Project III

Ruben Solis

Due: 1 April 2020

Contents

1 Derivation and Proofs **1**

2 Computer Project **2**
2.1 Reading in the data 2
2.2 Data cleaning 3
2.3 Obtaining a distance matrix 5
2.4 Clustering algorithm 6
2.5 *Post hoc* analysis 11

3 Discussion **12**

4 References **13**

1 Derivation and Proofs

Let $\mathcal{C}$ denote a clustering scheme that clusters data into K clusters $\{c_1, \ldots, c_K\}$. The *within-cluster point scatter* $W(\mathcal{C})$ is defined as

$$W(\mathcal{C}) = \frac{1}{2} \sum_{k=1}^{K} \sum_{i \in c_k} \sum_{i' \in c_k} \|\mathbf{x}_i - \mathbf{x}_{i'}\|^2.$$

Show that

$$W(\mathcal{C}) = \sum_{k=1}^{K} n_k \sum_{i \in c_k} \|\mathbf{x}_i - \bar{\mathbf{x}}_k\|^2,$$

where $\bar{\mathbf{x}}_k$ denotes the mean vector and n_k is the sample size for cluster c_k.

Proof.

$$\begin{aligned} W(\mathcal{C}) &= \frac{1}{2}\sum_{k=1}^{K}\sum_{i\in c_k}\sum_{i'\in c_k}\|\mathbf{x}_i-\mathbf{x}_{i'}\|^2, \\ &= \frac{1}{2}\sum_{k=1}^{K}\sum_{i\in c_k}\sum_{i'\in c_k}\|\mathbf{x}_i-\bar{\mathbf{x}}_k+\bar{\mathbf{x}}_k-\mathbf{x}_{i'}\|^2, \\ &= \frac{1}{2}\sum_{k=1}^{K}\sum_{i\in c_k}\sum_{i'\in c_k}\{\|\mathbf{x}_i-\bar{\mathbf{x}}_k\|^2+\|\bar{\mathbf{x}}_k-\bar{\mathbf{x}}_{i'}\|^2+2(\mathbf{x}_i-\bar{\mathbf{x}}_k)^T(\mathbf{x}_{i'}-\bar{\mathbf{x}}_k)\}. \end{aligned}$$

Moreover,

$$2\sum_{i'\in c_k}(\mathbf{x}_i-\bar{\mathbf{x}}_k)^T(\mathbf{x}_{i'}-\bar{\mathbf{x}}_k)=2(\mathbf{x}_i-\bar{\mathbf{x}}_k)^T\sum_{i'\in c_k}(\mathbf{x}_{i'}-\bar{\mathbf{x}}_k)\propto\sum_{i'\in c_k}(\mathbf{x}_{i'}-\bar{\mathbf{x}}_k)=\sum_{i'\in c_k}\mathbf{x}_{i'}-\sum_{i'\in c_k}\bar{\mathbf{x}}_k=0.$$

Thus we have,

$$\begin{aligned} W(\mathcal{C}) &= \frac{1}{2}\sum_{k=1}^{K}\sum_{i\in c_k}\sum_{i'\in c_k}\|\mathbf{x}_i-\bar{\mathbf{x}}_k\|^2+\frac{1}{2}\sum_{k=1}^{K}\sum_{i\in c_k}\sum_{i'\in c_k}\|\mathbf{x}_k-\bar{\mathbf{x}}_{i'}\|^2+0, \\ &= \frac{1}{2}\sum_{k=1}^{K}\sum_{i\in c_k}n_k\|\mathbf{x}_i-\bar{\mathbf{x}}_k\|^2+\frac{1}{2}\sum_{k=1}^{K}\sum_{i\in c_k}\sum_{i'\in c_k}\|\mathbf{x}_i-\bar{\mathbf{x}}_k\|^2, \\ &= \frac{1}{2}\sum_{k=1}^{K}\sum_{i\in c_k}n_k\|\mathbf{x}_i-\bar{\mathbf{x}}_k\|^2+\frac{1}{2}\sum_{k=1}^{K}\sum_{i\in c_k}n_k\|\mathbf{x}_i-\bar{\mathbf{x}}_k,\|^2, \\ &= \sum_{k=1}^{K}n_k\sum_{i\in c_k}\|\mathbf{x}_i-\bar{\mathbf{x}}_k\|^2. \end{aligned}$$

