Prediction of Diabetic Using Classification Techniques

PRESENTED BY:-
VIVEK SINGH RAJAWAT
BTECH 3RD YEAR, CSE
SBCET, JAIPUR

ABSTRACT

Data Science encompasses a set of principles, problem definitions, algorithms, and processes for extracting nonobvious and useful patterns from large data sets. Many elements of data science have been developed in related fields such as machine learning and data mining. Data science is a blend of various fields like Probability, statics, programming, analysis, cloud computing, etc; which is used to extract value from the data provided.

- ✓ This project is based on Diabetes value predictions. He we used a dataset of 759 patients in real life. Here we have 9 features in our dataset. This data is taken from those people. So that's why it's real life based project.

- ✓ Diabetes is a type of chronic disease which is more common among the of all age groups. Predicting this disease at an early stage can help necessary precautions and change his/her prevent the occurrence of this disease or control the disease(For people who already have the disease).

TASK TO BE PERFORMED

1. Prepare the data set using several methods to train the model.
2. Build a model which can give high accuracy of predicting the disease.

TABLE OF CONTENTS

CONTENTS **Pg.No.**

INTRODUCTION

The take home challenge problem or coding exercise is the most important step in the data scientist interview process. this is generally a data science problem e.g. machine learning model, logistic regression, naïve Bayes algorithm, knn algorithm etc. Generally, the interview team will provide you with project directions and the dataset.

Some coding challenge problems would specify that a formal project report be submitted together with a Jupyter notebook. This article will provide some guidelines on how to write a formal project report for the take home coding challenge problems.

In this problem, you will forecast the outcome of diabetes patient. We have so many features in our data sets and 700+ patient's details. Diabetes is the fast growing disease among the people even among the youngsters. In understanding diabetes and how it develops, we need to understand what happens in the body without diabetes. Sugar (glucose) comes from thefoods that we eat, specifically carbohydrate foods. Carbohydrate foods provide our body with its main energy source everybody, even those people with diabetes, needs carbohydrate. Carbohydrate foods include bread, cereal, pasta, rice, fruit, dairy products and vegetables (especially starchy vegetables). When we eat these foods, the body breaks them down intoglucose. The glucose moves around the body in the bloodstream. Some of the glucose is taken to our brain to help us think clearly and function. The remainder of the glucose is taken to the cells of our body for energy and also to our liver, where it is stored as energy that is used later by the body. In order for the body to use glucose for energy, insulin is required. Insulin is a hormone that is produced by the beta cells in the pancreas. Insulin works like a key to a door. Insulin attaches itself to doors on the cell, opening the door to allow glucose to move from the blood stream, through the door, and into the cell. If the pancreas is not able to produce enough insulin (insulin deficiency) or if the body cannot use the insulin it produces (insulin resistance), glucose builds up in the bloodstream (hyperglycaemia) and diabetes develops. Diabetes Mellitus means high levels of sugar (glucose) in the blood stream and in the urine.

Types of diabetes :-

Type 1 diabetes means that the immune system is compromised and the cells fail to produce insulin in sufficient amounts. There are no eloquent studies that prove the causes of type 1 diabetes and there are currently no known methods of prevention.

Type 2 diabetes means that the cells produce a low quantity of insulin or the body can't use the insulin correctly. This is the most common type of diabetes, thus affecting 90% of persons diagnosed with diabetes. It is caused by both genetic factors and the manner of living.

Gestational diabetes appears in pregnant women who suddenly develop high blood sugar. In two thirds of the cases, it will reappear during subsequent pregnancies. There is a great chance that type 1 or type 2 diabetes will occur after a pregnancy affected by gestational diabetes.

Symptoms of Diabetes

- Frequent Urination
- Increased thirst
- Tired/Sleepiness
- Weight loss
- Blurred vision
- Mood swings
- Confusion and difficulty concentrating
- frequent infections

Causes of Diabetes Genetic factors are the main cause of diabetes. It is caused by at least two mutant genes in the chromosome 6, the chromosome that affects the response of the body to various antigens. Viral infection may also influence the occurrence of type 1 and type 2 diabetes. Studies have shown that infection with viruses such as rubella, Coxsackievirus, mumps, hepatitis B virus, and cytomegalovirus increase the risk of developing diabetes.

