STAT 380 – Feature Selection Part 1 (Lecture 14)

RECALL: In the last lecture, we were using the Titanic dataset. As we modified the variables included in our model, the model performance (measured by the AUC in that problem) improved by a great deal.

IDEA: We want to explore various methods for determining which variables/features to include in a model.

RECALL: In multiple regression, we are considering a model with k explanatory variables (or inputs or features): $y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik} + \epsilon_i$. We must determine which variables to include in the model.

NOTE: We can run into problems if we:

- Overfit: Include too many regressors in the model
    - some of the regressors explain only a negligible portion of variability
    - model generalizes to new data poorly
- Underfit: Include too few regressors in the model
    - explain only a portion of the variation in the response that could be explained

Philosophies of Modeling

1. Based on the researcher's knowledge, natural laws, or historical data, we may only consider a single model.

2. Without the knowledge as to which predictors should be in the model, we may consider many models and try to find the "best model" or "best models."

GOAL: Consider all possible regressors (original regressors and functions of them) and determine which subset of them provides the "best" model(s). The following are some criteria, not an exhaustive list, for establishing the "best":

- Large R^2 or R^2_{adj} (not necessarily the largest)
- Small $\hat{\sigma}^2$ (square of Residual Standard Error (RSE))
- Inclusion of significant regressors (as evaluated by partial t-tests or partial F-tests)
- Want small standard error of coefficients
- Want a model that makes sense (no silly regressors with respect to subject matter or nonsense values)
- Best performance on Test data (small RMSE, high accuracy, etc.)
- All things equal, we prefer a simpler model over a more complex model (*parsimonious* models)

NOTE: There are several classes of approaches for performing feature (or variable) selection:

- subset selection – Forward Selection, Backward Elimination, Stepwise, All Possible Subsets
- shrinkage (or regularization) – LASSO, Ridge, ElasticNet
- dimension reduction – Principal Component Analysis (PCA)
- cross validation

NOTE: In order to perform feature selection, a number of decisions must be made. For example, we need to decide on a rule for deciding how variables enter or leave a model. We must decide on what makes a model the "best." It is common practice to use several different methods and compare the results. Different methods/decisions will often result in a different model being selected.

Selection Criteria

RECALL: Earlier in the semester, we defined the Residual sum of squares to be:

$$\text{Sum of Squared Residuals} = RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

NOTE: Many commonly used selection criteria use the RSS. For example,

$$R^2 = 1 - \frac{RSS}{SSTotal} \quad \text{where } SSTotal = \sum_{i=1}^{n} (y_i - \bar{y}_i)^2$$

$$R^2_{adj} = 1 - \frac{RSS/(n-p-1)}{SSTotal/(n-1)} \text{ where } p \text{ } is \text{ the number of parameters in the model}$$

Definition: The likelihood function (or more simply the likelihood) is a function over the parameters conditioned on the observed data. The likelihood describes the plausibility of the parameter values given the observed data. Many statistical approaches are based on maximizing the likelihood.

Definition: The Akaike Information Criterion (AIC) is a penalized log-likelihood measure. The AIC is given by:

$$AIC = -2\ln(L) + 2p$$

where L is the likelihood and p represents the number of parameters in the model. Smaller values of AIC are desired. Thus, we can see that the last term $(2p)$ in the equation serves as a penalty for the number of terms in the model.

Definition: The Bayesian Information Criterion (BIC) (Schwartz) is a penalized log-likelihood measure. The BIC is given by:

$$BIC = -2\ln(L) + p\ln(n)$$

where L is the likelihood and p represents the number of parameters in the model. Smaller values of BIC are desired. Thus, we can see that the last term $(p\ln(n))$ in the equation serves as a penalty for the number of terms and is connected to the sample size.

