Project 2

2022-09-25

Setting the Seed

```
set.seed(0)
```

1. Reading the Data

```
train <- read.table(file=
"http://archive.ics.uci.edu/ml/machine-learning-databases/optdigits/optdigits.tra",
sep=",", header = FALSE, na.strings = c("NA", "", " "),
col.names = c(paste("x", 1:64, sep=""), "digit"))

test <- read.table(file=
"http://archive.ics.uci.edu/ml/machine-learning-databases/optdigits/optdigits.tes",
sep=",", header = FALSE, na.strings = c("NA", "", " "),
col.names = c(paste("x", 1:64, sep=""), "digit"))
```

```
dim(train);  dim(test)
```

```
## [1] 3823   65
```

```
## [1] 1797   65
```

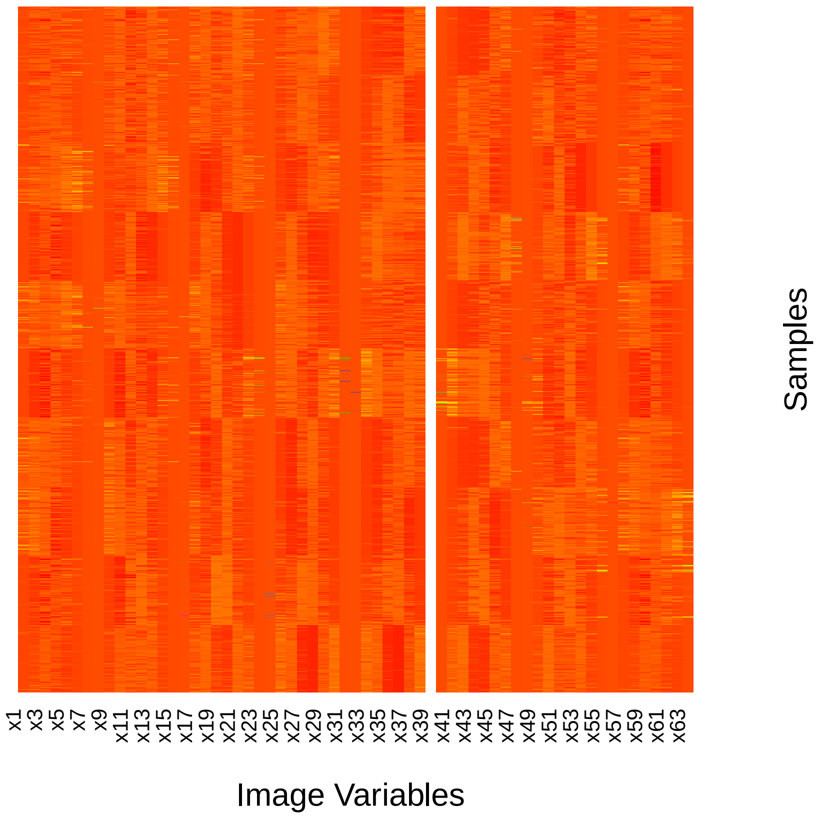
```
dat <- rbind(train, test); dim(dat)
```

```
## [1] 5620   65
```

2. Exploratory Data Analysis

```
dat0 <- data.matrix(dat[order(dat$digit), -65])
n <- NROW(dat0)
color <- rainbow(n, alpha = 0.8)
heatmap(dat0, col=color, scale="column", Rowv=NA, Colv=NA,
labRow=FALSE, margins=c(4,4), xlab="Image Variables", ylab="Samples",
main="Heatmap of Handwritten Digit Data")
```

