ISSN 2277 – 5528
Impact Factor- 4.015

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES &MANAGEMENT

ASSISTANCE TO PATRONS FOR DECISION MAKING IN BUSINESS BY CREATION AND CLUSTERING RECUMBENT LAYOUTS THROUGH RIFCM SOFT COMPUTING ALGORITHM

Kalluri Satya Naresh*[1], Divya Vani .Y[2] & Srikanth Sattenapalli[3]
*[1,2,&3]Alamuri Ratnamala Institute of Technology Mumbai, Maharashtra, India

ABSTRACT

Data Mining has esteemed preponderance in today's surpassingly vying business encompassment. Clustering intellection in data mining is benevolent for patrons for scrutinizing and decision making in business. How manifestly, agilely recumbent layouts assist in data mining errand or functionalities is excogitated in this paper. The conventional upstanding layouts hoarded from prevalent SQL queries are not precisely useful for data mining intellections. Data transmutation should be done on conventional upstanding layouts for using unswervingly but data redeem or preprocessing is lot of time conceiving and striving task. As conventional upstanding layouts are impotent for using precisely in data mining errand, recumbent layout are created by using recumbent aggregations. To save effort and time, recumbent layouts can be used precisely in data mining intellections or functionalities without performing any data redeem instead of conventional upstanding layouts. How the recumbent layout spawned by using recumbent aggregations, Elect-Project-Conjugate (EPC) method and how these layouts are useful for data mining clustering task or functionality is elucidated in this paper. Clustering can be transacted by soft computing algorithms like fuzzy c-means and hard k-means clustering algorithms but these algorithms cannot supervise inexactitude and vagueness of data. So in this paper Rough Intutionistic Fuzzy C-means (IRFCM) algorithm is excogitated for clustering recumbent layout which handles inexactitude and vagueness of data. Thus clustered recumbent layout output from RIFCM clustering can be useful for patrons for decision making in business escalation and the whole process is contemplated with the help of exemplar.

Keywords- *Recumbent aggregation, recumbent layout, conventional upstanding layout, Data mining, Clustering algorithm, Fuzzy set, Rough Set, Intutionistic Fuzzy set.*

I. INTRODUCTION

This paper predominantly contemplates about spawning of recumbent layouts by using recumbent aggregations and clustering of recumbent layout by using rough intutionistic fuzzy c-means (RIFCM) clustering algorithm by which business patrons can make decisions pertinent to business escalation.

Conventionally procuring a data set for data mining projects is a most encumbrance and time conceiving process. By opting conventional SQL aggregation operation or functions tables are spawned which are known as conventional upstanding layouts. These conventional layouts are stockpiled in database. Data warehousing is transacted first by congregating data from immense, enormous databases. Data warehouses consist of the conventional upstanding layouts for utilizing in data mining projects or functionalities. First data redeem or preprocessing should be transacted for using conventional layouts in data mining functionalities or tasks precisely. But data redeem or preprocessing consists of onus like data assortment, data transmogrification, data culling etc. These entire tasks data assortment, data transmogrification, data culling takes lot of time to transact and it is striving onus to accomplish. But without data redeem or preprocessing conventional upstanding layouts cannot be precisely useful for data mining errand or functionalities like clustering, classification, association rule mining , prediction, decision making etc. So spawning of recumbent layout is performed instead of conventional upstanding layouts spawning. These recumbent layouts can be directly, precisely used in data mining functionality like clustering, classification, association rule mining etc without any performing data redeem. So this paper elucidates spawning of recumbent layouts and how these recumbent layouts can be used in data mining algorithms agilely and without burden or onus.

The recumbent layout created by recumbent aggregations is hoarded in database. From database the recumbent layouts are queried and used for data mining functionalities like clustering, classification,

Association rule mining, prediction, decision making etc. This paper mainly induces how recumbent layouts can be predominantly used for clustering.

II. LITREATURE REVIEW

Data can be excavated with SQL and data has to be engendered as relational tables. Customarily the data is stockpiled as tables within database. Two sorts of data objects are pertinent for incessant item set discord and item set proceedings or transactions. For each variety, there are predominantly two intrinsic layouts for depicting the data objects which are called as denominated as conventional upstanding layouts and recumbent layouts.

