CS 498 AML HW7
Report

Pengyu Cheng (pcheng11)
Haotian Qiu (hqiu6)
Caiting Wu (cwu72)

12.3

1. Show your plot of the cross-validated deviance of the model against the regularization variable. This plot should come from `cv.glmnet`:

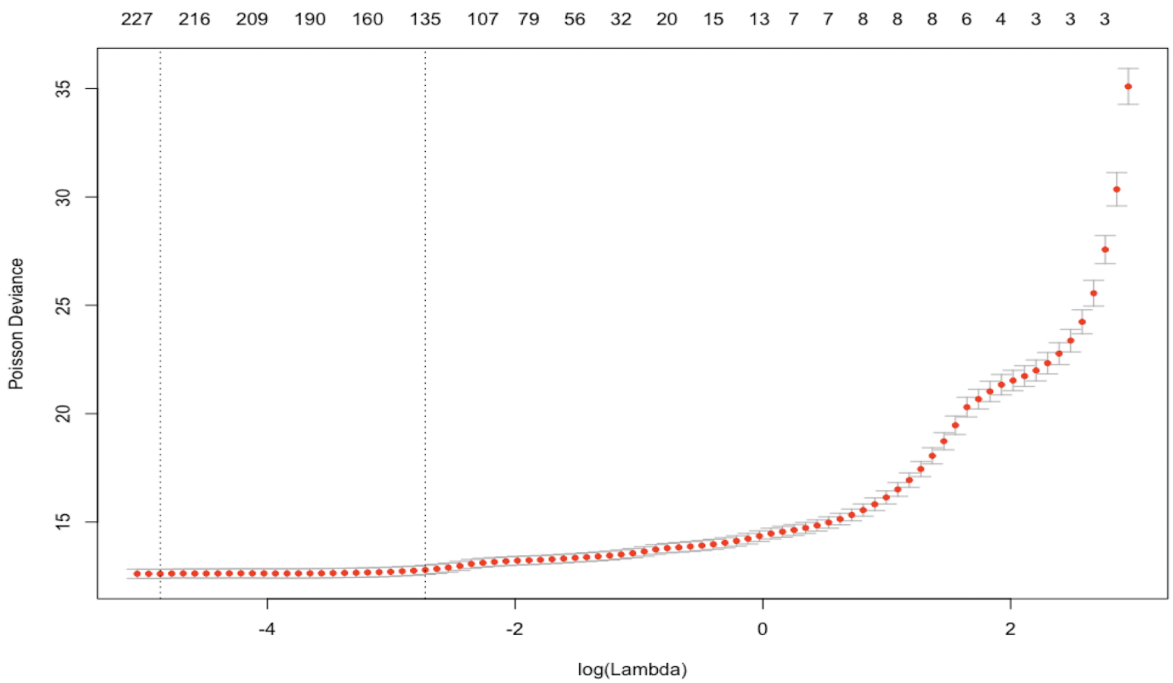


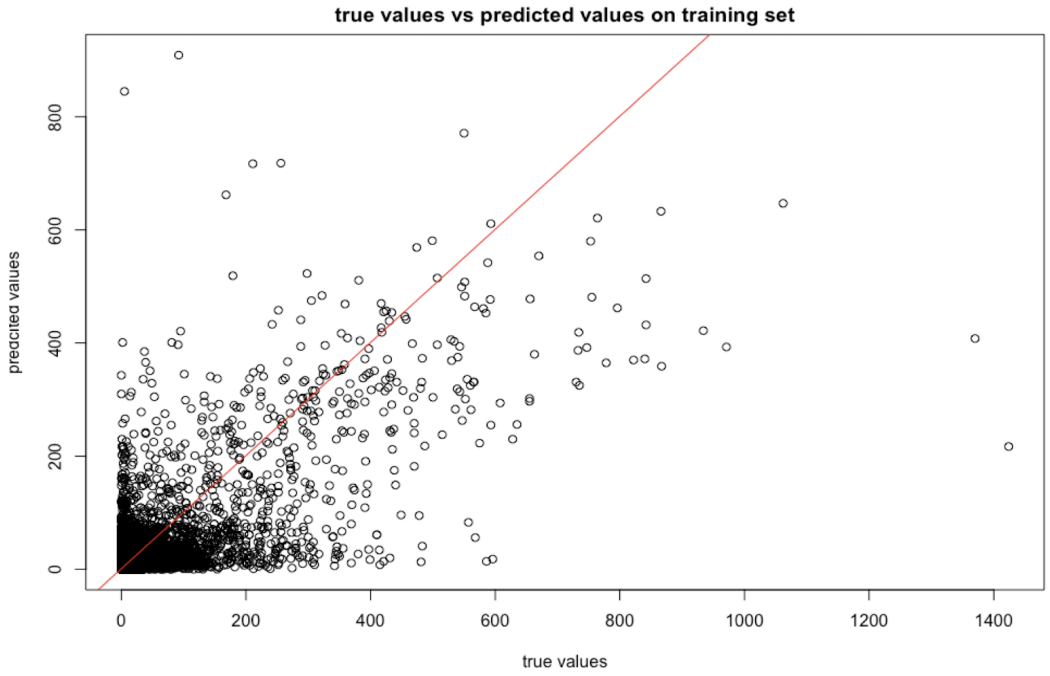
figure 1:*The plot of the cross-validated deviance of the model against the regularization variable*

CS 498 AML HW7
Report

Pengyu Cheng (pcheng11)
Haotian Qiu (hqiu6)
Caiting Wu (cwu72)

2. Indicate the value of regularization constant that you choose, and show the scatter plot of true values vs predicted values for your training data:

