HOUSING PROPERTY PRICE ESTIMATION IN LONDON

Nishant Kumar, Alok Kumar Singh, Ramesh Suragam, Pawan Rakesh Kumar Narayan Gowda

5/11/2020

```r
#Loading libraries
suppressMessages(suppressWarnings(library(tidyverse)))
suppressMessages(suppressWarnings(library(ggplot2)))
suppressMessages(suppressWarnings(library(gridExtra)))
suppressMessages(suppressWarnings(library(MASS)))
suppressMessages(suppressWarnings(library(randomForest)))
suppressMessages(suppressWarnings(library("PerformanceAnalytics")))
```

Introduction:

Loading Data:

```r
#Loading the data
LondonData <- suppressMessages(suppressWarnings(read_csv("data/DataScienceProj.csv")))
head(LondonData)
```

```
## # A tibble: 6 x 31
##      X1 Easting Northing Purprice BldIntWr BldPostW Bld60s Bld70s Bld80s
##   <dbl>   <dbl>    <dbl>    <dbl>    <dbl>    <dbl>  <dbl>  <dbl>  <dbl>
## 1    53  545500   173000    85000        0        0      1      0      0
## 2    73  525000   177800    71000        0        0      0      0      1
## 3    78  531100   183400    60000        0        0      0      0      0
## 4    95  538500   169400    64000        0        0      0      0      1
## 5   125  534000   168400   260000        0        0      0      0      1
## 6   153  528700   168800    48500        0        0      0      0      0
## # ... with 22 more variables: TypDetch <dbl>, TypSemiD <dbl>, TypFlat <dbl>,
## #   GarSingl <dbl>, GarDoubl <dbl>, Tenfree <dbl>, CenHeat <dbl>,
## #   BathTwo <dbl>, BedTwo <dbl>, BedThree <dbl>, BedFour <dbl>, BedFive <dbl>,
## #   NewPropD <dbl>, FlorArea <dbl>, NoCarHh <dbl>, CarspP <dbl>, ProfPct <dbl>,
## #   UnskPct <dbl>, RetiPct <dbl>, Saleunem <dbl>, Unemploy <dbl>,
## #   PopnDnsy <dbl>
```

```r
#Checking for correlation
M <- cor(LondonData[,c(4,23:31)])
head(round(M,2))
```

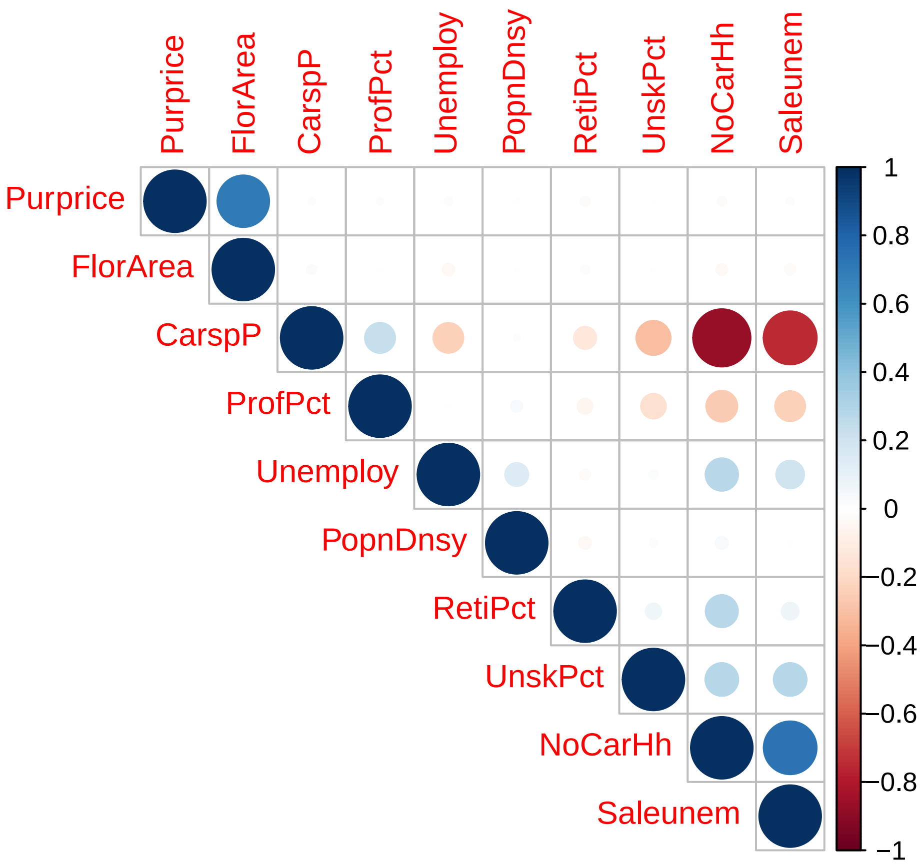
```
##          Purprice FlorArea NoCarHh CarspP ProfPct UnskPct RetiPct Saleunem
## Purprice     1.00     0.70   -0.02   0.01    0.01    0.00   -0.02    -0.01
## FlorArea     0.70     1.00   -0.03   0.02    0.00    0.01   -0.02    -0.03
## NoCarHh     -0.02    -0.03    1.00  -0.86   -0.25    0.28    0.27     0.73
## CarspP       0.01     0.02   -0.86   1.00    0.24   -0.31   -0.13    -0.74
## ProfPct      0.01     0.00   -0.25   0.24    1.00   -0.16   -0.06    -0.24
```

```
## UnskPct       0.00      0.01     0.28  -0.31    -0.16     1.00     0.06      0.28
##            Unemploy PopnDnsy
## Purprice     -0.02      0.01
## FlorArea     -0.04      0.00
## NoCarHh       0.28      0.04
## CarspP       -0.23     -0.01
## ProfPct       0.01      0.03
## UnskPct       0.02     -0.02
```

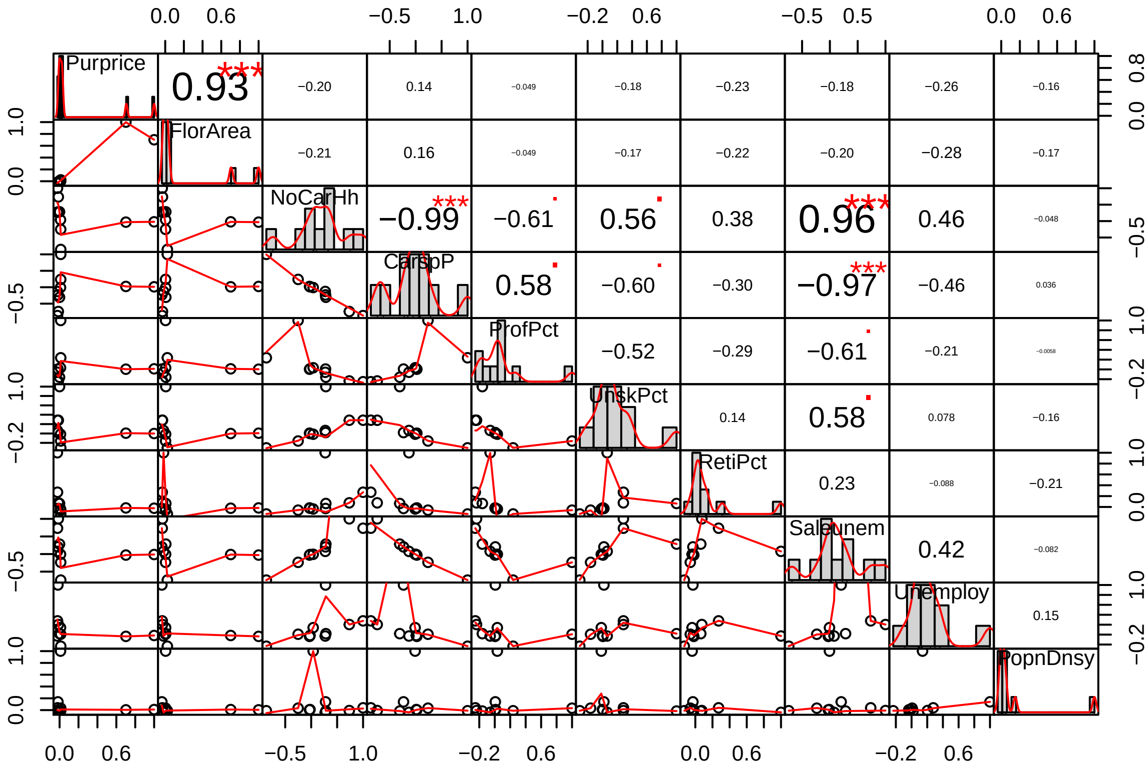
```r
library(corrplot)
```

```
## corrplot 0.84 loaded
```

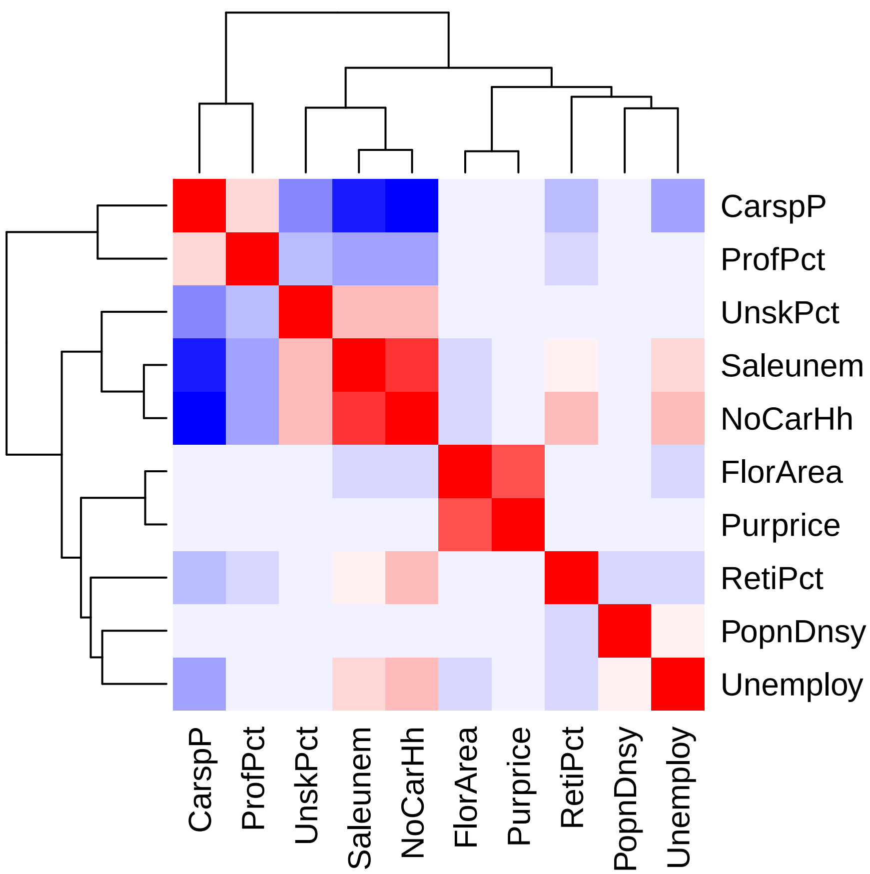
```r
corrplot(M, method="circle", type = "upper", order="hclust", sig.level = 0.01)
```




```r
chart.Correlation(M, histogram=TRUE, pch=19)
```




```r
col<- colorRampPalette(c("blue", "white", "red"))(20)
heatmap(x = M, col = col, symm = TRUE)
```



Data Cleanup:

Convert dummies to factors

```
Dummy2Factor <- function(mat,lev1="Level1") {
      mat <- as.matrix(mat)
      factor((mat %*% (1:ncol(mat))) + 1,
          labels = c(lev1, colnames(mat)))
}

Age      <- Dummy2Factor(LondonData[,5:9],"PreWW1")
Type     <- Dummy2Factor(LondonData[,10:12],"Others")
Garage   <- Dummy2Factor(LondonData[,13:14],"HardStnd")
Bedrooms <- Dummy2Factor(LondonData[,18:21],"BedOne")

MyData <- data.frame(LondonData[,c(2:4,15:17,22,23,26)],Age,Type,Garage,Bedrooms)
summary(MyData)
```

```
##     Easting          Northing          Purprice          Tenfree
##  Min.   :504400   Min.   :157200   Min.   :  8500   Min.   :0.0000
##  1st Qu.:517800   1st Qu.:172700   1st Qu.: 55000   1st Qu.:0.0000
##  Median :527600   Median :181200   Median : 70000   Median :1.0000
##  Mean   :527926   Mean   :180009   Mean   : 80018   Mean   :0.6835
##  3rd Qu.:536700   3rd Qu.:187400   3rd Qu.: 90000   3rd Qu.:1.0000
##  Max.   :558000   Max.   :200100   Max.   :850000   Max.   :1.0000
##     CenHeat          BathTwo           NewPropD          FlorArea
##  Min.   :0.0000   Min.   :0.00000   Min.   :0.00000   Min.   : 23.22
##  1st Qu.:1.0000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.: 71.77
##  Median :1.0000   Median :0.00000   Median :0.00000   Median : 91.02
##  Mean   :0.8789   Mean   :0.05392   Mean   :0.03638   Mean   : 96.49
##  3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:112.11
##  Max.   :1.0000   Max.   :1.00000   Max.   :1.00000   Max.   :278.00
##     ProfPct              Age             Type           Garage
##  Min.   :  0.000   PreWW1  :4261   Others  :3791   HardStnd:8306
##  1st Qu.:  0.000   BldIntWr:4365   TypDetch:1168   GarSingl:3923
##  Median :  5.556   BldPostW:1054   TypSemiD:3260   GarDoubl: 307
##  Mean   :  7.640   Bld60s  : 789   TypFlat :4317
##  3rd Qu.: 12.500   Bld70s  : 679
##  Max.   :100.000   Bld80s  :1388
##      Bedrooms
##  BedOne  :1713
##  BedTwo  :3785
##  BedThree:5723
##  BedFour :1100
##  BedFive : 215
##
```

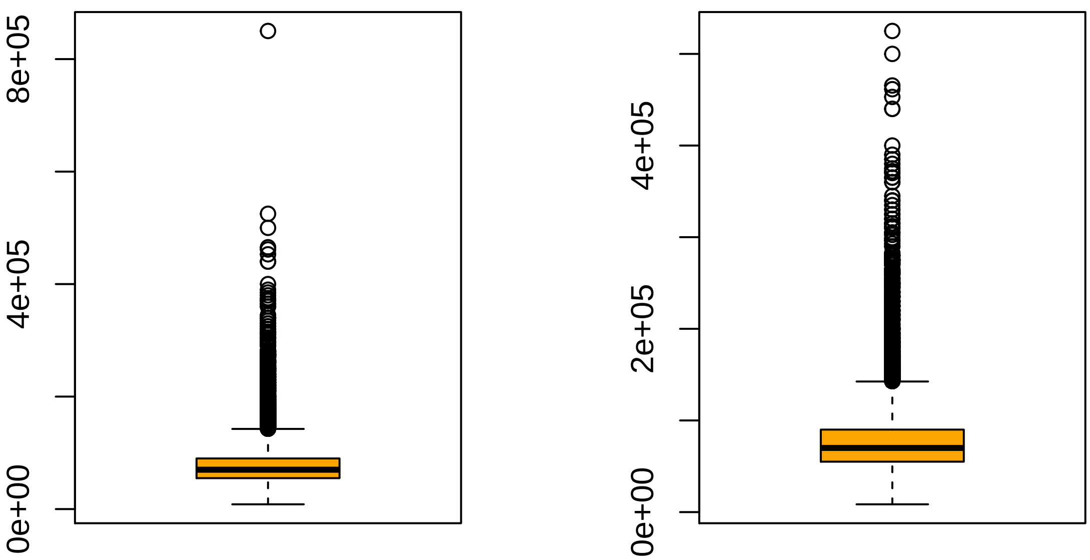
```
MyData$Tenfree <- factor(MyData$Tenfree)
MyData$CenHeat <- factor(MyData$CenHeat)
MyData$BathTwo <- factor(MyData$BathTwo)
MyData$NewPropD <- factor(MyData$NewPropD)

levels(MyData$Tenfree) <- c("no", "yes")
levels(MyData$CenHeat) <- c("no", "yes")
```

```r
levels(MyData$BathTwo) <- c("no", "yes")
levels(MyData$NewPropD) <- c("no", "yes")

head(MyData)
```