Subset Selection Methods

Backward Elimination Algorithm

- Start with the model that contains all variables of interest (we call this the full or saturated model)
- Drop variables *one at a time* that are deemed irrelevant based on some criterion. Common criterion include:
    - Drop variable that results in model with highest R^2_{adj}
    - Drop variable that results in the model with the lowest value of AIC or BIC
- Stop when no more variables can be removed from the model based on the criterion

Forward Selection Algorithm

- Start with the intercept-only model (i.e., the model that has no inputs/X's)
- Add variables *one at a time* that are deemed relevant based on some criterion. Common criterion include:
    - Add variable that results in model with highest R^2_{adj}
    - Add variable that results in the model with the lowest value of AIC or BIC
- Stop when no more variables can be added from the model based on the criterion

EXAMPLE 1: Apply forward selection based on AIC values to the Train of the Titanic data to determine the "best" subset of predictors using the step function from the stats library. Which variables are selected?

```r
#Build Intercept Only Model. NOTE: ~ 1 tells R that you only want an intercept
int_only_model <- glm(SurvivedNum ~ 1, family = binomial, data = Train)

#Build model with all potential regressors.
#In code below, SurvivedNum ~ . tells R to use all columns in dataset to predict SurvivedNum
#SurvivedNum ~ . -Survived tells R to use all columns except Survived to predict SurvivedNum
full_model <- glm(SurvivedNum ~ . -Survived, family = binomial, data = Train)

#Perform forward elimination
#Have R do it all
stats::step(object = int_only_model,
            scope = list(lower = int_only_model, upper = full_model),
            data = Train,
            direction = "forward")
```

```
## Start:  AIC=944.21
## SurvivedNum ~ 1
##
##            Df Deviance    AIC
## + Sex       1   741.25 745.25
## + Pclass    1   845.26 849.26
## + Fare      1   874.38 878.38
## + Parch     1   936.08 940.08
## <none>          942.21 944.21
## + Age       1   941.06 945.06
## + Siblings  1   941.52 945.52
##
## Step:  AIC=745.25
## SurvivedNum ~ Sex
##
##            Df Deviance    AIC
## + Pclass    1   652.16 658.16
## + Fare      1   703.88 709.88
## + Siblings  1   731.11 737.11
## <none>          741.25 745.25
## + Parch     1   739.99 745.99
## + Age       1   741.15 747.15
##
## Step:  AIC=658.16
## SurvivedNum ~ Sex + Pclass
##
##            Df Deviance    AIC
## + Age       1   632.55 640.55
## + Siblings  1   646.71 654.71
## <none>          652.16 658.16
## + Fare      1   650.50 658.50
## + Parch     1   651.70 659.70
##
## Step:  AIC=640.55
## SurvivedNum ~ Sex + Pclass + Age
##
##            Df Deviance    AIC
## + Siblings  1   620.16 630.16
## <none>          632.55 640.55
## + Parch     1   631.55 641.55
## + Fare      1   631.62 641.62
##
## Step:  AIC=630.16
## SurvivedNum ~ Sex + Pclass + Age + Siblings
##
##         Df Deviance    AIC
## + Fare   1   617.21 629.21
## <none>       620.16 630.16
## + Parch  1   620.13 632.13
##
## Step:  AIC=629.21
## SurvivedNum ~ Sex + Pclass + Age + Siblings + Fare
##
##         Df Deviance    AIC
## <none>       617.21 629.21
## + Parch  1   617.16 631.16
```

CAUTION: Automated variable selection procedures should be used with extreme caution!!!

Shrinkage (or Regularization)

IDEA: This approach involves fitting a model involving all *p* predictors. However, the estimated coefficients are shrunken towards zero relative to the least squares estimates. This shrinkage is also known as regularization.

EXAMPLE 2: How are the following formulas different?

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2$$

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2$$

NOTE: $\lambda \geq 0$

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j|$$

NOTE: $\lambda \geq 0$

NOTE: There is no penalty on the intercept.

Definition: In the case of a multiple linear regression with p potential explanatory variables (i.e., p regressors OR p inputs OR p features), we have the following model:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ip} + \epsilon_i$$

LASSO (Least Absolute Shrinkage and Selection Operator) is similar in spirit to the ordinary least squares method for estimating regression coefficients; however, LASSO estimates the regression coefficients using a modified form of the objection function. In particular, LASSO finds the $\hat{\beta}$'s by minimizing (with respect to the β's) the **objective function** given by:

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \qquad \text{for } \lambda \geq 0$$

NOTE: In the above expression, λ is called a tuning parameter and we must carefully select it. Unlike least squares, which generates only one set of coefficient estimates, lasso regression produces a different set of coefficient estimates for each value of λ. We will choose the value of the tuning parameter via cross validation.

IDEA: Fit a model for a variety of λ values (one model per λ). Choose the "best" λ based on some metric.

