Sectional Project 1 - Regression

Group 3

2/20/2021

Group Leader: Brianna Johnson

Member Names: Marco Sousa, Nikita Seleznev, Ben Pfeffer, Brianna Johnson

Introduction to the Boston Housing Dataset

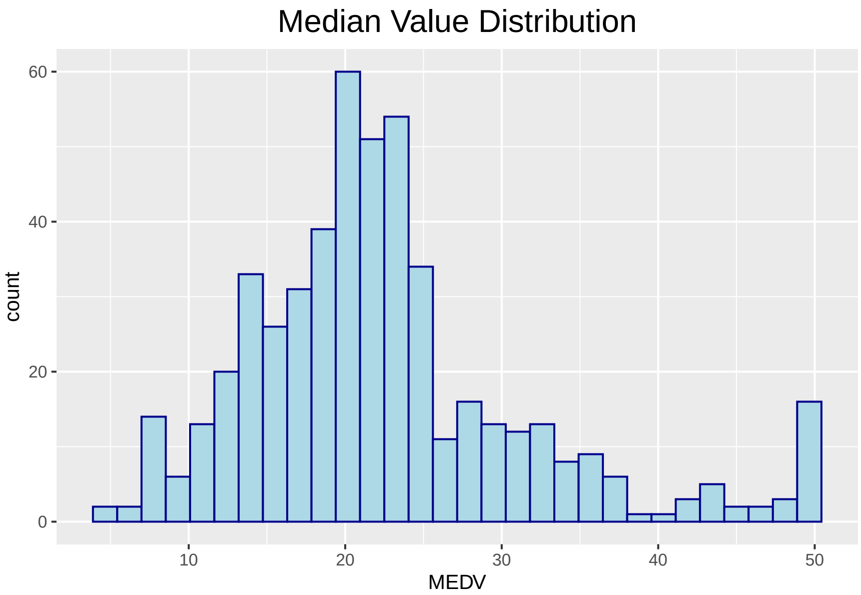
The Boston Housing dataset considers housing values and their associated properties in suburbs of Boston, Massachusetts. The dataset contains 506 observations and 14 attributes. We acquired the dataset from the Machine Learning Database (MLDB), found here. In particular, we are interested in constructing a model through regression techniques to gain insight on housing values. As such, we will use the 13 features to model 'MEDV', the median value of owner occupied homes (in $1,000s).

The data is displayed as follows.

```
##      CRIM ZN INDUS CHAS   NOX    RM  AGE    DIS RAD TAX PTRATIO      B LSTAT
## 1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296    15.3 396.90  4.98
## 2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17.8 396.90  9.14
## 3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17.8 392.83  4.03
## 4 0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222    18.7 394.63  2.94
## 5 0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222    18.7 396.90  5.33
## 6 0.02985  0  2.18    0 0.458 6.430 58.7 6.0622   3 222    18.7 394.12  5.21
##   MEDV
## 1 24.0
## 2 21.6
## 3 34.7
## 4 33.4
## 5 36.2
## 6 28.7
```

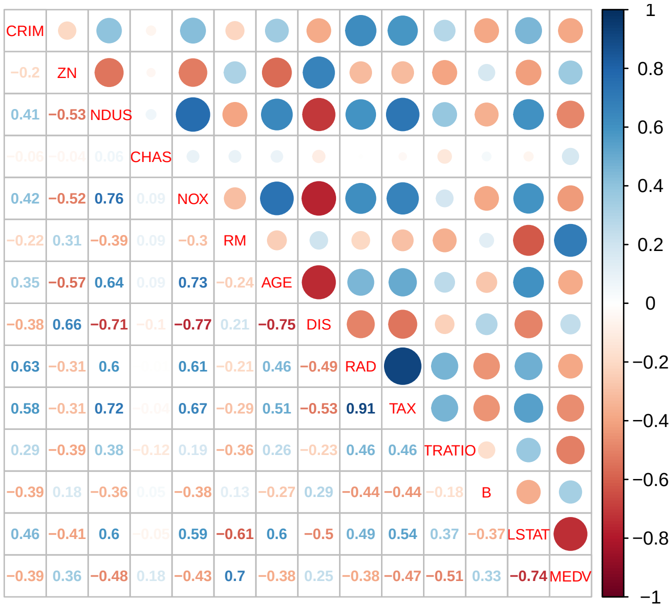
Exploratory Data Analysis

MEDV Distribution



The histogram demonstrates the values are not uniformly distributed. Rather, they follow a mostly normal distributions, with some outliers at the tail.

Correlation Matrix



While we produce many correlation values, we are firstly interested in how each attribute correlates to MEDV. This is represented by the bottom row or last column. We can immediately see the binary CHAS attribute does not correlate strongly with MEDV. However, it can be seen that RM (0.7) and LSTAT (-0.74) correlate with MEDV stronger than other attributes. Furthermore, the correlation between RM and LSTAT is -0.61. Since they do not correlate very strongly with one another, we can select both as predictor attributes without too much concern of collinearity for their case. The greatest correlation is between RAD and TAX of 0.91, and an R^2 of 0.82 thusly. Including both of these may raise some concerns regarding the minimal collinearity assumption of linear regression.