   We choose `lambda.min = 0.00771056`

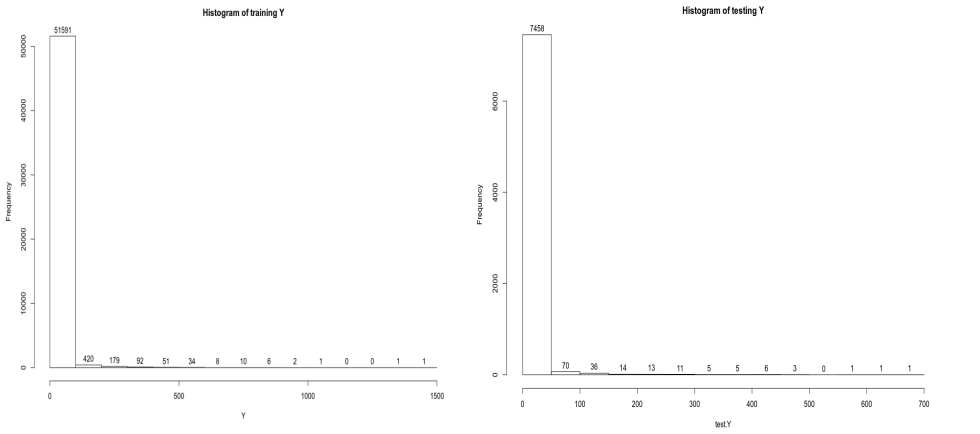


CS 498 AML HW7
Report

Pengyu Cheng (pcheng11)
Haotian Qiu (hqiu6)
Caiting Wu (cwu72)

3. Indicate the value of regularization constant that you choose, and show the scatter plot of true values vs predicted values for your testing data:
I choose `lambda.min = 0.00771056`

4. Compare the two plots and comment on the performance of the model. Provide comment on why this regression is difficult:
   - The distribution of the classes for the training data: (left)
   - The distribution of the classes for the test data: (right)



