STAT W4249: Assignment #3

Due on Wednesday, March 12, 2014

Regina 1:10pm

Fan Ye(fy2188)
Chenyun Zhao(cz2321)
Xiaoran Wang(xw2313)
Xiaoyao Yang(xy2231)
Wenjia Cao(wc2477)

Contents

1 Preprocessing

After downloading the dataset, we found that the dataset was totally a mess. We first use "adult.names" file to import column names of our dataset, then create a 0-1 variable that represents people with less than 50K income and larger than 50K income, respectively. We also set all missing value equal to "NA". After further looking into our dataset, we found that there were altogether 15 predictors and 2399 observations with missing values. Since the total dataset contains more than 30000 observations, we decided to exclude observations with missing values and the distributon of dataset remains the same. As for categorical variables, we change them into Dummy variable in order to do numeric analysis. As a result, we have dataset containing 89 variables and 30162 observations

To further simplify dataset, PCA was used to reduce dimention. As the result shown below, there are 71 components extacted 95% information. Here, we would like to compare results got from each model with original data and with data modified by PCA. For our convenience, we called model using PCA a "reduced model", while model using original data named as "Full model".

```
dat <- read.table("adult.data.txt", sep = ",", header = F, stringsAsFactors = FALSE)
# pre cleaning
des <- read.table("adult.des.txt", header = F, sep = ":")
des <- des$V1
des <- as.character(des)
des <- c(des, "y")
des[c(5, 6, 11, 12, 13, 14)] <- c("education_num", "marital_status", "capital_gain",
    "capital_loss", "hours_per_week", "native_country")
names(dat) <- des
dat$y[dat$y == " <=50K"] <- 0
dat$y[dat$y == " >50K"] <- 1
dat$y <- as.numeric(dat$y)
# delete sample containing ?
dat[dat == " ?"] <- NA
dat <- na.omit(dat)
# creating dummy variables
dummy_workclass <- model.matrix(~dat$workclass - 1)
dummy_education <- model.matrix(~dat$education - 1)
dummy_marital <- model.matrix(~dat$marital - 1)
dummy_occupation <- model.matrix(~dat$occupation - 1)
dummy_relationship <- model.matrix(~dat$relationship - 1)
dummy_race <- model.matrix(~dat$race - 1)
dummy_sex <- model.matrix(~dat$sex - 1)
dummy_native <- model.matrix(~dat$native_country - 1)

CreateDummy <- function(dat_var) {
    n <- length(dat_var)
    site <- as.factor(dat_var)
    site.t <- table(site)
    temp <- sort(site.t, decreasing = TRUE)
    site.cdf <- cumsum(temp)/sum(temp)
    site.names <- names(temp)[site.cdf < 0.99]
    # order number
    for (i in 1:n) {
        if (!sum(dat_var[i] == site.names)) {
            dat_var[i] <- "Others"
        }
    }
```

```
    dat <- dat_var
    dummy_var <- model.matrix(~dat - 1)
    return(dummy_var)
}
dummy_native <- CreateDummy(dat$native)
# build new data frame
datn <- data.frame(y = dat$y, dat$age, dummy_workclass, dat$fnlwgt, dummy_education,
    dat$education_num, dummy_marital, dummy_occupation, dummy_relationship,
    dummy_race, dummy_sex, dat$capital_gain, dat$capital_loss, dat$hours_per_week,
    dummy_native)

# seperate training and test
m <- dim(datn)[1]
n <- dim(datn)[2]
dat_train <- datn[1:(0.5 * m), ]
dat_test <- datn[(0.5 * m + 1):m, ]

# Creating PCA data
pca <- prcomp(x = datn[, -1], scale. = TRUE, center = TRUE)
summary(pca)$importance[, 68:70]

##                           PC68   PC69    PC70
## Standard deviation     0.89665 0.8951 0.85586
## Proportion of Variance 0.00914 0.0091 0.00832
## Cumulative Proportion  0.93284 0.9419 0.95027

dat_pca <- data.frame(y = as.factor(datn$y), pca$x[, 1:70])
dat_train_pca <- dat_pca[1:(0.5 * m), ]
dat_test_pca <- dat_pca[(0.5 * m + 1):m, ]
```