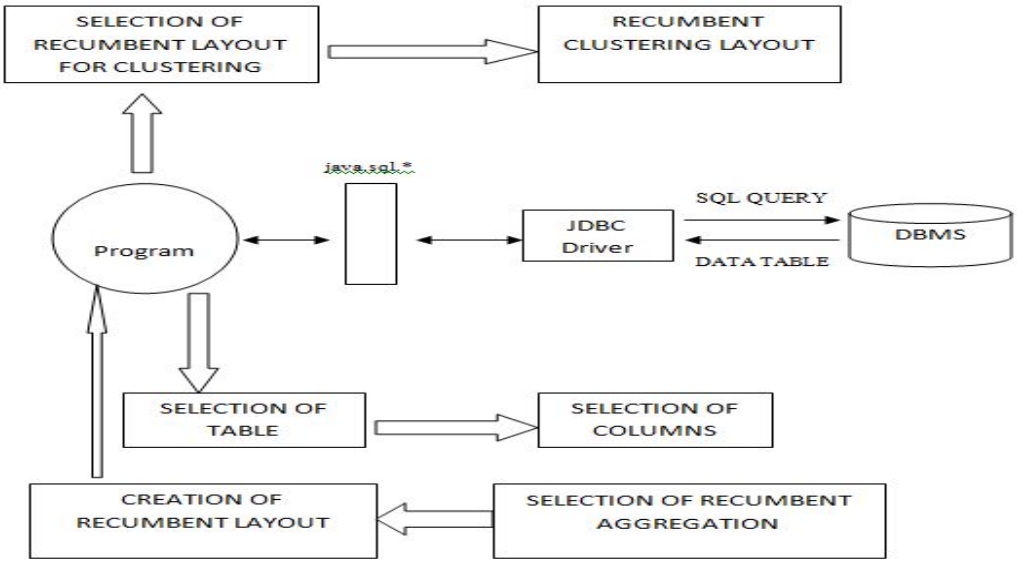
Clustering can be performed by using many soft computing clustering algorithms like fuzzy c-means, hard k-means, Rough C-means etc. Hard K-means induces characteristic function for clustering where it depicts whether a data object or point or element pertains to cluster or does not pertain to cluster. If a data point or element of data object pertains to a cluster, it characteristic function value is 1 and if the data point or element of data set does not pertain to cluster characteristic function value is 1. In Hard K-means 'K' denominates the number of data points or data elements or objects in cluster are steadfast. Fuzzy C-means clustering depicts fuzzy membership function which pertain values from 0 to 1 and clusters data points or elements or objects by maneuvering uncertainty of data. In fuzzy c-means clustering algorithm C denominates the number of clusters are cluster centers are steadfast.

Fuzzy C-means and Hard K-means cannot maneuver inexactitude and vagueness of data precisely, so rough set concept is induced to handle inexactitude and vagueness of data while transacting clustering. Intutionistic fuzzy set pertain participation and non-participation of data by using hesitation function. So in this paper rough intutionistic fuzzy set for clustering recumbent layouts or data sets handling all cases proficiently.

III. METHODOLOGY

The methodology contemplated in this paper is substantially propitious for data mining users by utilizing recumbent layout dexterously and agilely. The output of using recumbent layout in data mining province is fruitful and utilitarian for customers, business outcrier for assay, reconciliation.

Methodology is asunder into two modules, one is spawning of recumbent layout. The second module is clustering recumbent layout efficiently by using IRFCM algorithm. The architecture of methodology elucidated is as follows:



3.1 System Architecture

Module 1

Spawning of Recumbent layout:

The upstanding layouts created by using prevalent SQL interrogatory are not precise for using in data mining. Lot of exertion, onus is taken to use upstanding layouts in data mining errand. Data redeem should be performed for using

ISSN 2277 – 5528
Impact Factor- 4.015

upstanding layout in data mining which is tenacious, time conceiving and exhausting task. So in lieu of upstanding layouts recumbent layouts are created in this module. There is no need of data redeem or preprocessing to comply on horizontal layouts to use proficiently in data mining province. The following contrivance gives induction of creating conventional upstanding layouts and recumbent layouts.

Definitions

D is a database table interpolating primary key X, A1, A2,…….,Ai as autonomous columns, S as one statistical column and it is exemplified as D(X, D1, D2….Di, S). In online analytical transaction processing terms the above is elucidated as follows:

D is the fact table procuring X as primary key, i measurements or dimensions, S as quantum or measure column where Z is the immensity of the table, A1, A2 …….Ai are foreign keys in fact table and primary keys in look-see tables.

