PSET5_ParthDesai

Parth Desai

2023-03-02

Question 1

Part 1.1

```
load("CreditClaim.RData")
x <- credit_claim$x
y <- credit_claim$y

(length(subset(y, y==1))/length(y)) * 100
```

```
## [1] 25.84693
```

25.846% of documents claim credit

Part 1.2

There are 170,000 observations compared to the 6,046,839 predictors that are there. This would lead to overfitting as the number of predictors is vastly more than the observation count.

Part 1.3

```
top_20 <- as.matrix(colMeans(x))
top_20_ord <- top_20[order(-colMeans(x)),]
final_top_20 <- head(top_20_ord, 20)
final_top_20
```

```
##    congress     million      energy     funding legislation     release
##   0.8845671   0.8506901   0.8469260   0.8431619   0.8331242   0.8168130
##    american      byline    dateline    national         fed     contact
##   0.8005019   0.7452949   0.7452949   0.7365119   0.7151819   0.6675031
##     support        care         tax        help  government    security
##   0.6574655   0.6562108   0.6511920   0.6461731   0.5922208   0.5734003
##      people   president
##   0.5646173   0.5520703
```

The top 20 words all range being mention approximately 9/10ths to 3/5ths of the time in a document. Many of the words have to do with bureacracy or members of the bureacracy.

Part 1.4

```
log_model <- glm(y ~ x[,'congress'] + x[,'million'] + x[,'energy'] + x[,'funding'] + x[,'legislation'] +
summary(log_model)
```

```
##
## Call:
## glm(formula = y ~ x[, "congress"] + x[, "million"] + x[, "energy"] +
##     x[, "funding"] + x[, "legislation"] + x[, "release"] + x[,
##     "american"] + x[, "byline"] + x[, "dateline"] + x[, "national"] +
##     x[, "fed"] + x[, "contact"] + x[, "support"] + x[, "care"] +
##     x[, "tax"] + x[, "help"] + x[, "government"] + x[, "security"] +
##     x[, "people"] + x[, "president"], family = "binomial", data = credit_claim)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.2905  -0.6821  -0.4138   0.0844   3.0484
##
## Coefficients: (1 not defined because of singularities)
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -1.631975   0.344662  -4.735 2.19e-06 ***
## x[, "congress"]     -0.289763   0.100112  -2.894 0.003799 **
## x[, "million"]       0.318479   0.078150   4.075 4.60e-05 ***
## x[, "energy"]       -0.009716   0.041007  -0.237 0.812714
## x[, "funding"]       0.488295   0.071449   6.834 8.25e-12 ***
## x[, "legislation"] -0.158665   0.082511  -1.923 0.054484 .
## x[, "release"]       0.624101   0.214930   2.904 0.003687 **
## x[, "american"]     -0.091302   0.075090  -1.216 0.224024
## x[, "byline"]        0.753817   0.511096   1.475 0.140239
## x[, "dateline"]            NA         NA      NA       NA
## x[, "national"]      0.067053   0.069018   0.972 0.331279
## x[, "fed"]          -0.884098   0.471493  -1.875 0.060778 .
## x[, "contact"]       0.096060   0.188849   0.509 0.610990
## x[, "support"]      -0.008011   0.094969  -0.084 0.932776
## x[, "care"]         -0.208263   0.078769  -2.644 0.008194 **
## x[, "tax"]           0.117689   0.051452   2.287 0.022176 *
## x[, "help"]          0.453478   0.093385   4.856 1.20e-06 ***
## x[, "government"]   -0.391357   0.124727  -3.138 0.001703 **
## x[, "security"]     -0.095655   0.074807  -1.279 0.201005
## x[, "people"]       -0.297239   0.134328  -2.213 0.026912 *
## x[, "president"]    -0.548358   0.160677  -3.413 0.000643 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 910.89  on 796  degrees of freedom
## Residual deviance: 657.04  on 777  degrees of freedom
## AIC: 697.04
```

```
##
## Number of Fisher Scoring iterations: 6
```

Bonus

The word 'dateline' is dropped to NA because it is a singularity.

Part 1.5

```
predictions <- predict(log_model, type = 'response')
prediction_classes <- ifelse(predictions > 0.5, 1, 0)
head(prediction_classes, 10)
```

```
##  1  2  3  4  5  6  7  8  9 10
##  0  0  0  0  0  0  0  1  0  0
```

Part 1.6

```
error_rate <- mean(prediction_classes != y)
print(error_rate)
```

```
## [1] 0.1794228
```

Note

```
one_matrix <- as.data.frame(cbind(y, final_top_20))
```

Part 1.7

```
loocv_predictions <- nrow(x)

for (i in 1:nrow(one_matrix)) {
  gen_model <- glm(y ~ x[,'congress'] + x[,'million'] + x[,'energy'] + x[,'funding'] + x[,'legislation']

  loocv_predictions[i] <- ifelse(predict(gen_model, as.data.frame(one_matrix[i, ]), type = 'response') >
}
```