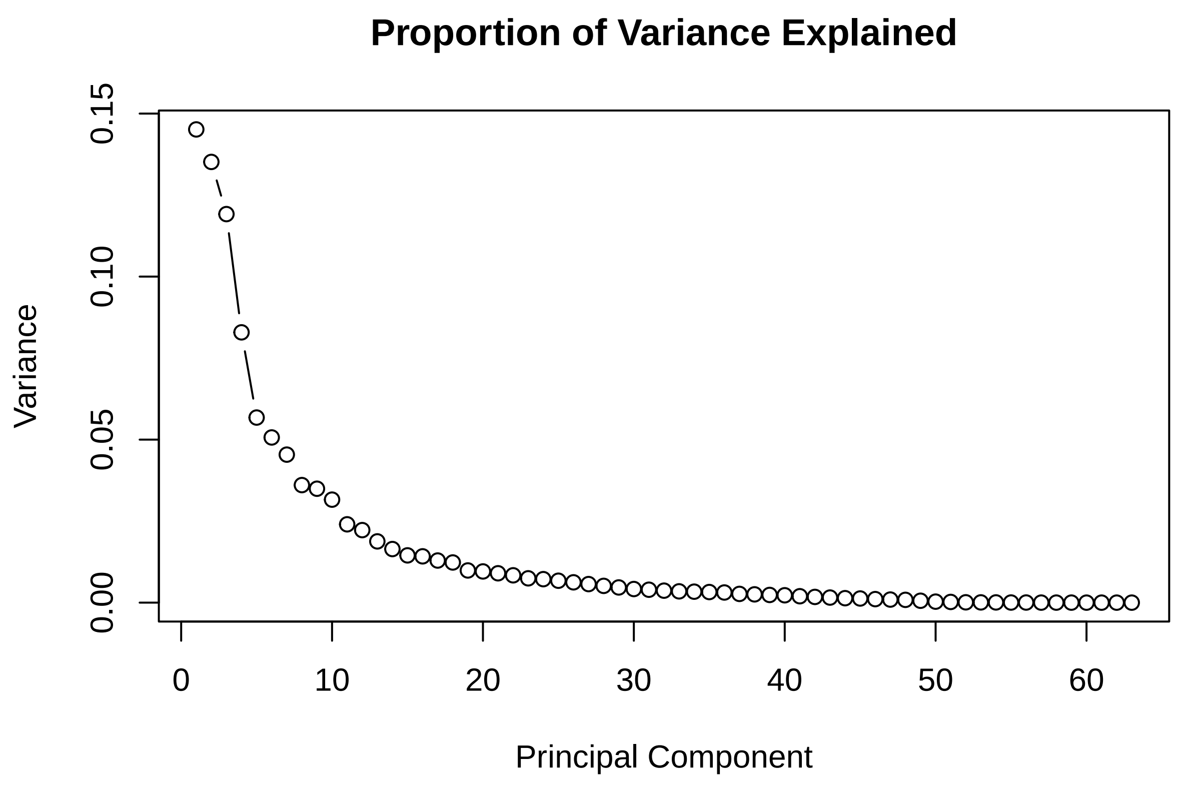
Heatmap of Handwritten Digit Data



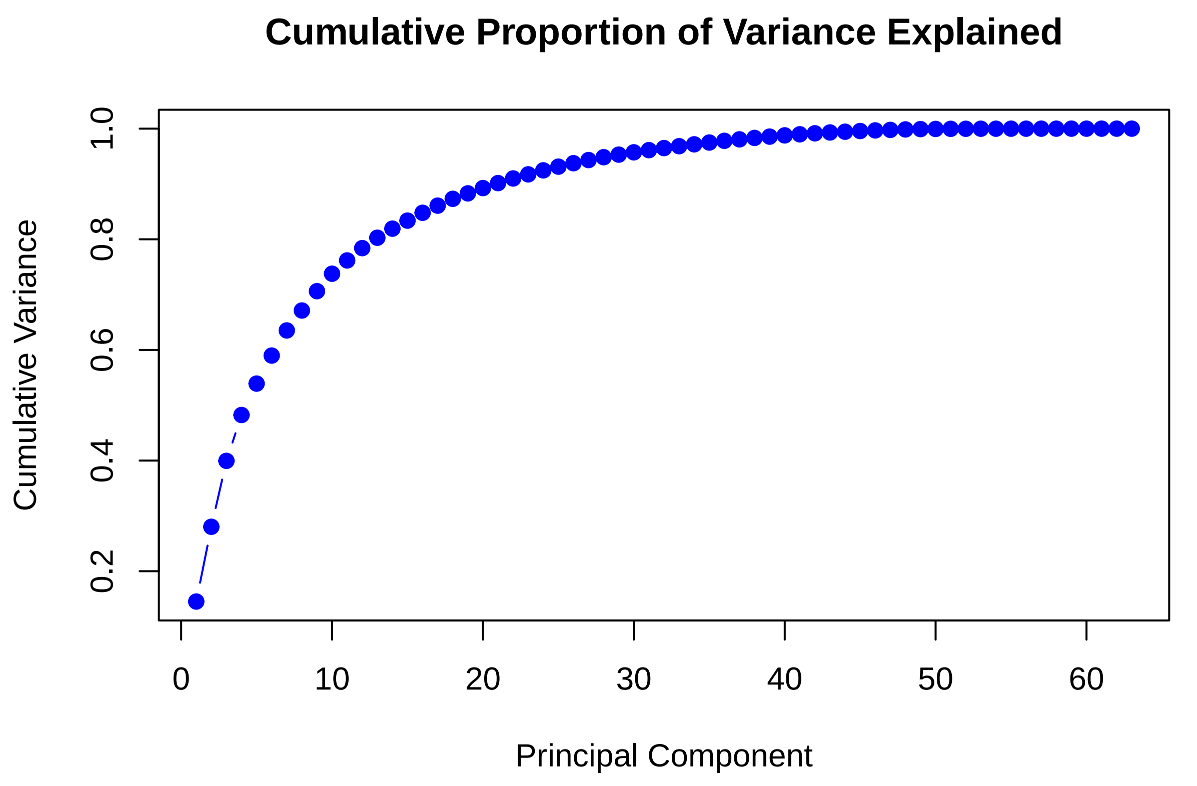
The most salient aspect of the heat map is that there is a column (variable) where all of the values are zero. This corresponds to column x40 and is depicted on the heat map as a single white line. We will remove this column for the ensuing analyses.

