**ABSTRACT**

A dataset look good from outside but had a lot of discrepancies in it. While extracting the useful set of data from a big set of data, dataset that suffers from imbalanced class distributions creates problem in practical working. In this paper, there are some classifications of the solutions, the process to carry out those solutions; and had discussed some problems related to data intrinsic characteristics which have to kept in front of eyes to provide better solutions for the correct identification of minority and majority classes that have to be balanced.

In order to find the problem in a huge dataset, some techniques are applied practically and made sure that if problem exists then it gets caught by that technique. [1]Problems like identification of areas with small disjuncts, lack of density in data, overlapping between the classes, impact of noisy data, the borderline examples, and dataset shift problem; have been taken into account, and some techniques like addition, subtraction, mean-variance method, multi-dimension array, percentages; are applied to catch these problem; then solutions to those problem are being discussed to remove them.

**INTRODUCTION**

A large dataset may contain some patterns or may have some anomalies that need to be find out to reach at a certain outcome. Thus, a process called data mining is used to observe the patterns in the whole large set(s) of data to get a desired result. Data mining allows us to understand the large un-structured data and get the relevant information out of it and make a good use of that extracted data.

Finding a consistent pattern and that too in a short span of time in a big data using data mining may be effective but there are some problems in data mining. For instance,

1. Noisy data, missing value or incorrect value degrades the quality of data.
2. Unavailability of data or access of data.
3. New data that gets uploaded time to time.
4. Dealing with unbalanced data.
5. Difficulties in storing large amount of data.
6. New and different approaches are needed to deal with huge data.
7. Rapid increase in the number of security and privacy concerns by the people related with data.

Now, looking at the above problems, it becomes crucial to some organizations to handle these problems and extract the useful and important information out of it. Thus some existing techniques that are being used in current time are:

1. Association Technique: This technique helps in finding out the followed pattern in the huge data, which is based on some relation between two or more objects or help in analyzing the common habit or ritual followed by data.
2. Classification Technique: This technique uses mathematical techniques like statistics, and observes the behavior in order to predict the unknown records and look the future behavior of that data.
3. Clustering Technique: This technique makes clusters or groups of data which show same behavior or which share common characteristics, and then identify the differences and similarities between the data.
4. Sequential Patterns: This technique helps in identifying the trends and pattern followed by the data and is then able to give suggestion about what will happen in future.
5. Anomaly Detection: Anomalies, also known as noise, deviations or exceptions; are the data items that do not follow an expected pattern and thus requires an immediate attention. Using this technique, we can remove the anomalous data from our dataset and increase the accuracy of data.

Thus, these techniques are being applied to large data sets in the real world problem to analyze the various patterns and trends followed by the data, so that the accuracy of data can be increased by removing the anomalies from data and predict the future work or behavior of the data in order to reach at our desired result.

**RESULTS AND DISCUSSIONS**

Now we have to carry out some techniques to check and resolve the problems hidden in the dataset. Thus, let’s take a dataset, say, the dataset based on the “health – related with the values on diabetes” and these values are been monitored so that the classes can be build to differentiate the positive and negative classes.

The problems that are occurring in the data, and the way they can be resolved are shown below:

1. **Problem of Small Disjuncts**

The problem of small disjuncts occurs in the database when there are formations of small groups of data which contains value that are more towards disturbing the accuracy of the whole dataset. The error-prone small disjuncts covers only a small fraction of the data points, but collectively, they cover a large portion of the dataset. These small groups are needed to be removed from the large dataset as they have more errors than the large disjuncts. Thus, it becomes important to remove such small groups of data to increase the accuracy of dataset. A technique has been discussed (Figure 1. and Figure 2.) to resolve such problem.

