CHAPTER IV

BUCKETIZATION BASED FLOW CLASSIFICATION ALGORITHM

4.1 INTRODUCTION

The data mining plays a major role in maintaining the huge volume of data from which it can derive the useful information. With the huge number of formation of data, the data wants to be structured in a limit to the charge of growth. This is complex to get over the set of meaningful information from the continuous set of data. Data stream mining is a method which can discover important information from a huge contract of prehistoric data. For identification of useful information, the classification of continuous data streams is done. Current approaches in classifying the data streams are processed using supervised learning algorithms, which can be qualified with tagged data. The manual classification of data is both expensive and time consuming. As a result, where massive amount of data emerge at a high speed, tagged data might be very sparse. Therefore, only a restricted amount of training data might be accessible for constructing the classification models, tend to badly trained classifiers.

This chapter proposes a novel bucketization algorithm to build a classification set with both unlabeled and a small amount of labeled instances. Bucketization algorithm is designed by using the Flow Classification (FC) algorithm. The FC algorithm is able to judge internally on set of marked data. Before classification, the correlation set of attribute in the each record set are grouped using bucketization algorithm to enhance further correlation.
The superiority of models updated from them is enough for utilization of unlabeled records, or whether more set of labeled records are needed for classification is processed. Experimental evaluation is conducted to test the performance of proposed bucketization algorithm with FC algorithm over its counterparts to find a set of diverse solution in terms of classification accuracy, execution time and classification error rate.

4.2 CLASSIFICATION OF DATA STREAMS IN PRIVACY PRESERVATION

Data mining reached a significant place with the advantage of offering large quantity of data maintenance and the crucial to haul out practical information from unrefined data. Numerous useful data prototypes can be chosen out, which assists expect effects of extraordinary scenarios. The information expanded from data mining can also be consequently utilized for diverse applications sorting from business organization to medicinal diagnosis. With process to data mining, main steps of Knowledge Discovery in Databases (KDD) also comprise data cleaning, combination, collection, renovation, prototype evaluation, and knowledge production. Since data is normally speeded with missing values and noise, which creates them disjointed. Data preprocessing has consequently turn out to be significant step before data mining to progress the quality of the data.

Data stream holds diverse uniqueness of data compilation to the conventional database representation. The appointment of data stream incessant creation with time progresses is the active member of data stream. Active data stream is the appearance of the data stream that is not proscribed by the array. The data of data stream can be interpreted and route supported with the classification of arrival. The order of data
cannot be misused to progress the outcome of conduct. Consequently, the processing of the data stream needs properties such as

- Every data element should be scrutinized at most one time, since it is idealistic to maintain the whole stream in the central memory.
- Every data element in data streams should be routed as fast as achievable.
- The memory procedure for mining data streams should be enclosed even though new fangled data elements are constantly produced.
- The results produced by the online algorithms should be immediately accessible when user demanded.

With these properties, the data are classified to provide trustworthy in the correlated data streams. The active data stream in data set is extracted in order to form array structure facilitating the process of classification.

*Stream Data Classification*

Stream data classification is a demanding crisis as of two significant possessions namely infinite span and developing nature. Data streams might develop in numerous methods like the prior possibility allocation \( p(c) \) of a class \( c \) might vary, or the subsequent prospect allocation \( p(c|x) \) of the class might change, or both the prior and posterior possibilities might change. In either case, confront is to build a classification representation that is steady with the present notion. Conventional learning algorithms need a number of passes on the training data to openly operate with the streaming situation, as the number of training instances would be infinite.

Manual cataloging of data is regularly expensive and time consuming. In a streaming atmosphere, where data emerge at a high speed, it might not be probable to physically tag all the data once they turn up. As a result, in practice, only a minute portion of each data chunk is expected to be tagged, leaving a main piece of the chunk
as unlabeled. A very restricted quantity of training data will be accessible for the administered learning algorithms.

Researchers do employ various models to improve the data correlation. Published table is one such model incorporated to improve the stream data classification. Published table is an effect of a data publishing algorithm that terminates when an isolation representation is correctly pleased or not. The privacy revelation devises a data publishing algorithm which is disparate by algorithm in the structure of revelation. Eliminate the algorithm in the outline revelation to method and practical safe algorithm for privacy preserving data distribution. Exclusion of algorithm in the structure of disclosure guide a high level of value saved for the available table. Algorithm safe data publishing is formed based on the concepts of published table that commenced to formalizes algorithm in the outline of revelation removal. The work needs additional evaluation in updating the published table. Hence, bucketized data are much smart idea that is available in utilizing adaptable publishing system. The proposed research intends to perform classification after bucketization of data streams.