3. Principal Components Analysis

```
dat_clean=subset(dat, select =-c(x40, digit))
pca.res <- prcomp(dat_clean, retx=TRUE)
var.pc <- (pca.res$sdev)^2
prop.pc <- var.pc/sum(var.pc)
plot(prop.pc, xlab = "Principal Component",
ylab = "Variance", type = "b", main="Proportion of Variance Explained")
```



```
plot(cumsum(prop.pc), xlab = "Principal Component", col="blue",
ylab = "Cumulative Variance" ,
main = "Cumulative Proportion of Variance Explained",
type = "b", pch=19)
```

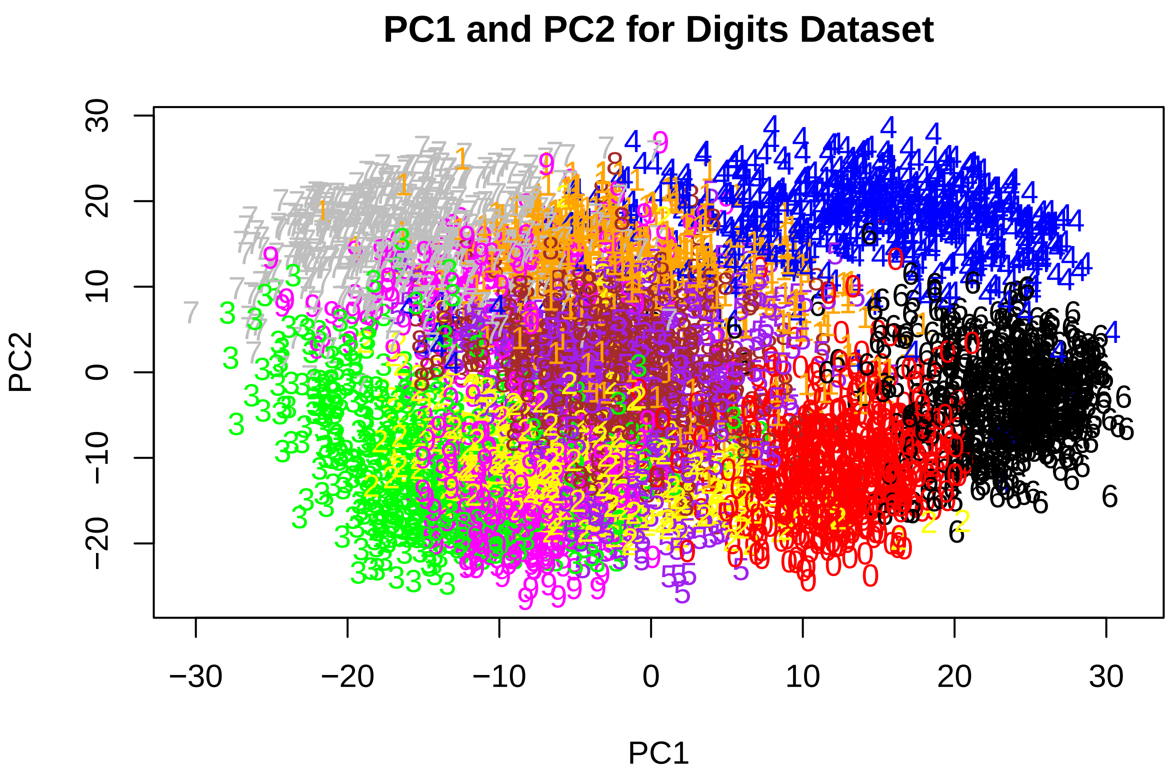


```
a1.a2 <- pca.res$rotation[,1:2]
a1.a2
```

```
##                PC1           PC2
## x1   8.487259e-19 -8.856109e-19
## x2  -1.515843e-02 -1.375129e-02
## x3  -1.648022e-01 -1.754153e-01
## x4  -9.076625e-02 -1.521058e-01
## x5  -9.877730e-02  8.126194e-02
## x6  -2.086157e-01  1.466983e-01
## x7  -8.499057e-02  8.873405e-02
## x8  -9.521817e-03  1.518466e-02
## x9  -5.037188e-05 -8.230294e-05
## x10 -7.324980e-02 -1.019224e-01
## x11 -1.266392e-01 -2.680143e-01
## x12  1.082640e-01  4.371570e-02
## x13 -7.854119e-02  1.559593e-02
## x14 -2.718339e-01  3.235654e-02
## x15 -1.090733e-01  1.216740e-01
## x16 -5.762094e-03  1.544607e-02
## x17 -4.248153e-05  1.170904e-04
## x18 -1.985927e-02 -1.008243e-01
## x19  1.788911e-01 -1.165633e-01
## x20  1.750365e-01  1.417578e-01
## x21 -2.125508e-01 -2.481575e-02
## x22 -2.252476e-01  3.177368e-02
## x23 -1.979210e-02  8.751328e-02
## x24  1.141274e-03  5.716743e-03
## x25  2.651465e-05  7.051974e-05
## x26  7.344722e-02  2.074697e-03
## x27  2.726804e-01  6.197098e-02
## x28 -8.692464e-02  