# Clustering Experiments on Big Transaction Data for Market Segmentation

Ashishkumar Singh[1] Grace Rumantir[1]
[1]Faculty of Information Technology
Monash University

Annie South
Australia New Zealand Bank
Australia

Blair Bethwaite
Monash eResearch Centre
Monash University

## ABSTRACT

This paper addresses the Volume dimension of Big Data. It presents a preliminary work on finding segments of retailers from a large amount of Electronic Funds Transfer at Point Of Sale (EFTPOS) transaction data. To the best of our knowledge, this is the first time a work on Big EFTPOS Data problem has been reported. A data reduction technique using the RFM (Recency, Frequency, Monetary) analysis as applied to a large data set is presented. Ways to optimise clustering techniques used to segment the big data set through data partitioning and parallelization are explained. Preliminary analysis on the segments of the retailers output from the clustering experiments demonstrates that further drilling down into the retailer segments to find more insights into their business behaviours is warranted.

## Categories and Subject Descriptors

H.3.3 [**Information Storage And Retrieval; Information Search and Retrieval**]: *Clustering and Information filtering*; H.2.8 [**Database Management; Database Application**]: *data mining*

## General Terms

Design, Experimentation

## Keywords

Data Reduction, Parallelisation, Clustering, Market Segmentation

## 1. INTRODUCTION

The concept of market segmentation was first introduced by [1] and defined as a "process of subdividing a market into distinct subsets of customers that behave in the same way or have similar needs. Each subset may conceivably be chosen as a market target to be reached with a distinctive marketing strategy" [2].

Data mining techniques, e.g. clustering, have been used to build market segmentation models using transaction data as input (e.g. [3],[4],[5],[6],[7],[8]). These techniques are designed to be scalable. However, the learning algorithms typically involve multiple sequential passes on the data set, therefore when dealing with Big Data, it is necessary to implement mechanisms to ensure that the development of the models can be completed within a reasonable amount of time and resources.


BigDataScience '14, August 04 - 07 2014, Beijing, China

http://dx.doi.org/10.1145/2640087.2644161

This paper outlines our experience in acquiring, secured-storing and processing Electronic Funds Transfer at Point Of Sale (EFTPOS) data from one of the four major banks in Australia. As the data is commercial in confidence in nature, there is privacy and security provisions that need to be put in place before the data can be released to Monash University for research purposes.

Based on what we have read in the literatures, this is the first time Big EFTPOS Data has been released for research purposes. Hence this is also the first time, market segmentation experiments on Big EFTPOS Data have been performed. The challenge in this project is most notably on the Volume characteristic of Big Data.

This paper is organized as follows: Section 2 explains the suitability of RFM analysis for market segmentation on EFTPOS data; Section 3 gives the workflow followed by the work done for this paper; Section 4 outlines how the data is acquired and stored for processing; Section 5 describes the data reduction process using the RFM analysis; Section 6 explains how the clustering algorithms are implemented for this Big Data project; Section 7 discusses the results of the experiments and Section 8 presents the conclusion and future work of the project.

## 2. MARKET SEGMENTATION EXPERIMENTS ON EFTPOS DATA

In Marketing, RFM (Recency, Frequency, Monetary value) analysis [9] is a popular technique used to analyse customer behaviour for the purpose of market segmentation (see e.g. [10],[11],[12],[13],[14],[15]). By observing a set of transactions each customer has made over a period of time, RFM analysis aims at grouping customers with similar buying patterns based on the following characteristics of each customer during the observation period:

- Recency - How recently did this customer make his/her last purchase?
- Frequency - How often did this customer make purchases?
- Monetary value - How much did this customer spend?

The market segmentation experiments in this paper use RFM analysis and clustering techniques to group retailers sharing common characteristics. To the best of our knowledge, there has only been one previous work (i.e. [16]) on EFTPOS data but on a much smaller scale of data set (30,524 Point of Sale terminals with 1,030,120 total transactions), hence no Big Data consideration has needed to be covered.

Apart from being a popular pattern recognition technique in Marketing, RFM analysis can be seen as a data reduction technique when dealing with Big Data. In this project, the size of the data set has been reduced from 130GB of raw transaction data to 80 MB of RFM values of the retailers.

