Final-Project

Arman Isakhani Mamaghani

February 6, 2016

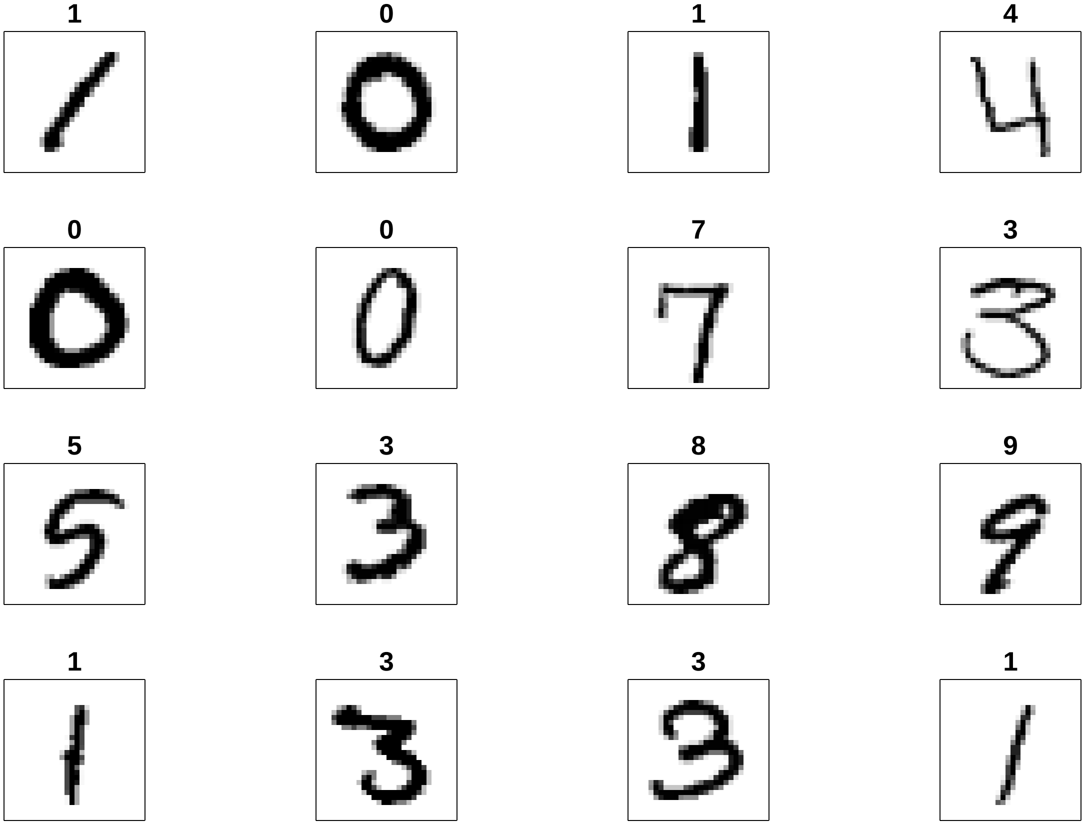
This project is about digit recognition. let first load the data:

```r
rm(list=ls())
library("data.table", lib.loc="~/R/win-library/3.2")

tab5rows <- read.csv("train.csv", header = TRUE, nrows = 5)
classes <- sapply(tab5rows, class)
pixels <- as.matrix(read.csv("train.csv", header = TRUE, colClasses = classes))
rm(classes , tab5rows)

lables <- pixels[,1]
pixels <- pixels[,-1]
N <- nrow(pixels)
```

Every digit is a 28*28 pixels and intensity of eac pixels has intensity between (0,256).

