Volume 4, Issue 5, May 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Credit Card Fraud Recognition by Modifying K-Means

MAHESH SINGH
CSE Department.
Maharshi Dayanand University, Rohtak

AASHIMA
CSE Department.
Maharshi Dayanand University, Rohtak

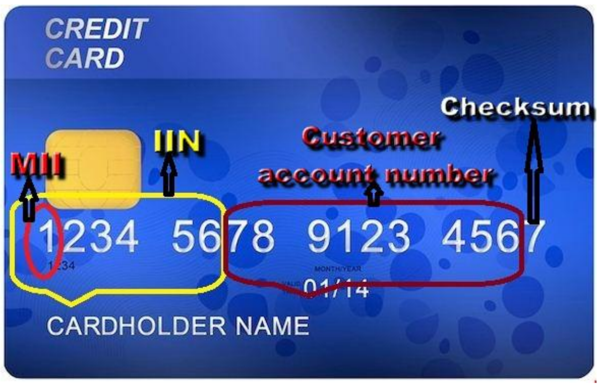
SANGEETA RAHEJA
CSE Department.
Maharshi Dayanand University, Rohtak

Abstract- Credit card is now days used by many people. No one is having time, so most of the people rely on online shopping. Online shopping is the best way for shopaholic people. But problem is many people try to use other credit cards by just altering the digits. To avoid this credit card's number is made by using the algorithm. So every system accepting credit card number should have mechanism to identify correct card number and avoid usage of fraud credit card number. By amending the K-means algorithm, fraud credit card numbers can be identified.

Keywords- Centroids, Clustering, Data Mining, K-Means, Luhn

I. INTRODUCTION

Credit Card numbers are (most times) 13 to 16 digit numbers which are protected by a special numerical condition, called Luhn check. The Luhn algorithm or Luhn method, also known as the "modulus 10" or "mod 10" algorithm, was made in the 1960s as a method of checking identification numbers. It is a simple checksum method used to validate a variety of account numbers, as credit card numbers and Canadian Social Insurance Numbers. Many of its notoriety comes from credit card companies' adoption of it shortly after its creation in the late 1960s by IBM scientist Hans Peter Luhn (1896–1964). The algorithm is in the public domain and is in wide use today. It is not projected to be a cryptographically safe hash function; it protects from unsystematic error, not malicious attack. Most credit cards and many government identification numbers use the algorithm as a simple method of distinguishing valid numbers from collections of random digits. For fraud detection, Luhn algorithm is to be checked against each card. To avoid this, K-mean algorithm is applied with some extension and epochs.



Credit Card Details

II. DATA MINING

Data mining is the core of knowledge discovery process Data Mining, also popularly known as Knowledge Discovery in Databases (KDD), refers to the nontrivial extraction of implicit, prior unknown and potentially useful information from data in databases. Data mining refers to deriving or "mining" knowledge from large amount of data.

Data Mining Architecture

Based on this view, the architecture of a typical data mining system may have the following major components :

Data Warehouse: Database, Data Warehouse, World Wide Web, or other information storehouse: This is one or a set of databases, data warehouses, spreadsheets and other kinds of information storehouse. Data cleaning and data integration methods may be applied on the data. The database or data warehouse server is responsible for fetching the relevant data, according to the user's data mining request.

Knowledge Base: This is the domain knowledge that is used to guide the search or evaluate the interestingness of resulting patterns. Such knowledge can contain concept hierarchies, used to categorize attributes or attribute values into different levels of abstraction. Knowledge like user beliefs, which can be utilized to assess a pattern's interestingness based on its unexpectedness, may be included also. Other examples of domain knowledge are additional interestingness constraints or thresholds, and metadata (e.g., describing data from multiple heterogeneous sources).