## 3. BIG DATA MARKET SEGMENTATION ARCHITECTURE

**Figure 1** shows the workflow of the work reported in this paper. The data we have been obtaining for this project is unique, as this is the first time a major bank has released their EFTPOS data for research in the world, the first time for this particular major bank in Australia released their financial data asset for research. The data is received by Monash University already removed from any sensitive information with certain provisions needed to be followed with respect to its transfer and secure storage in the University computing environment.

Due to the sheer size of the data set, data reduction process needs to be undertaken with the use of hash table and the clustering algorithms used for the retailer segmentation experiments need to be optimized.

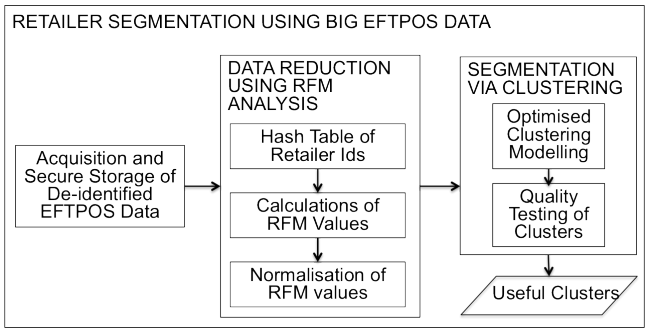


Figure 1. Processing architecture used for retailer segmentation from large EFTPOS data

## 4. DATA ACQUISITION AND STORAGE

The EFTPOS data for this research has been periodically extracted from the operational environment of the bank by their data warehouse team.

Because data is valuable asset for an organization, before the bank can release the EFTPOS data to the University, written confirmation from the bank's Information Security Office has to be obtained that:

1. The data has been sufficiently obfuscated before it leaves the bank
2. The security provisions on the Monash University computing environment are sufficient to host the data
3. The data is transmitted in a secure manner from the bank to the University.

The EFTPOS data contains 55 variables, some of which are confidential and sensitive in nature. To satisfy Point 1 of the data release conditions, these attributes, e.g. retailer names and ids, credit/debit card numbers, pin numbers, customer's billing address, etc., have been removed, masked or changed. These operations have been done in a consistent manner to ensure the bank can convert the data back to the original form once the market segmentation modeling has been completed.

To satisfy Point 2 on security provisions for the data, the following agreements have been reached:

- Access to the data is limited to named staff and students with password protection
- Transmission to third parties is prohibited.
- Data is to be properly destroyed on all storage media at the end of the project
- Data is not automatically backed up and cannot be copied
- The storage environment should have appropriate security controls in place such as firewall, Anti-Virus, logging and monitoring, IDS/IPS etc.

To satisfy Point 3 on secure transmission of the data, an agreement has initially been made to establish a secure Virtual Private Connection (VPN) from the University secure server to the bank's database where the extracted EFTPOS data resides. However, due to the bank's firewall policies, no VPN connection could be established. Data storage on a cloud environment is also against the bank's Information Security Office policies. It has then been decided that the data is to be fetched manually. The raw de-identified transactional data was first encrypted and subsequently transferred to an encrypted external hard disk drive which is then mounted onto the Monash Large Research Data Store (LaRDS).

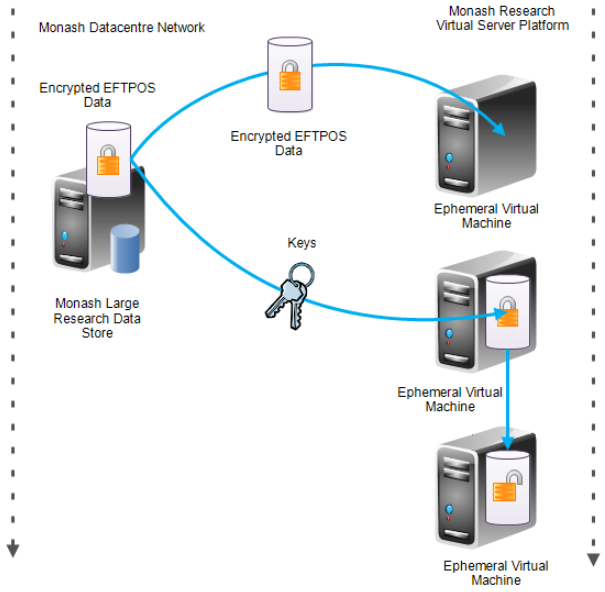


Figure 2. Secure server storage and access mechanism of EFTPOS data at Monash Large Research Data Store

**Figure 2** shows the server setup to host the EFTPOS data at the Monash Large Research Data Store (LaRDS). The server consists of Intel Xeon and AMD Opteron CPUs of various clock speeds with 16 cores, 32 GB RAM and 500 GB hard disk size.