Part 1.8

```
loocv_error_rate <- mean(loocv_predictions != credit_claim$y)
print(loocv_error_rate)
```

```
## [1] 0.2584693
```

Part 1.9

The out-of-sample error is greater than the in-sample error. #Question 2

```
library(glmnet)
library (ggplot2)
```

Part 2.1

```
load("CreditClaim.RData")
x <- credit_claim$x
y <- credit_claim$y
n.total <- length(y)
prop.train <- 0.7
set.seed(54321)
r <- sample(1:n.total,round(prop.train*n.total), replace = FALSE)
x.train <- x[r,]
x.test <- x[-r,]
y.train <- y[r]
y.test <- y[-r]
```

Part 2.2

```
set.seed(123)
cv.results <- cv.glmnet(x = x.train, y = y.train,
family = "binomial", nfolds = 5, alpha = 1)
```

The first line sets the seed for the random number generator, ensuring that results are reproducible. The second line uses the 'cv.glmnet' function, which will perform a cross-validation on the model, and saves it as cv.results. The arguments taken for 'cv.glmnet' are: the predictor data, in this case 'x.train', the response variable, in this case 'y.train', the family, in this it is binomial which will return either a 1 or 0, the nfolds, which is the number of folds in the cross-validation process (5), and alpha, which is the parameter for the penalties in the LASSO model and is set to 1 for a complete use of the L1 penalty and none of the L2 penalty.

Part 2.3

```
summary(cv.results)
```

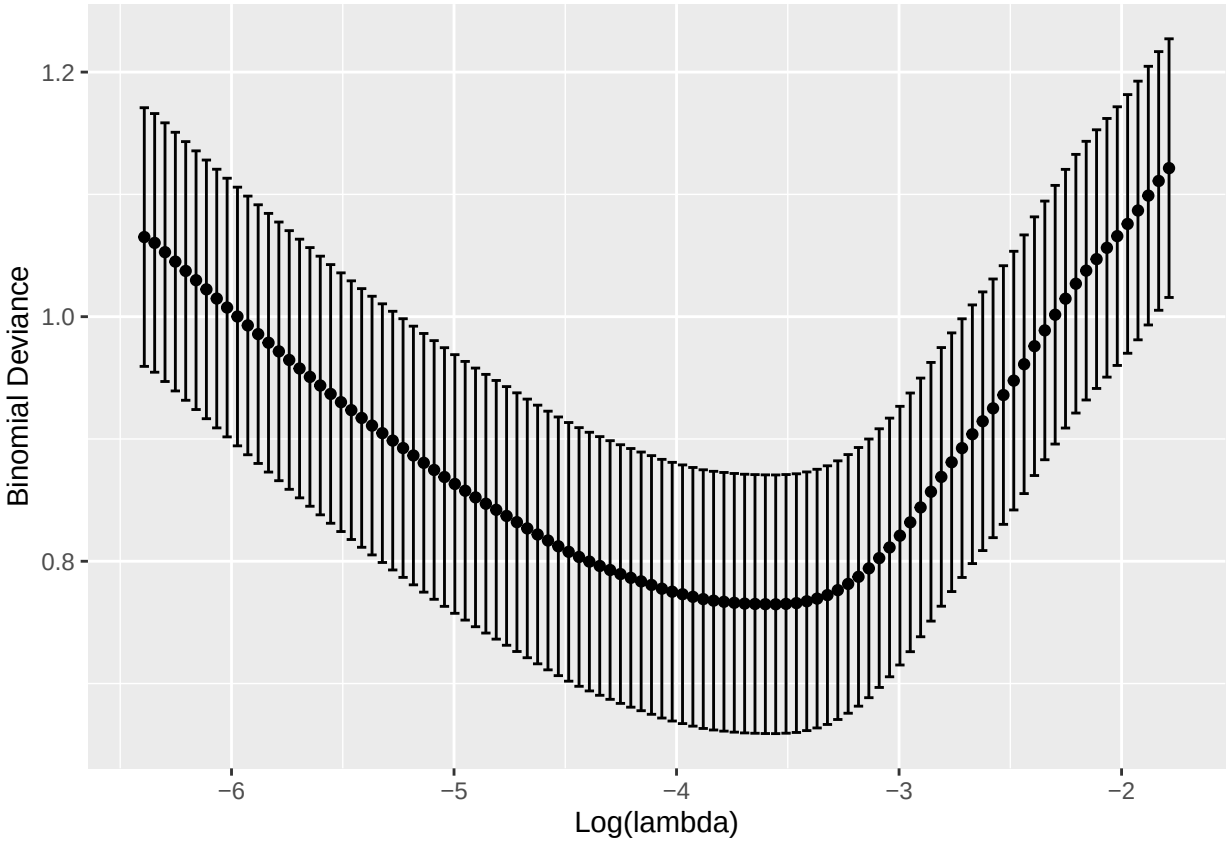
```
##            Length Class  Mode
## lambda     100    -none- numeric
## cvm        100    -none- numeric
## cvsd       100    -none- numeric
## cvup       100    -none- numeric
## cvlo       100    -none- numeric
## nzero      100    -none- numeric
```

```
## call         6    -none- call
## name         1    -none- character
## glmnet.fit  13    lognet list
## lambda.min   1    -none- numeric
## lambda.1se   1    -none- numeric
## index        2    -none- numeric
```