Modelling and Regression

MedV takes the value for Y, along 13 feature attributes of the dataset, in the form of $Y = \beta_0+\beta_1 x_1+...+\beta_n x_n$ + b.

Multiple Linear Regression

We chose to simply split the data in a 60/40 split for multiple Linear Regression.

```
#Setting seed for reproducible results
set.seed(1)

indices <-sample(1:nrow(data), 0.6 * nrow(data), replace = TRUE)
training <-data[indices,]
testing  <-data[-indices,]
```

Naively consider all features except the binary at onset for a MLR.

```
##
## Call:
## lm(formula = MEDV ~ CRIM + ZN + INDUS + NOX + RM + AGE + DIS +
##     RAD + TAX + PTRATIO + B + LSTAT, data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.7529  -2.3580  -0.3601   1.4662  25.0880
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 40.607269   6.933545   5.857 1.28e-08 ***
## CRIM        -0.117088   0.036419  -3.215  0.00145 **
## ZN           0.046371   0.016641   2.787  0.00568 **
## INDUS        0.098641   0.071828   1.373  0.17072
## NOX        -16.550316   5.088089  -3.253  0.00128 **
## RM           3.280962   0.577424   5.682 3.24e-08 ***
## AGE          0.013777   0.016774   0.821  0.41214
## DIS         -1.216793   0.250349  -4.860 1.92e-06 ***
## RAD          0.494834   0.078440   6.308 1.05e-09 ***
## TAX         -0.020556   0.004329  -4.749 3.22e-06 ***
## PTRATIO     -0.948615   0.160907  -5.895 1.04e-08 ***
## B            0.005826   0.003561   1.636  0.10289
## LSTAT       -0.635492   0.068081  -9.334  < 2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 4.596 on 290 degrees of freedom
## Multiple R-squared:  0.7209, Adjusted R-squared:  0.7094
## F-statistic: 62.42 on 12 and 290 DF,  p-value: < 2.2e-16
```

Some initial insight is that LSTAT and RM indeed were strong predictors. Removing INDUS, AGE, and B, every attribute becomes a significant predictor, depending on alpha. Let's consider what happens if we remove RAD, which varies strongly with TAX.

```
##
## Call:
## lm(formula = MEDV ~ CRIM + ZN + NOX + RM + DIS + TAX + PTRATIO +
##     LSTAT, data = training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -11.554  -2.727  -0.498   1.599  26.846
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.163e+01  6.530e+00   4.844 2.06e-06 ***
## CRIM        -7.328e-02  3.588e-02  -2.043   0.0420 *
## ZN           4.281e-02  1.755e-02   2.439   0.0153 *
## NOX         -1.114e+01  4.952e+00  -2.249   0.0252 *
## RM           3.799e+00  5.786e-01   6.566 2.34e-10 ***
## DIS         -1.247e+00  2.442e-01  -5.108 5.86e-07 ***
## TAX          5.472e-04  2.702e-03   0.203   0.8397
## PTRATIO     -8.010e-01  1.663e-01  -4.816 2.35e-06 ***
## LSTAT       -6.221e-01  6.418e-02  -9.693  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.886 on 294 degrees of freedom
## Multiple R-squared:  0.6802, Adjusted R-squared:  0.6715
## F-statistic: 78.16 on 8 and 294 DF,  p-value: < 2.2e-16
```

We can see that without RAD, TAX is no longer a strong predictor. As such, TAX adds predictive value in relation to RAD. The next model removes TAX and adds RAD back in.

```
##
## Call:
## lm(formula = MEDV ~ CRIM + ZN + NOX + RM + DIS + RAD + PTRATIO +
##     LSTAT, data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.1059  -2.7763  -0.2535   1.3876  25.2754
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  39.92696    6.64060   6.013 5.41e-09 ***
## CRIM         -0.12935    0.03657  -3.537 0.000471 ***
## ZN            0.03885    0.01712   2.270 0.023954 *
## NOX         -18.55609    4.73339  -3.920 0.000110 ***
## RM            3.51728    0.56977   6.173 2.22e-09 ***
## DIS          -1.31761    0.23908  -5.511 7.80e-08 ***
## RAD           0.18688    0.04921   3.797 0.000178 ***
## PTRATIO      -0.97901    0.16325  -5.997 5.89e-09 ***
## LSTAT        -0.63314    0.06259 -10.116  < 2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 4.771 on 294 degrees of freedom
## Multiple R-squared:  0.6951, Adjusted R-squared:  0.6868
## F-statistic: 83.77 on 8 and 294 DF,  p-value: < 2.2e-16
```

