

Appendix

Import Data

```
# Choose whether to reprocess data
reprocess = FALSE
reprocess = reprocess || !file.exists("../data/boston_data_raw.csv")
reprocess = reprocess || !file.exists("../data/boston_data.csv")
reprocess = reprocess || !file.exists("../data/boston_data_dummied.csv")

reprocess = reprocess || !file.exists("../data/boston_ddowntown.csv")
reprocess = reprocess || !file.exists("../data/boston_dairport.csv")
reprocess = reprocess || !file.exists("../data/boston_dboth.csv")

reprocess = reprocess || !file.exists("../data/boston_ddowntown_dummied.csv")
reprocess = reprocess || !file.exists("../data/boston_dairport_dummied.csv")
reprocess = reprocess || !file.exists("../data/boston_dboth_dummied.csv")

reprocess = reprocess || !file.exists("../data/boston_outliers.csv")

#key <- "AIzaSyAD1g6QxYLtpUpJvAGRLA00j9WcEzjJlj0"

#get_transit_dist <- function(origin_lat, origin_lng){
# orig <- paste(toString(origin_lat), "+", toString(origin_lng), sep= "")
# transits <- google_places(location = c(origin_lat,origin_lng),
#                             place_type = 'transit_station',
#                             radius = 5000,
#                             key = key)
# lats <- transits$results$geometry$location$lat[1]
# logs <- transits$results$geometry$location$lng[1]
# transit <- paste(toString(lats), "+", toString(logs), sep="")
# dist <- gmapsdistance(origin = orig, destination = transit, mode = "walking")
# return(dist)
#}

if (!reprocess) {
  # Read in existing data from file
  boston.data.raw <- read.csv("../data/boston_data_raw.csv", sep = ",", header=TRUE, na.strings=c("", "N", "NA"))
  boston.data <- read.csv("../data/boston_data.csv", sep = ",", header=TRUE, na.strings=c("", " ", "NA"))
  boston.dummied <- read.csv("../data/boston_data_dummied.csv", sep = ",", header=TRUE, na.strings=c("", " ", "NA"))

  boston.dboth <- read.csv("../data/boston_dboth.csv", sep = ",", header=TRUE, na.strings=c("", " ", "NA"))
  boston.ddowntown<- read.csv("../data/boston_ddowntown.csv", sep = ",", header=TRUE, na.strings=c("", " ", "NA"))
  boston.dairport <- read.csv("../data/boston_dairport.csv", sep = ",", header=TRUE, na.strings=c("", " ", "NA"))

  boston.dboth.dummied <- read.csv("../data/boston_dboth_dummied.csv", sep = ",", header=TRUE, na.strings=c("", " ", "NA"))
  boston.ddowntown.dummied <- read.csv("../data/boston_ddowntown_dummied.csv", sep = ",", header=TRUE, na.strings=c("", " ", "NA"))
  boston.dairport.dummied <- read.csv("../data/boston_dairport_dummied.csv", sep = ",", header=TRUE, na.strings=c("", " ", "NA"))

  boston.outliers <- read.csv("../data/boston_outliers.csv", sep = ",", header=TRUE, na.strings=c("", " ", "NA"))
}
```

```

full.data <- read.csv("../data/listings.csv", sep = ",", header=TRUE, na.strings=c("", " ", "NA"))
} else {
  # Read in full dataset
  full.data <- read.csv("../data/listings.csv", sep = ",", header=TRUE, na.strings=c("", " ", "NA"))

  #for (i in 1:nrow(full.data)) {
  # full.data[i, "dtransit"] <- get_transit_dist(full.data$longitude[i], full.data$latitude[i])
  #}

  # Select features to keep
  features_to_keep <- c("host_is_superhost", "host_identity_verified", "neighbourhood_cleansed", "property_type")
  boston.data.raw <- full.data[, features_to_keep, drop=FALSE]

  # Clean dataframe
  ## Omit NA values
  boston.data <- na.omit(boston.data.raw)

  ## Change price to numeric
  boston.data$price <- as.numeric(gsub(",", "", substr(boston.data$price, 2, length(boston.data$price))))

  ## Rename neighbourhood_cleansed to neighborhood
  names(boston.data)[names(boston.data) == "neighbourhood_cleansed"] <- "neighborhood"

  ## Keep up to 95th percentile of price
  value = quantile(boston.data$price, c(.95))[[1]]
  boston.outliers <- boston.data[boston.data$price > value, ]
  boston.data <- boston.data[boston.data$price <= value, ]

  # Dummy categorical features
  ## Remove categorical columns to be re-added
  categorical <- c("host_is_superhost", "host_identity_verified", "neighborhood", "property_type", "room_type")
  boston.dummied <- boston.data %>% select(-one_of(categorical))

  ## host_is_superhost
  boston.dummied <- cbind(boston.dummied, host_is_superhost=dummy(boston.data$host_is_superhost, sep="_"))

  ## host_identity_verified
  boston.dummied <- cbind(boston.dummied, host_identity_verified=dummy(boston.data$host_identity_verified, sep="_"))

  ## instant_bookable
  boston.dummied <- cbind(boston.dummied, instant_bookable=dummy(boston.data$instant_bookable, sep="_"))

  ## is_business_travel_ready
  boston.dummied <- cbind(boston.dummied, is_business_travel_ready=dummy(boston.data$is_business_travel_ready, sep="_"))

  ## property_type
  temp <- data.frame(dummy(boston.data$property_type))[, -1]
  boston.dummied <- cbind(boston.dummied, temp)

```

```

## room_type
temp <- data.frame(dummy(boston.data$room_type))[ , -1]
boston.dummied <- cbind(boston.dummied, temp)

## bed_type
temp <- data.frame(dummy(boston.data$bed_type))[ , -1]
boston.dummied <- cbind(boston.dummied, temp)

## cancellation_policy
temp <- data.frame(dummy(boston.data$cancellation_policy))[ , -1]
boston.dummied <- cbind(boston.dummied, temp)

# Construct distinct datasets

## Dataset with distance to downtown and airport
boston.dboth.dummied <- boston.dummied
boston.dboth.dummied$ddowntown <- 0
boston.dboth.dummied$dairport <- 0

boston.dboth <- boston.data
boston.dboth$ddowntown <- 0
boston.dboth$dairport <- 0

### Calculate driving distance from property location to downtown/airport
ddowntown = hashmap(levels(boston.data$neighborhood), integer(length(levels(boston.data$neighborhood))))
dairport = hashmap(levels(boston.data$neighborhood), integer(length(levels(boston.data$neighborhood))))

for (i in 1:length(levels(boston.data$neighborhood))) {
  s <- levels(boston.data$neighborhood)[[i]]
  s2 <- paste(s, ", Boston MA")
  s2 <- gsub(" ", "+", s2, fixed=TRUE)
  ddowntown[[s]] <- gmapsdistance(origin=s2, destination="42.3555925+-71.0624982", mode="driving")[[2]]
  dairport[[s]] <- gmapsdistance(origin=s2, destination="42.3656171+-71.0117542", mode="driving")[[2]]
}

for (i in 1:nrow(boston.dboth.dummied)) {
  boston.dboth.dummied[i, "ddowntown"] <- ddowntown[[boston.data$neighborhood[[i]]]]
  boston.dboth.dummied[i, "dairport"] <- dairport[[boston.data$neighborhood[[i]]]]

  boston.dboth[i, "ddowntown"] <- ddowntown[[boston.data$neighborhood[[i]]]]
  boston.dboth[i, "dairport"] <- dairport[[boston.data$neighborhood[[i]]]]
}

### Remove neighborhood columns
boston.dboth.dummied <- boston.dboth.dummied[ , !(names(boston.dboth.dummied) %in% c("neighborhood"))]
boston.dboth <- boston.dboth[ , !(names(boston.dboth) %in% c("neighborhood"))]

## Dataset with distance to downtown only
boston.ddowntown.dummied <- boston.dboth.dummied[ , !(names(boston.dboth.dummied) %in% c("dairport"))]
boston.ddowntown <- boston.dboth[ , !(names(boston.dboth) %in% c("dairport"))]

## Dataset with distance to airport only

```

```

boston.dairport.dummied <- boston.dboth.dummied[ , !(names(boston.dboth.dummied) %in% c("ddowntown"))]
boston.dairport <- boston.dboth[ , !(names(boston.dboth) %in% c("ddowntown"))]

# Dummy neighborhood
## neighborhood
temp <- data.frame(dummy(boston.data$neighborhood))[ , -1]
boston.dummied <- cbind(boston.dummied, temp)

# Save data
write.csv(boston.data.raw, file="../data/boston_data_raw.csv")
write.csv(boston.data, file="../data/boston_data.csv")
write.csv(boston.dummied, file="../data/boston_data_dummied.csv")

write.csv(boston.ddowntown, file="../data/boston_ddowntown.csv")
write.csv(boston.dairport, file="../data/boston_dairport.csv")
write.csv(boston.dboth, file="../data/boston_dboth.csv")

write.csv(boston.ddowntown.dummied, file="../data/boston_ddowntown_dummied.csv")
write.csv(boston.dairport.dummied, file="../data/boston_dairport_dummied.csv")
write.csv(boston.dboth.dummied, file="../data/boston_dboth_dummied.csv")

write.csv(boston.outliers, file="../data/boston_outliers.csv")
}

```

Preliminary Data Analysis

```

# Outlier investigation
summary(boston.data$price)

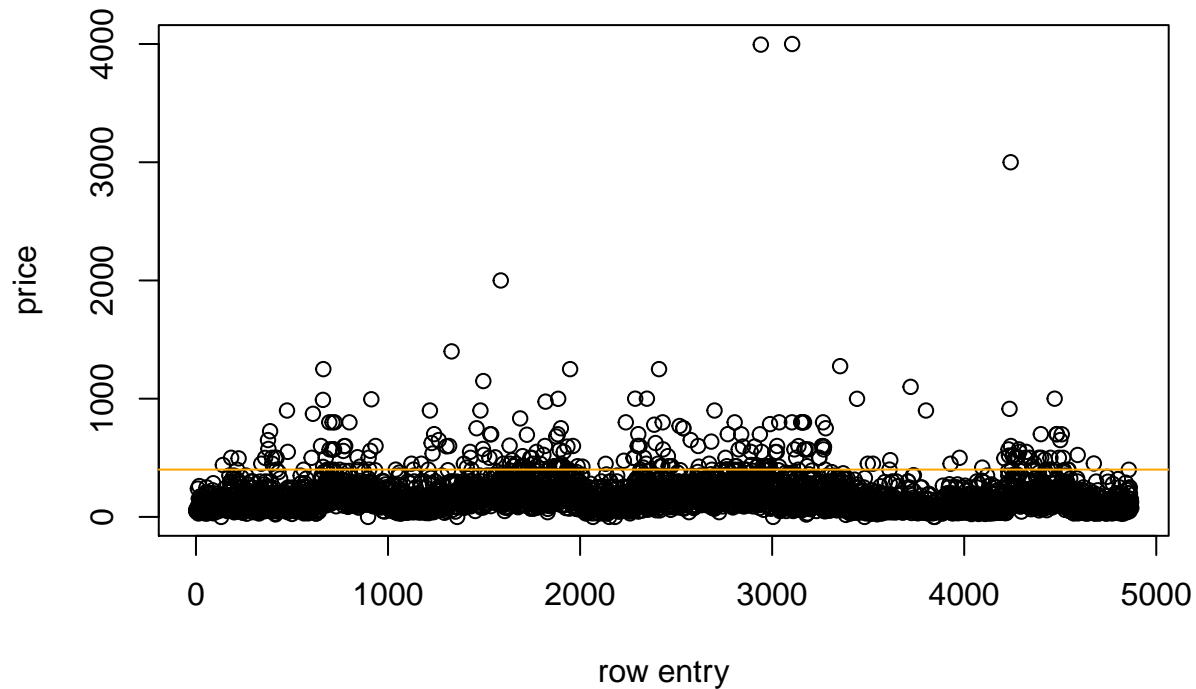
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      0.0   80.0   133.0   148.8   199.0   400.0

sqrt(var(boston.data$price))

## [1] 84.57632

prices.with.outliers <- as.numeric(gsub(",", "", substr(boston.data.raw$price, 2, length(boston.data.raw$price) - 1)))
plot(prices.with.outliers, xlab="row entry", ylab="price")
abline(h=400, col="orange")

```



```
# Visualizing the dataset on a map
```

```
# changing data for map visualization
```

```
full.data$price <- as.numeric(gsub(",", "", substr(full.data$price, 2, length(full.data$price) - 1)))
full.data <- full.data[!is.na(full.data$price),]
price_summary = summary(full.data$price)
```

```
# fetching data for map visualization
```

```
states <- map_data("state")
```

```
ma_df <- subset(states, region == "massachusetts")
```

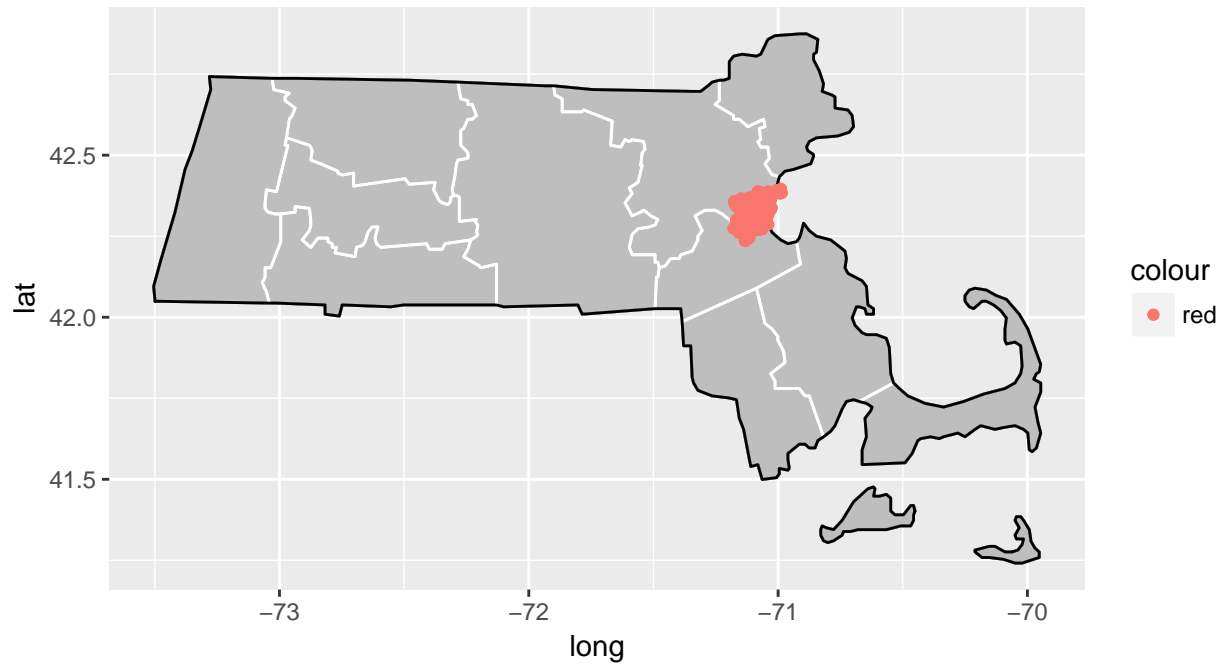
```
counties <- map_data("county")
```

```
ma_county <- subset(counties, region == "massachusetts")
```

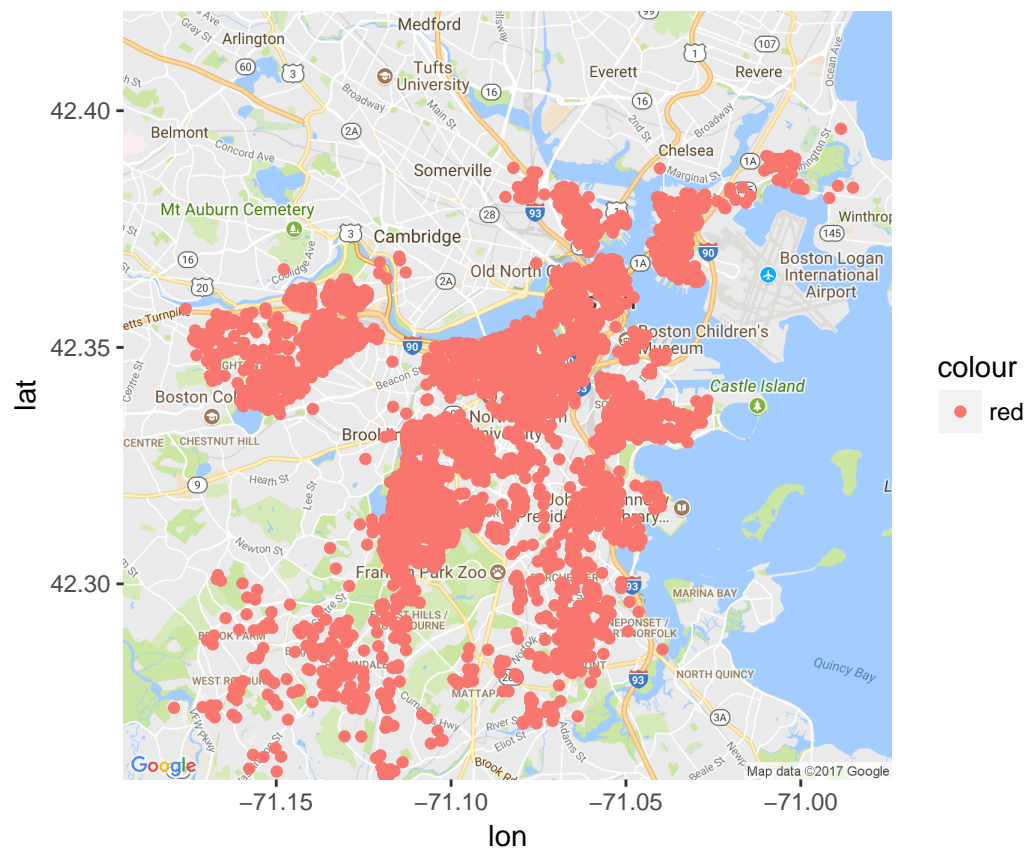
```
ma_base <- ggplot(data = ma_df, mapping = aes(x = long, y = lat, group = group)) + coord_fixed(1.3) +
```

```
ma_city <- ma_base + geom_polygon(data = ma_county, fill = NA, color = "white") + geom_polygon(color = "red")
```

```
ma_city + geom_point(aes(x=longitude,y=latitude,group=zipcode,color="red"),data=full.data)
```



```
map = get_map(location = c(mean(full.data$longitude), mean(full.data$latitude)), zoom = 12, source = "google")
## Map from URL : http://maps.googleapis.com/maps/api/staticmap?center=42.339999,-71.083943&zoom=12&size=600x400
ggmap(map) + geom_point(aes(x=longitude, y=latitude, group=zipcode, color="red"), data=full.data)
```

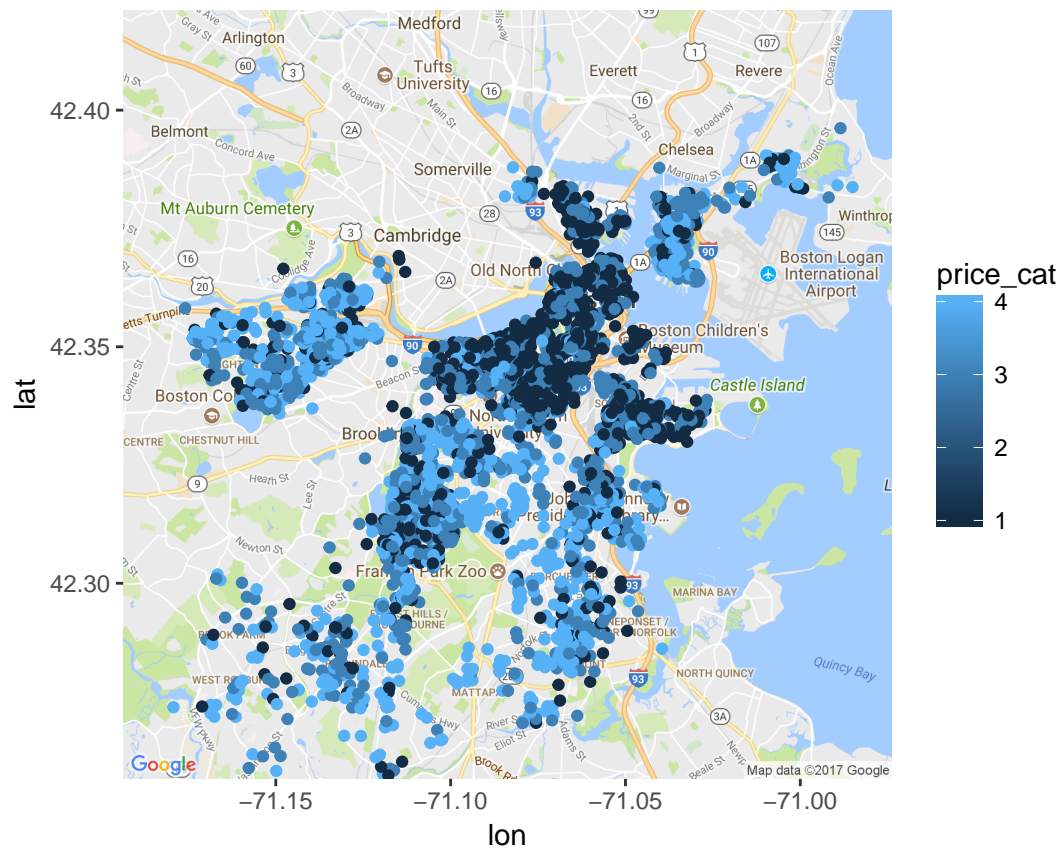


```

# setting price categories
for (i in 1:nrow(full.data))
{
  if (full.data$price[i]<price_summary[2])
  {full.data$price_cat[i]=4}
  if (full.data$price[i]>price_summary[2]&full.data$price[i]<price_summary[4])
  {full.data$price_cat[i]=3}
  if (full.data$price[i]>price_summary[4]&full.data$price[i]<price_summary[5])
  {full.data$price_cat[i]=2}
  if (price_summary[4]<full.data$price[i])
  {full.data$price_cat[i]=1}
}

# plot map
ggmap(map) + geom_point(aes(x=longitude,y=latitude,group=price_cat,color=price_cat),data=full.data)

```



Count Missing Values

```
sum(is.na(boston.data.raw))
```

```
## [1] 14
```


Divide data into training and validation sets

```
# Sample indices for training and validations sets
num_rows <- nrow(boston.data)

# 55% training
training <- sample(1:num_rows, floor(0.55 * num_rows))
rest <- (1:num_rows)[-training]

# 15% model selection
model_selection <- sample(rest, floor(0.15 * num_rows))
rest <- (1:num_rows)[-c(training, model_selection)]

# 15% validation #1
validation_1 <- sample(rest, floor(0.15 * num_rows))
rest <- (1:num_rows)[-c(training, model_selection, validation_1)]

# 15% validation #2
validation_2 <- sample(rest, floor(0.15 * num_rows))
rest <- (1:num_rows)[-c(training, model_selection, validation_1, validation_2)]

# Set datasets for each transformation
boston.data.training <- boston.data[training, ]
boston.data.model_selection <- boston.data[model_selection, ]
boston.data.validation_1 <- boston.data[validation_1, ]
boston.data.validation_2 <- boston.data[validation_2, ]
boston.data.model_selection.test <- boston.data.model_selection[ , "price"]
boston.data.validation_1.test <- boston.data.validation_1[ , "price"]
boston.data.validation_2.test <- boston.data.validation_2[ , "price"]

boston.dboth.training <- boston.dboth[training, ]
boston.dboth.model_selection <- boston.dboth[model_selection, ]
boston.dboth.validation_1 <- boston.dboth[validation_1, ]
boston.dboth.validation_2 <- boston.dboth[validation_2, ]
boston.dboth.model_selection.test <- boston.dboth.model_selection[ , "price"]
boston.dboth.validation_1.test <- boston.dboth.validation_1[ , "price"]
boston.dboth.validation_2.test <- boston.dboth.validation_2[ , "price"]

boston.ddowntown.training <- boston.ddowntown[training, ]
boston.ddowntown.model_selection <- boston.ddowntown[model_selection, ]
boston.ddowntown.validation_1 <- boston.ddowntown[validation_1, ]
boston.ddowntown.validation_2 <- boston.ddowntown[validation_2, ]
boston.ddowntown.model_selection.test <- boston.ddowntown.model_selection[ , "price"]
boston.ddowntown.validation_1.test <- boston.ddowntown.validation_1[ , "price"]
boston.ddowntown.validation_2.test <- boston.ddowntown.validation_2[ , "price"]

boston.dairport.training <- boston.dairport[training, ]
boston.dairport.model_selection <- boston.dairport[model_selection, ]
boston.dairport.validation_1 <- boston.dairport[validation_1, ]
boston.dairport.validation_2 <- boston.dairport[validation_2, ]
boston.dairport.model_selection.test <- boston.dairport.model_selection[ , "price"]
boston.dairport.validation_1.test <- boston.dairport.validation_1[ , "price"]
boston.dairport.validation_2.test <- boston.dairport.validation_2[ , "price"]
```



```

#Dummied
boston.dummied.training <- boston.dummied[training, ][,-1]
boston.dummied.model_selection <- boston.dummied[model_selection, ][,-1]
boston.dummied.validation_1 <- boston.dummied[validation_1, ][,-1]
boston.dummied.validation_2 <- boston.dummied[validation_2, ][,-1]
boston.dummied.model_selection.test <- boston.dummied.model_selection[ , "price"]
boston.dummied.validation_1.test <- boston.dummied.validation_1[ , "price"]
boston.dummied.validation_2.test <- boston.dummied.validation_2[ , "price"]

boston.dboth.dummied.training <- boston.dboth.dummied[training, ][,-1]
boston.dboth.dummied.model_selection <- boston.dboth.dummied[model_selection, ][,-1]
boston.dboth.dummied.validation_1 <- boston.dboth.dummied[validation_1, ][,-1]
boston.dboth.dummied.validation_2 <- boston.dboth.dummied[validation_2, ][,-1]
boston.dboth.dummied.model_selection.test <- boston.dboth.dummied.model_selection[ , "price"]
boston.dboth.dummied.validation_1.test <- boston.dboth.dummied.validation_1[ , "price"]
boston.dboth.dummied.validation_2.test <- boston.dboth.dummied.validation_2[ , "price"]

boston.ddowntown.dummied.training <- boston.ddowntown.dummied[training, ][,-1]
boston.ddowntown.dummied.model_selection <- boston.ddowntown.dummied[model_selection, ][,-1]
boston.ddowntown.dummied.validation_1 <- boston.ddowntown.dummied[validation_1, ][,-1]
boston.ddowntown.dummied.validation_2 <- boston.ddowntown.dummied[validation_2, ][,-1]
boston.ddowntown.dummied.model_selection.test <- boston.ddowntown.dummied.model_selection[ , "price"]
boston.ddowntown.dummied.validation_1.test <- boston.ddowntown.dummied.validation_1[ , "price"]
boston.ddowntown.dummied.validation_2.test <- boston.ddowntown.dummied.validation_2[ , "price"]

boston.dairport.dummied.training <- boston.dairport.dummied[training, ][,-1]
boston.dairport.dummied.model_selection <- boston.dairport.dummied[model_selection, ][,-1]
boston.dairport.dummied.validation_1 <- boston.dairport.dummied[validation_1, ][,-1]
boston.dairport.dummied.validation_2 <- boston.dairport.dummied[validation_2, ][,-1]
boston.dairport.dummied.model_selection.test <- boston.dairport.dummied.model_selection[ , "price"]
boston.dairport.dummied.validation_1.test <- boston.dairport.dummied.validation_1[ , "price"]
boston.dairport.dummied.validation_2.test <- boston.dairport.dummied.validation_2[ , "price"]

