STA363_Project_2

Zishan Shao

2022-10-20

```
knitr::opts_chunk$set(echo = TRUE)
set.seed(1)
```

```
# load the data from the dataset
data <- read.csv("../Project_2/VRBO.csv", header = TRUE)
```

Section 1: Introduction

Section 2: Data Cleaning

Data Loss Table

```
# explore the number of observations before cleaning
a = nrow(data)
# explore the number of observations after cleaning
b = nrow(na.omit(data))
# total removed observations
c = a - b

# percentage of omitted rows to the total rows
d = c / a * 100

# construct a table for the data losses
aa <- round(c(a,b,c,d), 2)
names <- c("Original", "Cleaned", "Total Removal", "Percent Data Loss")

hahaloss <- data.frame("Names" = names, "Count" = aa)

knitr::kable(hahaloss, caption = "Data Loss Table (before & after cleaning)")
```

Table 1: Data Loss Table (before & after cleaning)

Names	Count
Original	1561.00
Cleaned	1423.00
Total Removal	138.00

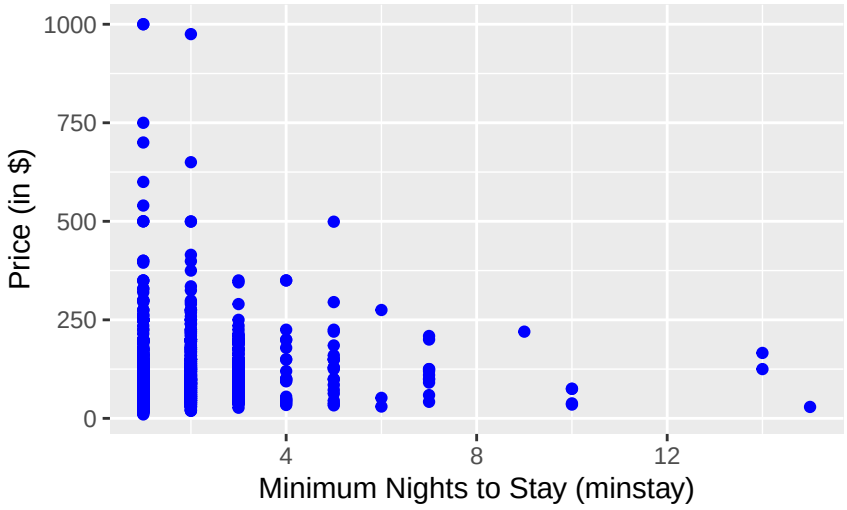
Names	Count
Percent Data Loss	8.84

```
# explore the number of null values in the minstay
# length(which(is.na(data$minstay)))

# find the total number of residual
e = nrow(na.omit(data[,-8])) # the num observation remain same as uncleaned data

# table for cleaned data after removing minstay
nomloss <- data.frame("Orignal" = a, "No Minstay" = e)

knitr::kable(nomloss, caption = "Number of Observations (with/not with minstay)")
```

Table 2: Number of Observations (with/not with minstay)

Orignal	No.Minstay
1561	1561

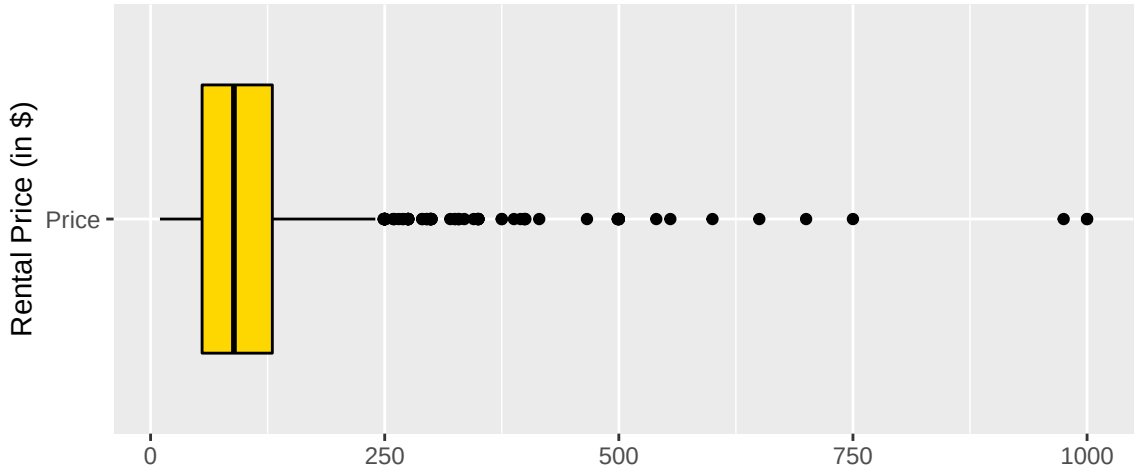
```
# import the ggplot2 pacakge
suppressMessages(library(ggplot2))
# create a scatterplot to study the relationship between minstay and price
ggplot(na.omit(data), aes(minstay, price)) + geom_point(color = "blue")+ labs(title =" ", x = "Minimum
```