MSE and RMSE for model 1 and 3.

```
multiPredictions <-predict(multiModel1, testing)
RSS <- sum((testing$MEDV - multiPredictions)^2)
MSE1 <- mean((testing$MEDV - multiPredictions)^2)
RMSE1 <- sqrt(MSE1)

MSE1
```

```
## [1] 22.55438
```

```
RMSE1
```

```
## [1] 4.749145
```

```
multiPredictions <-predict(multiModel3, testing)
RSS3 <- sum((testing$MEDV - multiPredictions)^2)
MSE3 <- mean((testing$MEDV - multiPredictions)^2)
RMSE3 <- sqrt(MSE3)

MSE3
```

```
## [1] 23.02552
```

```
RMSE3
```

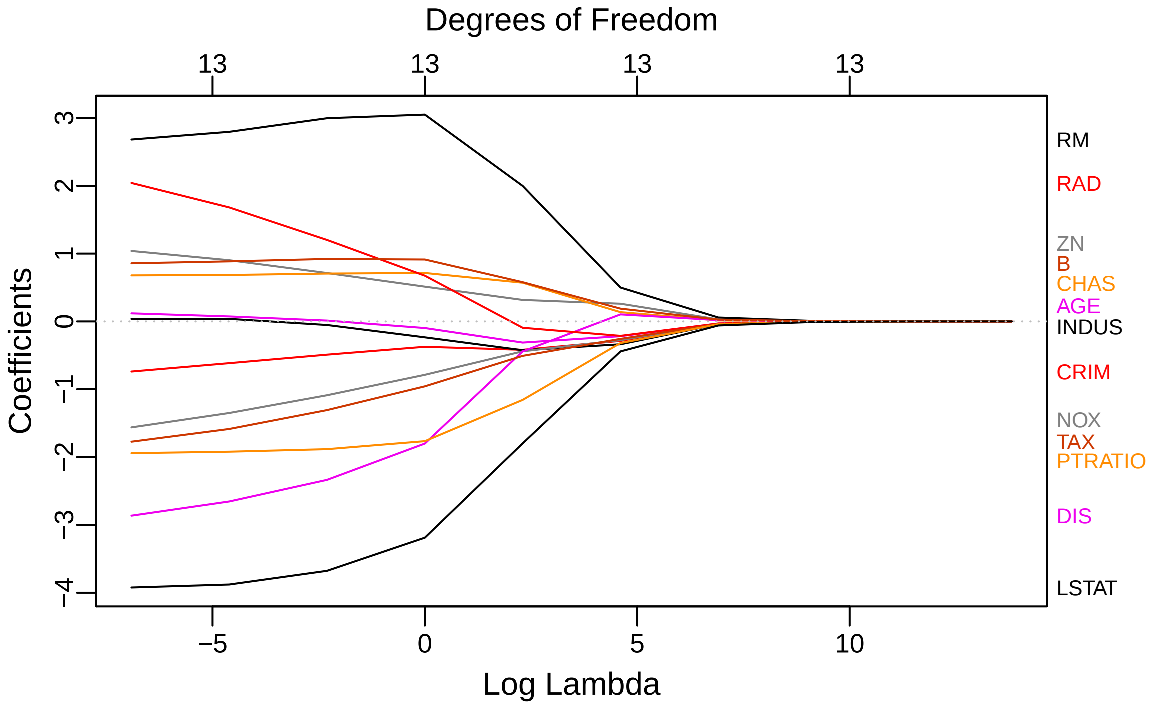
```
## [1] 4.798491
```

It seems we are left with 8 features. Lasso and Ridge will give us further insight into parameter selection.

Ridge Regression

Constructing a Ridge model.

```
#lambda grid
grid <- 10^seq(6, -3, length=10)
#ridgeModel
ridge.mod <- glmnet(scale(x), y, alpha = 0, lambda = grid, thresh = 1e-2, standardize = TRUE)
#Plotting the ridge.mod
plot_glmnet(ridge.mod, xvar = "lambda", label = 13)
```

