HW 5 Solution

Issac Li (zl368)

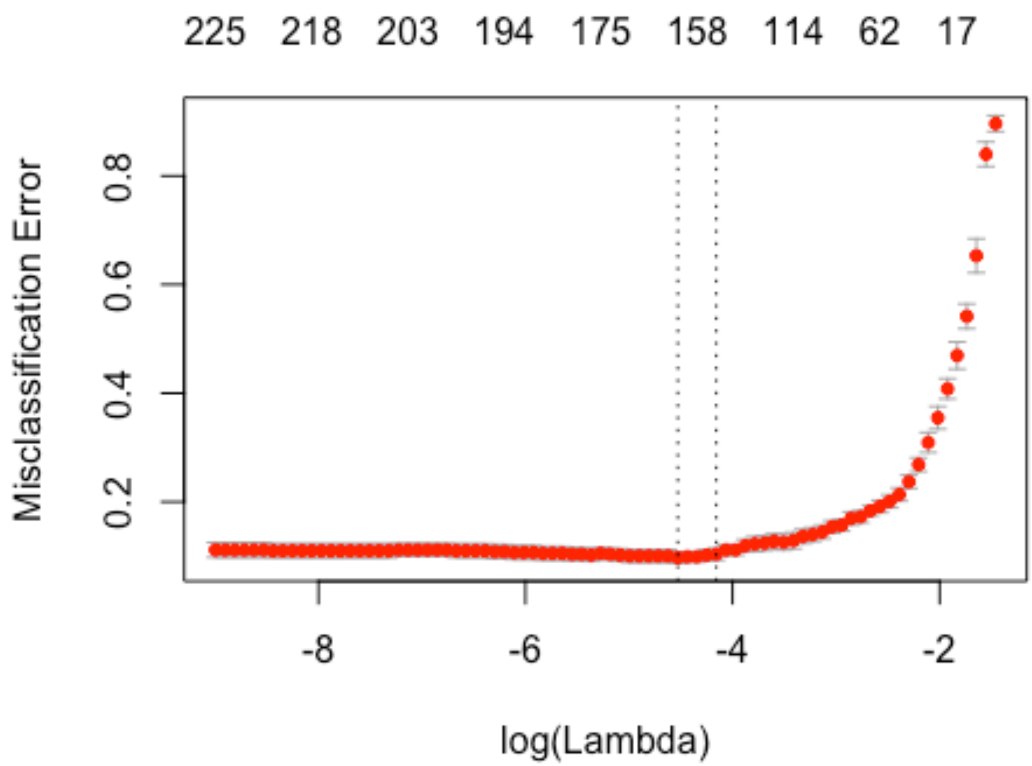
3/1/2017

Part 1

```
require(glmnet)
train <- read.csv("~/Documents/STAT665/HW5/digits_train.csv",stringsAsFactors
= FALSE)
valid <- read.csv("~/Documents/STAT665/HW5/digits_valid.csv", stringsAsFactor
s = FALSE)
X_train <- as.matrix(train[, 1:256])
y_train <- as.factor(train[, 257])
X_valid <- as.matrix(valid[, 1:256])
y_valid <- as.factor(valid[, 257])

# Grouped Lasso Regression
set.seed(111)
fit1 <- cv.glmnet(X_train, y_train, family = "multinomial",
                  type.multinomial = "grouped", type.measure = "class", nfold
s = 10)
```

1)

```
plot(fit1)
```




```r
2)

# Value of lam_min
cat(fit1$lambda.min)

## 0.01079447

# Value of lam_1se
cat(fit1$lambda.1se)

## 0.01566093

# Make Predictions
fit1.min <- glmnet(X_train, y_train, family = "multinomial",
                          type.multinomial = "grouped", lambda = fit1$lambda.
min)
fit1.1se <- glmnet(X_train, y_train, family = "multinomial",
                          type.multinomial = "grouped", lambda = fit1$lambda.
1se)
y_pred.min <- as.factor(predict(fit1.min, newx = X_valid, type = "class"))
y_pred.1se <- as.factor(predict(fit1.1se, newx = X_valid, type = "class"))
```

```r
# MSEs of the two standards
pred1 <- data.frame(Lambda = c("Min", "1SE"),
                    MeanError =signif(c(mean(y_pred.min != y_valid),
                                        mean(y_pred.1se != y_valid)),5))
pred1
```

```
##   Lambda MeanError
## 1    Min   0.11950
## 2    1SE   0.12893
```

3)