Consequently, $W(\mathcal{C})=\frac{1}{2}\sum_{k=1}^{K}\sum_{i\in c_k}\sum_{i'\in c_k}\|\mathbf{x}_i-\mathbf{x}_{i'}\|^2=\sum_{k=1}^{K}n_k\sum_{i\in c_k}\|\mathbf{x}_i-\bar{\mathbf{x}}_k\|^2$, as was to be shown. □

2 Computer Project

2.1 Reading in the data

We begin by first bringing in the data into R, and then by listing the missing data rate (in terms of percentage) for each of the variables in our data set with the following code.

```
dat <- read.csv("HMEQ.csv", header = TRUE, sep = ',')
vnames <- colnames(dat)
n <- nrow(dat)
out <- NULL
for (j in 1:ncol(dat)){
```

```
  vname <- colnames(dat)[j]
  x <- as.vector(dat[,j])
  nmiss <- sum(is.na(x))
  ncomplete <- n-nmiss
  out <- rbind(out, c(column.number=j, variable.names=vname,
              missing.percentage=(nmiss/n)*100))
}
out <- as.data.frame(out)
row.names(out) <- NULL
out
```

```
##    column.number variable.names missing.percentage
## 1              1            BAD                  0
## 2              2           LOAN                  0
## 3              3        MORTDUE   8.69127516778524
## 4              4          VALUE   1.87919463087248
## 5              5         REASON   4.22818791946309
## 6              6            JOB   4.68120805369128
## 7              7            YOJ   8.64093959731544
## 8              8          DEROG   11.8791946308725
## 9              9         DELINQ   9.73154362416107
## 10            10          CLAGE   5.16778523489933
## 11            11           NINQ   8.55704697986577
## 12            12           CLNO    3.7248322147651
## 13            13        DEBTINC    21.258389261745
```

We see that we have missing values for all of the following variables except `BAD` and `LOAN`, so we will need to do some missing value imputation. The good news is that we do not have any missing values for the target variable `BAD`.

We can visualize the missing values in our data set in an attempt to try to identify any potential patterns in the missing values.

```
suppressMessages(library(VIM, quietly = TRUE))
aggr(dat, col=c('brown3', 'blue'), numbers = TRUE, labels = names(dat),
     cex.axis=.7, gap=5, ylab=c("Missing data", "Patterns"))
```

We have a warning saying we do not having enough vertical space because of the size of our data, but the subset of the data that is displayed in Figure 1 is still informative. We see that for many of the applications there are multiple missing values for each of the variable. In fact, for at least two applicants, almost all of the predictive variables are missing except a few. This appears to be very common for the subset of the data that is displayed. This is an interesting find since banks have become more conservative in terms of providing loans since the financial crisis that occurred in 2009.