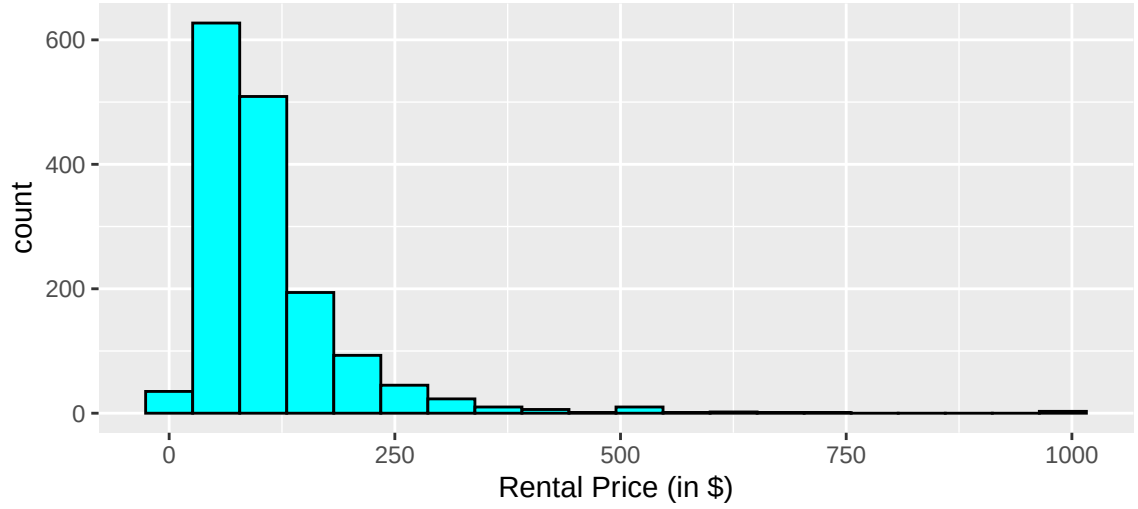
Distribution of Response Variable

```
# create the design matrix XD by removing the UnitNumber and the response variable Price
dataClean <- data[,-c(1,8)]
# head(dataClean)
```

```r
# import the ggplot2 package
library(ggplot2)

# cluster of mosaic plots
op <- par( mfrow = c(1,2))

# see if there is unreasonable observation
ggplot(dataClean,aes(x ="Price", y = price)) + geom_boxplot(fill='gold', col = 'black') + labs(title=""
```




```r
# consider that the response variable Y is a continuous data, we are
# expected to use a histogram to visualize its distribution.
library(ggplot2)
ggplot(dataClean, aes(x=price)) + geom_histogram(fill='cyan', col = 'black', bins = 20) + labs(title=""
```



```
par(op)
```

Section 3: LSLR & ridge Regression

$$LSLR : \hat{\beta_{LS}} = (X_D^T X_D)^{-1} X_D^T Y$$

$$Ridge : \hat{\beta_{Ridge}} = (X_D^T X_D + \lambda_{ridge} I)^{-1} X_D^T Y$$

Correlation Plot

```
# import necessary libraries
suppressMessages(library(corrplot))
suppressMessages(library(RColorBrewer))

# Explore the correlation between features
M <-cor(dataClean[,-c(1, 4, 7, 8)])
colnames(M)[1] <- "satisfaction"
rownames(M)[1] <- "satisfaction"
corrplot(M, method="circle", type = "upper", title = "", mar = c(0,0,1,0))
```

