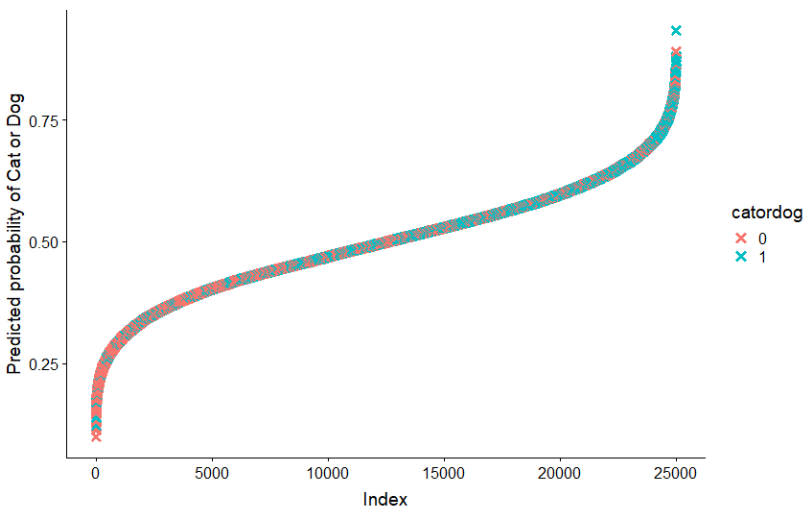
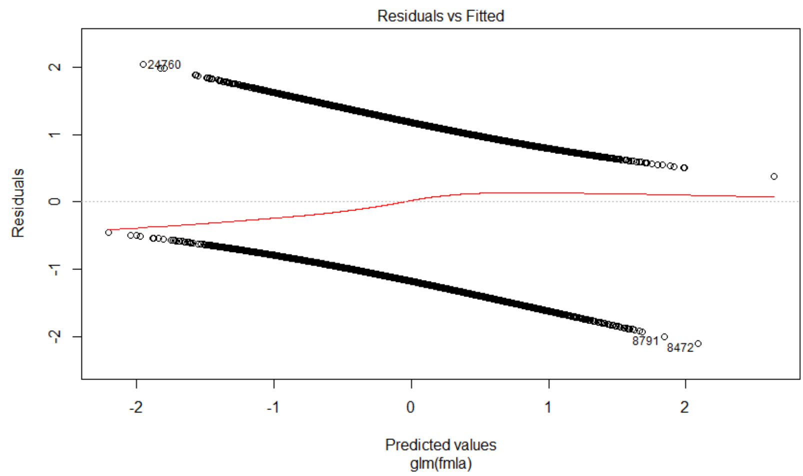
Blake Simmons
Daniel Gomes
John Gomes
CIS 490: Sectional Project 1

Part 1: Dogs vs. Cats – Logistic Regression

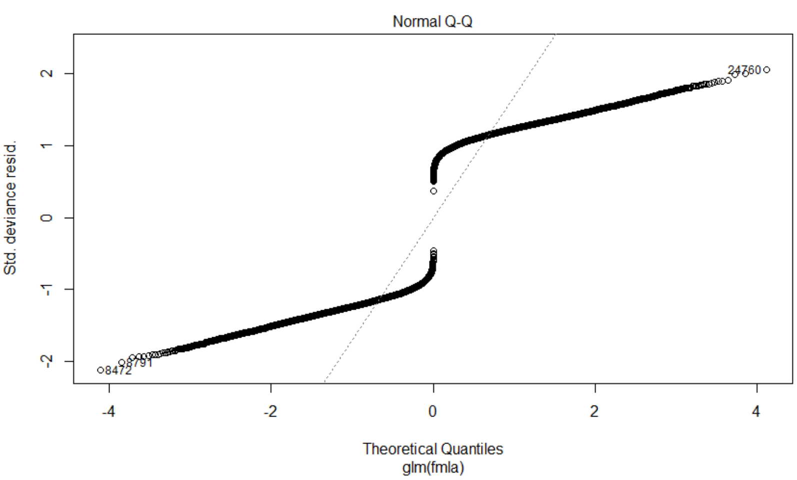
Graphs, tables, and descriptive statistics