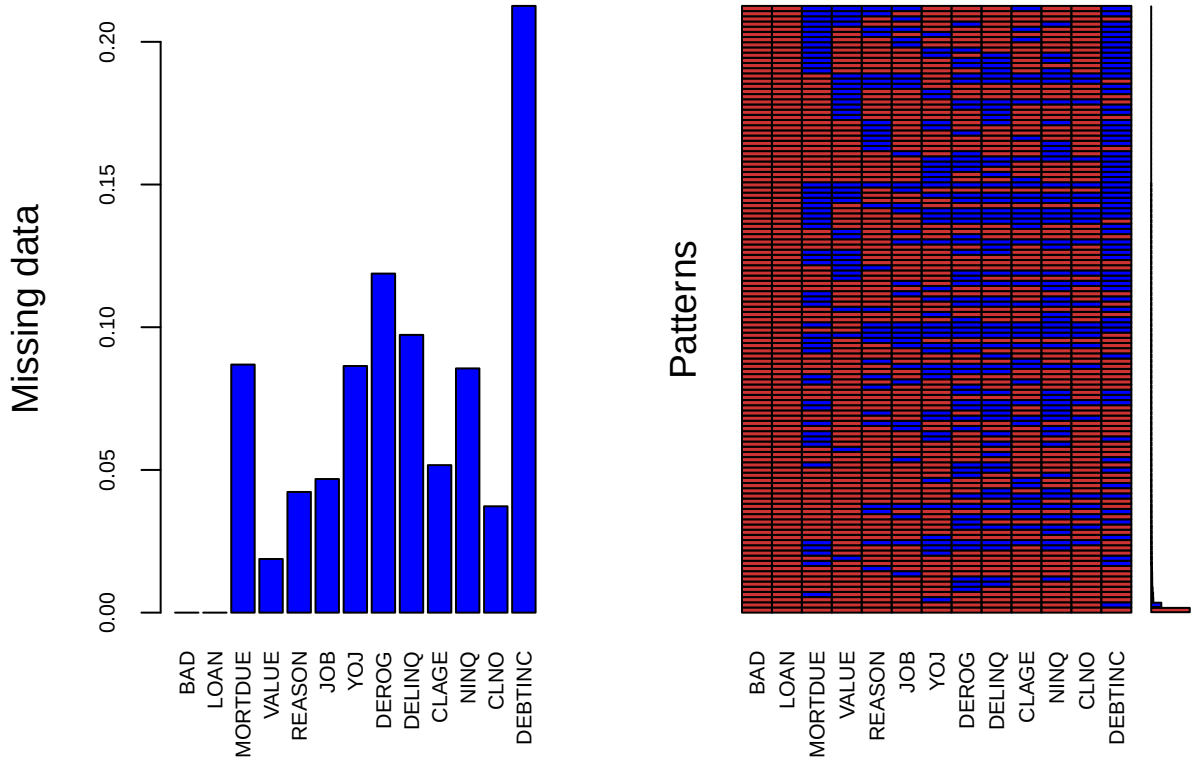

Figure 1: Plots detailing the missing values in our dataset. The plot on the left shows the proportion of missing values for each variable, and the plot on the right shows the pattern for the missing values with all of the variables.

2.2 Data cleaning

We now perform the data cleaning. We start by replacing the missing values for `JOB` and `REASON` with "Unknown", and print the frequency tables for each of these variables after replacement.

```
levels(dat$JOB) <- c(levels(dat$JOB), "Unknown")
dat$JOB[is.na(dat$JOB)] <- "Unknown"
levels(dat$REASON) <- c(levels(dat$REASON), "Unknown")
dat$REASON[is.na(dat$REASON)] <- "Unknown"
table(dat$JOB)
```

```
##
##     Mgr  Office   Other ProfExe   Sales    Self Unknown
##     767     948    2388    1276     109     193     279
```

```
table(dat$REASON)
```

```
##
## DebtCon HomeImp Unknown
##    3928    1780     252
```

Notice that 2,388 individuals had their employment classifies as "Other." When dealing with banks for loans, banks usually direct their attention to an applicant's employment - and future employment prospects - when deciding whether or not to approve a loan, as they need to know of the applicant will be capable of repaying the loan both now and in the future. Perhaps the real data has more employment classes than the data set we used, but working with the data on hand, this is interesting.

Now, we perform logarithmic transformations on `LOAN`, `VALUE`, `MORTDUE`, `YOJ`, and `CLAGE`. It should be noted that both `YOJ` and `CLAGE` had some zeros for recordings, thus we used the following code to avoid "domain errors" `log(x) := log(x + 1)` for these variables.

```
dat$LOAN <- log(dat$LOAN)
dat$VALUE <- log(dat$VALUE)
dat$MORTDUE <- log(dat$MORTDUE)
dat$YOJ <- log(dat$YOJ + 1)
dat$CLAGE <- log(dat$CLAGE + 1)
```

Next, we need to impute our missing values, so we decided to employ `mice()` from the `mice` package, as it is much quicker in terms of time completion relative to the `missForest` package's imputation.

```
library(mice, quietly = TRUE)
fit.mice <- mice(dat, m=1, maxit = 50, method = 'pmm', seed=500, printFlag = FALSE)
dat.imp <- complete(fit.mice, 1)
```

Finally, we store our known values into a vector - to be used for plotting later - and then delete them from the data set, so that we can proceed with the cluster analysis

```
labs <- dat.imp[, c(1)] # save for plotting labels later
dat.imp <- dat.imp[, -c(1)] # Remove BAD
```

2.3 Obtaining a distance matrix

We should note that our data set is of mixed types. That is, we have some continuous variables and some dichotomous variables in the same data set. Rather than use `daisy()` with `gower=TRUE` we will use `model.matrix` to prepare for the cluster analysis. This approach has some shortcomings, but we are content with them and their consequences, and thus proceed as follows.