EXISTING METHODS

In this section we shall learn about the various classfiers used in machine learning to predict diabetes. We shall also explain our proposed methodology to improve the accuracy. Five different methods were used in this paper. The different methods used are defined below. The output is the accuracy metrics of the machine learning models. Then, the model can be used in prediction.

1. Logistic Regression
2. Naïve Bayes Algorithm
3. Random Forest Algorithm
4. KNN Algorithm
5. Decision Tree algorithm
6. SVC

METHODOLOGY

First of all we are importing pandas, numpy, seaborn, matplotlib to proceed this project. Then we added the data set "diabetes.csv". In our data sets there are so many datatypes are available in sets.

```
In [1]: import pandas as pd
        import numpy as np
        import seaborn as sns
        import matplotlib.pyplot as plt

        IMPORTING DATA

In [2]: data=pd.read_csv('diabetes.csv')

In [3]: data.dtypes

Out[3]: Pregnancies                 int64
        Glucose                     int64
        BloodPressure               int64
        SkinThickness               int64
        Insulin                     int64
        BMI                       float64
        DiabetesPedigreeFunction  float64
        Age                         int64
        Outcome                     int64
        dtype: object
```

As we can see that in our data set we have 9 features with 2000 data of diabetic patients. Here Dtype we can see int,float. There are 2000 entries (0 to 1999). To see this result , just put data.info().

```
In [4]: data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               2000 non-null   int64
 1   Glucose                   2000 non-null   int64
 2   BloodPressure             2000 non-null   int64
 3   SkinThickness             2000 non-null   int64
 4   Insulin                   2000 non-null   int64
 5   BMI                       2000 non-null   float64
 6   DiabetesPedigreeFunction  2000 non-null   float64
 7   Age                       2000 non-null   int64
 8   Outcome                   2000 non-null   int64
dtypes: float64(2), int64(7)
memory usage: 140.8 KB
```

In this section we can see that there is no any cell is empty. But reality is this there are some patients are not available so there is ‘0’ in our data sets.

```
In [62]: data.isnull().sum()

Out[62]: Pregnancies                 0
         Glucose                     0
         BloodPressure               0
         SkinThickness               0
         Insulin                     0
         BMI                         0
         DiabetesPedigreeFunction    0
         Age                         0
         Outcome                     0
         dtype: int64
```

no null values hence cleaning of data is not requeried

To remove ‘0’ we change it with the mean value of the colume

```
data['Glucose'] = data['Glucose'].replace(0, data['Glucose'].mean())

data['BloodPressure'] = data['BloodPressure'].replace(0, data['BloodPressure'].mean())

data['BMI'] = data['BMI'].replace(0, data['BMI'].median())

data['SkinThickness'] = data['SkinThickness'].replace(0, data['SkinThickness'].median())
data['Insulin'] = data['Insulin'].replace(0, data['Insulin'].median())
```

To see the descriptions of our data sets just we write here data.describe() so that is why we are getting such results where we can completely analysis the data set that we are using.

```
In [10]: data.describe()
```

Out[10]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	2000.000000	2000.000000	2000.000000	2000.00000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000
mean	3.703500	121.970186	72.257047	27.52450	99.374000	32.645200	0.470930	33.090500	0.342000
std	3.306063	30.533180	11.968614	9.56374	98.438245	7.190254	0.323553	11.786423	0.474498
min	0.000000	44.000000	24.000000	7.00000	14.000000	18.200000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	64.000000	23.00000	40.000000	27.600000	0.244000	24.000000	0.000000
50%	3.000000	118.000000	72.000000	23.00000	40.000000	32.300000	0.376000	29.000000	0.000000
75%	6.000000	141.000000	80.000000	32.00000	130.000000	36.800000	0.624000	40.000000	1.000000
max	17.000000	199.000000	122.000000	110.00000	744.000000	80.600000	2.420000	81.000000	1.000000