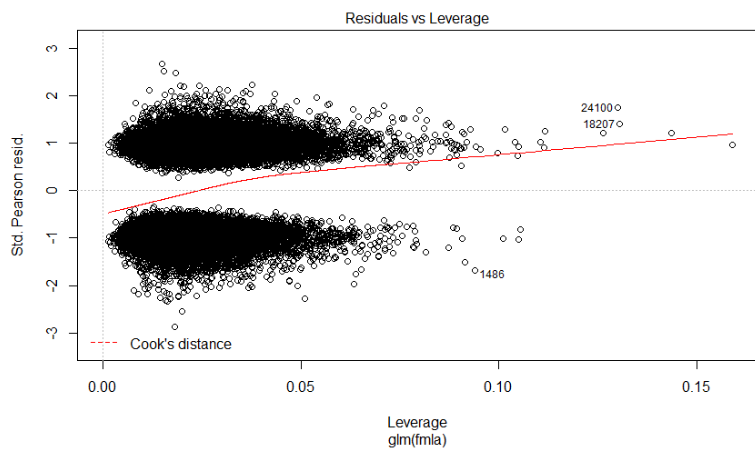
fmla = V626 ~ V1 + V2 + V3 … V625

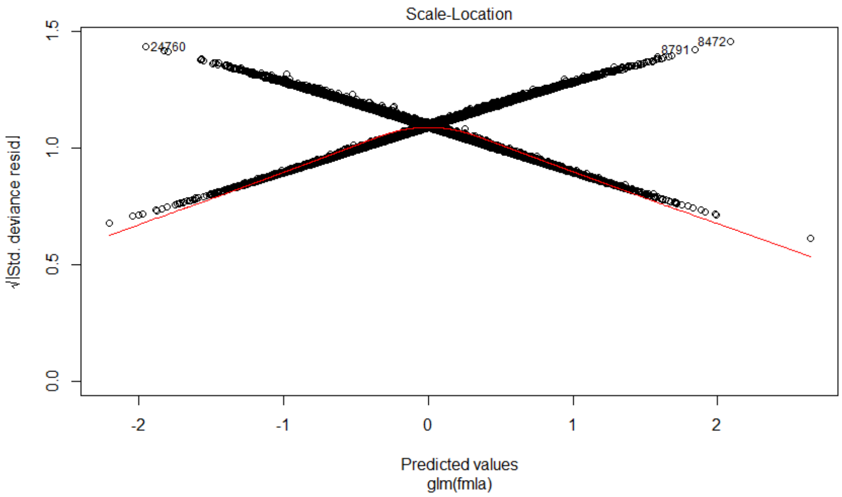


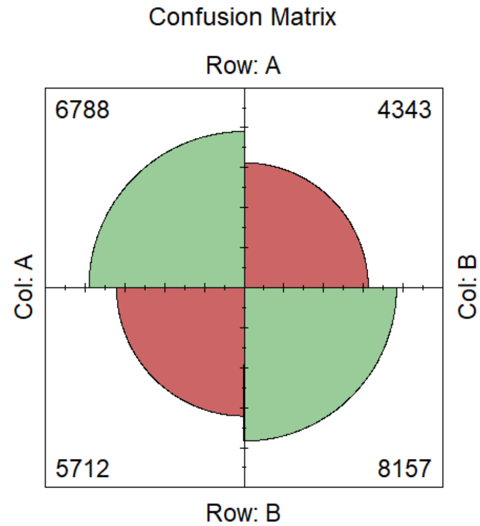


Accuracy of Logit Model









Descriptive Statistics on V626	Outcome
Mean	0.5
Variance	0.25001
Median	0.5
Std	0.5001
Min, 1st quartile, Median, 3rd quartile, Max	0,0,0,1,1

Determining X and Y

Figuring out what X and Y for this dataset was fairly easy. The dataset consisted of over 600 variables, most of them of continuous values, except one.

Y:

The variable V626, (or column 627), contained values 0 or 1. This column indicated that 0 is for “cat” and 1 is for “dog”. From that we can consider this variable as the, the predicted value, so it is our Y.

X:

All other variables we can be considered as predictor values. From that, we decided to put these values as our X. These values will help predict whether it’s 0 or 1, “cat” or “dog”.

Code and Output - See end of report

Final Estimated Model - V626= -0.0384792 + V1* 0.2449542 + ... + V625 * 0.3200868

(Too many variables)

Validation, cross validation, or classification evaluation results

Concordance: 63.4% Sensitivity: 65.256% AUROC Curve: 63.31%

Conf matrix: 0 1 Miss classification error: 40.22%

0 6788 4343

1 5712 8157

Describe Validation

From concordance, we are trying to see if our logistic model predicted score "Good" is greater than "Bad". We have a concordance of 63.4% which isn't too bad but not great.

From sensitivity, the logistic model had a 65.256% proportion of cases that were correctly identified by the test.

From the AUROC curve, will tell us how much the model is capable of distinguishing from cat or dog. 63.31% means it could be better as it barely passed the test.

From the miss class. Error, we want it to be low. %40.22 isn't to bad but again not good.

From the confusion matrix, the logistic model was able to correctly find 6788 Cats and 8157 Dogs. The model incorrectly found 5712 Cats and 4343 Dogs

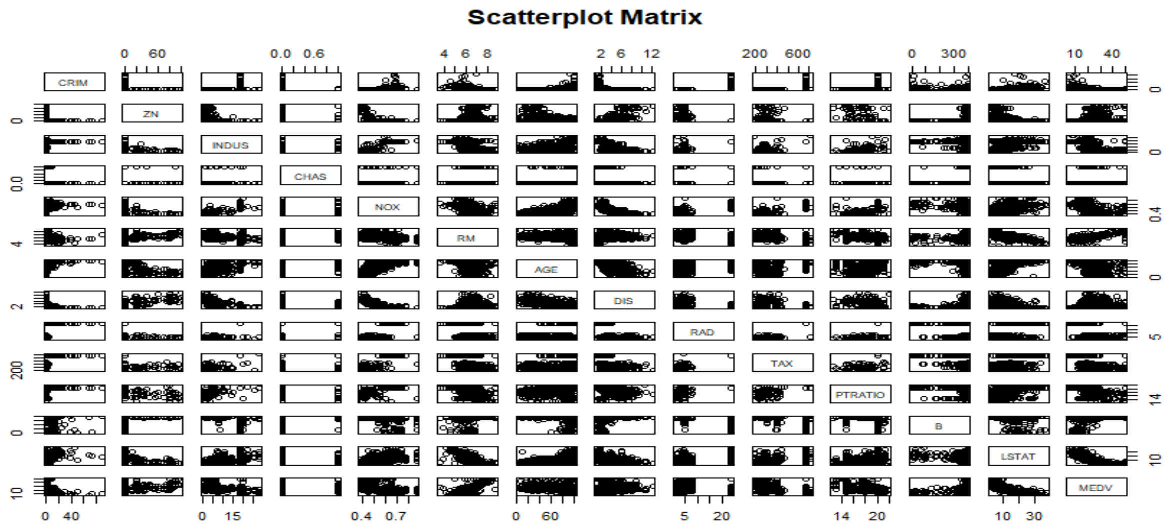
Part 2: Housing – Multiple Linear, Ridge and Lasso Regression

General Statistics

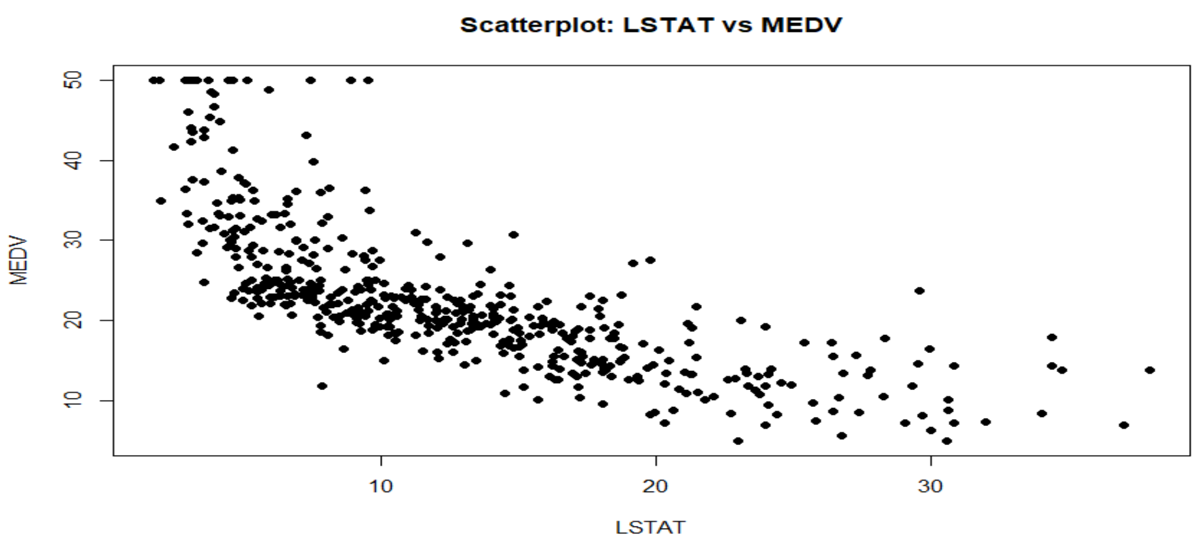
Mean		Variance		Median	
CRIM:	3.61	**CRIM:**	7.39e+01	**CRIM:**	0.26
ZN:	11.36	**ZN:**	5.44e+02	**ZN:**	0.00
INDUS:	11.14	**INDUS:**	4.71e+01	**INDUS:**	9.69
CHAS:	0.069	**CHAS:**	6.43e-02	**CHAS:**	0.00
NOX:	0.55	**NOX:**	1.34e-02	**NOX:**	0.54
RM:	6.28	**RM:**	4.94e-01	**RM:**	6.21
AGE:	68.57	**AGE:**	7.92e+02	**AGE:**	77.50
DIS:	18.45	**DIS:**	4.43+00	**DIS:**	3.21
RAD:	9.55	**RAD:**	7.58e+01	**RAD:**	5.00
TAX:	408.24	**TAX:**	2.84e+04	**TAX:**	330.00
PTRATIO:	18.45	**PTRATIO:**	4.69e+00	**PTRATIO:**	19.05
B:	356.67	**B:**	8.33e+03	**B:**	391.44
LSTAT:	12.65	**LSTAT:**	5.09e+01	**LSTAT:**	11.36
MEDV:	22.53	**MEDV:**	8.46e+01	**MEDV:**	21.20