The encrypted EFTPOS data is transferred to the server at Monash Large Research Data Store (LaRDS). Authorised access and analysis of the data is performed through the Ephemeral Virtual Machines created on the Monash research virtual server platform. Hence two levels of authentication are required to access the data, first, via password-protected logon to the virtual machines and second, via encryption keys to decrypt the stored data.

## 5. DATA REDUCTION USING RFM ANALYSIS

After data acquisition and its secure storage, RFM analysis is done to reduce the data, involving the creation of a hash table for the `Retail Ids`, the calculations and then normalization of the RFM values of the retailers.

## 5.1 Hash Table of `Retailer Ids`

This paper reports the preliminary work we have done on the EFTPOS transaction data that the bank has been extracting on our behalf. For this first set of experiments, we use data for a total period of 18 days (from 19-September-2013 to 07-October-2013). We use these data to find clusters of retailers with similar characteristics. The 18 daily transaction files contain a total of 77.5 million transaction records for over 1 million unique retailer accounts. This volume of data makes even the basis operations, such as finding the total monetary amount of each retailer in the data set, very time consuming and resource intensive.

To overcome this problem, hash table is used. The hashing function on Java (class `Hashtable`) is used to convert each unique `Retailer Id` to a hash code (key) of a bucket as shown in **Figure 3**. Hence the buckets of retailer ids along with their hash code comprise the hash table. Using this hash table, the calculations of the monetary values and the frequency of transactions for each retailer as well as the search for each retailer's most recent transaction become a lot quicker than having to loop through the raw dataset sequentially.

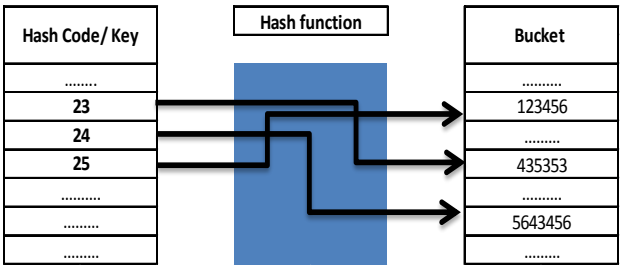


Figure 3. Hash table for `Retailer Ids`

## 5.2 RFM Analysis

Not all of the 55 attributes in the raw daily transaction data files are required for the calculations of the RFM values for each retailer. To reduce memory requirement of the experiments, smaller daily data files have been created by copying the values of relevant attributes, such as `Time`, `Retailer Id`, `Amount`.

### *5.2.1 Recency*

In this project, the Recency value for each retailer is the time interval between a global datum and his/her latest transaction. A retailer with a smaller Recency value is seen as more current in his/her business activities than a retailer with a bigger Recency value. The global datum is chosen as midnight after the last transaction day in the data set (i.e. the midnight of 8th October 2013 (00:00:000000 in HH:MM:SSSSSS format). Hence, all the Recency values are calculated from midnight of 8th October 2013 backward.

There were many attributes related to the recency of a transaction in the raw data set, e.g. the time a transaction occurred, recorded, etc. In this project, the time and date when the transaction occurred are used for the calculations of Recency.

The transaction records in each of the 18 daily transaction files are not sorted based on time. Hence, for each retailer, each transaction in each daily transaction file (starting from the latest date) needs to be checked to find the one that is the latest in time of the day. Making use of the hash table of the `retailer ids`, the search for the Recency values for all of the retailers are done in parallel for each day where the Recency values are stored in an array. Compared to the calculations of Frequency and Monetary values, the calculation of Recency is the most time consuming and resource intensive.

The following is the pseudo code of the program used to calculate the Recency values of the retailers for a given day making use of the `get()` method of the Java class `HashTable` to fetch the key representing the `Retailer Id` in the hash table.