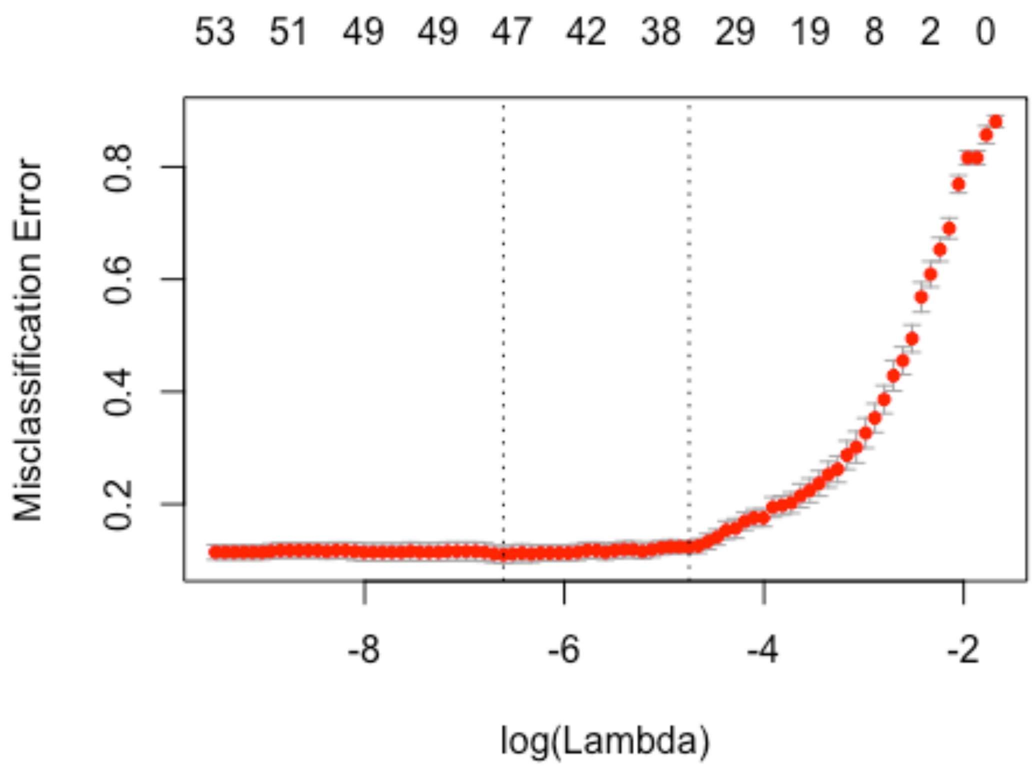
```r
# Number of non-zero coefficients.
sum(coefficients(fit1.1se)[[1]]!=0)
```

```
## [1] 153
```

The number of predictors (excluding the intercept) is 153-1 = 152 in this case for grouped regression. The number of non-zero coefficients is 153*10=15300.

4)

```r
# Ungrouped Lasso Regression

fit1_ug <- cv.glmnet(X_train, y_train, family = "multinomial",
                     type.multinomial = "ungrouped", type.measure = "class",
                     nfolds = 10)
plot(fit1_ug)
```