Determining X and Y

We used the following scatter plot matrix to determine variables with high correlations to others. In the end MEDV seemed like the best choice for the predicted value, and we chose to use all other variables as predictors.



An example of a scatter plot showing a strong correlation between LSTAT and MEDV:



Final formula

Y = “MEDV”

X = All other variables

Cross Validation Method

Holdout (80% vs 20%)

Summary of model fit

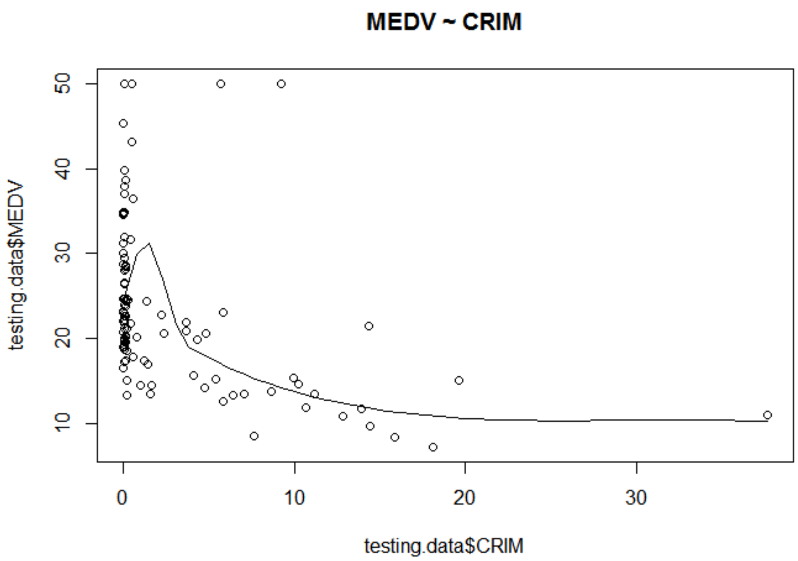
```
Call:
lm(formula = frml, data = housing)

Residuals:
    Min      1Q  Median      3Q     Max
-15.595  -2.730  -0.518   1.777  26.199

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.646e+01  5.103e+00   7.144 3.28e-12 ***
CRIM        -1.080e-01  3.286e-02  -3.287 0.001087 **
ZN           4.642e-02  1.373e-02   3.382 0.000778 ***
INDUS        2.056e-02  6.150e-02   0.334 0.738288
CHASTRUE     2.687e+00  8.616e-01   3.118 0.001925 **
NOX         -1.777e+01  3.820e+00  -4.651 4.25e-06 ***
RM           3.810e+00  4.179e-01   9.116  < 2e-16 ***
AGE          6.922e-04  1.321e-02   0.052 0.958229
DIS         -1.476e+00  1.995e-01  -7.398 6.01e-13 ***
RAD          3.060e-01  6.635e-02   4.613 5.07e-06 ***
TAX         -1.233e-02  3.760e-03  -3.280 0.001112 **
PTRATIO     -9.527e-01  1.308e-01  -7.283 1.31e-12 ***
B            9.312e-03  2.686e-03   3.467 0.000573 ***
LSTAT       -5.248e-01  5.072e-02 -10.347  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.745 on 492 degrees of freedom
Multiple R-squared:  0.7406,	Adjusted R-squared:  0.7338
F-statistic: 108.1 on 13 and 492 DF,  p-value: < 2.2e-16
```

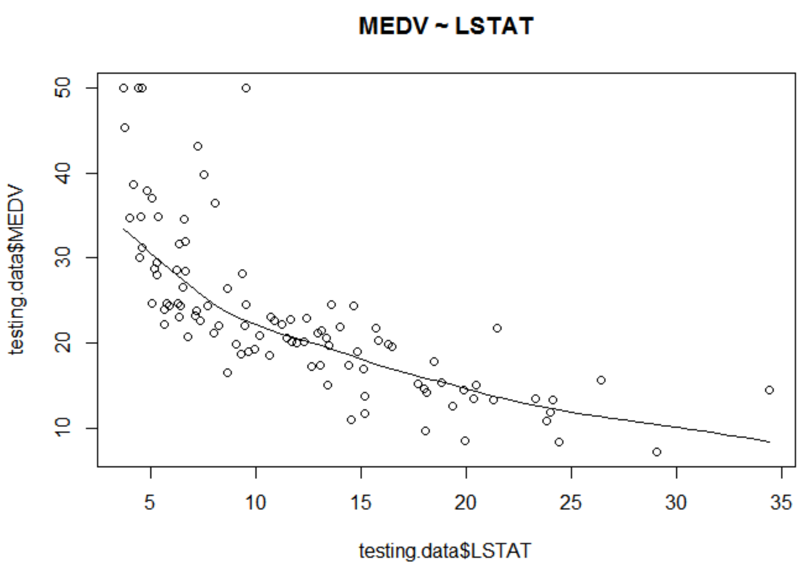
Many of the variables are shown to be good predictors of the MEDV variable as shown by the stars beside their p-values. The Multiple R-Squared value of 0.7406 and low total p-value indicates a good fit. We could potentially improve the fit by removing the variables AGE and INDUS, which appear to be poor predictors.