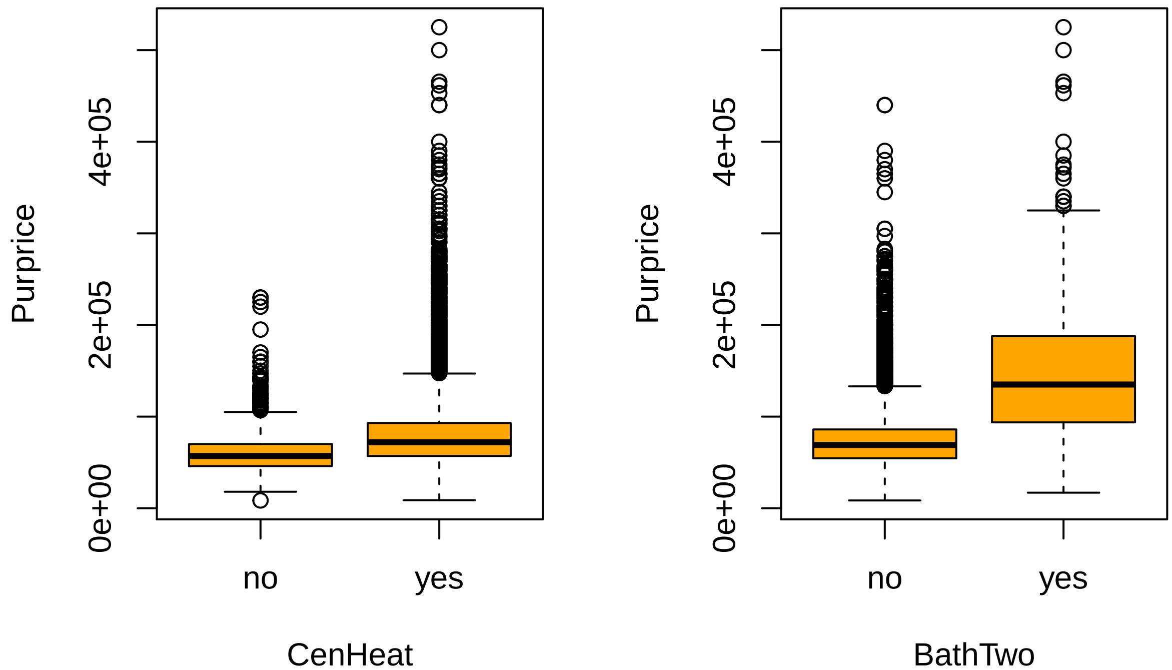
```
##   Easting Northing Purprice Tenfree CenHeat BathTwo NewPropD  FlorArea ProfPct
## 1  545500   173000    85000     yes     yes      no       no  76.16146  0.0000
## 2  525000   177800    71000     yes     yes      no       no  98.45262  6.2500
## 3  531100   183400    60000     yes     yes     yes       no 124.73761  0.0000
## 4  538500   169400    64000     yes     yes      no      yes 127.00000  0.0000
## 5  534000   168400   260000     yes     yes     yes       no 190.40366  9.0909
## 6  528700   168800    48500     yes     yes      no       no  87.00000 16.6667
##      Age     Type   Garage Bedrooms
## 1 Bld60s TypDetch GarSingl BedThree
## 2 Bld80s TypDetch GarSingl BedThree
## 3 PreWW1 TypSemiD HardStnd  BedFour
## 4 Bld80s TypDetch GarSingl BedThree
## 5 Bld80s TypDetch GarDoubl  BedFour
## 6 PreWW1  TypFlat HardStnd BedThree
```

Remove Outliers:

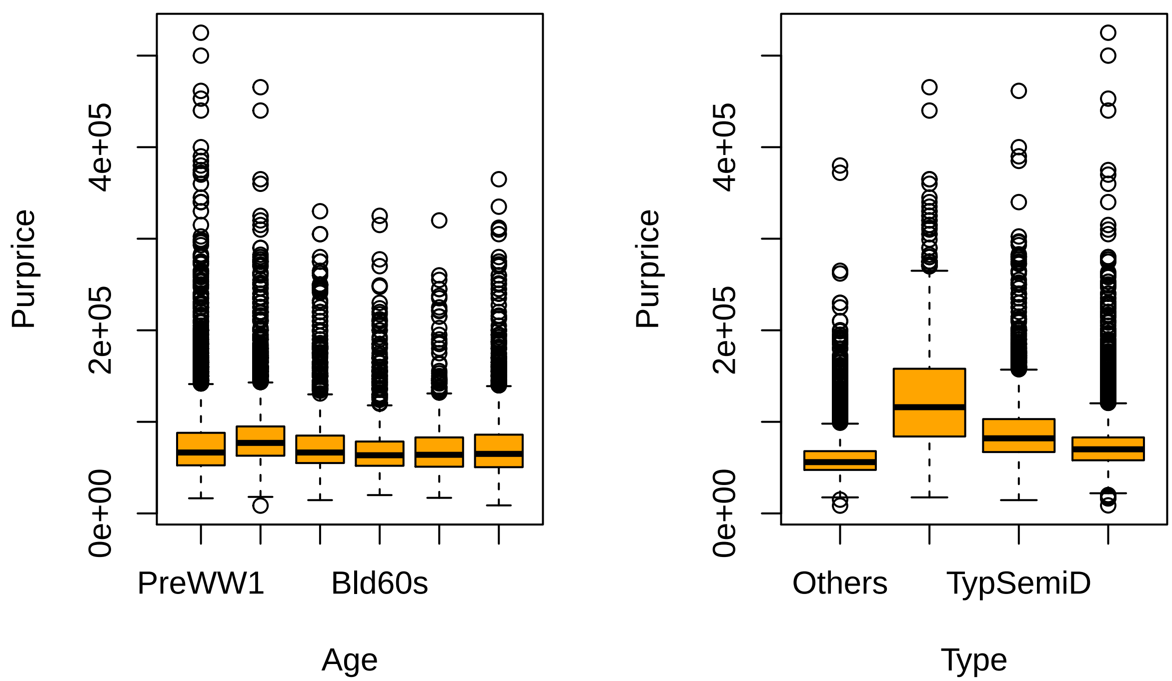
```r
par(mfrow= c(1,2))
boxplot(MyData$Purprice, col = 'orange')
# From boxplot we can see that purprice greater then 600000 is an outlier so we will remove that
MyData <- MyData[MyData$Purprice<600000,]
boxplot(MyData$Purprice, col = 'orange')
```



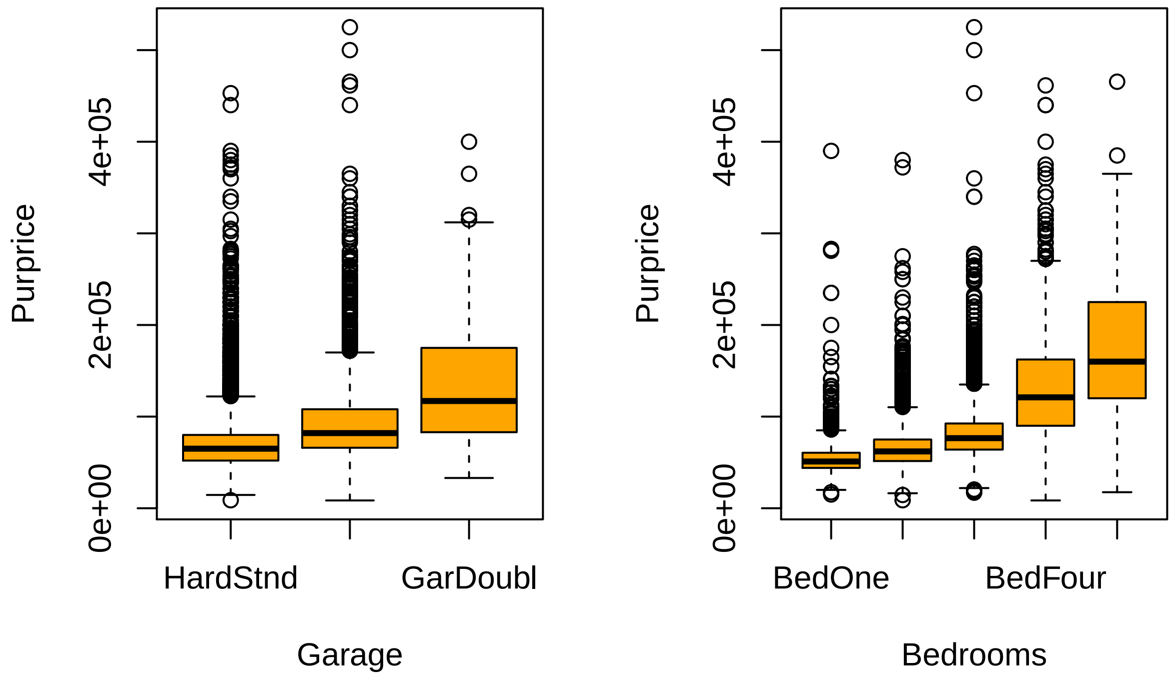

```r
boxplot(Purprice~CenHeat,data=MyData, col = 'orange')
boxplot(Purprice~BathTwo,data=MyData, col = 'orange')
```




```r
boxplot(Purprice~Age,data=MyData, col = 'orange')
boxplot(Purprice~Type,data=MyData, col = 'orange')
```




```r
boxplot(Purprice~Garage,data=MyData, col = 'orange')
boxplot(Purprice~Bedrooms,data=MyData, col = 'orange')
```

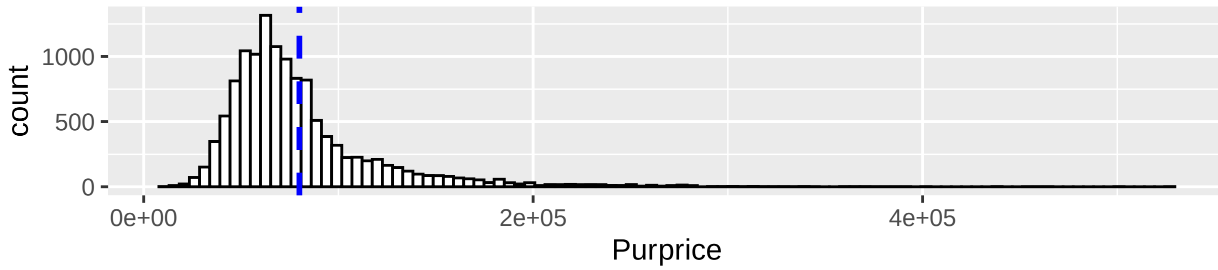


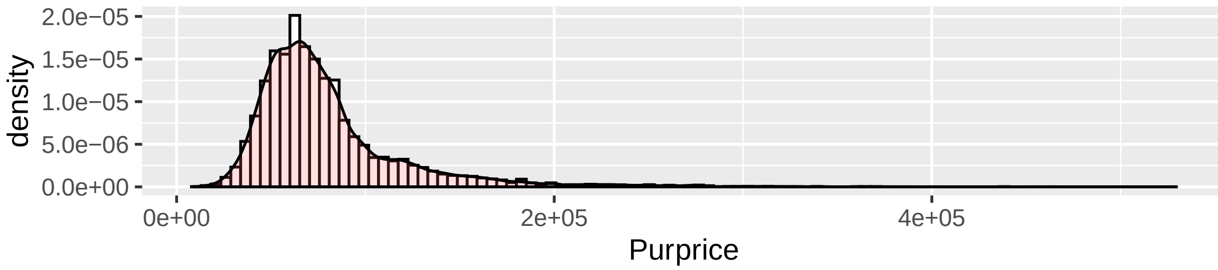

```
par(mfrow= c(1,1))
```

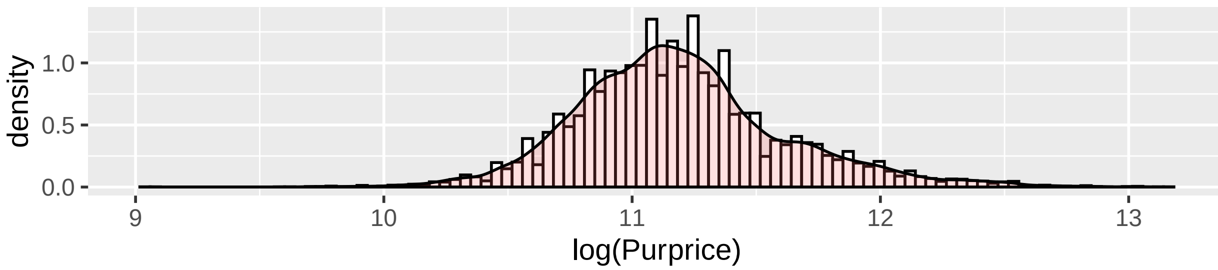
Exploratory Analysis:

Checking for price:

```
p1 <- ggplot(MyData, aes(x=Purprice)) + geom_histogram(bins = 100, color="black", fill="white")+
  geom_vline(aes(xintercept=mean(Purprice)),
            color="blue", linetype="dashed", size=1)

p2 <- ggplot(MyData, aes(x=Purprice)) +
 geom_histogram(bins = 100, aes(y=..density..), colour="black", fill="white")+
 geom_density(alpha=.2, fill="#FF6666")

p3 <- ggplot(MyData, aes(x=log(Purprice))) +
 geom_histogram(bins = 100, aes(y=..density..), colour="black", fill="white")+
 geom_density(alpha=.2, fill="#FF6666")

grid.arrange(p1, p2, p3, nrow=3)
```





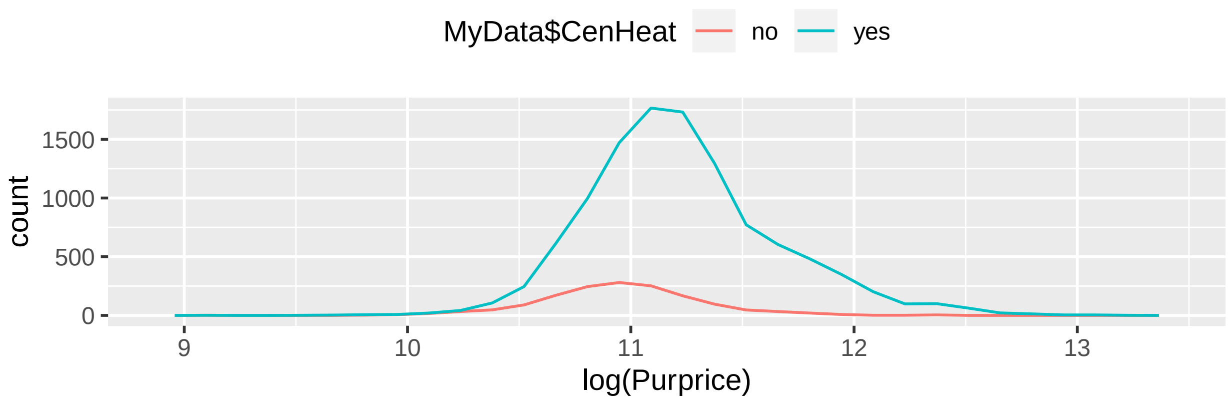


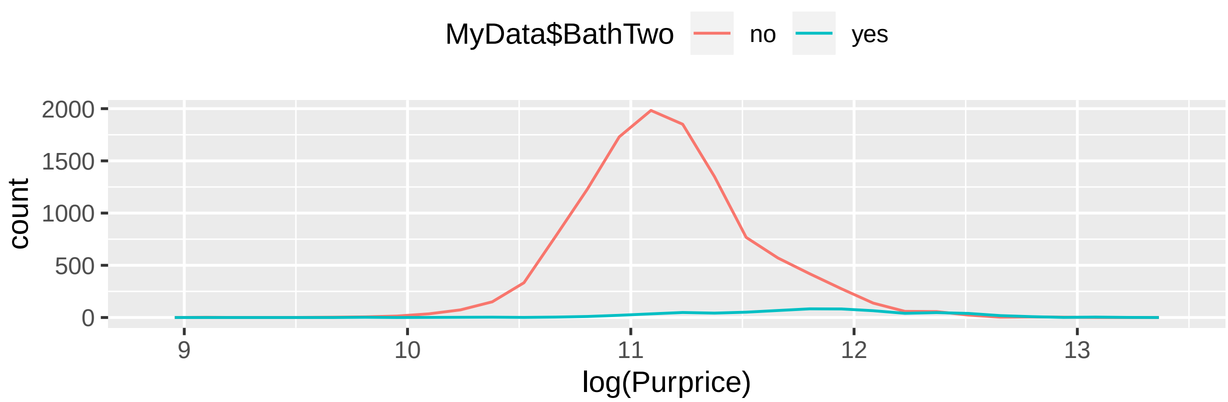
From the plots we can see that it is skewed towards the left so we can say that to make it a normal distribution we have to apply some transformation. After applying log transformation on purprice we see that purprice is normally distributed.

```
p4 <- ggplot(MyData, aes(x=log(Purprice), color=MyData$CenHeat)) +
  geom_histogram(bins = 50, fill= "white", alpha=0.5, position="identity")+
  theme(legend.position = "top")

p5 <- ggplot(MyData, aes(x=log(Purprice), color=MyData$CenHeat)) +
  geom_freqpoly()+
  theme(legend.position = "top")

p6 <- ggplot(MyData, aes(x=log(Purprice), color=MyData$Tenfree)) +
  geom_freqpoly()+
  theme(legend.position = "top")

p7 <- ggplot(MyData, aes(x=log(Purprice), color=MyData$NewPropD)) +
  geom_freqpoly()+
  theme(legend.position = "top")

p8 <- ggplot(MyData, aes(x=log(Purprice), color=MyData$Age)) +
  geom_freqpoly()+
  theme(legend.position = "top")

p9 <- ggplot(MyData, aes(x=log(Purprice), color=MyData$Type)) +
  geom_freqpoly()+
  theme(legend.position = "top")
```

```r
p10 <- ggplot(MyData, aes(x=log(Purprice), color=MyData$Garage)) +
  geom_freqpoly()+
  theme(legend.position = "top")

p11 <- ggplot(MyData, aes(x=log(Purprice), color=MyData$Bedrooms)) +
  geom_freqpoly()+
  theme(legend.position = "top")

p12 <- ggplot(MyData, aes(x=log(Purprice), color=MyData$BathTwo)) +
  geom_freqpoly()+
  theme(legend.position = "top")

grid.arrange(p5, p12, nrow=2)
```