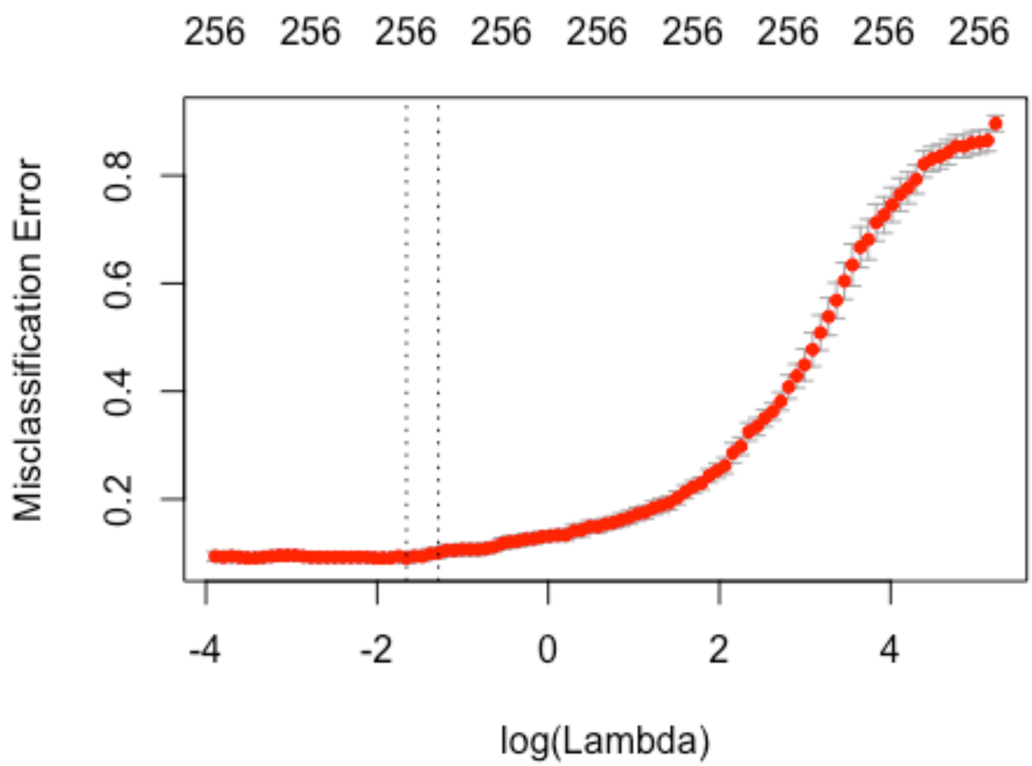



```
# Value of lam_min
cat(fit1_ug$lambda.min)

## 0.001345828

# Value of lam_1se
cat(fit1_ug$lambda.1se)

## 0.008651081

y_pred_ug.min <- predict(fit1_ug, newx=as.matrix(valid[,-257]), s=fit1_ug$lam
bda.min ,type="class")
y_pred_ug.1se <- predict(fit1_ug, newx=as.matrix(valid[,-257]), s=fit1_ug$lam
bda.1se ,type="class")

# Show MSEs
pred1_ug <- data.frame(Lambda = c("Min", "1SE"),
                          MeanError = signif(c(mean(y_pred_ug.min != y_valid)
,
                                        mean(y_pred_ug.1se != y_valid)),5))
pred1_ug
```

```
##   Lambda MeanError
## 1    Min   0.13836
## 2    1SE   0.16038

# number of predictors
coefs <- coef.cv.glmnet(fit1_ug,s="lambda.1se")

# find predictors used for each class and then use the union
# to find number of unique predictors.
predictors=sapply(coefs,function(x) ifelse(x!=0,1,0))
n_pred=sum(apply(predictors,1,function(x) ifelse(sum(x)==0,0,1)))

cat("Number of predictors:",n_pred-1)

## Number of predictors: 196

# number of non-zero coefficients
coef_n=sapply(coefs,function(x) sum(x!=0))

# number of non-zero coefficients per class
cat("Number of non-zero coefficients per class:")

## Number of non-zero coefficients per class:

coef_n

##  0  1  2  3  4  5  6  7  8  9
## 22 50 37 32 38 29 38 45 26 38

# number of total non-zero coefficients
cat("Number of total non-zero coefficients:",sum(coef_n))

## Number of total non-zero coefficients: 355
```

Part 2

```
set.seed(111)
fit2 <- cv.glmnet(X_train, y_train, family = "multinomial",
                  type.measure = "class", nfolds = 10, alpha = 0)
plot(fit2)
```




```r
# Value of lam_min
cat(fit2$lambda.min)

## 0.1907676

# Value of lam_1se
cat(fit2$lambda.1se)

## 0.2767712

# Make Predictions

y_pred2.min <- as.factor(predict(fit2, newx = X_valid, s = fit2$lambda.min, t
ype = "class"))
y_pred2.1se <- as.factor(predict(fit2, newx = X_valid, s = fit2$lambda.1se,ty
pe = "class"))

# Summarize
pred2 <- data.frame(Lambda = c("Min", "1SE"),
                          MeanError = c(mean(y_pred2.min != y_valid),
                                        mean(y_pred2.1se != y_valid)))
pred2
```

```
##   Lambda MeanError
## 1    Min 0.1194969
## 2    1SE 0.1226415

#
```

Part 3

The best shrinkage model (if we consider dimension reduction as shrinkage) from HW3 is the LDA model which yields a misclassification rate of 0.22 on validation set. Grouped LASSO gives minimum misclassification rate of 0.120 and 1se misclassification rate 0.129. Ungrouped LASSO has minimum misclassification rate 0.138 and 1se misclassification rate 0.160. In contrast, ridge regression yields misclassification rate of 0.119 and 1se misclassification rate 0.123, with number of coefficients equal to 25600. If I were to choose one method, I would choose ridge regression for a balance of speed and accuracy. Ridge is faster than grouped LASSO on my laptop and I think that's because of closed form solutions exist for ridge but not for grouped LASSO. Ungrouped LASSO is faster than ridge but the difference in prediction accuracy is substantial. LDA method is much worse for this application and should not be used.