*Create a Recency array of size equal to the number of retailers*

*Initialise each cell of the array to zero*

*FOR each transaction record*

*Use the get method of class HashTable to retrieve the key of a retailer id*

*Store the key in variable Index*

*Store the transaction time to variable Temp_Time*

*IF Temp_Time is greater than Recency[Index]*

*Set Recency[Index] to Temp_Time*

*End IF*

*End FOR*

### *5.2.2 Frequency*

Frequency is the number of transactions belonging to a retailer in the data set. Matlab's tabulate function is used to calculate the Frequency value for each retailer:

*Frequency_table = tabulate(Retailer_id)*

where Retailer id is an array containing retailer values.

### *5.2.3 Monetary Value*

Monetary value refers to the total amount of money a retailer has accumulated from the transactions in the data set. In order to save time and reduce processor load, each daily transaction file is divided into 20 small subsets and the calculation is done on all the subsets (i.e. 20x18=360 subsets) in parallel. The hash table of the `Retailer Ids` is used to calculate the Monetary value of each retailer. The following is the pseudo code of the program run on each subset. The Monetary value of each retailer is the sum of his/her monetary values in all of the 360 subsets.

*Create a Monetary value array of size equal to the number of retailers.*

*Initialise each cell of the array to zero.*

*FOR each transaction record*

*Use the get method of hashtable class to retrieve the key of a retailer id*

*Store the key in variable Index*

*Store the transaction amount to variable Temp_Money*

*Add Temp_Money to Money[Index]*

*End FOR*

## 5.3 Normalisation of RFM Values

Normalization or standardization is a process by which values of all attribute are made into proportion with one another [18]. Normalization is used so that attribute values can be compared fairly in terms of information content.

Normalisation is very important in this clustering project as clustering algorithms use various distance measures (e.g. Euclidean Distance) between attribute values. An attribute with values covering a large range, like retailer's Monetary values, may dominate over another attribute with smaller range of values, like retailer's Recency and Frequency values.

To alleviate such a problem, in this project, the min-max normalization technique is used where the values of all three attributes are normalised into a range between 0 and 100. This effectively means the transformed values of each attribute are expressed as percentage of the original values:

$$NormX_i = 100 \times \left(\frac{X_i - X_{min}}{X_{max} - X_{min}}\right)$$

In the above equation, $NormX_i$ is the normalized value of $X_i$. $X_{\max}$ is the maximum value of attribute X and $X_{\min}$ is the minimum value of attribute X.

# 6. CLUSTERING EXPERIMENTS ON BIG DATA

Once the intermediate data set containing the RFM values of each retailer has been created, it is used to create clustering models. Clustering aims to organize data into groups with high intra-cluster similarity and low inter-cluster similarity. At this preliminary stage of the project, K-means and Agglomerative Hierarchical Clustering techniques have been chosen as they are to-date the most popular in the literatures (see [17]). We have chosen to create 19 clusters from the data set as explained in [17].

Clustering is a highly iterative process where the objective function is calculated and re-calculated for each transaction record for a predefined number of iterations or until a predefined threshold has been reached. Therefore clustering experiments on a large data set is a very time consuming and processor intensive process. The process needs to be optimized so that it can be completed at a reasonable cost and in a reasonable time.

## 6.1 K-Means Clustering on Big Data

K-means algorithm aims to partition a set of objects into K number of clusters. The objective function of K-means is to minimise the distance between the objects in a cluster and maximise the distance between clusters. The following is the pseudo code of the K-means algorithm:

```
1  Set the initial seed/center points
2  DO
3     FOR each data point in the data set do
4        Find the nearest seed point using the distance metric
5        Assign the data point to the cluster with the nearest
         seed point
6     End FOR
7     Calculate the mean of the data points in each cluster
8     Assign these mean values as the new center points
9  WHILE at least one data point changes clusters
```

The for-loop in the K-means algorithm (lines 3 to 6 in the pseudo code) is ideal for parallelisation as it can be implemented in parallel on subsets of the data set. For the 16 core processors we currently have access to, the data points are divided into 13 subsets and the for-loop are run on all of these subsets simultaneously in parallel as illustrated in **Figure 4**.