Let Plot some of the digits:

```r
colors<-c('white','black')
cus_col<-colorRampPalette(colors=colors)
par(mfrow=c(4,4),pty='s',mar=c(1.5,1.5,1.5,1.5),xaxt='n',yaxt='n')
for(i in 1:16)
{
  z<-array(pixels[i,],dim=c(28,28))
  z<-z[,28:1] ##right side up
  image(z,main=lables[i],col=cus_col(256))
}
```



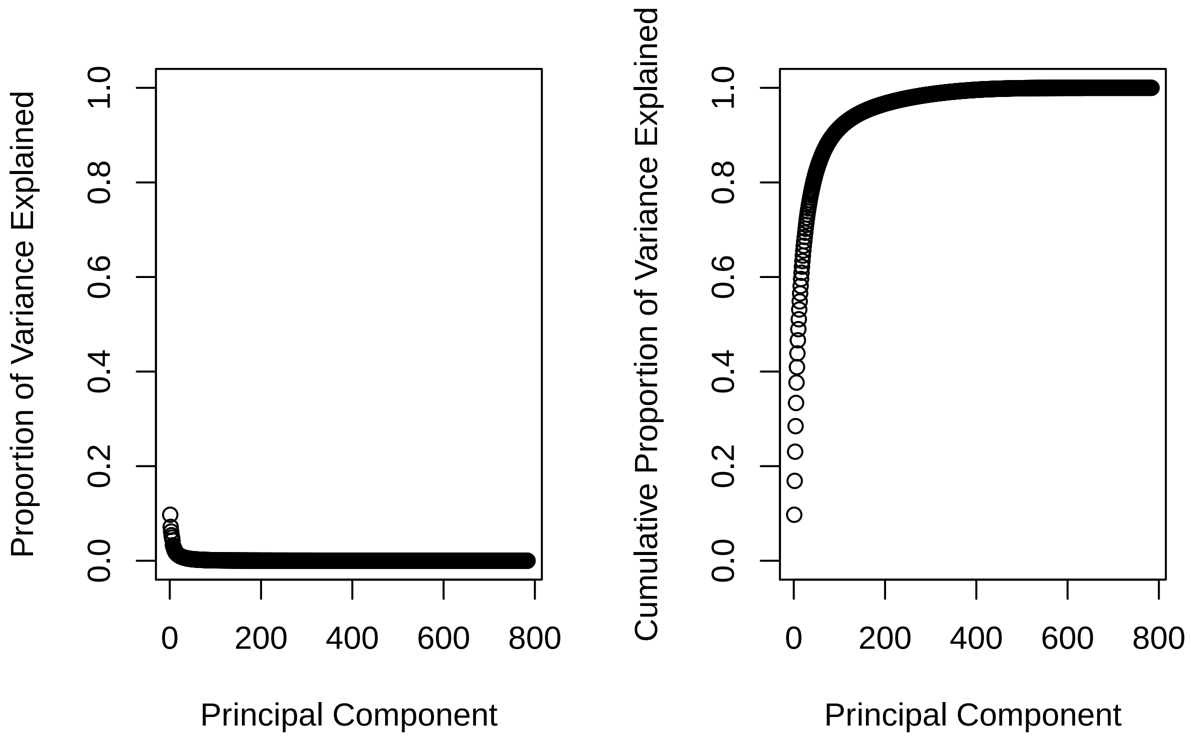
First we split the data to training and test data:

```
set.seed(1)
train <- sample(c(TRUE,FALSE), nrow(pixels),rep=TRUE,prob = c(0.8,0.2))
sum(train)/nrow(pixels)
```

```
## [1] 0.7999286
```

We will find principle components and see which proportion of variance is explained by these components:

```
pr.out <- prcomp(pixels[train, ])
pr.var <- pr.out$sdev^2
pve <- pr.var/sum(pr.var)
resetPar <- function() {
  dev.new()
  op <- par(no.readonly = TRUE)
  dev.off()
  op
}
par(resetPar())
par(mfrow = c(1,2))
plot(pve, xlab="Principal Component", ylab="Proportion of Variance Explained", ylim=c(0,1),type='b')
plot(cumsum(pve), xlab="Principal Component", ylab="Cumulative Proportion of Variance Explained", ylim=
```



Now we rotate our train and test data to new component space:

```
test.rotated  <- pixels[!train, ] %*% pr.out$rotation
train.rotated <- pixels[train, ] %*% pr.out$rotation
```