Open file

FT = Scan file

Close file

Make two arrays

Intialize count1 and count2 to 0

Intialize a and b to 1 as row variables

Initialize i to 1

while i not equal to 816

set val equal to FT{1,4}(i)

if val is less than 110

put value in array1 and increment a by 1

Increment count1 by 1

else

put value in array2 and increment b by 1

Increment count2 by 1

end if

Increment i by 1

end while

Calculate difference by subtracting count1 from count2

Check the percentage and must be atleast 60% of count1

if difference is greater than percentage

Balance data by adding 111 to array1

Increment and decrement the count1 and count2 respectively

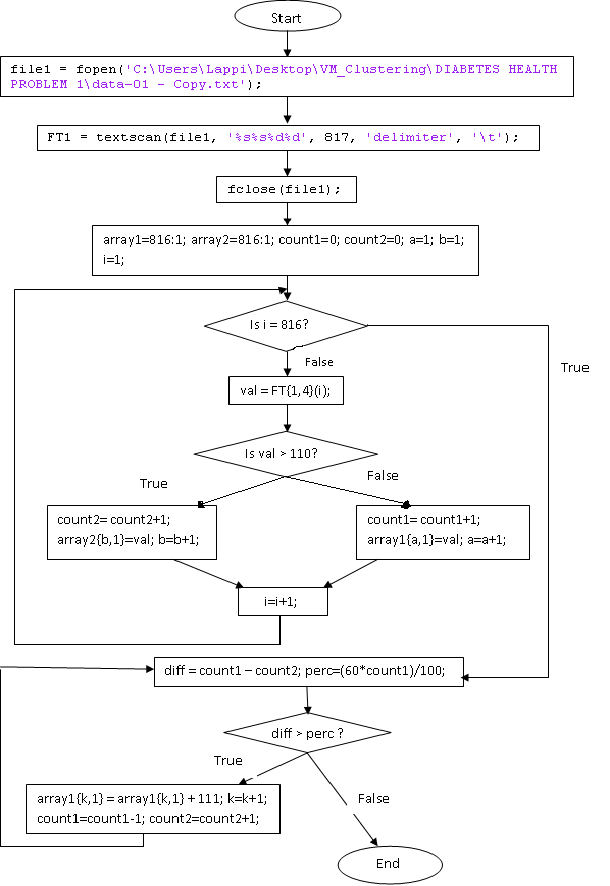
Calculate difference by subtracting count1 from count2

Check the percentage and must be atleast 60% of count1

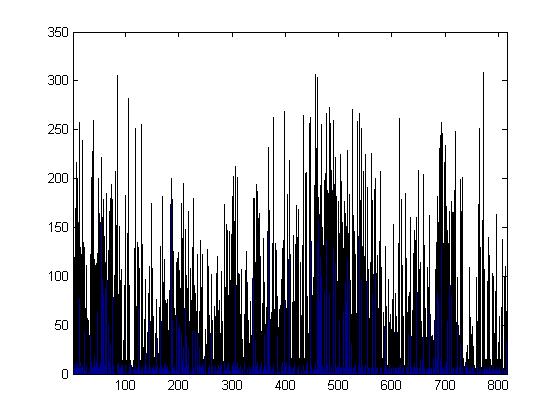
Break when percentage is greater than difference

end if

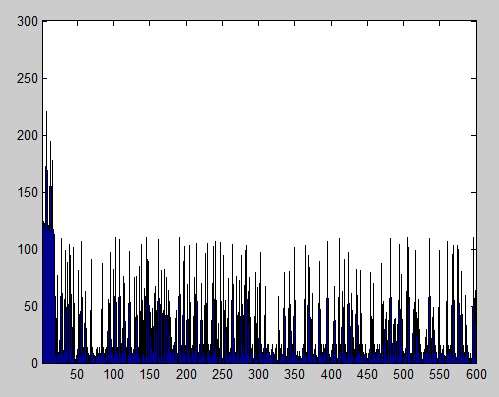
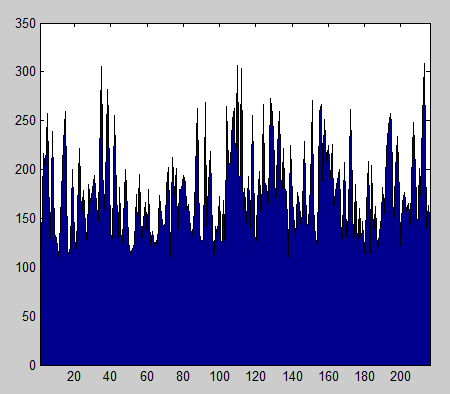
**Figure 1.** Pseudo code of the small disjuncts problem algorithm



**Figure 2.** Flowchart of the small disjuncts problem algorithm



**Picture 1.** Data before balancing

**Picture 2.** Data after balancing

1. **Problem of Lack of Density**

The problem of lack of density occurs when there is “lack of density” or “lack of information” where the techniques that are being implied are not able to classify classes to reach at a certain outcome. Thus, it becomes necessary to increase the density of that particular region where the density of data is less in order to achieve a desired result. Thus a technique can be applied as shown in Figure 3. and Figure 4.

