hex') #plotting pregnancies against age

#the code below splits the dataset into 3 age categories

def age\_group(age):

if (age < 41):

return 0

elif (age > 40) & (age < 60):

return 1

else:

return 2

dataframe['Age\_Group'] = dataframe['Age'].apply(lambda x: age\_group(x)) # applying the age group function to the dataframe

sns.countplot(x = 'Age\_Group', data = dataframe, hue = 'Outcome') #Counting the age groups in #the different outcomes

#the codes below separates the glucose levels into 3

def glucose\_group(glucose):

if (glucose < 90):

return '0'

elif (glucose > 89) & (glucose < 125):

return '1'

else:

return '2'

dataframe['Glucose\_Level'] = dataframe['Glucose'].apply(lambda x: glucose\_group(x)) #applies the glucose group function to the glucose

sns.countplot(x = 'Glucose\_Level', data = dataframe, hue = 'Outcome') #Counting the glucose #groups in the different outcomes

sns.kdeplot(dataframe[dataframe['Outcome']==0]['BMI'], legend=False) #plotting kernel density #estimation plot for negative outcome

sns.kdeplot(dataframe[dataframe['Outcome']==1]['BMI'], legend=False) #plotting kernel density #estimation plot for positive outcome

plt.xlabel('Outcomes') #labels x axis

plt.title('BMI of the Different Outcomes') #Shows title of graph

sns.pairplot(dataframe) #shows a pair plot of all variables

sns.heatmap(dataframe.isnull(), yticklabels = False, cbar=False, cmap = 'viridis') #shows if empty vare present

A\_G = pd.get\_dummies(dataframe['Age\_Group'], drop\_first = True) #getting dummy variables

df = pd.concat([dataframe, A\_G], axis = 1) #merge the varo\iables with the table

df.drop(['Age', 'Age\_Group'], axis = 1, inplace = True) #drop columns which are not needed

from sklearn.model\_selection import train\_test\_split #import the train test split library from #scikit learn

X = df[['Pregnancies','Glucose','BloodPressure','SkinThickness','Insulin','BMI','DiabetesPedigreeFunction',1,2]] #setting the variables for prediction

y = df['Outcome'] #setting the outcome

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.3, random\_state=101)

#splitting the dataset for training, 70% for train, 30% for test

from sklearn.svm import SVC #import support vector machine library

model = SVC() #initializing the algorithm

model.fit(X\_train, y\_train) #fitting the model

pred\_svm = model.predict(X\_test) #getting predictions

print(confusion\_matrix(y\_test, pred\_svm)) #getting the confusion matrix

print('\n') #next line

print(classification\_report(y\_test, pred\_svm)) #getting the classification report

from sklearn.ensemble import RandomForestClassifier #import random forest library

rfc = RandomForestClassifier(n\_estimators = 200, verbose =100) #initializing the algorithm with #200 estinmators and setting verbose in order to monitor progress

rfc.fit(X\_train, y\_train) #fitting the model

rfc\_pred = rfc.predict(X\_test) #getting predictions

print(confusion\_matrix(y\_test, rfc\_pred)) #getting the confusion matrix

print('\n') #next line

print(classification\_report(y\_test, rfc\_pred)) #getting the classification report

from sklearn.preprocessing import StandardScaler #import standard scaler

scaler = StandardScaler() #initializing the algorithm

scaler.fit(df.drop(['Outcome'], axis = 1)) #dropping the outcome column and fitting the model

scaled\_features = scaler.transform(df.drop('Outcome', axis = 1)) #transfoming the dataset and #getting the scaled features

col = df.columns.drop('Outcome') #getting the column names

df\_feat = pd.DataFrame(scaled\_features, columns=col) #new dataset of scaled data

X = df\_feat #setting the variables for prediction

y = df['Outcome'] #setting the outcome

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.3, random\_state=101)

#splitting the scaled dataset for training, 70% for train, 30% for test

import tensorflow as tf #import TensorFlow library

from keras.models import Sequential #import sequential from keras

from keras.layers import Dense #import dense from keras

model.compile(loss='binary\_crossentropy', optimizer='adam', metrics=['accuracy']) #compile model using binary crossentropy, adam optimizer and setting metrics to accuracy

model.fit(X\_train, y\_train,epochs=20, batch\_size=1, verbose=1) # fitting the model using 20 epochs, I batch and setting verbose to monitor progress

y\_pred=(model.predict(X\_test) > 0.5).astype("int32") #getting predictions with threshold grether than 0.5

print(confusion\_matrix(y\_test, y\_pred)) #getting the confusion matrix

print('\n') #next line

print(classification\_report(y\_test, y\_pred.round())) #getting the classification report