CSCI-B 565 DATA MINING
Homework 4
Evening Class
Computer Science Core
Fall 2013
Indiana University

Tousif Ahmed
touahmed@indiana.edu

1 December, 2013

All the work herein is solely mine.

1 Fraudulent Sales Data

According to the results of the 14th annual CyberSource Online Fraud Report, U.S. merchants lost an estimated $ 3.4 billion to fraud in 2011 [1]. Companies lose more than 5% of their revenue annually. So, the necessity for detecting the fraud data is increasing day by day. Using data mining algorithms to identify the fraud data is common approach and they are very efficient to detect the fraud data. In this assignment, we have worked with a special case of fraud data and applied some well known data mining algorithm. Normally, fraud detection is difficult to identify, because they are normally different from most other objects. They can be outliers or they can be anomalous behaviour. Outliers are data objects that have characteristics different from others or they can be unusual value for a specific attribute. For this case, most of the time it is difficult to identify the anomalous data from the outliers.

Problem Statement

In this task, we will use the transactions reported by the salespeople of some company. These salespeople sell a set of products and report their quantity and total monetary value of that product.The salespeople are free to set the selling price according to their own policy and market. At the end of each month, they report back to the company their transactions. The goal of this data mining application is to help in the task of verifying the veracity of these reports given past experience of the company that has detected both errors and fraud attempts in these transaction reports [2].There are total 401146 sales records, which are sufficiently large dataset. And,from these records we know the result of almost 15000 data, which are identified as either ok or fraud transactions. We will use these 15K records to identify the unknown results. We will use some well known data mining approach predict the outcome for unknown records.

Technically, We have 401146 sales record, where almost 15K samples are classified either as ok transaction or fraud transaction. There are 5 attributes in these dataset, which are ID, Prod,Quant,Val and Insp. We will implement some well known data mining approach like $k-$means, Naïve Bayes, ID3 and $k-nn$. We will use the known samples to learn about the common and uncommon behaviours and apply them to unknown dataset to classify the unknown results. Simple, using these 15K data we will predict the outcome of the unknown data set. We will take both clustering approach and classification approach to predict the outcome. In the classification approach, we first learn from the known samples. Like, in Naïve Bayes algorithm, we will

calculate the probabilities from the known samples and using those probabilities we will classify the unknown result. And in the clustering approach like, we will divide the the known samples to several clusters and using those clusters we will classify the unknown data.

This is sufficiently large dataset. But, if we consider the known samples there might not sufficient dataset. But still it is a large dataset and they are sufficient to classify the unknown data. But there is some problems with the number of attributes with the data size. There are only 5 attributes. And from these 5 attributes we can use only 4 attributes on $k-$means algorithm, which might be a problem for k-means. For this reason, I have added one extra attribute which is unit price. Unit price is obtained by dividing the value with quant.

Analysis

1. **Analysis of original data Δ**

   (a) I start analysing by loading the data into R. I have mentioned previously that there are 401,146 data. The following R code gives us the summary of the data.

```
> summary(sales)
```

```
       ID              Prod            Quant                Val          
 v431   : 10159   p1125  :  3923   Min.   :      100   Min.   :   1005  
 v54    :  6017   p3774  :  1824   1st Qu.:      107   1st Qu.:   1345  
 v426   :  3902   p1437  :  1720   Median :      168   Median :   2675  
 v1679  :  3016   p1917  :  1702   Mean   :     8442   Mean   :  14617  
 v1085  :  3001   p4089  :  1598   3rd Qu.:      738   3rd Qu.:   8680  
 v1183  :  2642   p2742  :  1519   Max.   :473883883   Max.   :4642955  
 (Other):372409   (Other):388860   NA    :13842       NA   :1182       
    Insp       
 ok   : 14462  
 unkn :385414  
 fraud:  1270  
```

   From the summary we can see that there are 5 attributes ID, Prod, Quant, Val and Insp. Where ID represents the salesman id, Prod represents the product id,Quant represents the the number of reported sold units of the product ,Val indicates the reported total monetary value of the sale and Insp a factor with three possible values: **ok** if the transaction was inspected and considered valid by the company, **fraud** if the transaction was found to be fraudulent, and **unkn** if the transaction was not inspected at all by the company.

   (b) I have used Hmisc package to get more information about the data set. The following R code give us more detail about the data set.

```
> load("/home/touahmed/Desktop/Dropbox/DMHW4/DMHW4_P1/sales.RData")
> nlevels(sales$ID)
```

```
[1] 6016
```

```
> nlevels(sales$Prod)
```

```
[1] 4548
```

```
> library("Hmisc")
> describe(sales)
```

```
sales

 5  Variables      401146  Observations
--------------------------------------------------------------------------------
ID
      n missing  unique
 401146       0    6016
```

```
lowest : v1    v2    v3    v4    v5   , highest: v6066 v6067 v6068 v6069 v6070
---------------------------------------------------------------------------
Prod
      n missing  unique
 401146       0    4548

lowest : p1    p2    p3    p4    p5   , highest: p4544 p4545 p4546 p4547 p4548
---------------------------------------------------------------------------
Quant
      n missing  unique    Mean     .05     .10     .25     .50     .75     .90
 387304   13842   20956    8442     100     101     107     168     738    4877
    .95
  12916

lowest :       100       101       102       103       104
highest:  56590926 164244544 173844544 194044544 473883883
---------------------------------------------------------------------------
Val
      n missing  unique    Mean     .05     .10     .25     .50     .75     .90
 399964    1182   21821   14617    1040    1085    1345    2675    8680   27250
    .95
  52995

lowest :    1005    1010    1015    1020    1025
highest: 4161740 4308620 4475360 4616735 4642955
---------------------------------------------------------------------------
Insp
      n missing  unique
 401146       0       3

ok (14462, 4%), unkn (385414, 96%), fraud (1270, 0%)
---------------------------------------------------------------------------
```

