Project 3

Paige McKenzie, Brett Scroggins, Jiahao Ye, Yanbin Kang

March 20, 2018

Project 3

Load data and packages

```
library("slam")
library("gurobi")
load("data.Rdata")
```

Question 2

MIQP with Gurobi

```
p = ncol(X)
k = 8
M = .25
sol = list()
sol$x = c(M)

while (M %in% sol$x) {
  M = M*2
  Q = matrix(0, ncol=2*p, nrow=2*p)
  Q[1:p,1:p] = t(X) %*% X
  c = -2 * c(t(y) %*% X, rep(0,p))

  A = matrix(0, nrow=2*p+1, ncol=2*p)
  #all zi sum to k
  A[1,] = c(rep(0,p), rep(1,p))

  #all -Mzi<=Bi
  A[2:(p+1),1:p] = diag(-1,p)
  A[2:(p+1),(p+1):ncol(A)] = diag(-M,p)

  A[(p+2):(2*p+1),1:p] = diag(-1,p)
  A[(p+2):(2*p+1),(p+1):ncol(A)] = diag(M,p)

  dir = c(rep("<=", (p+1)), rep(">=", p))
  b = c(k, rep(0,2*p))

  # define model
  model = list()
  model$A = A
  model$obj = c
  model$sense = dir
  model$rhs = b
  model$vtype = c(rep("C", p), rep("B", p))
```

```
  model$Q = Q

  sol = gurobi(model)
}

pred_gurobi = X %*% sol$x[1:p]
```

```
cat("Gurobi prediction RMSE:", sqrt(mean((y-pred_gurobi)^2)))
```

```
## Gurobi prediction RMSE: 1.107163
```

Question 2

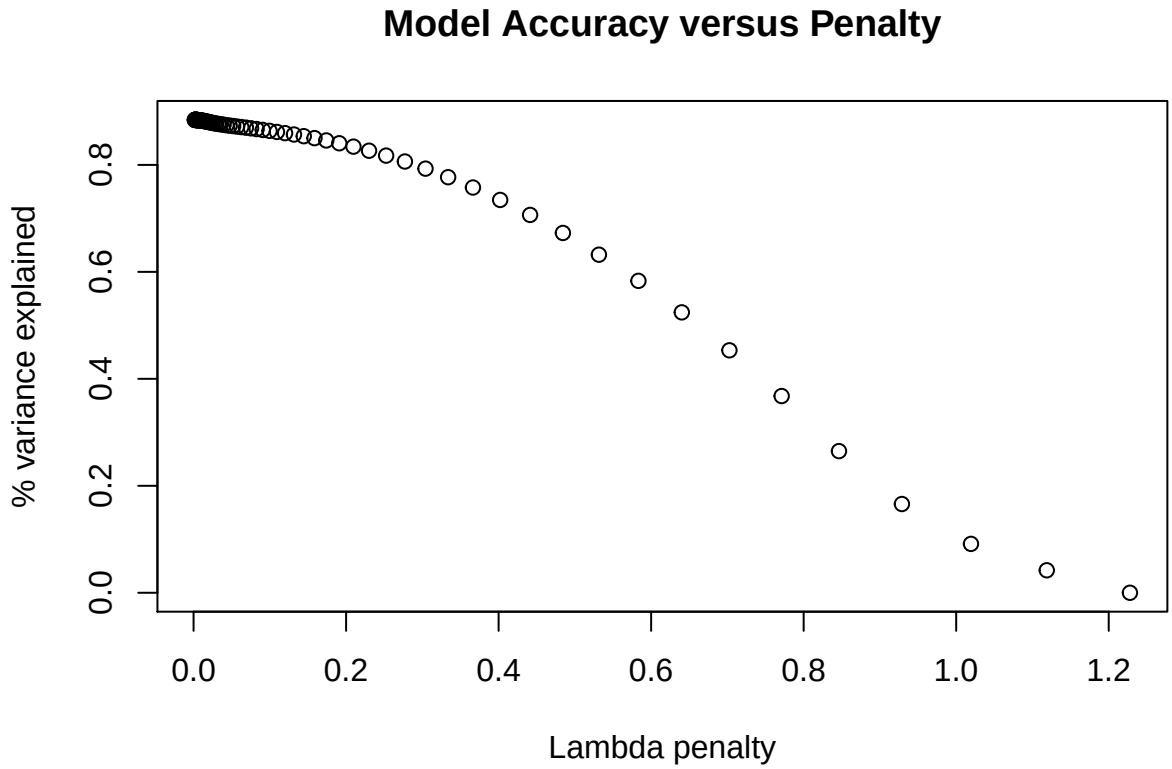
Lasso with Glmnet

```
library("glmnet")
```

```
## Loading required package: Matrix
## Loading required package: foreach
## Loaded glmnet 2.0-10
```

```
glmmod = glmnet(X, y=y, alpha=1)

plot(glmmod$lambda, glmmod$dev.ratio, ylab="% variance explained",
     xlab="Lambda penalty", main="Model Accuracy versus Penalty")
```