2 Predictive Modeling

2.1 Ridge and Lasso

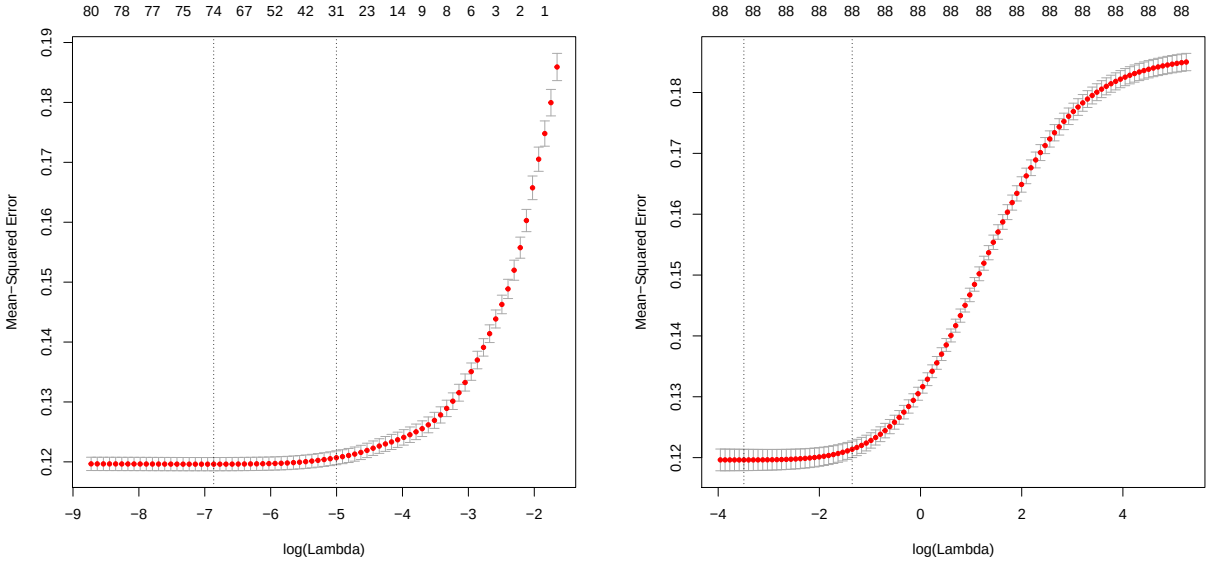
```
require(glmnet)

## Loading required package:  glmnet
## Loading required package:  Matrix
## Loaded glmnet 1.9-5

# Lasso
fit_lasso <- cv.glmnet(x = as.matrix(dat_train[, -1]), y = as.numeric(dat_train[,
    1]), alpha = 1, family = "gaussian")
plot(fit_lasso)
pred <- predict(fit_lasso, as.matrix(dat_test[, -1]))
pred[pred > 0.5] <- 1
pred[pred <= 0.5] <- 0
TestError <- sum(pred != dat_test[, 1])/length(pred)
cat(TestError)

## 0.1682

table(pred, dat_test$y)
```



(a) Lasso (b) Ridge

Figure 1: Model selection for Lasso and Ridge

```
##
## pred     0     1
##    0 10762  1997
##    1   539  1783

# Ridge
fit_ridge <- cv.glmnet(x = as.matrix(dat_train[, -1]), y = as.numeric(dat_train[,
    1]), alpha = 0, type.measure = "mse", nfolds = 5)
plot(fit_ridge)
pred <- predict(fit_ridge, as.matrix(dat_test[, -1]), s = "lambda.min")
pred[pred > 0.5] <- 1
pred[pred <= 0.5] <- 0
TestError <- sum(pred != dat_test[, 1])/length(pred)
cat(TestError)

## 0.1654

table(pred, dat_test$y)

##
## pred     0     1
##    0 10699  1892
##    1   602  1888
```