```

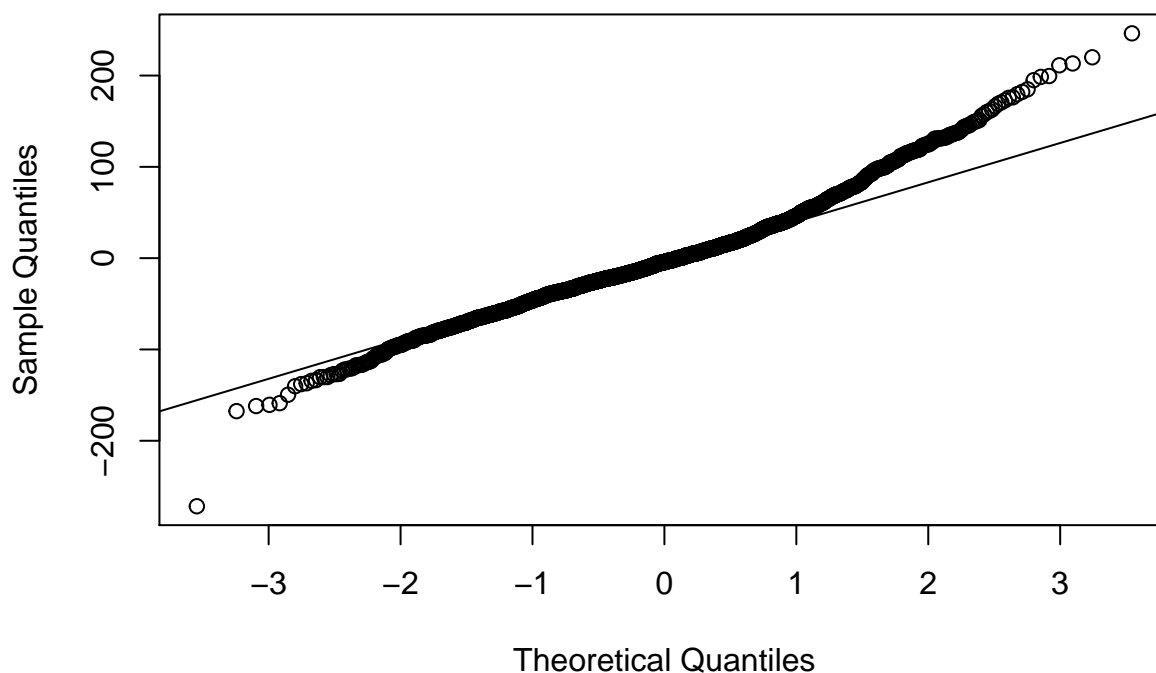
Linear Regression

```

lm_train <- lm(price~.,data = boston.dummied.training)
qqnorm(lm_train$residuals, main = "Normal qqplot of residuals")
qqline(lm_train$residuals)

```

Normal qqplot of residuals



```
summary(lm_train)
```

```
##
## Call:
## lm(formula = price ~ ., data = boston.dummied.training)
##
## Residuals:
```

	Min	1Q	Median	3Q	Max
	-271.727	-32.041	-4.277	26.032	246.226

```
##
## Coefficients:
```

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	77.82281	13.34716	5.831	6.24e-09
accommodates	6.64028	1.26379	5.254	1.61e-07
bathrooms	9.53191	2.78692	3.420	0.000636
bedrooms	21.74546	2.17508	9.998	< 2e-16
beds	-1.27594	1.78775	-0.714	0.475471
guests_included	5.97530	1.14571	5.215	1.99e-07
minimum_nights	-0.55959	0.17343	-3.227	0.001269
number_of_reviews	-0.07602	0.02955	-2.573	0.010154
host_is_superhost	4.24667	2.89429	1.467	0.142433
host_identity_verified	-4.00203	2.29749	-1.742	0.081648
instant_bookable	-15.28341	2.32885	-6.563	6.42e-11
is_business_travel_ready	5.57453	3.43213	1.624	0.104456
property_type.Bed...Breakfast	15.91293	14.06906	1.131	0.258140
property_type.Boat	27.01379	24.19628	1.116	0.264340
property_type.Boutique.hotel	-53.49514	31.67050	-1.689	0.091323
property_type.Condominium	14.10607	3.77584	3.736	0.000191
property_type.Dorm	-23.66661	53.28921	-0.444	0.656997
property_type.Guest.suite	33.07917	21.86623	1.513	0.130459

## property_type.Guesthouse	93.50281	30.78664	3.037	0.002413
## property_type.Hostel	-36.57018	54.36344	-0.673	0.501202
## property_type.House	4.97656	3.42256	1.454	0.146060
## property_type.In.law	-21.41527	24.08282	-0.889	0.373963
## property_type.Loft	28.03977	12.09199	2.319	0.020483
## property_type.Other	56.32753	9.72999	5.789	7.97e-09
## property_type.Serviced.apartment	-34.30252	53.30508	-0.644	0.519951
## property_type.Timeshare	221.67984	53.34252	4.156	3.35e-05
## property_type.Townhouse	11.63049	8.81182	1.320	0.186999
## property_type.Villa	23.95943	20.28778	1.181	0.237725
## room_type.Private.room	-59.96130	3.18379	-18.833	< 2e-16
## room_type.Shared.room	-69.55790	10.36401	-6.711	2.38e-11
## bed_type.Couch	10.82588	29.59114	0.366	0.714509
## bed_type.Futon	9.04917	16.93452	0.534	0.593139
## bed_type.Pull.out.Sofa	5.81692	18.84600	0.309	0.757610
## bed_type.Real.Bed	2.63486	11.64667	0.226	0.821039
## cancellation_policy.moderate	7.41513	3.18136	2.331	0.019844
## cancellation_policy.strict	-8.04279	2.80355	-2.869	0.004155
## cancellation_policy.super_strict_30	41.86365	14.10633	2.968	0.003029
## cancellation_policy.super_strict_60	101.02557	53.46241	1.890	0.058921
## neighborhood.Back.Bay	83.21054	5.94132	14.005	< 2e-16
## neighborhood.Bay.Village	50.58238	14.93415	3.387	0.000718
## neighborhood.Beacon.Hill	71.88781	6.29898	11.413	< 2e-16
## neighborhood.Brighton	7.29242	5.97593	1.220	0.222468
## neighborhood.Charlestown	56.17710	7.88925	7.121	1.40e-12
## neighborhood.Chinatown	49.62942	8.20624	6.048	1.69e-09
## neighborhood.Dorchester	5.68429	5.78911	0.982	0.326248
## neighborhood.Downtown	73.80523	6.37621	11.575	< 2e-16
## neighborhood.East.Boston	10.85957	6.22746	1.744	0.081314
## neighborhood.Fenway	42.65976	5.93631	7.186	8.77e-13
## neighborhood.Hyde.Park	-9.66046	11.57767	-0.834	0.404133
## neighborhood.Jamaica.Plain	12.42193	5.61299	2.213	0.026984
## neighborhood.Leather.District	106.30611	27.64638	3.845	0.000123
## neighborhood.Longwood.Medical.Area	59.90400	27.05814	2.214	0.026926
## neighborhood.Mattapan	-9.92169	13.28813	-0.747	0.455340
## neighborhood.Mission.Hill	13.09720	7.55366	1.734	0.083062
## neighborhood.North.End	26.34956	7.21092	3.654	0.000263
## neighborhood.Roslindale	-14.09843	8.22510	-1.714	0.086640
## neighborhood.Roxbury	9.14193	6.41293	1.426	0.154125
## neighborhood.South.Boston	33.23096	6.31483	5.262	1.54e-07
## neighborhood.South.Boston.Waterfront	91.60338	10.19628	8.984	< 2e-16
## neighborhood.South.End	62.53158	5.92833	10.548	< 2e-16
## neighborhood.West.End	49.06793	11.14381	4.403	1.11e-05
## neighborhood.West.Roxbury	-2.82290	9.70120	-0.291	0.771088
##				
## (Intercept)	***			
## accommodates	***			
## bathrooms	***			
## bedrooms	***			
## beds				
## guests_included	***			
## minimum_nights	**			
## number_of_reviews	*			
## host_is_superhost				

```

## host_identity_verified .
## instant_bookable ***
## is_business_travel_ready
## property_type.Bed...Breakfast
## property_type.Boat
## property_type.Boutique.hotel .
## property_type.Condominium ***
## property_type.Dorm
## property_type.Guest.suite
## property_type.Guesthouse **
## property_type.Hostel
## property_type.House
## property_type.In.law
## property_type.Loft *
## property_type.Other ***
## property_type.Serviced.apartment
## property_type.Timeshare ***
## property_type.Townhouse
## property_type.Villa
## room_type.Private.room ***
## room_type.Shared.room ***
## bed_type.Couch
## bed_type.Futon
## bed_type.Pull.out.Sofa
## bed_type.Real.Bed
## cancellation_policy.moderate *
## cancellation_policy.strict **
## cancellation_policy.super_strict_30 **
## cancellation_policy.super_strict_60 .
## neighborhood.Back.Bay ***
## neighborhood.Bay.Village ***
## neighborhood.Beacon.Hill ***
## neighborhood.Brighton
## neighborhood.Charlestown ***
## neighborhood.Chinatown ***
## neighborhood.Dorchester
## neighborhood.Downtown ***
## neighborhood.East.Boston .
## neighborhood.Fenway ***
## neighborhood.Hyde.Park
## neighborhood.Jamaica.Plain *
## neighborhood.Leather.District ***
## neighborhood.Longwood.Medical.Area *
## neighborhood.Mattapan
## neighborhood.Mission.Hill .
## neighborhood.North.End ***
## neighborhood.Roslindale .
## neighborhood.Roxbury
## neighborhood.South.Boston ***
## neighborhood.South.Boston.Waterfront ***
## neighborhood.South.End ***
## neighborhood.West.End ***
## neighborhood.West.Roxbury
## ---

```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 53.01 on 2477 degrees of freedom
## Multiple R-squared:  0.6145, Adjusted R-squared:  0.605
## F-statistic: 64.74 on 61 and 2477 DF,  p-value: < 2.2e-16

library(leaps)
regfit.full <- regsubsets(price~., data = boston.dummied.training, really.big = T)
reg.summary <- summary(regfit.full)
reg.summary

## Subset selection object
## Call: regsubsets.formula(price ~ ., data = boston.dummied.training,
##      really.big = T)
## 61 Variables (and intercept)
##
##               Forced in Forced out
## accommodates          FALSE      FALSE
## bathrooms             FALSE      FALSE
## bedrooms              FALSE      FALSE
## beds                  FALSE      FALSE
## guests_included       FALSE      FALSE
## minimum_nights        FALSE      FALSE
## number_of_reviews     FALSE      FALSE
## host_is_superhost     FALSE      FALSE
## host_identity_verified FALSE      FALSE
## instant_bookable      FALSE      FALSE
## is_business_travel_ready FALSE      FALSE
## property_type.Bed...Breakfast FALSE      FALSE
## property_type.Boat     FALSE      FALSE
## property_type.Boutique.hotel FALSE      FALSE
## property_type.Condominium FALSE      FALSE
## property_type.Dorm      FALSE      FALSE
## property_type.Guest.suite FALSE      FALSE
## property_type.Guesthouse FALSE      FALSE
## property_type.Hostel    FALSE      FALSE
## property_type.House     FALSE      FALSE
## property_type.In.law    FALSE      FALSE
## property_type.Loft      FALSE      FALSE
## property_type.Other     FALSE      FALSE
## property_type.Serviced.apartment FALSE      FALSE
## property_type.Timeshare FALSE      FALSE
## property_type.Townhouse FALSE      FALSE
## property_type.Villa     FALSE      FALSE
## room_type.Private.room  FALSE      FALSE
## room_type.Shared.room   FALSE      FALSE
## bed_type.Couch          FALSE      FALSE
## bed_type.Futon          FALSE      FALSE
## bed_type.Pull.out.Sofa  FALSE      FALSE
## bed_type.Real.Bed       FALSE      FALSE
## cancellation_policy.moderate FALSE      FALSE
## cancellation_policy.strict FALSE      FALSE
## cancellation_policy.super_strict_30 FALSE      FALSE
## cancellation_policy.super_strict_60 FALSE      FALSE
## neighborhood.Back.Bay  FALSE      FALSE
## neighborhood.Bay.Village FALSE      FALSE
```

```

## neighborhood.Beacon.Hill                FALSE    FALSE
## neighborhood.Brighton                    FALSE    FALSE
## neighborhood.Charlestown                 FALSE    FALSE
## neighborhood.Chinatown                   FALSE    FALSE
## neighborhood.Dorchester                   FALSE    FALSE
## neighborhood.Downtown                     FALSE    FALSE
## neighborhood.East.Boston                  FALSE    FALSE
## neighborhood.Fenway                       FALSE    FALSE
## neighborhood.Hyde.Park                    FALSE    FALSE
## neighborhood.Jamaica.Plain                FALSE    FALSE
## neighborhood.Leather.District             FALSE    FALSE
## neighborhood.Longwood.Medical.Area        FALSE    FALSE
## neighborhood.Mattapan                     FALSE    FALSE
## neighborhood.Mission.Hill                 FALSE    FALSE
## neighborhood.North.End                    FALSE    FALSE
## neighborhood.Roslindale                   FALSE    FALSE
## neighborhood.Roxbury                      FALSE    FALSE
## neighborhood.South.Boston                 FALSE    FALSE
## neighborhood.South.Boston.Waterfront      FALSE    FALSE
## neighborhood.South.End                    FALSE    FALSE
## neighborhood.West.End                     FALSE    FALSE
## neighborhood.West.Roxbury                 FALSE    FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: exhaustive
##      accommodates bathrooms bedrooms beds guests_included
## 1  ( 1 ) " "          " "          " "          " " " "
## 2  ( 1 ) "*"          " "          " "          " " " "
## 3  ( 1 ) "*"          " "          " "          " " " "
## 4  ( 1 ) " "          " "          "*"          " " " "
## 5  ( 1 ) " "          " "          "*"          " " " "
## 6  ( 1 ) " "          " "          "*"          " " " "
## 7  ( 1 ) " "          " "          "*"          " " " "
## 8  ( 1 ) "*"          " "          "*"          " " " "
##      minimum_nights number_of_reviews host_is_superhost
## 1  ( 1 ) " "          " "          " "
## 2  ( 1 ) " "          " "          " "
## 3  ( 1 ) " "          " "          " "
## 4  ( 1 ) " "          " "          " "
## 5  ( 1 ) " "          " "          " "
## 6  ( 1 ) " "          " "          " "
## 7  ( 1 ) " "          " "          " "
## 8  ( 1 ) " "          " "          " "
##      host_identity_verified instant_bookable is_business_travel_ready
## 1  ( 1 ) " "          " "          " "
## 2  ( 1 ) " "          " "          " "
## 3  ( 1 ) " "          " "          " "
## 4  ( 1 ) " "          " "          " "
## 5  ( 1 ) " "          " "          " "
## 6  ( 1 ) " "          " "          " "
## 7  ( 1 ) " "          " "          " "
## 8  ( 1 ) " "          " "          " "
##      property_type.Bed...Breakfast property_type.Boat
## 1  ( 1 ) " "          " "
## 2  ( 1 ) " "          " "

```

```

## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) " " " "
## 7 ( 1 ) " " " "
## 8 ( 1 ) " " " "
##      property_type.Boutique.hotel property_type.Condominium
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "
## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) " " " "
## 7 ( 1 ) " " " "
## 8 ( 1 ) " " " "
##      property_type.Dorm property_type.Guest.suite
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "
## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) " " " "
## 7 ( 1 ) " " " "
## 8 ( 1 ) " " " "
##      property_type.Guesthouse property_type.Hostel property_type.House
## 1 ( 1 ) " " " " " "
## 2 ( 1 ) " " " " " "
## 3 ( 1 ) " " " " " "
## 4 ( 1 ) " " " " " "
## 5 ( 1 ) " " " " " "
## 6 ( 1 ) " " " " " "
## 7 ( 1 ) " " " " " "
## 8 ( 1 ) " " " " " "
##      property_type.In.law property_type.Loft property_type.Other
## 1 ( 1 ) " " " " " "
## 2 ( 1 ) " " " " " "
## 3 ( 1 ) " " " " " "
## 4 ( 1 ) " " " " " "
## 5 ( 1 ) " " " " " "
## 6 ( 1 ) " " " " " "
## 7 ( 1 ) " " " " " "
## 8 ( 1 ) " " " " " "
##      property_type.Serviced.apartment property_type.Timeshare
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "
## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) " " " "
## 7 ( 1 ) " " " "
## 8 ( 1 ) " " " "
##      property_type.Townhouse property_type.Villa
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "

```



```

## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) " " " "
## 7 ( 1 ) " " " "
## 8 ( 1 ) " " " "
##      room_type.Private.room room_type.Shared.room bed_type.Couch
## 1 ( 1 ) "*" " " " "
## 2 ( 1 ) "*" " " " "
## 3 ( 1 ) "*" "*" " "
## 4 ( 1 ) "*" "*" " "
## 5 ( 1 ) "*" "*" " "
## 6 ( 1 ) "*" "*" " "
## 7 ( 1 ) "*" "*" " "
## 8 ( 1 ) "*" " " " "
##      bed_type.Futon bed_type.Pull.out.Sofa bed_type.Real.Bed
## 1 ( 1 ) " " " " " "
## 2 ( 1 ) " " " " " "
## 3 ( 1 ) " " " " " "
## 4 ( 1 ) " " " " " "
## 5 ( 1 ) " " " " " "
## 6 ( 1 ) " " " " " "
## 7 ( 1 ) " " " " " "
## 8 ( 1 ) " " " " " "
##      cancellation_policy.moderate cancellation_policy.strict
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "
## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) " " " "
## 7 ( 1 ) " " " "
## 8 ( 1 ) " " " "
##      cancellation_policy.super_strict_30
## 1 ( 1 ) " "
## 2 ( 1 ) " "
## 3 ( 1 ) " "
## 4 ( 1 ) " "
## 5 ( 1 ) " "
## 6 ( 1 ) " "
## 7 ( 1 ) " "
## 8 ( 1 ) " "
##      cancellation_policy.super_strict_60 neighborhood.Back.Bay
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "
## 3 ( 1 ) " " " "
## 4 ( 1 ) " " "*"
## 5 ( 1 ) " " "*"
## 6 ( 1 ) " " "*"
## 7 ( 1 ) " " "*"
## 8 ( 1 ) " " "*"
##      neighborhood.Bay.Village neighborhood.Beacon.Hill
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "

```

```

## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) " " " "
## 7 ( 1 ) " " "*"
## 8 ( 1 ) " " "*"
##      neighborhood.Brighton neighborhood.Charlestown
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "
## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) " " " "
## 7 ( 1 ) " " " "
## 8 ( 1 ) " " " "
##      neighborhood.Chinatown neighborhood.Dorchester
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "
## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) " " " "
## 7 ( 1 ) " " " "
## 8 ( 1 ) " " " "
##      neighborhood.Downtown neighborhood.East.Boston
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "
## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) "*" " "
## 6 ( 1 ) "*" " "
## 7 ( 1 ) "*" " "
## 8 ( 1 ) "*" " "
##      neighborhood.Fenway neighborhood.Hyde.Park
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "
## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) " " " "
## 7 ( 1 ) " " " "
## 8 ( 1 ) " " " "
##      neighborhood.Jamaica.Plain neighborhood.Leather.District
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "
## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) " " " "
## 7 ( 1 ) " " " "
## 8 ( 1 ) " " " "
##      neighborhood.Longwood.Medical.Area neighborhood.Mattapan
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "

```

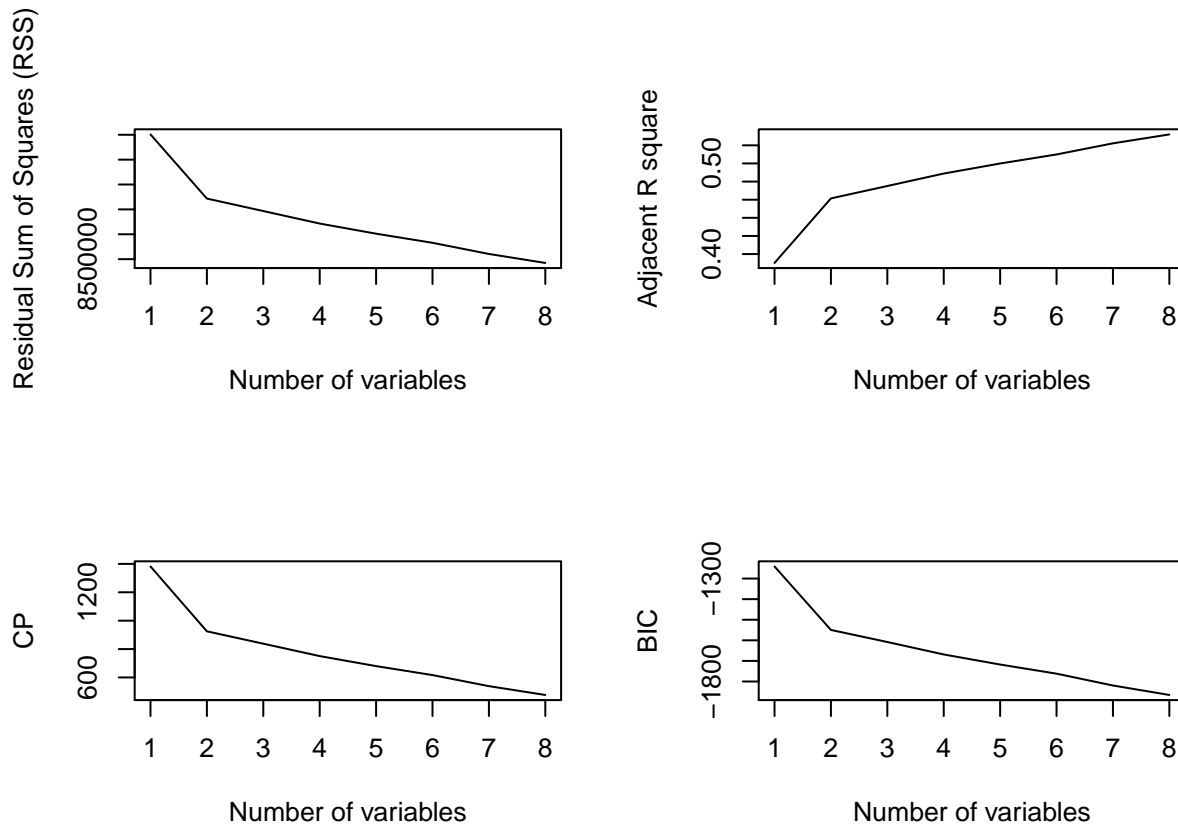
```

## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) " " " "
## 7 ( 1 ) " " " "
## 8 ( 1 ) " " " "
##      neighborhood.Mission.Hill neighborhood.North.End
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "
## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) " " " "
## 7 ( 1 ) " " " "
## 8 ( 1 ) " " " "
##      neighborhood.Roslindale neighborhood.Roxbury
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "
## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) " " " "
## 7 ( 1 ) " " " "
## 8 ( 1 ) " " " "
##      neighborhood.South.Boston neighborhood.South.Boston.Waterfront
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "
## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) " " " "
## 7 ( 1 ) " " " "
## 8 ( 1 ) " " "*"
##      neighborhood.South.End neighborhood.West.End
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "
## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) "*" " "
## 7 ( 1 ) "*" " "
## 8 ( 1 ) "*" " "
##      neighborhood.West.Roxbury
## 1 ( 1 ) " "
## 2 ( 1 ) " "
## 3 ( 1 ) " "
## 4 ( 1 ) " "
## 5 ( 1 ) " "
## 6 ( 1 ) " "
## 7 ( 1 ) " "
## 8 ( 1 ) " "

par(mfrow = c(2,2))
plot(reg.summary$rss, xlab = "Number of variables", ylab = "Residual Sum of Squares (RSS)", type = "l")

```