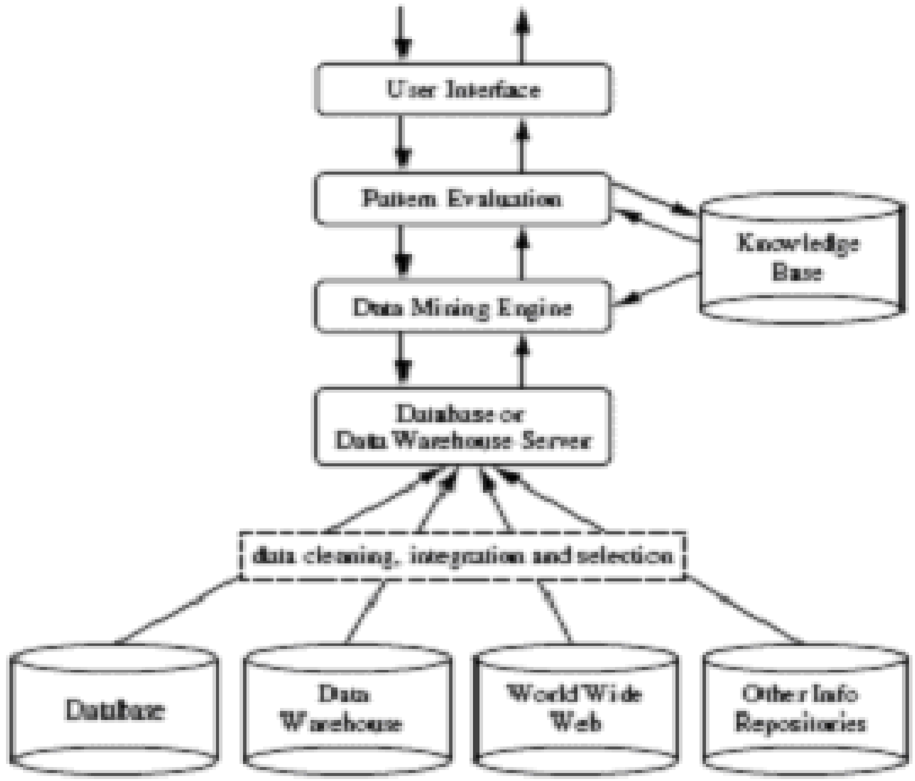


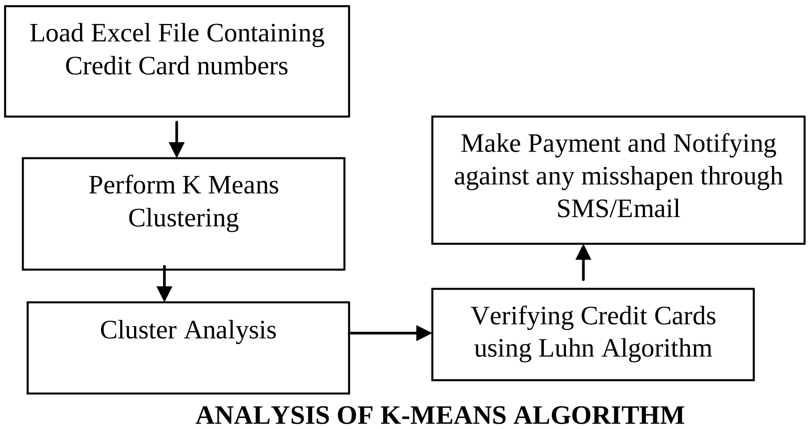
Fig 1: Architecture of Data Mining System

Data Mining Engine: This is necessary to the data mining system and ideally consists of a set of functional modules for tasks such as characterization, association and correlation analysis, classification, prediction, cluster analysis, outlier analysis, and evolution analysis.

Pattern Evaluation Module: This component typically employs interestingness measures and interacts with the data mining modules so as to *focus* the search toward interesting patterns. It may apply interestingness thresholds to filter out discovered patterns. on the other hand, the pattern evaluation module may be integrated with the mining module, depending on the execution of the data mining method used. For efficient data mining, it is very much recommended to push the evaluation of pattern interestingness as deep as possible into the mining process so as to confine the search to only the interesting patterns.

User Interface: This module communicates between users and the data mining system, providing the user to interact with the system by specifying a data mining query or task, giving information to help focus the search, and acting exploratory data mining based on the intermediate data mining results. Also, this component allows the user to browse database and data warehouse schemas or data structures, estimating mined patterns, and visualize the patterns in different forms.

III. BASIC K-MEAN CLUSTERING TECHNIQUE


Load Excel File Containing Credit Card numbers
Perform K Means Clustering
Cluster Analysis
Verifying Credit Cards using Luhn Algorithm
Make Payment and Notifying against any misshapen through SMS/Email

ANALYSIS OF K-MEANS ALGORITHM
Fig 2: Basic K-Means Clustering Technique

Suppose that a dataset of n data points l_1, l_2, …, l_n such that each data point is in R^d, the problem of extracting the minimum variance clustering of the dataset into k clusters is that of finding k points $\{m_j\}$ (j=1, 2, …, k) in R^d such that

$$\frac{1}{n} \sum_{i=1}^{n} [\min_{j} d^2(x_i, m_j)] \qquad \text{(i)}$$

is minimized, where $d(x_i, m_j)$ denotes the Euclidean distance between x_i and m_j. The points $\{m_j\}$ (j=1, 2, …, k) are known as cluster centroids. The thing in above Equation is to find k cluster centroids, so that the average squared Euclidean distance (mean squared error, **MSE**) among a data point and its nearest cluster centroid is minimized.