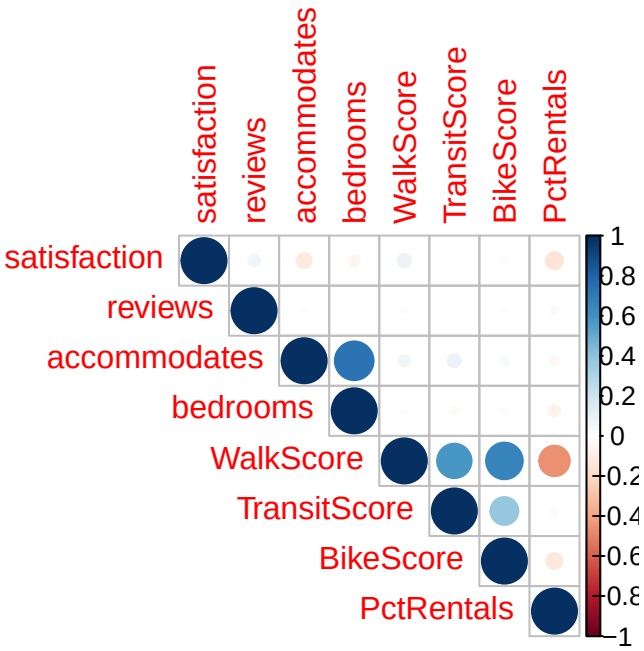


Figure 3.1: Correlation Plot

```
suppressMessages(library(glmnet))

# designed matrix
```

```r
XD <- model.matrix(price ~ ., data = dataClean)
# dim(XD)
```

Tuning Parameter (Ridge)

```r
# Run ridge Regression
suppressMessages(library(glmnet))
set.seed(1)
cv.out.ridge <- cv.glmnet(XD[,-1], dataClean$price, alpha = 0,lambda = seq(from = 0, to = 100, by = .5))

# Default plot
# plot(cv.out.ridge)
```

```r
# The plot in ggplot2 Dr. Dalzell built
ridgePlot <- function(ridge.mod, metric, title){
  library(ggplot2)

  smallestLambda <- ridge.mod$lambda[which.min(ridge.mod$cvm)]

  if(metric == "MSE"){
  g1 <- ggplot( data.frame(ridge.mod$lambda), aes( x = ridge.mod$lambda, y = (ridge.mod$cvm))) + geom_p

  }

  if(metric == "RMSE"){
  g1 <- ggplot( data.frame(ridge.mod$lambda), aes( x = ridge.mod$lambda, y = sqrt(ridge.mod$cvm))) + ge

  }

  g1
}
```

```r
#Plotting using ggplot2
ridgePlot(cv.out.ridge, metric = "MSE", title = "") + theme(text = element_text(size = 12))
```