Classification Methods in Privacy Preservation

It is observed huge volumes of data to be composed on a huge size. Determined by common benefits, or by policies that need definite data to be available, there is a command for distributing unruffled data to the public. Data publishing has engendered much distress on individual privacy. Current work has revealed that diverse setting knowledge can carry a variety of threats to the seclusion of available data. The author Hui Wang et al., considered the privacy intimidation from the Full Functional Dependency (FFD) that is utilized as part of opponent knowledge.
Statistical revelation power also referred as privacy preserving data mining of micro data is on discharging data sets with the answers of considered respondents confined in such a method that (i) the respondents equivalent to the unrestricted records cannot be reproduced; (ii) the unconstrained data wait logically helpful. The confined data set is produced by either perturbing the unique data or by engendering simulated statistics of the creative data.

Let $V$ be the data set whose attributes are in the form of numerical value and they come into the category of confidential attributes $X = (X_1, X_2, ... X_l)$ with non-confidential attributes $Y = (Y_1, Y_2, ... Y_m$). Let $V'$ denote hybrid data set obtained from $V$ with the aid of R-microhybrid, whose attributes include $X' = (X'_1, X'_2, ..., X'_l)$ where $X'$ denote the hybrid version of $X$.

Given a tuple $t$ and bucket $b$, the probability that $t$ is in the $b$ depends on the fraction of $t'$ column values that match the column values in $b$. if some column value of $t$ does not appear in the corresponding column $b$ can potentially match $|b|^c$ tuples, where $|b|$ is the number of tuple in the $b$. without additional knowledge one has to assume that the column values are independent therefore each of the $|b|^c$ tuples equally likely to be an original tuple. The probability that $t$ is in $b$ depends on the fraction of the $|b|^c$ tuples that match $t$.

In Privacy Preserving Data Mining (PPDM), an extensively utilized algorithm for attaining data mining objectives as protecting privacy is supported with on k anonymity. The most widespread algorithm for achieving observance with k anonymity is to restore definite values with fewer specific semantically reliable values. The author Nissim Matatov et al., proposed a diverse algorithm for determining k anonymity by splitting up the unusual data set into numerous ridges
such that all one of them sticks to k anonymity [NLO10]. A Quasi Identifier (QI) constraint for determining k anonymity of $S$ is a subset of an input feature set:

$$QI = \{Q_1, Q_2, \ldots, Q_m\}$$

Then the data set anonymity level with respect to $Q_I$, denoted as $DAL_{QI}(S)$ is given as below

$$DAL_{QI}(S) = QI \ast g_{min}(count) \ast \pi(QI)(S)$$

The author Keke Chen and Ling Liu explained numerous features of the algorithm. First, numerous kinds of well known data mining representations distributed a similar stage of representation quality over the geometrically disconcerted data set as in excess of the creative data set [KCL10].

Geometric data perturbation comprises of a sequence of geometric transformations in the form of random order, with the aid of multiplicative transformation ($R$), translation transformation ($\varphi$), and distance perturbation ($\Delta$). Then,

$$G(X) = R_x + \varphi + \Delta$$

Like k anonymity, the kernel method in k anonymity estimates the class label for a point $x$ using class labels of its neighbors. Let $K\lambda (x, x_i)$ denote the kernel function with k anonymity for weighting any point $x_i$ in $x$’s neighborhood, and let $\lambda$ define the geometric width for k-anonymity. Let us assume $\{x_1, x_2, \ldots, x_n\}$ be the points in the $x$’s neighborhood determined by $\lambda$, then then kernel classifier for continuous class labels is defined as

$$f(x) = \frac{\sum_{i=1}^{n} K_{\lambda}(x, x_i) y_i}{\sum_{i=1}^{n} K_{\lambda}(x, x_i)}$$

The kernel $K_{\lambda}$ with K-anonymity is defined as

$$K_{\lambda}(x, x_i) = D \frac{|x - x_i|}{\lambda}$$
Then the mean square error or loss function for kernel $K$ with K-anonymity is given as below

$$L(d(X), y) = \sum_{i=1}^{n} f(x_i) - y_i$$

With the limits in existing research, the proposed system starts with the focus is on classifying data stream and the occasion to evaluate each record using FC algorithm. The FC algorithm is able to judge internally on set of marked data. The superiority of models updated from them is good enough for deployment on unlabeled records, or whether additional labeled records are required for classification for further evaluation.