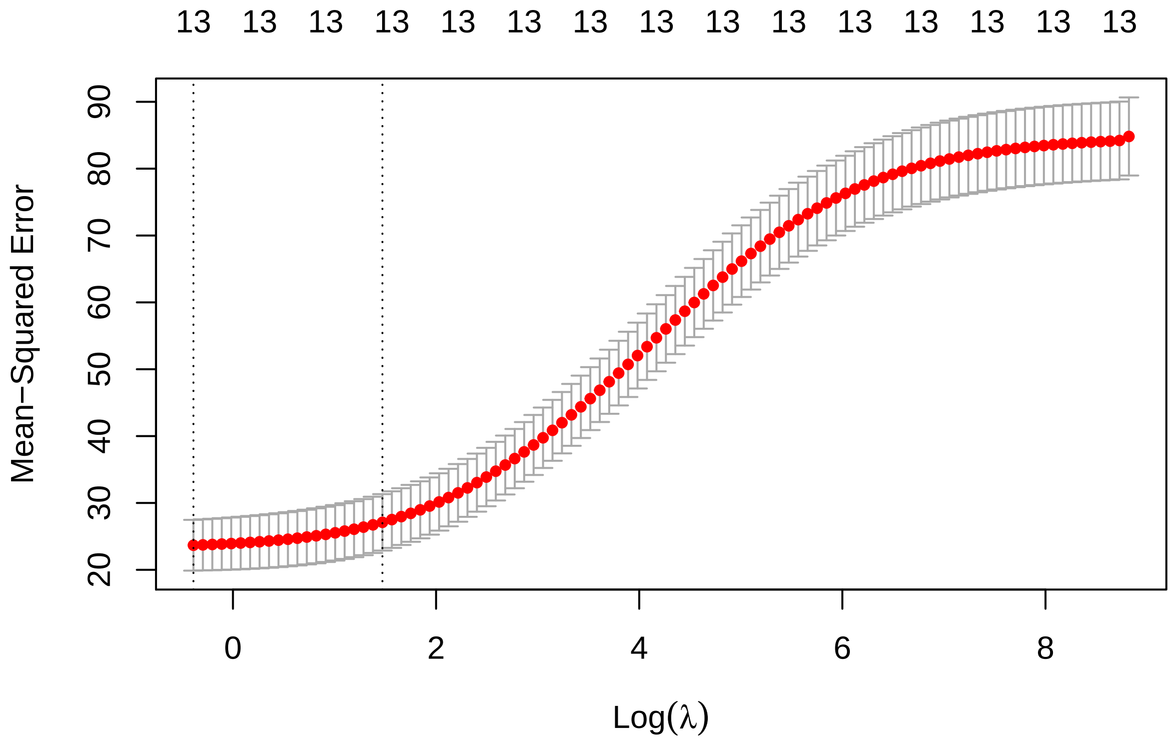


Considering a best lambda for the model and hyperparameter tuning.

```
cv.out <- cv.glmnet(x, y, alpha=0, nfolds = 10)
cv.out
```

```
##
## Call:  cv.glmnet(x = x, y = y, nfolds = 10, alpha = 0)
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure    SE Nonzero
## min  0.678   100   23.68 3.797      13
## 1se  4.357    80   27.10 4.220      13
```

```
plot(cv.out)
```



```
best.lambda <- cv.out$lambda.min
best.lambda
```

```
## [1] 0.6777654
```

Considering coefficients for Ridge.

```
#Viewing coefficients of scaled full ridge model
predict(ridge.mod, type="coefficients", s=best.lambda)
```

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept) 22.53280632
## CRIM        -0.41568230
## ZN           0.58454162
## INDUS       -0.16968729
## CHAS         0.71113369
## NOX         -0.89496390
## RM           3.03015102
## AGE         -0.05816081
## DIS         -1.99196163
## RAD          0.86276996
## TAX         -1.08247269
## PTRATIO     -1.80779900
## B            0.91568362
## LSTAT       -3.36321495
```

```
#Selecting above abs(1): NOX, DIS, PTRATIO, LSTAT, RM
newX = data.mat[, c(5,6,8,11,13)]
head(newX)
```

```
##     NOX    RM    DIS PTRATIO LSTAT
## 1 0.538 6.575 4.0900    15.3  4.98
## 2 0.469 6.421 4.9671    17.8  9.14
## 3 0.469 7.185 4.9671    17.8  4.03
## 4 0.458 6.998 6.0622    18.7  2.94
## 5 0.458 7.147 6.0622    18.7  5.33
## 6 0.458 6.430 6.0622    18.7  5.21
```

```
#Final ridge with all coefficients
ridge.final1 <- glmnet(x, y, alpha = 0, lambda = best.lambda, thresh=1e-2, standardsize = TRUE)
predict(ridge.final1, type="coefficients", s=best.lambda)
```

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept)  24.811787388
## CRIM         -0.051540889
## ZN            0.025444744
## INDUS        -0.111469005
## CHAS          2.950592527
## NOX         -12.202495338
## RM            4.572000162
## AGE          -0.011122797
## DIS          -1.212875203
## RAD           0.096752050
## TAX          -0.004456126
## PTRATIO      -0.836004412
## B             0.009830260
## LSTAT        -0.405092069
```

```
#Final ridge with only 5 coefficients
ridge.final2 <- glmnet(newX, y, alpha = 0, lambda = best.lambda, thresh=1e-2, standardsize = TRUE)
predict(ridge.final2, type="coefficients", s=best.lambda)
```

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept)  29.4519872
## NOX         -12.6387422
## RM            4.5085207
## DIS          -0.7280733
## PTRATIO      -1.0318123
## LSTAT        -0.5087480
```

Considering MSE and RMSE for Ridge using all coefficients.

```
ridge.pred1 <- predict(ridge.final1, s=best.lambda, newx=x)
ridge.MSE = mean((ridge.pred1 -y)^2)
ridge.RMSE = sqrt(mean((ridge.pred1 - y)^2))

ridge.MSE
```

```
## [1] 22.89447
```

```
ridge.RMSE
```

```
## [1] 4.784817
```

Considering MSE and RMSE for Ridge using only 5 coefficients.