We scale our data by columns using `scale()` in `R`. While this may seem reasonable initially, the reader is directed to Hastie, Tibshirani, and Friedman (2009) for a more in-depth treatment for some problems that may occur with this approach, but we will take their considerations into account and perform this operation none-the-less.

```
library(cluster)
dat.imp <- as.data.frame(dat.imp)
dat0 <- model.matrix(~.-1, data = dat.imp)
dat0.scaled <- apply(dat0, 2, scale)
```

The methods that we have chosen to use for cluster analysis do not require a distance matrix, but only the data, and since the project assignment requires that we compute a distance matrix then we do so here.

```
dat0.dist <- dist(dat0.scaled, method = "euclidean")
```

2.4 Clustering algorithm

We will use K-means and Partitioning Around Medoids (PAM) clustering for this project

2.4.1 *K*-means clustering

We need to determine the optimal number of clusters to be used, because this "optimal" value must be specified when performing the clustering. While there exists a variety of measures and techniques capable of accomplishing this task, we will only rely on the average silhouette width due to it's computational efficiency.

While some other readers may wish to rely exclusively on a scree plot and look for an elbow (e.g. the elbow method), this technique is too subjective and not much to our liking. Interestingly, Everitt and Hothorn (2011), state that the elbow method was initially developed for factor analysis from Cattell (1966), and that caution should be exercised when using this technique for other methods.

Other methods that can be used to determine the optimal number of clusters take much longer to compute than the silhouette width, even when decreasing the number of bootstraps.

```
suppressMessages(library(factoextra))
fviz_nbclust(dat0.scaled, FUNcluster = kmeans, method = "silhouette", k.max = 20)
```

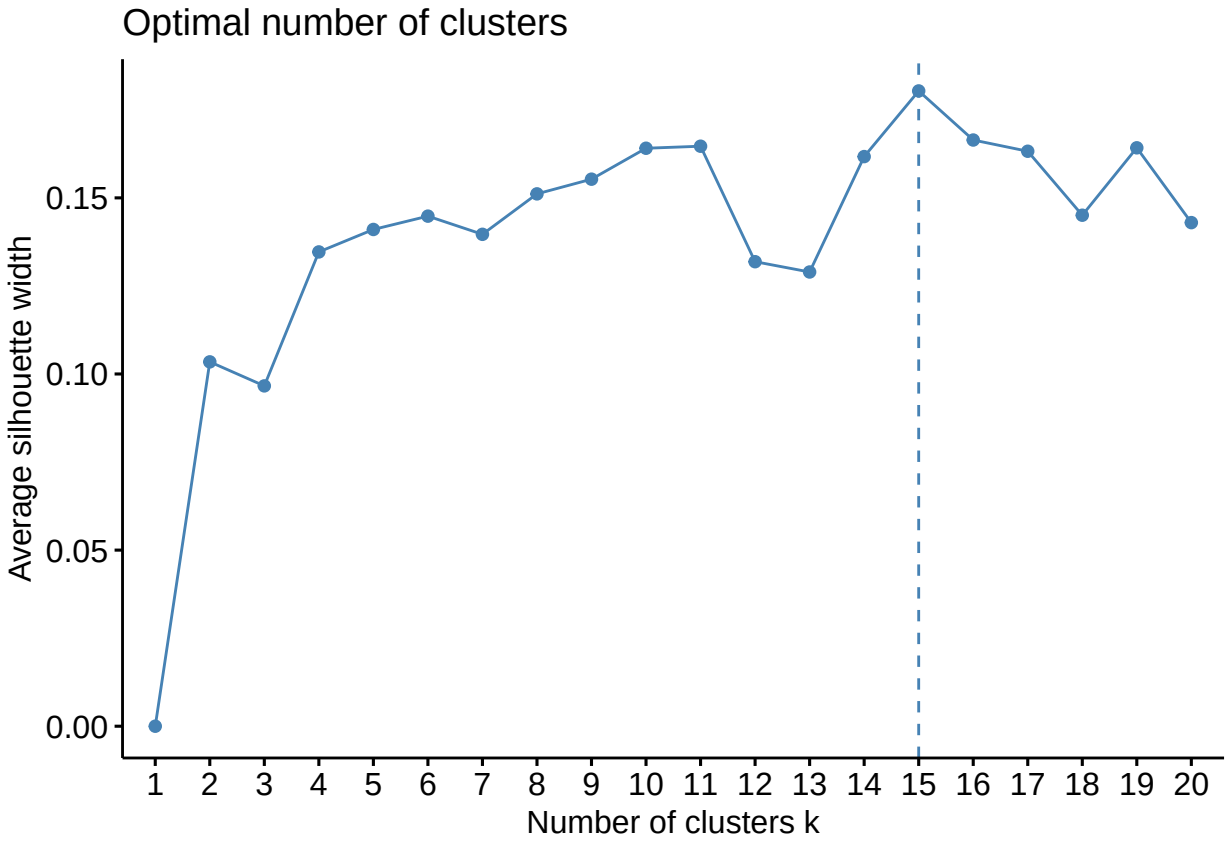

Figure 2: Plot for the average silhouette width used to determine the optimal number of clusters to be used in K-means clustering