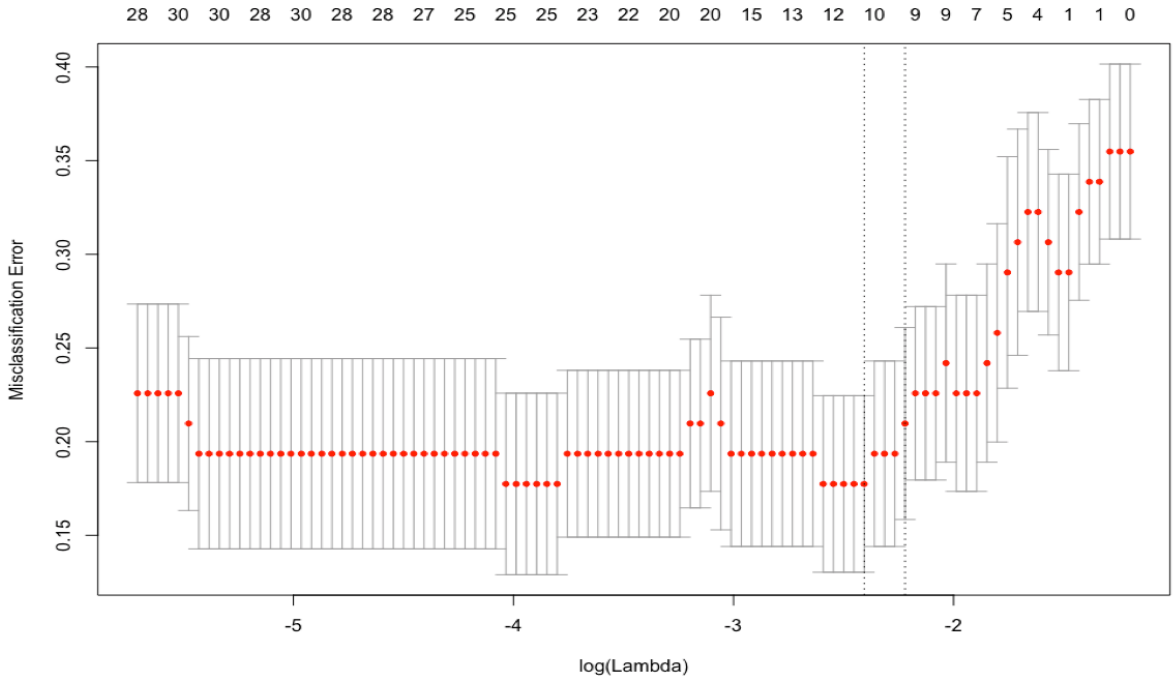
We can see the large portion of the classes which are from 0 to 100 contributing to the **imbalance** of this dataset (both train or test data set). Regressing on this sort of dataset (**high dimensional** and **imbalanced**) is always difficult because the model tends to **overfit** in favor of the majority of the dataset. From the previous two plots, we can also see the **outliers** (which should be removed) which also contributes to the difficulty of the regression.

CS 498 AML HW7
Report

Pengyu Cheng (pcheng11)
Haotian Qiu (hqiu6)
Caiting Wu (cwu72)

12.4

5. Show the plot of the classification error of the model against the regularization variable. Indicate the value of regularization constant of your choice and provide comment on the model performance compared with the baseline. Remember to include the classification accuracy in the comparison.