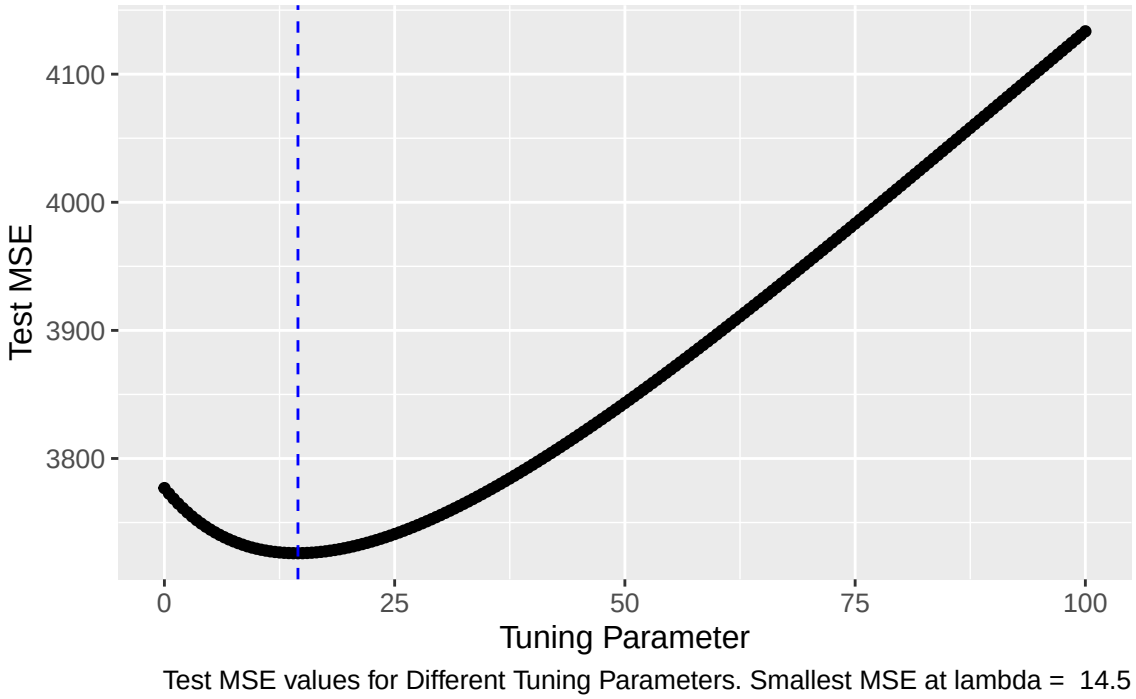


Figure 3.2: Test MSE vs Tuning Parameter of Ridge Regression model

```
# Metrics

# Run the process again (same seed) with only the two lambdas of interest
set.seed(1)
cv.out.ridge.small <- cv.glmnet(XD[,-1], dataClean$price, alpha = 0,lambda = c(0,14.5))
# print("Lambda: ")
# cv.out.ridge.small$lambda
# print("MSE: ")
# cv.out.ridge.small$cvm
# print("RMSE: ")
# sqrt(cv.out.ridge.small$cvm)

# construct a matrix to store the output values
show <- matrix(data = c(cv.out.ridge.small$lambda, cv.out.ridge.small$cvm, sqrt(cv.out.ridge.small$cvm)

# name the rows and columns
rownames(show) <- c("Ridge", "LSLR")
colnames(show) <- c("Lambda", "Test MSE", "Test RMSE")

# construct a table to show the output
knitr::kable(show, caption = "Metrics of LSLR vs Ridge")
```

Table 3: Metrics of LSLR vs Ridge

	Lambda	Test MSE	Test RMSE
Ridge	14.5	3726.107	61.04185
LSLR	0.0	3776.969	61.45705

Evaluation

```
# ridge regression lambda = 14.5

# we need to remove the first row of XD as glmnet will add another line for coefficient
reg.ridge <- glmnet(XD[,-1], dataClean$price, alpha = 0,lambda = 14.5, standardize =TRUE)
# LSLR lambda = 0
reg.LSLR <- glmnet(XD[,-1], dataClean$price, alpha = 0,lambda = 0, standardize = TRUE)
Betas <- data.frame("LSLR" = as.numeric(coefficients(reg.LSLR)), "ridge" = as.numeric(coefficients(reg.
rownames(Betas) <- colnames(XD)

# compute the MSE for Ridge and LSLR
MSE <- mean( (data$price - predict(reg.LSLR, newx = XD[,-1]))^2)
MSEr <- mean( (data$price - predict(reg.ridge, newx = XD[,-1]))^2)

#reg.LSLR$beta
#coefficients(reg.LSLR)

knitr::kable(round(Betas, 3),
             caption = "Coefficients with Tuning Parameter = 0, 14.5")
```

Table 4: Coefficients with Tuning Parameter = 0, 14.5

	LSLR	ridge
(Intercept)	-158.756	-237.574
overall_satisfaction	34.213	27.470
reviews	-0.121	-0.098
room_typePrivate room	-25.239	-25.796
room_typeShared room	-57.118	-49.615
accommodates	12.856	11.138
bedrooms	29.215	27.895
neighborhoodArcher Heights	-16.708	2.769
neighborhoodAvondale	-13.347	-4.378
neighborhoodBeverly	-7.548	19.510
neighborhoodBridgeport	-25.213	-11.567
neighborhoodBrighton Park	-42.259	-13.847
neighborhoodBurnside	-32.570	-2.299
neighborhoodCalumet Heights	-6.104	17.293
neighborhoodEast Garfield Park	-34.903	-14.024
neighborhoodEdgewater	4.703	-8.142
neighborhoodEdison Park	-8.374	14.670
neighborhoodEnglewood	3.770	25.027
neighborhoodGage Park	-10.355	3.515
neighborhoodHegewisch	-40.994	1.664
neighborhoodHermosa	-24.103	-13.750
neighborhoodHumboldt Park	-21.456	-11.252
neighborhoodHyde Park	-7.820	-2.096
neighborhoodIrving Park	35.494	31.587
neighborhoodJefferson Park	-9.141	-3.391
neighborhoodKenwood	-9.129	7.465
neighborhoodLincoln Park	38.331	25.989
neighborhoodLincoln Square	17.133	11.785