```
ridge.pred2 <- predict(ridge.final2, s=best.lambda, newx=newX)
ridge.MSE = mean((ridge.pred2 -y)^2)
ridge.RMSE = sqrt(mean((ridge.pred2 - y)^2))

ridge.MSE
```

```
## [1] 25.09644
```

```
ridge.RMSE
```

```
## [1] 5.009635
```

Considering R squared for each model manually

```
yBar = mean(y)
RSS1 = sum((ridge.pred1 - y)^2)
TSS1 = sum((y - yBar)^2)
rsq1 = 1 - (RSS1/TSS1)
rsq1
```

```
## [1] 0.7288013
```

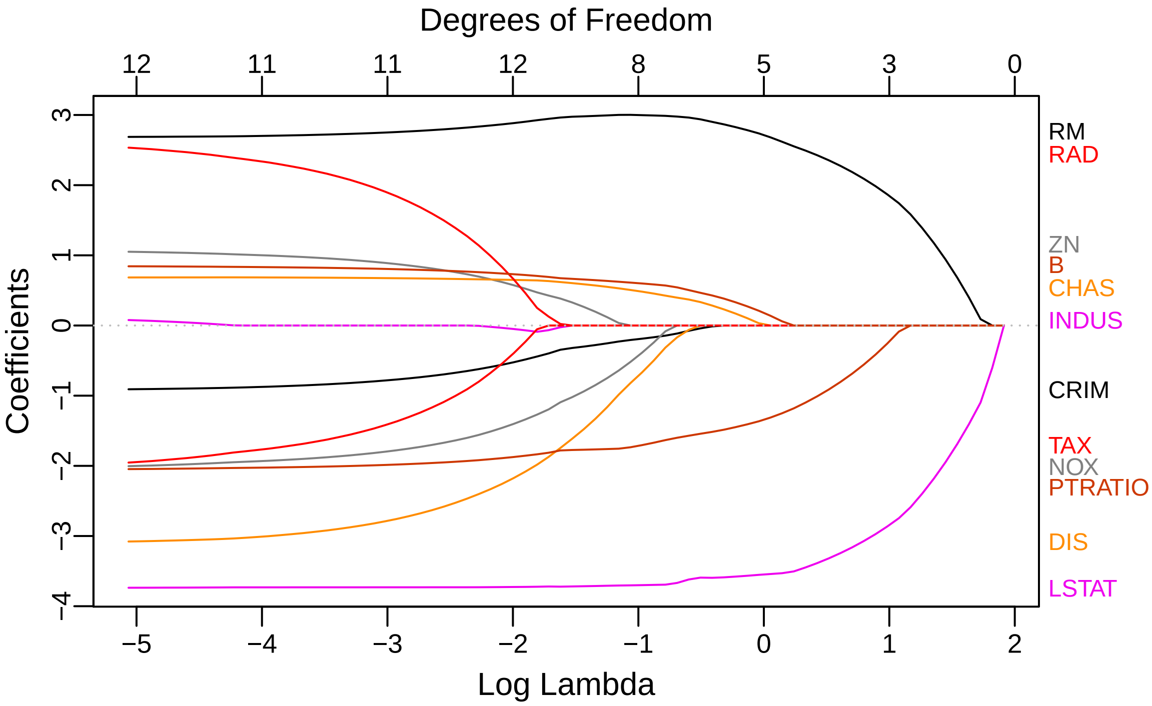
```
RSS2 = sum((ridge.pred2 - y)^2)
TSS2 = sum((y - yBar)^2)
rsq2 = 1 - (RSS2/TSS2)
rsq2
```

```
## [1] 0.7027177
```

Lasso Regression

The Lasso regression will allow us to conduct selection of the most influential independent variables to predict our dependent variable, which is the house price.
Again, we will use the same range of the regularization parameter, λ, values, but will sample this range more densely as it proved to provide better results for the parameter selection. The output of the Lasso regression for various values of λ are given below. It is evident that with increasing regularization parameter the number of non-zero regressors is decreasing.