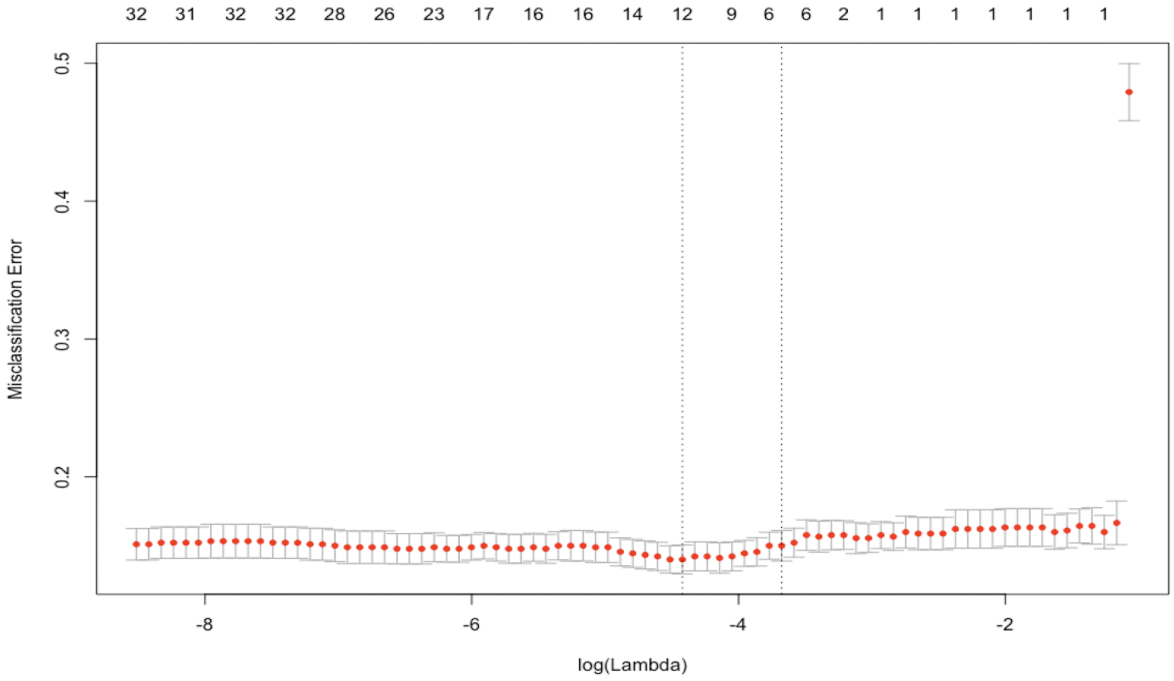
- My chosen regularization constant = 0.09016021 (`lambda.min`)
- Baseline accuracy: 0.6451613
- This model's accuracy: 0.8709677 (obtained using '**`predict`**')
- Compared with the baseline that predicts the most common class, this model's accuracy is a lot better.

CS 498 AML HW7
Report

Pengyu Cheng (pcheng11)
Haotian Qiu (hqiu6)
Caiting Wu (cwu72)

12.5

6. Predict gender with the features. Show the plot of the classification error of the model against the regularization variable. Indicate the value of regularization constant of your choice and provide comment on the model performance compared with the baseline. Remember to include the classification accuracy in the comparison:

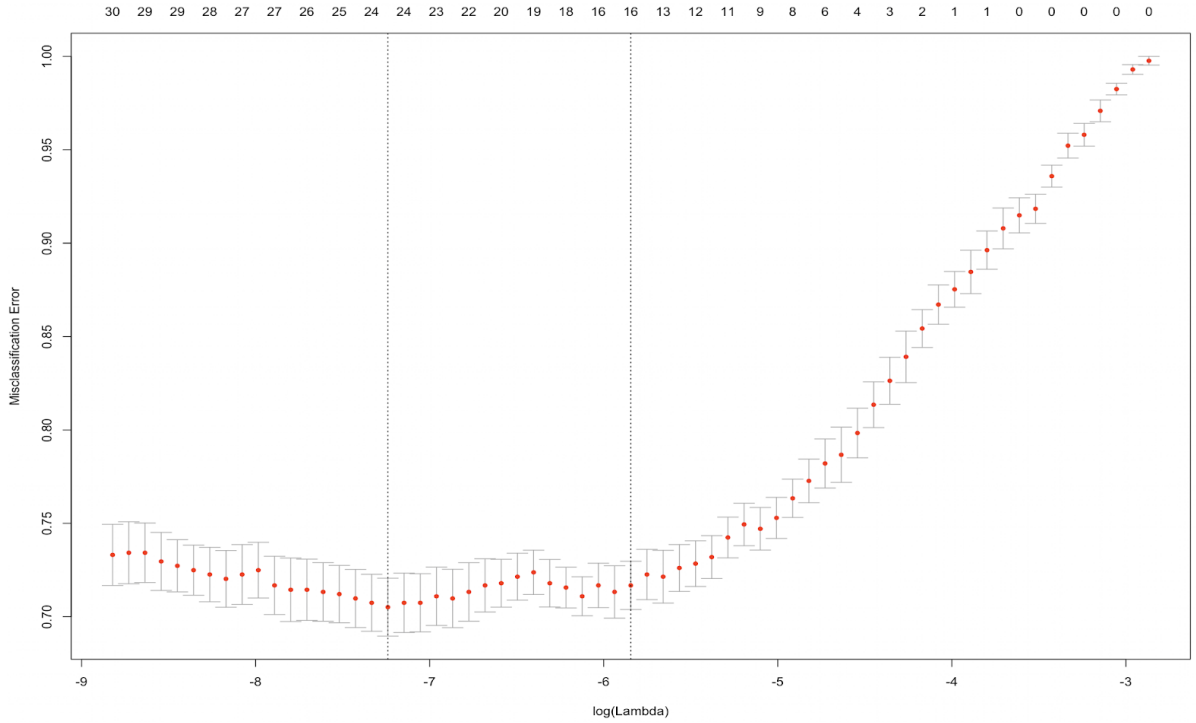


- My chosen regularization constant = 0.01202526 (`lambda.min`)
- Baseline accuracy: 0.50883 (predicting the most common gender male)
- This model's accuracy: 0.8642384 (obtained using '**`predict`**')
- Compared with the baseline that predicts the most common class, this model's accuracy is a lot better.

CS 498 AML HW7
Report

Pengyu Cheng (pcheng11)
Haotian Qiu (hqiu6)
Caiting Wu (cwu72)

7. Predict the strain of a mouse with the features. Show the plot of the classification error of the model against the regularization variable. Indicate the value of regularization constant of your choice and provide comment on the model performance compared with the baseline. Remember to include the classification accuracy in the comparison:



- My chosen regularization constant = 0.0007172174 (lambda.min)
- Baseline accuracy:
  - 0.01818182 (predicting a strain at random (**theoretical value**))
  - 0.02797203 (predicting a strain at random (**through simulation**)
  - This model's accuracy: 0.6655012 (obtained using **`'predict'`**)
- Compared with the baseline that predicts the class randomly, this model's accuracy is a lot better.

CS 498 AML HW7
Report

Pengyu Cheng (pcheng11)
Haotian Qiu (hqiu6)
Caiting Wu (cwu72)

8. 1 page Code screenshot. It should include code for using glmnet, making the plot and data preprocess:

```r
setwd("/Users/pengyucheng/Desktop/Applied-ML/")
blog.data = read.csv(file="data/cs498_aml_hw7_dataset/blogData_train.csv", header=TRUE, sep=",")
head(blog.data)
Y = as.matrix(blog.data[, 281])
X = as.matrix(blog.data[, -281])
row.nums = nrow(blog.data)
col.nums = ncol(blog.data)
```

12.3
a)

```r
library(glmnet)
require(doMC)
registerDoMC(cores=2)
cv.fit = cv.glmnet(X, Y,family= "poisson", alpha=1)
plot(cv.fit)
```

b)

```r
cv.fit$lambda.min
fitted.values = predict(cv.fit, newx=X, s="lambda.min", type = "response")
plot(Y, floor(fitted.values), xlab='true values', ylab='predcited values', main='true values vs predicted values on training set')
abline(1,1,col='red')
```

c)

```r
setwd("/Users/pengyucheng/Desktop/Applied-ML/data/cs498_aml_hw7_dataset/test_data")
file_list = list.files()
flag = 1
for (file in file_list){
    if (flag == 1) {
      blog.test.data = read.csv(file, header=F, sep=",")
      flag = 0
    }
    else {
      new_dataset = read.csv(file, header=F, sep=",")
      blog.test.data = rbind(blog.test.data, new_dataset)
      rm(new_dataset)
    }
}
nrow(blog.test.data)
test.Y = as.matrix(blog.test.data[,281])
test.X = as.matrix(blog.test.data[,-281])
fitted.values = predict(cv.fit, newx=test.X, s="lambda.min", type = "response")
plot(test.Y, floor(fitted.values), xlab='true values', ylab='predcited values', main='true values vs predicted values on test set')
abline(1,1,col='red')
```