Final Model

E(Y) = 35.259615 + -0.108821*CRIM + 0.054896*ZN + 0.023760*INDUS + 2.524163*CHAS + -17.573132*NOX + 3.665491*RM + 0.000461*AGE + -1.554546*DIS + 1.488905*RAD2 + 4.681254*RAD3 + 2.576234*RAD4 + 2.918493*RAD5 * 1.185839*RAD6 + 4.878992*RAD7 + 4.839836*RAD8 + 7.461674*RAD24 -0.008748*TAX + -0.972419*PTRATIO + 0.009394*B + -0.529226*LSTAT

Predictions against variables (showing only the highest correlating graphics):

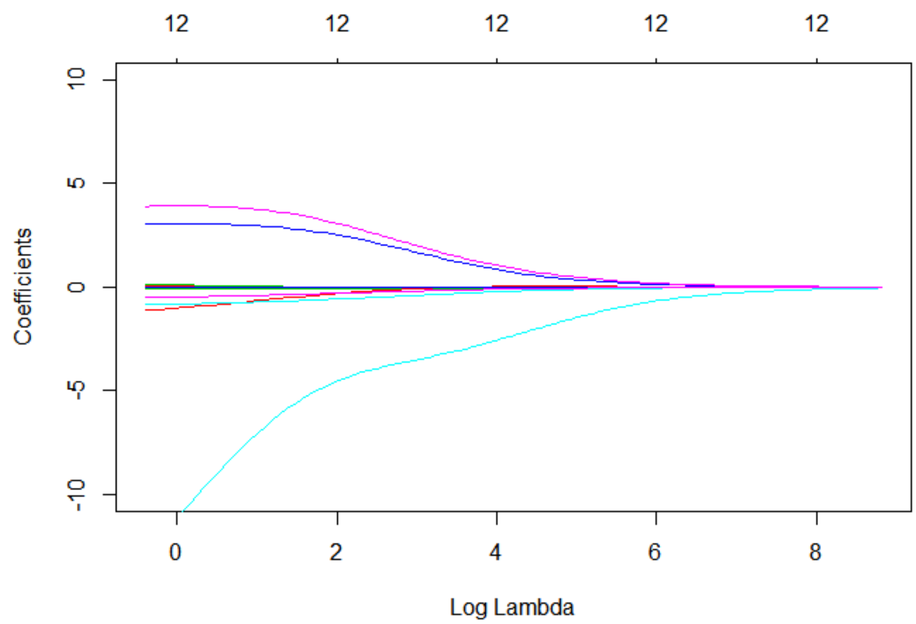




The prediction based on each predictor variable follows a path tighter to the predicted values.

Ridge Regression

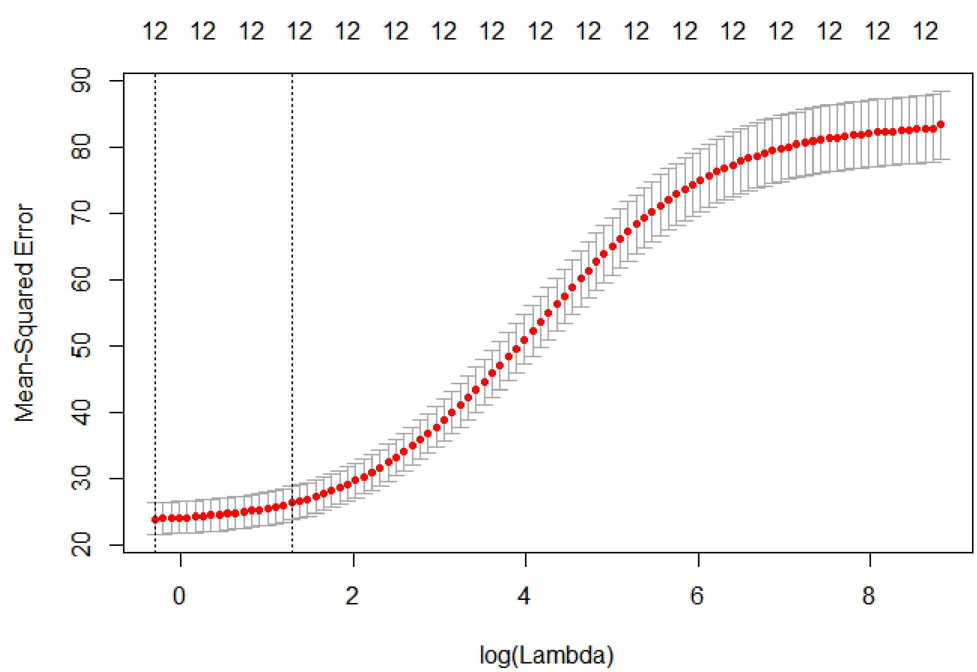
Best Lambda: 0.619915



When lambda is set to 8, the Sum of Squares becomes close to zero. As it is lessened the SSE becomes bigger.

Cross-Validation Used

K-Fold (10 folds, visualization below)



The cross validation tells us where the best value range for lambda lies. Towards the left of the graph, the mean-squared error levels off at a low amount, suggesting the best lambda is within this range.

Results

The ridge regression lambda effectively brings some coefficients very close to zero. The results of this are:

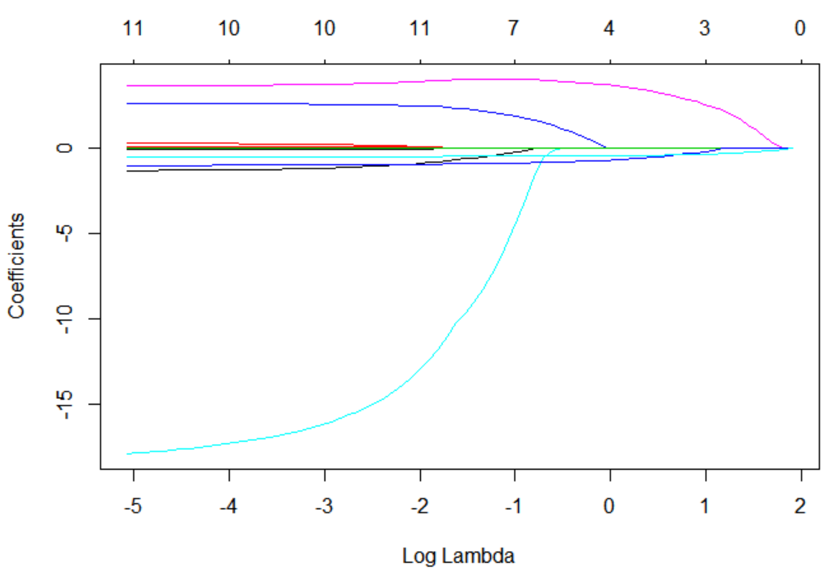
```
> ridge.results
14 x 1 sparse Matrix of class "dgCMatrix"
                        1
(Intercept)  32.262004014
(Intercept)   .
CRIM         -0.099228668
ZN            0.032089861
INDUS        -0.046346825
CHASTRUE      3.044015331
NOX         -12.361675962
RM            3.906808828
AGE          -0.002168196
DIS          -1.101591472
RAD           0.132836240
TAX          -0.005960766
PTRATIO      -0.836374783
LSTAT        -0.489658705
```