Open file

FT = Scan file

Close file

Initialize i to 1

Initialize sum to 0

Initialize varsum to 0

Initilize sum2 to 0

while i not equal to 800

update sum by adding the value from file

increment i by 1

end while

calculate mean

calculate variance for value ranging between 110 to 150 in file

calculate stndev i.e standard deviation

set i to 1

while i not equal to 800

add stndev to value FT{1,4}(i) in between 110 to 150

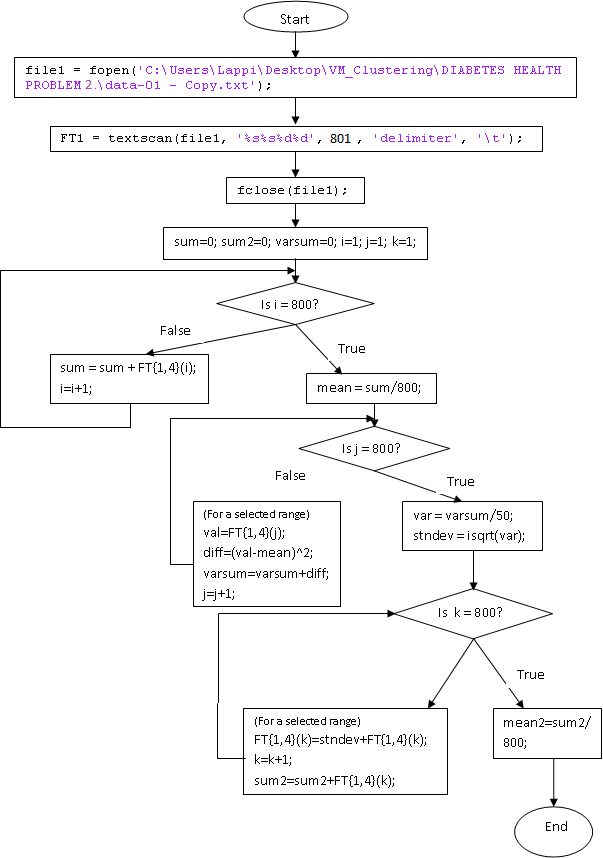
increment i

end while

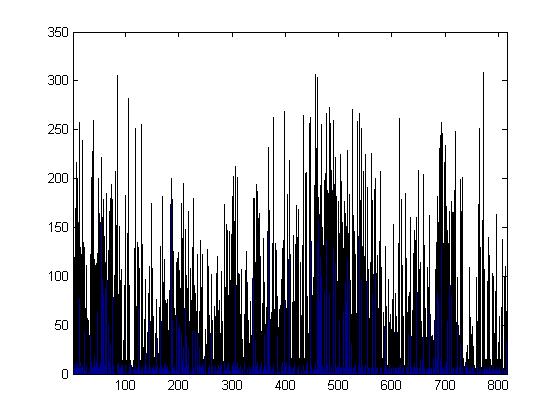
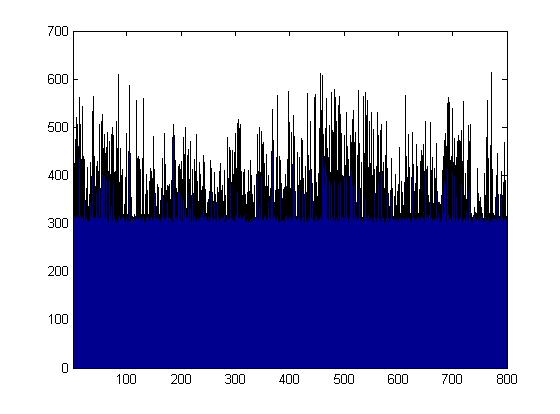
calculate sum2 by adding new value

Calculate the mean2

**Figure 3.** Pseudo code of the lack of density problem algorithm

****

**Figure 4.** Flowchart of the lack of density problem algorithm

** **

**Picture 3.** Data before balancing  **Picture 4.** Data after balancing

1. **Problem of Overlapping classes**

The problem of overlapping classes occurs when there is a condition where a data value or range lies in both sides of two different classes. In such a situation, it becomes difficult to decide whether to put that data value in positive class or in negative class. Thus, a technique is to be carried out (Figure 5. and Figure 6.) so that overlapping of data can be decreased and our data become more accurate.