D is the data table or database table or input table from which conventional upstanding layouts or recumbent layouts are created. Conventional upstanding layouts are depicted as Du, and recumbent layouts are depicted as D_R. Transmogrification of conventional upstanding layout to recumbent layout is the intension of this module. By adopting recumbent functions transmogrification of conventional upstanding Layout to recumbent layout can be done.

Let us deem the following data table or database table as example having X as primary key, A1, A2 as Autonomous columns and S as statistical column.

K	A1	A2	S
1	3	X	9
2	2	Y	6
3	1	Y	10
4	1	Y	0
5	2	X	1
6	1	X	NULL
7	3	X	8
8	2	X	7

3.2 Database Table

Consider the query Select A1, A2, Sum (S) from D group by A1, A2 order by A1, A2.

A1	A2	S
1	X	NULL
1	Y	10
2	X	8
2	Y	6
3	X	7

3.3 Conventional Upstanding Layout

The above conventional upstanding layout is spawned by applying query Select A1, A2, Sum (S) from D group by A1, A2 order by A1, A2. In this example sum aggregation operation or function is used for creating conventional upstanding layout. Conventional aggregation operations or functions like sum (), max(), avg(), min() etc are used for creating conventional upstanding layouts. The above upstanding layout procures A1, A2 as primary key and one statistical column that is S. This upstanding layout is impotent for data mining errand like classification, association

rule analysis and clustering etc. Hence recumbent layout is created instead of conventional upstanding layout. The following table is the recumbent layout procuring two aggregated columns and one primary key.

A1	A2	S
1	NULL	10
2	8	6
3	17	NULL

3.4 Recumbent Layout

Recumbent layouts are created by using recumbent aggregations. In the query, conventional aggregation operations or functions are replaced by recumbent aggregations to create recumbent layouts. The layouts created by recumbent function can be useful precisely for algorithms or tasks of data mining without data redeem.

Recumbent aggregations renovate the conventional upstanding layout to recumbent layout by mutating the aggregation column S to list of transmogrifying columns T1…….TK, where T1…..Tk are ration from A1…..Am. The syntax for perceiving conventional upstanding layout is as follows.

Select A1,….Am, sum (N) from D group by A1….Am.
The above query fruitage an upstanding layout data set equipping m+1 columns where Sum (S) is the sole aggregated column and m columns A1…Am act as primary key in the upstanding layout.

To restyle the upstanding layouts to recumbent layout, recumbent aggregations are used and are epitomized as Ra. The exploitation of recumbent aggregations is to metamorphose aggregated column N by a list of transmogrifying columns T1……Tk where the columns T1….. Tk are ration of columns A1……Am and k<m. So to elucidate SQL code by recumbent aggregations four specifications are used, they are D is the data or database table, A1….Am, are the congregate columns, T1….Tk are transmogrifying columns and S is the aggregate column.

The scaffolding for creating recumbent aggregations is agnate to the scaffolding for conventional upstanding aggregations. The recumbent aggregation is insinuated by Ra(S BY T1,….,Tk) where Ra is the conventional SQL aggregation operation or function ,where T1……..Tk are the transmogrifying columns and S is the aggregation column. Metamorphosing of substantial aggregation operation or function to recumbent aggregation is accomplished by applying “By” clause which mutates the aggregation column S to list of transmogrifying columns Y1……Yk which conceives a recumbent layout instead of conventional upstanding layout.

The syntax for prefabricating of recumbent layout is as follows:
SELECT A1,….,Aj , Ra(S BY T1,….,Tk) FROM F GROUP BY T1,….,Tj .
Consider a discernable database as example consisting boutique data in Table called Purchase. Purchase table is endowed with columns or attributes like boutiqueid, boutiquename, departmentid, department name, year, month, day, date, cost, quantity, purchaseamt, itemid, itemquantity, totalcost etc as . Suppose we crave to known total purchase for each boutique by each month.

Then the substantial statement using SQL for the above query is Select boutiqueid, month, sum (purchaseamt) from Purchase group by boutiqueid, month order by strid, day.
This gives conventional layout as below

ISSN 2277 – 5528
Impact Factor- 4.015

Boutique id	Month	Total purchase amount
1	Jan	120
1	Feb	111
1	Mar	98
-	-	-
-	-	-
-	-	-
1	Dec	180
2	Jan	300
2	Feb	250
2	Mar	275
-	-	-
-	-	-
-	-	-
2	Dec	200
-	-	-
-	-	-
-	-	-
-	-	-

3.5 Conventional Upstanding Layout

This upstanding layout is impotent for data mining tasks or algorithms as it posses sole aggregated column and boutiqueid, month concertedly act as primary key procuring abounding records. So by using recumbent aggregations recumbent layout is spawned procuring bounteous aggregated columns and bountiqueid is the sole primary key.