There are 100 lambdas tested. Given 5 folds and 100 lambda values, a total of 500 LASSO models were fitted. The binomial deviance loss function is being used to compute CV error.

Part 2.4

```
a <- ggplot(data = as.data.frame(cv.results$lambda), aes(x=log(cv.results$lambda), y=cv.results$cvm, la
a + geom_point() + geom_errorbar(aes(ymin = cv.results$cvm-sd(cv.results$cvm), ymax=cv.results$cvm+sd(c
```



Part 2.5

Given the above graphic, CV error is minimized when lambda is between the log values of -3.5 to -3.1.

Part 2.6

```
optimal_lam <- cv.results$lambda.min
optimal_lam
```

```
## [1] 0.02859915
```

```
las_model <- glmnet(x = x.train, y=y.train, family="binomial", alpha = 1, lambda = optimal_lam)
lasso.coef <- coef(las_model)
print(las_model)
```

```
##
## Call:  glmnet(x = x.train, y = y.train, family = "binomial", alpha = 1,      lambda = optimal_lam)
##
##   Df  %Dev Lambda
## 1 88 58.89 0.0286
```

```
num_nonzero <- sum(lasso.coef[] != 0)
num_nonzero
```

```
## [1] 89
```

There are 88 coefficients not shrunk to 0, and 1 intercept. It can be alternatively be said that there are 88 degrees of freedom.

Bonus

```
lasso.coef_indices <- which(lasso.coef[-1] != 0)
lasso.coef_names <- colnames(x.train)[lasso.coef_indices]
lasso.coef_names
```

```
##  [1] "aeronautics"     "agricultures"    "alternatives"    "announce"
##  [5] "announced"       "announces"       "appropriations"  "arsenal"
##  [9] "base"            "brac"            "buster"          "byers"
## [13] "cazayoux"        "cents"           "chamber"         "childs"
## [17] "common"          "communitys"      "congress"        "contributions"
## [21] "doe"             "dot"             "ears"            "endorsed"
## [25] "equipment"       "exhaust"         "expand"          "faa"
## [29] "facility"        "fairpoint"       "firefighter"     "fostering"
## [33] "funding"         "gillies"         "grant"           "harder"
## [37] "homeport"        "linda"           "loophole"        "manufacture"
## [41] "maryland"        "milk"            "nation"          "neighborhoods"
## [45] "nfip"            "park"            "patty"           "political"
## [49] "pollution"       "portland"        "postcard"        "preserve"
## [53] "programming"     "project"         "prosecutors"     "questions"
## [57] "rain"            "reaffirms"       "reauthorization" "reductions"
## [61] "regarding"       "regional"        "reinvestment"    "rent"
## [65] "residential"     "river"           "rmt"             "rutgers"
```

```
## [69] "schultz"         "secured"         "sediment"        "shoring"
## [73] "simultaneously"  "smart"           "spare"           "stand"
## [77] "station"         "stimulating"     "streets"         "tag"
## [81] "tested"          "treasure"        "tricare"         "understand"
## [85] "urban"           "venture"         "watersheds"      "weatherization"
```

Part 2.7

```
lasso.probs <- predict(las_model, newx = x.test, type = "response")
lasso.pred <- ifelse(lasso.probs > 0.5, 1, 0)
test.error <- mean(lasso.pred != y.test)
test.error
```

```
## [1] 0.2175732
```

Part 2.8

```
B <- 200
pred.probs <- c()
for (i in 1:B) {
  boot.sample <- sample(nrow(x.train), replace = TRUE)
  x.boot <- x.train[boot.sample, ]
  y.boot <- y.train[boot.sample]
  boot.fit <- glmnet(x = x.boot, y = y.boot, alpha = 1, lambda = cv.results$lambda.1se)
  x.test.first <- as.matrix(x.test[i, ])
  pred.probs[i] <- predict(boot.fit, newx = x.test.first[,1], type = "response")
}


conf.int <- quantile(pred.probs, c(0.025, 0.975))

conf.int
```

```
##       2.5%      97.5%
## 0.08905583 0.76466193
```