Open file

FT = Scan file

Close file

Make four arrays

Initialize two count variables to 0, say c1,c2

Initialize four row variables to zero, say a,b,c,d

Initialize i to 1

Initialize j and k to 1

while i not equal to 800

store FT{1,4}(i) in array1{a,1}

increment a by 1

increment i by 1

end while

Set i to 1

while i not equal to 800

if array1{i,1} is greater than or equal to 110 and is less than or equal to 115

store value in array2

increment b by 1

increment count1 by 1

end if

if array1{i,1} is greater than or equal to 130 and is less than or equal to 135

store value in array3

increment c by 1

increment count2 by 1

end if

end while

Set b and c equal to 1

if count1 is greater than count2

while j is not equal to count1

Initialize a random variable ranging between 1 to 10

Add or subtract accordingly to array2 and store in array4

increment the row variable of array4 by 1

end while

else

while k is not equal to count2

Initialize a random variable ranging between 1 to 10

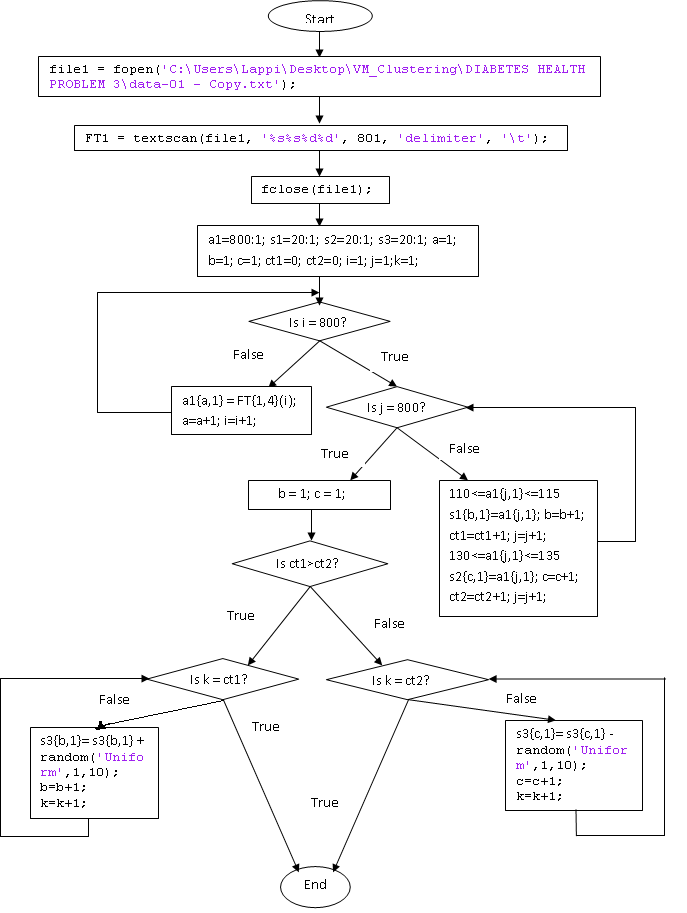
Add or subtract accordingly to array3 and store in array4