We can see that the model accuracy, measured by % variance explained, decreases as lambda values increase. For that reason, we should select a model with a smaller lambda, since it will be more accurate.

```
glmmod
```

```
##
## Call:  glmnet(x = X, y = y, alpha = 1)
##
##       Df    %Dev   Lambda
##  [1,]  0 0.00000 1.228000
##  [2,]  2 0.04203 1.119000
##  [3,]  3 0.09139 1.019000
##  [4,]  6 0.16590 0.928800
##  [5,]  8 0.26480 0.846300
##  [6,]  8 0.36780 0.771100
##  [7,]  8 0.45330 0.702600
##  [8,]  8 0.52430 0.640200
##  [9,]  8 0.58320 0.583300
## [10,]  8 0.63210 0.531500
## [11,]  8 0.67280 0.484300
## [12,]  8 0.70650 0.441300
## [13,]  8 0.73450 0.402100
## [14,]  8 0.75770 0.366300
## [15,]  8 0.77700 0.333800
## [16,]  8 0.79300 0.304100
## [17,]  8 0.80630 0.277100
## [18,]  8 0.81740 0.252500
## [19,]  8 0.82650 0.230100
## [20,]  8 0.83420 0.209600
## [21,]  8 0.84050 0.191000
## [22,]  8 0.84570 0.174000
## [23,]  8 0.85010 0.158600
## [24,]  9 0.85370 0.144500
## [25,]  9 0.85680 0.131700
## [26,]  9 0.85940 0.120000
## [27,] 11 0.86170 0.109300
## [28,] 13 0.86370 0.099600
## [29,] 15 0.86540 0.090750
## [30,] 16 0.86700 0.082690
## [31,] 17 0.86840 0.075340
## [32,] 19 0.86970 0.068650
## [33,] 21 0.87080 0.062550
## [34,] 21 0.87170 0.056990
## [35,] 25 0.87260 0.051930
## [36,] 25 0.87350 0.047320
## [37,] 27 0.87430 0.043110
## [38,] 29 0.87510 0.039280
## [39,] 30 0.87590 0.035790
## [40,] 32 0.87650 0.032610
## [41,] 33 0.87730 0.029720
## [42,] 36 0.87790 0.027080
## [43,] 37 0.87860 0.024670
## [44,] 42 0.87930 0.022480
## [45,] 43 0.88000 0.020480
## [46,] 46 0.88050 0.018660
```

```
## [47,] 49 0.88110 0.017000
## [48,] 50 0.88150 0.015490
## [49,] 50 0.88190 0.014120
## [50,] 51 0.88230 0.012860
## [51,] 53 0.88250 0.011720
## [52,] 53 0.88280 0.010680
## [53,] 53 0.88300 0.009731
## [54,] 55 0.88320 0.008866
## [55,] 56 0.88340 0.008078
## [56,] 57 0.88350 0.007361
## [57,] 57 0.88360 0.006707
## [58,] 58 0.88370 0.006111
## [59,] 58 0.88380 0.005568
## [60,] 59 0.88390 0.005074
## [61,] 60 0.88390 0.004623
## [62,] 61 0.88400 0.004212
## [63,] 61 0.88400 0.003838
## [64,] 62 0.88410 0.003497
## [65,] 62 0.88410 0.003186
## [66,] 62 0.88410 0.002903
## [67,] 62 0.88410 0.002645
## [68,] 63 0.88420 0.002410
## [69,] 63 0.88420 0.002196
## [70,] 63 0.88420 0.002001
## [71,] 63 0.88420 0.001823
## [72,] 63 0.88420 0.001661
```

Here, we can see each of the 72 different models have a different number of non-zero coefficients (first column), produced by different lambda values.

```
cat("Selected model: ", which.min(glmmod$lambda[glmmod$df<=8]))
```

```
## Selected model:  23
```

We selected the 23rd model, as it has 8 non-zero coefficients and has the smallest lambda value of all models with 8 non-zero coefficients (matching our k from MIQP).

```
pred_ridge = predict(glmmod, X)[,23]
cat("Lasso prediction RMSE:", sqrt(mean((y-pred_ridge)^2)))
```

```
## Lasso prediction RMSE: 1.195113
```

It also produces the smallest RMSE of all models with 8 non-zero coefficients.

Question 3

Error

```
norm_vec_sq <- function(x) {return(sum(x^2))}

error_MIQP = norm_vec_sq(X %*% sol$x[1:p] - X %*% beta_real) / norm_vec_sq(X %*% beta_real)
cat("Error from MIQP: ", error_MIQP)
```

```
## Error from MIQP:  0.004456055
```

```
error_lasso = norm_vec_sq(X %*% glmmod$beta[,23] - X %*% beta_real) / norm_vec_sq(X %*% beta_real)
cat("Error from Lasso: ", error_lasso)
```

```
## Error from Lasso:  0.02446623
```

MIQP has a smaller prediction error than the Lasso model we selected.