EXAMPLE 3: Using The file L14_Credit.csv contains information from a random sample of customers who are applying for a credit card. Information about the variables follows.

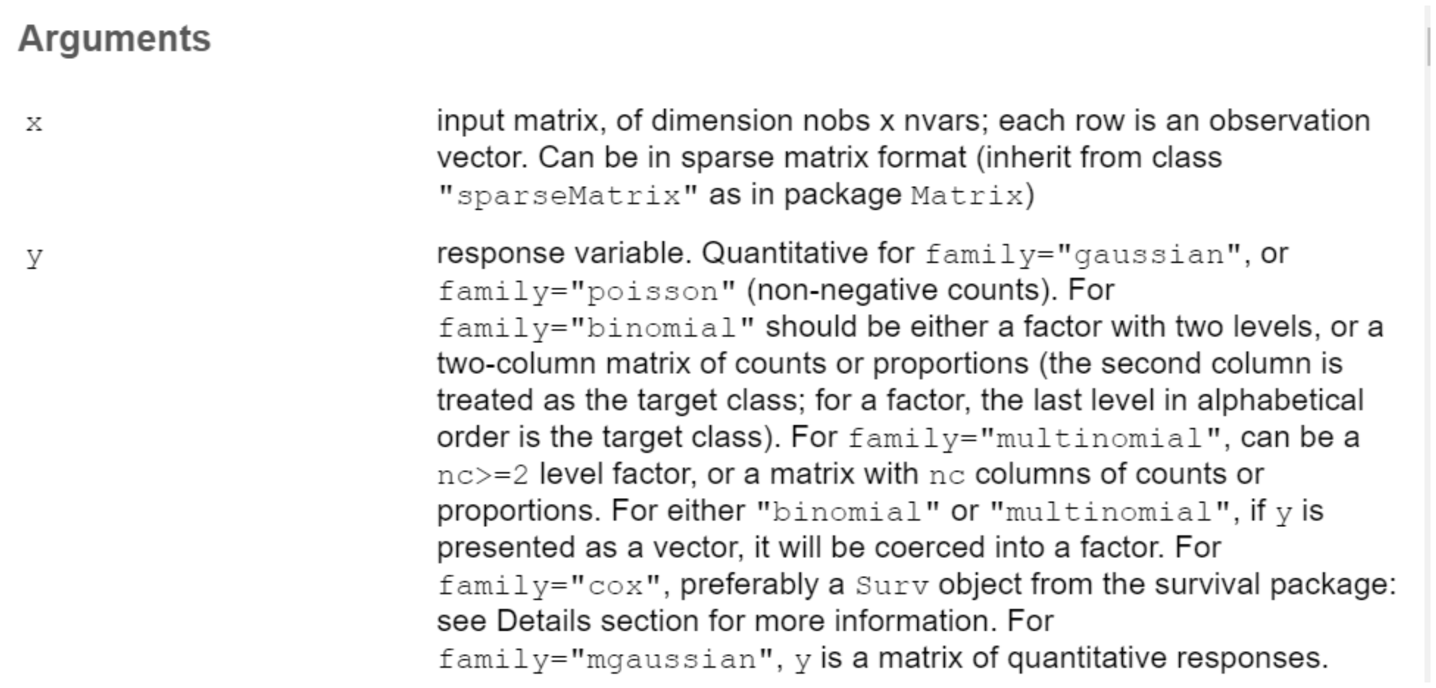
Variable	Meaning
Income	The individual's income (in thousands of dollars)
Limit	The individual's credit limit
Rating	The individual's credit rating
Cards	The number of credit cards the individual has
Age	Age in years
Education	Education in years
Own	Indicates whether the individual owns a home
Student	Indicates whether the individual is a student
Married	Indicates whether the individual is married
Region	The individual's geographic location
Balance	Average credit card balance in dollars

Our goal is to use LASSO for a regression model in which we wish to predict the \`Balance\` variable.

NOTE: When performing LASSO in R, we use a package called glmnet. Unfortunately, the initial setup for the problem is a little bit different than the procedure used for the other regression models (including *k*NN). Specifically, we will use the function glmnet (same name as the package) and it takes the form:

glmnet(x = , y =, family = “gaussian”, alpha = 1, lambda = NULL standardize = TRUE, …)

Notice that we do not build the model using the y ~ x1 + x2 + … notation that we used in the lm function.