Let try to predict the test data by fitting a lda Model to train data:

(Note: we test offline for choosing the best number of components to use and 60 was the best)

```
library(MASS)

rotated.Data <- pixels %*% pr.out$rotation
rotated.Data <- data.frame(cbind("lables" =lables, rotated.Data[,1:60]))

lda.fit=lda(lables ~ . ,data = rotated.Data, subset = train)
lda.pred=predict(lda.fit, rotated.Data[!train,])
table(lda.pred$class ,lables[!train])
```

```
##
##       0   1   2   3   4   5   6   7   8   9
##   0 769   0   4   3   0  12   6   3   8   4
##   1   0 916  32  19  11  11   6  29  51   4
##   2   3   4 706  28   6   4   7   9   2   2
##   3   6   1  20 734   0  34   2   5  33  12
##   4   2   0  15   2 705  13  11  18   8  49
##   5  22   5   8  28   3 597  33   6  36   7
##   6  10   1  19   3  11  15 732   2   7   1
##   7   0   1  22  17   0   4   0 710   1  17
##   8  11  21  37  19   9  35   8   5 642   4
##   9   2   2  10  20  82  10   1  53  25 760
```

Finding the test Error:

```
result.lda <- data.table("predict" = lda.pred$class, "lable" = lables[!train])
result.lda[, correct := lable == predict]
print(paste0('Test error is: ', sum(result.lda[, correct])/ nrow(result.lda)))
```

```
## [1] "Test error is: 0.865286207306914"
```

Let's try to predict the test data by fitting a KNN Model to train data:

(Note: we test offline for choosing the best number of components to use and 50 was the best.)

```
library(FNN)
```

```
## Warning: package 'FNN' was built under R version 3.2.3
```

```
knn.pred <- knn(train.rotated[,1:50], test.rotated[,1:50],lables[train] ,k=10)
table(knn.pred ,lables[!train])
```

```
##
## knn.pred   0   1   2   3   4   5   6   7   8   9
##        0 820   0   3   0   0   0   4   0   2   2
##        1   1 948   1   4   7   2   0  16   8   1
##        2   0   1 846   3   0   1   1   3   2   1
##        3   0   0   4 842   0   7   0   0  12   3
##        4   0   0   0   0 797   0   3   3   0  10
##        5   1   0   0   6   0 716   2   0  13   3
##        6   3   1   1   1   3   6 796   0   6   1
##        7   0   0  13   8   1   1   0 813   3  15
##        8   0   0   4   5   0   1   0   0 761   2
##        9   0   1   1   4  19   1   0   5   6 822
```

Finding the test Error:

```
result.knn <- data.table("predict.knn" = knn.pred, "lable" = lables[!train])
result.knn[, correct := lable == predict.knn]
print(paste0('Test error is: ', sum(result.knn[, correct])/ nrow(result.knn)))
```

```
## [1] "Test error is: 0.97120076163275"
```

In final let look at the letter numbers that knn predict uncorrect:

I run it before and attach the result as a pdf with the name 'not_correct_test_letters.pdf'

Note: the numbers that is written above the pictures is the predicted number by knn.

```r
not.correct.predicted <- cbind(knn.pred[!result.knn[,correct]],
                               (pixels[!train, ])[!result.knn[,correct],])

colors<-c('white','black')
cus_col<-colorRampPalette(colors=colors)
f <- function(m) t(m)[,nrow(m):1]

pdf('not_correct_test_letters.pdf')
par(mfrow=c(4,4),pty='s',mar=c(2,2,2,2),xaxt='n',yaxt='n')
for(i in 1:nrow(not.correct.predicted))
{
  M = matrix(not.correct.predicted[i,-1],c(28,28) , byrow =TRUE)
  image(1:28, 1:28, f(M),main=not.correct.predicted[i,1]-1, col = cus_col(256))
}
dev.off()
```

```
## pdf
##   2
```