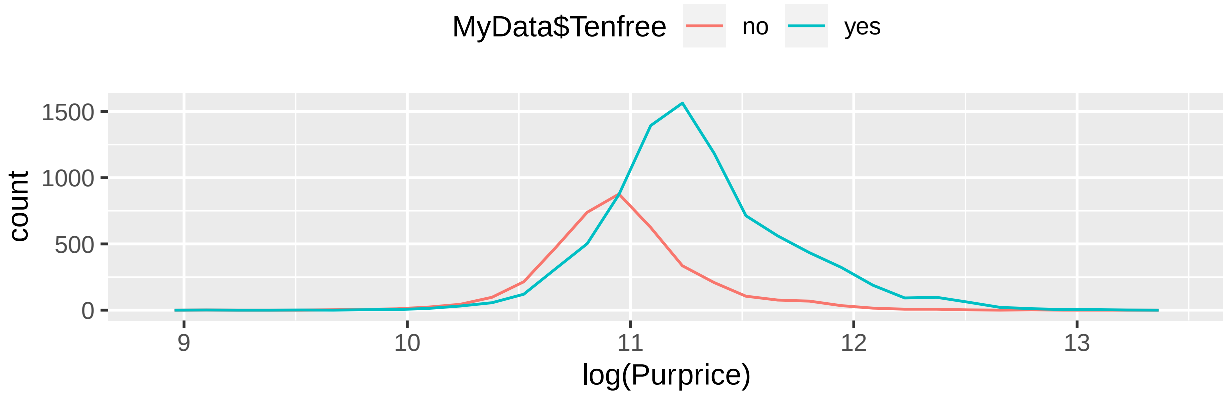
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

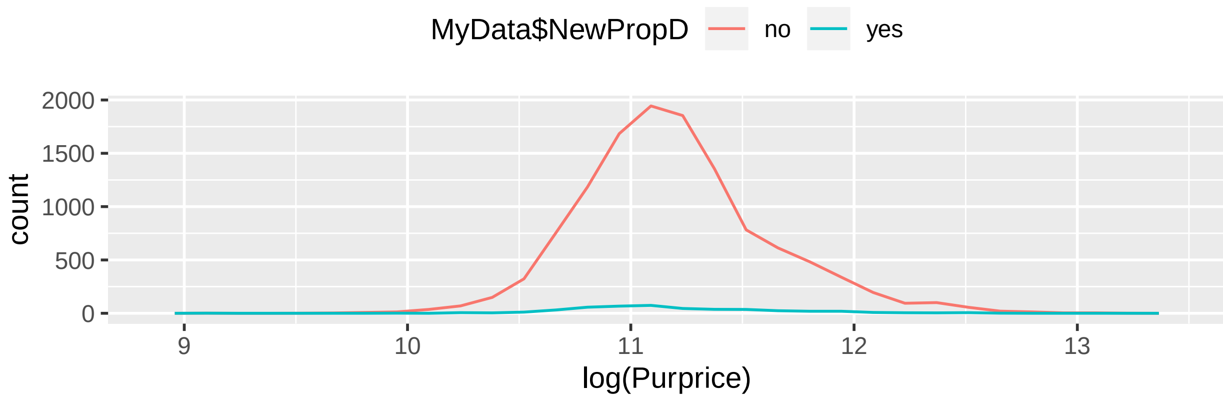





```r
grid.arrange(p6, p7, nrow=2)
```

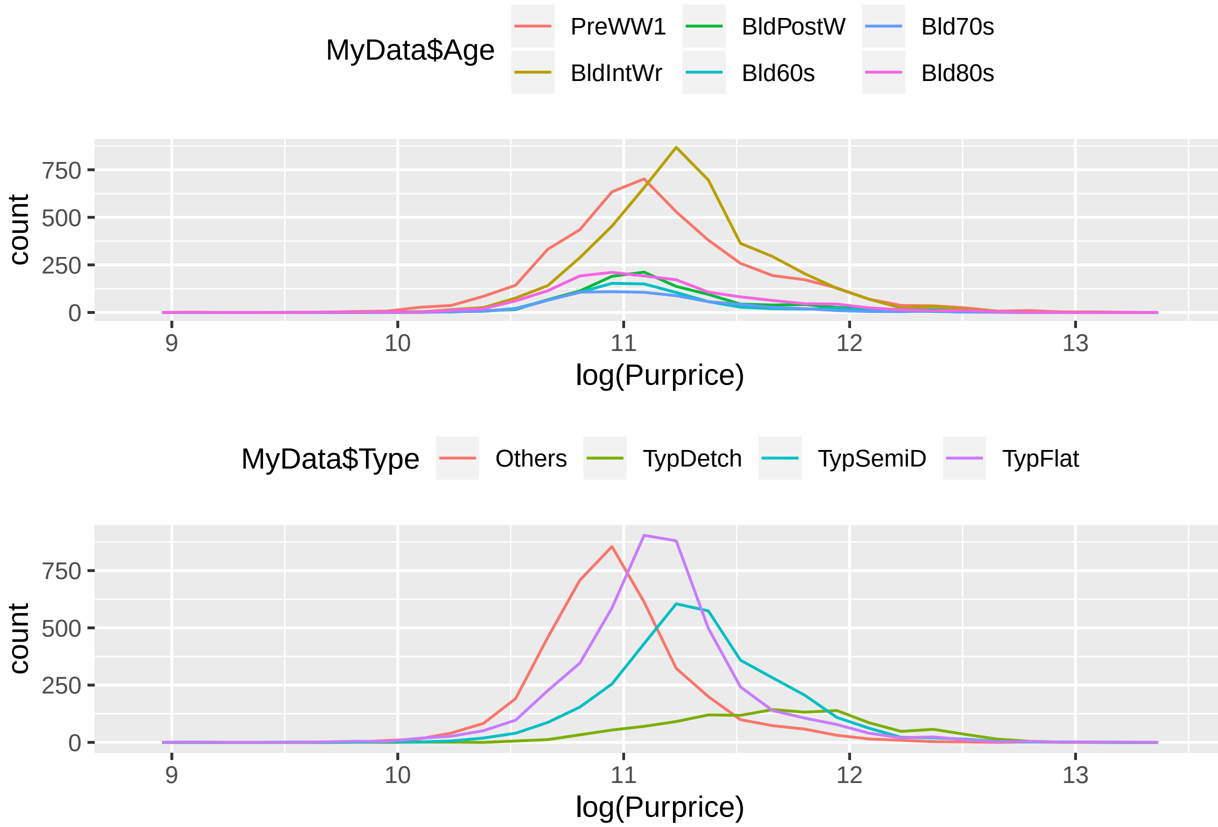
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```





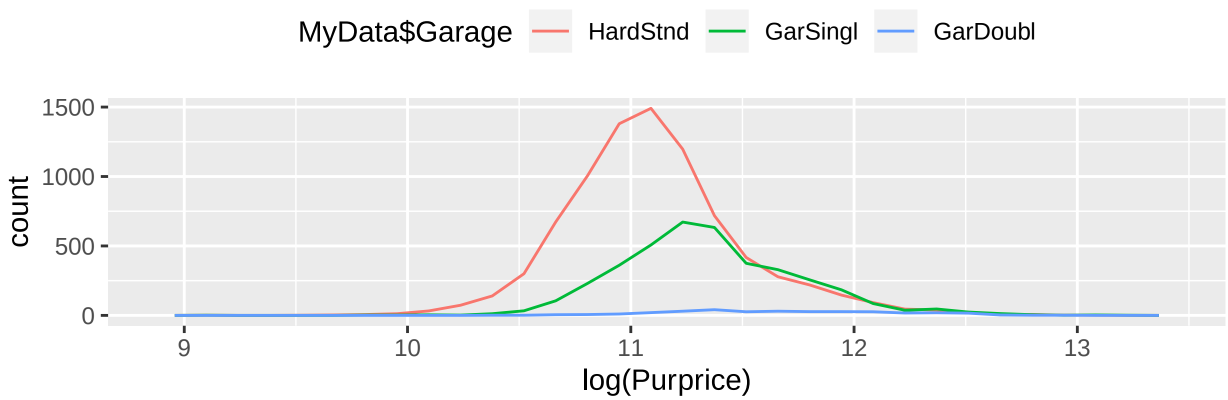

```r
grid.arrange(p8, p9, nrow=2)
```

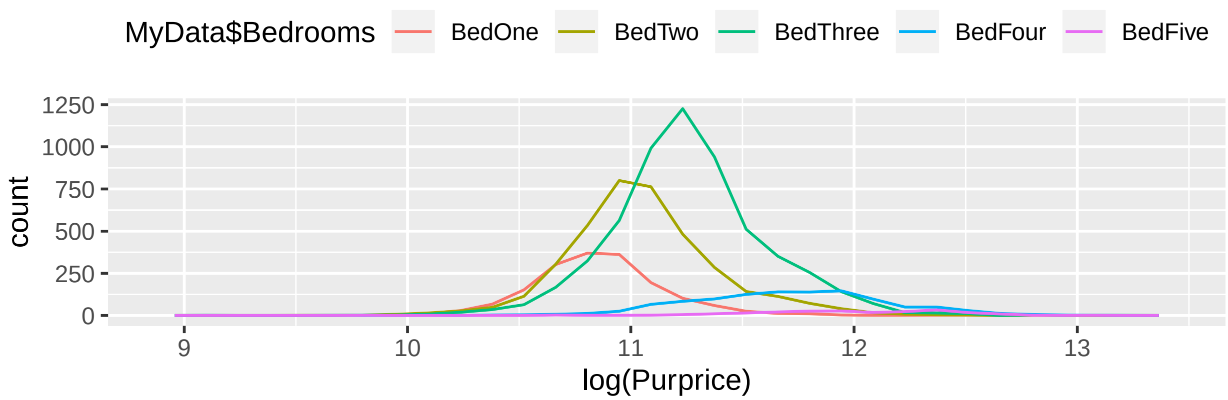
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
grid.arrange(p10, p11, nrow=2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```






```
ld_model <- lm(Purprice~Tenfree+CenHeat+BathTwo+NewPropD+FlorArea+ProfPct+Age+Type+Garage+Bedrooms, data
step <- stepAIC(ld_model, direction="both")
```

```
## Start:  AIC=255965.1
## Purprice ~ Tenfree + CenHeat + BathTwo + NewPropD + FlorArea +
##     ProfPct + Age + Type + Garage + Bedrooms
##
##            Df  Sum of Sq        RSS    AIC
## - NewPropD  1 1.1105e+09 9.2263e+12 255965
## <none>                   9.2252e+12 255965
## - ProfPct   1 2.6037e+09 9.2278e+12 255967
## - Tenfree   1 1.5377e+10 9.2406e+12 255984
## - Garage    2 4.1633e+10 9.2669e+12 256018
## - Bedrooms  4 8.6698e+10 9.3119e+12 256074
## - Age       5 1.3264e+11 9.3579e+12 256134
## - CenHeat   1 1.8185e+11 9.4071e+12 256208
## - Type      3 2.4028e+11 9.4655e+12 256281
## - BathTwo   1 2.9356e+11 9.5188e+12 256356
## - FlorArea  1 2.6197e+12 1.1845e+13 259096
##
## Step:  AIC=255964.6
## Purprice ~ Tenfree + CenHeat + BathTwo + FlorArea + ProfPct +
##     Age + Type + Garage + Bedrooms
##
##            Df  Sum of Sq        RSS    AIC
## <none>                   9.2263e+12 255965
## + NewPropD  1 1.1105e+09 9.2252e+12 255965
## - ProfPct   1 2.6641e+09 9.2290e+12 255966
```

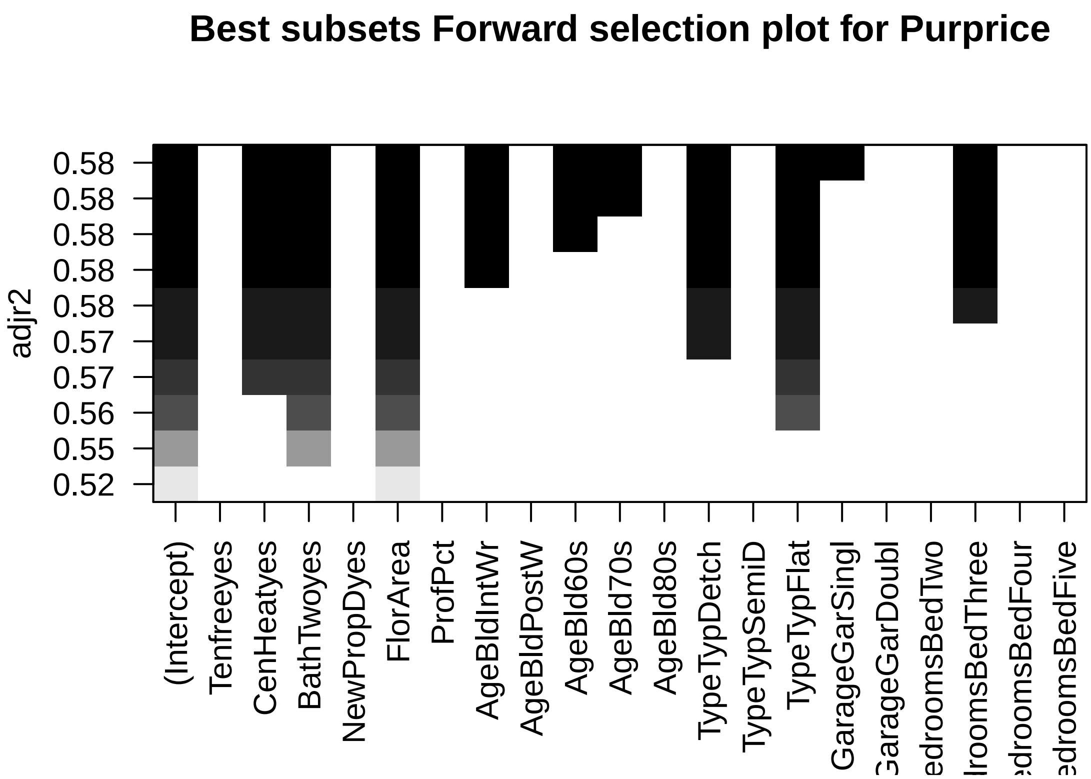
```
## - Tenfree   1 1.5170e+10 9.2415e+12 255983
## - Garage    2 4.1396e+10 9.2677e+12 256017
## - Bedrooms  4 8.6713e+10 9.3131e+12 256074
## - Age       5 1.3364e+11 9.3600e+12 256135
## - CenHeat   1 1.8197e+11 9.4083e+12 256207
## - Type      3 2.4081e+11 9.4671e+12 256282
## - BathTwo   1 2.9504e+11 9.5214e+12 256357
## - FlorArea  1 2.6199e+12 1.1846e+13 259096
```

```r
step$anova
```