The k-means algorithm provides an easy method to implement approximate solution to this Equation. The cause of the popularity of k-means is ease and simplicity of execution, scalability, speed of convergence and adaptability to sparse data.

The k-means algorithm can be thought of as a gradient descent procedure, which initiate at starting cluster centroids, and iteratively modify these centroids to decrease the objective function in the Equation listed above. The k-means always congregate to a local minimum. The particular local minimum search depends on the starting cluster centroids. The problem of searching the global minimum is NP-complete. The *k*-means algorithm modifies cluster centroids till local minimum is found.

A. k-Means Clustering Algorithm

```
1. MSE = largenumber;
2. Select initial cluster centroids $\{mj\}_j^k=1$;
3. Do
4. OldMSE = MSE;
5. MSE1 = 0;
6. For j=1 to k
7. mj=0; nj=0;
8. endfor
9. For i = 1 to n
10. For j = 1 to k
11. Compute squared Euclidean distance $d^2$ (xi, mj);
12. endfor
13. Find the closest centroid mj to xi;
14. mj = mj+xi; nj = nj+1;
15. MSE1 = MSE1 + $d^2$ (xi, mj);
16. endfor
17. For j = 1 to k
18. nj = max (nj, 1); mj = mj/nj;
19. endfor
20. MSE = MSE1;
while (MSE < OldMSE)
```

Before the *k*-means algorithm converges, distance and centroid computations are done while loops are executed a number of times, say l, where the positive integer l is known as the number of k-means iterations. The precise value of l varies according to the initial starting cluster centroids even on the same dataset.

So the time complexity of the algorithm is O(nkl), where n is the total number of objects in the dataset, k is the required number of clusters we identified and l is the number of iterations, $k \le n$, $l \le n$.

B. Limitations of original k-means algorithm

- It is computationally very costly as it involves several distance calculations of each data point from all the centroids in each iteration.
- The final cluster results greatly depend on the selection of initial centroids which causes it to converge at local optimum.

IV. AN EFFICIENT ENHANCED K-MEAN CLUSTERING TECHNIQUE

The following algorithm makes k-means more efficient by removing the first limitation as it limits the number of computations to some extent. The scheme makes k-means more efficient, mainly for dataset containing large number of clusters. Each iteration in the k-means algorithm computes the distances between data point and all centers; this is computationally very costly especially for huge datasets. Therefore, we do can benefit from earlier iteration of k-means algorithm. For every data point, we can keep the distance to the nearest cluster. At the next iteration, we calculate the distance to the previous nearest cluster. If the latest distance is less than or equal to the earlier distance, the point remains

in its cluster, and there is no need to calculate its distances to the other cluster centers. This saves the time required to calculate distances to k−1 cluster centers.
Following fig. explains the idea.
Fig. 3 represents the dataset points and the initial 3 centroids.

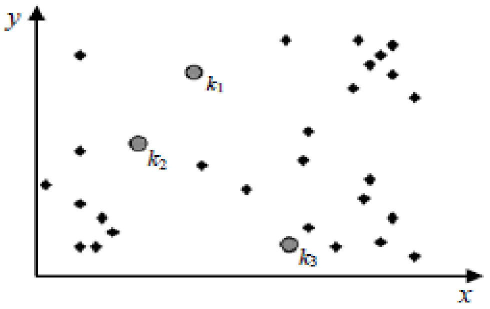


Fig 3: Initial Centroids to a dataset

Fig.4 shows points distribution over the initial 3 centroids, and the new centroids for the further iteration.

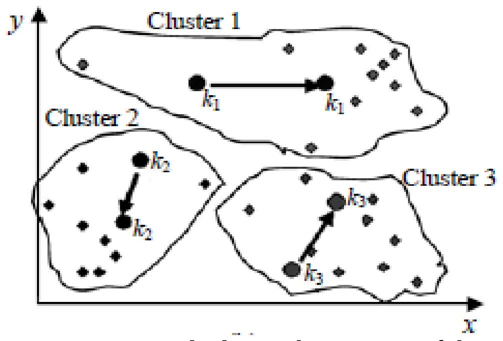


Fig 4: Recalculating the position of the centroids

Fig. 5 shows the final clusters and their centroids.