We can see that there are 6016 distinct sales ID, and 4548 product IDs. We can see from the summary and the description of the sales data that there are some missing values in the Quant and Val attribute. I will explain about missing values later.For now, we can see that there are total 13842 values missing in quant and there are 1182 values are missing in Val. And We can see the details of Insp. There are 14462 records which are identified as ok transaction and 1270 records are identified as fraudulent sales data. The rest of 385414 records are unknown. We have to classify them. The percentage of results are 4%(exactly 3.6%) of the data are identified as ok, 0.3% of the data are identified as fraud and 96% of the data are unknown. We can see that, the percentage of identified records are relatively low and moreover the number of fraud transaction is pretty small. This will explain the difficulty of classify the fraud data.

(c) To get an idea about the sales person number of transaction I analyse the salesperson ID and product ID . The following R code helps us to analyse the salesperson ID and product ID [2] .

```
> totS <- table(sales$ID)
> totP <- table(sales$Prod)
> barplot(totS, main = "Transactions per salespeople", names.arg = "",
+         xlab = "Salespeople", ylab = "Amount")
> barplot(totP, main = "Transactions per product", names.arg = "",
+         xlab = "Products", ylab = "Amount")
```

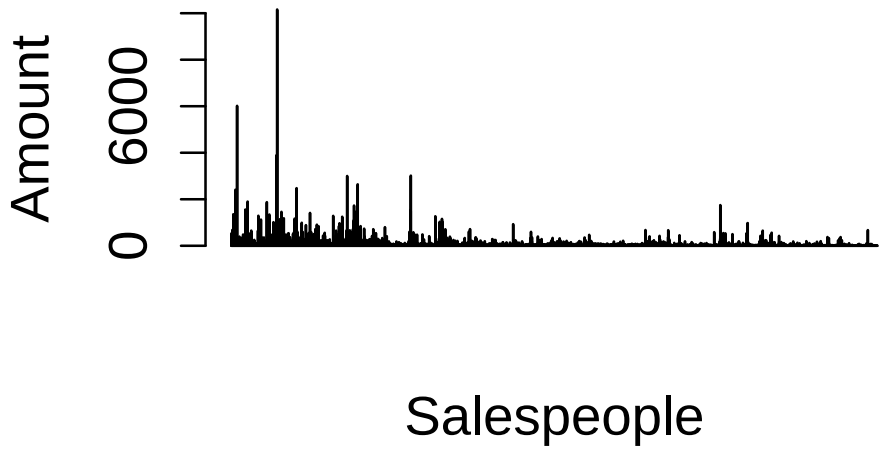

Figure 1: The number of transactions per salesperson against Value

The transactions per salesman and product against value is shown in Figure 1 and 2. We can see from the figure that they are very diverse. There is no common pattern. The transaction against sales person and product is variable.So, we can conclude that the sales id and the product id might have some effect over the result. If a sales person is identified as doing fraud transactions several times, then this can be predicted that his rest of the transactions are also questionable. Similarly, if a product is identified in fraud data several times, that could mean we need to give a closer look over those data which contains this product.

(d) I already mentioned that there are several missing values. Among 401,146 data almost 15K data are missing. There are 13842 missing values in Quant attribute and 1182 missing values in Val attribute. The number missing values on both of the attributes can be calculate by following R code:

```
> length(which(is.na(sales$Quant) & is.na(sales$Val)))
[1] 888
```

So,
Total number of rows = 401,146
Total number of missing data= 13842 + 1182 -888 = 14136
Total data after removing missing data= 401,146-14,136= 387, 010
Percentage of missing data= $\frac{14136}{401146} * 100$=3.5%

Almost 3.5% data are missing. The missing data could be significant if there are large number of known records. If the missing records contains more percentage of missing records, specially the fraud sales records, then we can not discard the missing records. Because we already have small percentage of fraud records. And we have only 4% of known records. The following R code give us more details about the missing records.