	LSLR	ridge
neighborhoodLogan Square	-13.751	-8.594
neighborhoodMcKinley Park	-16.382	-0.102
neighborhoodMontclare	-38.250	-16.658
neighborhoodMorgan Park	-32.915	-1.605
neighborhoodNear West Side	26.797	19.666
neighborhoodNorth Center	-5.527	1.440
neighborhoodNorth Park	11.986	18.493
neighborhoodO'Hare	11.202	33.065
neighborhoodPortage Park	-21.875	-11.114
neighborhoodPullman	-12.557	25.651
neighborhoodRogers Park	0.937	-8.402
neighborhoodSouth Chicago	-18.502	-2.263
neighborhoodSouth Shore	-31.585	-3.833
neighborhoodThe Loop	58.659	31.844
neighborhoodUptown	28.311	9.395
neighborhoodWashington Park	-2.702	21.163
neighborhoodWest Elsdon	-22.100	1.478
neighborhoodWest Englewood	-131.238	-72.480
neighborhoodWest Lawn	-65.419	-33.227
neighborhoodWest Town	14.048	5.742
neighborhoodWoodlawn	-32.325	-3.504
districtFar North	-16.310	-5.678
districtFar Southeast	1.146	6.822
districtFar Southwest	0.788	5.582
districtNorth	1.786	-3.039
districtNorthwest	2.006	7.489
districtSouth	0.287	-4.358
districtSouthwest	0.258	-3.924
districtWest	-0.806	-0.751
WalkScore	-0.150	0.567
TransitScore	0.382	0.983
BikeScore	0.268	0.467
PctRentals	9.676	-6.586

```
# construct a matrix to store the output values
show2 <- matrix(data = c(cv.out.ridge.small$lambda, MSEr, MSE, abs(sum(reg.ridge$beta)), abs(sum(reg.LSl

# name the rows and columns
rownames(show2) <- c("Ridge", "LSLR")
# in this case, it should be MSE, as it is for training model
colnames(show2) <- c("Lambda", "MSE", "Sum Coefficients (abs)")

# construct a table to show the output
knitr::kable(round(show2,1), caption = "Metrics of LSLR vs Ridge")
```