After the initialization of the cluster centroids, the memberships of each cluster in each of the 13 subsets are calculated in parallel. So, each cluster has members in each of the 13 subsets. The new centroid of each cluster is calculated based on all of the cluster members in all of the 13 subsets. This parallel version of the K-means algorithm yields identical final set of clusters as the sequential K-means algorithm with less processing time.

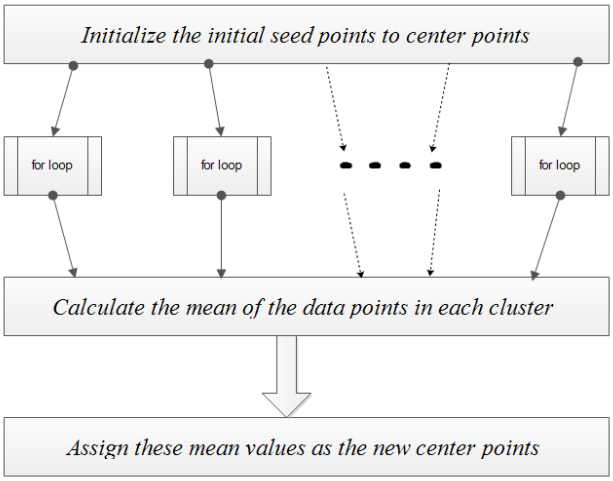


Figure 4. Parallelisation of the execution of the K-means algorithm

## 6.2 Agglomerative Hierarchical Clustering on Big Data

Agglomerative Hierarchical Clustering (AHC) method is a bottom up approach. It starts with treating each data point as a cluster. For each data point, its distance with each of the other data points are calculated and it is then merged with the data point with the shortest distance and the new cluster replaces the two data points in the distance matrix. This process continues in an iterative manner until there is only one cluster left. The time complexity for the calculations of the distance matrix is $O(n^3)$.

At this preliminary stage of the project, to reduce the time required to build the clusters, 60% of the data set are used to build the clusters using the AHC algorithm and each data point in the remaining 40% is assigned to the cluster with the shortest distance to its centroid as shown in the following pseudo code.

```
Create subset S1 by randomly selecting n data points from the
data set
Put the remaining data points into subset S2
FOR each data point/cluster i in S1 do
   Calculate the distance δ(i,j) where i ≠ j
   Put δ(i,j) in ascending order in a distance matrix
End FOR
DO
   Find the minimum value of δ(i,j)
   Merge i and j into a new cluster C and remove i and j
   Calculate the distances from C
   Update the distance matrix
```

*WHILE the maximum number of clusters is not reached or there are still more than one cluster*

*Compute the centroid of each final cluster*

*FOR each data point in $S_2$*

*Find the nearest cluster centroid*

*Assign the data point to that cluster*

*End FOR*

## 7. RESULTS AND DISCUSSION

In this preliminary work, we have parallelised just the K-means algorithm. **Table 1** compares the number of time steps required in the K-means calculations between sequential and parallel processings. Note, it is assumed that, the total time taken by any step in the for-loop done in parallel over subsets of the data is counted as the time taken by 1 step, it is not multiplied by the number subsets.

Using the formula, $T_{seq}$ and $T_{par}$, shown in this table, the time step reduction in the parallel processing is calculated for the 13 subsets of retailer data used in this paper and the result is shown in **Figure 1**.

$$TimeStepReduction = \left(\frac{T_{seq} - T_{par}}{T_{seq}}\right) \times 100\%$$

Table 1. The difference in the number of time steps required in the K-means calculations between sequential and parallel processings

| **Line number in K-means pseudo code** | **Time steps required ($r$ = number of retailers; s = number of data subsets; c = number of clusters)** | |
|---|---|---|
| | **Sequential** | **Parallelised** |
| 4 | $r \times c$ | $\frac{r}{s} \times c$ |
| 5 | r | $\frac{r}{s}$ |
| 7 | $r \times c$ | $r \times c$ |
| Total | $T_{seq} = (r \times c) + r + (r \times c)$ | $T_{par} = \left(\frac{r}{s} \times c\right) + \frac{r}{s} + (r \times c)$ |

When the K-means algorithm is parallelised over 13 data subsets, as shown in **Figure 5**, the time step reduction is the highest when the number of clusters is low. The time step reduction becomes asymptotic as the number of clusters increases. This is because, with constant number of subsets, the increase in the number of clusters will cause more clusters to process in each subset (see the time step formula for line 4 of the pseudo code of the K-means algorithm in **Table 1**).