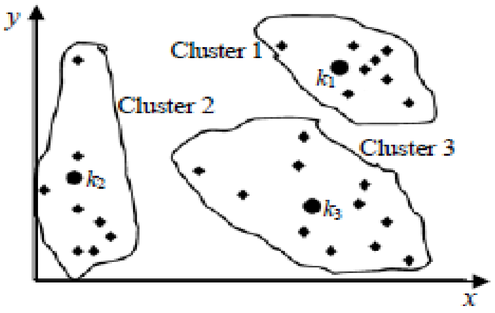


Figure 5: Final position of the Centroids

When we examine Fig. 4 in Clusters 1, 2 we make a note of that, the most points become closer to their new center, just one point in Cluster 1, and 2 points in Cluster 2 will be redistributed (their distances to all centroids must be calculated), and the final clusters are presented in Fig. 6 Based on this idea, the proposed algorithm saves a lot of time.
In the proposed method, we write two functions. The first function is basic function of the k-means algorithm, that calculate the nearest center for each data point, by calculating the distances to the *k* centers, and for every data point keeps its distance to the nearest center.
The first function is shown in Algorithm 5.1., which is similar to that in Algorithm 4.1, with adding a simple data structure to keep the distance between each point and its closest cluster. This function is called distance().

Algorithm 5.1:

An Efficient amended k-Mean Clustering Algorithm: First Function

```
Function distance()
//assign each point to its nearby cluster

1. For i = 1 to n
2. For j = 1 to k
3. Compute squared Euclidean distance $d^2$(xi, mj);
4. endfor
5. Find the closest centroid mj to xi;
6. mj = mj+xi; nj = nj+1;
7. MSE = MSE + $d^2$(xi, mj);
```

```
8. Clusterid[i] = number of the closest centroid;
9. Pointdis[i] = Euclidean distance to the closest centroid;
10. endfor
11. For j = 1 to k
12. mj = mj/nj;
13. endfor
```

In Line 3 the function finds the distance between point number i and all k centroids. Line 5 searches for the nearest centroid to point number i, say the nearest centroid is number j. Line 6 sums point number i to cluster number j, and raise the count of points in cluster j by one. Lines 8 and 9 are utalized to enable us to execute the proposed idea; these two lines keep the number of the closest cluster and the distance to the nearest cluster. Line 12 does centroids recalculation.
The other function is shown in Algorithm5.2, is called distance_new(). Line 1 finds the distance between the current point i and the new cluster center assigned to it in the earlier iteration, if the computed distance is smaller than or equal to the distance to the old center, the point remains in its cluster that was assigned to in previous iteration, and there is no need to calculate the distances to the other k−1 centers. Lines 3~5 will be executed if the computed distance is larger than the distance to the old center, this is for the reason that the point may change its cluster, so Line 4 calculates the distance between the current point and all k centers. Line 6 finds for the
nearest center, Line 7 assigns the current point to the closest cluster and increases the count of points in this cluster by one, Line 8 modifies mean squared error. Lines 9 and 10 continue the cluster id, for the current point given to it, and its distance to it to be utilized in next call of that function (i.e. next iteration of that function). This information is reserved in Line 9 and Line 10 allows this function to reduce the distance calculation required to assign each point to the nearest cluster, and this allows the function faster than the function distance in Algorithm 5.1

Algorithm 5.2:
An Efficient Amended k-Mean Clustering Algorithm : Second Function

```
Function distance_new()
//Assign each point to its close cluster

1.  For i = 1 to n
      Calculate squared Euclidean distance
      $d^2$ (xi, Clusterid[i]);
      If ($d^2$ (xi, Clusterid[i] ) <= Pointdis[i] )
      Point remain in its cluster;
2.    Else
3.    For j = 1 to k
4.    Evaluate squared Euclidean distance $d^2$(xi, mj);
5.    endfor
6.    Find the closest centroid *mj* to *xi*;
7.    mj = mj+xi; nj = nj+1;
8.    MSE = MSE + $d^2$(xi, mj);
9.    Clustered[i] = number of the closest centroid;
10.   Pointdis[i] = Euclidean distance to the closest centroid;
11.   endfor
12.   For j = 1 to k
13.   mj = mj/nj;
14.   endfor
```

Complexity
As discussed before, the k-means algorithm converges to local minimum. previous to the k-means converges, the centroids computed number of times, and the entire points are assigned to their closest centroids, i.e., entire redistribution of points according to new centroids, this takes O(nkl), in which n is the number of points, k is the number of clusters and l is the number of iterations.
In the enhanced k-means algorithm, to obtain original clusters, this process requires O(nk). Here, some points stays in its cluster, the others shift to another cluster. If the point remains in its cluster this require O(1), or else require O(k). If we suppose that half points move from their clusters, this requires O(nk/2), as the algorithm converges to local minimum, the number of points shifted from their clusters decreases in every iteration. So we expect the total cost is
nk Σ 1/i.
Even for large number of iteration,
i=1
l
nk Σ 1/nk is much less than **nkl**. So the cost of using enhanced k-means algorithm
i=1
approximately is **O(nk),** not **O(nkl)**.