4.3 **BUCKETIZATION BASED FC ALGORITHM**

Bucketization based FC algorithm is developed to construct a classification set with both unlabeled and a small amount of labeled instances. The idea of bucketization based model is to support data by using the FC algorithm. The FC algorithm is able to decide internally on set of marked data. Before classification, the correlation set of attribute in the each record set are combined using bucketization algorithm. The superiority of models updated from them is enough for use of unlabeled records, or whether more set of labeled records are needed for classification is processed. The environment of data stream consists of diverse set of requirements for the task of determining the labeled data. The significant part of the bucketization model based data stream approach is defined as

- Practice the set of processed data one at a instance of time,
- Utilize a restricted amount of memory,
- Process the data streams in limited time, and
Predict the characteristics of the data.

The data streams consist of numerous confronts for data mining algorithm design. The derived procedure saves the resource constraints such as energy and uses less number of resources to achieve the distribution of data streams over the set of data items. The architecture diagram of the bucketization based FC algorithm is illustrated in Fig. 4.1.

Fig. 4.1 Process of Bucketization Based FC Algorithm

Fig. 4.1 describes the data stream classification process based on the bucketization algorithm. With the Bucketing process, the data in the continuous data streams are grouped based on the frequency of occurrence of data in the dataset. After formation of group, the classification under the data processing is done through the procedure of FC algorithm.

Design Notations in terms of Data Stream

Consider a set of items to be presented as, \( I = \{i_1, i_2, \ldots, i_n\} \) with \( n \) number of items. The set of items is presented as a subset of \( I \). A transaction of the data in the
Data streams are defined as, $T (id, iset)$, where $id$ is the ID of the transaction, $iset$ determined the set of items contained in the transaction.

**Traditional Privacy Preservation without Data streams**

### Step 1:

```plaintext
repeat
    Generate PerformSelection (Population, nPop)
    Generate CreateNewGeneration
    (Population, Pcrossover, Pmutation)
    Evaluation (Z) = EvaluatePartitioning (Z)
end do
```

### Step 2:

```plaintext
EvaluatePartitioning (Z)
Z Partitioning
fv Partitioning fitness value
if $PALQ_1 (Z) < k$ then
    fv = 0
else
    fv = PerformWrapperCrossValidation (Z, S, I, nFolds)
end if
return fv
```

### Step 3:

```plaintext
SelectBestValidPartitioning (Population)
Population Population in descending order of fitness values
Z Best partitioning satisfying set k-anonymity
repeat
    Z
    Get next best partition in the population
    if $SPALQ_1 (Z) > k$ return Z
    until no partitions are left
    return Z = φ
    PrepareAnonymisedData (Z)
end if
```

### Step 4:

```plaintext
Z Optimal partitioning
S' Set of projections
for each $G_r \in Z$
    Do $PDS_r \rightarrow G_r$
    Randomize ($PDS_r$)
end for
```
While performing the transaction of set of data streams, the data are sent as i.

In terms of infinite sequence of blocks referred as $B_1, B_2, ..., B_n$, where each block is processed with an identifier. The length of the data stream is expressed as,

$$L(DS) = |B_1| + |B_2| + \cdots + |B_n|$$

Where $DS$ represents the data stream collection measures in length wise $L$. The total length of the data stream includes the infinite number of sequence blocks $B_1, B_2, ..., B_n$.

**Bucketized Data Formations with Data Streams**

//Bucketization

**Step 1:** Extract the data set from the database

**Step 2:** Divide the set of records present in the data set and place the set of Records $R_1, R_2, R_3, \ldots, R_n$ in data set with set of Records

**Step 3:** Identify the sensitive attributes based on information details and sort the remaining set of records based on the occurrence of the sensitive attributes

**Step 4:** Group similar set of records with set of buckets with set of Records $R_3$ as Continuous nominal such as $A_3, A_8, A_{11}$ and $A_{15}$, Records $R_3$ with another set of bucket as smaller value nominal such as $A_1$ and $A_{13}$ and Records $R_3$ with another set of bucket as larger value nominal such as $A_6$

**Step 5:** Analyze the set of records in each bucket and compute the look up table where diversity maintains the set of buckets with matching attributes and combine the set of correlated attributes