In **Figure 6**, we can see that if the number of subsets in low (i.e. low level of parallelism), an increase in the number of clusters will not affect the time step reduction much. On the other hand, if the number of subsets is high (i.e. high level of parallelism), like with the 13 subsets done for this paper, increasing the number of clusters does asymptotically increase the time step reduction as explained using **Figure 5**.

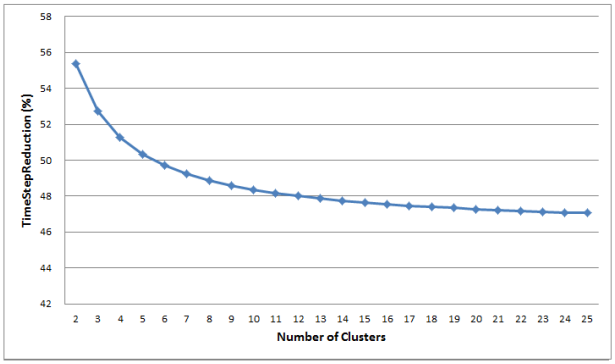


Figure 5. Time step reduction gained by parallelising the K-means algorithm. This graph is for parallel processing over 13 subsets of the retailer RFM data

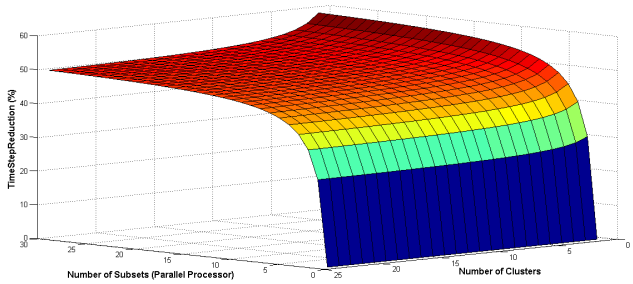


Figure 6. The time step reductions for different combinations of number of subsets on which the K-means algorithm is run in parallel and the number of clusters to be formed.

**Figure 6** also shows that increasing the number of subsets (i.e. level of parallelism) sharply increases the time step reduction at the beginning, which later becomes asymptotic at close to 60%. This is because the time saving due to parallelism is only gained in the statements in the for-loop of the K-means algorithm (pseudo code lines 4 and 5 in **Table 1**). The statement outside the parallelised for-loop (line 7 in **Table 1**) requires the same number of steps (about 50% of the total number of steps) irrespective of whether it is executed in a sequential or parallel manner.

As for the results of the clustering experiments, we demonstrate in [17], that a clustering model with 19 clusters is chosen as its optimum number of clusters. **Table 2** shows the centroids of the 19 clusters. Each of the ranges of Recency, Frequency and Monetary values is divided into 5 regions where 1 being the lowest value and 5 being the highest value. A three-digit label is assigned to the centroid of each cluster depending on the regions the R, F and M values fall in.

Lower values of Recency suggest that there are recent transactions generated by the retailers. Hence, low value of Recency coupled with high values of both Frequency and Monetary is the feature of a retailer who actively generating regular transactions. In **Table 2**, Clusters 1, 2 and 18 show such characteristics. These retailers make 34.23% of the total retailers. This retailer segment is arguably the most profitable for the bank, hence will be the segment where the bank's customer retention strategy should be concentrated on.