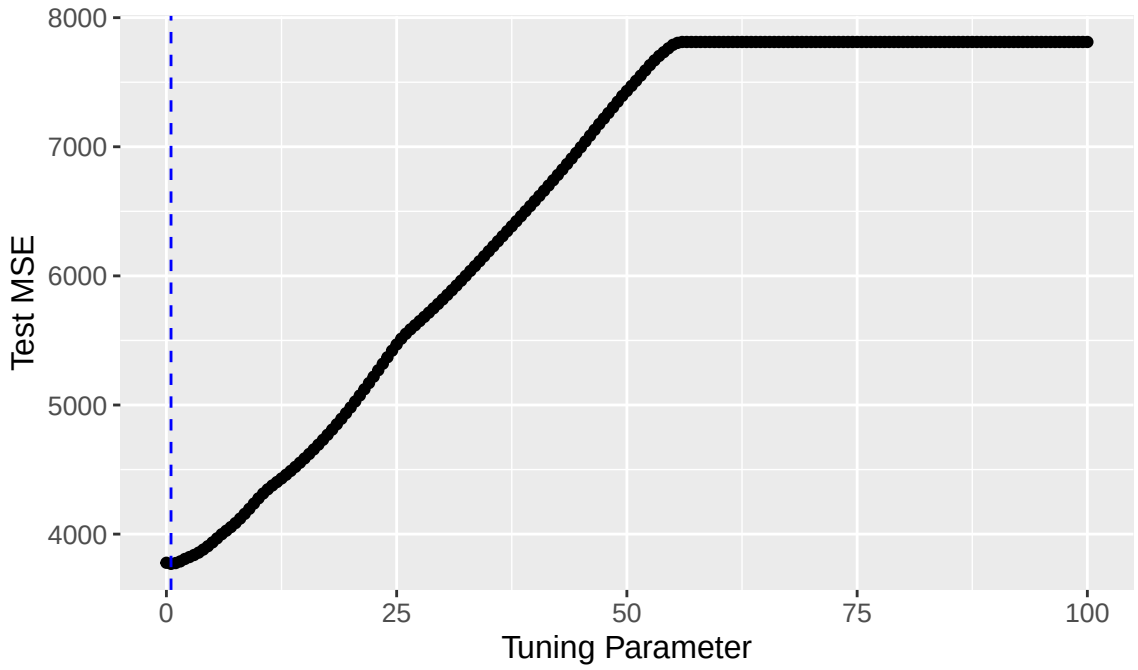
Table 5: Metrics of LSLR vs Ridge

	Lambda	MSE	Sum Coefficients (abs)
Ridge	14.5	3480.2	71.2
LSLR	0.0	3449.7	510.6

Section 4: Lasso

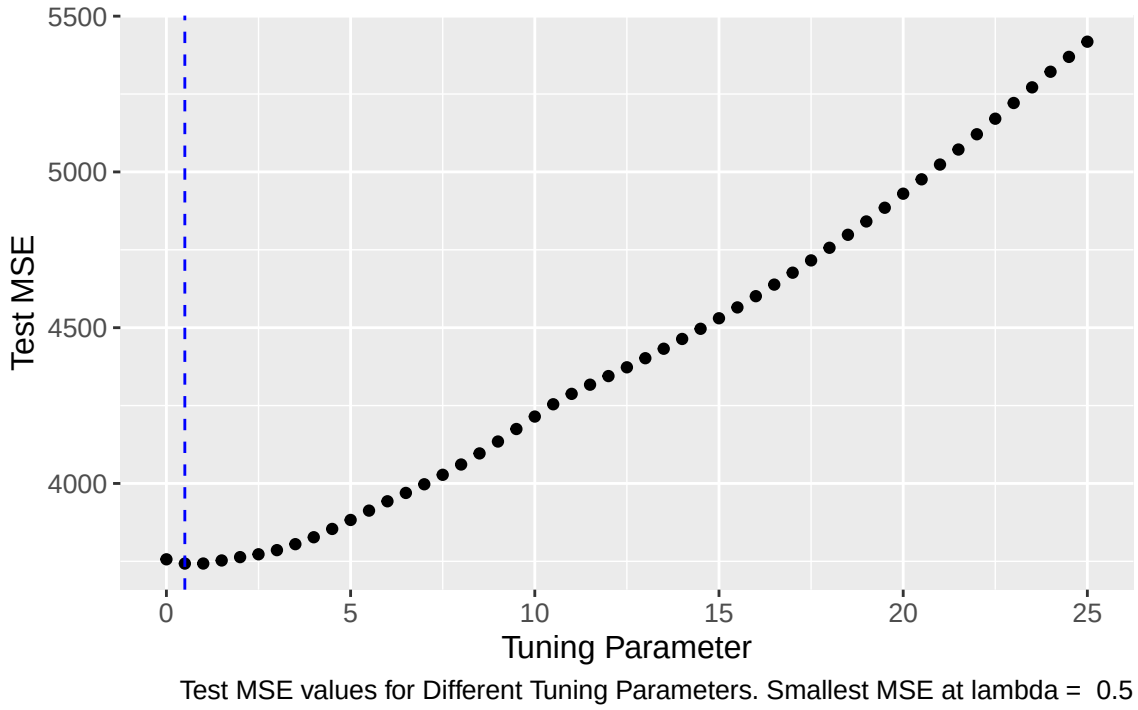
Tuning Parameter (Lasso)

```
# import the ggplot2 package
library(ggplot2)

# Run 10-fold CV
set.seed(1)
cv.out.lasso <- cv.glmnet(XD[,-1], dataClean$price, alpha = 1,lambda = seq(from = 0, to = 100, by = .5))

# Use a smaller range
cv.out.lasso2 <- cv.glmnet(XD[,-1], dataClean$price, alpha = 1,lambda = seq(from = 0, to = 25, by = .5))


# cluster of mosaic plots
op <- par( mfrow = c(2,1))

# Plot the results using the default plotting tool
# plot(cv.out.lasso)


#Plotting using ggplot2
ridgePlot(cv.out.lasso, metric = "MSE", title = "") + theme(text = element_text(size = 12))
```



Test MSE values for Different Tuning Parameters. Smallest MSE at lambda = 0.5

```
# Use a smaller range
ridgePlot(cv.out.lasso2, metric = "MSE", title = "") + theme(text = element_text(size = 12))
```




```
par(op)
```

Figure 4.1: Test MSE vs Tuning Parameter of Lasso Regression model

```
# Choosing the lambda we want
cv.out.lasso$lambda.min
```

```
## [1] 0.5
```

Evaluation

```
# Obtaining coefficients

# Train Lasso
reg.lasso <- glmnet(XD[,-1], dataClean$price, alpha =1 ,
                    lambda = 0.5)

# Store the coefficients
lasso_betas <- as.numeric(coefficients(reg.lasso))

# Create a data frame
# Betas <- data.frame("LSLR" = lslr.betas, "ridge" = ridge.betas, "Lasso" = lasso.betas)
```

```
Betas_pro <- cbind(Betas, "Lasso" = lasso_betas)

rownames(Betas) <- colnames(XD)

knitr::kable(round(Betas_pro, 3), caption = "Coefficients Table of LSLR, Ridge, Lasso")
```