We see that we should select 15 clusters for K-means, from Figure 2. It is worth noting that the width of the silhouette is not too great, and some sources would consider its width as "no substantial structure [having] been found" (Spector, 2007). Nonetheless, we will take this value found to be the optimal number of clusters.

Now, we focus on the arguments for our clustering. According to James, Witten, Hastie, and Tibshirani, (2013), they strongly recommend that `nstart` should be chosen to be between 20 and 50 because "otherwise an undesirable local optimum may be obtained." So, we choose the minimum of their recommendation. We also set the number of available colors to 15 as this is the maximum number of colors for this clustering method and the one that follows. Moreover, we have tinkered with the parameters for tSNE to achieve a somewhat maximum dispersion as outlined in Wattenberg, Viegas, and Johnson (2016) in Figure 3.

```
library(RColorBrewer)
palette(rainbow(15))
library(Rtsne)
set.seed(5474)
tsne <- Rtsne(dat0.scaled, dims=2, perplexity=50, max_iter=750)
km.cl <- kmeans(dat0.scaled, centers = 15, nstart = 20)
plot(tsne$Y, t="n", main="tSNE for km")
text(tsne$Y, labels = labs, col=km.cl$cluster)
abline(h=0, v=0, lty=2)
```

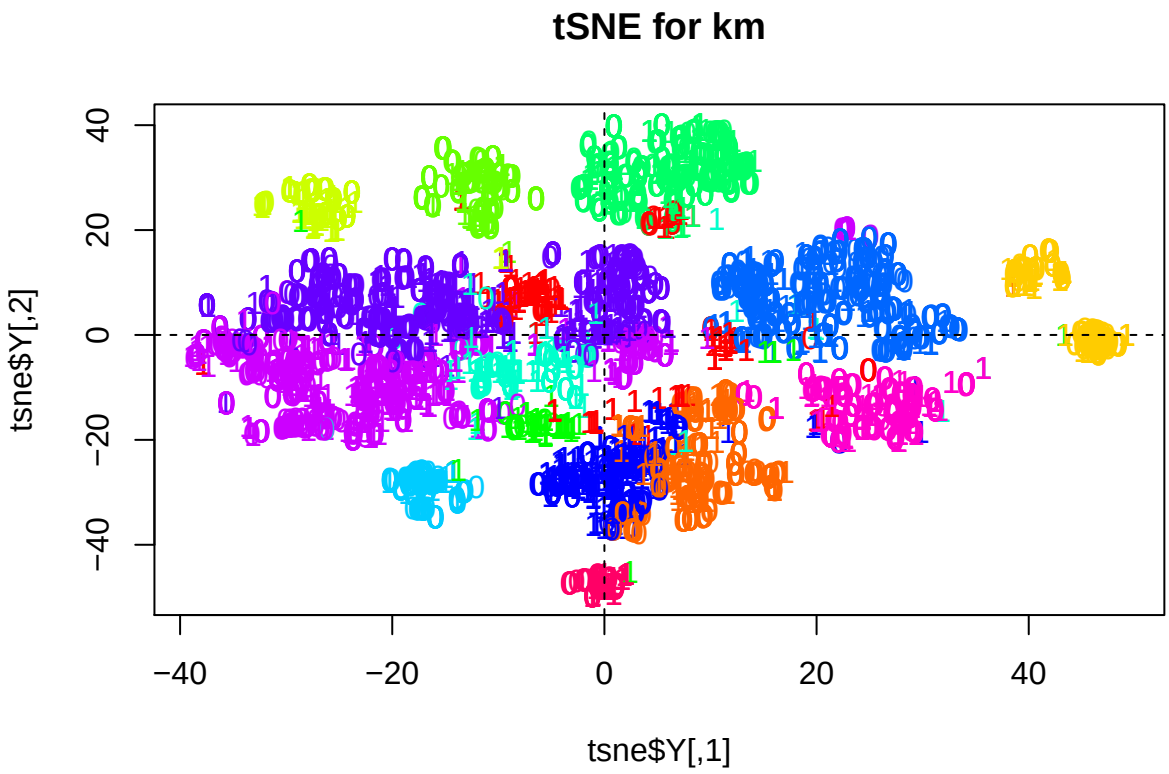

Figure 3: Plot for K-means clustering with t-SNE

Next we print the cluster memberships for our target variable BAD with the following code .

```
table(labs, km.cl$cluster)
```

```
##
## labs   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
##    0  53 397 204 135 268  21 514 131 159 696 201 845 735 341  71
##    1 142  62  37  56  25  55  71  69  10  88 162 150 167  58  37
```

There are several interesting findings with the output, but these findings will be discussed in

the *post hoc* analysis portion of this project.