```
plot(reg.summary$adjr2, xlab = "Number of variables", ylab = "Adjusted R square", type = "l")
plot(reg.summary$cp, xlab = "Number of variables", ylab = "CP", type = "l")
plot(reg.summary$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
```



```
which.min(reg.summary$bic)
```

```
## [1] 8
```

8 predictors from Best Subset

```
# linear model based on 8 predictors
```

```
subset.model <- lm(price ~ room_type.Private.room + room_type.Shared.room + accommodates + bedrooms + i
```

```
# coefficients of the predictors
```

```
coef(regfit.full, 8)
```

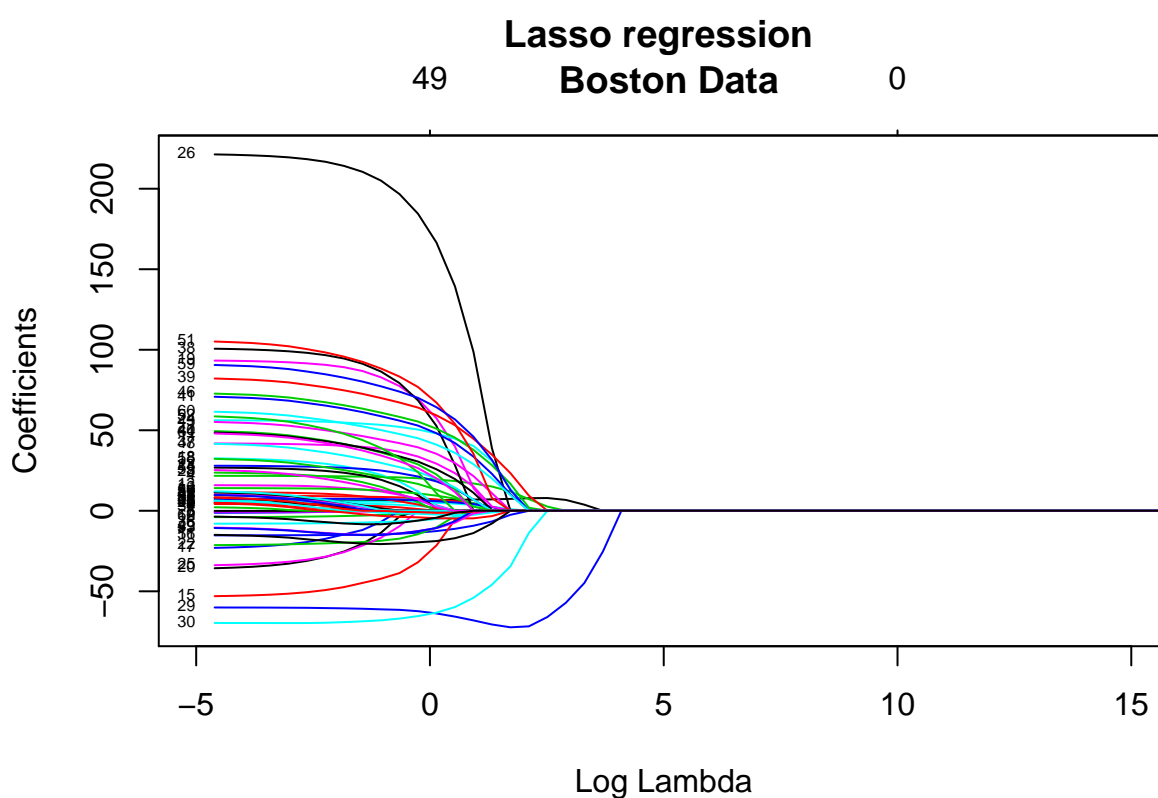
```
##                (Intercept)                accommodates
##                105.497182                8.300836
##                bedrooms                room_type.Private.room
##                22.178145                -66.779191
##                neighborhood.Back.Bay        neighborhood.Beacon.Hill
##                56.987277                47.159364
##                neighborhood.Downtown neighborhood.South.Boston.Waterfront
##                51.630679                75.450650
##                neighborhood.South.End
##                43.273643
```

Lasso Regression

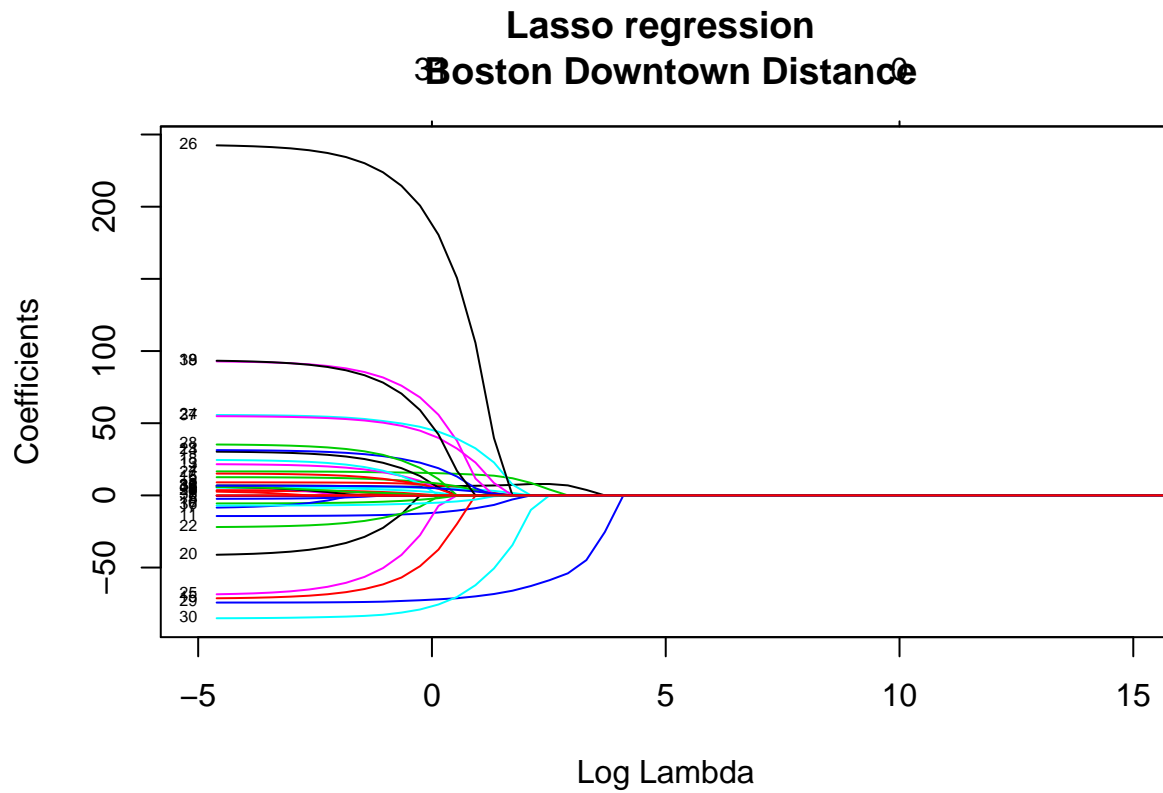
```
x.boston.data <- model.matrix(price~.,boston.dummied.training)
x.boston.dboth <- model.matrix(price~.,boston.dboth.dummied.training)
x.boston.ddowntown <- model.matrix(price~.,boston.ddowntown.dummied.training)
x.boston.dairport <- model.matrix(price~.,boston.dairport.dummied.training)

y.boston.data <- boston.dummied.training$price
y.boston.dboth <- boston.dboth.dummied.training$price
y.boston.ddowntown <- boston.ddowntown.dummied.training$price
y.boston.dairport <- boston.dairport.dummied.training$price

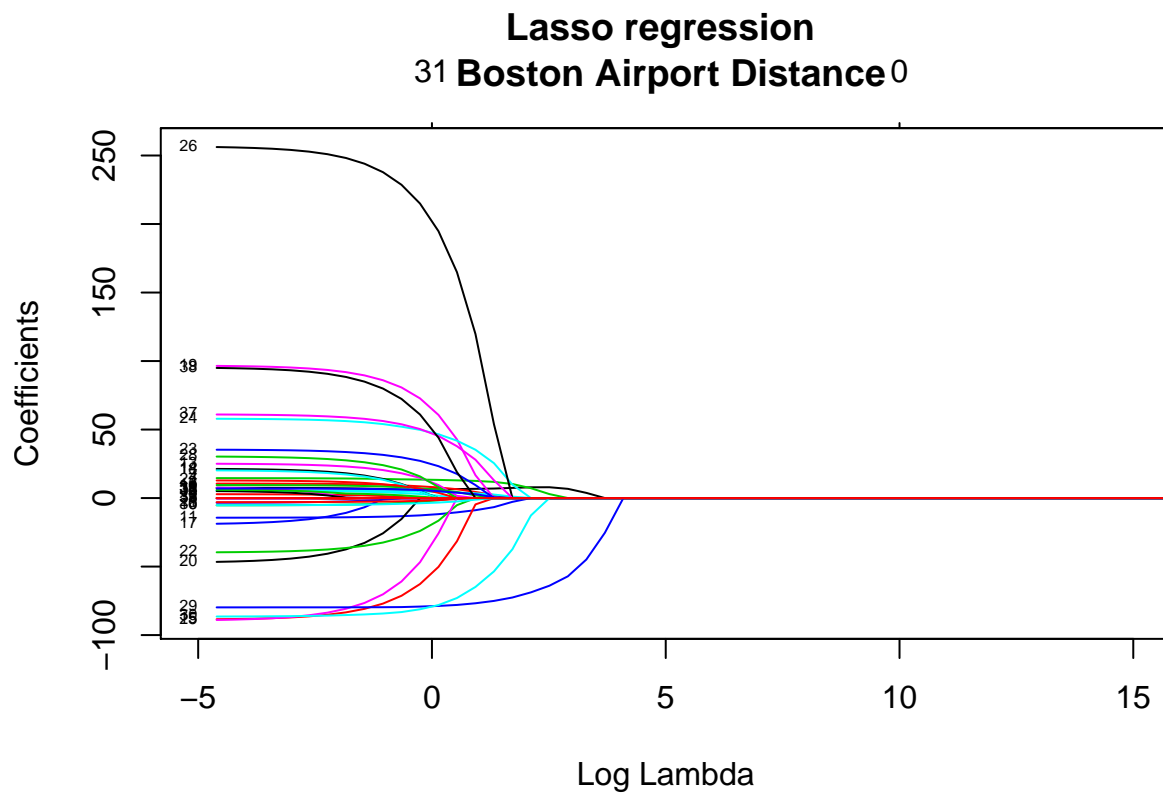
grid = 10^seq(15,-2, length = 100)
lasso.boston.data <- glmnet(x.boston.data,y.boston.data, alpha = 1, lambda = grid)
plot(lasso.boston.data, main = "Lasso regression \n Boston Data", label = TRUE, xvar = "lambda", xlim =
```



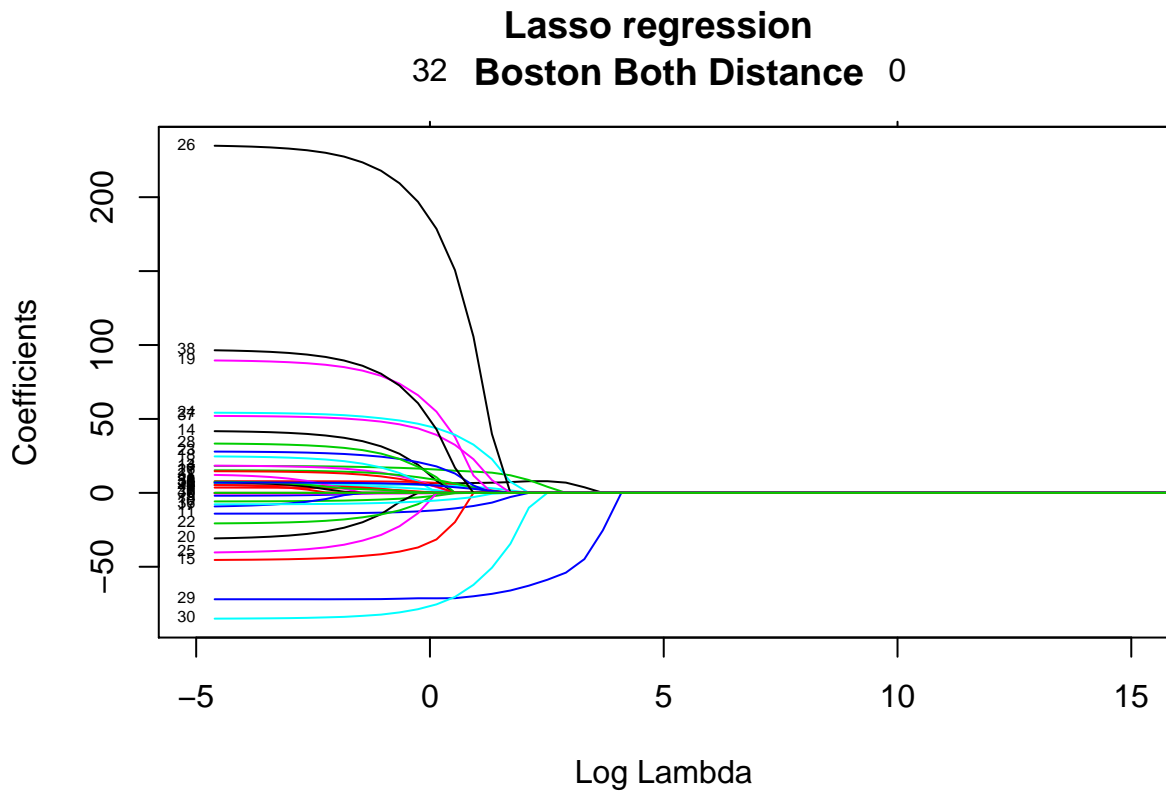
```
lasso.boston.ddowntown <- glmnet(x.boston.ddowntown,y.boston.ddowntown, alpha = 1, lambda = grid)
plot(lasso.boston.ddowntown, main = "Lasso regression \n Boston Downtown Distance", label = TRUE, xvar =
```



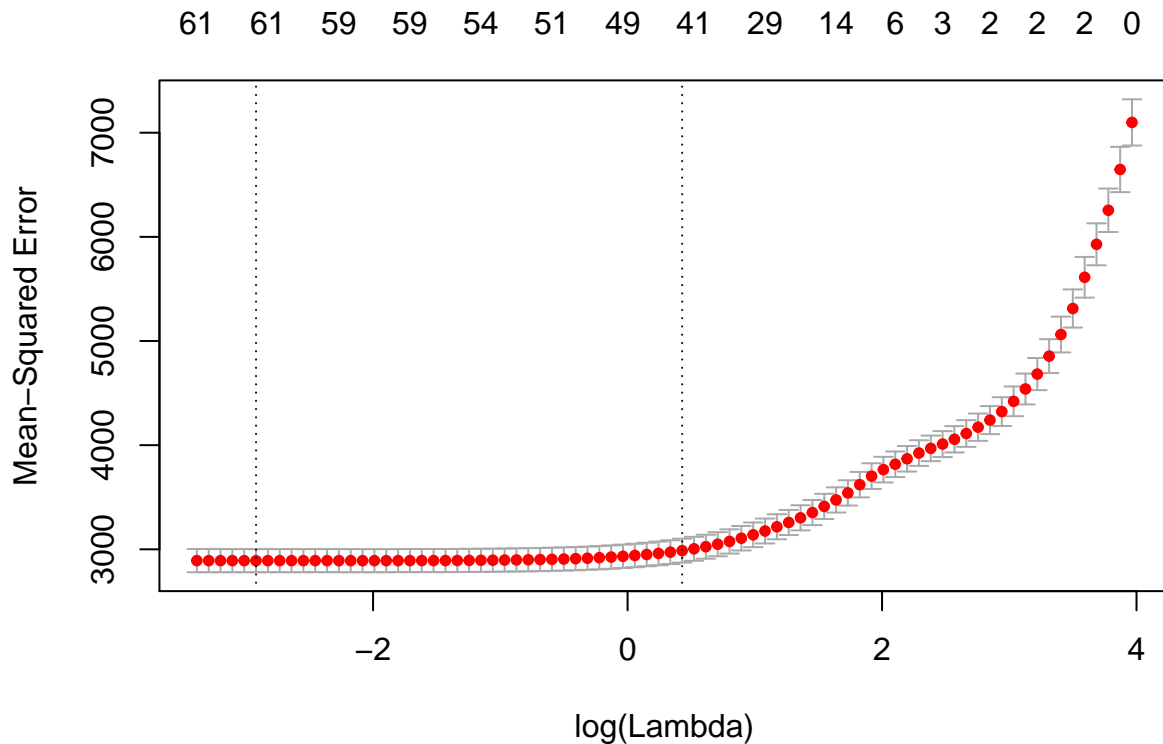
```
lasso.boston.dairport <- glmnet(x.boston.dairport, y.boston.dairport, alpha = 1, lambda = grid)
plot(lasso.boston.dairport, main = "Lasso regression \n Boston Airport Distance", label = TRUE, xvar =
```



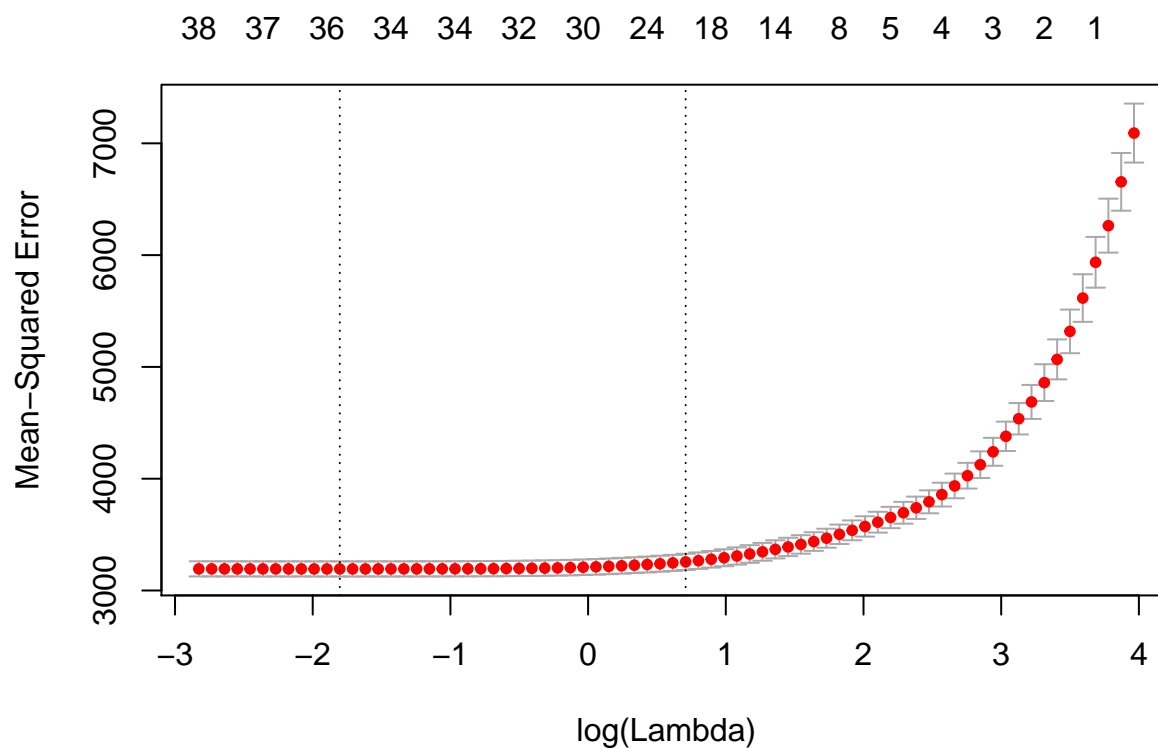
```
lasso.boston.dboth <- glmnet(x.boston.dboth,y.boston.dboth, alpha = 1, lambda = grid)
plot(lasso.boston.dboth, main = "Lasso regression \n Boston Both Distance", label = TRUE, xvar = "lambda")
```



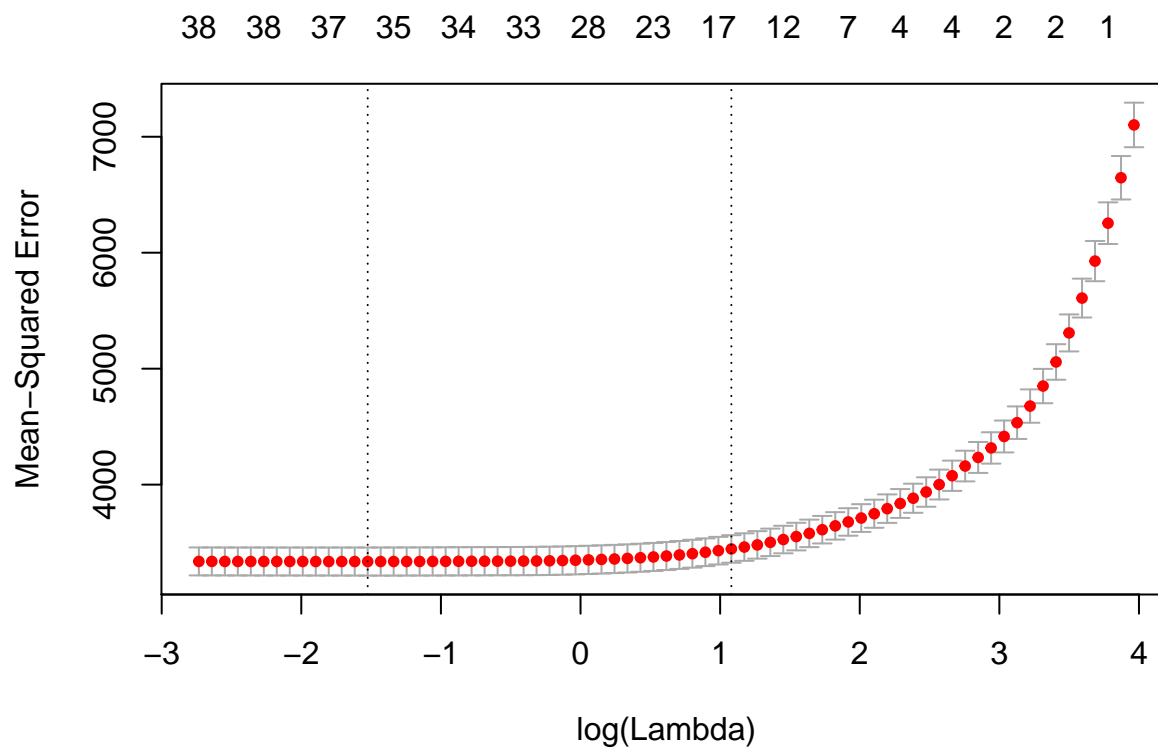
```
cv.out.data <- cv.glmnet(x.boston.data,y.boston.data,alpha = 1)
plot(cv.out.data)
```



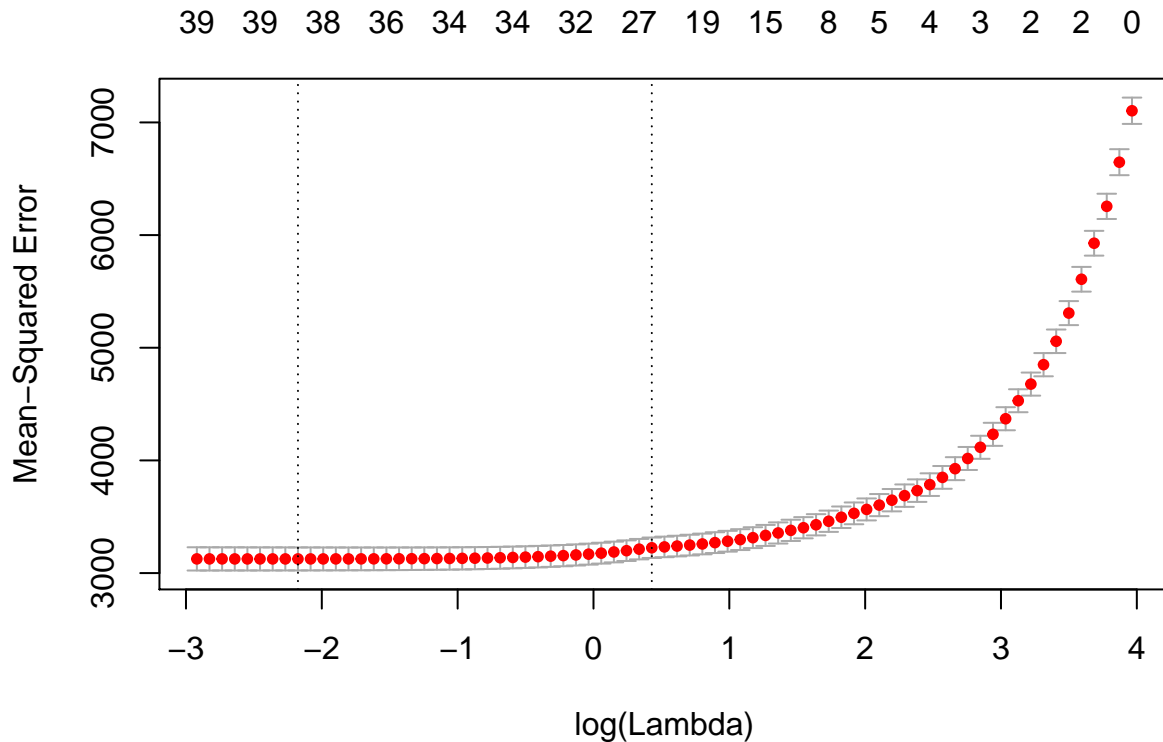