6.238655e-02
## x29 -3.298888e-01  2.303801e-02
## x30 -1.309033e-01  1.057520e-01
## x31  2.933372e-02  5.650600e-02
## x32  2.350635e-04  3.029077e-04
## x33  8.883157e-05  1.336067e-04
## x34  1.401270e-01  6.345165e-02
## x35  3.161810e-01  1.958917e-01
## x36 -5.519794e-03  2.228880e-01
## x37 -1.090754e-01  2.230123e-01
## x38  7.116774e-03  6.628375e-02
## x39  3.864416e-02 -2.344849e-02
## x41  2.304827e-03  2.409756e-03
## x42  1.074462e-01  4.372237e-02
## x43  3.425818e-01  3.745742e-02
## x44  9.097143e-02  2.458343e-01
## x45  5.700196e-03  3.615449e-01
## x46  1.163051e-01 -5.560767e-02
## x47  1.050426e-01 -1.820152e-01
## x48  2.319728e-03 -4.656431e-04
## x49  1.325624e-03  1.184741e-03
## x50  1.255747e-02 -1.913966e-02
## x51  9.160666e-02 -2.357817e-01
## x52  3.351973e-02  5.242170e-02
## x53  4.498532e-02  1.301683e-01
## x54  9.241704e-02 -2.839868e-01
## x55  1.002923e-01 -2.153388e-01
## x56  9.068402e-03 -5.706071e-03
## x57 -5.690691e-06  1.257183e-05
## x58 -1.295820e-02 -1.162726e-02
## x59 -1.913538e-01 -1.661979e-01
## x60 -7.705514e-02 -1.618808e-01
## x61  1.574341e-01 -1.800538e-01
## x62  1.052628e-01 -2.368844e-01
## x63  3.690494e-02 -1.029558e-01
## x64  5.466823e-05 -1.243701e-02
```

```
assign_col=function(number) {
  switch(number+1, "red", "orange", "yellow",
"green", "blue", "purple", "black", "grey", "brown", "magenta")
}

dat$colors=sapply(dat$digit, assign_col)
plot(pca.res$x[,1:2], pch="", main="PC1 and PC2 for Digits Dataset")
text(pca.res$x[,1:2], labels = dat$digit, col = dat$colors)
```