CS 498 AML HW7
Report

Pengyu Cheng (pcheng11)
Haotian Qiu (hqiu6)
Caiting Wu (cwu72)

12.4

```{r}
set.seed(22)
setwd("/Users/pengyucheng/Desktop/Applied-ML/")
raw.data.x = read.csv(file ="data/cs498_aml_hw7_dataset/gene_data_x.txt", header = F, sep = " ")
gene.data.x = t(as.matrix(raw.data.x))
raw.data.y = read.csv(file ="data/cs498_aml_hw7_dataset/gene_data_y.txt", header = F, sep = " ")
gene.data.y = as.matrix(raw.data.y)
gene.data.y = as.matrix(apply(gene.data.y, 1, FUN=function(x){if (x > 0) {x = 0} else {x = 1}}))
cv.gene.fit = cv.glmnet(gene.data.x, gene.data.y,family= "binomial", type.measure="class")
plot(cv.gene.fit)
sum(gene.data.y)/nrow(gene.data.y)
(gene.lambda = cv.gene.fit$lambda.min)
mean(predict(cv.gene.fit, gene.data.x, s=gene.lambda, type="class") == gene.data.y)
```

12.5 a)

```{r}
setwd("/Users/pengyucheng/Desktop/Applied-ML/")
raw.data = read.csv(file = "data/cs498_aml_hw7_dataset/Crusio1.csv")
crusio.data = raw.data[,c(2, 4:41)]
crusio.data
crusio.data = na.omit(crusio.data)
crusio.y = data.matrix(crusio.data[,1])
crusio.x = data.matrix(crusio.data[,-1])
cv.crusio.fit = cv.glmnet(crusio.x, crusio.y, family= "binomial", type.measure="class", alpha=1)
plot(cv.crusio.fit)
crusio.y == "f"
which(crusio.y == "f")
length(which(crusio.y == "m"))/nrow(crusio.y)
(crusio.lambda = cv.crusio.fit$lambda.min)
mean(predict(cv.crusio.fit, crusio.x, s=crusio.lambda, type="class") == crusio.y)
```

b)

```{r}
setwd("/Users/pengyucheng/Desktop/Applied-ML/")
raw.data = read.csv(file = "data/cs498_aml_hw7_dataset/Crusio1.csv")
crusio.data.s = raw.data[,c(1, 4:41)]
#omit nas
crusio.data.s = na.omit(crusio.data.s)
#drop strains less than 10 rows
for (s in unique(crusio.data.s$strain)) {
  if (nrow(crusio.data.s[which(crusio.data.s$strain == s), ]) < 10) {
    crusio.data.s = crusio.data.s[!crusio.data.s$strain == s, ]
}
crusio.s.y = data.matrix(crusio.data.s[,1])
crusio.s.x = data.matrix(crusio.data.s[,-1])
cv.crusio.s.fit = cv.glmnet(crusio.s.x, crusio.s.y, family= "multinomial", type.measure="class", alpha=1)
plot(cv.crusio.s.fit)
(crusio.s.lambda = cv.crusio.s.fit$lambda.min)
mean(predict(cv.crusio.s.fit, crusio.s.x, s=crusio.s.lambda, type="class") == crusio.s.y, alpha=1)
#basline theoretical value:
1/length(levels(crusio.data.s$strain))
#simulation value:
set.seed(22)
sum(sample(unique(crusio.s.y), length(crusio.s.y), replace=TRUE)==crusio.s.y)/nrow(crusio.data.s)
```