```
cv.out.ddowntown <- cv.glmnet(x.boston.ddowntown,y.boston.ddowntown,alpha = 1)
plot(cv.out.ddowntown)
```



```
cv.out.dairport <- cv.glmnet(x.boston.dairport,y.boston.dairport,alpha = 1)
plot(cv.out.dairport)
```



```
cv.out.dboth <- cv.glmnet(x.boston.dboth,y.boston.dboth,alpha = 1)
plot(cv.out.dboth)
```



```
bestlam.lasso.data <- cv.out.data$lambda.min
cat("Best lambda Boston Data" , bestlam.lasso.data, "\n")
```

```
## Best lambda Boston Data 0.05393749
```

```
cat("Best log lambda Boston Data", log(bestlam.lasso.data), "\n")
```

```
## Best log lambda Boston Data -2.91993
```

```
bestlam.lasso.ddowntown <- cv.out.ddowntown$lambda.min
cat("Best lambda Boston Downtown" , bestlam.lasso.ddowntown, "\n")
```

```
## Best lambda Boston Downtown 0.1647173
```

```
cat("Best log lambda Boston Downtown" , log(bestlam.lasso.ddowntown), "\n")
```

```
## Best log lambda Boston Downtown -1.803525
```

```
bestlam.lasso.dairport <- cv.out.dairport$lambda.min
cat("Best lambda Boston Airport" , bestlam.lasso.dairport, "\n")
```

```
## Best lambda Boston Airport 0.2177466
```

```
cat("Best log lambda Boston Airport" , log(bestlam.lasso.dairport), "\n")
```

```
## Best log lambda Boston Airport -1.524423
```

```
bestlam.lasso.dboth <- cv.out.dboth$lambda.min
cat("Best lambda Boston Both" , bestlam.lasso.dboth, "\n")
```

```
## Best lambda Boston Both 0.1135332
```

```

cat("Best log lambda Boston Both" , log(bestlam.lasso.dboth), "\n")

## Best log lambda Boston Both -2.17566
lasso.model.data <- glmnet(x.boston.data, y.boston.data, alpha=1, lambda = bestlam.lasso.data)
lasso.model.ddowntown <- glmnet(x.boston.ddowntown, y.boston.ddowntown, alpha=1, lambda = bestlam.lasso.data)
lasso.model.dairport <- glmnet(x.boston.dairport, y.boston.dairport, alpha=1, lambda = bestlam.lasso.data)
lasso.model.dboth <- glmnet(x.boston.dboth, y.boston.dboth, alpha=1, lambda = bestlam.lasso.dboth)

rmse <- function(test_data, model) {
  return(sqrt(mean((test_data$price - predict(model, newx = model.matrix(price~.,test_data)))^ 2)))
}

cat("RMSE for Boston Data using Lasso", rmse(boston.dummied.model_selection, lasso.model.data), "$\n")

## RMSE for Boston Data using Lasso 56.25078 $
cat("RMSE for Boston Downtown Data using Lasso", rmse(boston.ddowntown.dummied.model_selection, lasso.model.ddowntown), "$\n")

## RMSE for Boston Downtown Data using Lasso 59.17001 $
cat("RMSE for Boston Airport Data using Lasso", rmse(boston.dairport.dummied.model_selection, lasso.model.dairport), "$\n")

## RMSE for Boston Airport Data using Lasso 60.42277 $
cat("RMSE for Boston Both Data using Lasso", rmse(boston.dboth.dummied.model_selection, lasso.model.dboth), "$\n")

## RMSE for Boston Both Data using Lasso 58.62441 $

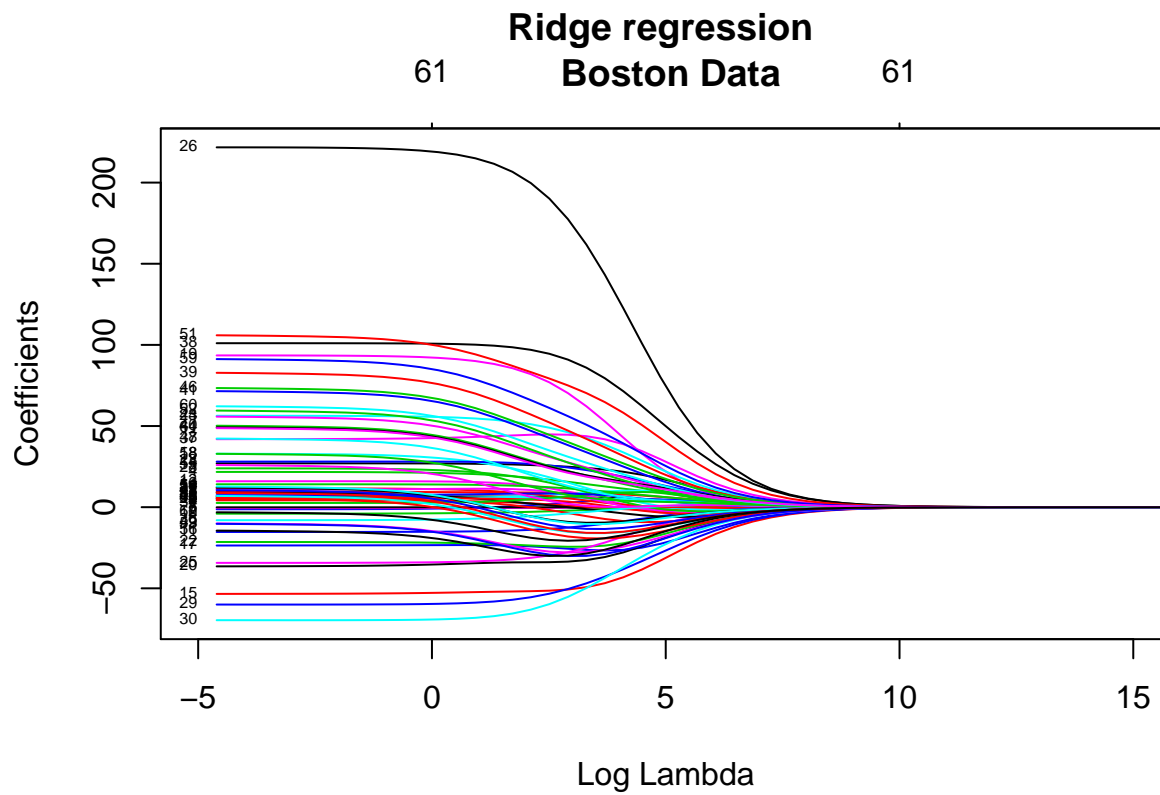
```

Ridge Regression

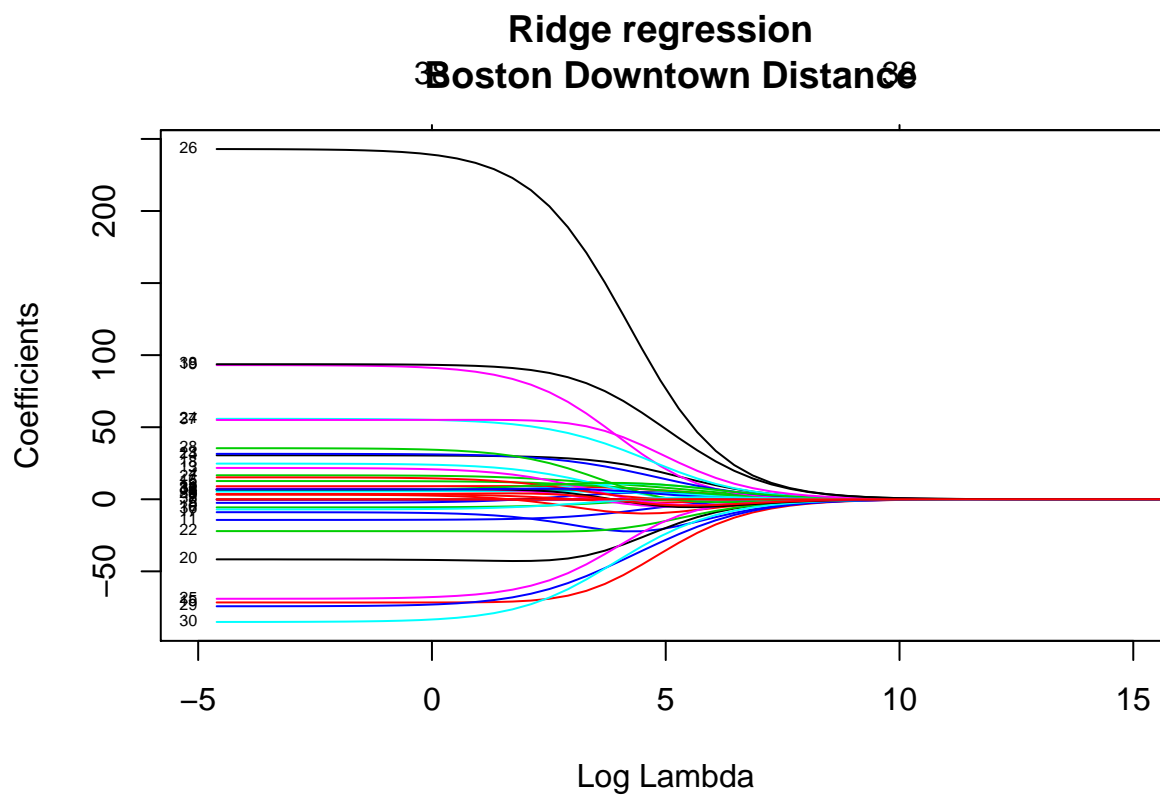
```

grid = 10^seq(15,-2, length = 100)
ridge.boston.data <- glmnet(x.boston.data,y.boston.data, alpha = 0, lambda = grid)
plot(ridge.boston.data, main = "Ridge regression \n Boston Data", label = TRUE, xvar = "lambda", xlim =

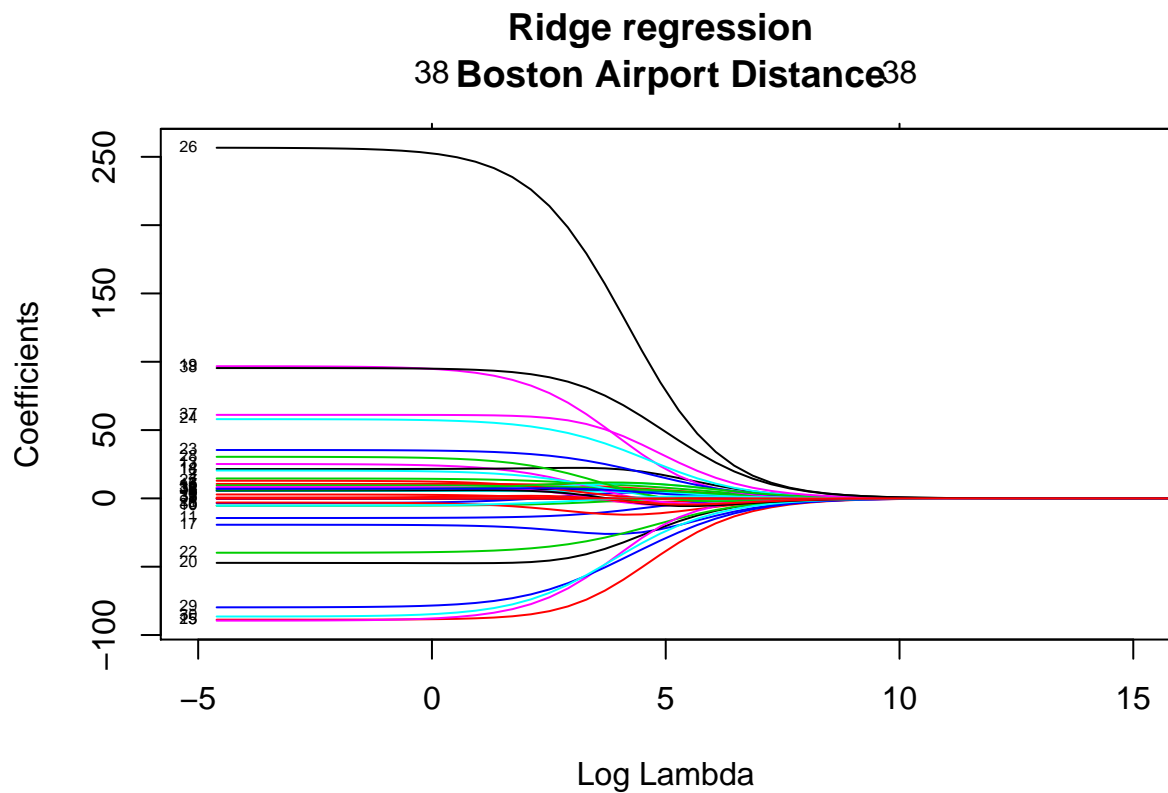
```



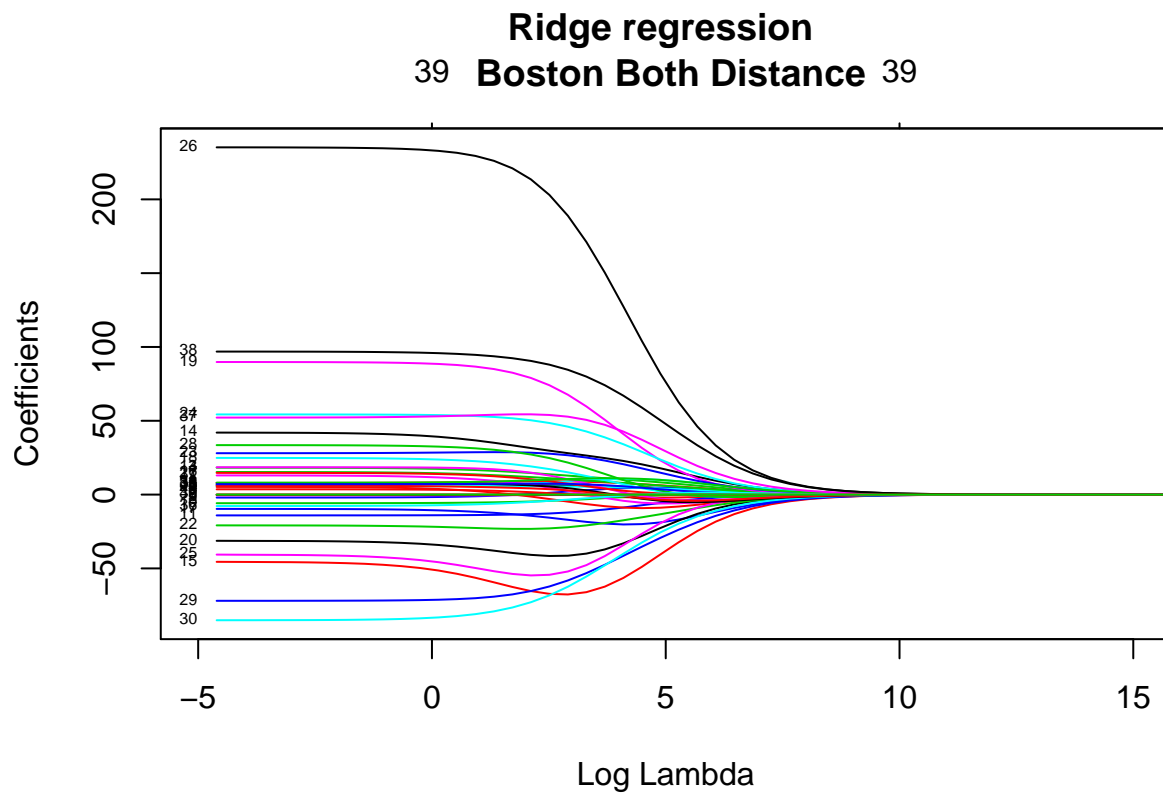
```
ridge.boston.downtown <- glmnet(x.boston.downtown, y.boston.downtown, alpha = 0, lambda = grid)
plot(ridge.boston.downtown, main = "Ridge regression \n Boston Downtown Distance", label = TRUE, xvar = "Log Lambda")
```



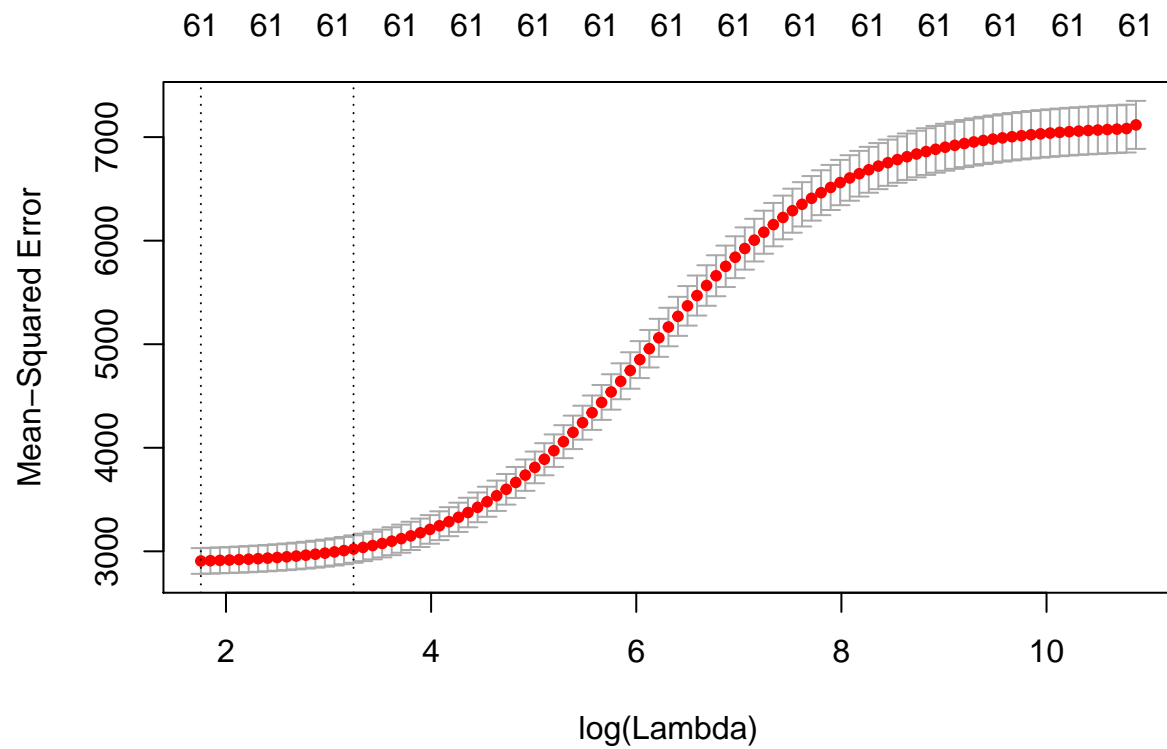
```
ridge.boston.dairport <- glmnet(x.boston.dairport,y.boston.dairport, alpha = 0, lambda = grid)
plot(ridge.boston.dairport, main = "Ridge regression \n Boston Airport Distance", label = TRUE, xvar =
```



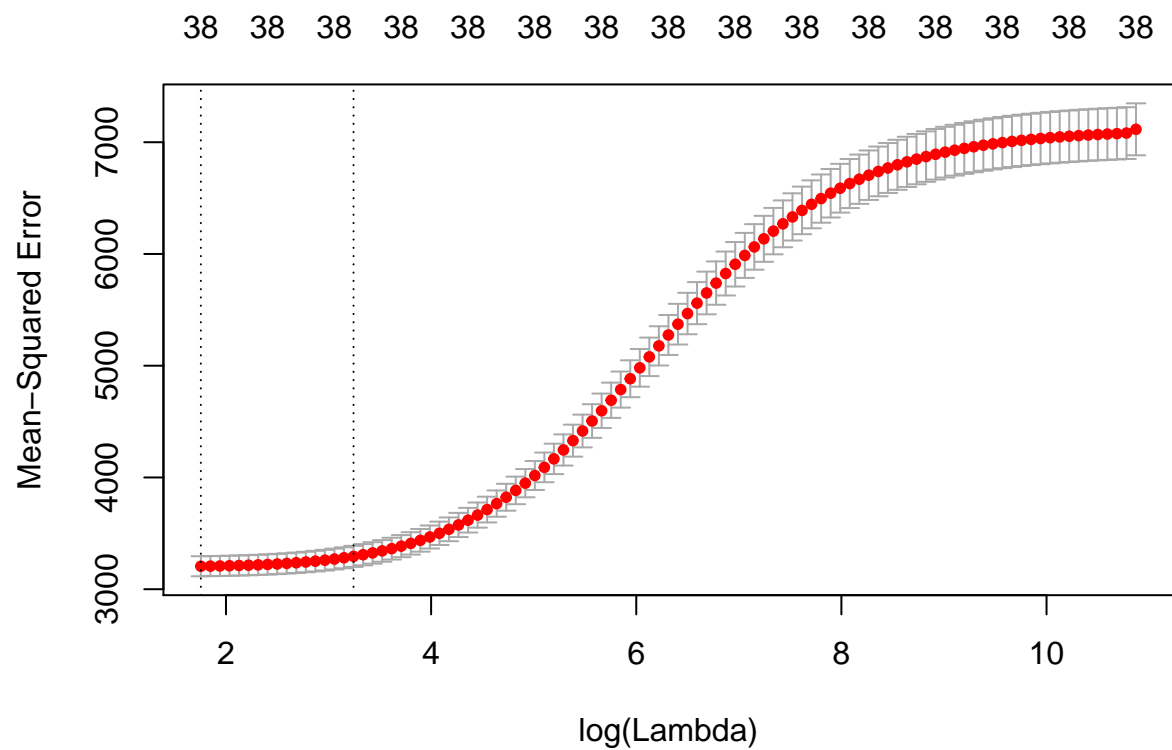
```
ridge.boston.dboth <- glmnet(x.boston.dboth,y.boston.dboth, alpha = 0, lambda = grid)
plot(ridge.boston.dboth, main = "Ridge regression \n Boston Both Distance", label = TRUE, xvar = "lambda")
```



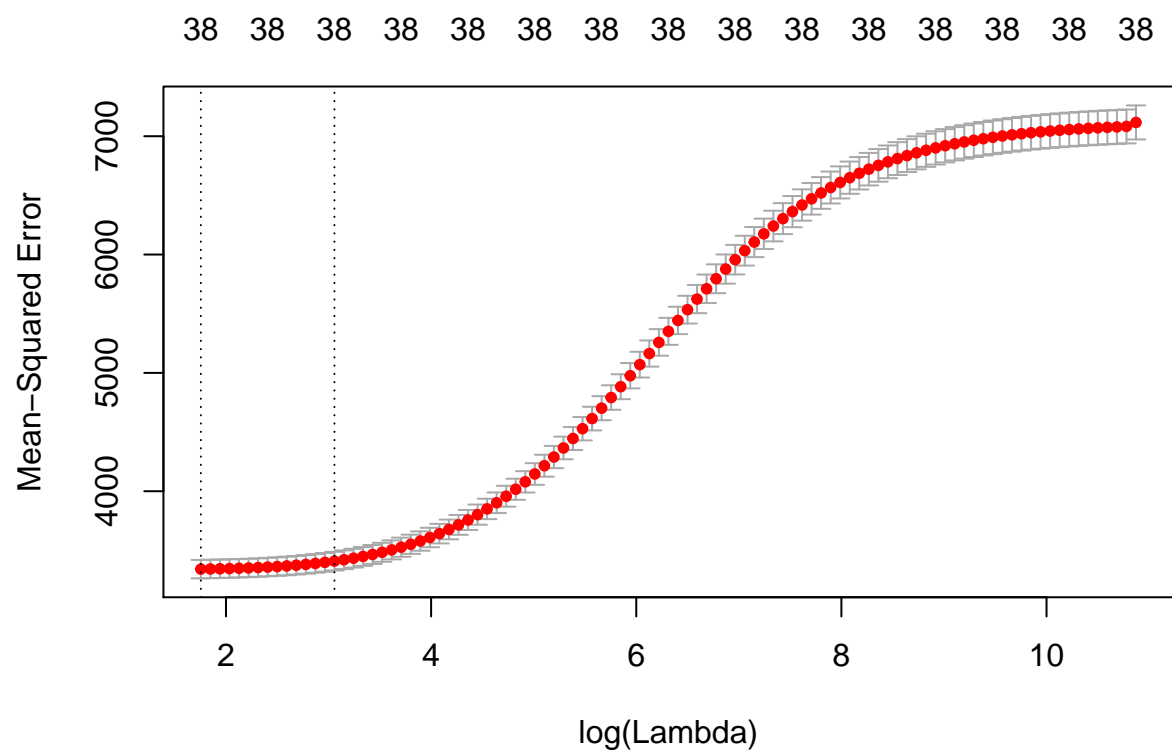
```
cv.out.data <- cv.glmnet(x.boston.data,y.boston.data,alpha = 0)
plot(cv.out.data)
```



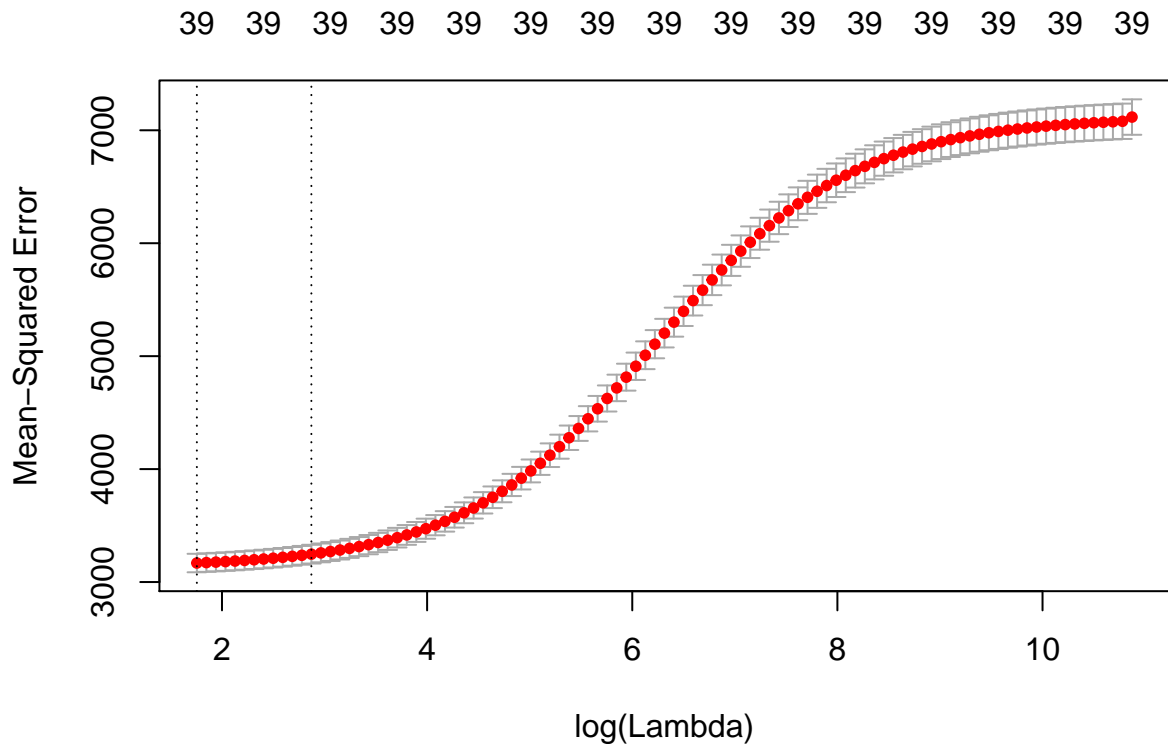
```
cv.out.ddowntown <- cv.glmnet(x.boston.ddowntown,y.boston.ddowntown,alpha = 0)
plot(cv.out.ddowntown)
```