2.4.2 Partition around medoids (PAM) clustering

The second method that we employ is Partitioning Around Medoids (PAM) clustering. This method has an advantage over K-means as one of the data points must be the center of each cluster; it is more computationally expensive, however.

We need to determine the optimal number of clusters with this method, and for consistency we again examine the silhouette width.

```
fviz_nbclust(dat0.scaled, FUNcluster = pam, method = "silhouette", k.max = 20) +
  labs(title = "Optimal number of clusters for PAM")
```

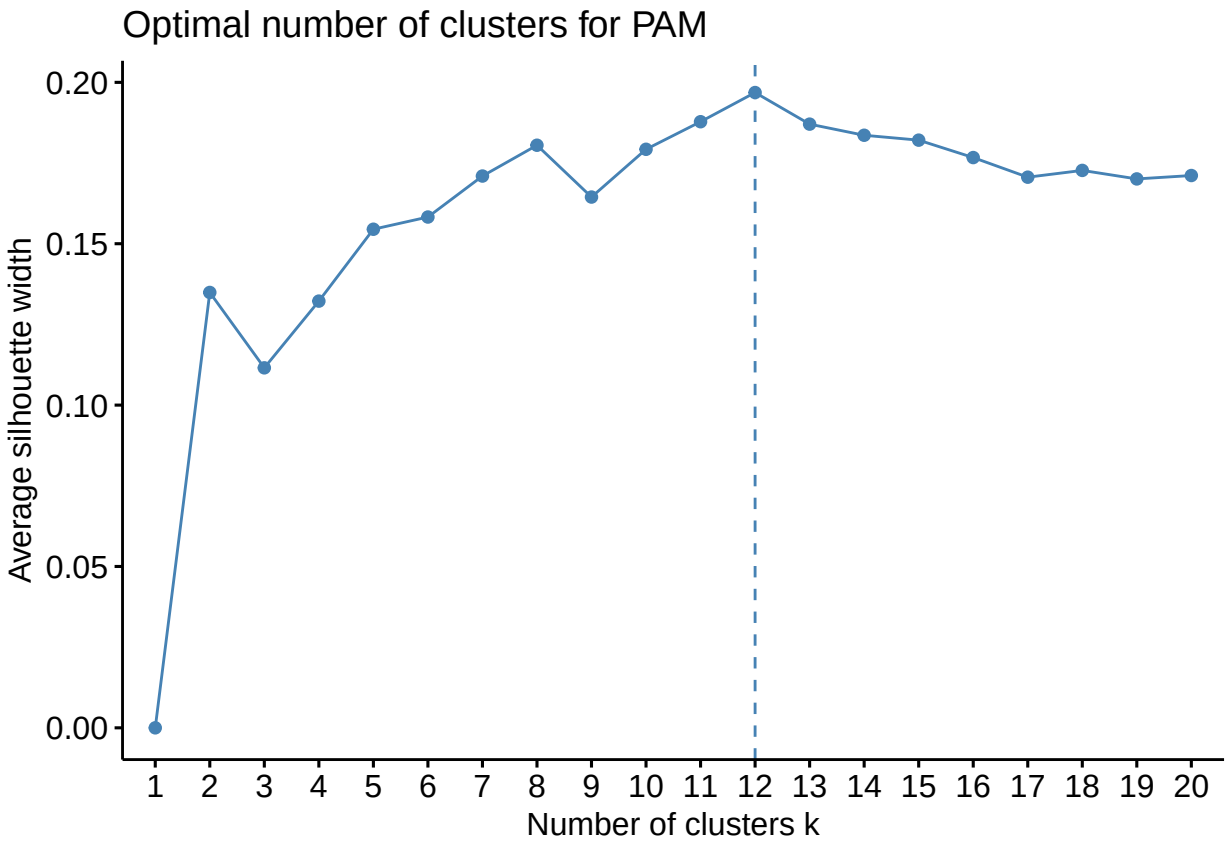

Figure 4: Plot for the average silhouette width used to determine the optimal number of clusters for PAM clustering