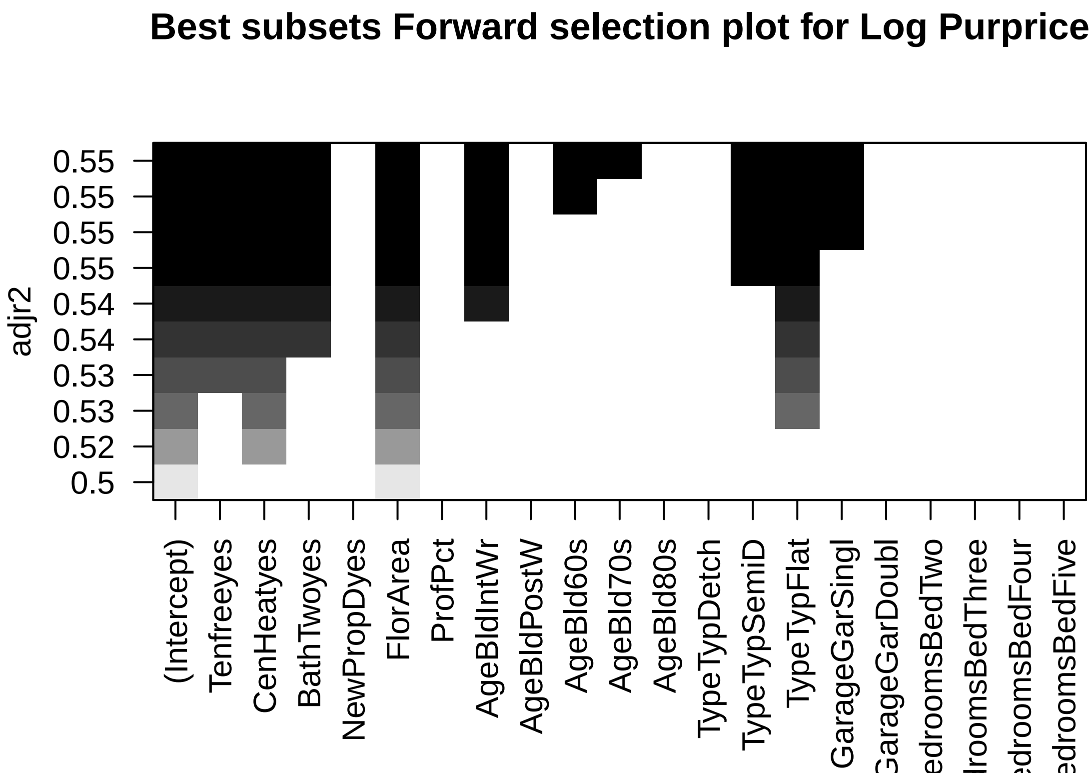
```
## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## Purprice ~ Tenfree + CenHeat + BathTwo + NewPropD + FlorArea +
##     ProfPct + Age + Type + Garage + Bedrooms
##
## Final Model:
## Purprice ~ Tenfree + CenHeat + BathTwo + FlorArea + ProfPct +
##     Age + Type + Garage + Bedrooms
##
##
##         Step Df   Deviance Resid. Df   Resid. Dev      AIC
## 1                              12514 9.225229e+12 255965.1
## 2 - NewPropD  1 1110546555     12515 9.226339e+12 255964.6
```

```r
library(leaps)

set.seed(123)
sample <- sample(c(TRUE, FALSE), nrow(MyData), replace = T, prob = c(0.6,0.4))
train <- MyData[sample, ]
test <- MyData[!sample, ]

#Best subsets plots for Purprice and log(Purprice)
# ld_orgnl <- regsubsets(Purprice~Tenfree+CenHeat+BathTwo+NewPropD+FlorArea+ProfPct+Age+Type+Garage+Bed
# ld_model <- regsubsets(log(Purprice)~Tenfree+CenHeat+BathTwo+NewPropD+FlorArea+ProfPct+Age+Type+Garag
# results <- summary(ld_model)
# plot(ld_orgnl,scale="adjr2")
# title(main= "Best subsets plot for Purprice")
# plot(ld_model,scale="adjr2")
# title(main= "Best subsets plot for Log Purprice")
#
# # extract and plot results
# tibble(predictors = 1:10,
#        adj_R2 = results$adjr2,
#        Cp = results$cp,
#        BIC = results$bic) %>%
#   gather(statistic, value, -predictors) %>%
#   ggplot(aes(predictors, value, color = statistic)) +
#   geom_line(show.legend = F) +
#   geom_point(show.legend = F) +
#   facet_wrap(~ statistic, scales = "free")
#
# which.max(results$adjr2)
```

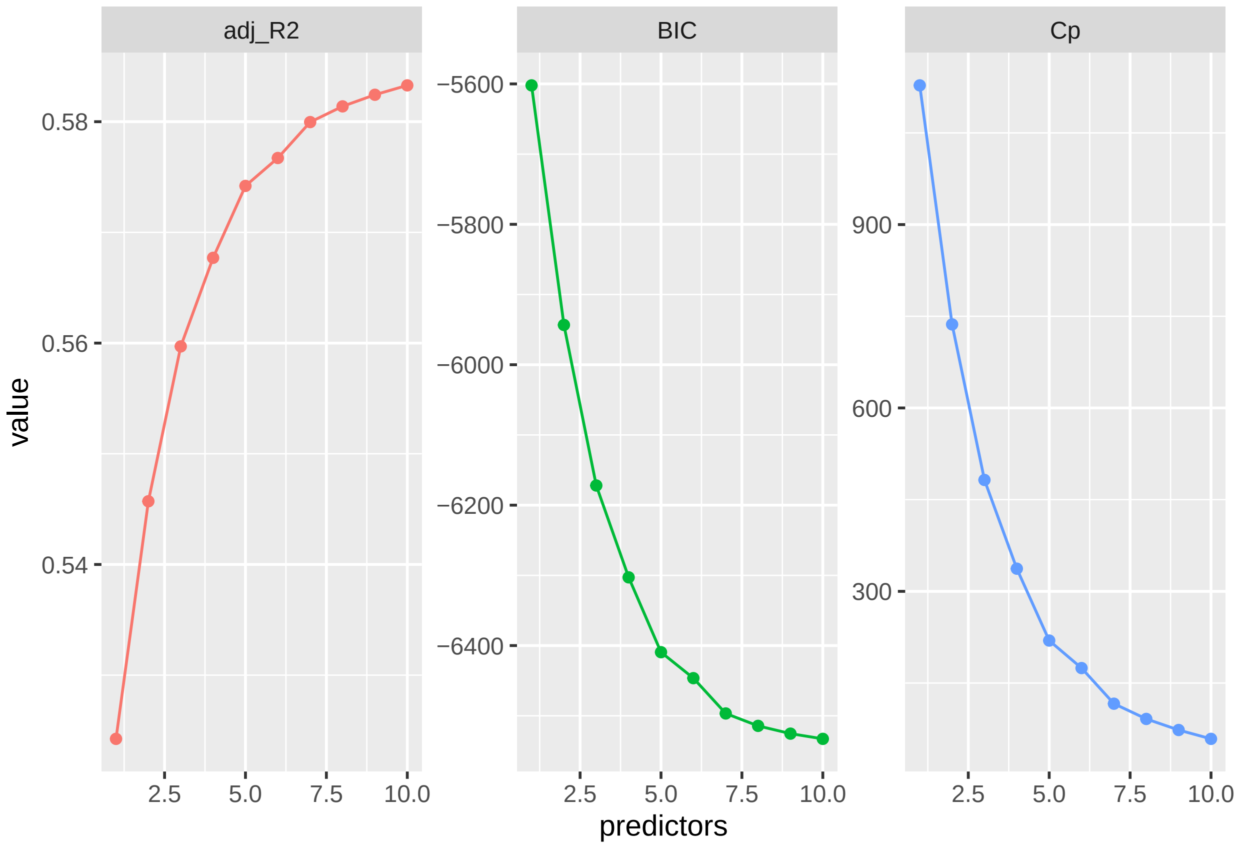
```r
# which.min(results$bic)
# which.min(results$cp)

#Best subsets with Forward selection
ld_orgnl_frwd <- regsubsets(Purprice~Tenfree+CenHeat+BathTwo+NewPropD+FlorArea+ProfPct+Age+Type+Garage+B
ld_frwd <- regsubsets(log(Purprice)~Tenfree+CenHeat+BathTwo+NewPropD+FlorArea+ProfPct+Age+Type+Garage+Be
results <- summary(ld_orgnl_frwd)
# results <- summary(ld_frwd)
plot(ld_orgnl_frwd,scale="adjr2")
title(main= "Best subsets Forward selection plot for Purprice")
```




```r
plot(ld_frwd,scale="adjr2")
title(main= "Best subsets Forward selection plot for Log Purprice")
```



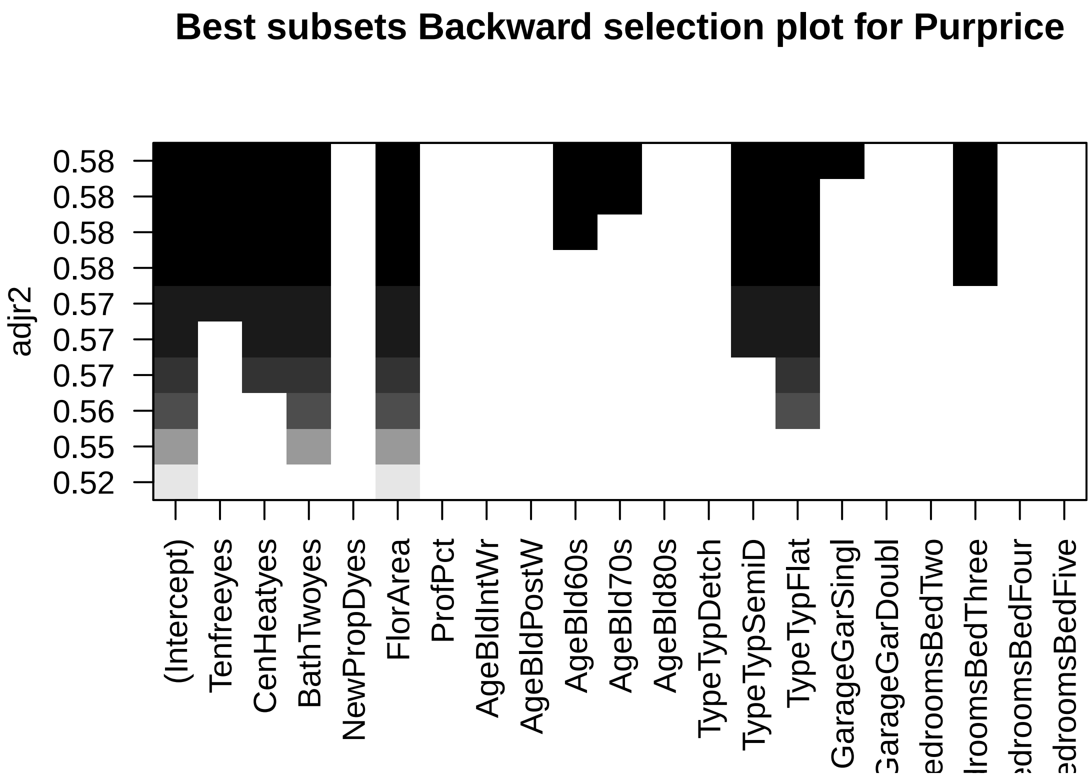

```
tibble(predictors = 1:10,
       adj_R2 = results$adjr2,
       Cp = results$cp,
       BIC = results$bic) %>%
  gather(statistic, value, -predictors) %>%
  ggplot(aes(predictors, value, color = statistic)) +
  geom_line(show.legend = F) +
  geom_point(show.legend = F) +
  facet_wrap(~ statistic, scales = "free")
```



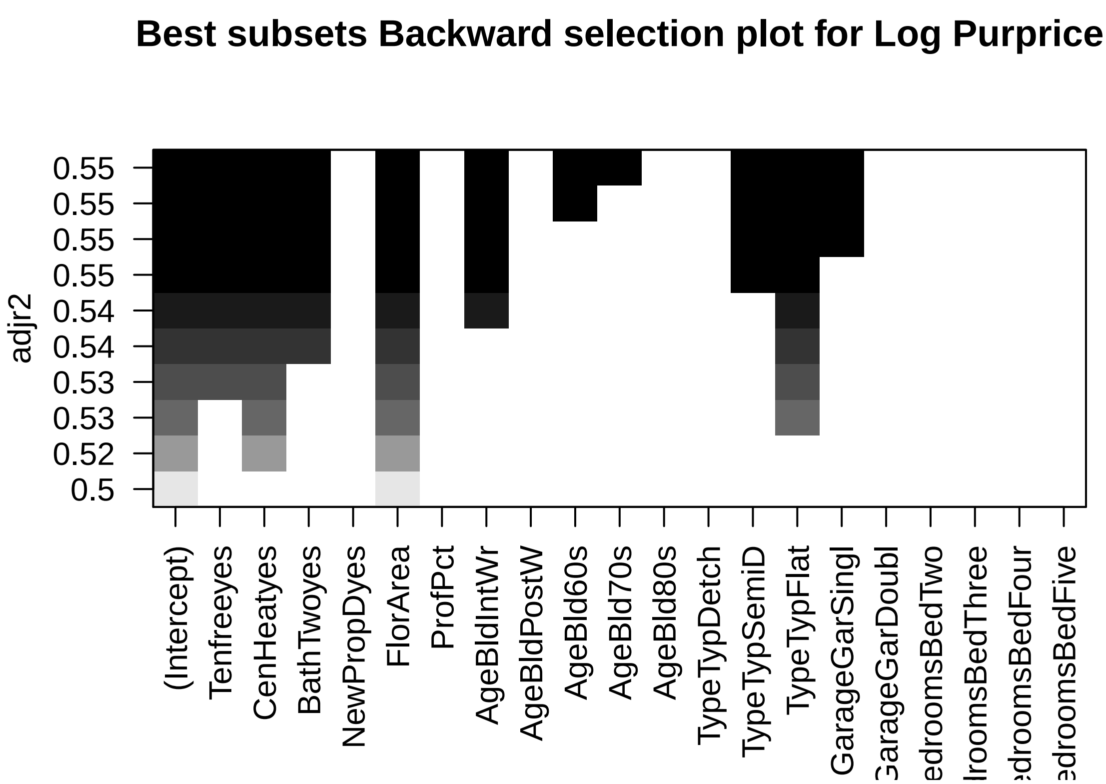

```
which.min(results$cp)
```

```
## [1] 10
```

```
#Best subsets with Backward selection
ld_orgnl_bkwd <- regsubsets(Purprice~Tenfree+CenHeat+BathTwo+NewPropD+FlorArea+ProfPct+Age+Type+Garage+B
ld_bkwd <- regsubsets(log(Purprice)~Tenfree+CenHeat+BathTwo+NewPropD+FlorArea+ProfPct+Age+Type+Garage+Be
results <- summary(ld_bkwd)
plot(ld_orgnl_bkwd,scale="adjr2")
title(main= "Best subsets Backward selection plot for Purprice")
```