**Step 6:** Display the set of secured data.

The objective of the FC algorithm is to ensure whether a lookup table satisfies bucketization i.e., $\ell$ diversity. For the presence of each set of attribute $a$ in the group-n, the FC algorithm sustains a list of matching buckets, in which, each element in the list comprise with one matching bucket $b$, with a matching probability and the allocation of sensitive values $d(a,B)$. The FC algorithm scans each bucket $b$ to trace the occurrence of each value $v$ in bucket $b$. The bucketization based FC algorithm achieves the classification with the set of data in the data stream.
Bucketization is the process of defining the several grouped records based on their sensitive values. The apparent sensitive values of the attribute are identified and sorted based on the frequencies in ascending order. After sorting, the contiguous sensitive values are grouped into similar bucket. Only the buckets contain at least $\ell$ distinct sensitive values which are kept after bucketing process completion. In order to add sensitive value, choose the relevant bucket to pick and precede the process of bucketization. After all sensitive values are bucketed the larger buckets are divided into set of smaller number of buckets. The splitting is order preserving, for the frequency of all impressionable values in the bucket is smaller than that of all sensitive values in the split bucket. After splitting, all the buckets are ordered by maximum frequency of their impressionable values in ascending order. At the end of the process, bucketization model returns a set of disjoint buckets and least $\ell$ distinct impressionable values.

The purpose of bucketizing the attribute is to establish the least cost splitting up of a multidimensional data set into $B$ set of buckets, where the value of $B$ is smaller than the number of data points in the dataset. The value of a bucket is calculated by the cost function. The cost measure detains the compression of a bucket. A formation of bucketization is measured based on minimizing the average of total number of buckets of a bucket’s weight, number of data points and its perimeter or sum of its extents along each dimension.

Assuming the bucketization process is performed on a database with a set of tables comprising a set of records. Here a financial database DB with a set of tables $T$ comprises with a set of records $R$ is taken for instance. Each record includes several set of attribute with a set of values specified. Among the set of attribute, the sensitive attribute are identified and grouped into a set of buckets $B = \{B_1,B_2,\ldots,B_n\}$. Here,
credit approval data set from UCI repository is taken for result justification and real
time verification purpose. Data set file concerns credit card applications. All attribute
names and values have been changed to meaningless symbols to protect
confidentiality of the data. This data set is a mix of attribute such as continuous,
nominal with small numbers of values, and nominal with larger numbers of values. It
contains few missing values.

The flow diagram of the bucketization process with FCA in credit card
application is given in Fig. 4.2 the credit card information is extracted from the user
initially. This information is alphabetically sorted by frequency ascending or
descending. Next it is given into the grouping process and there are three groups in
the process. Group A contains the continuous nominal, group B holds small nominal
and group C holds large nominal.

![Flow Diagram in Credit Card Approval with FCA](image)

*Fig. 4.2 Flow Diagram in Credit Card Approval with FCA*
The process of sorting and grouping is referred as the bucketization process. Next classification takes places using FC algorithm. It is performed to classify the correlated attribute in secured manner. The privacy over the set of correlated set of attributes in the credit approval data set is obtained by adopting FC algorithm. While applying the FC algorithm to the bucketized data, the privacy level over the data is high. The FCA is a decomposition based algorithm, which presents very high research throughput with low memory. FC algorithm achieves the equivalent lookups on every individual field of correlated set of attributes first. The algorithmic steps elaborated in FC algorithm are usually processed by a lookup table so as to attain the optimal throughput presentation. The table size with the set of attributes depends on the number of bits presented in header field.

The entry given in the lookup table with set of bucketized records practically accommodates the set of filters for which the record \( r_i \) covers up the total number of records \( i \). In a grouped set of records, each unique set of filters is assigned as index. The identifiers of the determined set of attributes are formed as aided and utilized for data stream classification.

### Table 4.1 Sample Data Set of Credit Approval

<table>
<thead>
<tr>
<th>Name</th>
<th>Zipcode</th>
<th>Age</th>
<th>Sex</th>
<th>Credit Card Nominal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>14850</td>
<td>23</td>
<td>M</td>
<td>A13</td>
</tr>
<tr>
<td>Bella</td>
<td>14589</td>
<td>24</td>
<td>F</td>
<td>A6</td>
</tr>
<tr>
<td>David</td>
<td>14863</td>
<td>25</td>
<td>M</td>
<td>A1</td>
</tr>
<tr>
<td>Edward</td>
<td>14850</td>
<td>21</td>
<td>M</td>
<td>A15</td>
</tr>
<tr>
<td>Stewart</td>
<td>15478</td>
<td>27</td>
<td>M</td>
<td>A3</td>
</tr>
<tr>
<td>Charle</td>
<td>14589</td>
<td>29</td>
<td>M</td>
<td>A8</td>
</tr>
<tr>
<td>Robert</td>
<td>14850</td>
<td>26</td>
<td>F</td>
<td>A11</td>
</tr>
</tbody>
</table>
The connection of all the record table lookup is closely the set of marked data that matches with a given set of packet identification. The group of records are processed and assigned with an id by presenting a lookup table.