```
grid <- 10^seq(6, -3, length=100)
lasso.mod <- glmnet(scale(x), y) #default alpha=1
plot_glmnet(lasso.mod, xvar="lambda", label = 12)
```

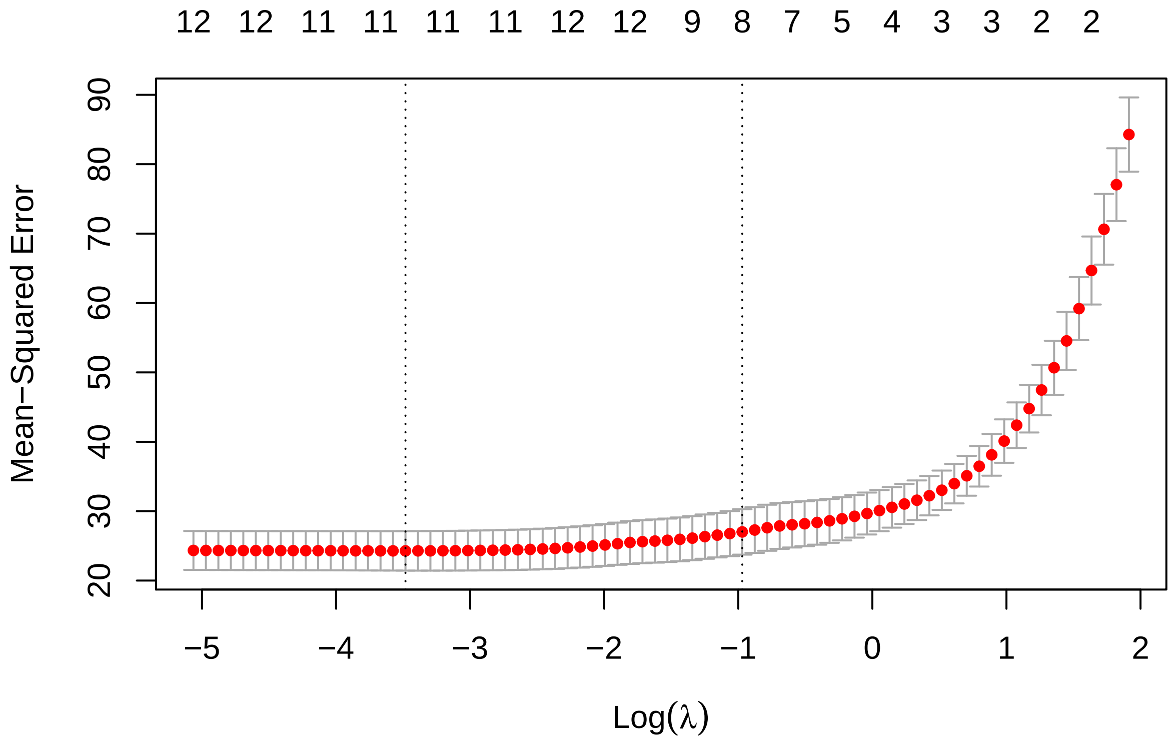


Now we will conduct 10-fold cross-validation to select optimum regularization parameter lambda. The minimum MSE is produced for $\lambda = 0.0307$, which keeps 11 non-zero regressors. Increasing λ to 0.3789 reduces the number fo regressors to 8.

```
##cross-validation
lasso.cv.out <- cv.glmnet(scale(x), y, alpha=1, nfolds = 10)
lasso.cv.out
```

```
##
## Call:  cv.glmnet(x = scale(x), y = y, nfolds = 10, alpha = 1)
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure    SE Nonzero
## min 0.0307    59   24.28 2.844      11
## 1se 0.3789    32   27.01 3.275       8
```

```
plot(lasso.cv.out)
```



Now we will predict the coefficients of the Lasso regression for 11 non-zero predictors. Also, we will plot the predicted home price versus the true home prices and calculate the MSE and RMSE errors to check how well our mode performs.

```
#keep few predictors in lasso
lasso.best.lambda <- lasso.cv.out$lambda[which.max(lasso.cv.out$nzero == 11)]
#lasso.final <- glmnet(scale(x), y, alpha=1, lambda=grid)
lasso.final <- glmnet(x, y, alpha=1, lambda=grid)
predict(lasso.final, type="coefficients", s=lasso.best.lambda )
```

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept) 23.512082531
## CRIM        -0.041774090
## ZN           0.016232882
## INDUS       -0.005184322
## CHAS         2.437746370
## NOX         -9.548458409
## RM           4.212602160
## AGE          .
## DIS         -0.829159065
## RAD          0.007102791
## TAX          .
## PTRATIO     -0.828101442
## B            0.007433893
## LSTAT       -0.520933847
```