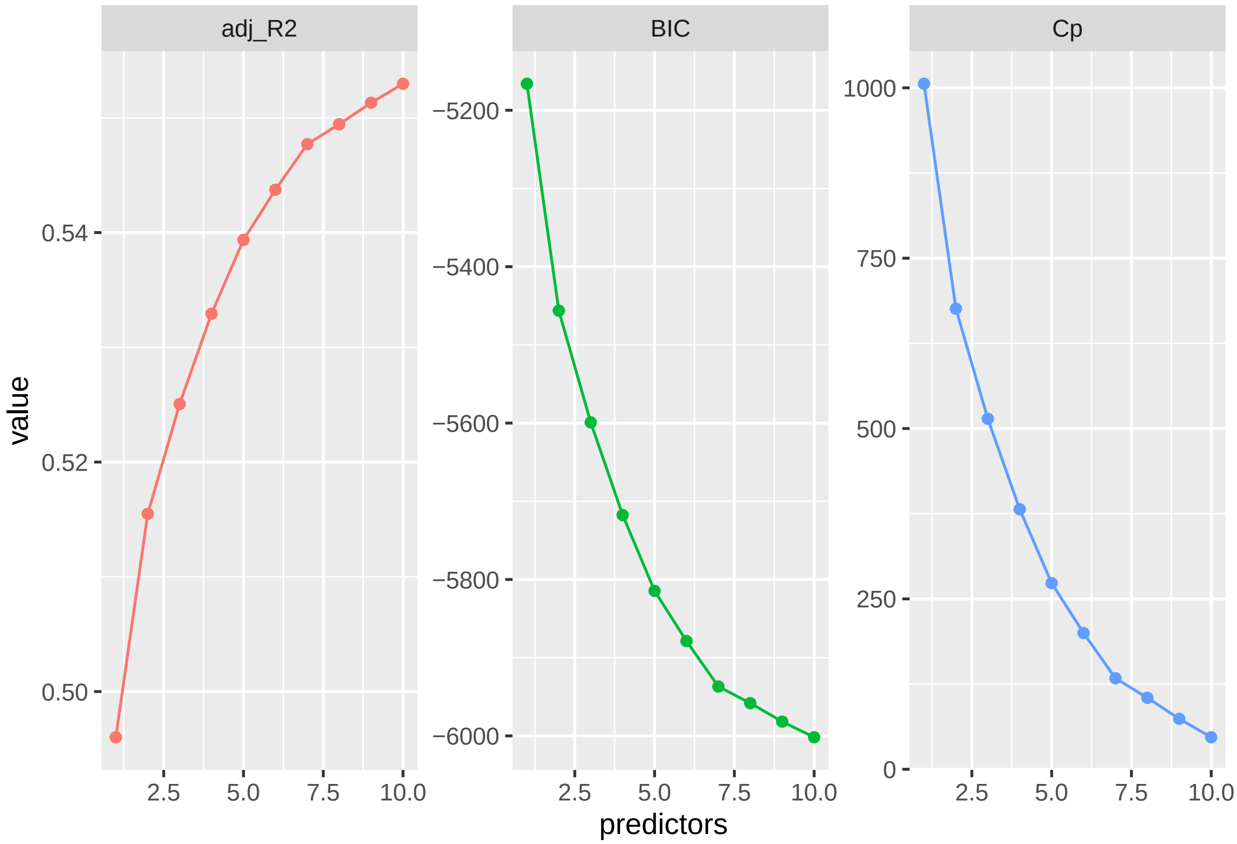



```r
plot(ld_bkwd,scale="adjr2")
title(main= "Best subsets Backward selection plot for Log Purprice")
```




```r
tibble(predictors = 1:10,
       adj_R2 = results$adjr2,
```

```r
      Cp = results$cp,
      BIC = results$bic) %>%
  gather(statistic, value, -predictors) %>%
  ggplot(aes(predictors, value, color = statistic)) +
  geom_line(show.legend = F) +
  geom_point(show.legend = F) +
  facet_wrap(~ statistic, scales = "free")
```




```r
which.min(results$cp)
```

```
## [1] 10
```

```r
#Plotting using models and required number of variables
coef(ld_model,10)
```

```
##      (Intercept)       Tenfreeyes        CenHeatyes       BathTwoyes
##       6184.99958        6176.19785       11851.02315      24006.96253
##      NewPropDyes          FlorArea           ProfPct      AgeBldIntWr
##       1896.22846         677.99715          44.83954       4045.76814
##      AgeBldPostW         AgeBld60s         AgeBld70s        AgeBld80s
##      -1136.56704       -7346.42319       -6725.04819        307.19266
##     TypeTypDetch      TypeTypSemiD       TypeTypFlat   GarageGarSingl
##       5660.72489       -6701.98958      -11590.32066       3797.91709
##    GarageGarDoubl   BedroomsBedTwo BedroomsBedThree  BedroomsBedFour
##       9288.54166       -3432.11465       -7874.67308      -1767.50740
##   BedroomsBedFive
##       3948.63829
```

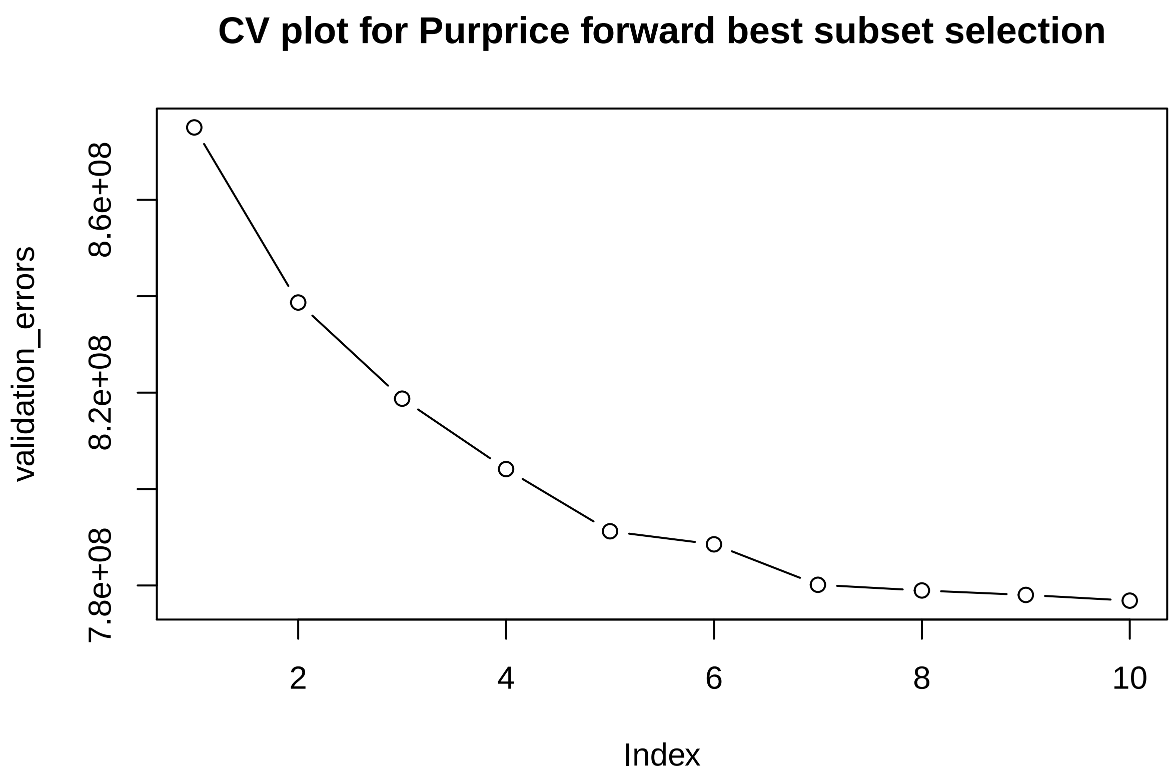
```r
coef(ld_frwd,10)
```

```
##     (Intercept)      Tenfreeyes      CenHeatyes      BathTwoyes        FlorArea 
##    10.345570773     0.141601889     0.172871907     0.163571806     0.006712639 
##     AgeBldIntWr       AgeBld60s       AgeBld70s    TypeTypSemiD     TypeTypFlat 
##     0.050166334    -0.090246092    -0.079859427    -0.096002370    -0.171086108 
## GarageGarSingl 
##     0.054988865
```

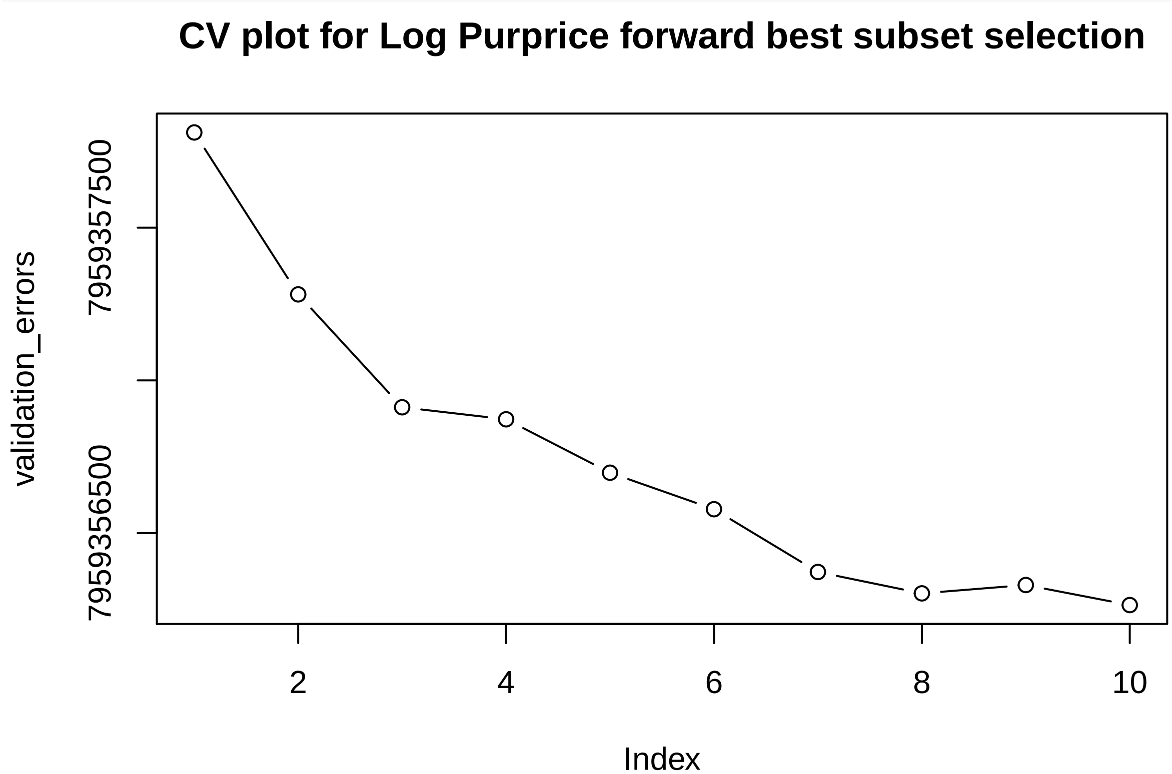
```r
coef(ld_bkwd,10)
```

```
##     (Intercept)      Tenfreeyes      CenHeatyes      BathTwoyes        FlorArea 
##    10.345570773     0.141601889     0.172871907     0.163571806     0.006712639 
##     AgeBldIntWr       AgeBld60s       AgeBld70s    TypeTypSemiD     TypeTypFlat 
##     0.050166334    -0.090246092    -0.079859427    -0.096002370    -0.171086108 
## GarageGarSingl 
##     0.054988865
```

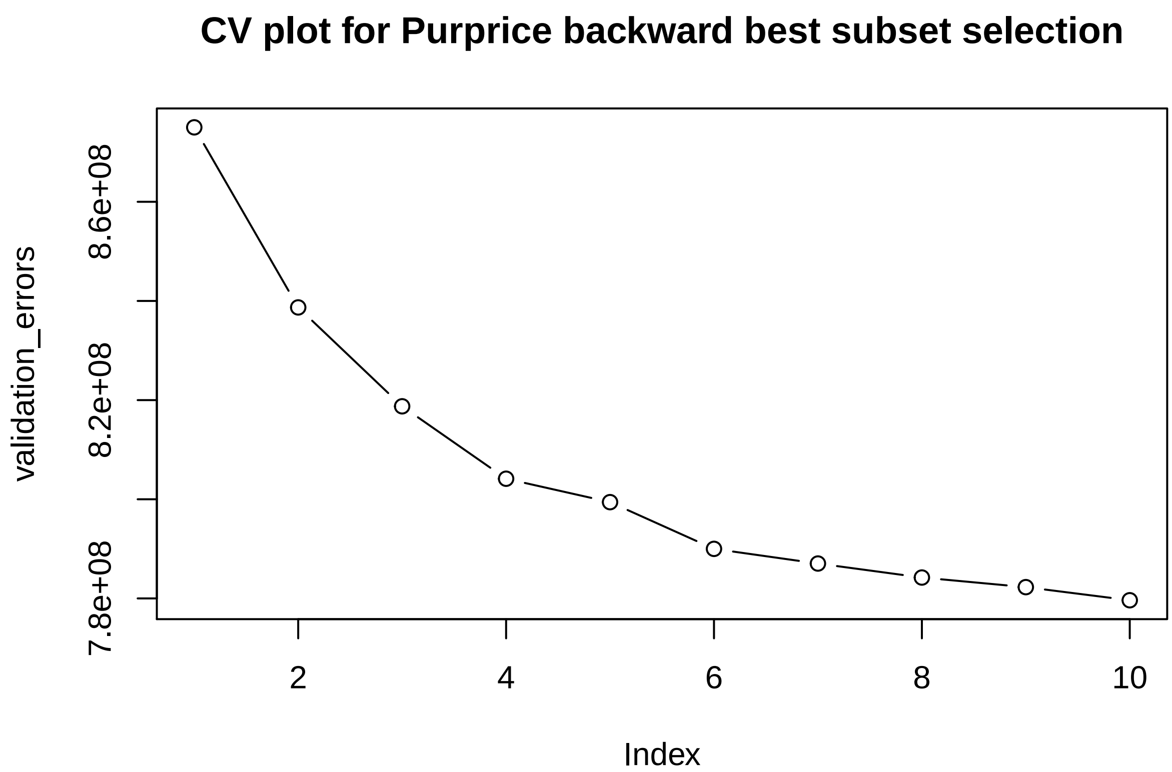
```r
#Cross Validation with test data
test_m <- model.matrix(log(Purprice) ~Tenfree+CenHeat+BathTwo+NewPropD+FlorArea+ProfPct+Age+Type+Garage

validation_errors <- vector("double", length = 10)

val_error <- function(myModel){
for(i in 1:10) {
  coef_x <- coef(myModel, id = i)                        # extract coefficients for model size i
  pred_x <- test_m[ , names(coef_x)] %*% coef_x           # predict salary using matrix algebra
  validation_errors[i] <- mean((test$Purprice - pred_x)^2)  # compute test error btwn actual & predicte
}
plot(validation_errors, type = "b")
}

val_error(myModel = ld_orgnl_frwd)
title(main = "CV plot for Purprice forward best subset selection")
```



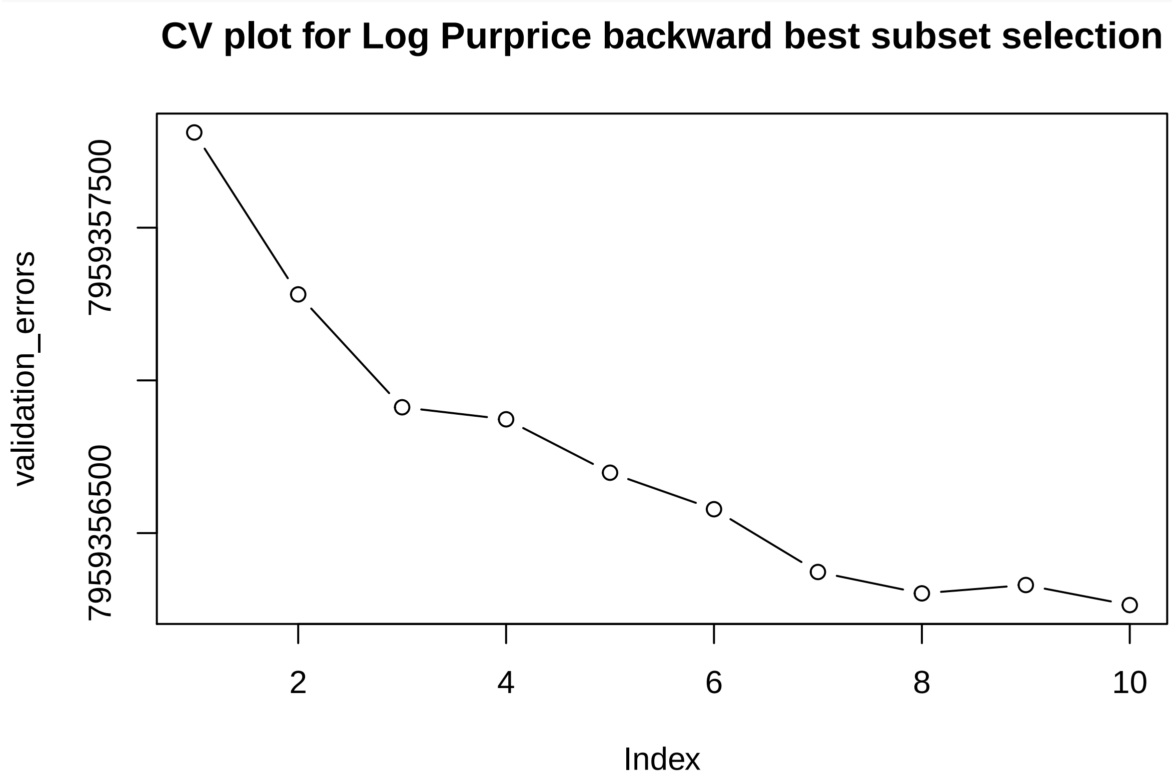

```r
val_error(myModel = ld_frwd)
title(main = "CV plot for Log Purprice forward best subset selection")
```




```r
val_error(myModel = ld_orgnl_bkwd)
title(main = "CV plot for Purprice backward best subset selection")
```