The syntax for procuring recumbent layout with the sustenance is as follows:
Select boutiqueid, sum (purchase BY month) from transaction group by boutiqueid.

Boutique id	Total purchase amount Jan Feb March..... Dec
1	120 111 98 180
2	300 250 275 200
.	- - - ---------- -
.	- - - ---------- -

3.6 Recumbent Layout

Creation of recumbent layout is percolated by using Elect-Project-Conjugate(EPC) method.
First we need to elect the data table or database table from database and elect the columns that we want aggregate, group by, transmogrify for which recumbent layouts have to be created.
Elect the group by columns A_1…..A_j
Select the statistical column S to aggregate
Elect the transmogrify columns Y_1….Y_k.
In this province aggregation of the column is elucidated by spawning recumbent layout with the help of EPC method. Create each table with a conventional aggregation function for each result column then conjugate all the tables to procure recumbent layout. We aggregate from R into X through EPC aggregation queries. Each table R_1 represents to one combination of sub conjugating and has {A_1… A_j} as primary key, aggregation column S as the sole non-key column. To get wrap up result it is important to induce additional table R_0 that will be outer joined with projected tables.

ISSN 2277 – 5528
Impact Factor- 4.015

Three important Steps in EPC Method to spawn recumbent layout:

(.) First Table R_0 is spawn having distinct combination of group by columns $A_1,\ldots\ldots,A_j$.
(.) For each identical combination of Transmogrifying columns $T_1,\ldots\ldots,T_k$, Tables $R_1,\ldots\ldots,R_x$ are created.
(.) At the end to wrap up table R_0 is left outer joined with each tables R_1 to R_X.

How the tables are spawned is precisely contemplated below.

Table R_0 expounds the number of output rows and creates the primary key. R_0 is spwaned so that it accords every persisting combination of $A_1,\ldots\ldots,Aj$ and R_0 procures $A_1,\ldots\ldots,A_j$ as primary key.
INSERT INTO R0 SELECT DISTINCT $A_1,\ldots,A_j$ FROM D.

We should spawn tables R_1 to R_x .
Tables $R_1,,\ldots\ldots,R_d$ contain indentical aggregations for each combination of $T_1,\ldots,T_k$. The primary key of table $R_1\ldots.R_x$ is $T_1,\ldots\ldots\ldots,T_k$ and S is aggregated column.

INSERT INTO T_1 SELECT $T_i,\ldots.T_j$, V(N) FROM T/T_v
WHERE $R_1 = v_{11}$ AND …… $R_k= V_{k1}$ GROUP BY $A_i,\ldots.A_j$.
Finally, to get recumbent layout D_H is spawned by using the following query.
INSERT INTO D_H SELECT $R_0.A_1$, $T_0.A_2$, . . . , $R_0.A_j$, $R_1.S$, $R_2.S,\ldots,R_d.S$ FROM R_0 LEFT OUTER JOIN R_1 ON $R_0.A_1 = R_1.A_1$ and . . . and $R_0.A_j = R_1.Aj$ LEFT OUTER JOIN R_2 ON $R_0.A_1= R_2.A_1$ and . . . and $R_0.A_j = R_2.Aj$……..LEFT OUTER JOIN Rx ON $R_0.A1 = R_x.A1$ and . . . and $R_0.A_j = R_x.A_j$.

Real Time Example for Spawning Recumbent Layout:

Consider a discernable database as example consisting boutique data in Table called Purchase. Purchase table is endowed with columns or attributes like boutiqueid, boutiquename, departmentid, department name, year, month, day, date, cost, quantity, purchase, itemid, itemquantity, totalcost etc . Suppose we crave to known total purchase for each boutique by each month.

The following queries should be accorded to spawn recumbent layout by using EPC method

Query1:
INSERT INTO R_0 SELECT DISTINCT boutiqueid FROM Purchase.