Final Model

E(Y) = 2890.461+ 0.619915*(-0.108821*CRIM + 0.054896*ZN + 0.023760*INDUS + 2.524163*CHAS + -17.573132*NOX + 3.665491*RM + 0.000461*AGE + -1.554546*DIS + 1.488905*RAD2 + 4.681254*RAD3 + 2.576234*RAD4 + 2.918493*RAD5 * 1.185839*RAD6 + 4.878992*RAD7 + 4.839836*RAD8 + 7.461674*RAD24 -0.008748*TAX + -0.972419*PTRATIO + 0.009394*B + -0.529226*LSTAT)^2

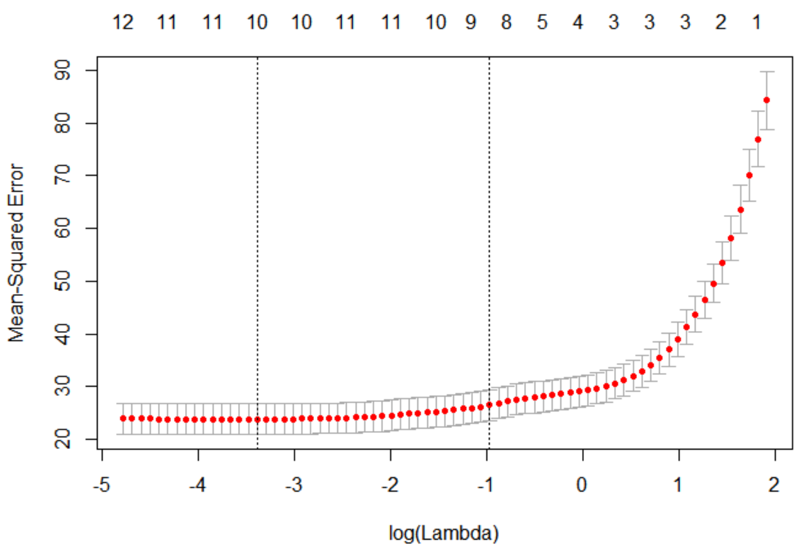
LASSO Regression

Best Lambda: 0.3789258



Cross Validation used

K-fold (10 folds, visualization below)



Cross validation once again determines the best lambda for the model

LASSO Model results

```
(Intercept) 21.6834491048
(Intercept)  .
CRIM        -0.0361155884
ZN           .
INDUS        .
CHASTRUE     2.0058948525
NOX         -4.4629140052
RM           4.1558518545
AGE          .
DIS         -0.3357429179
RAD          .
TAX         -0.0002192817
PTRATIO     -0.7942328505
LSTAT       -0.5358125716
```

The lasso model, unlike the ridge model, is actually capable of bringing the coefficients of some variables to zero as shown above in ZN, INDUS, AGE, and RAD

Final Model

3267.739 + 0.3789258 * |-0.108821*CRIM + 0.054896*ZN + 0.023760*INDUS + 2.524163*CHAS + -17.573132*NOX + 3.665491*RM + 0.000461*AGE + -1.554546*DIS + 1.488905*RAD2 + 4.681254*RAD3 + 2.576234*RAD4 + 2.918493*RAD5 * 1.185839*RAD6 + 4.878992*RAD7 + 4.839836*RAD8 + 7.461674*RAD24 -0.008748*TAX + -0.972419*PTRATIO + 0.009394*B + -0.529226*LSTAT|

ALL CODE FOR BOTH PARTS AT END OF PAPER

RMSE Comparison

Multiple Linear RMSE: 22.753303

Ridge RMSE: 5.32333085023029

LASSO RMSE: 4.19126837643412

Ridge RMSE is the lowest, suggesting Ridge Regression appears to be the most accurate modeling method.

References

"Concept: Sensitivity and Specificity - Using the ROC Curve to Measure." *Term: Singleton Birth*, 21 Oct. 2001, mchp-appserv.cpe.umanitoba.ca/viewConcept.php?printer=Y&conceptID=1047.

"LOGIT REGRESSION | R DATA ANALYSIS EXAMPLES." *IDRE Stats*, stats.idre.ucla.edu/r/dae/logit-regression/.

Narkhede, Sarang. "Understanding AUC - ROC Curve – Towards Data Science." *Towards Data Science*, Towards Data Science, 26 June 2018, towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5.

Phillips, Nathaniel D. "YaRrr! The Pirate's Guide to R." *Home*, 22 Jan. 2018, bookdown.org/ndphillips/YaRrr/logistic-regression-with-glmfamily-binomial.html.

Prabhakaran, Selva. "Logistic Regression." *Missing Value Treatment*, 2017, r-statistics.co/Logistic-Regression-With-R.html.

Simplilearn. "Logistic Regression in R | Logistic Regression in R Example | Data Science Algorithms | Simplilearn." *YouTube*, YouTube, 9 Aug. 2018, www.youtube.com/watch?v=XycruVLySDg&feature=youtu.be.

Starmer, StatQuest with Josh. "Logistic Regression in R, Clearly Explained!!!!" *YouTube*, YouTube, 26 July 2018, www.youtube.com/watch?v=C4N3_XJJ-jU.

Prabhakaran, Selva. "Ridge Regression." *r-Statistics.co*, r-statistics.co/Ridge-Regression-With-R.html.