Arguments

`x`	input matrix, of dimension nobs x nvars; each row is an observation vector. Can be in sparse matrix format (inherit from class `"sparseMatrix"` as in package `Matrix`)
`y`	response variable. Quantitative for `family="gaussian"`, or `family="poisson"` (non-negative counts). For `family="binomial"` should be either a factor with two levels, or a two-column matrix of counts or proportions (the second column is treated as the target class; for a factor, the last level in alphabetical order is the target class). For `family="multinomial"`, can be a `nc>=2` level factor, or a matrix with `nc` columns of counts or proportions. For either `"binomial"` or `"multinomial"`, if `y` is presented as a vector, it will be coerced into a factor. For `family="cox"`, preferably a `Surv` object from the survival package: see Details section for more information. For `family="mgaussian"`, `y` is a matrix of quantitative responses.

a. In order to create input matrix (matrix of x values) and response variable, run the code shown below. Explain the meaning of the first line.

```
#Put data frame in form needed for glmnet
Xmat <- model.matrix(Balance ~ . , data=Credit)[ ,-1]
y <- Credit$Balance
```

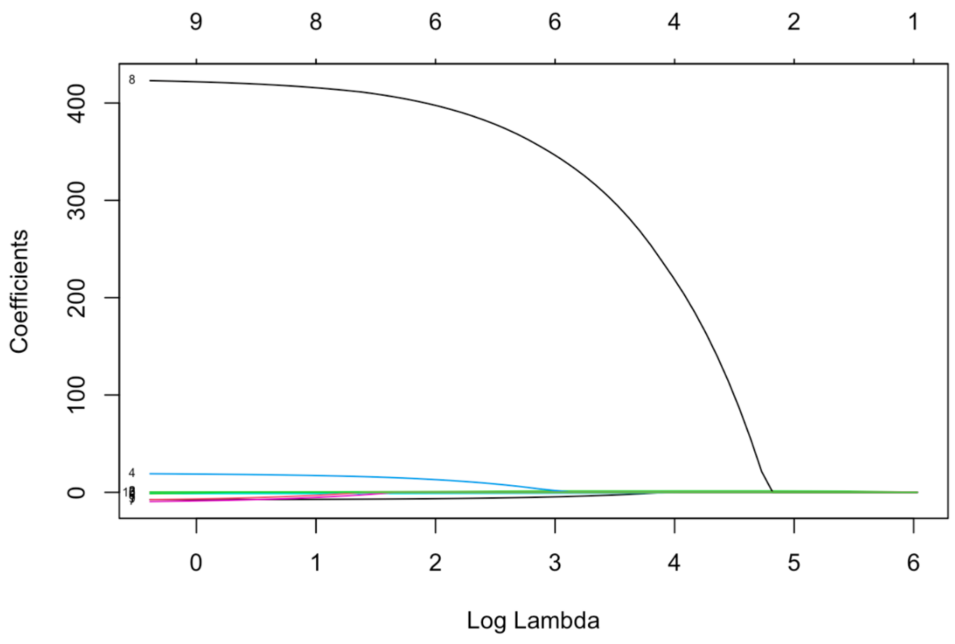
b. Using a seed of 1, perform a 50/50 training/testing split.

```
set.seed(1)
train_ind <- sample(1:nrow(Xmat), floor(0.5*nrow(Xmat)))
set.seed(NULL)

X_mat_train <- Xmat[train_ind,]
X_mat_test <- Xmat[-train_ind,]
y_train <- y[train_ind]
y_test <- y[-train_ind]
```

NOTE: We will consider a variety of values for lambda and choose the "best" one using a cross validation procedure. If you do not specify lambda (in other words you use lambda = NULL), R has a procedure for determining a set of lambda's for consideration.