From Figure 4, we see that we should select 12 clusters, and again the width is fairly low. So, we use `pam()` with `k=12` to find these clusters with the code below, and again plot the memberships with tSNE.

```
plot(tsne$Y, t="n", main="tSNE for pam")
pam.cl <- pam(dat0.scaled, k=12, diss = FALSE)
text(tsne$Y, labels=labs, col=pam.cl$clustering)
abline(h=0, v=0, lty=2)
```

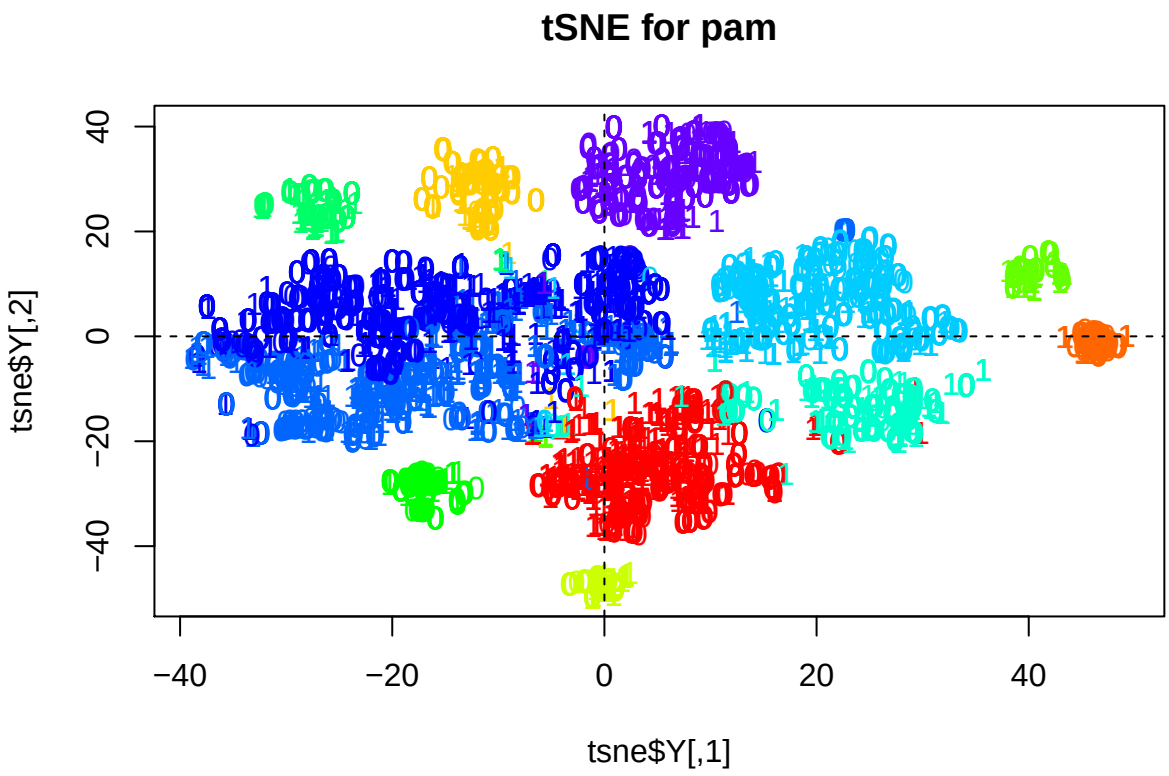

Figure 5: Plot for tSNE with PAM clustering

Next we print the cluster memberships for our target variable `BAD` with the following code.

```
table(labs, pam.cl$clustering)
```

```
##
## labs   1   2   3   4   5   6   7   8   9  10  11  12
##    0 574  97 268  71 107 159 135 375 709 776 966 534
##    1 249  10  33  38  32  13  58  82 134 230 218  92
```

Looking at the table it is difficult to see if there is a clear difference between the clusters and our target variable `BAD`. We will look into the cluster memberships for K-means shortly.