Libraries Used:

Caret, graphics, InformationValue, car, gdata, ggplot2, cowplot, leaps, pscl

CODE AND OUPUT BEGINS HERE

PART 1

```
> # Daniel Gomes, Blake Simmons, John Gomes
> # CIS490 Machine Learning
> # catsvsdogs - Logit Regression, Plotting, Variable 93 and 623
>
> # this deletes every variable in the workspce except catsvsdogs and .glm
> keep(catsvsdogs, catsvsdogs.glm, sure = TRUE)
Warning message:
In keep(catsvsdogs, catsvsdogs.glm, sure = TRUE) :
  you tried to keep "catsvsdogs" which doesn't exist in workspace - nothing w
as removed
>
> # libraries needed
> library(caret)
> library(graphics)
> library(InformationValue) # has optimalCutoff
> library(car) # has vif
> library(gdata) # has keep
> library(ggplot2)
> library(cowplot)
> library(pscl)
>
> # load data
> if (!exists("catsvsdogs")){
+   catsvsdogs <- read.csv("cats-vs-dogs25.csv")
+ }
>
> # these next two lines create our formula for all variables in the set
> xnam <- paste0("V", 1:625)
> fmla <- as.formula(paste("V626 ~", paste(xnam, collapse= "+")))
>
> # convert column to categorical
> # logistic regression model on whole set
> if (!exists("catsvsdogs.glm")){
+   catsvsdogs.glm <- glm(fmla, data = catsvsdogs, family = "binomial"(link =
"logit"))
+ }
>
> # gives the beta coefficients, Standard error, z Value and p Value
> summary(catsvsdogs.glm)

Call:
glm(formula = fmla, family = binomial(link = "logit"), data = catsvsdogs)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.10139  -1.12781  -0.04275   1.13171   2.04245

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.0384792  0.0595082  -0.647 0.517878
```

```
V1           0.2449542  0.1362867   1.797 0.072281 .
V2          -0.3538989  0.1452682  -2.436 0.014843 *
V3           0.1314706  0.1458886   0.901 0.367497
V4          -0.0810379  0.1415996  -0.572 0.567117
V5           0.1563675  0.1423259   1.099 0.271917
V6           0.0083211  0.1413228   0.059 0.953048
V7          -0.0980113  0.1403191  -0.698 0.484872
V8           0.0182275  0.1384780   0.132 0.895279
V9           0.0606220  0.1349729   0.449 0.653329
V10          0.1048742  0.1343548   0.781 0.435051
V11          0.0279389  0.1359978   0.205 0.837231
V12          0.0681751  0.1364798   0.500 0.617410
V13         -0.1211476  0.1337877  -0.906 0.365189
V14         -0.2195757  0.1347969  -1.629 0.103326
V15          0.1330690  0.1362515   0.977 0.328746
V16         -0.0694667  0.1337163  -0.520 0.603407
V17          0.0971324  0.1355961   0.716 0.473784
V18         -0.0078639  0.1382705  -0.057 0.954646
V19         -0.3262506  0.1387319  -2.352 0.018690 *
 [ reached getOption("max.print") -- omitted 606 rows ]
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 34657  on 24999  degrees of freedom
Residual deviance: 33289  on 24374  degrees of freedom
AIC: 34541

Number of Fisher Scoring iterations: 4

> pR2(catsvsdogs.glm)
          llh       llhNull            G2      McFadden          r2ML 
r2CU 
-1.664425e+04 -1.732868e+04  1.368851e+03  3.949670e-02  5.328203e-02  7.1042
71e-02
>
>
> # plots some nice graphs that I don't know what they mean yet
> plot(catsvsdogs.glm, ask=F)
>
> # convert to factor
> catsvsdogs$V626 <- factor(catsvsdogs$V626)
>
> # show predicted probabilites that cat was cat or dog was dog
> predicted.data <- data.frame(probability.of.catordog = catsvsdogs.glm$fitte
d.values, catordog = catsvsdogs$V626)
> predicted.data <- predicted.data[order(predicted.data$probability.of.catord
og, decreasing = FALSE),]
> predicted.data$rank <- 1:nrow(predicted.data)
>
> # plot prediction
> ggplot(data = predicted.data, aes(x = rank, y = probability.of.catordog)) +
+   geom_point(aes(color = catordog), alpha = 1, shape = 4, stroke = 2) +
+   xlab("Index") + ylab("Predicted probability of Cat or Dog")
>
> # predicted scores
> predicted <- predict(catsvsdogs.glm, catsvsdogs, type="response")
>
```

```
> # find the optimal cutoff to improve the prediction of 1's, 0's
> optCutOff <- optimalCutoff(catsvsdogs$V626, predicted)[1]
>
> # these need to have VIF well below 4
> vif(catsvsdogs.glm)
      V1       V2       V3       V4       V5       V6       V7       V8
V9      V10      V11      V12      V13
7.875026 8.805332 8.899014 8.403481 8.433346 8.288225 8.114385 7.884631 7.450
487 7.365577 7.524649 7.564850 7.294677
     V14      V15      V16      V17      V18      V19      V20      V21
V22      V23      V24      V25      V26
7.392139 7.597184 7.332494 7.520124 7.866875 7.989995 8.034254 8.209757 8.430
977 8.545688 8.900721 7.663646 9.445758
     V27      V28      V29      V30      V31      V32      V33      V34
V35      V36      V37      V38      V39
9.314515 8.880666 8.115098 7.848822 7.490748 7.213470 6.937267 6.761227 6.638
146 6.730919 6.659914 6.769738 6.712297
     V40      V41      V42      V43      V44      V45      V46      V47
V48      V49      V50      V51      V52
6.656668 6.447724 6.625632 6.995265 7.169918 7.326195 7.680601 8.334773 8.677
395 9.402797 9.575770 9.368985 8.997930
     V53      V54      V55      V56      V57      V58      V59      V60
V61      V62      V63      V64      V65
8.077938 7.146270 6.622066 6.166584 6.072522 5.778552 5.771607 5.781121 5.788
741 5.820013 5.702428 5.739319 5.591384
     V66      V67      V68      V69      V70      V71      V72      V73
V74      V75      V76      V77      V78
5.629079 5.597689 5.779995 6.169500 6.288982 6.703670 7.364431 7.986639 8.702
572 9.503611 9.079436 8.860788 7.549525
     V79      V80      V81      V82      V83      V84      V85      V86
V87      V88      V89      V90      V91
6.689227 6.061851 5.694241 5.406810 5.117683 5.139802 4.990966 5.100715 5.306
115 5.283603 5.214130 4.950295 5.070586
     V92      V93      V94      V95      V96      V97      V98      V99     V
100
5.012437 5.173229 5.499125 5.905430 6.158074 6.633311 7.665395 8.425585 9.011
850
 [ reached getOption("max.print") -- omitted 525 entries ]
>
> # The lower the misclassification error, the better is your model.
> misClassError(catsvsdogs$V626, predicted, threshold = optCutOff)
[1] 0.4022
>
> # Greater the area under the ROC curve, better the predictive ability of th
e model.
> plotROC(catsvsdogs$V626, predicted)
>
> # higher the concordance, the better is the quality of model
> Concordance(catsvsdogs$V626, predicted)
$`Concordance`
[1] 0.6336454

$Discordance
[1] 0.3663546

$Tied
[1] -5.551115e-17

$Pairs
```

```
[1] 156250000

>
> sensitivity(catsvsdogs$V626, predicted, threshold = optCutOff)
[1] 0.65256
> specificity(catsvsdogs$V626, predicted, threshold = optCutOff)
[1] 0.54304
>
> # do the confusion matrix
> confusionMatrix(catsvsdogs$V626, predicted, threshold = optCutOff)
     0    1
0 6788 4343
1 5712 8157
> ctable <- as.table(matrix(c(6788, 4343,5712, 8157), nrow = 2, byrow = TRUE)
)
> fourfoldplot(ctable, color = c("#CC6666", "#99CC99"),
+              conf.level = 0, margin = 1, main = "Confusion Matrix")
> training.indices <- sample(1:nrow(catsvsdogs), 0.8 * nrow(catsvsdogs))
> training.data <- catsvsdogs[training.indices, ]
> testing.data <- catsvsdogs[-training.indices, ]
catsvsdogs.glm <- glm(fmla, data = training.data, family = "binomial"(link =
"logit"))
>
> summary(catsvsdogs.glm)

Call:
glm(formula = fmla, family = binomial(link = "logit"), data = training.data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0318  -1.1264   0.6042   1.1240   1.9614

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.0691677  0.0670735  -1.031 0.302437
V1           0.3608173  0.1527381   2.362 0.018161 *
V2          -0.4362991  0.1636641  -2.666 0.007680 **
V3           0.1420824  0.1634280   0.869 0.384635
V4           0.0728197  0.1585012   0.459 0.645928
V5           0.0779964  0.1600809   0.487 0.626095
V6           0.0439133  0.1593503   0.276 0.782873
V7          -0.2285973  0.1598364  -1.430 0.152661
V8           0.0497947  0.1563732   0.318 0.750155
V9           0.1697184  0.1525407   1.113 0.265876
V10          0.0067515  0.1512697   0.045 0.964400
V11         -0.0014087  0.1535333  -0.009 0.992679
V12          0.1451403  0.1536491   0.945 0.344852
V13         -0.0831486  0.1513211  -0.549 0.582673
V14         -0.2423650  0.1521736  -1.593 0.111230
V15          0.1529396  0.1532982   0.998 0.318444
V16         -0.2039239  0.1499002  -1.360 0.173704
V17          0.1908554  0.1530790   1.247 0.212479
V18          0.0290096  0.1558877   0.186 0.852372
V19         -0.3744723  0.1573099  -2.380 0.017290 *
 [ reached getOption("max.print") -- omitted 606 rows ]
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)
```

```
    Null deviance: 27726  on 19999  degrees of freedom
Residual deviance: 26490  on 19374  degrees of freedom
AIC: 27742

Number of Fisher Scoring iterations: 4

> pR2(catsvsdogs.glm)
          llh       llhNull            G2      McFadden          r2ML
r2CU
-1.324511e+04 -1.386278e+04  1.235353e+03  4.455645e-02  5.989870e-02  7.9865
36e-02
```