Table 4.1 describes the sample data set comprises with set of good mix of attribute like sensitive and non sensitive. In the data set, name, zipcode, age are non sensitive attribute. Credit card nominal value is a sensitive attribute. With the set of sensitive attribute obtained, the buckets are created in which it arbitrarily produce each set of sensitive attribute values among each set of bucket formed.

Table 4.2 Bucketized Data

<table>
<thead>
<tr>
<th>Name</th>
<th>Zipcode</th>
<th>Age</th>
<th>Sex</th>
<th>Credit card Nominal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stewart</td>
<td>15478</td>
<td>27</td>
<td>M</td>
<td>Continuous Nominal (A3, A8, A11 and A15)</td>
</tr>
<tr>
<td>Charle</td>
<td>14589</td>
<td>29</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Robert</td>
<td>14850</td>
<td>26</td>
<td>F</td>
<td>Nominal with small numbers of values (A1 and A13)</td>
</tr>
<tr>
<td>Edward</td>
<td>14850</td>
<td>21</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>David</td>
<td>14863</td>
<td>25</td>
<td>M</td>
<td>Nominal with large numbers of values (A6)</td>
</tr>
<tr>
<td>Alice</td>
<td>14850</td>
<td>23</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Bella</td>
<td>14589</td>
<td>24</td>
<td>F</td>
<td></td>
</tr>
</tbody>
</table>

Rather than specifying the sensitive attribute name as bucket name. Assign a name to sensitive values with a group name as group-1, group-2, and group-n. Instead of putting continuous, smaller and larger nominals into group-1, group-2 and so on this bucketization process partitions the set of attribute in the data set then interrelated attribute are specified in the same column. This process is very adaptable for enhancing the privacy over the set of data in the dataset. In terms of data service, grouping the set of interrelated attribute conserves the associations amongst those attribute. Usually, the connection of uncorrelated set of attribute provides less security.
in data of the data set than the association of interrelated attribute since the associations of uncorrelated attribute values is much less common and thus more exclusive.

*Fig. 4.3 Index based Reference in the Bucketization Process*

With the set of records maintained in the dataset, the bucketization of data is performed. Bucketization process is done through identifying and grouping the set of sensitive attribute. With the obtained set of grouped data, the privacy of the system is enhanced.

With the set of records maintained in the dataset, the bucketization of data is performed. Bucketization process is done through identifying and grouping the set of sensitive attribute. With the obtained set of grouped data, the privacy of the system is enhanced. Further, in enhancing the privacy, the design of FC algorithm buckets based on record sets referring index show in Fig. 4.3.
Process of Bucketization based FC Algorithm

//Bucketization

**Step 1:** Extract the data set from the Database
- Taking credit approval data set from UCI repository
- Nominal Attribute: continuous, smaller values and larger value
  - Continuous: $A_3, A_8, A_{11}$ and $A_{15}$
  - Smaller values: $A_1$ and $A_{13}$
  - Larger values: $A_6$

**Step 2:** Divide the set of records present in the data set.
- Set of Records $R_1, R_2, R_3, ..., R_n$ in data set
- Set of Records $R_1$ as Nam
- Records $R_2$ as Age
- Records $R_3$ as Credit card nominal details and so on

**Step 3:** Identify the sensitive attribute
- Sensitive attribute: Credit Card Nominal and information details

**Step 4:** Sort the remaining set of records based on the occurrence of the sensitive attribute

**Step 5:** Group the similar set of records with set of buckets.
- Set of Records $R_3$ with set of buckets as Continuous nominal such as $A_3, A_8, A_{11}$ and $A_{15}$
- Records $R_3$ with another set of bucket as smaller value nominal such as $A_1$ and $A_{13}$
- Records $R_3$ with another set of bucket as larger value nominal such as $A_6$

// FC Algorithm

**Step 6:** Analyze the set of records in each bucket

**Step 7:** Compute the look up table

**Step 8:** Diversity maintains the set of buckets with matching attribute

**Step 9:** Combine the set of correlated attribute

**Step 10:** Display the set of secured data.
The FC algorithm will derive a lookup table for the set of each packet data in the data set. By setting up the id for each type of data record value, the lookup table is formed with the set of correlated attribute to process. Each column in the lookup table consists of bucket with values for each partitioned data. It is also provided a say that the occurrence of the value in each one of the FC algorithm checks the diversity of the correlated set of attribute value. The objective of the FC algorithm is to ensure whether a lookup table satisfies bucketization i.e, \( \ell \)-diversity. For the presence of each set of attribute \( (a) \) in the group \( n \), the FC algorithm sustains a list of matching buckets, in which, each element in the list comprise with one matching bucket \( b \), with a matching probability and the allocation of sensitive values \( d (a,B) \). The FC algorithm scans each bucket \( b \) to trace the occurrence of each value \( v \) in bucket \( b \). the bucketization based FC algorithm achieve the classification with the set of data in the data stream.