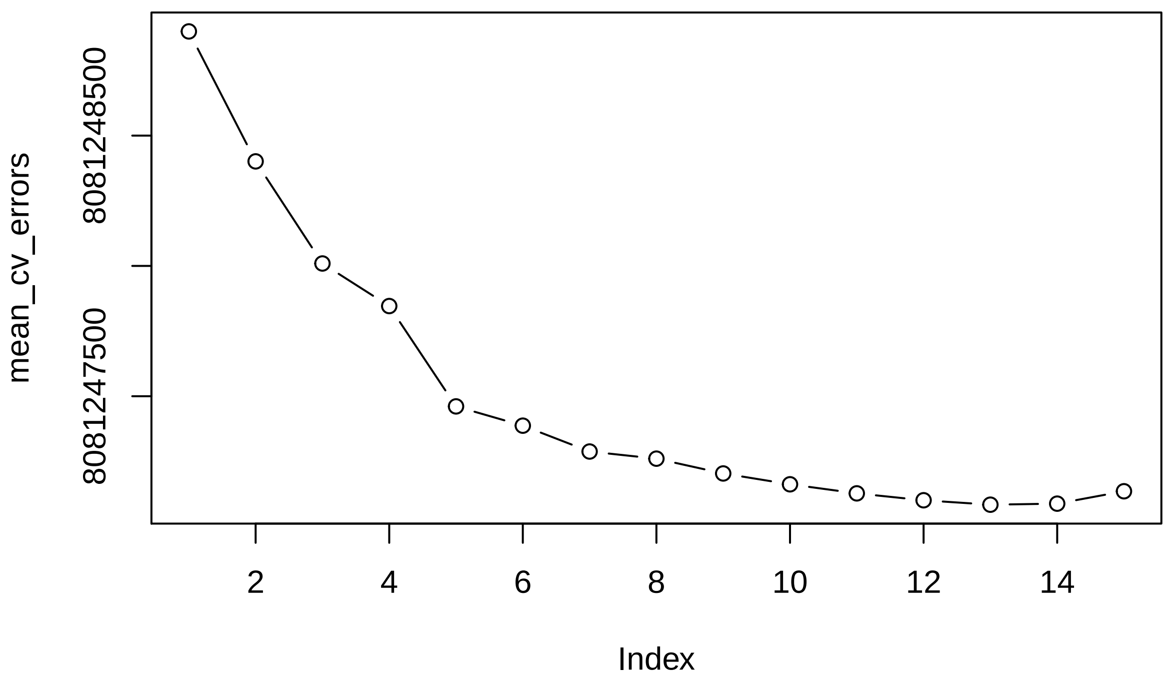



```r
val_error(myModel = ld_bkwd)
title(main = "CV plot for Log Purprice backward best subset selection")
```




```r
# ld_model <- lm(Purprice~FlorArea+BathTwo+CenHeat+Type,data = MyData)
# # ld_model <- lm(Purprice~FlorArea+BathTwo+CenHeat+Type+Easting+Bedrooms+Age,data = MyData)
# summary(ld_model)
```

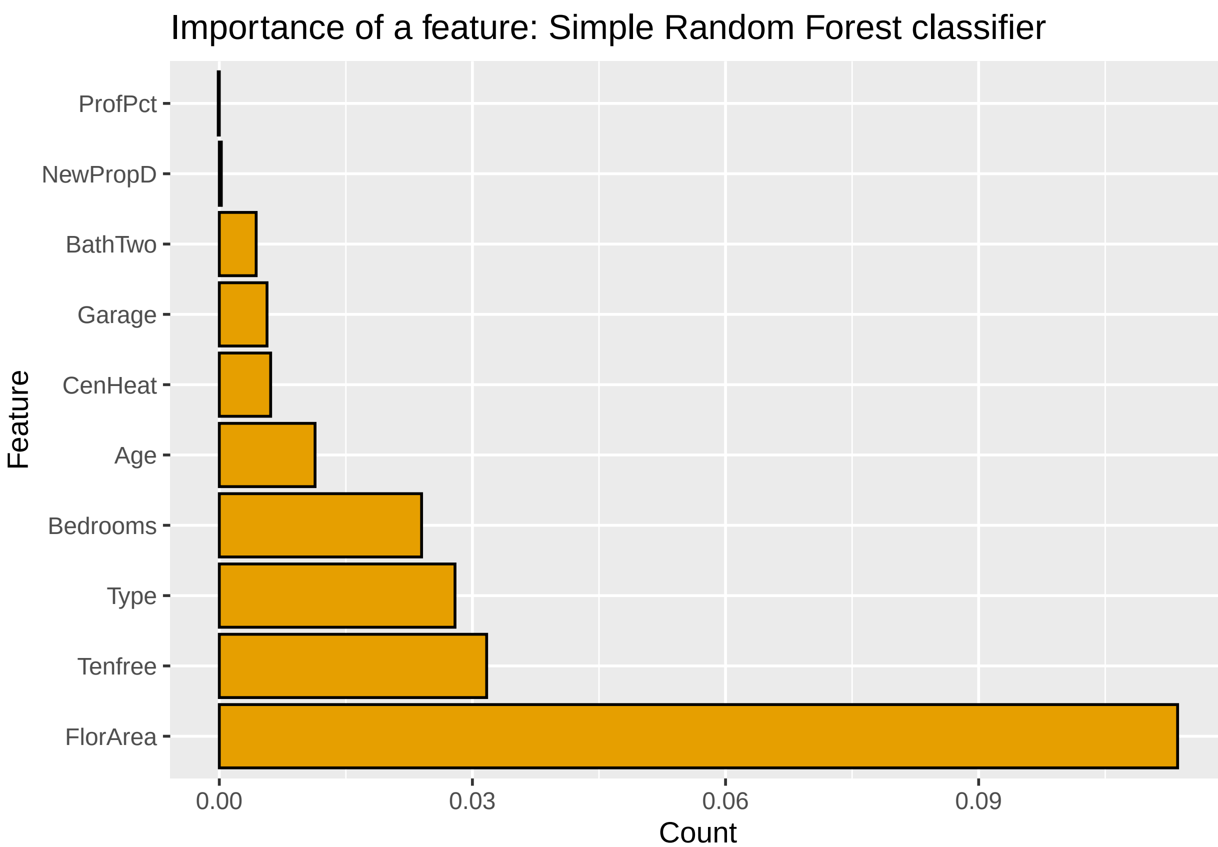
```
predict.regsubsets <- function(object, newdata, id ,...) {
  form <- as.formula(object$call[[2]])
  mat <- model.matrix(form, newdata)
  coefi <- coef(object, id = id)
  xvars <- names(coefi)
  mat[, xvars] %*% coefi
}

k <- 10
set.seed(1)
folds <- sample(1:k, nrow(MyData), replace = TRUE)
cv_errors <- matrix(NA, k, 15, dimnames = list(NULL, paste(1:15)))

for(j in 1:k) {

  # perform best subset on rows not equal to j
  ld_model <- regsubsets(log(Purprice) ~Tenfree+CenHeat+BathTwo+NewPropD+FlorArea+ProfPct+Age+Type+Garag

  # perform cross-validation
  for( i in 1:15) {
    pred_x <- predict.regsubsets(ld_model, MyData[folds == j, ], id = i)
    cv_errors[j, i] <- mean((MyData$Purprice[folds == j] - pred_x)^2)
    }
}

mean_cv_errors <- colMeans(cv_errors)

plot(mean_cv_errors, type = "b")
```




```
final_best <- regsubsets(log(Purprice) ~Tenfree+CenHeat+BathTwo+NewPropD+FlorArea+ProfPct+Age+Type+Garag
coef(final_best, 15)
```

```
##      (Intercept)       Tenfreeyes       CenHeatyes       BathTwoyes
##      10.353520267      0.131989007      0.172189237      0.157698257
##         FlorArea      AgeBldIntWr      AgeBldPostW        AgeBld60s
##      0.006400832      0.046193006     -0.035729546     -0.092546361
##        AgeBld70s     TypeTypSemiD      TypeTypFlat    GarageGarSingl
##     -0.083574547     -0.094678432     -0.168521161      0.060540511
##    GarageGarDoubl    BedroomsBedTwo BedroomsBedThree  BedroomsBedFour
##      0.084110135      0.033080239      0.031098321      0.065597488
```

```
fit <- randomForest(log(Purprice) ~Tenfree+CenHeat+BathTwo+NewPropD+FlorArea+ProfPct+Age+Type+Garage+Bed
          data = train,importance=TRUE,ntree=60)

importance.features <- tibble::rownames_to_column(data.frame(fit$importance[,c(1)]))
colnames(importance.features) <- c("rowname", "value")

ggplot(importance.features, aes(x = reorder(rowname, -value), y = value)) +
  geom_bar(stat = "identity", position = "dodge", fill="#E69F00", colour="black") +
  xlab("Feature") + ylab("Count") + ggtitle("Importance of a feature: Simple Random Forest classifier")
  coord_flip()
```



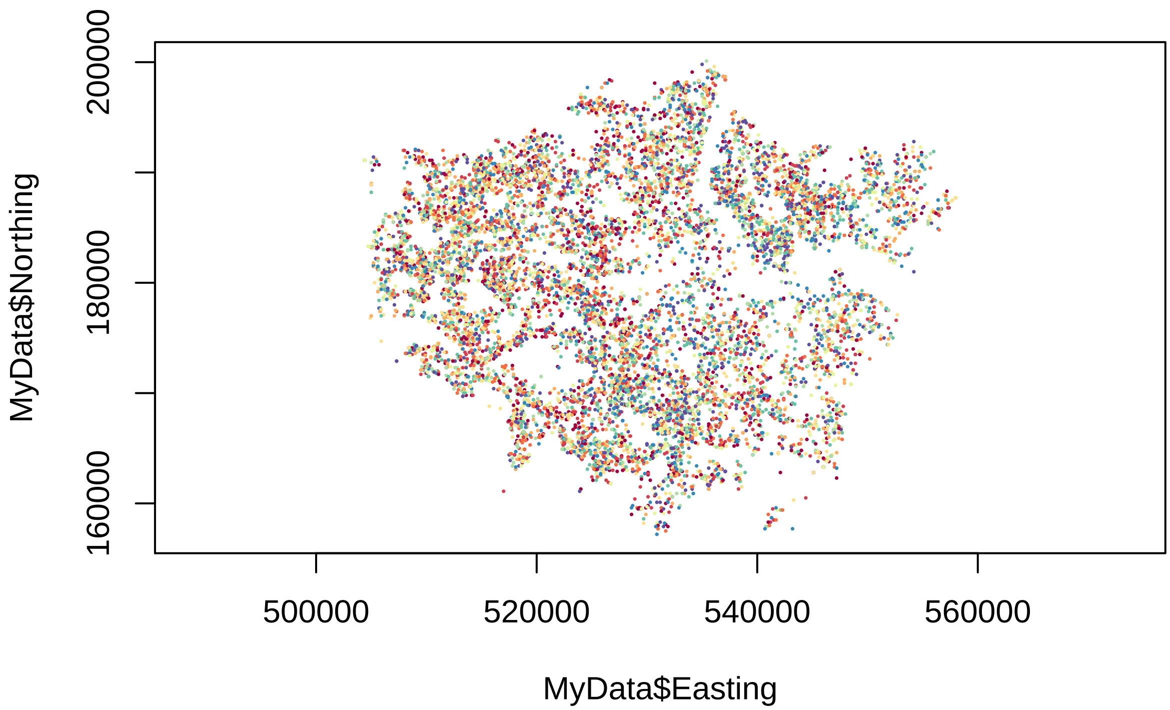
#printing the final model

```
model.9v <- lm(Purprice~FlorArea+Bedrooms+Type+BathTwo+Garage+Tenfree+CenHeat+Age+ProfPct,data=MyData)
summary(model.9v)
```

```
##
## Call:
## lm(formula = Purprice ~ FlorArea + Bedrooms + Type + BathTwo +
##     Garage + Tenfree + CenHeat + Age + ProfPct, data = MyData)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -136550  -13463   -1378   10388  371517
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        6183.51    1187.85   5.206 1.96e-07 ***
## FlorArea            678.01      11.37  59.613  < 2e-16 ***
## BedroomsBedTwo    -3434.19     869.05  -3.952 7.80e-05 ***
## BedroomsBedThree  -7872.74    1068.10  -7.371 1.80e-13 ***
## BedroomsBedFour   -1749.22    1541.74  -1.135 0.256574
## BedroomsBedFive    3937.03    2504.03   1.572 0.115911
## TypeTypDetch       5715.04    1657.99   3.447 0.000569 ***
## TypeTypSemiD      -6671.47    1440.82  -4.630 3.69e-06 ***
## TypeTypFlat      -11564.76    1395.83  -8.285  < 2e-16 ***
## BathTwoyes        24054.83    1202.43  20.005  < 2e-16 ***
## GarageGarSingl     3784.77     614.42   6.160 7.50e-10 ***
## GarageGarDoubl     9266.06    1676.12   5.528 3.30e-08 ***
## Tenfreeyes         6132.37    1351.89   4.536 5.78e-06 ***
## CenHeatyes        11855.05     754.57  15.711  < 2e-16 ***
## AgeBldIntWr        4052.62     656.80   6.170 7.03e-10 ***
## AgeBldPostW       -1135.27     975.05  -1.164 0.244316
## AgeBld60s         -7345.56    1089.66  -6.741 1.64e-11 ***
## AgeBld70s         -6721.82    1164.33  -5.773 7.97e-09 ***
## AgeBld80s           915.55     898.67   1.019 0.308325
## ProfPct              45.35      23.86   1.901 0.057327 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27150 on 12515 degrees of freedom
## Multiple R-squared:  0.5648, Adjusted R-squared:  0.5641
## F-statistic: 854.8 on 19 and 12515 DF,  p-value: < 2.2e-16
```

```r
library(classInt)
library(RColorBrewer)

nClass = 10
Palette <- rev(brewer.pal(nClass,"Spectral"))
Classes <- classIntervals(MyData$Purprice,nClass,"quantile")
Colours <- findColours(Classes,Palette)
plot(MyData$Easting,MyData$Northing,pch=16,cex=0.25,col=Colours,asp=1)
```



Geography - look at trends with linear and quadratic trend surfaces

```
x <- MyData$Easting/1000
y <- MyData$Northing/1000
m.tr1 <- lm(Purprice~x+y,data=MyData)
AIC(m.tr1)
```

```
## [1] 301910.2
```

```
m.tr2 <- lm(Purprice~x+y+I(x^2)+I(y^2)+I(x*y),data=MyData)
AIC(m.tr2)
```

```
## [1] 301887.1
```

```
summary(m.tr1) # lower prices as we move east, slightly lower as w move south
```

```
##
## Call:
## lm(formula = Purprice ~ x + y, data = MyData)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -72863 -24957 -10018   9714 443417
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 165151.00   17668.70   9.347  < 2e-16 ***
## x             -135.10      30.42  -4.441 9.03e-06 ***
## y              -77.06      40.45  -1.905   0.0568 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 41090 on 12532 degrees of freedom
## Multiple R-squared:  0.001854,   Adjusted R-squared:  0.001694
## F-statistic: 11.64 on 2 and 12532 DF,  p-value: 8.937e-06
```

```r
summary(m.tr2) # lower AIC # higher price as we move west
```

```
##
## Call:
## lm(formula = Purprice ~ x + y + I(x^2) + I(y^2) + I(x * y), data = MyData)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -73924 -24782  -9828   9862 444261
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3.153e+06  8.741e+05  -3.607 0.000311 ***
## x            1.225e+04  2.793e+03   4.387 1.16e-05 ***
## y            3.525e+02  2.916e+03   0.121 0.903766
## I(x^2)      -1.074e+01  2.555e+00  -4.203 2.66e-05 ***
## I(y^2)       7.372e+00  4.717e+00   1.563 0.118080
## I(x * y)    -5.727e+00  4.323e+00  -1.325 0.185350
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 41050 on 12529 degrees of freedom
## Multiple R-squared:  0.004172,   Adjusted R-squared:  0.003774
## F-statistic:  10.5 on 5 and 12529 DF,  p-value: 4.507e-10
```

```r
stepAIC(m.tr2)
```

```
## Start:  AIC=266312.3
## Purprice ~ x + y + I(x^2) + I(y^2) + I(x * y)
##
##            Df  Sum of Sq        RSS    AIC
## - y         1 2.4632e+07 2.1111e+13 266310
## - I(x * y)  1 2.9561e+09 2.1114e+13 266312
## <none>                   2.1111e+13 266312
## - I(y^2)    1 4.1163e+09 2.1115e+13 266313
## - I(x^2)    1 2.9761e+10 2.1141e+13 266328
## - x         1 3.2429e+10 2.1143e+13 266330
##
## Step:  AIC=266310.3
## Purprice ~ x + I(x^2) + I(y^2) + I(x * y)
##
##            Df  Sum of Sq        RSS    AIC
## <none>                   2.1111e+13 266310
## - I(y^2)    1 7.3282e+09 2.1118e+13 266313
## - I(x * y)  1 7.5460e+09 2.1119e+13 266313
## - I(x^2)    1 3.0205e+10 2.1141e+13 266326
## - x         1 3.7066e+10 2.1148e+13 266330

##
## Call:
## lm(formula = Purprice ~ x + I(x^2) + I(y^2) + I(x * y), data = MyData)
##
```

```
## Coefficients:
## (Intercept)            x       I(x^2)       I(y^2)     I(x * y)
##  -3.087e+06    1.213e+04   -1.069e+01    7.725e+00   -5.300e+00
```