4. just to see our first 5 data set, we can use here data.head()

```
data.head()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	2	138.0	62.0000	35	40	33.6	0.127	47	1
1	0	84.0	82.0000	31	125	38.2	0.233	23	0
2	0	145.0	69.1455	23	40	44.2	0.630	31	1
3	0	135.0	68.0000	42	250	42.3	0.365	24	1
4	1	139.0	62.0000	41	480	40.7	0.536	21	0

5. if you want to check our average value of our data sets including while we have 9 features. there are around 684 diabetic and 1316 non diabetic patients.

In [13]:
```
print(f"Count of Diabetic pateients    : {diabetic} ")
print(f"Count of Non-diabetic patients : {non_diabetic}")
```

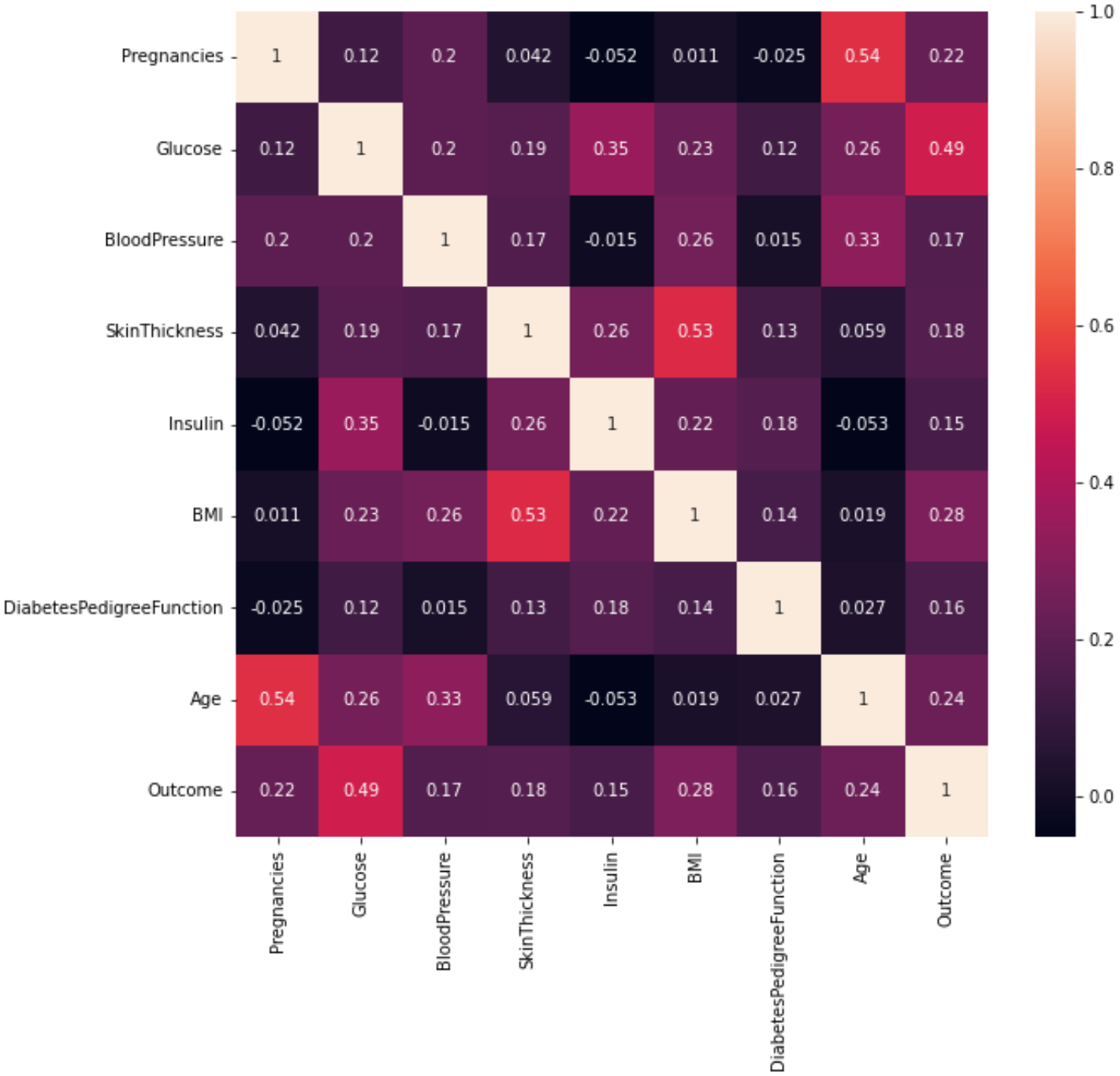
```
Count of Diabetic pateients    : 684
Count of Non-diabetic patients : 1316
```

In [12]:
```
data.corr()
```

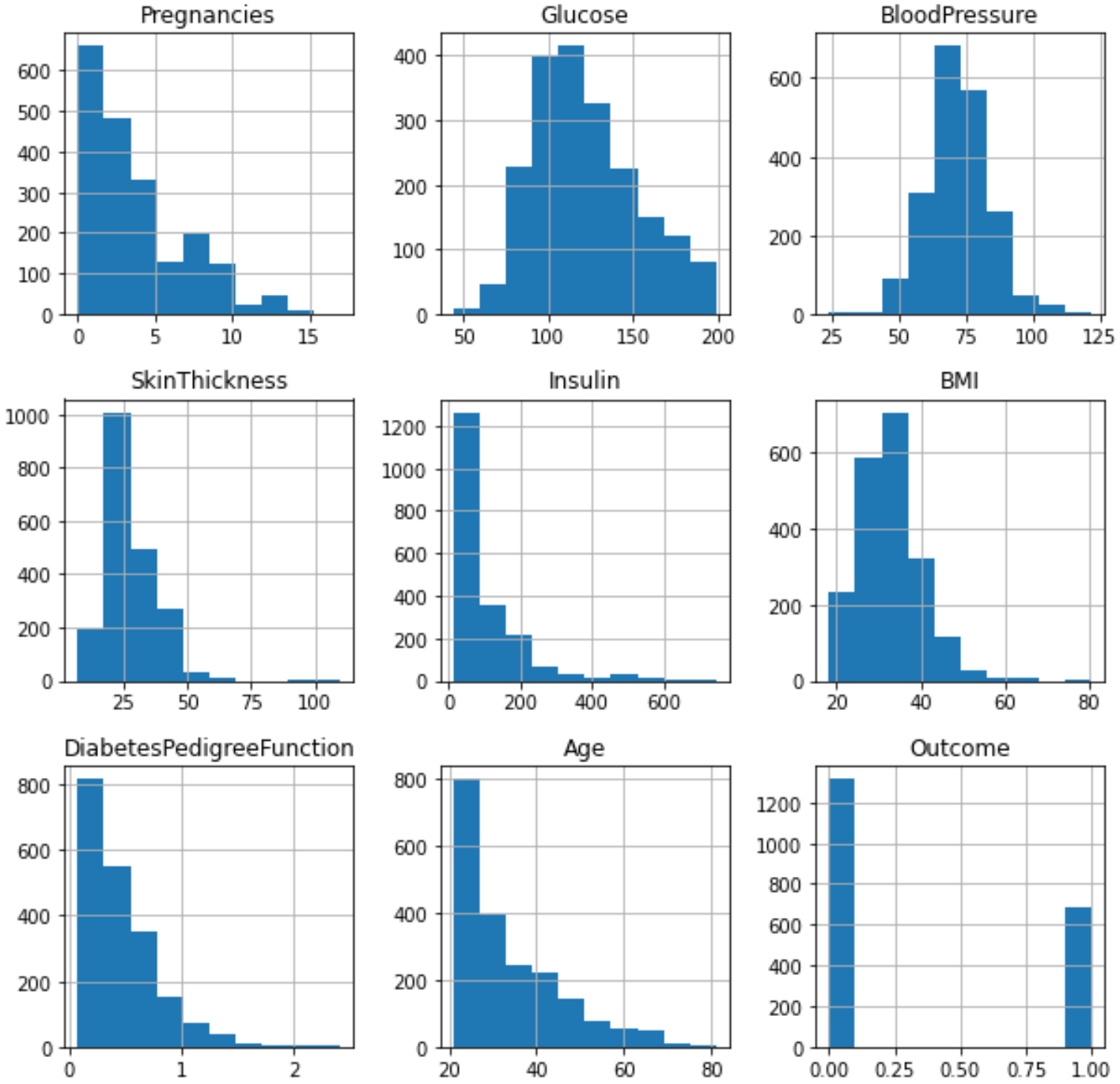
Out[12]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