4.1 EXPERIMENTAL EVALUATION

4.4.1 Performance Evaluation of FCA for Credit Approval Data Set

The performance of FCA for credit approval data set, with the metrics of classification accuracy, execution time and classification error rate is obtained from the equation 3.1, 3.2 and 3.3 respectively and the same is compared with the existing methods DMDP and FFD.

a) Classification Accuracy

The classification accuracy for the two methods DMDP and FFD is tabulated below and compared with the proposed Bucketization based FCA.
Table 4.3 lists out the classification accuracy for different methods, such as DMPD, FFD and bucketization based FCA. The lowest classification accuracy is seen in the DMDP method for different data streams ranging from 67.01 – 67.96 %, whereas the classification accuracy of FFD is in the range of 78.08 – 78.14 %. But bucketization based FCA shows higher rate of accuracy is 90.08 – 90.25 % and it is better than other two methods. Fig. 4.4 shows the measure of classification accuracy of proposed Bucketization based FCA and compared with the existing DMPD and FFD for credit approval dataset. This is also evident that with the increase in the number of data streams, the classification accuracy is not decreased using the proposed FCA than the state of the art methods. The classification accuracy is increased by 22 – 23 % when compared with DMPD and 11 – 12 % from FFD.
b) Execution Time

While considering the time taken for execution, Table 4.4 shows the execution time of Bucketization based FC algorithm is comparatively low in the range of 107 – 134 ms whereas the existing DMPD recorded the execution time in the range of 132 – 159 ms and FFD recorded 120 – 153 ms.

Table 4.4 Comparison Table for Execution Time of FCA (Credit Approval Data Set)

<table>
<thead>
<tr>
<th>No. of Data Streams</th>
<th>Execution Time (ms)</th>
<th>Bucketization based FCA</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMPD</td>
<td>FFD</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>132</td>
<td>120</td>
</tr>
<tr>
<td>100</td>
<td>134</td>
<td>127</td>
</tr>
<tr>
<td>150</td>
<td>137</td>
<td>134</td>
</tr>
<tr>
<td>200</td>
<td>145</td>
<td>140</td>
</tr>
<tr>
<td>250</td>
<td>156</td>
<td>146</td>
</tr>
<tr>
<td>300</td>
<td>159</td>
<td>153</td>
</tr>
</tbody>
</table>
The Fig. 4.5 proved that the execution time of proposed FCA is comparatively less than the other two methods because of the application of bucketization principle which helps in reducing the time taken for execution process by 13 – 20 ms compared to DMPD and 20 – 30 ms compared to FFD.

c) Classification Error Rate

Table 4.3 shows the classification error rate for all three methods. Bucketization based FC algorithm gives lower classification error rate in the range of 0.211 – 0.218 %. The two existing methods DMPD recorded the classification error rate in the range of 0.401 – 0.411 % whereas FFD recorded the classification error rate as 0.331 – 0.348 %.
Table 4.5 Comparison Table for Classification Error Rate of FCA (Credit Approval Data Set)

<table>
<thead>
<tr>
<th>No. of Data Streams</th>
<th>Classification Error Rate (%)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DMPD</td>
<td>FFD</td>
</tr>
<tr>
<td>50</td>
<td>0.401</td>
<td>0.338</td>
</tr>
<tr>
<td>100</td>
<td>0.421</td>
<td>0.337</td>
</tr>
<tr>
<td>150</td>
<td>0.402</td>
<td>0.348</td>
</tr>
<tr>
<td>200</td>
<td>0.411</td>
<td>0.331</td>
</tr>
<tr>
<td>250</td>
<td>0.403</td>
<td>0.332</td>
</tr>
<tr>
<td>300</td>
<td>0.409</td>
<td>0.334</td>
</tr>
</tbody>
</table>

Fig. 4.6 Comparison Graph for Classification Error Rate of FCA (Credit Approval Data Set)

Fig. 4.6 shows the measure of classification error rate with respect to different sizes of data streams in the range of 50 to 300. The Fig. 4.6 provides perfect evidence that the classification error rate is comparatively minimized by applying using the
proposed Bucketization based FC algorithm by 18 – 20 % compared to DMPD and 5 – 6 % compared to FFD respectively.