```
cv.out.dairport <- cv.glmnet(x.boston.dairport,y.boston.dairport,alpha = 0)
plot(cv.out.dairport)
```



```
cv.out.dboth <- cv.glmnet(x.boston.dboth,y.boston.dboth,alpha = 0)
plot(cv.out.dboth)
```

```
bestlam.ridge.data <- cv.out.data$lambda.min
cat("Best lambda Boston Data" , bestlam.ridge.data, "\n")
```

```
## Best lambda Boston Data 5.78354
```

```
cat("Best log lambda Boston Data", log(bestlam.ridge.data), "\n")
```

```
## Best log lambda Boston Data 1.755016
```

```
bestlam.ridge.ddowntown <- cv.out.ddowntown$lambda.min
cat("Best lambda Boston Downtown" , bestlam.ridge.ddowntown, "\n")
```

```
## Best lambda Boston Downtown 5.78354
```

```
cat("Best log lambda Boston Downtown" , log(bestlam.ridge.ddowntown), "\n")
```

```
## Best log lambda Boston Downtown 1.755016
```

```
bestlam.ridge.dairport <- cv.out.dairport$lambda.min
cat("Best lambda Boston Airport" , bestlam.ridge.dairport, "\n")
```

```
## Best lambda Boston Airport 5.78354
```

```
cat("Best log lambda Boston Airport" , log(bestlam.ridge.dairport), "\n")
```

```
## Best log lambda Boston Airport 1.755016
```

```
bestlam.ridge.dboth <- cv.out.dboth$lambda.min
cat("Best lambda Boston Both" , bestlam.ridge.dboth, "\n")
```

```
## Best lambda Boston Both 5.78354
```

```
cat("Best log lambda Boston Both" , log(bestlam.ridge.dboth), "\n")
```

```
## Best log lambda Boston Both 1.755016
```

```

ridge.model.data <- glmnet(x.boston.data, y.boston.data, alpha=1, lambda = bestlam.ridge.data)
ridge.model.ddowntown <- glmnet(x.boston.ddowntown, y.boston.ddowntown, alpha=1, lambda = bestlam.ridge.data)
ridge.model.dairport <- glmnet(x.boston.dairport, y.boston.dairport, alpha=1, lambda = bestlam.ridge.data)
ridge.model.dboth <- glmnet(x.boston.dboth, y.boston.dboth, alpha=1, lambda = bestlam.ridge.dboth)

rmse <- function(test_data, model) {
  return(sqrt(mean((test_data$price - predict(model, newx = model.matrix(price~.,test_data)))^ 2)))
}

cat("RMSE for Boston Data using Ridge", rmse(boston.dummied.model_selection, ridge.model.data), "$\n")

## RMSE for Boston Data using Ridge 62.27309 $
cat("RMSE for Boston Downtown Data using Ridge", rmse(boston.ddowntown.dummied.model_selection, ridge.model.ddowntown), "$\n")

## RMSE for Boston Downtown Data using Ridge 61.98493 $
cat("RMSE for Boston Airport Data using Ridge", rmse(boston.dairport.dummied.model_selection, ridge.model.dairport), "$\n")

## RMSE for Boston Airport Data using Ridge 63.11274 $
cat("RMSE for Boston Both Data using Ridge", rmse(boston.dboth.dummied.model_selection, ridge.model.dboth), "$\n")

## RMSE for Boston Both Data using Ridge 61.98493 $
cat("RMSE for Boston Data using Lasso - Validation Set 1", rmse(boston.dummied.validation_1, lasso.model.data), "$\n")

## RMSE for Boston Data using Lasso - Validation Set 1 56.81055 $
cat("RMSE for Boston Downtown Data using Lasso - Validation Set 1", rmse(boston.ddowntown.dummied.validation_1, lasso.model.ddowntown), "$\n")

## RMSE for Boston Downtown Data using Lasso - Validation Set 1 58.98283 $
cat("RMSE for Boston Airport Data using Lasso - Validation Set 1", rmse(boston.dairport.dummied.validation_1, lasso.model.dairport), "$\n")

## RMSE for Boston Airport Data using Lasso - Validation Set 1 60.7616 $
cat("RMSE for Boston Both Data using Lasso - Validation Set 1", rmse(boston.dboth.dummied.validation_1, lasso.model.dboth), "$\n")

## RMSE for Boston Both Data using Lasso - Validation Set 1 58.15547 $
a = predict(lasso.model.data, s = bestlam.lasso.data, type = "coefficients")
cat("Predictors for Lasso Boston Data", nrow(a), "\n")

## Predictors for Lasso Boston Data 63
a = predict(lasso.model.ddowntown, s = bestlam.lasso.ddowntown, type = "coefficients")
cat("Predictors for Lasso Boston Data", nrow(a), "\n")

## Predictors for Lasso Boston Data 40
a = predict(lasso.model.dairport, s = bestlam.lasso.dairport, type = "coefficients")
cat("Predictors for Lasso Boston Data", nrow(a), "\n")

## Predictors for Lasso Boston Data 40
a = predict(lasso.model.dboth, s = bestlam.lasso.dboth, type = "coefficients")
cat("Predictors for Lasso Boston Data", nrow(a), "\n")

## Predictors for Lasso Boston Data 41

```

```

plot_predicts <-function(predicted_price, real_price, text, color){
  plot(x = real_price, y = predict_price, xlab = "actual price",
       ylab = "predicted prices", main = text,xlim= c(0,500), ylim= c(0,500), col = color, asp=1)
  abline(a = 0, b = 1)
}

par(mfrow=c(2,2))
predict_price = predict(lasso.model.data, newx = model.matrix(price~.,boston.dummied.model_selection))
real_price = boston.dummied.model_selection$price
plot_predicts(predict_price, real_price, "Lasso prediction for Boston Data", "red")

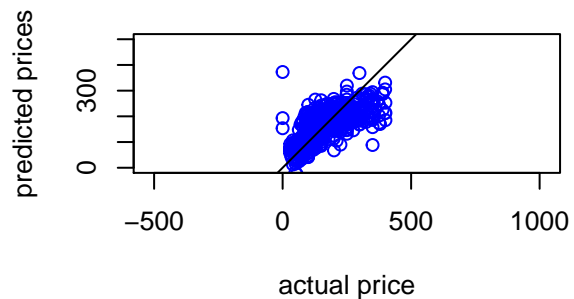
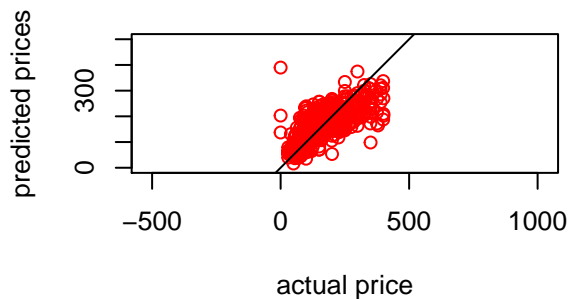
predict_price = predict(lasso.model.downtown, newx = model.matrix(price~.,boston.downtown.dummied.model_selection))
real_price = boston.downtown.dummied.model_selection$price
plot_predicts(predict_price, real_price, "Lasso prediction for Boston Downtown Data", "blue")

predict_price = predict(lasso.model.airport, newx = model.matrix(price~.,boston.airport.dummied.model_selection))
real_price = boston.airport.dummied.model_selection$price
plot_predicts(predict_price, real_price, "Lasso prediction for Boston Airport Data", "green")

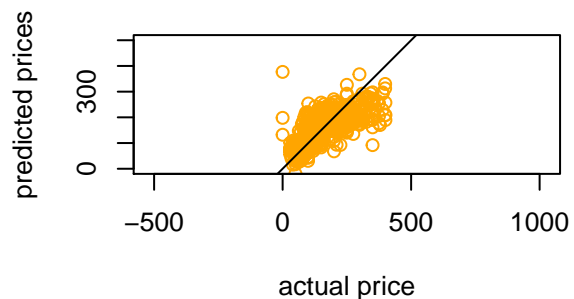
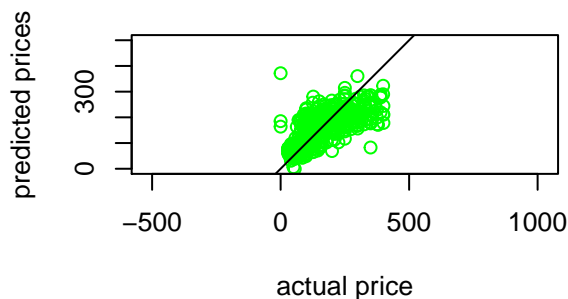
predict_price = predict(lasso.model.both, newx = model.matrix(price~.,boston.both.dummied.model_selection))
real_price = boston.both.dummied.model_selection$price
plot_predicts(predict_price, real_price, "Lasso prediction for Boston Both Data", "orange")

```

Lasso prediction for Boston Data Lasso prediction for Boston Downtown Data



Lasso prediction for Boston Airport Data Lasso prediction for Boston Both Data



GAM

```
# Helper function

## @param test_data: data frame representing the test dataset
## @param model: GAM to be evaluated
## @returns: the root mean square error of the given GAM's predictions
rmse <- function(test_data, model) {
  return(sqrt(mean((test_data$price - predict.gam(model, test_data)) ^ 2)))
}
```

Variable selection methods

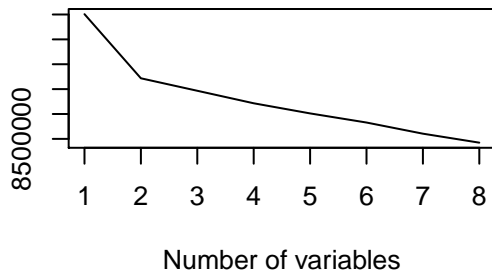
Exploring three variable selection methods on the original dataset.

Regression subset selection

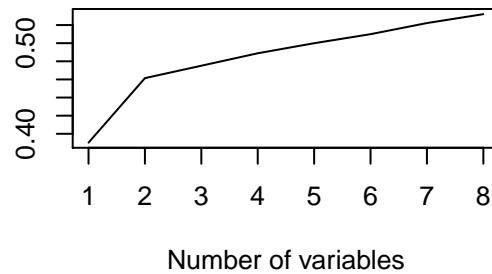
```
reg1 <- regsubsets(price~., data = boston.data.training, really.big = T)
reg1.summary <- summary(reg1)

par(mfrow = c(2,2))
plot(reg1.summary$rss, xlab = "Number of variables", ylab = "Residual Sum of Squares (RSS)", type = "l")
plot(reg1.summary$adjr2, xlab = "Number of variables", ylab = "Adjusted R square", type = "l")
plot(reg1.summary$cp, xlab = "Number of variables", ylab = "CP", type = "l")
plot(reg1.summary$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
```

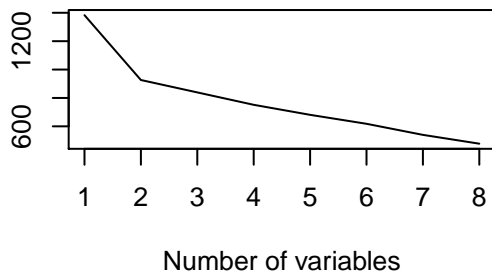
Residual Sum of Squares (RSS)



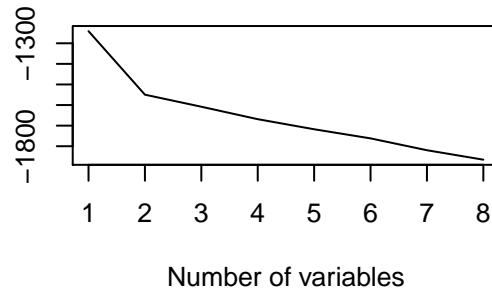
Adjusted R square



CP



BIC



```
which.min(reg1.summary$bic)
```

```
## [1] 8
```

```
# Results: neighborhood, room_type, bedrooms
```

```
# GAM based on the above predictors
```

```
gam.var1 <- gam(price ~ neighborhood + room_type + s(bedrooms, k=5, bs="cr"), data=boston.data.training,
summary(gam.var1)
```

```
##
```

```
## Family: gaussian
```

```
## Link function: identity
```

```
##
```

```
## Formula:
```

```
## price ~ neighborhood + room_type + s(bedrooms, k = 5, bs = "cr")
```

```
##
```

```
## Parametric coefficients:
```

```
##
```

	Estimate	Std. Error	t value	Pr(> t)
## (Intercept)	141.9999	4.7695	29.772	< 2e-16
## neighborhoodBack Bay	77.4922	6.1283	12.645	< 2e-16
## neighborhoodBay Village	58.8643	15.5698	3.781	0.00016
## neighborhoodBeacon Hill	66.5157	6.5376	10.174	< 2e-16
## neighborhoodBrighton	6.7294	6.1968	1.086	0.27761
## neighborhoodCharlestown	56.6379	8.1568	6.944	4.85e-12
## neighborhoodChinatown	53.9698	8.4620	6.378	2.13e-10
## neighborhoodDorchester	5.5711	5.9301	0.939	0.34758
## neighborhoodDowntown	73.7092	6.5780	11.205	< 2e-16
## neighborhoodEast Boston	8.4492	6.3624	1.328	0.18430

```

## neighborhoodFenway          41.7047      6.1351      6.798 1.32e-11
## neighborhoodHyde Park       -8.9881     11.9928     -0.749 0.45365
## neighborhoodJamaica Plain   14.3229      5.7816      2.477 0.01330
## neighborhoodLeather District 119.0248     28.2959      4.206 2.69e-05
## neighborhoodLongwood Medical Area 55.9726     28.2588      1.981 0.04773
## neighborhoodMattapan        -5.3499     13.8659     -0.386 0.69966
## neighborhoodMission Hill    11.0646      7.8554      1.409 0.15910
## neighborhoodNorth End       19.0931      7.4269      2.571 0.01020
## neighborhoodRoslindale      -8.2625      8.4977     -0.972 0.33098
## neighborhoodRoxbury         7.6340      6.6357      1.150 0.25007
## neighborhoodSouth Boston    36.7689      6.5236      5.636 1.93e-08
## neighborhoodSouth Boston Waterfront 93.1898     10.6667      8.737 < 2e-16
## neighborhoodSouth End       61.9936      6.1380     10.100 < 2e-16
## neighborhoodWest End        54.4899     11.5385      4.722 2.46e-06
## neighborhoodWest Roxbury     0.1855     10.0652      0.018 0.98530
## room_typePrivate room       -64.6402      2.9962    -21.574 < 2e-16
## room_typeShared room        -75.5122     10.1753     -7.421 1.58e-13
##
## (Intercept)                ***
## neighborhoodBack Bay        ***
## neighborhoodBay Village     ***
## neighborhoodBeacon Hill     ***
## neighborhoodBrighton
## neighborhoodCharlestown     ***
## neighborhoodChinatown       ***
## neighborhoodDorchester
## neighborhoodDowntown        ***
## neighborhoodEast Boston
## neighborhoodFenway          ***
## neighborhoodHyde Park
## neighborhoodJamaica Plain   *
## neighborhoodLeather District ***
## neighborhoodLongwood Medical Area *
## neighborhoodMattapan
## neighborhoodMission Hill
## neighborhoodNorth End       *
## neighborhoodRoslindale
## neighborhoodRoxbury
## neighborhoodSouth Boston    ***
## neighborhoodSouth Boston Waterfront ***
## neighborhoodSouth End       ***
## neighborhoodWest End        ***
## neighborhoodWest Roxbury
## room_typePrivate room       ***
## room_typeShared room        ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##              edf Ref.df      F p-value
## s(bedrooms) 3.447  3.795 150.1 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##

```

```
## R-sq.(adj) = 0.563   Deviance explained = 56.8%
## -REML = 13723   Scale est. = 3110.5   n = 2539
# R^2: 0.563

gam.var1.rmse <- rmse(boston.data.model_selection, gam.var1)
gam.var1.rmse # 57.4993

## [1] 57.4993
```

Forward Selection

```
null <- lm(price~1, data=boston.data.training)
full <- lm(price~., data=boston.data.training)

forward_selection <- step(null, scope=list(lower=null, upper=full), direction="forward")

## Start:  AIC=22521.81
## price ~ 1
##
##               Df Sum of Sq    RSS   AIC
## + room_type      2   7477354 10580368 21168
## + accommodates    1   5243844 12813878 21653
## + neighborhood   24   4619557 13438165 21820
## + beds           1   3155623 14902099 22036
## + bedrooms       1   2992742 15064980 22064
## + guests_included 1   1557511 16500211 22295
## + is_business_travel_ready 1 1148253 16909469 22357
## + property_type   16 1199234 16858488 22379
## + cancellation_policy 4   630948 17426775 22440
## + bathrooms      1   570100 17487622 22442
## + X              1   200775 17856948 22495
## + bed_type        4   216139 17841584 22499
## + number_of_reviews 1    62511 17995211 22515
## + instant_bookable 1    60881 17996841 22515
## + host_is_superhost 1    42853 18014869 22518
## <none>                      18057722 22522
## + host_identity_verified 1    13545 18044177 22522
## + minimum_nights    1     1741 18055981 22524
##
## Step:  AIC=21168.53
## price ~ room_type
##
##               Df Sum of Sq    RSS   AIC
## + accommodates    1   1113390  9466978 20888
## + bedrooms       1   1098259  9482108 20892
## + neighborhood   24   981822  9598546 20969
## + beds           1   744614  9835754 20985
## + bathrooms      1   427798 10152570 21066
## + guests_included 1   390851 10189517 21075
## + property_type   16   366792 10213576 21111
## + cancellation_policy 4   117898 10462470 21148
## + instant_bookable 1    79146 10501222 21152
## + number_of_reviews 1    39960 10540408 21161
```



```

## + minimum_nights      1      38454 10541914 21161
## + is_business_travel_ready 1      23840 10556528 21165
## <none>                                10580368 21168
## + X                      1         6263 10574105 21169
## + host_is_superhost      1         1748 10578620 21170
## + host_identity_verified  1          482 10579886 21170
## + bed_type               4       12712 10567656 21174
##
## Step:  AIC=20888.21
## price ~ room_type + accommodates
##
##              Df Sum of Sq      RSS      AIC
## + neighborhood      24   1434186 8032792 20519
## + property_type      16   345329 9121649 20826
## + bedrooms           1   166560 9300418 20845
## + cancellation_policy 4   147637 9319341 20856
## + instant_bookable    1   121042 9345936 20858
## + bathrooms           1    74763 9392215 20870
## + number_of_reviews    1    53794 9413184 20876
## + guests_included      1    38644 9428334 20880
## + X                   1    10349 9456629 20887
## <none>                                9466978 20888
## + minimum_nights      1      6866 9460113 20888
## + host_identity_verified 1     1012 9465966 20890
## + beds                1       468 9466510 20890
## + host_is_superhost    1       179 9466799 20890
## + is_business_travel_ready 1         1 9466977 20890
## + bed_type             4     4942 9462036 20895
##
## Step:  AIC=20519.11
## price ~ room_type + accommodates + neighborhood
##
##              Df Sum of Sq      RSS      AIC
## + bedrooms           1   353970 7678822 20407
## + property_type      16   270772 7762020 20464
## + cancellation_policy 4   191046 7841746 20466
## + instant_bookable    1   138697 7894095 20477
## + bathrooms           1    88977 7943815 20493
## + guests_included      1    81470 7951321 20495
## + number_of_reviews    1    33899 7998892 20510
## + minimum_nights      1    13282 8019509 20517
## + beds                1    12465 8020327 20517
## <none>                                8032792 20519
## + host_is_superhost    1     3930 8028862 20520
## + X                    1     1890 8030901 20520
## + is_business_travel_ready 1       978 8031814 20521
## + host_identity_verified 1         4 8032788 20521
## + bed_type             4       487 8032305 20527
##
## Step:  AIC=20406.69
## price ~ room_type + accommodates + neighborhood + bedrooms
##
##              Df Sum of Sq      RSS      AIC
## + property_type      16   274726 7404096 20346

```

```

## + cancellation_policy      4    172742 7506080 20357
## + instant_bookable         1    133193 7545629 20364
## + guests_included          1     79699 7599123 20382
## + bathrooms                 1     30843 7647978 20398
## + minimum_nights           1     29218 7649603 20399
## + number_of_reviews         1     18830 7659992 20402
## + host_is_superhost         1       6755 7672066 20406
## <none>                      1       7678822 20407
## + X                         1       3337 7675485 20408
## + is_business_travel_ready  1        954 7677868 20408
## + beds                     1        798 7678024 20408
## + host_identity_verified     1        298 7678524 20409
## + bed_type                   4       2423 7676399 20414
##
## Step: AIC=20346.19
## price ~ room_type + accommodates + neighborhood + bedrooms +
##   property_type
##
##              Df Sum of Sq    RSS    AIC
## + cancellation_policy      4    155343 7248752 20300
## + instant_bookable         1    137533 7266562 20301
## + guests_included          1     71180 7332916 20324
## + minimum_nights           1     25664 7378431 20339
## + bathrooms                 1     24768 7379328 20340
## + number_of_reviews         1     16366 7387729 20343
## + host_is_superhost         1       6344 7397752 20346
## <none>                      1       7404096 20346
## + X                         1       4646 7399450 20347
## + is_business_travel_ready  1       1836 7402260 20348
## + host_identity_verified     1        293 7403803 20348
## + beds                     1          8 7404088 20348
## + bed_type                   4       3012 7401083 20353
##
## Step: AIC=20300.35
## price ~ room_type + accommodates + neighborhood + bedrooms +
##   property_type + cancellation_policy
##
##              Df Sum of Sq    RSS    AIC
## + instant_bookable         1    121214 7127538 20260
## + guests_included          1     64392 7184361 20280
## + bathrooms                 1     33350 7215402 20291
## + number_of_reviews         1     16252 7232501 20297
## + minimum_nights           1     14770 7233983 20297
## + host_is_superhost         1       6312 7242440 20300
## + X                         1       6225 7242527 20300
## <none>                      1       7248752 20300
## + is_business_travel_ready  1       3934 7244818 20301
## + host_identity_verified     1       2694 7246058 20301
## + beds                     1        152 7248600 20302
## + bed_type                   4       2031 7246721 20308
##
## Step: AIC=20259.54
## price ~ room_type + accommodates + neighborhood + bedrooms +
##   property_type + cancellation_policy + instant_bookable

```

```

##
##           Df Sum of Sq      RSS      AIC
## + guests_included      1      71043 7056496 20236
## + minimum_nights      1      24288 7103250 20253
## + bathrooms           1      24025 7103513 20253
## + is_business_travel_ready 1      10230 7117308 20258
## + number_of_reviews     1       7556 7119982 20259
## + host_identity_verified 1       6616 7120922 20259
## <none>                      7127538 20260
## + X                       1       4561 7122977 20260
## + host_is_superhost     1       4230 7123308 20260
## + beds                 1        163 7127375 20262
## + bed_type             4       1840 7125698 20267
##
## Step: AIC=20236.1
## price ~ room_type + accommodates + neighborhood + bedrooms +
##         property_type + cancellation_policy + instant_bookable +
##         guests_included
##
##           Df Sum of Sq      RSS      AIC
## + bathrooms      1      32231 7024265 20226
## + minimum_nights 1      25366 7031129 20229
## + number_of_reviews 1     10844 7045651 20234
## + host_identity_verified 1      7011 7049485 20236
## <none>                      7056496 20236
## + is_business_travel_ready 1      4773 7051722 20236
## + X               1      4530 7051966 20236
## + host_is_superhost 1      2009 7054486 20237
## + beds            1      1528 7054967 20238
## + bed_type        4      1584 7054911 20244
##
## Step: AIC=20226.48
## price ~ room_type + accommodates + neighborhood + bedrooms +
##         property_type + cancellation_policy + instant_bookable +
##         guests_included + bathrooms
##
##           Df Sum of Sq      RSS      AIC
## + minimum_nights      1    26895.3 6997369 20219
## + number_of_reviews     1     9730.4 7014534 20225
## + host_identity_verified 1     6887.1 7017378 20226
## + is_business_travel_ready 1    5534.4 7018730 20226
## <none>                      7024265 20226
## + X                   1    4356.0 7019909 20227
## + host_is_superhost     1    2222.4 7022042 20228
## + beds                 1    1090.2 7023174 20228
## + bed_type             4    1570.3 7022694 20234
##
## Step: AIC=20218.74
## price ~ room_type + accommodates + neighborhood + bedrooms +
##         property_type + cancellation_policy + instant_bookable +
##         guests_included + bathrooms + minimum_nights
##
##           Df Sum of Sq      RSS      AIC
## + number_of_reviews     1    12718.8 6984651 20216

```