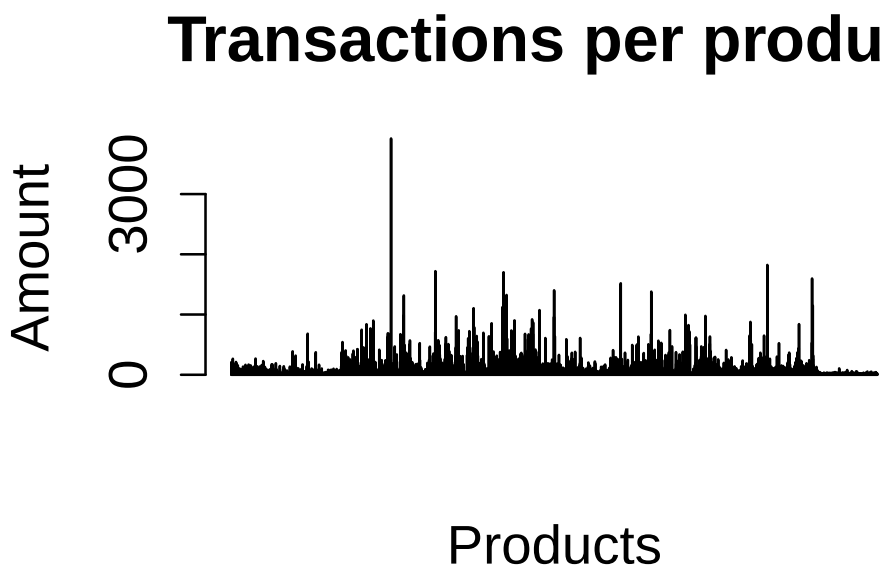

Figure 2: The number of transactions per product against Value

```
> na_sales <-sales[!complete.cases(sales),]
> table(na_sales$Insp)

   ok  unkn fraud
  115 13950    71

> table(na_sales$Insp)/nrow(na_sales) * 100

        ok       unkn      fraud
 0.8135257 98.6842105  0.5022637

> table(na_sales$Insp)/nrow(sales$) * 100

        ok       unkn      fraud
0.02866787 3.47753686 0.01769929
```

We can see that, among the missing rows 115(0.81%) of them are ok records, 71(0.5%) of them are fraud records and the rest of 13950(98%) are unknown records. So, among the missing records 1.3% are from the known records. And from the over all data only 0.03% of the known records are missing, which might not be significant.

The following results supports that the percentage of missing data is not significant.First I run the Naïve Bayes algorithm and $k-$ means algorithm on the original dataset.Then, I replaced the missing data using mean and median. Then I ran the Naïve Bayes algorithm and $k-$ means algorithm on these replaced data set. The summary of the result is given below:

Naïve Bayes Algorithm:

Weighted PPV with missing data records removed= 0.8757
Weighted PPV with missing data records replaced by mean =0.8699
Weighted PPV with missing data records replaced by median= 0.8698

k-means Algorithm:

Weighted PPV with missing data records removed= 0.9085
Weighted PPV with missing data records replaced by mean =0.9044
Weighted PPV with missing data records replaced by median= 0.9044

The PPV is calculated using 10-fold cross validation. So, from the result we can see that the algorithms are performing better when the missing data records are removed. Though there is slight difference on weighted PPV when the missing records are replaced, though we can conclude that they do not have any significant effect on the overall data set. So, the missing values are insignificant.

(e) Analysing more over the data I have found out that there are significant amount of duplicate data. The following code gives us the details of duplicate data :

```
> v <- duplicated(removed_missing)
>table(v)

   FALSE   TRUE
  308354   78656
```

We can see that there are 78656 duplicate data. We cannot keep this data. Because, if we think about Naïve Bayes algorithm, where we need to calculate the prior probabilities.If the duplicated data do not have equal percentage of ok data and fraud data, then these duplicated data can change result. It will be clear if we see the following R code:

```
> duplicate <-removed_missing[v,c("ID","Prod","Quant","Val","Insp")]
> table(duplicate$Insp$)

   ok  unkn fraud
 2467 76188     1
```

We can see that among the duplicate records, 2467 of them are from ok records and only 1 of them are from fraud records. If we keep the duplicate data, it will have significant affect over the prior probabilities. The result will be more biased over ok data.

(f) We cannot discard the outliers here. Because, the outliers could be the fraudulent sales data. The following R code supports our claim:

```
> table(cleaned_data$Quant > 10000000)

 FALSE   TRUE
308342     12
```

```
> possible_outliers <- cleaned_data[cleaned_data$Quant > 10000000,c("ID","Prod","Quant","Val","
> table(possible_outliers$Insp)

   ok  unkn fraud
    3     1     8
```

```
> possible_outliers <- cleaned_data[cleaned_data$Val > 4000000,c("ID","Prod","Quant","Val","Ins
> table(possible_outliers$Insp$)

   ok  unkn fraud
    0     2     3
```

We can see that only 12 values of Quant attribute are greater than 10000000, which can be possible outliers. But if we look at the output of those codes we can see that 8 of them are fraud. Similarly for the Val attribute only 5 of them are greater than 4000000. But among these 5, three of them are fraud data. So, the result supports our claim they might not detect the outliers over here. So, we can not delete any data points by considering as a possible outlier.

(g) To get an insight of the data space, I write the following R code, which gives us the scatter plot for $\log(Quant)$ against $\log(val)$:

```
> plot(log(Quant)~log(Val),data=cleaned_data, col=rgb(0,0,0,0.5),pch=16)
```