Table 6: Coefficients Table of LSLR, Ridge, Lasso

	LSLR	ridge	Lasso
(Intercept)	-158.756	-237.574	-248.006
overall_satisfaction	34.213	27.470	32.111
reviews	-0.121	-0.098	-0.109
room_typePrivate room	-25.239	-25.796	-24.627
room_typeShared room	-57.118	-49.615	-53.399
accommodates	12.856	11.138	12.638
bedrooms	29.215	27.895	29.064
neighborhoodArcher Heights	-16.708	2.769	0.000
neighborhoodAvondale	-13.347	-4.378	-0.431
neighborhoodBeverly	-7.548	19.510	12.950
neighborhoodBridgeport	-25.213	-11.567	-7.369
neighborhoodBrighton Park	-42.259	-13.847	-10.273
neighborhoodBurnside	-32.570	-2.299	0.000
neighborhoodCalumet Heights	-6.104	17.293	9.634
neighborhoodEast Garfield Park	-34.903	-14.024	-19.253
neighborhoodEdgewater	4.703	-8.142	-4.368
neighborhoodEdison Park	-8.374	14.670	0.000
neighborhoodEnglewood	3.770	25.027	5.806
neighborhoodGage Park	-10.355	3.515	0.000
neighborhoodHegewisch	-40.994	1.664	0.000
neighborhoodHermosa	-24.103	-13.750	0.000
neighborhoodHumboldt Park	-21.456	-11.252	-8.127
neighborhoodHyde Park	-7.820	-2.096	0.000
neighborhoodIrving Park	35.494	31.587	45.435
neighborhoodJefferson Park	-9.141	-3.391	-5.651
neighborhoodKenwood	-9.129	7.465	0.000
neighborhoodLincoln Park	38.331	25.989	32.698
neighborhoodLincoln Square	17.133	11.785	11.235
neighborhoodLogan Square	-13.751	-8.594	-3.433
neighborhoodMcKinley Park	-16.382	-0.102	0.000
neighborhoodMontclare	-38.250	-16.658	-3.852
neighborhoodMorgan Park	-32.915	-1.605	0.000
neighborhoodNear West Side	26.797	19.666	18.639
neighborhoodNorth Center	-5.527	1.440	4.049
neighborhoodNorth Park	11.986	18.493	13.236
neighborhoodO'Hare	11.202	33.065	20.463
neighborhoodPortage Park	-21.875	-11.114	0.000
neighborhoodPullman	-12.557	25.651	8.646
neighborhoodRogers Park	0.937	-8.402	-8.965
neighborhoodSouth Chicago	-18.502	-2.263	0.000
neighborhoodSouth Shore	-31.585	-3.833	-5.984
neighborhoodThe Loop	58.659	31.844	33.654
neighborhoodUptown	28.311	9.395	7.647

	LSLR	ridge	Lasso
neighborhoodWashington Park	-2.702	21.163	2.740
neighborhoodWest Elsdon	-22.100	1.478	0.000
neighborhoodWest Englewood	-131.238	-72.480	-82.608
neighborhoodWest Lawn	-65.419	-33.227	-29.951
neighborhoodWest Town	14.048	5.742	11.203
neighborhoodWoodlawn	-32.325	-3.504	-13.365
districtFar North	-16.310	-5.678	0.000
districtFar Southeast	1.146	6.822	2.469
districtFar Southwest	0.788	5.582	0.000
districtNorth	1.786	-3.039	0.000
districtNorthwest	2.006	7.489	0.000
districtSouth	0.287	-4.358	0.000
districtSouthwest	0.258	-3.924	0.000
districtWest	-0.806	-0.751	0.000
WalkScore	-0.150	0.567	0.000
TransitScore	0.382	0.983	1.365
BikeScore	0.268	0.467	0.422
PctRentals	9.676	-6.586	0.000

```
# coefficients equals zero
# length(which(lasso_betas == 0))
```

```
# MSE of the trained lasso
MSEl <- mean( (data$price - predict(reg.lasso, newx = XD[,-1]))^2)
# MSEl
# sum(reg.lasso$beta)
```

Section 5: Elastic Net

Tuning Parameters (Elastic Net)

```
# Choose a sequence of values for alpha
alphaseq <- seq(from = 0, to =1 , by =.01)

storageEN <- data.frame("Alpha" = rep(NA,length(alphaseq)), "Lambda" = rep(NA,length(alphaseq)), "MSE" =

a = 1
# Run 10-fold CV
set.seed(1)
for( i in alphaseq ){
  cv.out <- cv.glmnet(XD[ , -1], dataClean$price, alpha = i,lambda = seq(from = 0, to = 25, by = .5))
  storageEN$Lambda[a] <- cv.out$lambda.min  # store the optimal lambda at this alpha
  storageEN$MSE[a] <- (min(cv.out$cvm))  # store the test MSE
  storageEN$Alpha[a] <- i  # store the alpha
  a = a + 1
}
```

```r
# The plot in ggplot2 Dr. Dalzell built
elasticNetPlot <- function(ridge.mod, metric, title){
  library(ggplot2)

  smallestLambda <- ridge.mod$Lambda[which.min(ridge.mod$MSE)]
  smallestAlpha <- ridge.mod$Alpha[which.min(ridge.mod$MSE)]

  if(metric == "MSE"){
  g1 <- ggplot( data.frame(ridge.mod$Alpha), aes( x = ridge.mod$Alpha, y = (ridge.mod$MSE))) + geom_poi

  }

  if(metric == "RMSE"){
  g1 <- ggplot( data.frame(ridge.mod$lambda), aes( x = ridge.mod$lambda, y = sqrt(ridge.mod$MSE))) + ge

  }

  g1
}
```

```r
elasticNetPlot(storageEN, "MSE", " ") + theme(text = element_text(size = 15))
```