4.4.2 Performance Evaluation of FCA for Spam Base Data Set

The performance evaluation of FCA based on classification accuracy, execution time and classification error rate for spam base data set is obtained and tested with existing methods DMDP, FFD using the equations mentioned in 3.1, 3.2, and 3.3 respectively.

a) Classification Accuracy

Here, the classification accuracy for the proposed Bucketization based FC algorithm is compared with the existing DMPD approach and Full Functional dependency (FFD) for classification of data streams using Eq.3.1.

Table 4.6 Comparison Table for Classification Accuracy of FCA (Spam Base Data Set)

<table>
<thead>
<tr>
<th>No. of Data Streams</th>
<th>Classification Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DMPD</td>
</tr>
<tr>
<td>50</td>
<td>67.11</td>
</tr>
<tr>
<td>100</td>
<td>67.12</td>
</tr>
<tr>
<td>150</td>
<td>67.14</td>
</tr>
<tr>
<td>200</td>
<td>67.15</td>
</tr>
<tr>
<td>250</td>
<td>67.11</td>
</tr>
<tr>
<td>300</td>
<td>67.17</td>
</tr>
</tbody>
</table>

Table 4.6 also shows the classification accuracy values of the proposed bucketization based FC algorithm for spam base data set is efficient than other two approaches.
Fig. 4.7 depicts classification accuracy measurements of proposed Bucketization based FCA provides 14 – 23 % high accuracy when compared to the existing DMPD approach and 6 – 13 % higher accuracy compared with FFD.

b) Execution Time

The execution time required for three different methods is listed below in Table 4.7 Execution time is measured in milliseconds (ms) as depicted in table 4.7.

The Fig. 4.8 shows clearly proposed bucketization based FC algorithm compared with existing algorithm DMDP and FFD, the experiment results shows that proposed algorithm FCA consumes less time of about 10 – 18 ms and 15 – 24 % respectively.
Table 4.7 Comparison Table for Execution Time of FCA (Spam Base Data Set)

<table>
<thead>
<tr>
<th>No. of Data Streams</th>
<th>FFD</th>
<th>DMPD</th>
<th>Bucketization based FCA</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>134</td>
<td>124</td>
<td>110</td>
</tr>
<tr>
<td>100</td>
<td>136</td>
<td>128</td>
<td>112</td>
</tr>
<tr>
<td>150</td>
<td>140</td>
<td>137</td>
<td>120</td>
</tr>
<tr>
<td>200</td>
<td>146</td>
<td>141</td>
<td>126</td>
</tr>
<tr>
<td>250</td>
<td>150</td>
<td>148</td>
<td>130</td>
</tr>
<tr>
<td>300</td>
<td>157</td>
<td>152</td>
<td>142</td>
</tr>
</tbody>
</table>

Fig. 4.8 Comparison Graph for Execution Time of FCA (Spam Base Data Set)
c) **Classification Error Rate**

The classification error rate is defined as the error occurred during the classification of data records which are measured with respect to number of data streams.

**Table 4.8 Comparison Table of Classification Error Rate of FCA (Spam Base Data Set)**

<table>
<thead>
<tr>
<th>No. of Data Streams</th>
<th>Classification Error Rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DMPD</td>
</tr>
<tr>
<td>50</td>
<td>0.412</td>
</tr>
<tr>
<td>100</td>
<td>0.415</td>
</tr>
<tr>
<td>150</td>
<td>0.419</td>
</tr>
<tr>
<td>200</td>
<td>0.461</td>
</tr>
<tr>
<td>250</td>
<td>0.448</td>
</tr>
<tr>
<td>300</td>
<td>0.464</td>
</tr>
</tbody>
</table>

Table 4.8 tabulates the classification error rate for the proposed Bucketization based FC algorithm and comparison was made DMPD and FFD methods.

**Fig. 4.9 Comparison Graph for Classification Error Rate of FCA (Spam Base Data Set)**
Fig. 4.9 shows the classification error rate obtained for the three methods bucketization based FC algorithm, DMPD and FFD respectively with respect to varying data streams. The Fig. shows that the error rate for the data streams is less by the utilization proposed method than when compared to DMPD which is 20 – 38 % and FFD 10 – 21 %.