```

## + host_identity_verified      1      7691.7 6989678 20218
## <none>                        6997369 20219
## + is_business_travel_ready   1      4388.0 6992981 20219
## + X                           1      3623.3 6993746 20219
## + host_is_superhost          1      2412.9 6994956 20220
## + beds                       1       722.4 6996647 20220
## + bed_type                    4      1072.8 6996297 20226
##
## Step:  AIC=20216.12
## price ~ room_type + accommodates + neighborhood + bedrooms +
##         property_type + cancellation_policy + instant_bookable +
##         guests_included + bathrooms + minimum_nights + number_of_reviews
##
##              Df Sum of Sq      RSS      AIC
## + is_business_travel_ready  1      8437.8 6976213 20215
## + host_is_superhost         1      7061.9 6977589 20216
## + host_identity_verified     1      5663.1 6978987 20216
## <none>                       6984651 20216
## + X                         1      4795.6 6979855 20216
## + beds                     1       731.7 6983919 20218
## + bed_type                  4      1079.4 6983571 20224
##
## Step:  AIC=20215.05
## price ~ room_type + accommodates + neighborhood + bedrooms +
##         property_type + cancellation_policy + instant_bookable +
##         guests_included + bathrooms + minimum_nights + number_of_reviews +
##         is_business_travel_ready
##
##              Df Sum of Sq      RSS      AIC
## + host_identity_verified     1      7033.2 6969180 20214
## <none>                       6976213 20215
## + host_is_superhost          1      4628.0 6971585 20215
## + X                         1      4525.2 6971688 20215
## + beds                     1      1110.5 6975102 20217
## + bed_type                  4      1087.9 6975125 20223
##
## Step:  AIC=20214.49
## price ~ room_type + accommodates + neighborhood + bedrooms +
##         property_type + cancellation_policy + instant_bookable +
##         guests_included + bathrooms + minimum_nights + number_of_reviews +
##         is_business_travel_ready + host_identity_verified
##
##              Df Sum of Sq      RSS      AIC
## + host_is_superhost          1      5779.8 6963400 20214
## <none>                       6969180 20214
## + X                         1      5068.1 6964112 20215
## + beds                     1      1252.9 6967927 20216
## + bed_type                  4      1240.9 6967939 20222
##
## Step:  AIC=20214.38
## price ~ room_type + accommodates + neighborhood + bedrooms +
##         property_type + cancellation_policy + instant_bookable +
##         guests_included + bathrooms + minimum_nights + number_of_reviews +
##         is_business_travel_ready + host_identity_verified + host_is_superhost

```

```
##
##           Df Sum of Sq      RSS   AIC
## <none>                6963400 20214
## + X              1    5027.2 6958373 20214
## + beds           1    1466.6 6961933 20216
## + bed_type       4    1297.5 6962102 20222

# Results: price ~ room_type + accommodates + neighborhood + bedrooms + property_type + cancellation_po

# The same predictors were obtained when "backward" and "both" directions for steps selctions as well.

# GAM based on the above predictors
gam.var2 <- gam(price ~ room_type + accommodates + neighborhood + bedrooms + property_type + cancellati
summary(gam.var2)

##
## Family: gaussian
## Link function: identity
##
## Formula:
## price ~ room_type + accommodates + neighborhood + bedrooms +
##   property_type + cancellation_policy + instant_bookable +
##   s(guests_included, bs = "cr") + s(bathrooms, bs = "cr") +
##   s(minimum_nights, bs = "cr") + s(number_of_reviews, bs = "cr") +
##   is_business_travel_ready + host_identity_verified + host_is_superhost
##
## Parametric coefficients:
##
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)      97.6875     5.8839  16.602 < 2e-16
## room_typePrivate room      -58.5335     3.2102 -18.234 < 2e-16
## room_typeShared room      -67.4184     9.7726  -6.899 6.64e-12
## accommodates         6.1220     0.9721   6.298 3.56e-10
## neighborhoodBack Bay      83.2301     5.9168  14.067 < 2e-16
## neighborhoodBay Village   49.9878    14.8640   3.363 0.000783
## neighborhoodBeacon Hill   71.4754     6.2611  11.416 < 2e-16
## neighborhoodBrighton      7.3488     5.9401   1.237 0.216145
## neighborhoodCharlestown   56.5329     7.8425   7.209 7.48e-13
## neighborhoodChinatown     48.9366     8.1261   6.022 1.98e-09
## neighborhoodDorchester     4.8598     5.6759   0.856 0.391959
## neighborhoodDowntown      73.4380     6.3226  11.615 < 2e-16
## neighborhoodEast Boston   10.9405     6.1863   1.769 0.077099
## neighborhoodFenway        41.5173     5.9086   7.027 2.72e-12
## neighborhoodHyde Park    -10.4942    11.4873  -0.914 0.361047
## neighborhoodJamaica Plain  11.6964     5.5812   2.096 0.036214
## neighborhoodLeather District 105.1790    27.5033   3.824 0.000134
## neighborhoodLongwood Medical Area 54.9176    26.9565   2.037 0.041730
## neighborhoodMattapan     -10.1929    13.2371  -0.770 0.441356
## neighborhoodMission Hill   12.3863     7.5101   1.649 0.099213
## neighborhoodNorth End     25.4530     7.1687   3.551 0.000392
## neighborhoodRoslindale    -14.6916     8.1721  -1.798 0.072334
## neighborhoodRoxbury       9.4433     6.3623   1.484 0.137866
## neighborhoodSouth Boston   32.5004     6.2940   5.164 2.61e-07
## neighborhoodSouth Boston Waterfront 91.4593    10.1349   9.024 < 2e-16
## neighborhoodSouth End     62.5618     5.8812  10.638 < 2e-16
## neighborhoodWest End      49.9164    11.2268   4.446 9.13e-06
```

## neighborhoodWest Roxbury	-2.9702	9.6720	-0.307	0.758797
## bedrooms	20.7472	2.1618	9.597	< 2e-16
## property_typeBed & Breakfast	16.8029	14.0051	1.200	0.230344
## property_typeBoat	26.8314	24.0694	1.115	0.265065
## property_typeBoutique hotel	-52.8776	31.5254	-1.677	0.093610
## property_typeCondominium	13.8768	3.7636	3.687	0.000232
## property_typeDorm	-21.1384	53.0228	-0.399	0.690174
## property_typeGuest suite	31.1185	21.7520	1.431	0.152670
## property_typeGuesthouse	91.4793	30.6245	2.987	0.002844
## property_typeHostel	-38.4091	54.2395	-0.708	0.478925
## property_typeHouse	5.5204	3.4082	1.620	0.105413
## property_typeIn-law	-20.5059	23.9582	-0.856	0.392134
## property_typeLoft	27.7601	12.0450	2.305	0.021266
## property_typeOther	55.4374	9.6794	5.727	1.14e-08
## property_typeServiced apartment	-34.6741	53.0548	-0.654	0.513461
## property_typeTimeshare	218.7057	53.1553	4.114	4.01e-05
## property_typeTownhouse	11.6491	8.7768	1.327	0.184542
## property_typeVilla	22.7554	20.1767	1.128	0.259510
## cancellation_policymoderate	7.5854	3.1921	2.376	0.017564
## cancellation_policystrict	-8.0366	2.8229	-2.847	0.004450
## cancellation_policysuper_strict_30	40.5316	14.0493	2.885	0.003949
## cancellation_policysuper_strict_60	100.1431	53.1929	1.883	0.059866
## instant_bookable	-14.9786	2.3365	-6.411	1.73e-10
## is_business_travel_ready	5.4584	3.4315	1.591	0.111809
## host_identity_verified	-3.8760	2.2862	-1.695	0.090119
## host_is_superhost	3.8657	2.9080	1.329	0.183869
##				
## (Intercept)	***			
## room_typePrivate room	***			
## room_typeShared room	***			
## accommodates	***			
## neighborhoodBack Bay	***			
## neighborhoodBay Village	***			
## neighborhoodBeacon Hill	***			
## neighborhoodBrighton				
## neighborhoodCharlestown	***			
## neighborhoodChinatown	***			
## neighborhoodDorchester				
## neighborhoodDowntown	***			
## neighborhoodEast Boston	.			
## neighborhoodFenway	***			
## neighborhoodHyde Park				
## neighborhoodJamaica Plain	*			
## neighborhoodLeather District	***			
## neighborhoodLongwood Medical Area	*			
## neighborhoodMattapan				
## neighborhoodMission Hill	.			
## neighborhoodNorth End	***			
## neighborhoodRoslindale	.			
## neighborhoodRoxbury				
## neighborhoodSouth Boston	***			
## neighborhoodSouth Boston Waterfront	***			
## neighborhoodSouth End	***			
## neighborhoodWest End	***			

```

## neighborhoodWest Roxbury
## bedrooms ***
## property_typeBed & Breakfast
## property_typeBoat
## property_typeBoutique hotel .
## property_typeCondominium ***
## property_typeDorm
## property_typeGuest suite
## property_typeGuesthouse **
## property_typeHostel
## property_typeHouse
## property_typeIn-law
## property_typeLoft *
## property_typeOther ***
## property_typeServiced apartment
## property_typeTimeshare ***
## property_typeTownhouse
## property_typeVilla
## cancellation_policymoderate *
## cancellation_policystrict **
## cancellation_policysuper_strict_30 **
## cancellation_policysuper_strict_60 .
## instant_bookablet ***
## is_business_travel_readyt
## host_identity_verifiedt .
## host_is_superhostt
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##              edf Ref.df      F  p-value
## s(guests_included)  3.682  4.467  8.809 1.49e-07 ***
## s(bathrooms)        1.024  1.048 11.944 0.000525 ***
## s(minimum_nights)   4.947  5.707  3.798 0.001862 **
## s(number_of_reviews) 1.803  2.272  3.231 0.036207 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.609   Deviance explained = 61.9%
## -REML = 13477   Scale est. = 2778.7    n = 2539
# R^2: 0.609

gam.var2.rmse <- rmse(boston.data.model_selection, gam.var2)
gam.var2.rmse # 58.11913

## [1] 58.11913

```

Shrinkage method within GAM

```

gam.var3 <- gam(price ~ host_is_superhost + host_identity_verified + neighborhood + property_type + room
summary(gam.var3)

##

```

```

## Family: gaussian
## Link function: identity
##
## Formula:
## price ~ host_is_superhost + host_identity_verified + neighborhood +
##     property_type + room_type + s(accommodates, bs = "cs") +
##     s(bathrooms, bs = "cs") + s(bedrooms, k = 5, bs = "cs") +
##     s(beds, bs = "cs") + bed_type + s(guests_included, bs = "cs") +
##     s(minimum_nights, bs = "cs") + s(number_of_reviews, bs = "cs") +
##     instant_bookable + is_business_travel_ready + cancellation_policy
##
## Parametric coefficients:
##
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    143.0699    12.8311   11.150 < 2e-16
## host_is_superhostt      3.6971     2.8833    1.282 0.199880
## host_identity_verifiedt -4.1873     2.2990   -1.821 0.068672
## neighborhoodBack Bay    82.7494     5.9169   13.985 < 2e-16
## neighborhoodBay Village  51.2601    14.8473    3.452 0.000565
## neighborhoodBeacon Hill  71.6485     6.2707   11.426 < 2e-16
## neighborhoodBrighton     7.1954     5.9515    1.209 0.226770
## neighborhoodCharlestown  55.4652     7.8520    7.064 2.10e-12
## neighborhoodChinatown   48.7407     8.1604    5.973 2.67e-09
## neighborhoodDorchester    5.1311     5.7621    0.890 0.373290
## neighborhoodDowntown    73.5038     6.3479   11.579 < 2e-16
## neighborhoodEast Boston   9.1249     6.1957    1.473 0.140936
## neighborhoodFenway       42.8928     5.9205    7.245 5.76e-13
## neighborhoodHyde Park   -10.4673    11.5042   -0.910 0.362983
## neighborhoodJamaica Plain 10.9255     5.5869    1.956 0.050629
## neighborhoodLeather District 103.5865    27.4796    3.770 0.000167
## neighborhoodLongwood Medical Area 58.5869    26.8893    2.179 0.029439
## neighborhoodMattapan   -11.0450    13.2234   -0.835 0.403650
## neighborhoodMission Hill  13.3995     7.5247    1.781 0.075077
## neighborhoodNorth End    25.4344     7.1599    3.552 0.000389
## neighborhoodRoslindale  -14.1585     8.2079   -1.725 0.084655
## neighborhoodRoxbury      8.6691     6.3766    1.360 0.174104
## neighborhoodSouth Boston  33.2277     6.2935    5.280 1.41e-07
## neighborhoodSouth Boston Waterfront 91.5142    10.1392    9.026 < 2e-16
## neighborhoodSouth End    62.2599     5.8952   10.561 < 2e-16
## neighborhoodWest End     49.5433    11.1412    4.447 9.10e-06
## neighborhoodWest Roxbury  -4.5993     9.6646   -0.476 0.634191
## property_typeBed & Breakfast 15.8409    13.9800    1.133 0.257276
## property_typeBoat        28.7391    24.0674    1.194 0.232550
## property_typeBoutique hotel -51.5961    31.4758   -1.639 0.101292
## property_typeCondominium  13.0156     3.7669    3.455 0.000559
## property_typeDorm       -25.8631    52.9824   -0.488 0.625490
## property_typeGuest suite  31.0022    21.7553    1.425 0.154272
## property_typeGuesthouse   92.6975    30.5928    3.030 0.002471
## property_typeHostel     -33.9783    54.9511   -0.618 0.536410
## property_typeHouse        5.1960     3.4120    1.523 0.127917
## property_typeIn-law     -21.7541    23.9380   -0.909 0.363560
## property_typeLoft         27.7543    12.0470    2.304 0.021315
## property_typeOther       55.4651     9.6472    5.749 1.01e-08
## property_typeServiced apartment -29.6876    53.0053   -0.560 0.575471
## property_typeTimeshare   218.5677    53.0527    4.120 3.92e-05

```


## property_typeTownhouse	12.2324	8.7882	1.392	0.164076
## property_typeVilla	24.1438	20.1632	1.197	0.231256
## room_typePrivate room	-55.1765	3.4622	-15.937	< 2e-16
## room_typeShared room	-64.6100	10.4119	-6.205	6.38e-10
## bed_typeCouch	13.1526	29.4240	0.447	0.654913
## bed_typeFuton	6.6351	16.8786	0.393	0.694273
## bed_typePull-out Sofa	3.7651	18.7918	0.200	0.841217
## bed_typeReal Bed	0.2398	11.6532	0.021	0.983587
## instant_bookable	-16.1252	2.3274	-6.928	5.41e-12
## is_business_travel_ready	6.2816	3.4218	1.836	0.066510
## cancellation_policymoderate	6.7307	3.1877	2.111	0.034834
## cancellation_policystrict	-8.2661	2.7929	-2.960	0.003109
## cancellation_policysuper_strict_30	41.7982	14.0372	2.978	0.002933
## cancellation_policysuper_strict_60	97.2346	53.2487	1.826	0.067964
##				
## (Intercept)	***			
## host_is_superhost				
## host_identity_verified	.			
## neighborhoodBack Bay	***			
## neighborhoodBay Village	***			
## neighborhoodBeacon Hill	***			
## neighborhoodBrighton				
## neighborhoodCharlestown	***			
## neighborhoodChinatown	***			
## neighborhoodDorchester				
## neighborhoodDowntown	***			
## neighborhoodEast Boston				
## neighborhoodFenway	***			
## neighborhoodHyde Park				
## neighborhoodJamaica Plain	.			
## neighborhoodLeather District	***			
## neighborhoodLongwood Medical Area	*			
## neighborhoodMattapan				
## neighborhoodMission Hill	.			
## neighborhoodNorth End	***			
## neighborhoodRoslindale	.			
## neighborhoodRoxbury				
## neighborhoodSouth Boston	***			
## neighborhoodSouth Boston Waterfront	***			
## neighborhoodSouth End	***			
## neighborhoodWest End	***			
## neighborhoodWest Roxbury				
## property_typeBed & Breakfast				
## property_typeBoat				
## property_typeBoutique hotel				
## property_typeCondominium	***			
## property_typeDorm				
## property_typeGuest suite				
## property_typeGuesthouse	**			
## property_typeHostel				
## property_typeHouse				
## property_typeIn-law				
## property_typeLoft	*			
## property_typeOther	***			

```

## property_typeServiced apartment
## property_typeTimeshare ***
## property_typeTownhouse
## property_typeVilla
## room_typePrivate room ***
## room_typeShared room ***
## bed_typeCouch
## bed_typeFuton
## bed_typePull-out Sofa
## bed_typeReal Bed
## instant_bookable ***
## is_business_travel_readyt .
## cancellation_policymoderate *
## cancellation_policystrict **
## cancellation_policysuper_strict_30 **
## cancellation_policysuper_strict_60 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##               edf Ref.df      F  p-value
## s(accommodates)  3.66005     9  4.835 7.56e-11 ***
## s(bathrooms)     1.06148     8  1.408 0.000446 ***
## s(bedrooms)      2.90382     4 26.724 < 2e-16 ***
## s(beds)           0.01323     9  0.001 0.375208
## s(guests_included) 2.76851     9  3.588 2.64e-08 ***
## s(minimum_nights)  1.47237     9  0.954 0.002546 **
## s(number_of_reviews) 0.99390     9  0.708 0.007460 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.61   Deviance explained =  62%
## -REML = 13487   Scale est. = 2773.3      n = 2539
# R^2: 0.610

gam.var3.rmse <- rmse(boston.data.model_selection, gam.var3)
gam.var3.rmse # 55.14769

```

```
## [1] 55.14769
```

Since, the shrinkage method of variable selection has the best RMSE, it will be adopted as the variable selection method henceforth.

GAM on all the datasets

Original dataset

```

gam.neighborhood <- gam(price ~ host_is_superhost + host_identity_verified + neighborhood + property_type)
summary(gam.neighborhood)

##
## Family: gaussian
## Link function: identity

```

```
##
## Formula:
## price ~ host_is_superhost + host_identity_verified + neighborhood +
##     property_type + room_type + s(accommodates, bs = "cs") +
##     s(bathrooms, bs = "cs") + s(bedrooms, k = 5, bs = "cs") +
##     s(beds, bs = "cs") + bed_type + s(guests_included, bs = "cs") +
##     s(minimum_nights, bs = "cs") + s(number_of_reviews, bs = "cs") +
##     instant_bookable + is_business_travel_ready + cancellation_policy
##
## Parametric coefficients:
```

	Estimate	Std. Error	t value	Pr(> t)
## (Intercept)	143.0699	12.8311	11.150	< 2e-16
## host_is_superhostt	3.6971	2.8833	1.282	0.199880
## host_identity_verifiedt	-4.1873	2.2990	-1.821	0.068672
## neighborhoodBack Bay	82.7494	5.9169	13.985	< 2e-16
## neighborhoodBay Village	51.2601	14.8473	3.452	0.000565
## neighborhoodBeacon Hill	71.6485	6.2707	11.426	< 2e-16
## neighborhoodBrighton	7.1954	5.9515	1.209	0.226770
## neighborhoodCharlestown	55.4652	7.8520	7.064	2.10e-12
## neighborhoodChinatown	48.7407	8.1604	5.973	2.67e-09
## neighborhoodDorchester	5.1311	5.7621	0.890	0.373290
## neighborhoodDowntown	73.5038	6.3479	11.579	< 2e-16
## neighborhoodEast Boston	9.1249	6.1957	1.473	0.140936
## neighborhoodFenway	42.8928	5.9205	7.245	5.76e-13
## neighborhoodHyde Park	-10.4673	11.5042	-0.910	0.362983
## neighborhoodJamaica Plain	10.9255	5.5869	1.956	0.050629
## neighborhoodLeather District	103.5865	27.4796	3.770	0.000167
## neighborhoodLongwood Medical Area	58.5869	26.8893	2.179	0.029439
## neighborhoodMattapan	-11.0450	13.2234	-0.835	0.403650
## neighborhoodMission Hill	13.3995	7.5247	1.781	0.075077
## neighborhoodNorth End	25.4344	7.1599	3.552	0.000389
## neighborhoodRoslindale	-14.1585	8.2079	-1.725	0.084655
## neighborhoodRoxbury	8.6691	6.3766	1.360	0.174104
## neighborhoodSouth Boston	33.2277	6.2935	5.280	1.41e-07
## neighborhoodSouth Boston Waterfront	91.5142	10.1392	9.026	< 2e-16
## neighborhoodSouth End	62.2599	5.8952	10.561	< 2e-16
## neighborhoodWest End	49.5433	11.1412	4.447	9.10e-06
## neighborhoodWest Roxbury	-4.5993	9.6646	-0.476	0.634191
## property_typeBed & Breakfast	15.8409	13.9800	1.133	0.257276
## property_typeBoat	28.7391	24.0674	1.194	0.232550
## property_typeBoutique hotel	-51.5961	31.4758	-1.639	0.101292
## property_typeCondominium	13.0156	3.7669	3.455	0.000559
## property_typeDorm	-25.8631	52.9824	-0.488	0.625490
## property_typeGuest suite	31.0022	21.7553	1.425	0.154272
## property_typeGuesthouse	92.6975	30.5928	3.030	0.002471
## property_typeHostel	-33.9783	54.9511	-0.618	0.536410
## property_typeHouse	5.1960	3.4120	1.523	0.127917
## property_typeIn-law	-21.7541	23.9380	-0.909	0.363560
## property_typeLoft	27.7543	12.0470	2.304	0.021315
## property_typeOther	55.4651	9.6472	5.749	1.01e-08
## property_typeServiced apartment	-29.6876	53.0053	-0.560	0.575471
## property_typeTimeshare	218.5677	53.0527	4.120	3.92e-05
## property_typeTownhouse	12.2324	8.7882	1.392	0.164076
## property_typeVilla	24.1438	20.1632	1.197	0.231256

## room_typePrivate room	-55.1765	3.4622	-15.937	< 2e-16
## room_typeShared room	-64.6100	10.4119	-6.205	6.38e-10
## bed_typeCouch	13.1526	29.4240	0.447	0.654913
## bed_typeFuton	6.6351	16.8786	0.393	0.694273
## bed_typePull-out Sofa	3.7651	18.7918	0.200	0.841217
## bed_typeReal Bed	0.2398	11.6532	0.021	0.983587
## instant_bookablet	-16.1252	2.3274	-6.928	5.41e-12
## is_business_travel_readyt	6.2816	3.4218	1.836	0.066510
## cancellation_policymoderate	6.7307	3.1877	2.111	0.034834
## cancellation_policystrict	-8.2661	2.7929	-2.960	0.003109
## cancellation_policysuper_strict_30	41.7982	14.0372	2.978	0.002933
## cancellation_policysuper_strict_60	97.2346	53.2487	1.826	0.067964
##				
## (Intercept)	***			
## host_is_superhostt				
## host_identity_verifiedt	.			
## neighborhoodBack Bay	***			
## neighborhoodBay Village	***			
## neighborhoodBeacon Hill	***			
## neighborhoodBrighton				
## neighborhoodCharlestown	***			
## neighborhoodChinatown	***			
## neighborhoodDorchester				
## neighborhoodDowntown	***			
## neighborhoodEast Boston				
## neighborhoodFenway	***			
## neighborhoodHyde Park				
## neighborhoodJamaica Plain	.			
## neighborhoodLeather District	***			
## neighborhoodLongwood Medical Area	*			
## neighborhoodMattapan				
## neighborhoodMission Hill	.			
## neighborhoodNorth End	***			
## neighborhoodRoslindale	.			
## neighborhoodRoxbury				
## neighborhoodSouth Boston	***			
## neighborhoodSouth Boston Waterfront	***			
## neighborhoodSouth End	***			
## neighborhoodWest End	***			
## neighborhoodWest Roxbury				
## property_typeBed & Breakfast				
## property_typeBoat				
## property_typeBoutique hotel				
## property_typeCondominium	***			
## property_typeDorm				
## property_typeGuest suite				
## property_typeGuesthouse	**			
## property_typeHostel				
## property_typeHouse				
## property_typeIn-law				
## property_typeLoft	*			
## property_typeOther	***			
## property_typeServiced apartment				
## property_typeTimeshare	***			

```
## property_typeTownhouse
## property_typeVilla
## room_typePrivate room ***
## room_typeShared room ***
## bed_typeCouch
## bed_typeFuton
## bed_typePull-out Sofa
## bed_typeReal Bed
## instant_bookable ***
## is_business_travel_readyt .
## cancellation_policymoderate *
## cancellation_policystrict **
## cancellation_policysuper_strict_30 **
## cancellation_policysuper_strict_60 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##              edf Ref.df      F  p-value
## s(accommodates)  3.66005    9  4.835 7.56e-11 ***
## s(bathrooms)     1.06148    8  1.408 0.000446 ***
## s(bedrooms)      2.90382    4 26.724 < 2e-16 ***
## s(beds)           0.01323    9  0.001 0.375208
## s(guests_included) 2.76851    9  3.588 2.64e-08 ***
## s(minimum_nights) 1.47237    9  0.954 0.002546 **
## s(number_of_reviews) 0.99390    9  0.708 0.007460 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.61   Deviance explained =  62%
## -REML = 13487   Scale est. = 2773.3    n = 2539
```

```
# R^2: 0.610
```

```
gam.neighborhood.rmse <- rmse(boston.data.model_selection, gam.neighborhood)
gam.neighborhood.rmse # 55.14769
```