Query2:
INSERT INTO R_1 SELECT boutiqueid, sum(purchaseamt) AS totalpurchaseamt FROM Purchase WHERE month='Jan' GROUP BY boutiqueid;

Query3:
INSERT INTO R_2 SELECT boutiqueid, sum(purchaseamt) AS totalpurchaseamt FROM Purchase WHERE month='Feb' GROUP BY boutiqueid;

Query4:
INSERT INTO R_3 SELECT boutiqueid, sum(purchaseamt) AS totalpurchaseamt FROM Purchase WHERE month='Mar' GROUP BY boutiqueid;

Query5:
INSERT INTO R_4 SELECT boutiqueid, sum(purchaseamt) AS totalpurchaseamt FROM Purchase WHERE month='Apr' GROUP BY boutiqueid;

Query6:
INSERT INTO R_5 SELECT boutiqueid, sum(purchaseamt) AS totalpurchaseamt FROM Purchase WHERE month='May' GROUP BY boutiqueid;

ISSN 2277 – 5528
Impact Factor- 4.015

Query7:
INSERT INTO R_6 SELECT boutiqueid, sum(purchaseamt) AS totalpurchaseamt FROM Purchase WHERE month='Jun' GROUP BY boutiqueid;

Query8:
INSERT INTO R_7 SELECT boutiqueid, sum(purchaseamt) AS totalpurchaseamt FROM Purchase WHERE month='Jul' GROUP BY boutiqueid;

Query9:
INSERT INTO R_8 SELECT boutiqueid, sum(purchaseamt) AS totalpurchaseamt FROM Purchase WHERE month='Aug' GROUP BY boutiqueid;

Query10:
INSERT INTO R_9 SELECT boutiqueid, sum(purchaseamt) AS totalpurchaseamt FROM Purchase WHERE month='Sep' GROUP BY boutiqueid;

Query11:
INSERT INTO R_{10} SELECT boutiqueid, sum(purchaseamt) AS totalpurchaseamt FROM Purchase WHERE month='Oct' GROUP BY boutiqueid;

Query12:
INSERT INTO R_{11} SELECT boutiqueid, sum(purchaseamt) AS totalpurchaseamt FROM Purchase WHERE month='Nov' GROUP BY boutiqueid;

Query13:
INSERT INTO R_{12} SELECT boutiqueid, sum(purchaseamt) AS totalpurchaseamt FROM Purchase WHERE month='Dec' GROUP BY boutiqueid;

Query14:
INSERT INTO RH SELECT F0.boutiqueid, R1.totalpurchaseamt AS Jan-amt, R2.totalpurchaseamt AS Feb-amt, R3.totalpurchaseamt AS Mar-amt, R4.totalpurchaseamt AS Apr-amt, R5.totalpurchaseamt AS May-amt, R6.totalpurchaseamt AS Jun-amt, R7.totalpurchaseamt AS Jul-amt, R8.totalpurchaseamt AS Aug-amt, R9.totalpurchaseamt AS Sep-amt, R10.totalpurchaseamt AS Oct-amt, R11.totalpurchaseamt AS Nov-amt, R12.totalpurchaseamt AS Dec-amt FROM R0 LEFT OUTER JOIN R1 on R0.boutiqueid=R1.boutiqueid LEFT OUTER JOIN R2 on R0.boutiqueid=R2.boutiqueid LEFT OUTER JOIN R3 on R0.bountiqueid=R3.boutiqueid LEFT OUTER JOIN R4 on R0.boutiqueid=R4.boutiqueid LEFT OUTER JOIN R5 on R0.bountiqueid=R5.boutiqueid LEFT OUTER JOIN R6 on R0.boutiqueid=R6.boutiqueid LEFT OUTER JOIN R7 on R0.bountiqueid=R7.boutiqueid LEFT OUTER JOIN R8 on R0.boutiqueid=R8.boutiqueid LEFT OUTER JOIN R9 on R0.bountiqueid=R9.boutiqueid LEFT OUTER JOIN R10 on R0.boutiqueid=R10.boutiqueid LEFT OUTER JOIN R11 on R0.bountiqueid=R11.boutiqueid LEFT OUTER JOIN R12 on R0.boutiqueid=R12.boutiqueid

The above 14 queries gives the required Recumbent Layout.