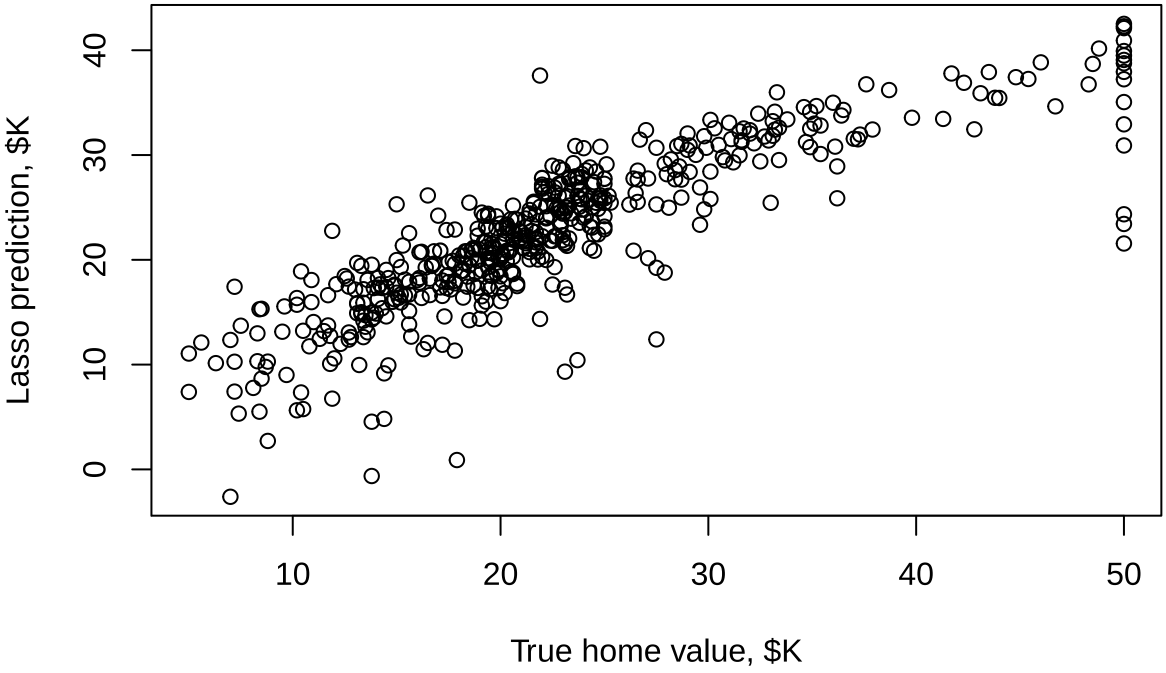
```
lasso.pred <- predict(lasso.final, s=lasso.best.lambda, newx=x)
print(paste('MSE:', mean((lasso.pred - y)^2)))
```

```
## [1] "MSE: 23.4455599763479"
```

```
print(paste('RMSE:', sqrt(mean((lasso.pred - y)^2))))
```

```
## [1] "RMSE: 4.84206154198271"
```

```
plot(y, lasso.pred,  xlab="True home value, $K", ylab="Lasso prediction, $K",)
```



Considering the R squared for the 11 selector Lasso model.

```
yBar = mean(y)
lassoRSS = sum((lasso.pred - y)^2)
lassoTSS = sum((y - yBar)^2)
rsq = 1 - (lassoRSS/lassoTSS)
rsq
```

```
## [1] 0.7222734
```

Now we will further reduced the number of regressors to keep only 5 most important ones. Below are the coefficients of the Lasso regression and plot of the predicted home price versus the true home prices. We also provide the MSE and RMSE errors. As expected, the scatter of the plot and the MSE/RMSE errors are slightly higher then for the case of 11 regressors, but our model is still doing a decent job at predicting the home values based on half the number of regressors.

```
# check for fewer predictors
lasso.best.lambda <- lasso.cv.out$lambda[which.max(lasso.cv.out$nzero == 5)]
lasso.final <- glmnet(x, y, alpha=1, lambda=grid)
predict(lasso.final, type="coefficients", s=lasso.best.lambda )
```

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept) 15.156492728
## CRIM         .
## ZN           .
## INDUS        .
## CHAS         0.130133129
## NOX          .
## RM           3.895113886
## AGE          .
## DIS          .
## RAD          .
## TAX          .
## PTRATIO     -0.630418762
## B            0.002297575
## LSTAT       -0.497652218
```

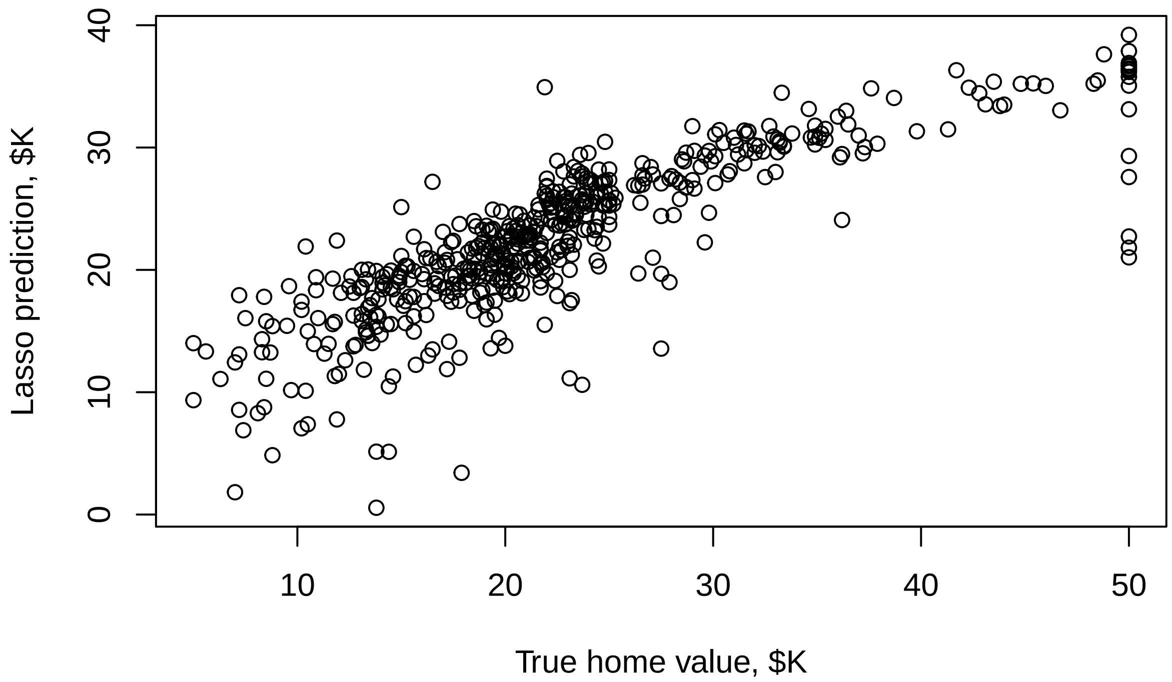
```
lasso.pred <- predict(lasso.final, s=lasso.best.lambda, newx=x)
print(paste('MSE:', mean((lasso.pred - y)^2)))
```