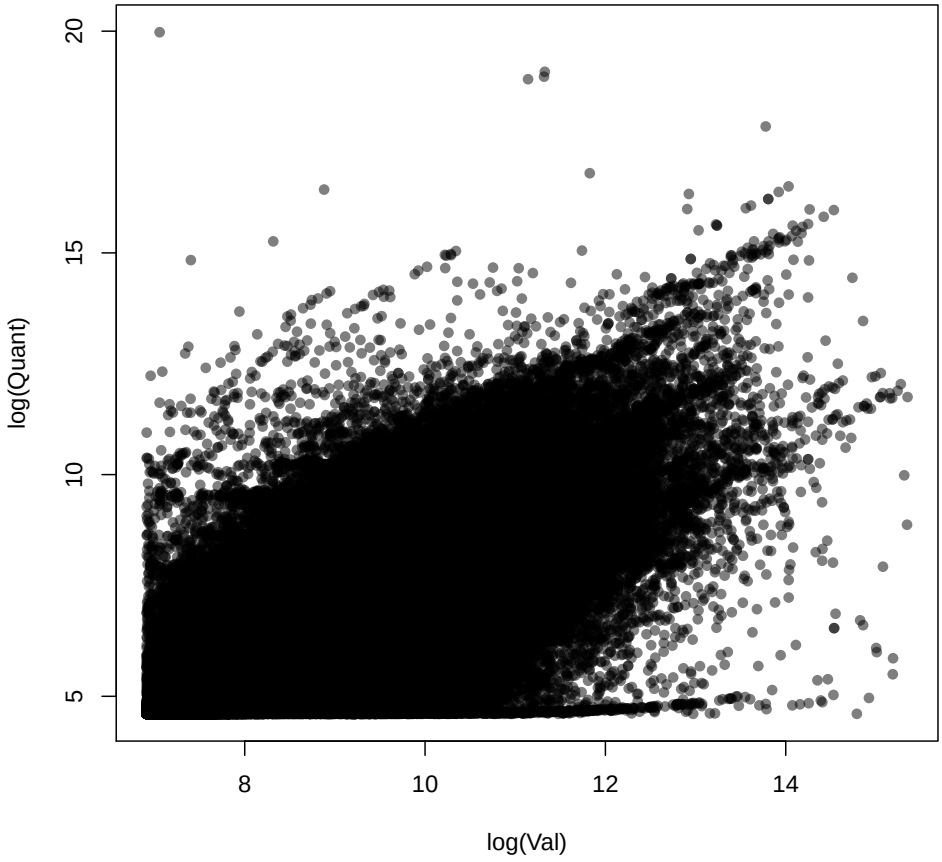

Figure 3: Quant against Value over complete data set

Figure 3 gives us the complete space for the overall dataset. We can see that they are clustered over in mid range, and very few of the values are scattered outside the clusters. The following R code gives us a plot for the known data set:

```
> data_ok<- cleaned_data[cleaned_data$Insp!="unkn",c("ID","Prod","Quant","Val","Insp")]
> data_ok$Insp<-as.numeric(data_ok$Insp)
> table(data_ok$Insp)

    1     3
11880  1198

> plot(log(Quant)~log(Val),data=data_ok, col=data_ok$Insp+7$,pch=16)
```

Figure 4 shows the Quant and Val space over known data set. The ok data sets are shown by gray color and fraud data values are shown by red color. We can see the values for fraud data and see that some of them are inside main cluster. This might be a problem when we run this data set on clustering algorithms like $k-$means.

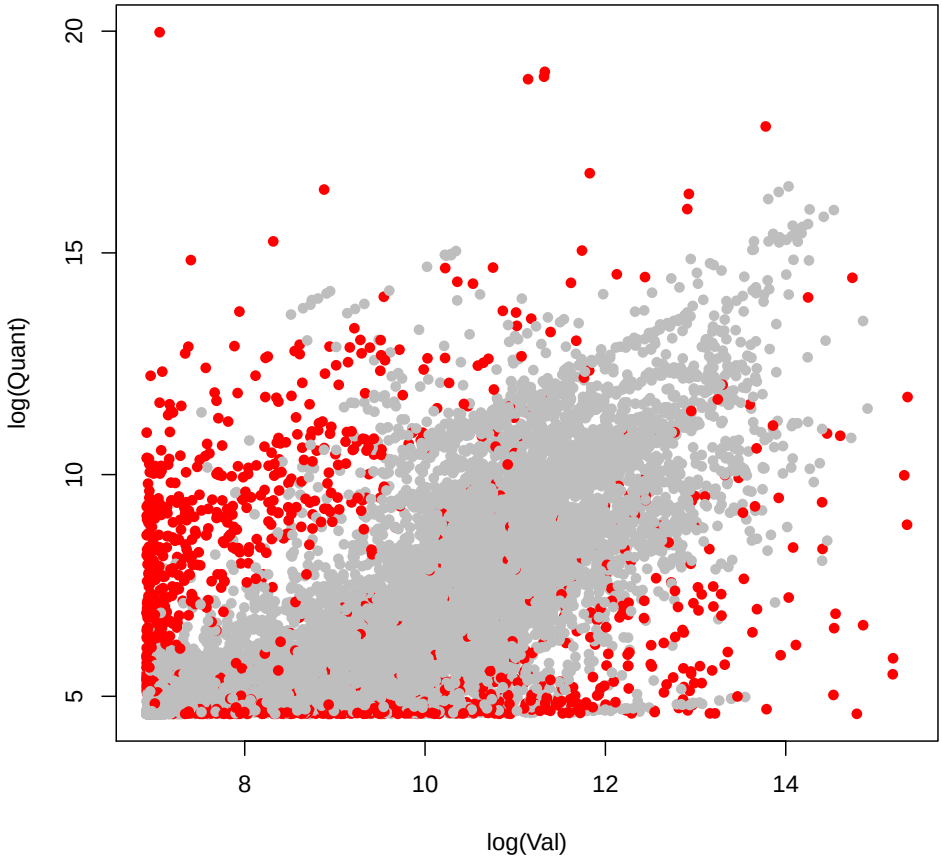

Figure 4: Quant against Value over known data set

(h) Finally, for the purpose of the algorithm I introduced a new attribute unit price which is calculated by dividing the Values with Quant.