Module 2

Clustering of Recumbent Layouts using RIFCM algorithm

The recumbent layouts are spawned by taking data table as input from database and the spawned recumbent layouts are furthermore stockpiled in database. The recumbent layouts extant in database can redeem by maneuver servers like JDBC driver. Through JDBC driver abettors of data mining can associate to database and can claim the recumbent layouts they want for using in data mining errand or functionalities. Recumbent layouts can be fabulous for many functionality of data mining but in this paper we inaugurate how recumbent layouts can be opted or used for data mining functionality clustering**.**

Regard to fuzzy sets the participation values and the non-participation values of elements are 1's counterpart of each other. In multifarious real life standpoints Attansov contemplated that this is not the contingency. An exemplar is accorded that people egress in polling. The provenance are like some people accord vote according to circumstances

benevolence, some vote opposing situation, where as some others abnegate from voting. Similarly, in a standpoint like "yes"/ "no" some contributors accord "yes" , some contributors accord "no" and some contributors 'can't say". Such standpoints can be designed accurately by intuitionistic fuzzy sets. This model intellection is expounded below.

An intuitionistic fuzzy set A elucidated over a universe U through two functions known as participation and non-participation functions insinuated by μ_A *and* ν_A and satiating the peculiarity that for any $x \in A$, $0 \le \mu_A(x) + \nu_A(x) \le 1$. An confederate function is associated with intuitionistic fuzzy set inaugurated as hesitation function. The hesitation function connated by $\pi_A(x)$ and is elucidated as $\pi_A(x) = 1 - (\mu_A(x) + \nu_A(x))$ *for* $\forall x \in A$.. The exaggerated participation function $\mu_A^{'}(x)$ for an intuitionistic fuzzy set A is insinuated as $\mu_A^{'}(x) = \mu_A(x) + \pi_A(x)$.

Parameters entangled

- $V_A(x)$ affords spare complaisance to the user and it procreates the algorithm more pragmatic and felicitous.
- $\Pi_a(x)$: is an confederated function with sole intuitionistic fuzzy set denominate the hesitation function, $\Pi_a(x) = 1-(\mu_a(x) + V_a(x))$ for $\forall_x\, x \in A$.
- d_{ik} : is the distance betwixt data element or object x_k from the centroid u_i .

RIFCM Algorithm

STEP 1: Impute inchoate means or cluster centers v_i for the c clusters

STEP 2: Reckon $\mu_{ik}^{'}$ by the formula $\mu'_{ik} = 1 / \left[\sum_{j=1}^{c} \left(\frac{d_{ik}}{d_{jk}} \right) \right]^{2/(m-1)}$

STEP 3: For sole data element or object x_k enumerate the participation data or values in the cluster centers v_i . Let μ'_{ik} be the zenith and μ'_{jk} be the next zenith values.

STEP 4: If $\mu'_{ik} - \mu'_{jk}$ should be less than a imperative predefined extremity value then accredit x_k to the extremity of both U_i *and* U_j. Else, ordain x_k to the lower estimate of U_i , where U_i *and* U_j. are cluster midpoints or centers or means.

STEP 5: Reckon the new center or midpoint or mean for each sole cluster U_i accomplishing mutation as

$$v_i = w_{low} \left(\frac{\sum_{x_k \in \underline{B}U_i} x_k}{|\underline{B}U_i|} \right) + w_{up} \left(\frac{\sum_{x_k \in BN(U_i)} x_k}{|BN(U_i)|} \right), \text{if } \underline{B}U_i \neq \phi \text{ and } BN(U_i) \neq \phi;$$

$$v_i = \left(\frac{\sum_{x_k \in BN(U_i)} x_k}{|BN(U_i)|} \right), \text{if } \underline{B}U_i = \phi \text{ and } BN(U_i) \neq \phi; \text{ and}$$

$$v_i = \left(\frac{\sum_{x_k \in \underline{B}U_i} x_k}{|\underline{B}U_i|} \right), \text{if } \underline{B}U_i \neq \phi \text{ and } BN(U_i) = \phi;$$

STEP 6: Reiterate steps 2 to 5 until consolidation.

The above algorithm is bestowed for clustering recumbent layout. There are multifarious soft computing clustering algorithms which can accord clustering. Hard K-means can be used for clustering but it doesn't handle

impreciseness and vagueness in data. Fuzzy c-means can be used for clustering but it does not handle vagueness of data. Rough set clustering algorithm handles impreciseness and vagueness of data but to cluster data inducing participation and non- participation intellection intutionistic fuzzy set is needed. Therefore to helve all predicaments efficiently while clustering recumbent layouts hybrid IRFCM is used.