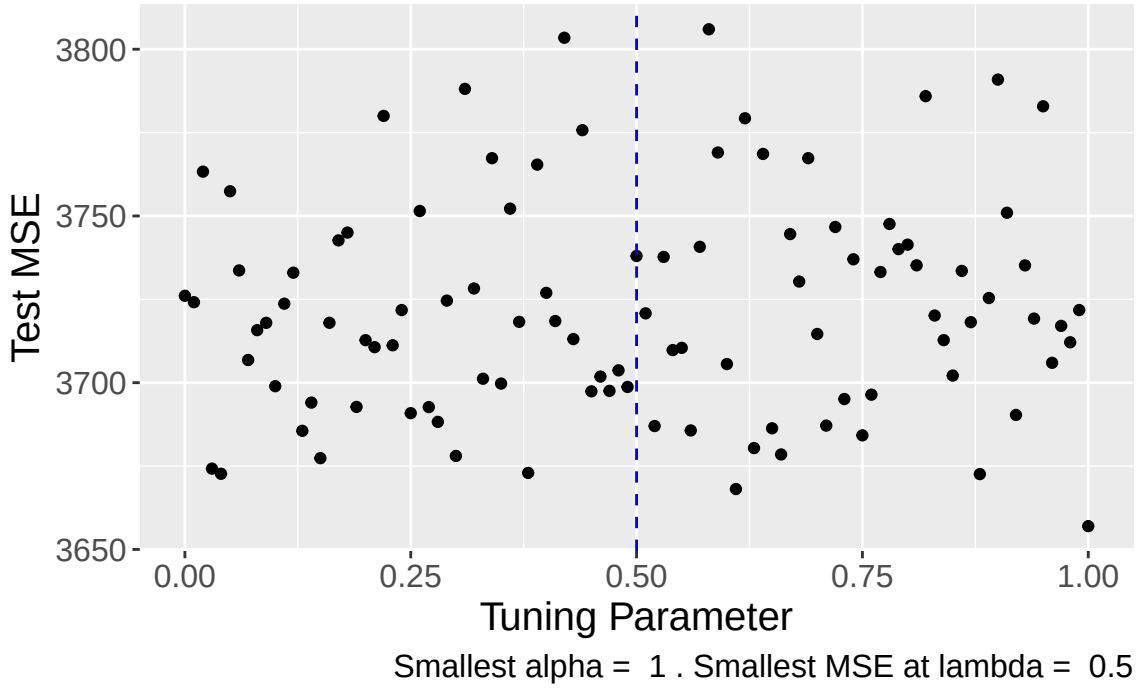


Figure 5.1: Test MSE vs Tuning Parameters of Elastic Net Model

```r
# Show the results
# knitr::kable(round(storageEN,2), caption = "Performance of Elastic Net")
```

Evaluation

Section 6: Comparison and Conclusions

Summary Table

```
# construct a matrix to store the output values for summary table
# because beta does not include the intercept, so we should also add it to the number of features
show3 <- matrix(data = c(reg.lasso$lambda, reg.ridge$lambda, reg.LSLR$lambda, MSEl, MSEr,MSE, abs(sum(r

# name the rows and columns
rownames(show3) <- c("Lasso", "Ridge", "LSLR")
# metric names
colnames(show3) <- c("Lambda", "MSE", "Sum Coefficients (abs)", "Num Features")

# construct a table to show the output
knitr::kable(round(show3,1), caption = "Summary Table: Lasso vs Ridge vs LSLR")
```

Table 7: Summary Table: Lasso vs Ridge vs LSLR

	Lambda	MSE	Sum Coefficients (abs)	Num Features
Lasso	0.5	3459.3	34.3	38
Ridge	14.5	3480.2	71.2	60
LSLR	0.0	3449.7	510.6	60