2. **Steps to clean and transform data to Δ^***

(a) I have removed the missing data values using the following R code:

```
> removed_missing <-sales[complete.cases(sales),]
> write.table(removed_missing,"sales_removing_missing_data.csv",sep=",",row.names=FALSE)
> dim(removed_missing)
```

```
[1] 387010      5
```

(b) I removed the duplicate data by using following R code. The value is saved into a csv file.

```
> v <- duplicated(removed_missing)
> table(removed_missing\$Insp)
```

```
    ok   unkn  fraud
 14347 371464   1199
```

```
> duplicate <-removed_missing[v,c("ID","Prod","Quant","Val","Insp")]
> table(duplicate\$Insp)
```

```
   ok  unkn fraud
 2467 76188     1
> cleaned_data<-subset(removed_missing,!duplicated(removed_missing))
> dim(cleaned_data)
[1] 308354      5
> table(cleaned_data$Insp$)
    ok   unkn  fraud
 11880 295276   1198
> write.table(cleaned_data,"cleaned_sales_data.csv",sep=",",row.names=FALSE)
```

(c) The following code is employed to replace the missing values using mean.

```
> sales$Quant[is.na(sales$Quant)] <- as.integer(mean(sales$Quant,na.rm=TRUE))
> sales$Val[is.na(sales$Val)] <- as.integer(mean(sales$Val,na.rm=TRUE))
> #duplicate
> v <- duplicated(sales)
> duplicate <-sales[v,c("ID","Prod","Quant","Val","Insp")]
> cleaned_data<-subset(sales,!duplicated(sales))
> cleaned_data$UnitPrice <- cleaned_data$Val/cleaned_data$Quant$
> write.table(cleaned_data,"replaced_data_mean.csv",sep=",",row.names=FALSE)
```

(d) The following code is employed to replace the missing values using median.

```
> sales$Quant[is.na(sales$Quant)] <- as.integer(median(sales$Quant,na.rm=TRUE))
> sales$Val[is.na(sales$Val)] <- as.integer(median(sales$Val,na.rm=TRUE))
> #duplicate
> v <- duplicated(sales)
> duplicate <-sales[v,c("ID","Prod","Quant","Val","Insp")]
> cleaned_data<-subset(sales,!duplicated(sales))
> cleaned_data$UnitPrice <- cleaned_data$Val/cleaned_data$Quant$
> write.table(cleaned_data,"replaced_data_median.csv",sep=",",row.names=FALSE)
```

3. **Data Mining algorithms employed to solve $\Pi(\Delta)$**

I have implemented four algorithms: Naïve Bayes algorithm, $k-$means algorithm,ID3 algorithm and $k-$nearest neighbour algorithm. the complete data set is divided into to 2 sets based on their output. The known data set is included as a train data set and the unknown data records are included in test data set. I used train data to train the algorithms, then I ran the whole test data set to classify the test records. To ensure the quality of the results, I applied 10-fold cross validation. I calculate the PPV on each subset, later calculate the weighted PPV to analyse the result. The analysis of the results is presented in the following sections.

(a) Naïve Bayes algorithm:

- As Naïve Bayes algorithm considers probability, I used all of the 5 attributes to calculate the conditional probability. I get better result when using 5 attributes instead of four.
- I used $m-$estimate of conditional probability. If the class conditional probability for one of the attributes is zero, then the overall posterior probability vanishes. I used the following function to calculate the $m-$estimate.

$$P(x_i|y_j) = \frac{n_c + mp}{n + m}$$

Where,
n=total number of instances from class y_j
n_c=no. of training examples from class y_j that take on the value x_i
m= equivalent sample size p= user specified parameter

- I have tested the result for different p values with sample size m=10

(b) $k-$means algorithm:

- For the sake of $k-$means algorithm, I used 3 attributes(QUant,Val and unit price) to measure the records distance from centroids. To calculate the distance, I used Euclidean distance.
- From Figure 4 we can see that, most of the values are clustered in the middle. But as 90% data is from ok class, so it gives the PPV almost 90%. But most of the cases it failed to classify the fraud dataset correctly. So, I used larger cluster size and tested the result with different k- values.
- As abnormal values are important for this problem, so I normalized the complete data set before fed them into $k-$means algorithm. Because outliers have influence on the clusters.
- When we are using larger k values, it is possible that some clusters might not have any data point. To solve this problem and get more efficient result I discarded the empty clusters and generate new clusters after each iteration.

(c) ID3 algorithm:

- I used 3 attributes(Quant,Val and unit price) to generate the decision tree. For the continuous attributes, I normalize them at first. The values are normalized in[1,10]. Then, I searched for best partition. The best partition is chosen considering which partition gives the best gain.

(d) K-nn algorithm:

- For the sake of $k-$means algorithm, I used 3 attributes(QUant,Val and unit price) to measure the records distance from centroids. To calculate the distance, I used Euclidean distance.
- The results are different for different values of k. I have tested the result for different k values.

Quality and Analysis of Results:

I used 10 fold cross validation to measure the quality of the results. Then I ran the test set using the complete trained set. The analysis for each algorithm is given below:

(a) **Naïve Bayes algorithm :**

- The following table Table 1 gives the PPV for Naïve Bayes algorithm. It is generated without m-estimate approach.