V. CONCLUSION:

The proposed enhanced algorithm is easy to implement and it proves to be a better method to determine the initial centroids to be used in the k-means clustering algorithm. As the end clustering results of the k-mean clustering method are highly dependent on the selection of initial centroids , so there should be a systematic method to determine the initial centroids which makes the k-mean algorithm to converge in global optima and unique clustering results. This requirement is fulfilled by the proposed algorithm. Besides solving the problem of non-unique results, our proposed algorithm is also widely applicable to different types to problems. The problems with uniform as well as the problems with non-uniform distribution of data points are better addressed by our proposed algorithm.

Our proposed algorithm tries to enhance the k-means clustering algorithm by eliminating one of its drawback. But still lots of work needs to be done to enhance the k-means algorithm to a greater extent. K-means can be applied on numerical data only. But in day to day life we encounter scenarios with a combination of both numerical and categorical data values. So future work can be carried out in the direction of making the k-means algorithm applicable for mixed type of data.

The proposed algorithm is easy to implement and it proves to be a better method to determine the validity of creditcard. algorithm to be used as a validity criterion for a given set of numbers. Almost all credit card numbers are extracted following this validity criterion…also called as the Luhn check or the Mod 10 check. It went without saying that the Luhn check is also used to verify a given existing card number. If a credit card number does not assure this check, it is **not** a valid number. So future work can be carried out in the direction of making the luhn algorithm applicable for different length of credit card numbers.

REFERENCES

Jiawei Han, MichelineKamber; Data Mining: Concepts and Techniques [1]

M. Halkidi, Y.Batistakis, M. Vazirgiannis; Clustering algorithms and validity measures : 0-7695-1218-6/01 2001 IEEE[2]

Rui Xu; Survey of Clustering Algorithms : IEEE Tansactions on Neural Networks, Vol 16, No. 3, May 2005[3]

Tian Zhang, Raghu Ramakrishnan, and MironLivny; BIRCH: An Efficient Data Clustering Method for Very Large Databases: Technical report, Computer sciences Dept., Univ. of Wisconsin Madison, 1996.[4]

Vladimir Estivill-Castro; Why so many clustering algorithm- A Position Paper: SIGKDD Explorations: Vol 4, Issue 1[5]

HesamIzakian;Clustering Categorical data using a Swarm-based method: 978-1-4244-5612-3/09 2009 IEEE [6]

AristidisLikas; the global k-means clustering algorithm: Pattern Recognition 36(2003).[7]

Rodrigo G.F. Soares; An Evolutionary approach for the Clustering data Problem : 978-1-4244-1821-3/08 2008 IEEE [8]

Yinghua Zhou, Hong Yu; A Novel k-means Algorithm for clustering and outlier detection: Second International conference on future information technology and management engineering: 2009 IEEE[9]

SudiptoGuha; ROCK: A robust clustering algorithm for categorical attributes: 0-7695-0071-4/99 1999 IEEE[10]

ĐinhQuangHuy and ĐinhMạnhTường :LINK-CONNECTED: A New Approach Of Clustering Algorithm For Categorical Attributes : Department of Computer Science and Engineering, Harbin institute of Technology, P.R.China, 2005.[11]

Maria Halkidi, YannisBatistakis, MichalisVazirgiannis; On Clustering Validation Techniques: Journal of Intelligent Information Systems, Vol. 17, pp. 107–145, 2001.[12]

T,Chiu, D.Fang, J.Chen, Y.Wang : A Robust and Scalable Clustering Algorithm for Mixed type attributes in large Database environment : Int.Conf. on Knowledge Discovery and Data Mining, pp. 263-268, 2001.[13]

C. Li, G. Biswas; Unsupervised Learning with Mixed Numeric and Nominal Data: IEEE Transaction On Knowledge and Data Engineering, Vol. 14, no. 4, 2002.[14]

SushmitaMitra, Sankar K. Pal and PabitraMitra; Data Mining in Soft Computing Framework: A Survey: in IEEE Transactions on Neural Networks, Vol. 13, No. 1, January 2002.[15]

P. Pantel; Clustering by Committee: Ph.d. dissertation, Department of Computing Science, University of Alberta, 2003[16]