Increment the row variable of array4 by 1

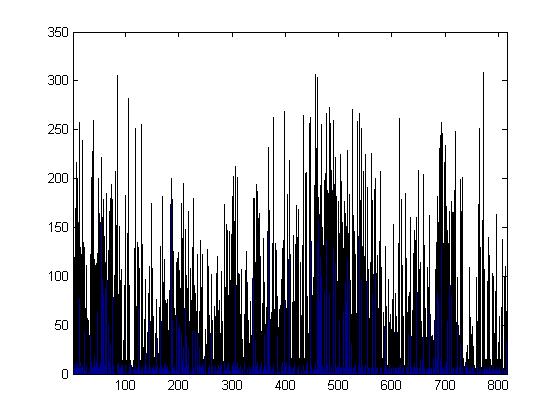
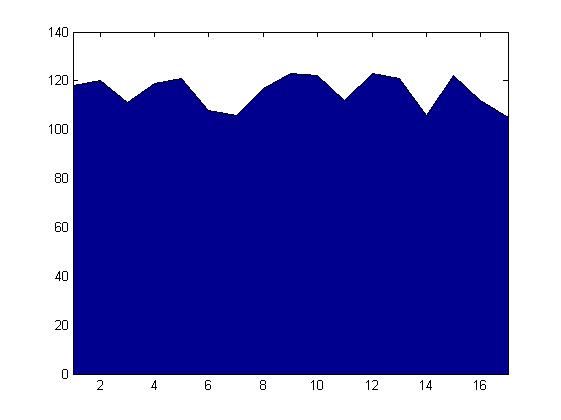
end while

end if

**Figure 5.** Pseudo code of the overlapping problem algorithm



**Figure 6.** Flowchart of the Overlapping problem algorithm

**Picture 5.** Data before balancing  **Picture 6.** Data after balancing

1. **Problem of Noisy Data**

Noisy data is data which contain meaningless information or which have values that contribute nothing to reach at a certain outcome. Such type of corrupt data is difficult to understand and it becomes difficult to interpret it correctly. The problem of noisy data can be sorted by seeing them as small disjuncts in the dataset. A technique (Figure 7.and Figure 8.) can be applied where all the noisy data can be brought together and combined efficiently so that problem can be resolved.

Open file

FT = scan file

Close file

Make two arrays

Initialize b and c to 1 as row variables for two arrays

Initialize i,j and k to 1

while i is not equal to 800

set val to FT{1,4}(i)

if 110<=val=120

Store val in array1 and increment b by 1

end if

if 130<=val=140

Store val in array2 and increment c by 1

end if

Increment i by 1

end while

while j is not equal to (b-1)

add a random number ranging between 1 to 11 in array1

increment j

end while

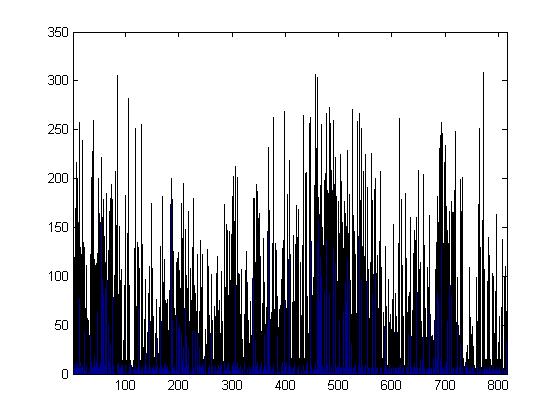
while k is not equal to (c-1)

add a random number ranging between 1 to 11 in array2

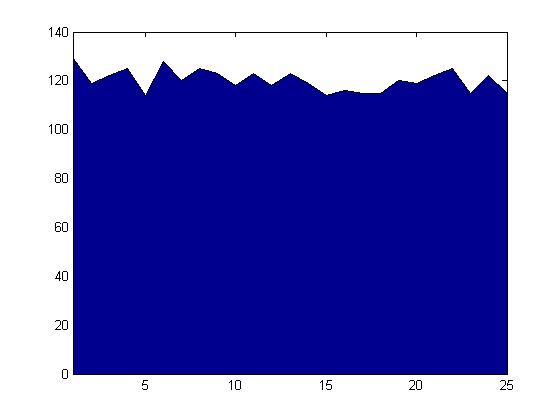
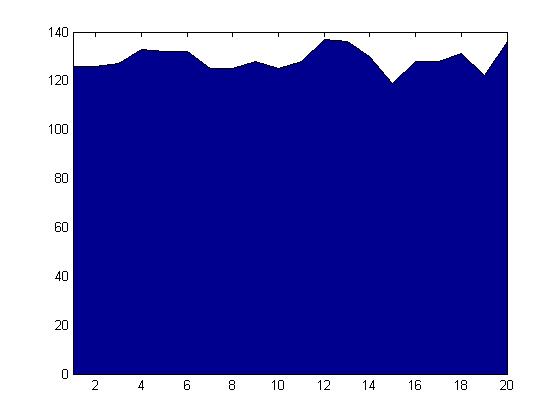
increment k