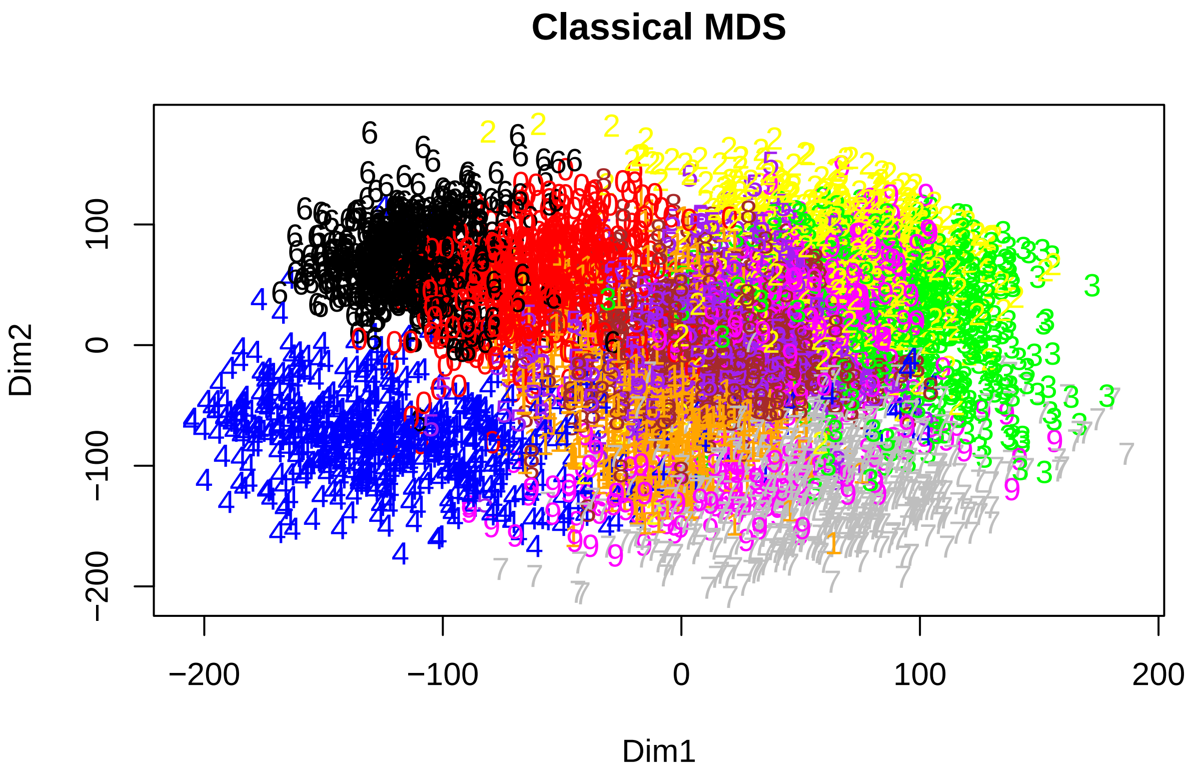
By plotting each number against the first two most important principal components, I notice even just two components do an okay job at clustering the numbers. In particular, a pattern I observe is that when travelling radially around the perimeter of the ellipse above, we see some defined clusters corresponding to 4, 6, 0, 3, and 7. Numbers 1,2,5,8 and 9 show a lot more overlap and are present in the center of the ellipse.