CODE AND OUTPUT: PART 2

```
> # Blake Simmons, Daniel Gomes, John Gomes
> # CIS490
> # Project 1
>
> library(ridge)
> library(glmnet)
> library(ggplot2)
> library(jtools)
>
>
> #import, naming, and partitioning of data
> housing <- read.table('housing.data', header = FALSE)
> names(housing) <- c("CRIM", "ZN", "INDUS", "CHAS", "NOX", "RM", "AGE", "DIS
", "RAD", "TAX", "PTRATIO", "B", "LSTAT", "MEDV")
> housing$CHAS = as.logical(housing$CHAS)
> sapply(housing, class)
     CRIM        ZN     INDUS      CHAS       NOX        RM       AGE
"numeric" "numeric" "numeric" "logical" "numeric" "numeric" "numeric"
      DIS       RAD       TAX   PTRATIO         B     LSTAT      MEDV
"numeric" "integer" "numeric" "numeric" "numeric" "numeric" "numeric"
> set.seed(86)
> training.indices <- sample(1:nrow(housing), 0.8 * nrow(housing))
>
>
> #multiple linear regression
> training.data <- housing[training.indices, ]
> testing.data <- housing[-training.indices, ]
>
> frml <- MEDV ~ CRIM + ZN + INDUS + CHAS + NOX + RM + AGE + DIS + RAD + TAX
+ PTRATIO + B + LSTAT
> multi.var.model <- lm(frml, data = training.data)
> print(summary(multi.var.model))

Call:
lm(formula = frml, data = training.data)

Residuals:
     Min       1Q   Median       3Q      Max
-14.6682  -2.8641  -0.6328   1.7493  26.6253

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  38.704860   5.493485   7.046 8.41e-12 ***
CRIM         -0.101555   0.033725  -3.011  0.00277 **
```

```
ZN             0.039328   0.015274   2.575  0.01040 *  
INDUS          0.035499   0.067568   0.525  0.59961    
CHASTRUE       2.435641   0.910881   2.674  0.00781 ** 
NOX          -17.757531   4.182255  -4.246 2.72e-05 ***
RM             3.691904   0.447782   8.245 2.55e-15 ***
AGE           -0.002027   0.014481  -0.140  0.88875    
DIS           -1.332287   0.221490  -6.015 4.14e-09 ***
RAD            0.295501   0.073359   4.028 6.76e-05 ***
TAX           -0.011194   0.004184  -2.676  0.00777 ** 
PTRATIO       -1.072440   0.148331  -7.230 2.57e-12 ***
B              0.008018   0.003053   2.626  0.00898 ** 
LSTAT         -0.498635   0.055112  -9.048  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.687 on 390 degrees of freedom
Multiple R-squared:  0.745,	Adjusted R-squared:  0.7365 
F-statistic: 87.65 on 13 and 390 DF,  p-value: < 2.2e-16

> 
> multi.var.predictions <- predict(multi.var.model, testing.data)
> test.multi.var.ssl <- sum((testing.data$CRIM - multi.var.predictions)^2)
> sprintf("SSL/SSR/SSE: %f", test.multi.var.ssl)
[1] "SSL/SSR/SSE: 52806.704419"
> test.multi.var.mse <- test.multi.var.ssl / nrow(testing.data)
> sprintf("MSE: %f", test.multi.var.mse)
[1] "MSE: 517.712788"
> test.multi.var.rmse <- sqrt(test.multi.var.mse)
> sprintf("RMSE: %f", test.multi.var.rmse)
[1] "RMSE: 22.753303"
> 
> #effect_plot(multi.var.predictions, pred = MEDV, interval = TRUE, plot.poin
ts = TRUE)
> 
> full.model <- lm(frml, data = housing)
> print(summary(full.model))

Call:
lm(formula = frml, data = housing)

Residuals:
    Min      1Q  Median      3Q     Max 
-15.595  -2.730  -0.518   1.777  26.199 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  3.646e+01  5.103e+00   7.144 3.28e-12 ***
CRIM        -1.080e-01  3.286e-02  -3.287 0.001087 ** 
ZN           4.642e-02  1.373e-02   3.382 0.000778 ***
INDUS        2.056e-02  6.150e-02   0.334 0.738288    
CHASTRUE     2.687e+00  8.616e-01   3.118 0.001925 ** 
NOX         -1.777e+01  3.820e+00  -4.651 4.25e-06 ***
RM           3.810e+00  4.179e-01   9.116  < 2e-16 ***
AGE          6.922e-04  1.321e-02   0.052 0.958229    
DIS         -1.476e+00  1.995e-01  -7.398 6.01e-13 ***
RAD          3.060e-01  6.635e-02   4.613 5.07e-06 ***
TAX         -1.233e-02  3.760e-03  -3.280 0.001112 ** 
PTRATIO     -9.527e-01  1.308e-01  -7.283 1.31e-12 ***
B            9.312e-03  2.686e-03   3.467 0.000573 ***
LSTAT       -5.248e-01  5.072e-02 -10.347  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
Residual standard error: 4.745 on 492 degrees of freedom
Multiple R-squared:  0.7406,	Adjusted R-squared:  0.7338 