end while

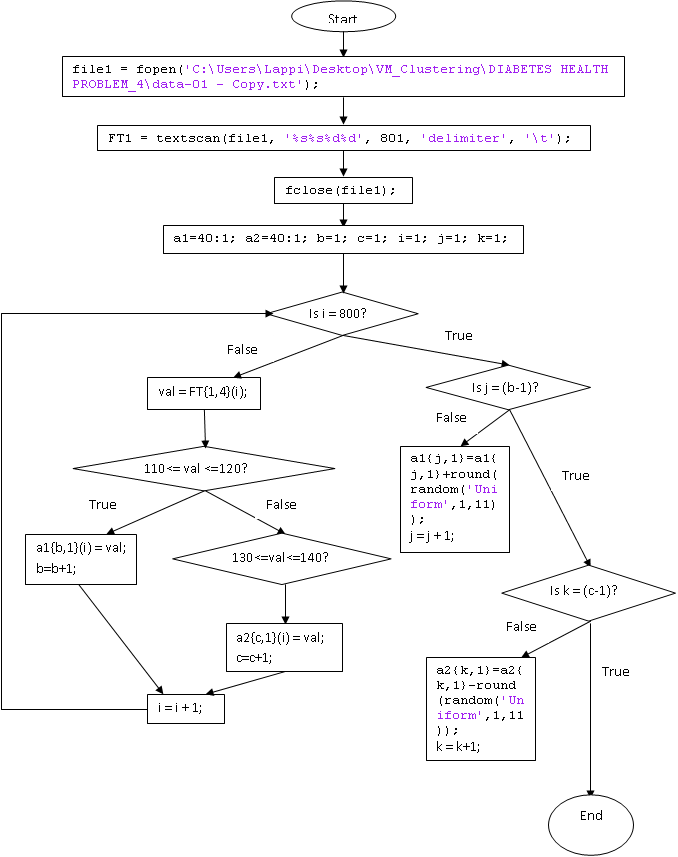
**Figure 7.** Pseudo code of the noisy data problem algorithm

****

**Picture 7.** Data before balancing

** **

**Picture 8. (a.)** Data after balancing for range [110,120] **(b.)** Data after balancing for range [130,140]



**Figure 8.** Flowchart of the noisy data problem algorithm

1. **Problem of Borderline instances**

Borderline instances are the instances which are located in the area surrounding class boundaries, where the positive and negative classes overlap. Few past series of experiments had shown that these borderline examples degrade the overall performance. Thus, we must stress upon this borderline examples and improve our dataset using technique as mentioned below (Figure 9. and Figure 10.).