```
## [1] 55.14769
```

Distance to both airport and downtown

```
gam.dboth <- gam(price ~ host_is_superhost + host_identity_verified + property_type + room_type + s(accommodates, bs = "cs") + s(bathrooms, bs = "cs") + s(bedrooms, k = 5, bs = "cs") + s(beds, bs = "cs") + bed_type + s(guests_included, bs = "cs") + s(minimum_nights, bs = "cs") + s(number_of_reviews, bs = "cs") + instant_bookable + is_business_travel_ready +
summary(gam.dboth)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## price ~ host_is_superhost + host_identity_verified + property_type +
##       room_type + s(accommodates, bs = "cs") + s(bathrooms, bs = "cs") +
##       s(bedrooms, k = 5, bs = "cs") + s(beds, bs = "cs") + bed_type +
##       s(guests_included, bs = "cs") + s(minimum_nights, bs = "cs") +
##       s(number_of_reviews, bs = "cs") + instant_bookable + is_business_travel_ready +
```

```

##      cancellation_policy + s(ddowntown, bs = "cs") + s(dairport,
##      bs = "cs")
##
## Parametric coefficients:
##
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      179.820     12.040   14.935 < 2e-16
## host_is_superhostt      3.064      2.902    1.056 0.291151
## host_identity_verifiedt  -4.503      2.313   -1.947 0.051666
## property_typeBed & Breakfast    17.782     14.082    1.263 0.206799
## property_typeBoat      30.327     24.329    1.247 0.212689
## property_typeBoutique hotel  -50.938     31.862   -1.599 0.110016
## property_typeCondominium    12.223      3.788    3.227 0.001269
## property_typeDorm     -22.933     53.598   -0.428 0.668786
## property_typeGuest suite    28.552     21.991    1.298 0.194281
## property_typeGuesthouse    87.730     30.913    2.838 0.004577
## property_typeHostel     -37.647     55.593   -0.677 0.498349
## property_typeHouse      3.475      3.418    1.017 0.309417
## property_typeIn-law    -26.717     24.178   -1.105 0.269258
## property_typeLoft      27.756     11.891    2.334 0.019668
## property_typeOther      54.536      9.639    5.658 1.71e-08
## property_typeServiced apartment -33.653     53.653   -0.627 0.530567
## property_typeTimeshare   221.508     53.640    4.130 3.75e-05
## property_typeTownhouse      9.214      8.845    1.042 0.297664
## property_typeVilla      23.443     20.371    1.151 0.249908
## room_typePrivate room   -57.901      3.455 -16.760 < 2e-16
## room_typeShared room   -68.185     10.502   -6.493 1.01e-10
## bed_typeCouch      4.670     29.654    0.157 0.874873
## bed_typeFuton      3.360     16.785    0.200 0.841350
## bed_typePull-out Sofa   -1.618     18.817   -0.086 0.931485
## bed_typeReal Bed     -1.763     11.612   -0.152 0.879317
## instant_bookablet    -15.601      2.331   -6.691 2.72e-11
## is_business_travel_readyt    5.600      3.450    1.623 0.104689
## cancellation_policymoderate    6.858      3.216    2.132 0.033092
## cancellation_policystRICT    -8.544      2.814   -3.036 0.002419
## cancellation_policysuper_strict_30 48.571     14.146    3.434 0.000605
## cancellation_policysuper_strict_60 93.666     53.872    1.739 0.082218
##
## (Intercept)          ***
## host_is_superhostt
## host_identity_verifiedt      .
## property_typeBed & Breakfast
## property_typeBoat
## property_typeBoutique hotel
## property_typeCondominium      **
## property_typeDorm
## property_typeGuest suite
## property_typeGuesthouse      **
## property_typeHostel
## property_typeHouse
## property_typeIn-law
## property_typeLoft            *
## property_typeOther          ***
## property_typeServiced apartment
## property_typeTimeshare      ***

```

```

## property_typeTownhouse
## property_typeVilla
## room_typePrivate room      ***
## room_typeShared room      ***
## bed_typeCouch
## bed_typeFuton
## bed_typePull-out Sofa
## bed_typeReal Bed
## instant_bookable      ***
## is_business_travel_readyt
## cancellation_policymoderate *
## cancellation_policystrict **
## cancellation_policysuper_strict_30 ***
## cancellation_policysuper_strict_60 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##              edf Ref.df      F  p-value
## s(accommodates)    3.79403    9  4.835 1.00e-10 ***
## s(bathrooms)       1.06742    8  1.446 0.000375 ***
## s(bedrooms)        2.81143    4 24.763 < 2e-16 ***
## s(beds)            0.01076    9  0.001 0.473394
## s(guests_included) 1.77026    9  3.176 4.35e-08 ***
## s(minimum_nights)  1.38292    9  0.808 0.005042 **
## s(number_of_reviews) 1.07710    9  0.940 0.002481 **
## s(ddowntown)       5.18910    9  3.603 2.70e-08 ***
## s(dairport)        5.31413    9 10.888 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.6   Deviance explained = 60.9%
## -REML = 13624   Scale est. = 2843.1    n = 2539
# R^2: 0.600

gam.dboth.rmspe <- rmse(boston.dboth.model_selection, gam.dboth)
gam.dboth.rmspe # 55.77279

## [1] 55.77279

```

Distance to airport

```

gam.dairport <- gam(price ~ host_is_superhost + host_identity_verified + property_type + room_type + s(
summary(gam.dairport)

##
## Family: gaussian
## Link function: identity
##
## Formula:
## price ~ host_is_superhost + host_identity_verified + property_type +
##       room_type + s(accommodates, bs = "cs") + s(bathrooms, bs = "cs") +
##       s(bedrooms, k = 5, bs = "cs") + bed_type + s(guests_included,

```

```

##      bs = "cs") + s(minimum_nights, bs = "cs") + s(number_of_reviews,
##      bs = "cs") + instant_bookable + is_business_travel_ready +
##      cancellation_policy + s(dairport, bs = "cs")
##
## Parametric coefficients:
##
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      180.312    12.102   14.900 < 2e-16
## host_is_superhostt      2.378     2.917    0.815 0.414965
## host_identity_verifiedt -5.056     2.320   -2.179 0.029392
## property_typeBed & Breakfast  18.890    14.156    1.334 0.182195
## property_typeBoat      17.969    24.184    0.743 0.457541
## property_typeBoutique hotel -52.989    32.047   -1.653 0.098356
## property_typeCondominium    9.431     3.772    2.501 0.012464
## property_typeDorm     -20.989    53.921   -0.389 0.697118
## property_typeGuest suite    26.397    22.112    1.194 0.232680
## property_typeGuesthouse    87.230    31.093    2.805 0.005063
## property_typeHostel     -51.621    55.806   -0.925 0.355053
## property_typeHouse        2.028     3.420    0.593 0.553278
## property_typeIn-law     -30.282    24.310   -1.246 0.213004
## property_typeLoft        29.828    11.915    2.503 0.012366
## property_typeOther       54.727     9.683    5.652 1.77e-08
## property_typeServiced apartment -35.817    53.976   -0.664 0.507023
## property_typeTimeshare   225.376    53.903    4.181 3.00e-05
## property_typeTownhouse     8.028     8.888    0.903 0.366462
## property_typeVilla       24.413    20.481    1.192 0.233375
## room_typePrivate room   -59.651     3.454 -17.271 < 2e-16
## room_typeShared room   -70.408    10.548   -6.675 3.04e-11
## bed_typeCouch          6.325    29.815    0.212 0.832007
## bed_typeFuton          2.384    16.874    0.141 0.887663
## bed_typePull-out Sofa   -2.905    18.912   -0.154 0.877921
## bed_typeReal Bed       -1.224    11.670   -0.105 0.916467
## instant_bookablet     -15.378     2.341   -6.570 6.10e-11
## is_business_travel_readyt  5.550     3.472    1.599 0.110048
## cancellation_policymoderate  7.650     3.231    2.367 0.017988
## cancellation_policystRICT -7.738     2.822   -2.742 0.006145
## cancellation_policysuper_strict_30 52.906    14.186    3.729 0.000196
## cancellation_policysuper_strict_60 84.551    54.120    1.562 0.118349
##
## (Intercept)      ***
## host_is_superhostt
## host_identity_verifiedt      *
## property_typeBed & Breakfast
## property_typeBoat
## property_typeBoutique hotel      .
## property_typeCondominium      *
## property_typeDorm
## property_typeGuest suite
## property_typeGuesthouse      **
## property_typeHostel
## property_typeHouse
## property_typeIn-law
## property_typeLoft      *
## property_typeOther      ***
## property_typeServiced apartment

```



```

## property_typeTimeshare          ***
## property_typeTownhouse
## property_typeVilla
## room_typePrivate room          ***
## room_typeShared room           ***
## bed_typeCouch
## bed_typeFuton
## bed_typePull-out Sofa
## bed_typeReal Bed
## instant_bookable                ***
## is_business_travel_readyt
## cancellation_policymoderate     *
## cancellation_policystrict       **
## cancellation_policysuper_strict_30 ***
## cancellation_policysuper_strict_60
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##              edf Ref.df      F  p-value
## s(accommodates)    3.711     9  4.848 7.89e-11 ***
## s(bathrooms)       1.065     8  1.418 0.000428 ***
## s(bedrooms)        2.754     4 21.536 < 2e-16 ***
## s(guests_included) 2.834     9  3.480 5.24e-08 ***
## s(minimum_nights)  1.311     9  0.680 0.009689 **
## s(number_of_reviews) 1.066     9  0.882 0.003313 **
## s(dairport)        7.051     9 48.773 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.596   Deviance explained = 60.4%
## -REML = 13632   Scale est. = 2876.8    n = 2539
# R^2: 0.596

gam.dairport.rmse <- rmse(boston.dairport.model_selection, gam.dairport)
gam.dairport.rmse # 56.23814

## [1] 56.23814

```

Distance to downtown

```

gam.ddowntown <- gam(price ~ host_is_superhost + host_identity_verified + property_type + room_type + s(
summary(gam.ddowntown)

##
## Family: gaussian
## Link function: identity
##
## Formula:
## price ~ host_is_superhost + host_identity_verified + property_type +
##       room_type + s(accommodates, bs = "cs") + s(bathrooms, bs = "cs") +
##       s(bedrooms, k = 5, bs = "cs") + bed_type + s(guests_included,
##       bs = "cs") + s(minimum_nights, bs = "cs") + s(number_of_reviews,

```

```

##      bs = "cs") + instant_bookable + is_business_travel_ready +
##      cancellation_policy + s(ddowntown, bs = "cs")
##
## Parametric coefficients:
##
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      182.833     12.301   14.864 < 2e-16
## host_is_superhostt      3.934      2.957    1.330 0.183501
## host_identity_verifiedt    -4.507      2.345   -1.922 0.054752
## property_typeBed & Breakfast    23.477     14.332    1.638 0.101521
## property_typeBoat      40.677     24.515    1.659 0.097179
## property_typeBoutique hotel   -67.487     32.257   -2.092 0.036527
## property_typeCondominium     12.078      3.826    3.157 0.001612
## property_typeDorm    -22.427     54.623   -0.411 0.681418
## property_typeGuest suite     28.527     22.408    1.273 0.203110
## property_typeGuesthouse     92.001     31.506    2.920 0.003530
## property_typeHostel    -49.630     56.658   -0.876 0.381131
## property_typeHouse      1.899      3.458    0.549 0.582951
## property_typeIn-law   -30.362     24.638   -1.232 0.217952
## property_typeLoft      24.668     12.087    2.041 0.041369
## property_typeOther     53.670      9.847    5.450 5.52e-08
## property_typeServiced apartment -59.294     54.535   -1.087 0.277019
## property_typeTimeshare    221.426     54.754    4.044 5.41e-05
## property_typeTownhouse     12.423      9.004    1.380 0.167773
## property_typeVilla     22.211     20.764    1.070 0.284850
## room_typePrivate room   -60.755      3.513  -17.294 < 2e-16
## room_typeShared room   -70.086     10.685   -6.559 6.55e-11
## bed_typeCouch      1.102     30.231    0.036 0.970920
## bed_typeFuton      2.909     17.138    0.170 0.865209
## bed_typePull-out Sofa    -3.651     19.196   -0.190 0.849182
## bed_typeReal Bed     -3.039     11.872   -0.256 0.797984
## instant_bookablet    -15.056      2.386   -6.311 3.27e-10
## is_business_travel_readyt      4.788      3.512    1.363 0.172920
## cancellation_policymoderate      5.421      3.279    1.653 0.098370
## cancellation_policystRICT      -9.105      2.886   -3.155 0.001627
## cancellation_policysuper_strict_30  48.182     14.392    3.348 0.000826
## cancellation_policysuper_strict_60  84.447     54.837    1.540 0.123697
##
## (Intercept)          ***
## host_is_superhostt
## host_identity_verifiedt      .
## property_typeBed & Breakfast
## property_typeBoat      .
## property_typeBoutique hotel      *
## property_typeCondominium      **
## property_typeDorm
## property_typeGuest suite
## property_typeGuesthouse      **
## property_typeHostel
## property_typeHouse
## property_typeIn-law
## property_typeLoft      *
## property_typeOther      ***
## property_typeServiced apartment
## property_typeTimeshare      ***

```

```

## property_typeTownhouse
## property_typeVilla
## room_typePrivate room      ***
## room_typeShared room      ***
## bed_typeCouch
## bed_typeFuton
## bed_typePull-out Sofa
## bed_typeReal Bed
## instant_bookable          ***
## is_business_travel_readyt
## cancellation_policymoderate .
## cancellation_policystrict  **
## cancellation_policysuper_strict_30 ***
## cancellation_policysuper_strict_60
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##              edf Ref.df      F p-value
## s(accommodates)  3.643    9  4.900 5.03e-11 ***
## s(bathrooms)     1.086    8  1.581 0.000209 ***
## s(bedrooms)      2.623    4 18.720 < 2e-16 ***
## s(guests_included) 1.439    9  3.024 7.84e-08 ***
## s(minimum_nights)  4.821    9  1.800 0.003453 **
## s(number_of_reviews) 1.150    9  1.446 0.000201 ***
## s(ddowntown)      4.621    9 39.472 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.585   Deviance explained = 59.3%
## -REML = 13662   Scale est. = 2954.7    n = 2539
# R^2: 0.585

gam.ddowntown.rmse <- rmse(boston.ddowntown.model_selection, gam.ddowntown)
gam.ddowntown.rmse # 58.05393

## [1] 58.05393

```

Results of the best model on validation_1 datasets

```

gam.best <- gam.neighborhood
gam.best.v1_rmse <- rmse(boston.data.validation_1, gam.best)
gam.best.v1_rmse # 55.43666

## [1] 55.43666

# Most important predictors:
# Parametric: neighborhood, property_type, room_type, instant_bookable
# Non-parametric: accommodates, bedrooms, guests_included

```

Regression Trees

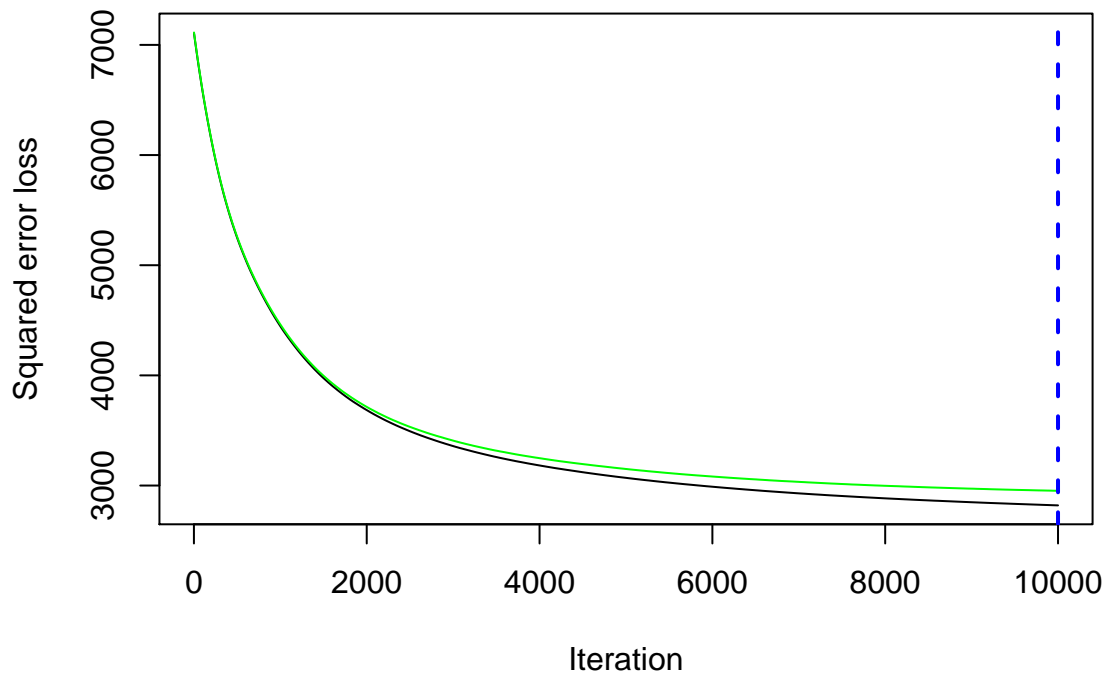
Train each of the transformations

```
boston.data.boost <- gbm(price ~ ., boston.data.training, distribution="gaussian", n.trees=10000, cv.folds=5)
boston.dboth.boost <- gbm(price ~ ., boston.dboth.training, distribution="gaussian", n.trees=10000, cv.folds=5)
boston.ddowntown.boost <- gbm(price ~ ., boston.ddowntown.training, distribution="gaussian", n.trees=10000, cv.folds=5)
boston.dairport.boost <- gbm(price ~ ., boston.dairport.training, distribution="gaussian", n.trees=10000, cv.folds=5)
```

Results of neighborhoods

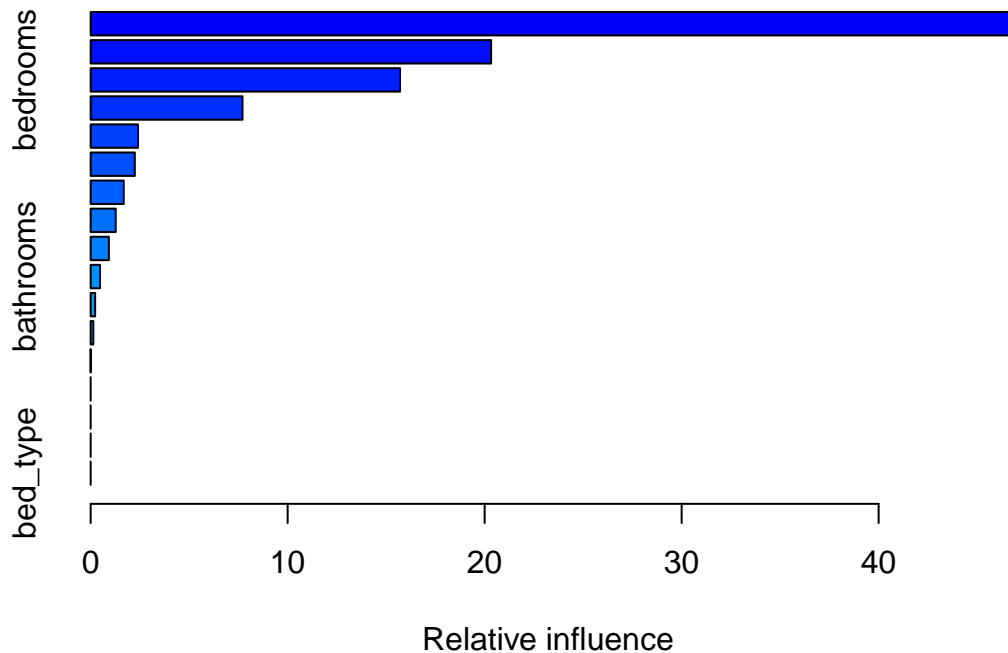
```
gbm.perf(boston.data.boost)
```

```
## Using cv method...
```



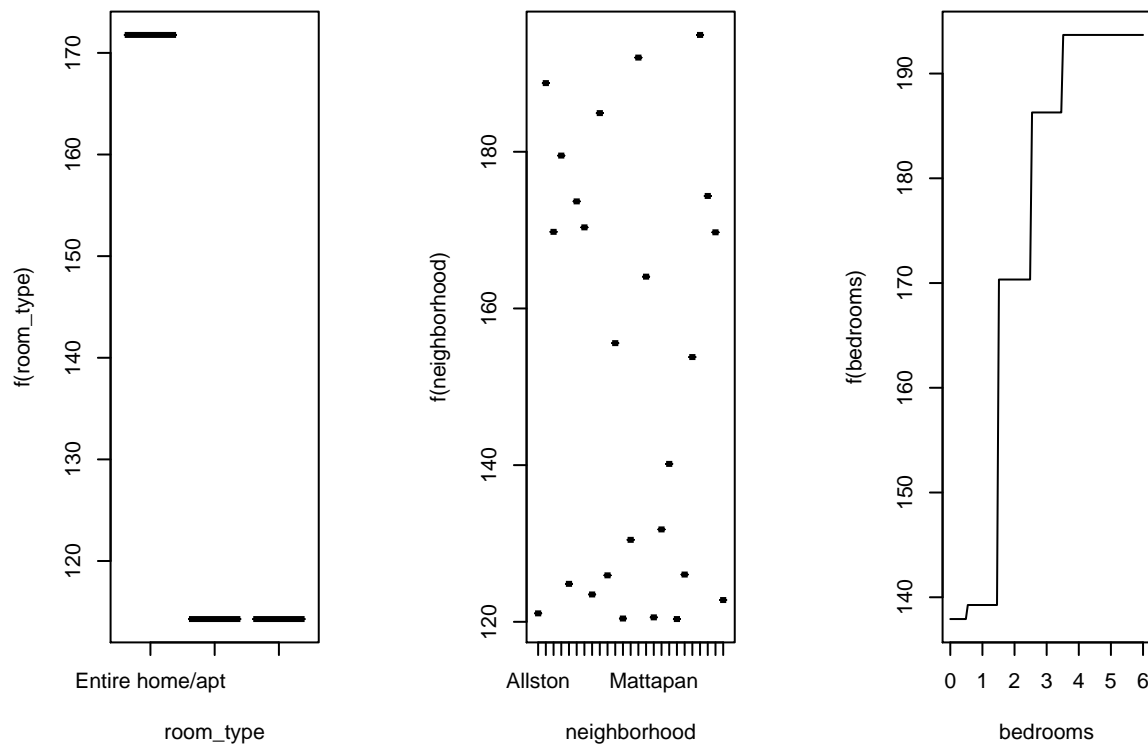
```
## [1] 10000
```

```
summary(boston.data.boost)
```



```
##          var      rel.inf
## room_type      room_type 46.902021406
## neighborhood  neighborhood 20.329109091
## bedrooms      bedrooms  15.704864188
## accommodates  accommodates  7.707761783
## property_type property_type  2.402183578
## guests_included guests_included 2.240084631
## beds          beds      1.677515267
## instant_bookable instant_bookable 1.271554395
## cancellation_policy cancellation_policy 0.924653047
## bathrooms      bathrooms  0.472544863
## minimum_nights minimum_nights 0.227962547
## number_of_reviews number_of_reviews 0.136885338
## is_business_travel_ready is_business_travel_ready 0.002859867
## X              X          0.000000000
## host_is_superhost host_is_superhost 0.000000000
## host_identity_verified host_identity_verified 0.000000000
## bed_type        bed_type  0.000000000
```

```
par(mfrow=c(1, 3))
plot(boston.data.boost, i=paste(summary(boston.data.boost, plotit=FALSE)$var[[1]], "", sep=""))
plot(boston.data.boost, i=paste(summary(boston.data.boost, plotit=FALSE)$var[[2]], "", sep=""))
plot(boston.data.boost, i=paste(summary(boston.data.boost, plotit=FALSE)$var[[3]], "", sep=""))
```



```
yhat <- predict(boston.data.boost, newdata=boston.data.validation_1)
```

```
## Using 10000 trees...
```

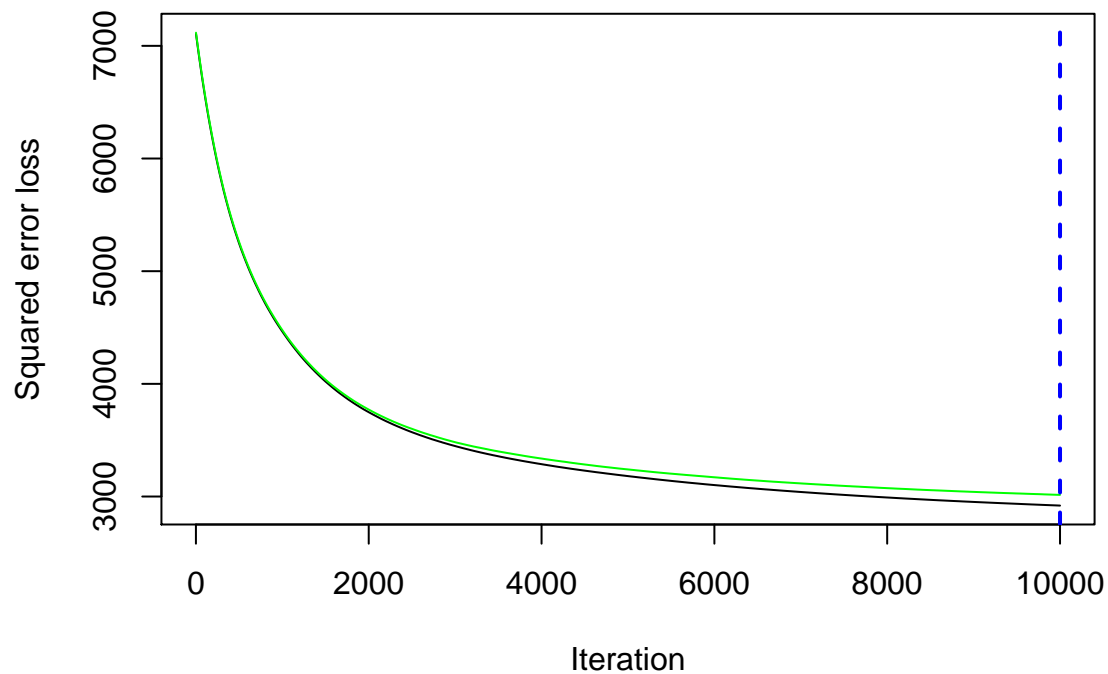
```
sqrt(mean((yhat - boston.data.validation_1.test)^2))
```