Pregnancies	1.000000	0.121569	0.199471	0.042450	-0.052486	0.011404	-0.025453	0.539457	0.224437
Glucose	0.121569	1.000000	0.200509	0.185503	0.352656	0.233088	0.124176	0.259853	0.487977
BloodPressure	0.199471	0.200509	1.000000	0.167622	-0.014978	0.259419	0.015216	0.325490	0.171844
SkinThickness	0.042450	0.185503	0.167622	1.000000	0.255889	0.526836	0.133548	0.059255	0.181365
Insulin	-0.052486	0.352656	-0.014978	0.255889	1.000000	0.217600	0.181461	-0.052681	0.145165
BMI	0.011404	0.233088	0.259419	0.526836	0.217600	1.000000	0.142019	0.018731	0.282515
DiabetesPedigreeFunction	-0.025453	0.124176	0.015216	0.133548	0.181461	0.142019	1.000000	0.026569	0.155459
Age	0.539457	0.259853	0.325490	0.059255	-0.052681	0.018731	0.026569	1.000000	0.236509
Outcome	0.224437	0.487977	0.171844	0.181365	0.145165	0.282515	0.155459	0.236509	1.000000

In [13]:
```
plt.figure(figsize=(10,9))
sns.heatmap(data.corr(),annot=True)
```