Open file

FT = Scan file

Close file

Make four arrays

Initialize two count variables to 0, say c1,c2

Initialize four row variables to zero, say a,b,c,d

Initialize i to 1

Initialize j and k to 1

while i not equal to 800

store FT{1,4}(i) in array1{a,1}

increment a by 1

increment i by 1

end while

Set i to 1

while i not equal to 800

if array1{i,1} is greater than or equal to 110 and is less than or equal to 115

store value in array2

increment b by 1

increment count1 by 1

end if

if array1{i,1} is greater than or equal to 130 and is less than or equal to 135

store value in array3

increment c by 1

increment count2 by 1

end if

end while

Set b and c equal to 1

if count1 is greater than count2

while j is not equal to count1

Initialize a random variable ranging between 1 to 10

Add or subtract accordingly to array2 and store in array4

increment the row variable of array4 by 1

end while

else

while k is not equal to count2

Initialize a random variable ranging between 1 to 10

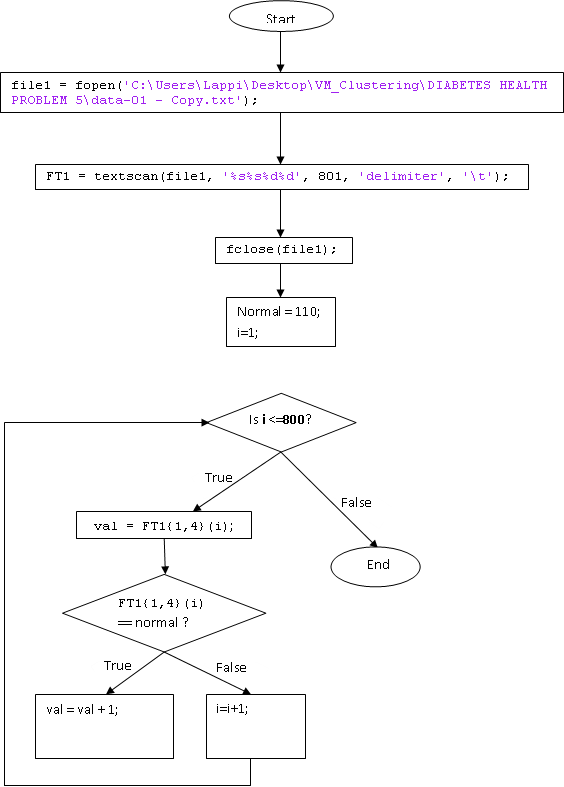
Add or subtract accordingly to array3 and store in array4

increment the row variable of array4 by 1

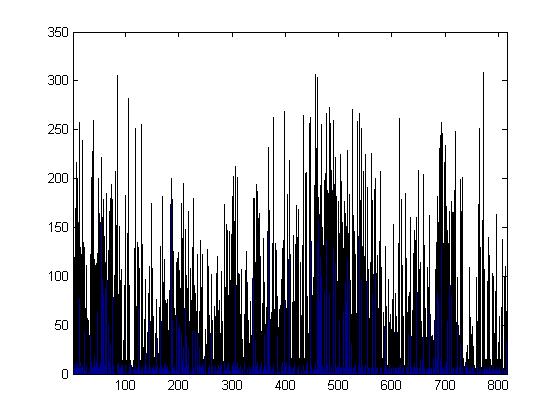
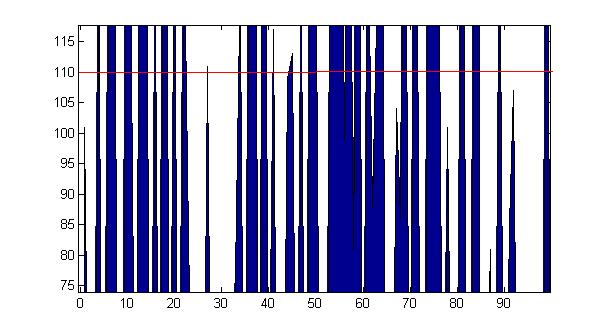
end while

end if

**Figure 9.** Pseudo code of the borderline instance problem algorithm

****

**Figure 10.** Flowchart of the borderline instances problem algorithm

**Picture 9.** Data before balancing **Picture 10.** Data after balancing

1. **Problem of Dataset Shift**

In most of the real world problems, there is a mild shift of dataset present in it. The problem of dataset shift occurs where the training and test data follow different distributions. Thus, such type of problem is needed to be removed by using a suitable technique (Figure 11. and Figure 12.), which can shift the data a bit from its original value.

Open file

Ft = Scan file

Close file

Make an array a of size 40

Initialize i to 1

Initialize j to 1

Initialize row to 1

While i is less than or equal to 801

Set val equal to Ft{1,4}(i);