```
## [1] 55.74535
```

Results of dboth

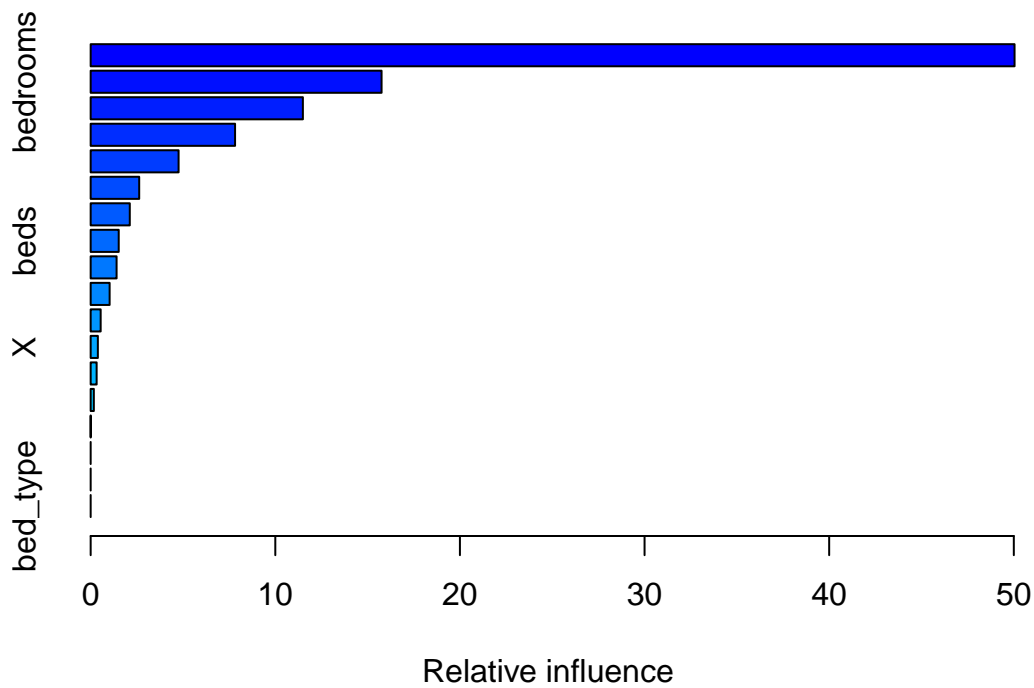
```
gbm.perf(boston.dboth.boost)
```

```
## Using cv method...
```



```
## [1] 10000
```

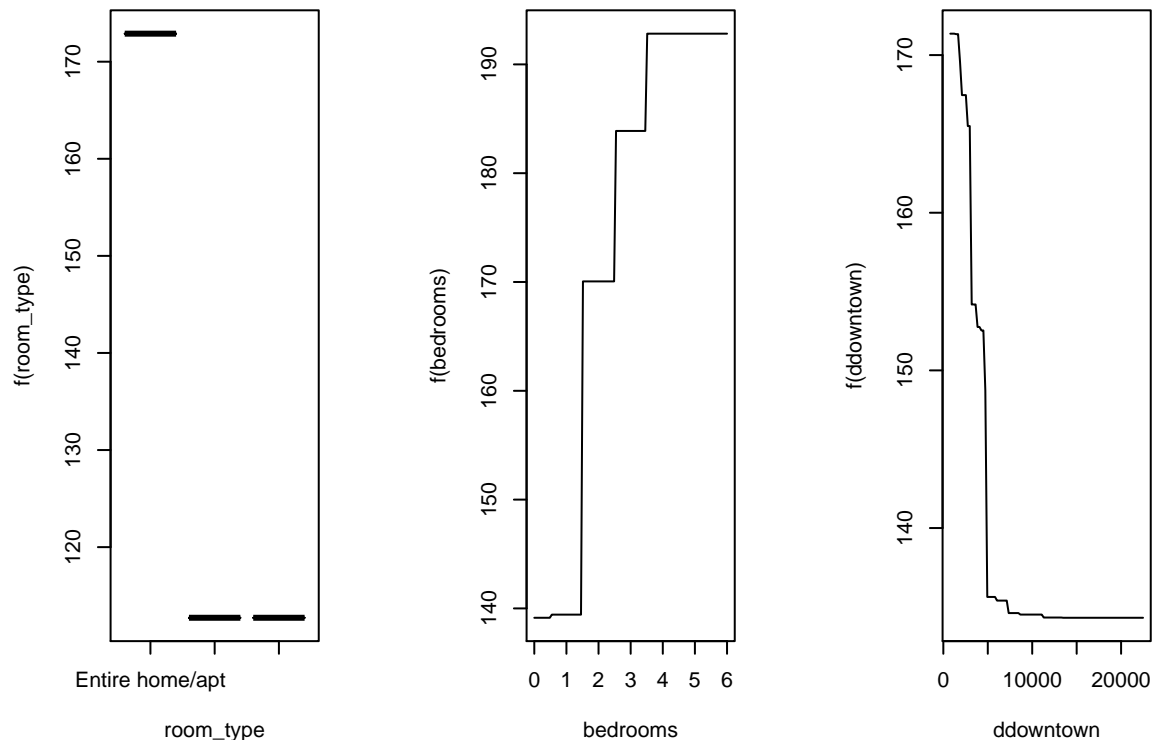
```
summary(boston.dboth.boost)
```



```
##           var      rel.inf
## room_type   room_type 50.04056656
## bedrooms    bedrooms 15.75653067
## ddowntown   ddowntown 11.49398791
## accommodates accommodates 7.82123675
## dairport    dairport  4.76219213
## property_type property_type 2.62872238
```

```
## guests_included      guests_included  2.11881030
## beds                 beds            1.51953974
## instant_bookable     instant_bookable 1.40348664
## cancellation_policy   cancellation_policy 1.02669766
## bathrooms            bathrooms       0.54091294
## X                     X              0.38991469
## number_of_reviews     number_of_reviews 0.32350215
## minimum_nights        minimum_nights  0.17139534
## is_business_travel_ready is_business_travel_ready 0.00250415
## host_is_superhost     host_is_superhost 0.00000000
## host_identity_verified host_identity_verified 0.00000000
## bed_type              bed_type        0.00000000
```

```
par(mfrow=c(1, 3))
plot(boston.dboth.boost, i=paste(summary(boston.dboth.boost, plotit=FALSE)$var[[1]], "", sep=""))
plot(boston.dboth.boost, i=paste(summary(boston.dboth.boost, plotit=FALSE)$var[[2]], "", sep=""))
plot(boston.dboth.boost, i=paste(summary(boston.dboth.boost, plotit=FALSE)$var[[3]], "", sep=""))
```



```
yhat <- predict(boston.dboth.boost, newdata=boston.dboth.validation_1)
```

```
## Using 10000 trees...
```

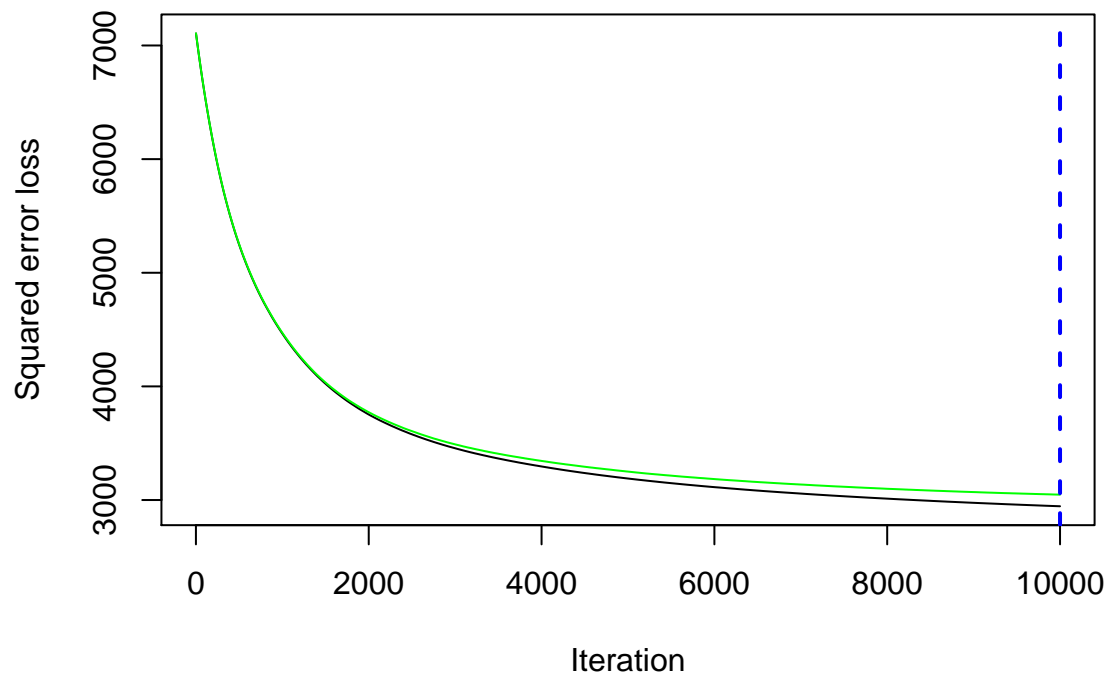
```
sqrt(mean((yhat - boston.dboth.validation_1.test)^2))
```

```
## [1] 56.552
```

Results of ddowntown

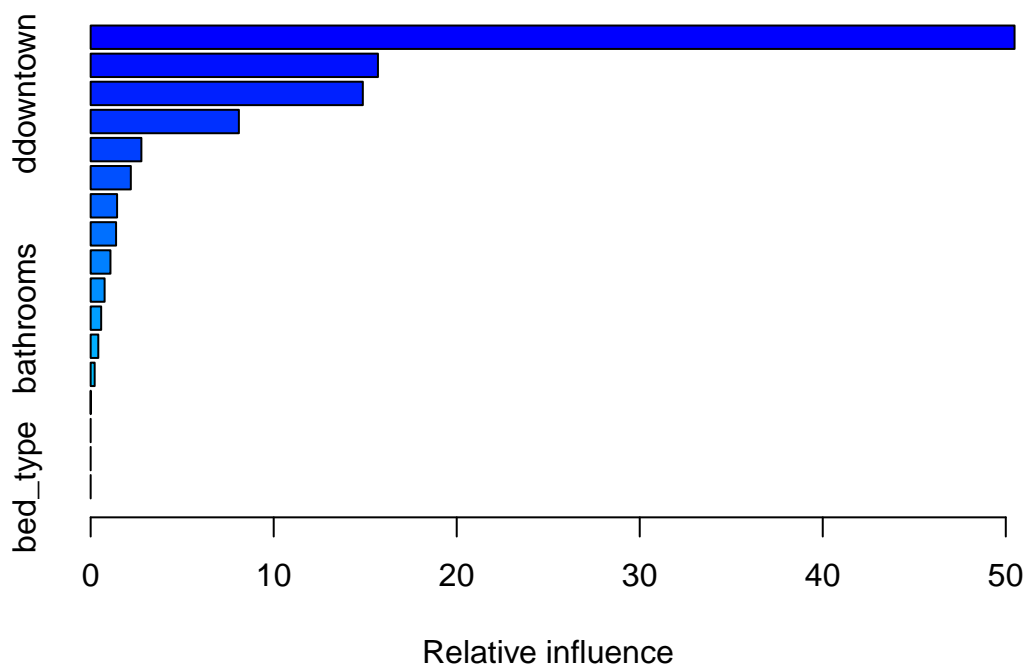
```
gbm.perf(boston.ddowntown.boost)
```

```
## Using cv method...
```

```
## [1] 10000
```

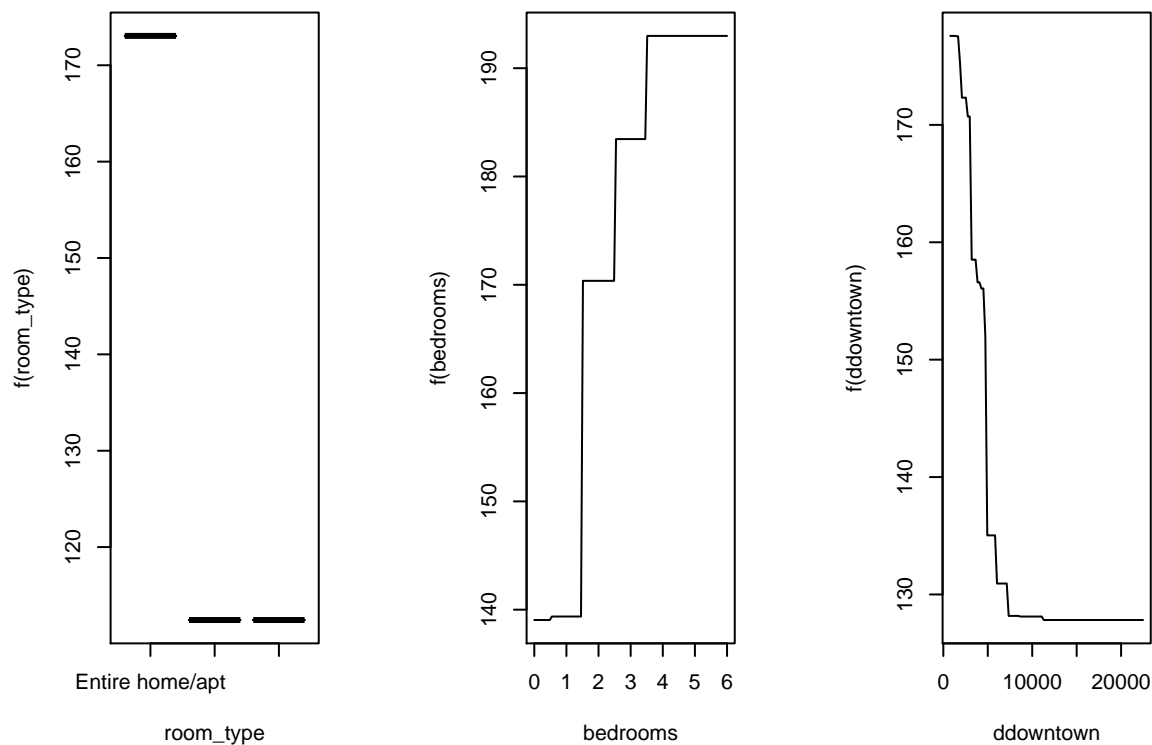
```
summary(boston.ddowntown.boost)
```



```
##          var      rel.inf
## room_type      room_type 50.479988132
## bedrooms        bedrooms 15.695991149
## ddowntown        ddowntown 14.870256667
## accommodates    accommodates 8.097831270
## property_type    property_type 2.771372775
## guests_included  guests_included 2.194769180
## instant_bookable instant_bookable 1.444161661
```

```
## beds                      beds 1.387947725
## cancellation_policy        cancellation_policy 1.081001274
## X                          X 0.762394147
## bathrooms                  bathrooms 0.573571076
## number_of_reviews          number_of_reviews 0.416098167
## minimum_nights             minimum_nights 0.218198543
## is_business_travel_ready   is_business_travel_ready 0.006418234
## host_is_superhost          host_is_superhost 0.000000000
## host_identity_verified     host_identity_verified 0.000000000
## bed_type                   bed_type 0.000000000
```

```
par(mfrow=c(1, 3))
plot(boston.ddowntown.boost, i=paste(summary(boston.ddowntown.boost, plotit=FALSE)$var[[1]], "", sep=""))
plot(boston.ddowntown.boost, i=paste(summary(boston.ddowntown.boost, plotit=FALSE)$var[[2]], "", sep=""))
plot(boston.ddowntown.boost, i=paste(summary(boston.ddowntown.boost, plotit=FALSE)$var[[3]], "", sep=""))
```



```
yhat <- predict(boston.ddowntown.boost, newdata=boston.ddowntown.validation_1)
```

```
## Using 10000 trees...
```

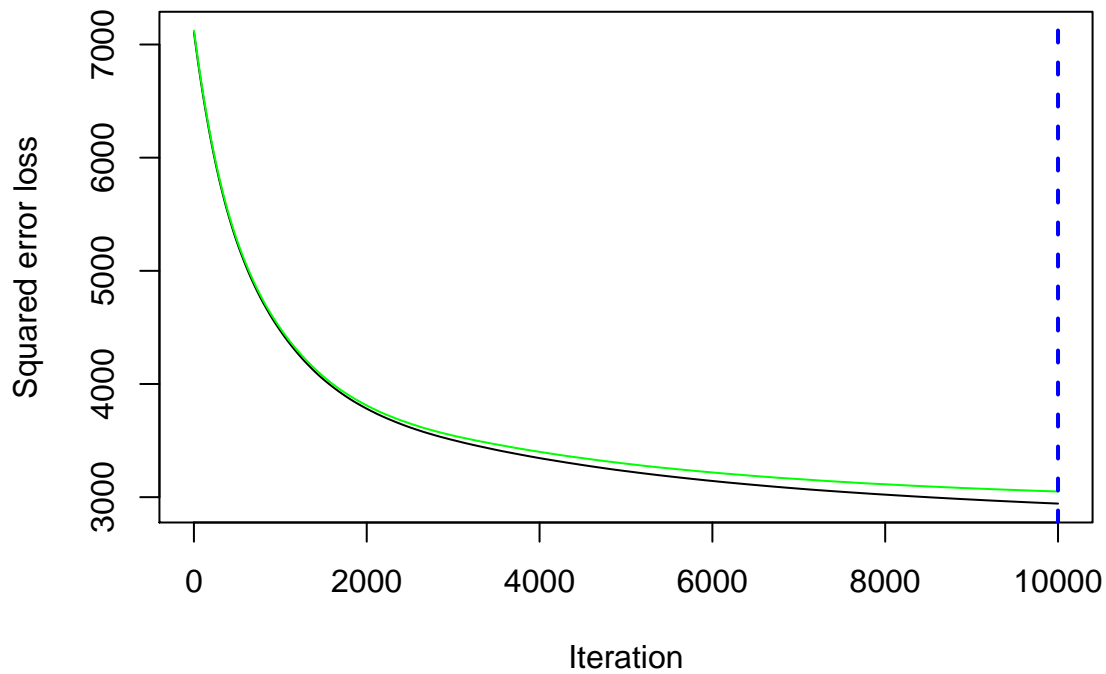
```
sqrt(mean((yhat - boston.ddowntown.validation_1.test)^2))
```

```
## [1] 56.95934
```

Results of dairport

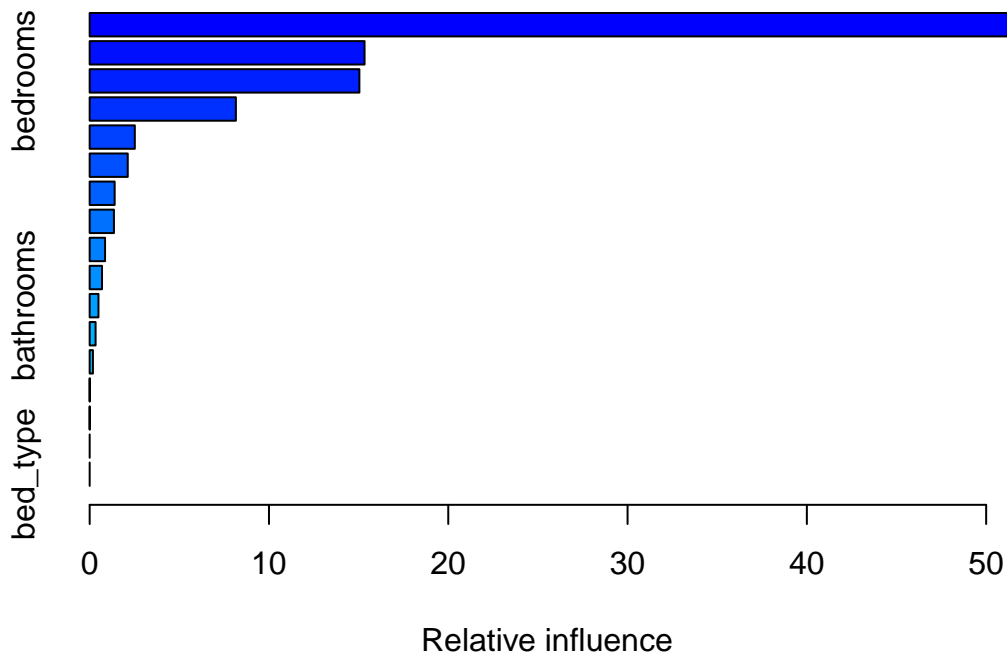
```
gbm.perf(boston.dairport.boost)
```

```
## Using cv method...
```



```
## [1] 10000
```

```
summary(boston.dairport.boost)
```

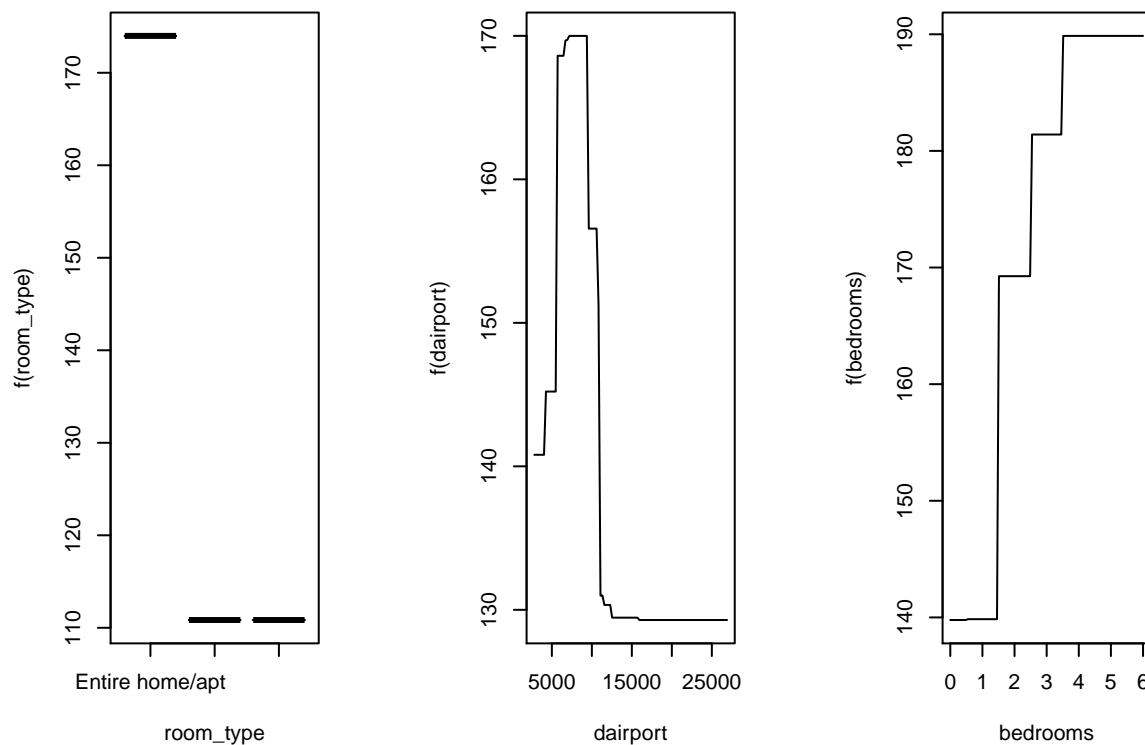


```
##           var      rel.inf
## room_type    room_type 51.529190299
## dairport     dairport 15.331317302
## bedrooms     bedrooms 15.042769665
## accommodates accommodates 8.155057630
## property_type property_type 2.518724816
## guests_included guests_included 2.123554144
## beds         beds      1.393094164
```

```
## instant_bookable          instant_bookable 1.353913000
## cancellation_policy       cancellation_policy 0.857509865
## X                          X 0.689249892
## bathrooms                 bathrooms 0.490492202
## number_of_reviews         number_of_reviews 0.330708392
## minimum_nights            minimum_nights 0.180950593
## is_business_travel_ready  is_business_travel_ready 0.002538401
## host_identity_verified     host_identity_verified 0.000929635
## host_is_superhost         host_is_superhost 0.000000000
## bed_type                  bed_type 0.000000000
```

```
par(mfrow=c(1, 3))
```

```
plot(boston.dairport.boost, i=paste(summary(boston.dairport.boost, plotit=FALSE)$var[[1]], "", sep=""))
plot(boston.dairport.boost, i=paste(summary(boston.dairport.boost, plotit=FALSE)$var[[2]], "", sep=""))
plot(boston.dairport.boost, i=paste(summary(boston.dairport.boost, plotit=FALSE)$var[[3]], "", sep=""))
```



```
yhat <- predict(boston.dairport.boost, newdata=boston.dairport.validation_1)
```

```
## Using 10000 trees...
```

```
sqrt(mean((yhat - boston.dairport.validation_1.test)^2))
```

```
## [1] 56.5845
```

Best Model

Results of the overall best model(GAM - Original dataset) on validation_2 dataset:

```
gam.best.v2_rmse <- rmse(boston.data.validation_2, gam.best)
gam.best.v2_rmse # 52.29939
```

```
## [1] 52.29939
```