The clustering intellection can be used as follows. For example if a patron wants to find out which boutiques have similar total sales of items. The recumbent layout consists of data of boutiques with items sold for each month then to find out which boutiques have sold similar total items, IRFCM clustering algorithm can be used. The recumbent layout can be taken as input directly from database for IRFCM clustering from and groups boutiques haves similar total items sold into clusters as output. So the patrons can scrutinize which boutiques sales are good, average, bad, best, excellent etc and make required decisions for business and purchases from clustering output.

IV. CONCLUSION

By using recumbent aggregations recumbent layouts can be contrived which are precisely commodious for data mining errand or province. Onus task data preprocessing is evaded while using recumbent layouts for data mining functionalities. The reinforcement of hard, rough and fuzzy clustering intellections untidily handles surmised, imperfect, inexplicable, uncertain data. RIFCM clustering algorithm is excogitated in the paper handing surmised, imperfect, in explicable, uncertain data which also interpolates hesitation constituent for membership of data, so voluminous gamut of subsisting life standpoints can be maneuvered efficaciously. Culling recumbent layouts in data mining errand is useful for patrons in business burgeoning. Output of clustering of recumbent layouts is propitious for patrons to assay, decision making for procuring customer enchantment and business burgeoning. Missing data is not managed in recumbent layout so new fangled intellections can be used in future to maneuver missing data

REFERENCES

1. *B.K Tripathy, Anurag Tripathy, k. Govindarajulu, Rohan Bhargav, " On kernel based RIFCM algorithm and comparative Analysis," Adavanced computing Networking and Informatics-volumne 27 of the series Smart Innovation, System and Technologies*
2. *G. Bhargava, P. Goel, and B.R. Iyer, "Hypergraph Based Reorderings of Outer Join Queries with Complex Predicates, "Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '95), pp. 304-315, 1995.*
3. *J.A. Blakeley, V. Rao, I. Kunen, A. Prout, M. Henaire, and C. Kleinerman, ".NET Database Programmability and Extensibility in Microsoft SQL Server," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '08), pp. 1087-1098, 2008.*
4. *J. Clear, D. Dunn, B. Harvey, M.L. Heytens, and P. Lohman, "Non- Stop SQL/MX Primitives for Knowledge Discovery," Proc. ACM SIGKDD Fifth Int'l Conf. Knowledge Discovery and Data Mining (KDD '99), pp. 425-429, 1999.*
5. *E.F. Codd, "Extending the Database Relational Model to Capture More Meaning," ACM Trans. Database Systems, vol. 4, no. 4, pp. 397-434, 1979.*
6. *C. Cunningham, G. Graefe, and C.A. Galindo-Legaria, "PIVOT and UNPIVOT: Optimization and Execution Strategies in an RDBMS," Proc. 13th Int'l Conf. Very Large Data Bases (VLDB '04), pp. 998-1009, 2004.*
7. *C. Galindo-Legaria and A. Rosenthal, "Outer Join Simplification and Reordering for Query Optimization," ACM Trans. Database Systems, vol. 22, no. 1, pp. 43-73, 1997.*
8. *H. Garcia-Molina, J.D. Ullman, and J. Widom, Database Systems: The Complete Book, first ed. Prentice Hall, 2001.*
9. *G. Graefe, U. Fayyad, and S. Chaudhuri, "On the Efficient Gathering of Sufficient Statistics for Classification from Large SQL Databases," Proc. ACM Conf. Knowledge Discovery and Data Mining (KDD '98), pp. 204-208, 1998.*
10. *J. Gray, A. Bosworth, A. Layman, and H. Pirahesh, "Data Cube: A Relational Aggregation Operator Generalizing Group-by, Cross- Tab and Sub-Total," Proc. Int'l Conf. Data Eng., pp. 152-159, 1996.*
11. *J. Han and M. Kamber, Data Mining: Concepts and Techniques, first ed. Morgan Kaufmann, 2001.*
12. *G. Luo, J.F. Naughton, C.J. Ellmann, and M. Watzke, "Locking Protocols for Materialized Aggregate Join Views," IEEE Trans. Knowledge and Data Eng., vol. 17, no. 6, pp. 796-807, June 2005.*
13. *C. Ordonez, "Horizontal Aggregations for Building Tabular Data Sets," Proc. Ninth ACM SIGMOD Workshop Data Mining and Knowledge Discovery (DMKD '04), pp. 35-42, 2004*