4. Multidimensional Scaling

```
d <- dist(dat_clean, method = "manhattan")
mds <- cmdscale(d,eig=TRUE, k=2)
```

```
x <- mds$points[,1]; y <- mds$points[,2]
```

```
plot(x, y, xlab="Dim1", ylab="Dim2",
  main="Classical MDS", pch="")
text(x, y, labels = dat$digit, col = dat$colors)
```

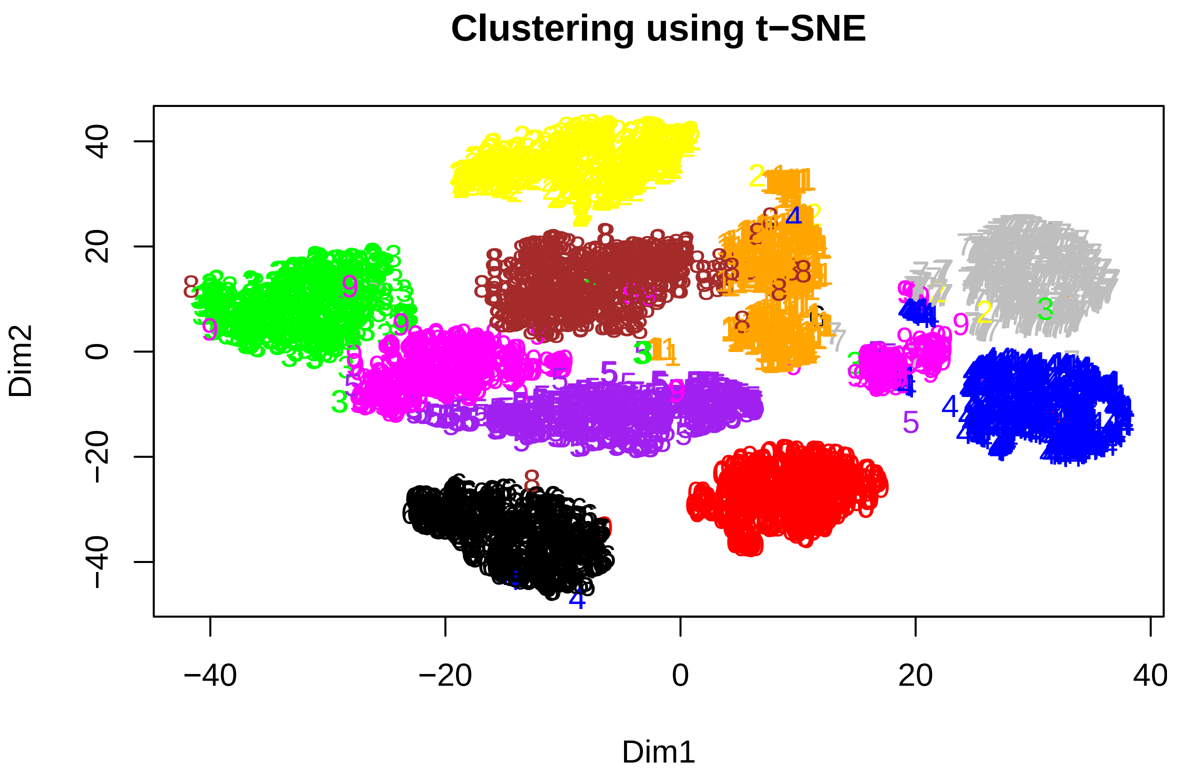


When I used classical MDS with a Manhattan-based distance matrix, I still observed distinct clusters in the periphery and a lot of overlap in the center. Of course, the scale of the dimensions was much different (200 by 200 instead of 30 by 30). Also, the clusters were in different positions in the graph. Interestingly, this approach resulted in less distinct clusters. Based on my subjective interpretation of the graph above, I think only the numbers 4,7, 3 and 6 are separated well. It would be interesting to try many different distances (in addition to the usual Euclidean and Manhattan).

5. Applying t-SNE

```
library(Rtsne)
t=Rtsne(d, dims=2, num_threads=0)
```

```
plot(t$Y, xlab="Dim1", ylab="Dim2",
  main="Clustering using t-SNE", pch="")
text(t$Y, labels = dat$digit, col = dat$colors)
```



From an initial inspection, we can observe t-SNE dramatically outperforms the other clustering methods. All numbers have their own contiguous clusters (for the most part). Also, there is little overlap between numbers. In summary, PCA and MDS performed similarly and t-SNE outperformed these other methods.