Train	Test	Total Data	True Positive	False Positive	PPV
$(D^* - d_1^*)$	d_1^*	1308	1237	71	0.9457
$(D^* - d_2^*)$	d_2^*	1308	1222	86	0.9343
$(D^* - d_3^*)$	d_3^*	1308	1104	204	0.8440
$(D^* - d_4^*)$	d_4^*	1308	1112	196	0.8502
$(D^* - d_5^*)$	d_5^*	1308	1045	263	0.7989
$(D^* - d_6^*)$	d_6^*	1308	1056	252	0.8073
$(D^* - d_7^*)$	d_7^*	1308	1046	262	0.7997
$(D^* - d_8^*)$	d_8^*	1308	985	323	0.7531
$(D^* - d_9^*)$	d_9^*	1308	1069	239	0.8173
$(D^* - d_{10}^*)$	d_{10}^*	1306	1075	231	0.8231

Table 1: Output of Naïve Bayes algorithm without m-estimated approach

In this case,
Weighted PPV= 0.8374
On Test Set: Total Number of Data classified as ok= 291872
Total Number of Data classified as Fraud : 3404

The weighted PPV is 0.8374 which is acceptable.

- The following table Table 2 is generated using m estimate approach. The p value used for ok set is 0.9 and 0.1 for fraud set. If we see the actual distribution we can observe similar probability. In this case, we get slightly better accuracy. But over test set total number of data classified as fraud is reduced. So, the probability p has somewhat affect on final results, which will be clear on the next results.

Train	Test	Total Data	True Positive	False Positive	PPV
$(D^* - d_1^*)$	d_1^*	1308	1262	46	0.9648
$(D^* - d_2^*)$	d_2^*	1308	1246	62	0.9526
$(D^* - d_3^*)$	d_3^*	1308	1164	144	0.8899
$(D^* - d_4^*)$	d_4^*	1308	1162	146	0.8884
$(D^* - d_5^*)$	d_5^*	1308	1107	201	0.8463
$(D^* - d_6^*)$	d_6^*	1308	1129	179	0.8631
$(D^* - d_7^*)$	d_7^*	1308	1109	199	0.8479
$(D^* - d_8^*)$	d_8^*	1308	1040	268	0.7951
$(D^* - d_9^*)$	d_9^*	1308	1113	195	0.8509
$(D^* - d_{10}^*)$	d_{10}^*	1306	1121	185	0.8583

Table 2: Output of Naïve Bayes algorithm with m-estimated approach, p=0.9 for ok and p=0.1 for fraud

In this case, Weighted PPV= 0.8757

On Test Set: Total Number of Data classified as ok= 294183
Total Number of Data classified as Fraud : 1093

- The following table table 3 is generated using m estimate approach. The p value used for ok set is 0.7 and 0.3 for fraud set. In this case, we get slightly reduced accuracy. But over test set total number of data classified as fraud is increased and total 1893 of the data is identified as fraud.

Train	Test	Total Data	True Positive	False Positive	PPV
$(D^* - d_1^*)$	d_1^*	1308	1254	54	0.9587
$(D^* - d_2^*)$	d_2^*	1308	1239	69	0.9472
$(D^* - d_3^*)$	d_3^*	1308	1146	162	0.8761
$(D^* - d_4^*)$	d_4^*	1308	1149	159	0.8784
$(D^* - d_5^*)$	d_5^*	1308	1087	221	0.8310
$(D^* - d_6^*)$	d_6^*	1308	1112	196	0.8502
$(D^* - d_7^*)$	d_7^*	1308	1094	214	0.8364
$(D^* - d_8^*)$	d_8^*	1308	1020	288	0.7798
$(D^* - d_9^*)$	d_9^*	1308	1098	210	0.8394
$(D^* - d_{10}^*)$	d_{10}^*	1306	1103	203	0.8446

Table 3: Output of Naïve Bayes algorithm with m-estimated approach, p=0.7 for ok and p=0.3 for fraud

In this case, Weighted PPV= 0.8642
On Test Set: Total Number of Data classified as ok= 293383
Total Number of Data classified as Fraud : 1893

- The following table Table 4 is generated using m estimate approach. The p value used for ok set is 0.5 and 0.5 for fraud set. In this case, we get almost similar result of the accuracy which we calculated without $m-$estimate approach.
In this case, Weighted PPV= 0.8393
On Test Set: Total Number of Data classified as ok= 291916

Train	Test	Total Data	True Positive	False Positive	PPV
$(D^* - d_1^*)$	d_1^*	1308	1237	71	0.9457
$(D^* - d_2^*)$	d_2^*	1308	1222	86	0.9343
$(D^* - d_3^*)$	d_3^*	1308	1108	200	0.8471
$(D^* - d_4^*)$	d_4^*	1308	1114	194	0.8517
$(D^* - d_5^*)$	d_5^*	1308	1048	260	0.8012
$(D^* - d_6^*)$	d_6^*	1308	1061	247	0.8112
$(D^* - d_7^*)$	d_7^*	1308	1052	256	0.8043
$(D^* - d_8^*)$	d_8^*	1308	988	320	0.7554
$(D^* - d_9^*)$	d_9^*	1308	1071	237	0.8188
$(D^* - d_{10}^*)$	d_{10}^*	1306	1076	230	0.8239

Table 4: Output of Naïve Bayes algorithm with m-estimated approach, p=0.5 for ok and p=0.5 for fraud

Total Number of Data classified as Fraud : 3360

- So, If analyse the result, we can see that using $m-$estimate approach we get better results. But,running the algorithm several times I have found out that the prior probability have significant effect on the result. And I have faced lots of problems, like mistakenly I set the prior probability 0. And I got PPV of 0.9083. The reason behind this high accuracy is almost 90% of the data are from ok set. So, though the outputs are incorrect it can atleast classify the ok data correctly. So, that is why we get such high accuracy.