2.4.3 Compairing *K*-means and PAM

Now we compare the clustering results in the following two-way contingency table. We used the Jaccard and Rand similarity indices.

```
library(clusteval)
jac <- cluster_similarity(km.cl$cluster, pam.cl$clustering,
                          similarity = "jaccard")
ran <- cluster_similarity(km.cl$cluster, pam.cl$clustering,
                          similarity = 'rand')
matrix(c("Jaccard", jac, "Rand", ran), byrow = TRUE, ncol=2)
```

```
##      [,1]      [,2]
## [1,] "Jaccard" "0.594695996681332"
## [2,] "Rand"    "0.941624197114285"
```

In the table we see that the two methods are fairly similar. This was somewhat expected, as the methods are similar, but I did not expect the difference of the indices to be as large as it is.

2.5 *Post hoc* analysis

We first investigate the relationship of the clusters and the applicant's occupation. The table below shows this relationship.

```
tableJOB <- table(dat$JOB, km.cl$cluster)
tableJOB
```

```
##
##              1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
##   Mgr       44 104  19   0   0  11   0  66   0   0  37 354 113  19   0
##   Office    36   0  27   0 293   3 585   4   0   0   0   0   0   0   0
##   Other     69 355  66   0   0  40   0 122   0   0 319 639 774   4   0
##   ProfExe   45   0  23   0   0  16   0   8   0 784   7   2  15 376   0
##   Sales      0   0   0   0   0   1   0   0   0   0   0   0   0   0 108
##   Self       1   0   0 191   0   1   0   0   0   0   0   0   0   0   0
##   Unknown    0   0 106   0   0   4   0   0 169   0   0   0   0   0   0
```

We see that for most clusters one occupation dominates that cluster. For example, cluster 9 is made up exclusively with the `JOB` being Unknown, cluster 10 being `ProfExe`, cluster 4 begin entirely `Self`, and cluster 15 being only `Sales`.

Next we want to investigate the relationship of the clusters and the applicant's reason for requesting a loan.

```
tableNINQ <- table(dat$REASON, km.cl$cluster)
tableNINQ
```

```
##
##              1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
##   DebtCon  157   0   0  72   0  50 585 174 113 784   0 995 902   0  96
##   HomeImp   36 459   0 114 293  22   0  26  56   0 363   0   0 399  12
##   Unknown    2   0 241   5   0   4   0   0   0   0   0   0   0   0   0
```

Again, we see that for some clusters the reason for requesting a loan dominates. For example, clusters 12 and 13 only have applicants that are seeking debt consolidation, while clusters 2, 11,and 14 only contain applicants that are seeking funds for home improvement.

Last but not least, we want to investigate the relationship of the clusters and our target variable `BAD`.

```
tableBAD <- table(dat$BAD, km.cl$cluster)
tableBAD
```

```
##
##       1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
##   0  53 397 204 135 268  21 514 131 159 696 201 845 735 341  71
##   1 142  62  37  56  25  55  71  69  10  88 162 150 167  58  37
```

Here we do not have any clear indications as to whether BAD influenced cluster assignment. Since we do not observe any zeros, we perform a χ^2 test to see if `BAD` is independent of cluster assignment.

```
chisq.test(tableBAD)
```

```
##
##  Pearson's Chi-squared test
##
## data:  tableBAD
## X-squared = 804.28, df = 14, p-value < 2.2e-16
```

We see that our p-value is essentially zero, and thus the result of this test is significant for any reasonable level of significance α. So, we conclude that `BAD` is not independent of cluster membership.

3 Discussion

Due to the large number of clusters in both K-means and PAM, the coloring for the tSNE plots is not ideal. I have experimented with different color schemes, but ultimately settled for `rainbow(15)`. I say that it is not ideal because the plots are not colorblind friendly, and it can be difficult to separate several shades of blue and green, for those not color blind.

4 References

- Cattell, R. B. (1966), "The scree test for the number of factors," *Multivariate Behavioural Research*, 1, 245-276
- Everitt, B., Hothorn, T. (2011), *An Introducion to Applied Multivariate Analysis with R*, New York: Springer.
- Hastie, T., Tibshirani, R., and Freidman, J., (2009), *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, New York: Springer.
- James, G., Witten, D., Hastie, and Tibshairani, R. (2013), *An Introduction to Statistical Learing with Applications in R*, New York: Springer.
- Spector, P. (2007), "Performing and interpreting cluster analysis," Retrieved from: `https://www.stat.berkeley.edu/ spector/s133/Clus.html`
- Wattenberg, M., Viégas, F., Johnson, I. (2016), "How to use t-SNE effectively" Retrieved from: `https://distill.pub/2016/misread-tsne/`