F-statistic: 108.1 on 13 and 492 DF,  p-value: < 2.2e-16

> 
> #testing data plots
> scatter.smooth(x = testing.data$ZN, y = testing.data$MEDV, main="MEDV ~ ZN"
)
There were 35 warnings (use warnings() to see them)
> scatter.smooth(x = testing.data$INDUS, y = testing.data$MEDV, main="MEDV ~ 
INDUS")
> scatter.smooth(x = testing.data$CHAS, y = testing.data$MEDV, main="MEDV ~ C
HAS")
There were 40 warnings (use warnings() to see them)
> scatter.smooth(x = testing.data$NOX, y = testing.data$MEDV, main="MEDV ~ NO
X")
> scatter.smooth(x = testing.data$RM, y = testing.data$MEDV, main="MEDV ~ RM"
)
> scatter.smooth(x = testing.data$AGE, y = testing.data$MEDV, main="MEDV ~ AG
E")
> scatter.smooth(x = testing.data$DIS, y = testing.data$MEDV, main="MEDV ~ DI
S")
> scatter.smooth(x = testing.data$RAD, y = testing.data$MEDV, main="MEDV ~ RA
D")
> scatter.smooth(x = testing.data$TAX, y = testing.data$MEDV, main="MEDV ~ TA
X")
> scatter.smooth(x = testing.data$PTRATIO, y = testing.data$MEDV, main="MEDV 
~ PTRATIO")
> scatter.smooth(x = testing.data$B, y = testing.data$MEDV, main="MEDV ~ B")
> scatter.smooth(x = testing.data$LSTAT, y = testing.data$MEDV, main="MEDV ~ 
LSTAT")
> scatter.smooth(x = testing.data$CRIM, y = testing.data$MEDV, main="MEDV ~ C
RIM")
> 
> 
> 
> #ridge regression
> housing.mat <- model.matrix(MEDV ~ ., data = housing)
> housing.mat <- housing.mat[,-13]
> 
> X <- housing.mat
> Y <- housing[,'MEDV']
> 
> 
> 
> X.train <- X[training.indices,]
> X.test <- X[-training.indices,]
> Y.train <- Y[training.indices]
> Y.test <- Y[-training.indices]
> 
> linear.mod <- lm(Y.train ~ X.train)
> #summary(linear.mod)
> 
> lm.pred <- predict(linear.mod, newx = X.test)
> rmse <- sqrt(mean((lm.pred - Y.test)^2))
Warning message:
In lm.pred - Y.test :
  longer object length is not a multiple of shorter object length
> sprintf("RMSE: %f", rmse)
[1] "RMSE: 12.429801"
> 
> grid <- 10^seq(10, -2, length=1000)
> ridge.mod <- glmnet(X.train, Y.train, alpha=0, lambda=grid, thresh=1e-12)
```

```
> cv.out <- cv.glmnet(X.train, Y.train, alpha=0, nfolds = 10)
> plot(cv.out)
>
> best.lambda <- cv.out$lambda.min
> sprintf("Ridge Best Lambda: %f", best.lambda)
[1] "Ridge Best Lambda: 0.741674"
>
> ridge.pred <- predict(ridge.mod, s=best.lambda, newx = X.test)
> print(paste('RMSE:', sqrt(mean((ridge.pred - Y.test)^2))))
[1] "RMSE: 5.32333085023029"
> #ridge.pred
>
> ridge.out <- glmnet(X, Y, alpha = 0)
> plot(ridge.out, xvar = "lambda", ylim = c(-10, 10))
>
> ridge.results <- predict(ridge.out, type = "coefficients", s=best.lambda)
> ridge.results
14 x 1 sparse Matrix of class "dgCMatrix"
                        1
(Intercept)  32.262004014
(Intercept)   .
CRIM         -0.099228668
ZN            0.032089861
INDUS        -0.046346825
CHASTRUE      3.044015331
NOX         -12.361675962
RM            3.906808828
AGE          -0.002168196
DIS          -1.101591472
RAD           0.132836240
TAX          -0.005960766
PTRATIO      -0.836374783
LSTAT        -0.489658705
>
> #LASSO Regression model
> lasso.mod <- glmnet(X.train, Y.train)
> plot(lasso.mod, xvar="lambda")
>
> lasso.cv.out <- cv.glmnet(X, Y, alpha=1, nfolds = 10)
> plot(lasso.cv.out)
>
> lasso.mod.pred <- predict(lasso.mod, X.test)
> rmse <- sqrt(apply((lasso.mod.pred - Y.test)^2, 2, mean))
> lasso.best.lambda <- lasso.cv.out$lambda.1se
> print(lasso.best.lambda)
[1] 0.3789258
>
> lasso.pred <- predict(lasso.mod, s=lasso.best.lambda, newx=X.test)
> print(paste('RMSE:', sqrt(mean((lasso.pred - Y.test)^2))))
[1] "RMSE: 5.6600934930813"
>
> lasso.best.lambda
[1] 0.3789258
>
> lasso.out <- glmnet(X, Y, alpha=1, lambda=grid)
> lasso.coef <- predict(lasso.out, type="coefficients", s=lasso.best.lambda)
> lasso.coef
14 x 1 sparse Matrix of class "dgCMatrix"
                        1
(Intercept) 21.6834491048
(Intercept)  .
CRIM        -0.0361155884
ZN           .
```

```
INDUS          .
CHASTRUE      2.0058948525
NOX          -4.4629140052
RM            4.1558518545
AGE            .
DIS          -0.3357429179
RAD            .
TAX          -0.0002192817
PTRATIO      -0.7942328505
LSTAT        -0.5358125716
```