Pregnancies 1 0.12 0.2 0.042 -0.052 0.011 -0.025 0.54 0.22
Glucose 0.12 1 0.2 0.19 0.35 0.23 0.12 0.26 0.49
BloodPressure 0.2 0.2 1 0.17 -0.015 0.26 0.015 0.33 0.17
SkinThickness 0.042 0.19 0.17 1 0.26 0.53 0.13 0.059 0.18
Insulin -0.052 0.35 -0.015 0.26 1 0.22 0.18 -0.053 0.15
BMI 0.011 0.23 0.26 0.53 0.22 1 0.14 0.019 0.28
DiabetesPedigreeFunction -0.025 0.12 0.015 0.13 0.18 0.14 1 0.027 0.16
Age 0.54 0.26 0.33 0.059 -0.053 0.019 0.027 1 0.24
Outcome 0.22 0.49 0.17 0.18 0.15 0.28 0.16 0.24 1
Pregnancies
Glucose
BloodPressure
SkinThickness
Insulin
BMI
DiabetesPedigreeFunction
Age
Outcome
1.0
0.8
0.6
0.4
0.2
0.0

This is our graphical representations of our data sets of 9 features. there we can see our view of bar charts.



Now we are using several methods to get the accuracy of our data sets. that is why we can identify very high accuracy and we can find our diabetes patients.

1. Logictics Regression