(b) **k-means Algorithm**

In the k- means algorithm the result was significantly dependent on the value of k. Though, almost all of the cases we have higher PPV, still we cannot rely on them. First of all, it has high discrepancy over distribution on the ok data set and fraud dataset. So, it is difficult to identify the correct fraud clusters.As, the total number of ok data is larger than the total number of fraud data. Ok data dominates over the cluster and for this reason fraud data is also misclassified as ok data. Table 5 shows sample 10 fold cross validation for k=10. Table 6 shows different weighted PPV values. The weighted PPV varies depending on k. But, if look at the number of test data classified as fraud is gradually increasing while k increases. It cannot classify a single data when k=2 and 10.

Train	Test	Total Data	True Positive	False Positive	PPV
$(D^* - d_1^*)$	d_1^*	1308	1278	30	0.9771
$(D^* - d_2^*)$	d_2^*	1308	1270	38	0.9709
$(D^* - d_3^*)$	d_3^*	1308	1212	96	0.9266
$(D^* - d_4^*)$	d_4^*	1308	1195	113	0.9136
$(D^* - d_5^*)$	d_5^*	1308	1169	139	0.8937
$(D^* - d_6^*)$	d_6^*	1308	1187	121	0.9075
$(D^* - d_7^*)$	d_7^*	1308	1150	158	0.8792
$(D^* - d_8^*)$	d_8^*	1308	1121	187	0.8570
$(D^* - d_9^*)$	d_9^*	1308	1154	154	0.8823
$(D^* - d_{10}^*)$	d_{10}^*	1306	1145	161	0.8767

Table 5: Output of k-means algortihm, for k=10

k	Weighted PPV for 10-fold	Total Ok	Total Fraud
2	0.9087	295276	0
10	0.9085	295276	0
50	0.9104	295150	126
100	0.9101	295061	215
200	0.9098	295152	124
500	0.9098	295138	138
1000	0.9080	295115	161
2000	0.9081	294980	296
5000	0.9069	294281	995

Table 6: Output of k-means algorithm for different k values

(c) **ID3 Algorithm:**
Table 7 shows the result for ID3 algorithm over train data set. Though, it has weighted average of 0.9083, still it did not perform well over the test data set. It could not classify any data as fraud data.

Train	Test	Total Data	True Positive	False Positive	PPV
$(D^* - d_1^*)$	d_1^*	1308	1278	30	0.9771
$(D^* - d_2^*)$	d_2^*	1308	1270	38	0.9709
$(D^* - d_3^*)$	d_3^*	1308	1213	95	0.9274
$(D^* - d_4^*)$	d_4^*	1308	1195	113	0.9136
$(D^* - d_5^*)$	d_5^*	1308	1169	139	0.8937
$(D^* - d_6^*)$	d_6^*	1308	1186	122	0.9067
$(D^* - d_7^*)$	d_7^*	1308	1151	157	0.8800
$(D^* - d_8^*)$	d_8^*	1308	1117	191	0.8540
$(D^* - d_9^*)$	d_9^*	1308	1155	153	0.8830
$(D^* - d_{10}^*)$	d_{10}^*	1306	1146	160	0.8775

Table 7: Output of ID3 algorithm

(d) **K nearest neighbor algorithm:**
Table 8 shows the result for k-nn algorithm over train data set. Though, it has weighted PPV of 0.8871, still it could not perform well over the test data set. It could not classify any data as fraud data.

Train	Test	Total Data	True Positive	False Positive	PPV
$(D^* - d_1^*)$	d_1^*	1308	1240	68	0.9480
$(D^* - d_2^*)$	d_2^*	1308	1240	68	0.9480
$(D^* - d_3^*)$	d_3^*	1308	1185	123	0.9060
$(D^* - d_4^*)$	d_4^*	1308	1167	141	0.8922
$(D^* - d_5^*)$	d_5^*	1308	1143	165	0.8739
$(D^* - d_6^*)$	d_6^*	1308	1160	148	0.8869
$(D^* - d_7^*)$	d_7^*	1308	1123	185	0.8586
$(D^* - d_8^*)$	d_8^*	1308	1092	216	0.8349
$(D^* - d_9^*)$	d_9^*	1308	1133	175	0.8662
$(D^* - d_{10}^*)$	d_{10}^*	1306	1118	188	0.8560

Table 8: Output of K nearest neighbour algorithm

2 Strange Data Set

By giving the first look of the data we can observe that the first column represents the year and the month of the corresponding year is concatenated. Year and month is concatenated by dot.The data gives results starting from 1945 to 1994. It might be survey report from 1945 to 1994. The next row can be considered as a single column or can be considered as 2 separate columns as if we consider the first column. Because they can be concatenated with dot as first column. There are total 600 observations.

If we consider the second column as one attribute, then the second column has 509 unique values and has a mean of 79.02. It has values from 0.1 to 229.2. If we consider the second column as two different attributes there are values range 0 to 229 for the first attribute and 0 to 9 for the second attribute. So,two different attributes for second column does not make sense. If it has any specific meaning then it might not be distributed over 0 to 9. So, we can consider the second column as a single entity. The second column has median of 68.85. If we see Figure 5, we can see the values and frequency for this column. We can see that the frequency is larger when the value is less than 50.