```
## [1] "MSE: 28.2507097708968"
```

```
print(paste('RMSE:', sqrt(mean((lasso.pred - y)^2))))
```

```
## [1] "RMSE: 5.31513967557738"
```

```
plot(y, lasso.pred, xlab="True home value, $K", ylab="Lasso prediction, $K",)
```



Considering the R squared for the 5 selector Lasso model.

```
yBar = mean(y)
lassoRSS = sum((lasso.pred - y)^2)
lassoTSS = sum((y - yBar)^2)
rsq = 1 - (lassoRSS/lassoTSS)
rsq
```

```
## [1] 0.6653535
```

K-Fold Cross Validation

In a standard approach a part fo the data is reserved as a test set and the rest of it is used as a training set. This can lead to a highly variable test error depending on how the dataset was split. Moreover, the machine learning approaches tend to perform worse when less data are used for training, which happens when the training set is only a part of the entire dataset.

We are employing a cross-validation (CV) technique as it provides helps to alleviate the above problems. Specifically, we use a K-fold cross validation. In this approach one fold is used as a test set while the remaining K-1 folds are used as the training set. After one of the folds is tested the algorithm moves to the other one. Thus, every data point in the set is a part of the training and the test set for some folds. The final K-fold CV mean squared error (MSE) error is computed as the mean of the MSE of individual folds.

In practice K = 5 or 10 tend to provide the best results as they yield a test error that does not suffer from excessively high bias, nor from very high variance. We chose to use 10-fold cross-validation approach (K = 10).

The 10-fold CV MSE is computed for each value of λ in the predefined grid. Finally, we chose the λ value that minimizes 10-fold CV MSE. For Lasso regression we also test other values of λ to select the most important regressors.

The final MSE's, RMSE's, and R^2, achieved by different regression approaches are given in the table below. The number following the method indicates the number of included features.

	MSE	RMSE	R^2
Multiple 12	22.554	4.749	0.720
Multiple 8	23.025	4.798	0.695
Ridge 13	22.894	4.784	0.728
Ridge 5	25.096	5.009	0.702
Lasso 11	23.445	4.842	0.722
Lasso 5	28.250	5.315	0.665

Having more independent variables as an input to regression provides overall smaller MSE and RMSE errors. However, the difference in errors between Ridge regression and Lasso regression with 11 input variables is small. Furthermore, Lasso regression with only 5 most significant regressors still yields RMSE error, which is only 9.8 % higher compared to the optimum result. Thus, Lasso enables significant reduction in the dataset requirements without loosing much of the prediction accuracy.

Summary

The goal was to modell MEDV, the median value of owner occupied homes (in $1,000s), based on given features. MEDV follows a skewed normal distribution with some outliers at its tail. Three regression techniques were performed to model the data: multiple linear regression, Ridge, and Lasso. It was observed that TAX only became a predictor in MLR when RAD was included in, so TAX was removed. Removing insignificant predictors (to alpha) left 8 features. Ridge was performed using all features. Then Ridge was modified to include the 5 greatest predictors. This was determine by taking the greatest absolute value of the coefficients when input was scaled. Lasso was performed with 11 then 5 predictors. Among all methods, LSTAT, RM, and PTRatio remained as significant predictors. Having more independent variables as an input to regression provides overall smaller MSE and RMSE errors. However, the difference in errors between Multiple Ridge and Lasso, each with mostly all input variables (11, 12, or 13), was small. This changed by different amounts when hyperparamater tuning took place.

Citations

[1] Fang, Julia. "CIS490_LS8_21S_LassoReg&CrossValidation." MyCourses, 2021, linked umassd pdf

[2] Fang, Julia. "R_RidgeLassoCV_Final.pdf" MyCourses, 2021, linked umassd pdf.

[3]Index of /Ml/Machine-Learning-Databases/Housing, archive.ics.uci.edu/ml/machine-learning-databases/housing/.