If val is greater than equal to 105 and less than equal to 115

Put val in array and increment row variable by 1

End if

Increment i by 1

End while

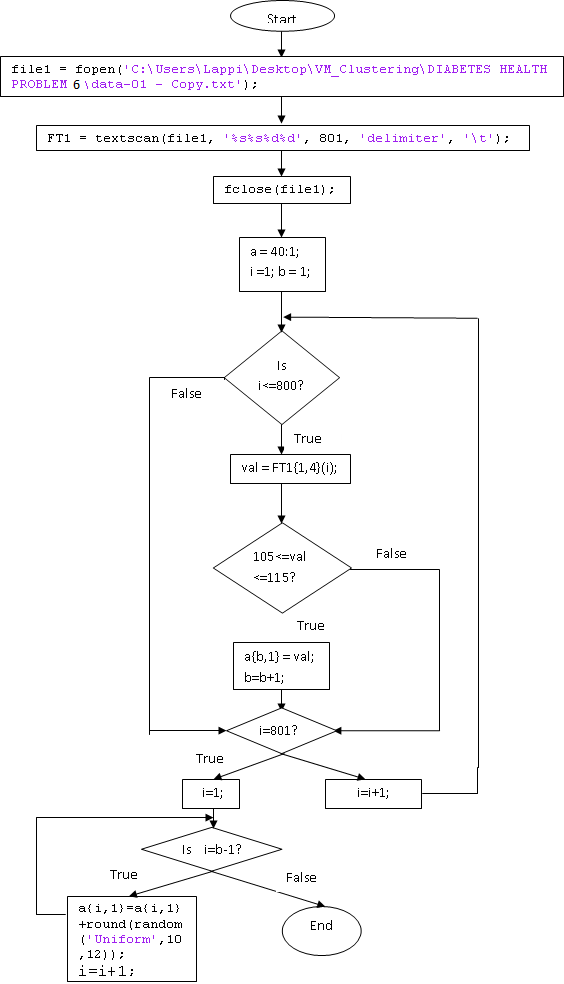
While j is less than equal to (row-1)

Add a random variable between 10 to 12 in the val stored in array

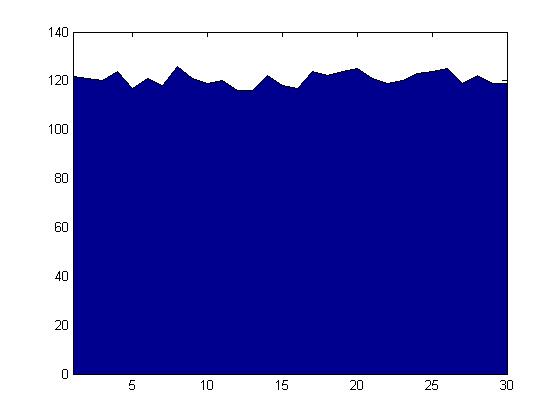
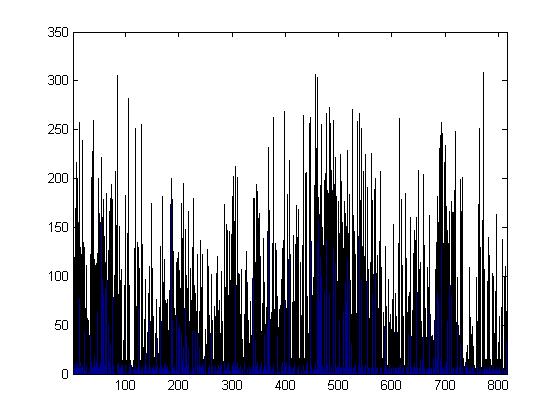
Increment j by 1

End while

**Figure 11.** Pseudo code of the dataset shift problem algorithm



**Figure 12.** Flowchart of dataset shift problem algorithm

****

**Picture 11.** Data before balancing **Picture 12.** Data after balancing

**CONCLUSION**

In this paper, we have discussed about: (1) the various discrepancies’ present in the dataset, (2) the effect of data intrinsic characteristics, and (3) discussions on the problems caused by these problems and how to solve them.

For every mentioned problem, there is a specific technique applied to handle that problem and make the dataset perfect. This paper emphasizes that currently there is a need to study more about the intrinsic characteristics of the data, so that in future researches on classification with imbalanced data, researchers should focus more on detecting and measuring the most significant data properties, in order to be able to give more and better solutions to overcome the problems.

**FUTURE WORK**

Most common situations that can arise in a large raw database have been discussed earlier in the paper. Few conditions that can arise in real world and which have not been discussed are:

(1.) Data which have high percentage of incorrect data as compared to correct data. For instance: In case of data based on “diabetes”, data may state that 90% people fall in not normal category and only 10% people are normal.

(2.) Data may become more imbalanced when combined with another data of same type. For instance: In case of data based on “international flights”, after solving the problems of one country, on combining with other country’s flight data, data may get more imbalanced.

(3.) In case of data which keeps on changing from time to time. For instance: In case of data related with “stocks”, the value keeps on changing every time.

**REFERENCES**

[1] An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics by Victoria Lopez, Alberto Fernandez, Salvador Garcia, Vasile Palade, Francisco Herrera