c. Fit the LASSO model using the glmnet function as shown below. Correct the mistake.

```
#Fit model
lasso.mod <- glmnet(x = X_mat_train, y = y_train,
                    alpha = 1,
                    stanardize = TRUE)

#Create plot of coefficients
plot(lasso.mod, xvar="lambda",label=TRUE)
```



d. LASSO regression produces a different set of coefficient estimates for each value of λ. Extract the 44th and 17th values of λ and the coefficients they produce.

```
#Explore parts of lasso.mod object
lasso.mod$lambda #vector of lambdas chosen by R
```

```
##  [1] 416.0384632 379.0787528 345.4024412 314.7178402 286.7591747 261.2842800
##  [7] 238.0725047 216.9227995 197.6519759 180.0931191 164.0941427 149.5164714
## [13] 136.2338401 124.1312013 113.1037275 103.0559040  93.9007016  85.5588220
## [19]  77.9580121  71.0324372  64.7221112  58.9723772  53.7334338  48.9599036
## [25]  44.6104407  40.6473721  37.0363716  33.7461624  30.7482464  28.0166569
## [31]  25.5277343  23.2599208  21.1935735  19.3107949  17.5952771  16.0321611
## [37]  14.6079081  13.3101818  12.1277420  11.0503468  10.0686644   9.1741920
## [43]   8.3591821   7.6165754   6.9399398   6.3234147   5.7616600   5.2498100
## [49]   4.7834313   4.3584845   3.9712887   3.6184904   3.2970337   3.0041343
## [55]   2.7372553   2.4940851   2.2725174   2.0706332   1.8866839   1.7190761
## [61]   1.5663581   1.4272072   1.3004180   1.1848925   1.0796299   0.9837185
## [67]   0.8963277   0.8167004   0.7441470   0.6780390
```

```
dim(coef(lasso.mod)) #dimension of coefficent's estimated
```

```
## [1] 12 70
```

```
coef_mat <- as.matrix(coef(lasso.mod)) #Create object to hold coefficients

#Explore coefficients for 83rd lambda
lasso.mod$lambda[44]
```

```
## [1] 7.616575
```

```
log(lasso.mod$lambda[44])
```

```
## [1] 2.030327
```

```
coef_mat[,44]
```

```
##  (Intercept)       Income        Limit       Rating        Cards          Age
## -428.4793292   -6.4754378    0.1977562    0.7942805   12.8695635   -1.0653796
##    Education       OwnYes   StudentYes   MarriedYes  RegionSouth   RegionWest
##    0.0000000    0.0000000  396.6092725    0.0000000    0.0000000    0.0000000
```

```
#Explore coefficients for 68th lambda
lasso.mod$lambda[17]
```

```
## [1] 93.9007
```

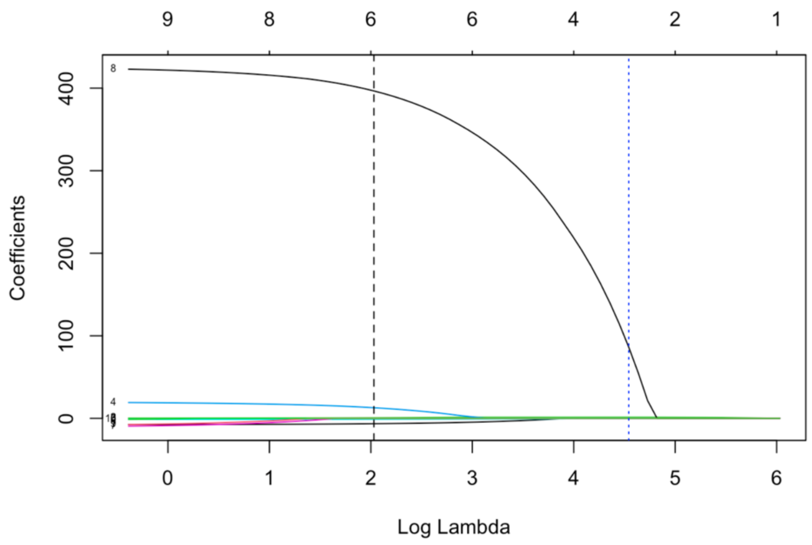
```
log(lasso.mod$lambda[17])
```

```
## [1] 4.542238
```

```
coef_mat[,17]
```

```
##   (Intercept)        Income         Limit        Rating         Cards
## -163.48637182    0.00000000    0.03170999    1.46327527    0.00000000
##           Age     Education        OwnYes    StudentYes    MarriedYes
##    0.00000000    0.00000000    0.00000000   86.16622704    0.00000000
##   RegionSouth    RegionWest
##    0.00000000    0.00000000
```

```r
#Plot
#Create plot of coefficients
plot(lasso.mod, xvar="lambda",label=TRUE)
abline(v=log(lasso.mod$lambda[44]),col="black", lty = "dashed")
abline(v=log(lasso.mod$lambda[17]),col="blue", lty = "dotted")
```