2.2 Logistic Regression

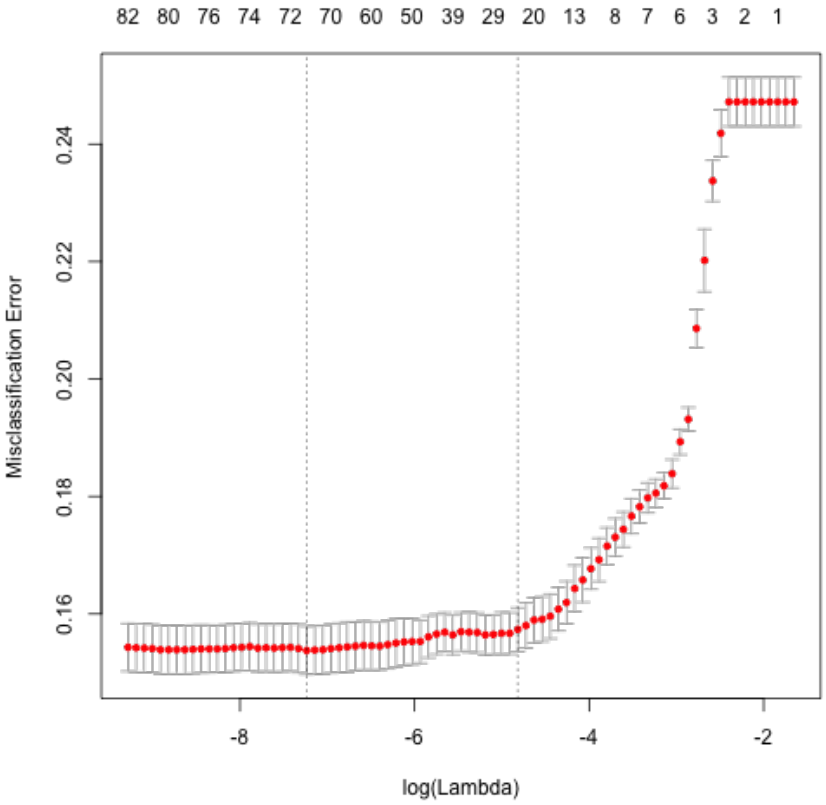


Figure 2: Best parameter of Logistic Model

```
# logistic
require(glmnet)

cvfit = cv.glmnet(x = as.matrix(dat_train[, -1]), y = dat_train[, 1], family = "binomial",
    type.measure = "class")
plot(cvfit)
```

```
pred <- predict(cvfit, as.matrix(dat_test[, -1]), type = "class", s = "lambda.min")
TestError <- sum(pred != dat_test[, 1])/length(pred)
cat("Test error of Logistic Model is", TestError)

## Test error of Logistic Model is 0.1514
```

2

2.3 LDA and QDA

```
# LDA
require(MASS)

## Loading required package:  MASS
```

```
fit_lda <- lda(y ~ ., data = dat_train_pca)
pred <- (predict(fit_lda, dat_test_pca[, -1]))
pred <- pred$class
TestError.lda <- sum(pred != dat_test_pca[, 1])/length(pred)
cat(TestError.lda)

## 0.1621

# QDA
fit_lda <- qda(y ~ ., data = dat_train_pca, cv = T)
pred2 <- (predict(fit_lda, dat_test_pca[, -1]))
pred2 <- pred2$class
TestError <- sum(pred2 != dat_test_pca[, 1])/length(pred2)
cat(TestError)

## 0.3751
```

2.4 KNN

```
require(class)

## Loading required package:  class

fit2 <- knn(train = dat_train[, -1], test = dat_test[, -1], k = 5, cl = dat_train[,
    1])
TestError <- sum(fit2 != dat_test[, 1])/length(dat_test[, 1])
cat(TestError)

## 0.2369
```

3 Summary

Table 1: Different Model Results

Model	PCA test error rate	without PCA test error rate
LOGISTIC	0.1512	0.1516
KNN	0.1821	0.2366
LDA	0.1621	*
QDA	0.3751	*
LASSO	0.7494	0.1692
RIDGE	0.7494	0.1651

The table above gives us a brief comparison between differents models. From the results of KNN method, we can find that in this case, KNN does not perform very well, especially when we use the full model, which has 23.66% test error. We think the reason is about the curse of dimensionality, the volume of the space increases so fast that the available data become sparse. Maybe it is also the reason why after we applied PCA method to extract the features, the test error become a little better, but still not as good as other methods like logistic regression.

The table above gives us a brief comparison between different models. From the results of KNN method, we can find that in this case, KNN does not perform very well, especially when we use the full model, which has 23.66% test error. We think the reason is about the curse of dimensionality, the volume of the space increases so fast that the available data become sparse. Maybe it is also the reason why after we applied PCA method to extract the features, the test error became a little better, but still not as good as other methods like logistic regression.

Ridge and Lasso are not good in sense of test error for classification problem. We classified data by comparing the result to 0.5, if predict value is greater than 0.5, we consider its income is greater than 50K, otherwise is less than 50K. Compared this result to actual income, we can calculate test error(test error= wrongly predicted observations / total observations). These two regression methods both focus on coefficients shrinkage. Ridge regression does a proportional shrinkage and Lasso translates each coefficient by a constant factor, truncating at zero. So Lasso has a function of selecting variables.

The test error from ridge regression is 74.94%, which is the same as the test error from lasso regression. But both test errors are much bigger than those from other classification methods, also are much bigger than those obtained from full model. That is because, when using PCA method before building the model, predictors are uncorrelated, so variable-selection of Lasso regression is impaired, so we get nearly the same results using Lasso regression and Ridge regression. As for the larger test errors, we guess that it is because after using PCA, we already loose some information from the original data, using shrinkage methods like Ridge regression and Lasso regression, we further loose more information, which cause an increase in test error rate.

We found that we couldnt train data in full model using LDA and QDA algorithms; this may because of the collinearity of the predictor variables which can greatly increase the model variance. We then successfully trained LDA and QDA in the reduced model, which lead error rate 16.21% and 37.51% respectively. Comparing to LDA, there is no assumption that the covariance of each of the classes is identical in QDA. This may be the reason that LDA has a better performance in this case. Another guess is that, probably with such high dimension data, linear classifier is prefered.

Logistic model provides better result over all the other models, corresponding to its theoretical advantage in classification. The reduced model has a better test error(15.12%) than full model(15.16%), which shows that PCA successfully extracted most of the important information our data contains and provides reliable perdition results. In addition to that, by applying PCA, reduced model contains 71 variables instead of 89 variables in our full model. So the reduced model is also a wise choice in sense of model efficiency.