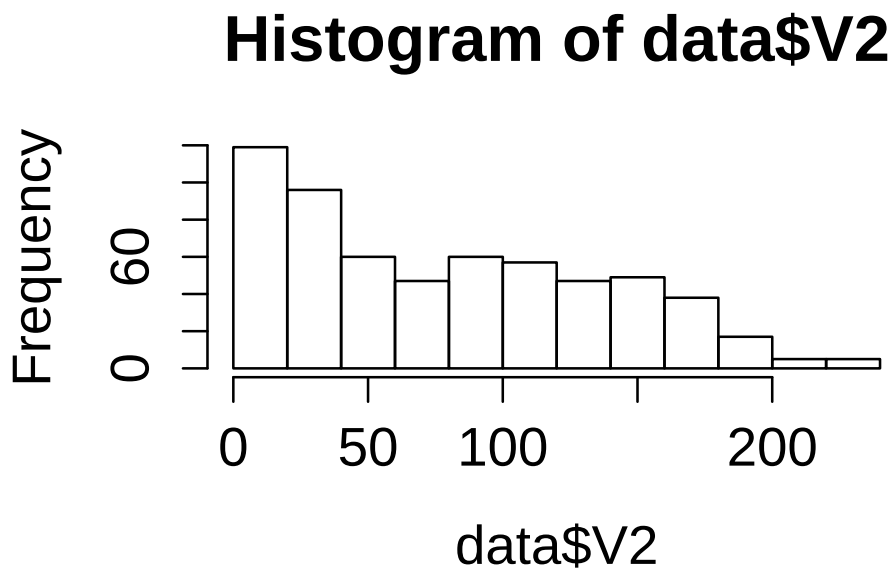

Figure 5: Histogram of 2nd column

If we look at the Figure 6 we can see a curve which is similar to sin curve. So, we can conclude that the value fluctuates over time.

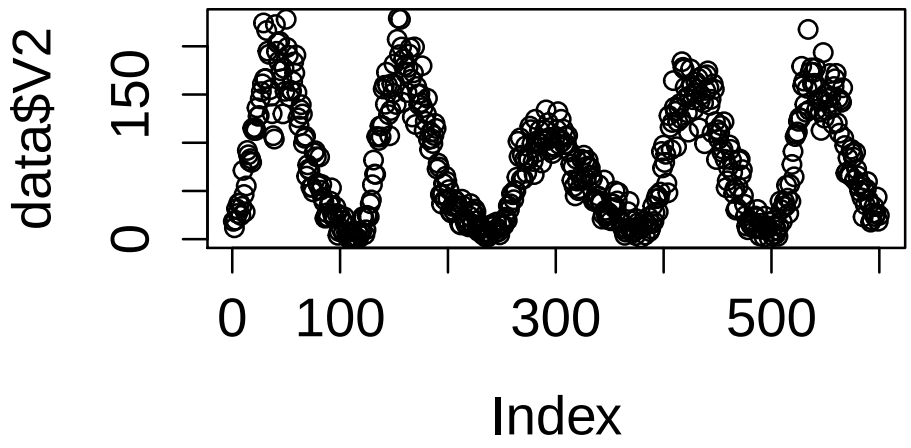

Figure 6: Plot for 2nd column

From the figure we can find out interesting characteristics about the data. Any survey result which shows similar kinds of plots, can be the intended data.

3 US Congress Data

The data gives us a statistics about the voting data from the 35th Session to the 112th US Congressional Sessions. This is the voting result from 1857 to January,2013.

1. Over this 41386 voting records the Republican and Democratic members have same opinion on 17929 times and they have different opinion on the rest of the 23457 times. Among this 23457 times different opinion 11457 times Republican members have the majority and the rest of the 12210 times Democratic members have the majority. They have same opinion by voting yes was 14495 times and the rest 3434 times they have same opinion on voting no. Republican members have voted equal number of yes or no 128 times and Democratic members have voted equal number of yes or no 103 times.

   Same Opinion: 17929,
   Different opinion: 23457,
   Majority Republican:11247,
   Majority Democratic: 12210

2. Over this 41386 voting records the Northern Democratic and Southern Republican members have same opinion on 17957 times and they have different opinion on the rest of the 23429 times. Among this 23429 times different opinion 22852 times Northern Democratic members have the majority and the rest of the 577 times Southern Republican members have the majority. They have same opinion by voting yes was 13463 times and the rest 4494 times they have same opinion on voting no. Northern Democratic members have voted equal number of yes or no 169 times and Southern Republic members have voted equal number of yes or no 3877 times.

   Same Opinion: 17957,
   Different opinion: 23429,
   Majority Democratic:22852
   Majority Republican: 577

3. Over this 41386 voting records the Northern Republican and Southern Democratic members have same opinion on 20208 times and they have different opinion on the rest of the 21178 times. Among this 21178 times different opinion 19635 times Northern Republican members have the majority and the rest of the 1543 times Southern Democratic members have the majority. They have same opinion by voting yes was 15207 times and the rest 4494 times they have same opinion on voting no. Northern Republican members have voted equal number of yes or no 144 times and Southern Democratic members have voted equal number of yes or no 1557 times.

   Same Opinion: 20208,
   Different opinion: 21178,
   Majority Democratic:19635
   Majority Republican: 1543

References

[1] Cybersource. 2013 Online Fraud Report, 14th Edition. `http://forms.cybersource.com/forms/fraudreport2013?cid=1-51664141&lsr=web/`, 2013.

[2] L.Torgo. *Data Mining with R Learning with Case Studies*. Chapman & Hall, 2010.