```
In [21]: from sklearn.linear_model import LogisticRegression
         LoR=LogisticRegression()

In [22]: LoR.fit(x_train,y_train)

Out[22]: LogisticRegression()

In [23]: y_pred=LoR.predict(x_test)

In [24]: #Compute classification Accuracy for Logistic Regression
         from sklearn import metrics
         print(f"Logistic regression Accuracy:{(metrics.accuracy_score(y_test,y_pred))*100}")

         Logistic regression Accuracy:78.25

In [25]: miss_classify=(y_test!=y_pred).sum()
         print(f"Missclassified Sample by Logistic algorithm:{miss_classify}")
         print(f"Correctly classified Sample:{(y_test==y_pred).sum()}")

         Missclassified Sample by Logistic algorithm:87
         Correctly classified Sample:313
```

In logicstic regression we are getting accuracy 78.25%.

2. Naïve Bayes Algorithm (Gaussian Algorithm)

```
         2---->Naive Bayes Algorithm (Gaussian Algorithm)

In [27]: from sklearn.naive_bayes import GaussianNB

In [28]: GaNB= GaussianNB()

In [29]: # fit the model with the training data
         GaNB.fit(x_train,y_train)

Out[29]: GaussianNB()

In [30]: y_pred = GaNB.predict(x_test)

In [31]: print(f"Gaussian Naive Bayes Algorithm Accuracy:{(metrics.accuracy_score(y_test,y_pred))*100}")

         Gaussian Naive Bayes Algorithm Accuracy:76.0

In [32]: miss_classify=(y_test!=y_pred).sum()
         print(f"Missclassified Sample by Gaussian NB algorithm:{miss_classify}")
         print(f"Correctly classified Sample:{(y_test==y_pred).sum()}")

         Missclassified Sample by Gaussian NB algorithm:96
         Correctly classified Sample:304
```

In Naïve Bayes algorithm we are getting our accuracy 76.0%

3. KNN Algorithm

3--->KNN Alogrithem

In [33]:
```
from sklearn.neighbors import KNeighborsClassifier
```

In [34]:
```
knn=KNeighborsClassifier(n_neighbors=1)
```

In [35]:
```
knn.fit(x_train,y_train)
```

Out[35]:
```
KNeighborsClassifier(n_neighbors=1)
```

In [36]:
```
y_pred=knn.predict(x_test)
```

In [37]:
```
print(f"KNN Algorithm Accuracy (when kneighbors=1):{(metrics.accuracy_score(y_test,y_pred))*100}")
```

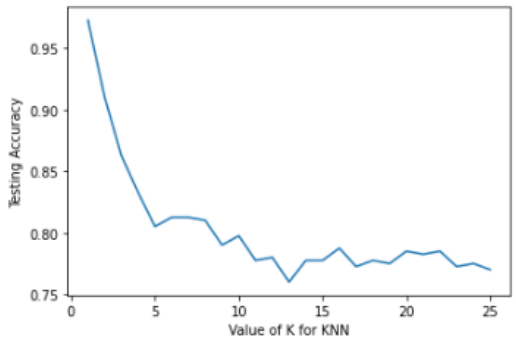
```
KNN Algorithm Accuracy (when kneighbors=1):97.25
```

In [38]:
```
k_range = list(range(1, 26))
scores = []
for k in k_range:
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(x_train,y_train)
    y_pred = knn.predict(x_test)
    scores.append(metrics.accuracy_score(y_test, y_pred))
```

In [39]:
```
# allow plots to appear within the notebook
%matplotlib inline

# plot the relationship between K and testing accuracy
plt.plot(k_range, scores)
plt.xlabel('Value of K for KNN')
plt.ylabel('Testing Accuracy')
```

Out[39]:
```
Text(0, 0.5, 'Testing Accuracy')
```



Highest test accuracy is when kneighbors=1

We are getting accuracy 97.25% through KNN algorithm. when kneighbours = 1.

4. Random Forest Algorithm

4--->Random forest algorithem

```
In [42]: from sklearn.ensemble import RandomForestClassifier

         rfc = RandomForestClassifier(random_state=4219,n_estimators=200)
         rfc.fit(x_train, y_train)
Out[42]: RandomForestClassifier(n_estimators=200, random_state=4219)

In [43]: rfc_train = rfc.predict(x_train)

In [44]: y_pred=rfc.predict(x_test)

In [45]: from sklearn import metrics
         print(f"random forest Accuracy:{(metrics.accuracy_score(y_test,y_pred))*100}")
         random forest Accuracy:99.0
```

Through Random forest algorithm we are getting 99.0% accuracy.

5. Decision tree

5--->Decision tree

```
In [46]: from sklearn.tree import DecisionTreeClassifier

         dtree = DecisionTreeClassifier()
         dtree.fit(x_train, y_train)
Out[46]: DecisionTreeClassifier()

In [47]: dtree_train = dtree.predict(x_train)

In [48]: y_pred=dtree.predict(x_test)