4.4.3 Performance Evaluation of FCA for Adult Data Set

a) Classification Accuracy

The rate of classification accuracy using adult data set is presented in the table 4.9, which shows the classification accuracy values for the proposed FCA and comparison made with state of the art methods. The method DMPD recorded the classification accuracy as 67.11 – 69.56 % whereas FFD recorded the values as 78.00 – 82.12%. The proposed method Bucketization based FC algorithm recorded highest value of classification accuracy in the range of 90.12 – 90.17 %.

<table>
<thead>
<tr>
<th>No. of Data Streams</th>
<th>Classification Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DMPD</td>
</tr>
<tr>
<td>50</td>
<td>67.11</td>
</tr>
<tr>
<td>100</td>
<td>67.12</td>
</tr>
<tr>
<td>150</td>
<td>69.56</td>
</tr>
<tr>
<td>200</td>
<td>67.15</td>
</tr>
<tr>
<td>250</td>
<td>68.23</td>
</tr>
<tr>
<td>300</td>
<td>67.17</td>
</tr>
</tbody>
</table>
Fig. 4.10 shows that the highest classification accuracy was obtained by bucketization based FCA and the resulting in the improvement from 09 – 18 % of DMPD and 8 – 12 % of FFD.

b) Execution time

Table 4.10 recorded the execution time with minimum execution time observed by the proposed Bucketization based FC algorithm. The proposed method recorded the execution time between the range of 120 – 130 ms compared to DMPD which recorded 144 – 180 ms and 123 – 150 ms using FFD respectively.
Table 4.10 Comparison Table for Execution Time of FCA for Adult Data Set

<table>
<thead>
<tr>
<th>No. of Data Streams</th>
<th>DMPD</th>
<th>FFD</th>
<th>Bucketization based FCA</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>144</td>
<td>123</td>
<td>120</td>
</tr>
<tr>
<td>100</td>
<td>149</td>
<td>122</td>
<td>123</td>
</tr>
<tr>
<td>150</td>
<td>152</td>
<td>128</td>
<td>126</td>
</tr>
<tr>
<td>200</td>
<td>157</td>
<td>136</td>
<td>127</td>
</tr>
<tr>
<td>250</td>
<td>168</td>
<td>147</td>
<td>129</td>
</tr>
<tr>
<td>300</td>
<td>180</td>
<td>150</td>
<td>130</td>
</tr>
</tbody>
</table>

Fig. 4.11 Comparison Graph for Execution Time of FCA (Adult Data Set)

Fig. 4.11 shows the measure of execution time among the three methods. From the Fig, it is illustrative that the execution time observed under the proposed method is comparatively lesser than the two existing methods and shows an improvement of 15 – 24ms from DMPD and 10 – 18 ms compared from FFD.
c) Classification Error Rate

The values for classification error rate is recorded in the table 4.11 and compared with other two methods. The Bucketization based FC algorithm measured the classification error rate between 0.211 – 0.231 %. While the others DMDP, FFD the recorded classification error rate 0.421 – 0.452 % and 0.342 – 0.381 % respectively.

Table 4.11 Comparison Table for Classification Error Rate of FCA for Adult Data Set

<table>
<thead>
<tr>
<th>No. of Data Streams</th>
<th>Classification Error Rate (%)</th>
<th>DMPD</th>
<th>FFD</th>
<th>Bucketization based FCA</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>0.421</td>
<td>0.351</td>
<td>0.211</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>0.415</td>
<td>0.343</td>
<td>0.212</td>
<td></td>
</tr>
<tr>
<td>150</td>
<td>0.434</td>
<td>0.372</td>
<td>0.224</td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>0.452</td>
<td>0.381</td>
<td>0.231</td>
<td></td>
</tr>
<tr>
<td>250</td>
<td>0.421</td>
<td>0.342</td>
<td>0.212</td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>0.432</td>
<td>0.339</td>
<td>0.216</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 4.12 shows the measure of classification rate for three methods. From this, the error rate derived from the proposed method is comparatively less than the two other methods. All those analysis proved that the proposed bucketization based FC algorithm offers better classification model with higher rate of accuracy with the set of data such as credit approval, spam base and adult data sets.
4.5 SUMMARY

A novel bucketization technique with the incorporation of FC algorithm is presented in this research work to achieve higher classification with various set of data in the data stream. At first, the data in the data set is bucketized by obtaining the set of buckets using bucketization technique. After obtaining bucketized set of data, the FC algorithm is applied to classify the data. Bucketization based FC algorithm transformed the original data set into the bucketized set of data with the formation of buckets in individual records, which provides perfectly accurate result for classification.