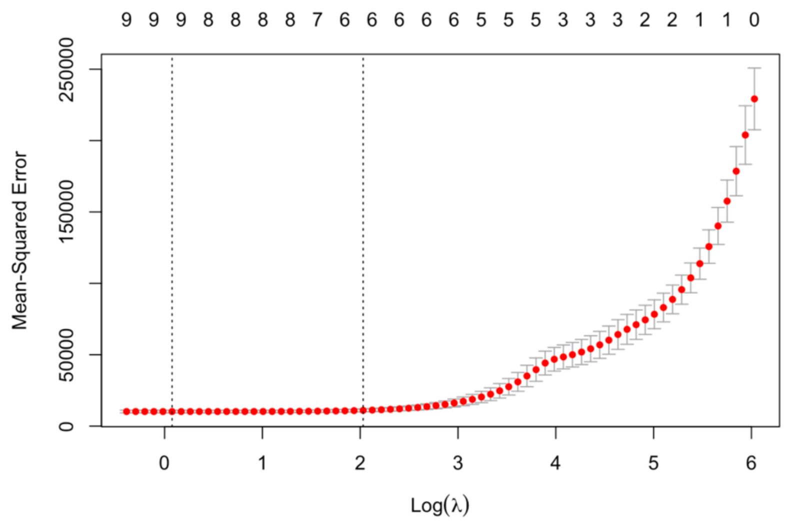


NOTE: As the value of the tuning parameter (λ) gets bigger, we are increasing the penalty on the size of the β's. This means the magnitudes of the coefficients are being shrunk (i.e., pushed towards 0).

IDEA: LASSO is viewed as a variable selection procedure because a coefficient of 0 indicates that the variable is not important to the model.

e. We will choose the value of the tuning parameter, λ, via cross validation. Fortunately, the R function cv.glmnet will perform k-fold cross validation for us. Use a random number seed of 1.

```r
set.seed (1)
cv.out <- cv.glmnet(x= X_mat_train, y = y_train,
                    alpha = 1, stanardize = TRUE,
                    nfolds=10)
plot(cv.out)
```



f. The plot in the previous part leads us to two options for λ. For each, identify the value of λ and determine which variables are excluded.

```
#Option 1: Pick the lambda that produce the best (min)MSE
bestlam1 <- cv.out$lambda.min
#Predict the responses for the test set (use for MSE calc)
lasso.pred1 <- predict(cv.out , s = bestlam1,
                      newx = X_mat_test)
#Find the coefficients
lasso.coef1 <- predict(cv.out , s = bestlam1,
                      type = "coefficients")
bestlam1
```

```
## [1] 1.07963
```

```
lasso.coef1
```

```
## 12 x 1 sparse Matrix of class "dgCMatrix"
##                         s1
## (Intercept) -449.3300349
## Income        -7.4565015
## Limit          0.2398972
## Rating         0.4002428
## Cards         18.7513316
## Age           -1.1594546
## Education      .
## OwnYes        -8.6029163
## StudentYes   421.5142729
## MarriedYes    -6.5841624
## RegionSouth   -0.5347754
## RegionWest     .
```

```
#Option 2: Pick the largest value of lambda such that error
#is within 1 standard error (1se) of the minimum
bestlam2 <- cv.out$lambda.1se
lasso.pred2 <- predict(cv.out, s = bestlam2,
                      newx = X_mat_test)
lasso.coef2 <- predict(cv.out, s = bestlam2,
                     type = "coefficients")
bestlam2
```

```
## [1] 7.616575
```

```
lasso.coef2
```

```
## 12 x 1 sparse Matrix of class "dgCMatrix"
##                         s1
## (Intercept) -428.4793292
## Income        -6.4754378
## Limit          0.1977562
## Rating         0.7942805
## Cards         12.8695635
## Age           -1.0653796
## Education      .
## OwnYes         .
## StudentYes   396.6092725
## MarriedYes     .
## RegionSouth    .
## RegionWest     .
```

g. How does the MSE of the test set for LASSO compare to the MSE of the test set for a regular regression model using all variables?