Table 2. Results of the clustering experiments

| Cluster Number | % | Average Recency | Average Frequency | Average Monetary Value | R,F,M labels 1 to 5 quantiles |
|---|---|---|---|---|---|
| **1** | **10.30** | **2.368704** | **0.004604** | **0.003177** | **255** |
| **2** | **15.27** | **1.018574** | **0.038917** | **0.017611** | **155** |
| 3 | 2.41 | 67.028300 | 0.000117 | 0.000333 | 523 |
| 4 | 2.30 | 71.897224 | 0.000105 | 0.000313 | 523 |
| 5 | 1.72 | 47.020206 | 0.000343 | 0.000331 | 433 |
| 6 | 3.20 | 42.105650 | 0.000293 | 0.000681 | 434 |
| 7 | 1.31 | 82.475670 | 0.000085 | 0.000163 | 522 |
| 8 | 2.14 | 76.985930 | 0.000071 | 0.000287 | 523 |
| 9 | 2.11 | 97.306885 | 0.000025 | 0.000220 | 523 |
| 10 | 2.32 | 92.311350 | 0.000067 | 0.000303 | 523 |
| 11 | 3.25 | 57.261173 | 0.000191 | 0.000473 | 523 |
| 12 | 2.79 | 62.190346 | 0.000169 | 0.000406 | 523 |
| 13 | 5.12 | 27.091051 | 0.000393 | 0.000818 | 434 |
| 14 | 4.46 | 32.134422 | 0.000318 | 0.000703 | 434 |
| 15 | 7.36 | 21.853570 | 0.000626 | 0.001149 | 434 |
| 16 | 11.78 | 17.150494 | 0.001816 | 0.002548 | 345 |
| 17 | 2.39 | 52.247010 | 0.000147 | 0.000263 | 523 |
| **18** | **8.66** | **6.925114** | **0.011165** | **0.004900** | **255** |
| 19 | 11.12 | 12.214928 | 0.002484 | 0.002045 | 245 |

Clusters 5, 6, 13, 14, 15 consist of retailers with moderately high Recency and Monetary values value with average Frequency value. These retailers are recently inactive for some time resulting in the moderate number of EFTPOS transactions. Since these retailers have the capacity to generate revenue, the bank may be want to employ a marketing strategy to provide incentives for them to improve their business activities.

Of the three attributes, Recency is seen as the most important indicator of customer loyalty. Clusters 3, 4, 7 to 12 and 17 have retailers with high Recency value and average Frequency and Monetary values. This segment of retailers, which constitutes 21% of the total retailers, has been inactive for a while, hence they may be at risk of attrition.

# 8. CONCLUSION AND FUTURE WORK

This paper presents our work on market segmentation using EFTPOS data. To the best of our knowledge, this is the first time a work on Big EFTPOS Data problem has been reported.

Due to the unique and sensitive nature of the data, we feel it would be of interest to outline how the data is acquired and stored and how it is accessed in a secure manner in the Monash e-Research environment. The popular RFM (Recency, Frequency, Monetary) analysis method is used for data reduction. The Volume problem in Big Data has already been experienced, even at this preliminary stage of the project, as we try to create clusters out of the RFM values of 1 million retailers. A way to parallelise the K-means clustering algorithm is explained with illustrations on where and how reduction in time steps to be performed can be achieved. The summary of the results of the clustering experiments demonstrates that even simple clustering of the retailers based on their RFM values can reveal the business behaviours of segments of retailers using the EFTPOS facilities.

The preliminary work reported in this paper only uses 18 days of EFTPOS data. To date, we have been accumulating months of the EFTPOS data which makes this project a truly challenging Big Data problem with respect to the volume of data to be processed and analysed.

A natural progression of this work is on finding ways to optimize the Agglomerative Hierarchical Clustering (AHC) algorithm for Big Data through parallelisation. Splitting the data set into two subsets like what we have done in this paper is obviously not scalable.

The bottleneck in the AHC algorithm is in the calculations of the distance metric between data points to build the distance matrix and to update it upon the creation of a new cluster. Time saving can be achieved by parallelisation of this part of the algorithm, e.g. the calculations of the distance between a set of data points with each other and with all the other data points are done on one processor. Hence a few processors will run in parallel, each is dedicated to a subset of the data set. The extendability of ideas like the one explained in [19] to Big Data problem will be explored. A Hadoop processing environment has recently been set up for this project which will enable parallel processing on very large unidentifiable intermediate data.

To gain more insight into segments of the retailers, we will perform rule inductions, association rules analysis and build causal models by using additional attributes, from the original data set as well as other exogenous attributes, like socio-demographic, advertising, social media data. Using additional attributes with large nominal values will introduce another Big Data challenge, i.e. Variety, into this project. Finding suitable attribute subset selection method for Big Data will be another exciting avenue in this research.

# 9. ACKNOWLEDGMENTS

The authors wish to thank Jon Scheele and Jo Spencer for their help in data acquisition.