In [49]: from sklearn import metrics
         print(f"Desision Tree Accuracy:{(metrics.accuracy_score(y_test,y_pred))*100}")
         Desision Tree Accuracy:98.0
```

Decision tree we are getting 98.0% accuracy

6. SVC

6--->svc

In [50]:
```
from sklearn.svm import SVC

svc_model = SVC()
svc_model.fit(x_train, y_train)
```

Out[50]:
```
SVC()
```

In [51]:
```
y_pred=svc_model.predict(x_test)
svc_pred = svc_model.predict(x_test)
```

In [52]:
```
from sklearn import metrics
print(f"svc  Accuracy:{(metrics.accuracy_score(y_test,y_pred))*100}")
```

```
svc  Accuracy:79.0
```

Through SVC we are getting 79.0% accuracy

IMPLEMENTATION & CONCLUSION

Algorithm	Accuracy
Logistic Regression	78.25%
Naïve Bayes Algorithm	76.0%
KNN (n=1)Alogrithem	97.25%
Random forest	99.0%
Decision tree	98.0%
SVM	79.0%

The highest accuracy we are getting through Random forest algorithm that is 99.0% and that’s good to use it.

FUTURE SCOPE

One of the important real-world medical problems is the detection of diabetes at its early stage. In this study, systematic efforts are made in designing a system which results in the prediction of diabetes. During this work, five machine learning classification algorithms are studied and evaluated on various measures. Experiments are performed on john Diabetes Database. Experimental results determine the adequacy of the designed system with an achieved accuracy of 99% using Decision Tree algorithm.

In future, the designed system with the used machine learning classification algorithms can be used to predict or diagnose other diseases. The work can be extended and improved for the automation of diabetes analysis including some other machine learning algorithms.

REFERENCES

[1]. Aljumah, A.A., Ahamad, M.G., Siddiqui, M.K., 2013. Application of data mining: Diabetes health care in young and old patients. Journal of King Saud University - Computer and Information Sciences 25, 127–136. doi:10.1016/j.jksuci.2012.10.003.

[2]. Arora, R., Suman, 2012. Comparative Analysis of Classification Algorithms on Different Datasets using WEKA. International Journal of Computer Applications 54, 21–25. doi:10.5120/8626-2492.

[3]. Bamnote, M.P., G.R., 2014. Design of Classifier for Detection of Diabetes Mellitus Using Genetic Programming. Advances in Intelligent Systems and Computing 1, 763–770. doi:10.1007/978-3-319-11933-5.

[4]. Choubey, D.K., Paul, S., Kumar, S., Kumar, S., 2017. Classification of Pima indian diabetes dataset using naive bayes with genetic algorithm as an attribute selection, in: Communication and Computing Systems: Proceedings of the International Conference on Communication and Computing System (ICCCS 2016), pp. 451–455.

[5]. Dhomse Kanchan B., M.K.M., 2016. Study of Machine Learning Algorithms for Special Disease Prediction using Principal of Component Analysis, in: 2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication, IEEE. pp. 5–10.

[6]. Sharief, A.A., Sheta, A., 2014. Developing a Mathematical Model to Detect Diabetes Using Multigene Genetic Programming. International Journal of Advanced Research in Artificial Intelligence (IJARAI) 3, 54–59. doi:doi:10.14569/IJARAI.2014.031007.

[7]. Sisodia, D., Shrivastava, S.K., Jain, R.C., 2010. ISVM for face recognition. Proceedings - 2010 International Conference on Computational Intelligence and Communication Networks, CICN 2010 , 554–559doi:10.1109/CICN.2010.109.

[8]. Sisodia, D., Singh, L., Sisodia, S., 2014. Fast and Accurate Face Recognition Using SVM and DCT, in: Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS 2012), December 28-30, 2012, Springer. pp. 1027–1038.

[9]. https://www.kaggle.com/johndasilva/diabetes

[10]. Rani, A. S., & Jyothi, S. (2016, March). Performance analysis of classification algorithms under different datasets. In Computing for Sustainable Global Development (INDIACom), 2016 3rd International Conference on (pp. 1584- 1589). IEEE.