Explore variation by borough - first load the data

```r
library(rgdal)
```

```
## Loading required package: sp
```

```
## rgdal: version: 1.4-8, (SVN revision 845)
##  Geospatial Data Abstraction Library extensions to R successfully loaded
##  Loaded GDAL runtime: GDAL 2.2.3, released 2017/11/20
##  Path to GDAL shared files: /usr/share/gdal/2.2
##  GDAL binary built with GEOS: TRUE
##  Loaded PROJ.4 runtime: Rel. 4.9.3, 15 August 2016, [PJ_VERSION: 493]
##  Path to PROJ.4 shared files: (autodetected)
##  Linking to sp version: 1.3-2
```

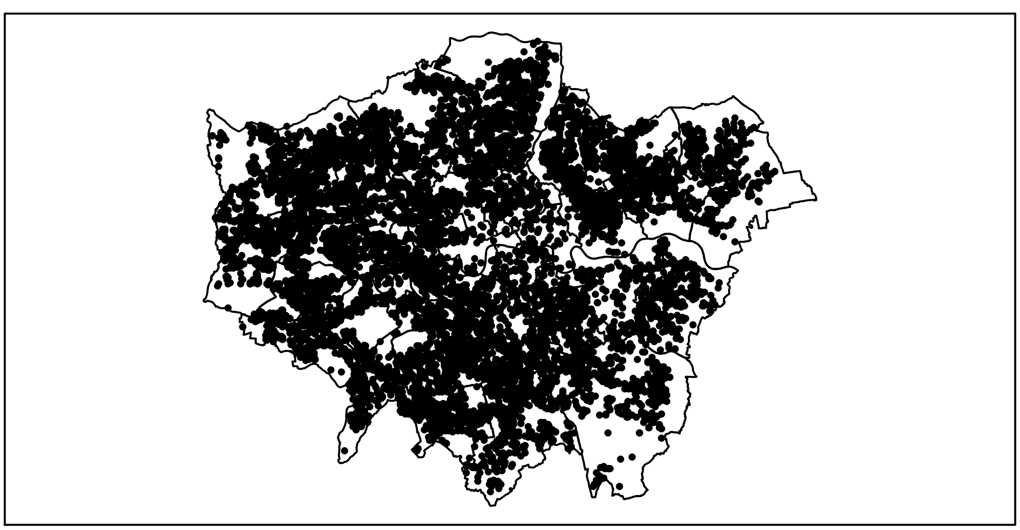
```r
library(rgeos)
```

```
## rgeos version: 0.5-2, (SVN revision 621)
##  GEOS runtime version: 3.6.2-CAPI-1.10.2
##  Linking to sp version: 1.3-1
##  Polygon checking: TRUE
```

```r
LB <- readOGR(dsn="LondonBoroughs",layer="LondonBoroughs",stringsAsFactors=FALSE)  # Boroughs
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "/users/students/19251101/HousingProject/LondonBoroughs", layer: "LondonBoroughs"
## with 33 features
## It has 15 fields
## Integer64 fields read as strings:  NUMBER NUMBER0 POLYGON_ID UNIT_ID
```

```r
LH <- SpatialPointsDataFrame(MyData[,1:2],MyData)                    # Houses
proj4string(LH) <- CRS(proj4string(LB))                              # copy CRS
plot(LB)
points(LH,pch=16,cex=0.5)
box()
```



Add Brough names to data - explore by type and borough - we'll need to do an overlay

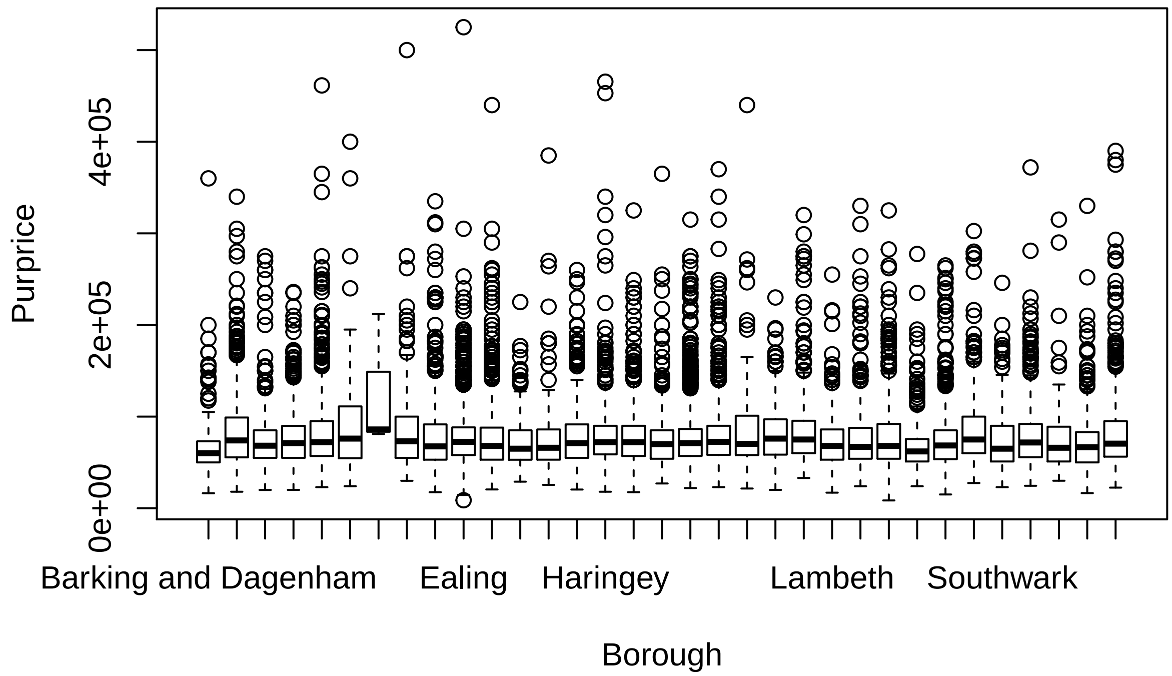
```r
LHLB <- over(LH,LB)    # spatial join: points first, then polygons
dim(LHLB)
```

```
## [1] 12535    15
```

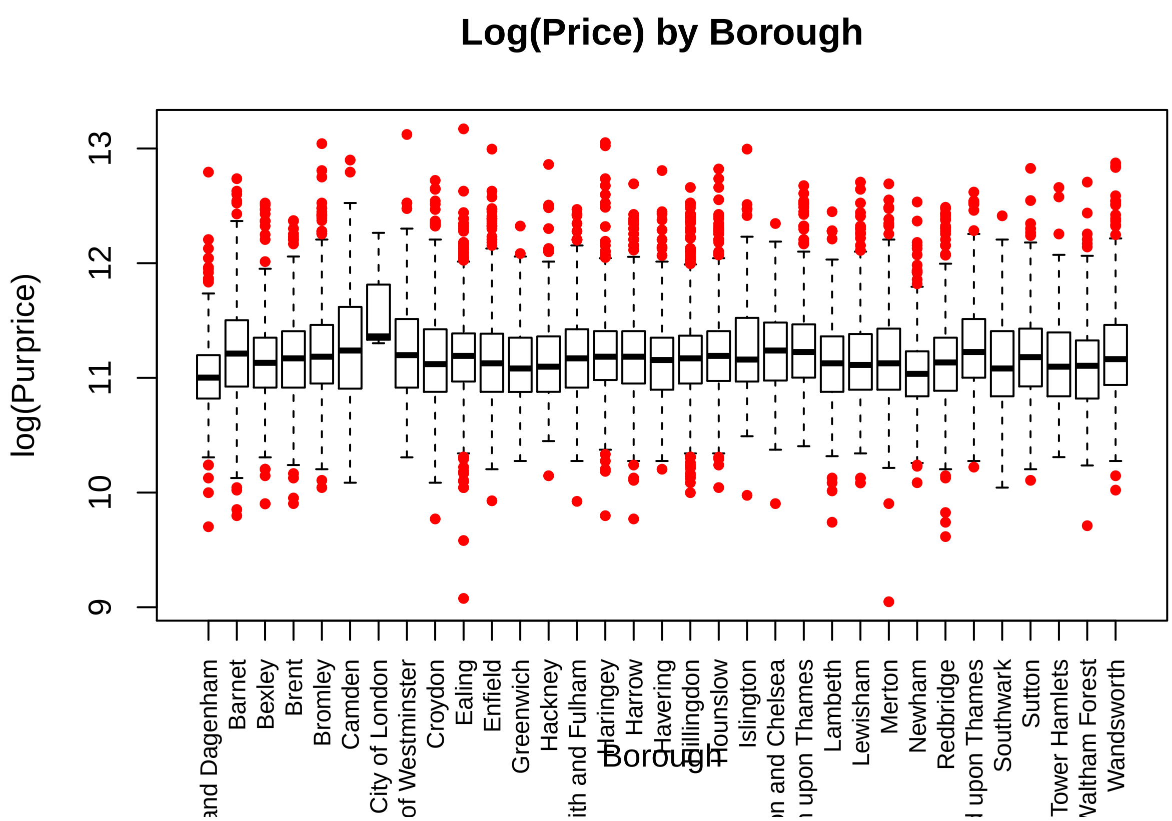
```r
head(LHLB)             # data frame has LB attributes in LH order
```

```
##                                  NAME AREA_CODE     DESCRIPTIO
## 1                  Bexley London Boro       LBO London Borough
## 2 Hammersmith and Fulham London Boro       LBO London Borough
## 3               Islington London Boro       LBO London Borough
## 4                 Bromley London Boro       LBO London Borough
## 5                 Croydon London Boro       LBO London Borough
## 6                  Merton London Boro       LBO London Borough
##                  FILE_NAME NUMBER NUMBER0 POLYGON_ID UNIT_ID      CODE
## 1 GREATER_LONDON_AUTHORITY     42    1080      50891   10759 E09000004
## 2 GREATER_LONDON_AUTHORITY     70    1254      50647   11259 E09000013
## 3 GREATER_LONDON_AUTHORITY     84    1357      50581   11281 E09000019
## 4 GREATER_LONDON_AUTHORITY      9     805      50904   10772 E09000006
## 5 GREATER_LONDON_AUTHORITY      6     781      51330   10896 E09000008
## 6 GREATER_LONDON_AUTHORITY     59    1213     122401   10995 E09000024
##    HECTARES    AREA TYPE_CODE                DESCRIPT0 TYPE_COD0 DESCRIPT1
## 1  6428.649 371.119        AA CIVIL ADMINISTRATION AREA      <NA>      <NA>
## 2  1715.409  75.648        AA CIVIL ADMINISTRATION AREA      <NA>      <NA>
## 3  1485.664   0.000        AA CIVIL ADMINISTRATION AREA      <NA>      <NA>
## 4 15013.487   0.000        AA CIVIL ADMINISTRATION AREA      <NA>      <NA>
## 5  8649.441   0.000        AA CIVIL ADMINISTRATION AREA      <NA>      <NA>
## 6  3762.466   0.000        AA CIVIL ADMINISTRATION AREA      <NA>      <NA>
```

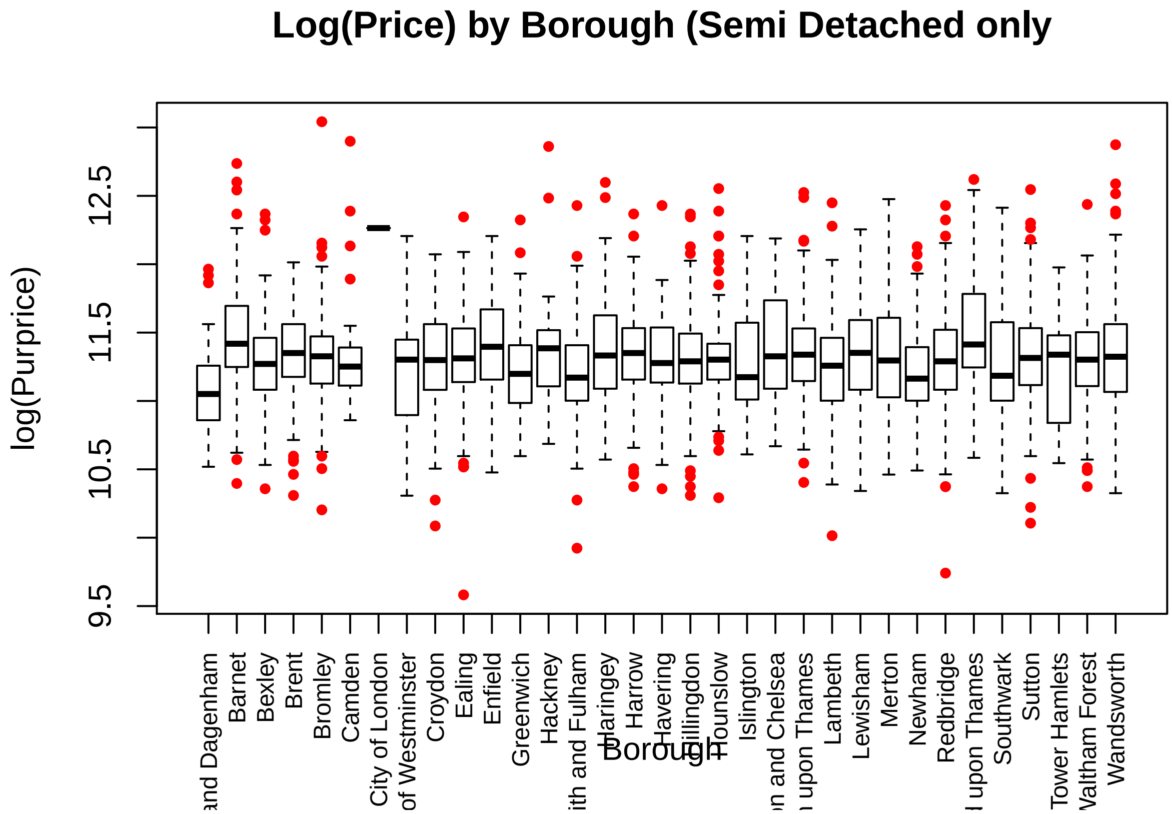
```r
MyData$Borough <- gsub(" London Boro","",LHLB$NAME)   # get the borough name

boxplot(Purprice~Borough,data=MyData)
```