# 10. REFERENCES

[1] W. R. Smith, “Product differentiation and market segmentation as alternative marketing strategies,” *The Journal of Marketing*, vol. 21, no. 1, pp. 3–8, 1956.

[2] C. Doyle, *A dictionary of marketing*. Oxford University Press, 2011.

[3] M. J. Brusco, J. D. Cradit, and A. Tashchian, “Multicriterion clusterwise regression for joint segmentation settings: An application to customer value,” *Journal of Marketing Research*, pp. 225–234, 2003.

[4] D. Chen, S. L. Sain, and K. Guo, “Data mining for the online retail industry: A case study of RFM model-based customer segmentation using data mining,” *Journal of Database Marketing & Customer Strategy Management*, vol. 19, no. 3, pp. 197–208, 2012.

[5] G. T. Ho, W. Ip, C. Lee, and W. Mou, “Customer grouping for better resources allocation using GA based clustering technique,” *Expert Systems with Applications*, vol. 39, no. 2, pp. 1979–1987, 2012.

[6] M. Namvar, M. R. Gholamian, and S. KhakAbi, “A two phase clustering method for intelligent customer segmentation,” in *Intelligent Systems, Modelling and Simulation (ISMS), 2010 International Conference on*, 2010, pp. 215–219.

[7] C. Hung and C.-F. Tsai, “Market segmentation based on hierarchical self-organizing map for markets of multimedia on demand,” *Expert Systems with Applications*, vol. 34, no. 1, pp. 780–787, 2008.

[8] S. Alam, G. Dobbie, P. Riddle, and M. A. Naeem, “Particle swarm optimization based hierarchical agglomerative clustering,” in *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on*, vol. 2, 2010, pp. 64–68.

[9] A. M. Hughes, *Strategic database marketing*. McGraw-Hill, 2006.

[10] N.-C. Hsieh, “An integrated data mining and behavioral scoring model for analyzing bank customers,” *Expert Systems with Applications*, vol. 27, no. 4, pp. 623–633, 2004.

[11] H.-C. Chang and H.-P. Tsai, “Group RFM analysis as a novel framework to discover better customer consumption behavior,” *Expert Systems with Applications*, vol. 38, no. 12, pp. 14499–14513, 2011.

[12] D. L. Olson, Q. Cao, C. Gu, and D. Lee, “Comparison of customer response models,” *Service Business*, vol. 3, no. 2, pp. 117–130, 2009.

[13] G. Lefait and T. Kechadi, “Customer Segmentation Architecture Based on Clustering Techniques,” in *Digital Society, 2010. ICDS’10. Fourth International Conference on*, 2010, pp. 243–248.

[14] Y.-S. Chen, C.-H. Cheng, C.-J. Lai, C.-Y. Hsu, and H.-J. Syu, “Identifying patients in target customer segments using a two-stage clustering-classification approach: A hospital-based assessment,” *Computers in Biology and Medicine*, vol. 42, no. 2, pp. 213–221, 2012.

[15] B. Baesens, S. Viaene, D. Van den Poel, J. Vanthienen, and G. Dedene, “Bayesian neural network learning for repeat purchase modelling in direct marketing,” *European Journal of Operational Research*, vol. 138, no. 1, pp. 191–211, 2002.

[16] M. Bizhani and M. J. Tarokh, “Behavioral rules of bank’s point-of-sale for segments description and scoring prediction,” *Int. J. Industrial Eng. Comput*, vol. 2, pp. 337–350, 2011.

[17] A. Singh, G. Rumantir, and A. South, "Market Segmentation of EFTPOS Retailers", *in the Proceedings of the Australasian Data Mining Conference (AusDM 2014), Brisbane, 27-28 November 2014. Conferences in Research and Practice in Information Technology, Vol. 158.* (in press).

[18] J. Wu, J. Chen, H. Xiong, and M. Xie, “External validation measures for K-means clustering: A data distribution perspective,” *Expert Systems with Applications*, vol. 36, no. 3, pp. 6050–6061, 2009.

[19] C. F. Olson, “Parallel algorithms for hierarchical clustering,” *Parallel computing*, vol. 21, no. 8, pp. 1313–1325, 1995.