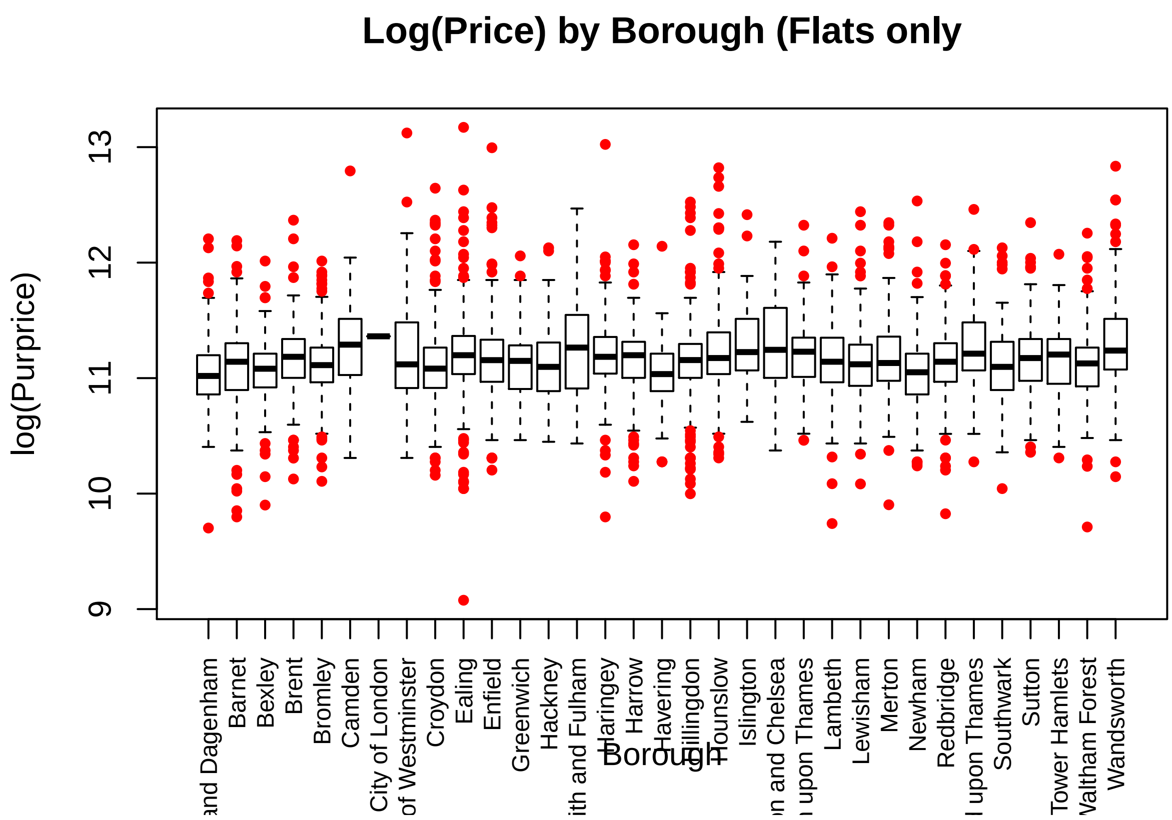

```r
Boroughs <- names(table(MyData$Borough))
NB <- length(Boroughs)
boxplot(log(Purprice)~Borough,data=MyData,outpch=16,outcol="red",outcex=0.75,xaxt="n")
axis(1,labels=Boroughs,at=1:NB,cex.axis=0.75,las=2)
title("Log(Price) by Borough")
```



```
boxplot(log(Purprice)~Borough,data=MyData[MyData$Type=="TypSemiD",],outpch=16,outcol="red",outcex=0.75,
axis(1,labels=Boroughs,at=1:NB,cex.axis=0.75,las=2)
title("Log(Price) by Borough (Semi Detached only")
```

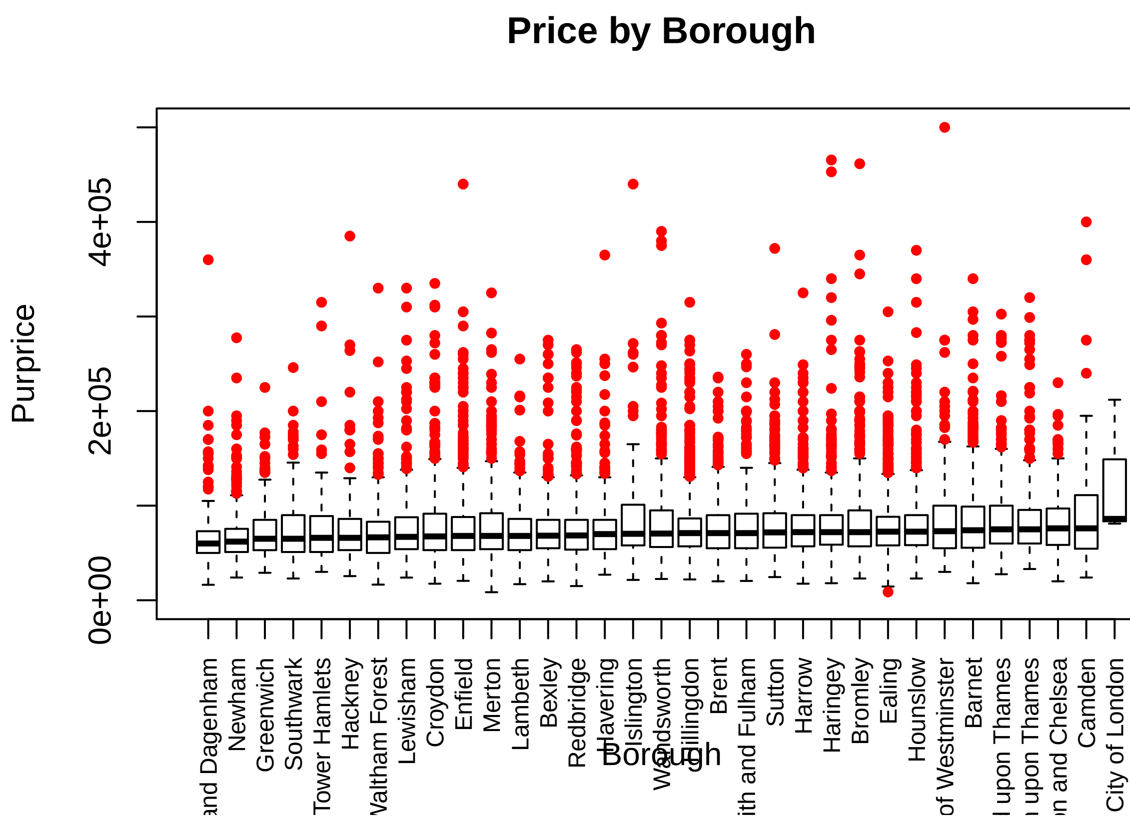



```
boxplot(log(Purprice)~Borough,data=MyData[MyData$Type=="TypFlat",],outpch=16,outcol="red",outcex=0.75,xa
axis(1,labels=Boroughs,at=1:NB,cex.axis=0.75,las=2)
title("Log(Price) by Borough (Flats only")
```

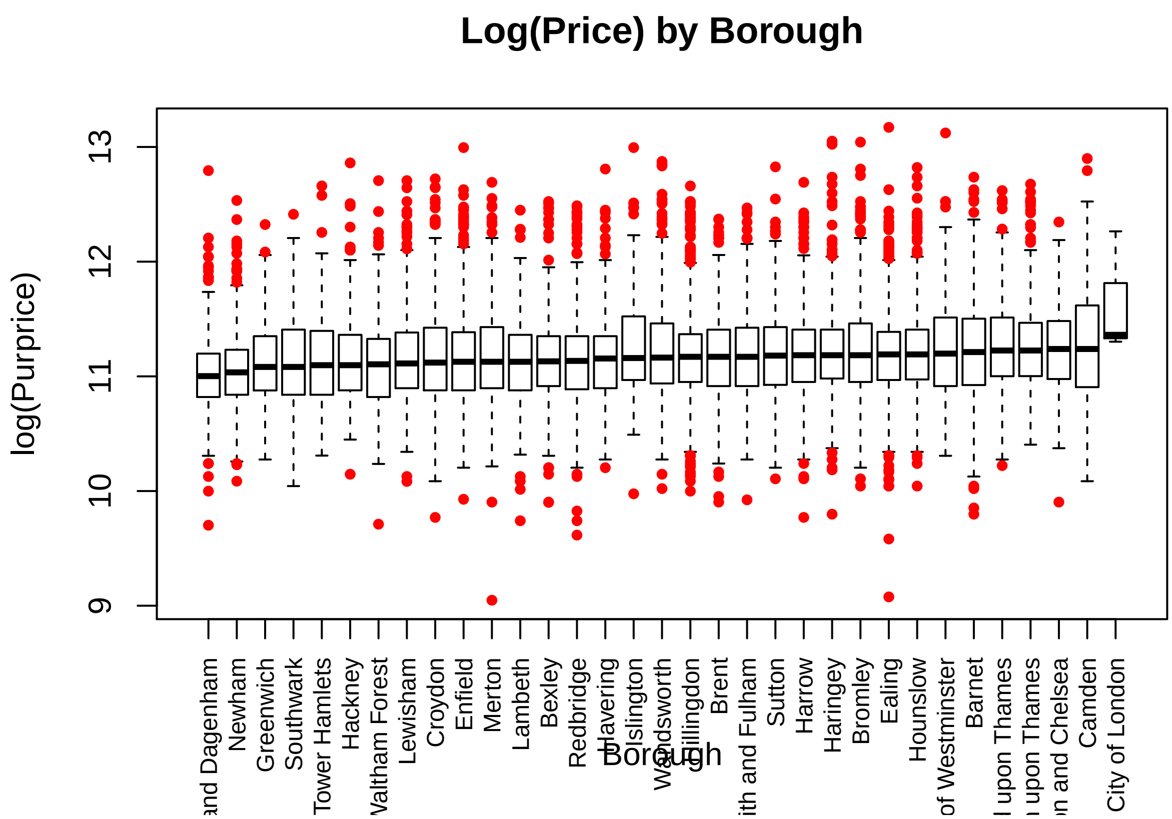


Ordered boxplot

```
b.order <- rank(tapply(MyData$Purprice+runif(nrow(MyData)),MyData$Borough,median))

boxplot(Purprice~Borough,data=MyData,outpch=16,outcol="red",outcex=0.75,xaxt="n",at=b.order,ylim=c(0,50
axis(1,labels=Boroughs,at=b.order,cex.axis=0.75,las=2)
title("Price by Borough")
```

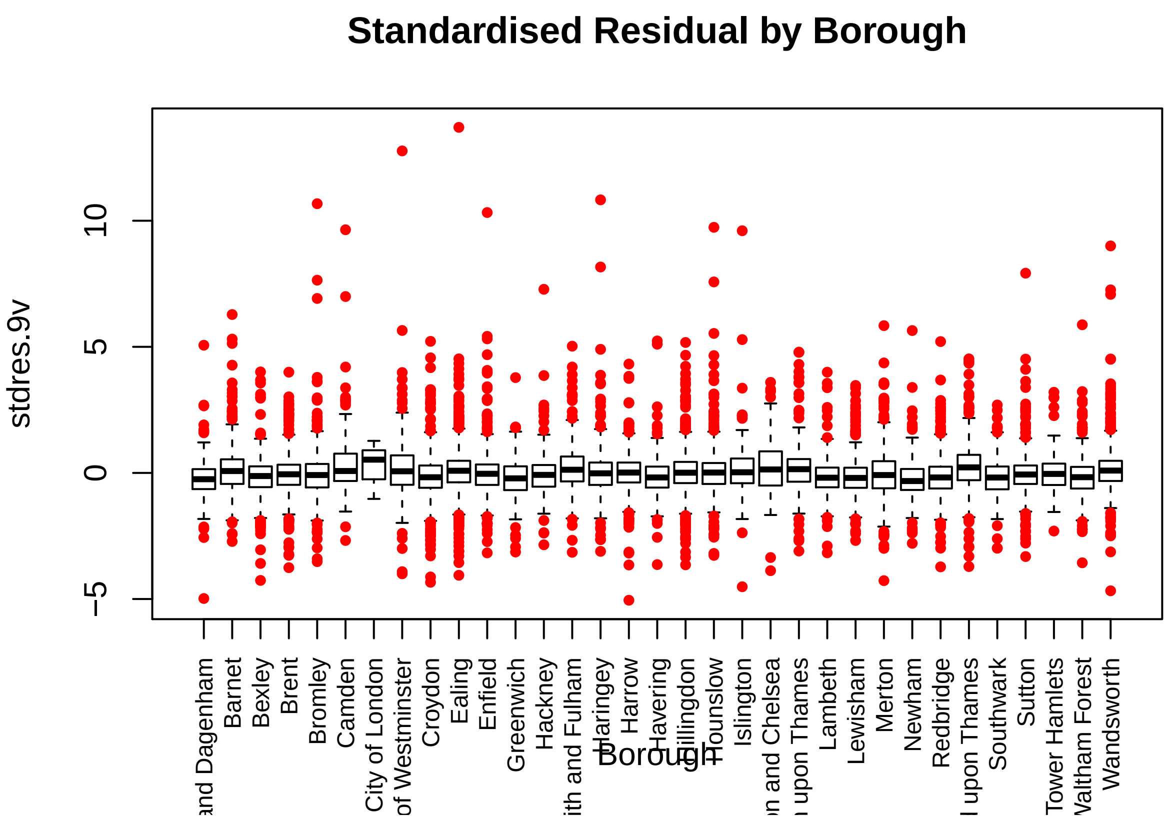



```
boxplot(log(Purprice)~Borough,data=MyData,outpch=16,outcol="red",outcex=0.75,xaxt="n",at=b.order)
axis(1,labels=Boroughs,at=b.order,cex.axis=0.75,las=2)
title("Log(Price) by Borough")
```

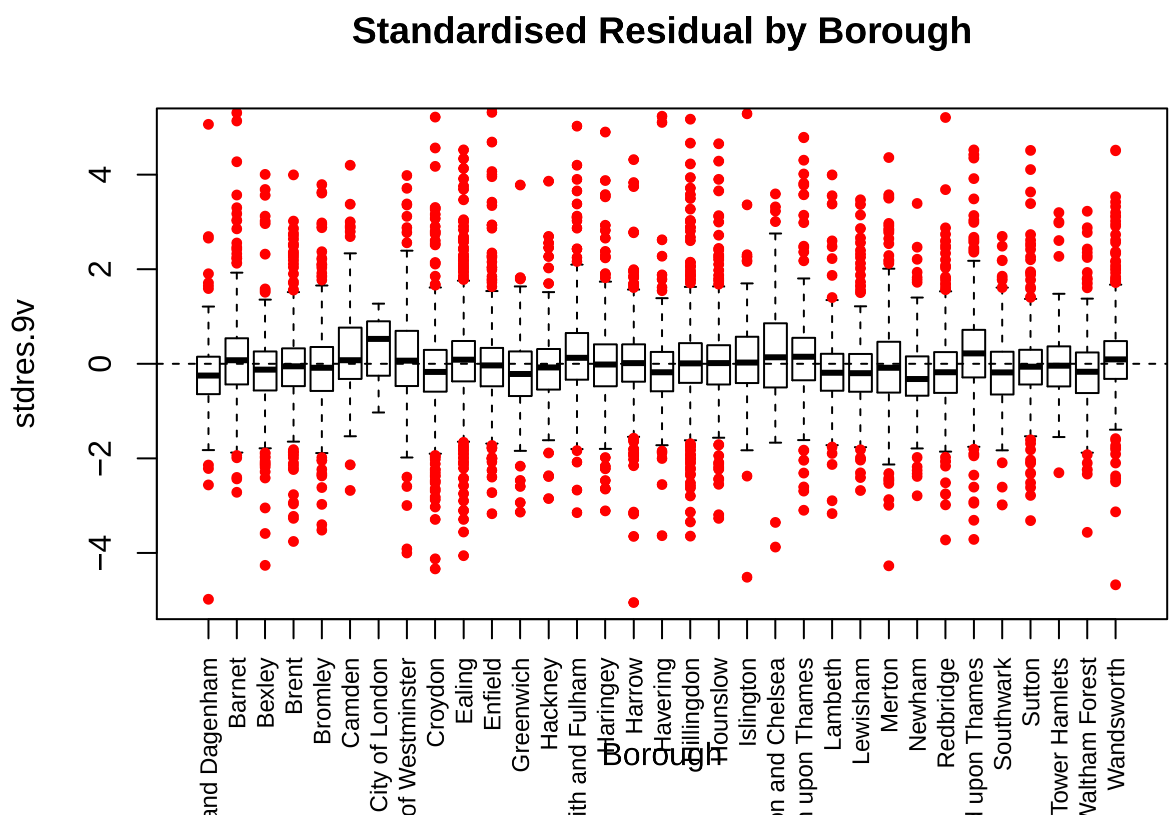


standardsed residuals -s there a apttern

```
MyData$stdres.9v <- stdres(model.9v)
boxplot(stdres.9v~Borough,data=MyData,outpch=16,outcol="red",outcex=0.75,xaxt="n")
axis(1,labels=Boroughs,at=1:NB,cex.axis=0.75,las=2)
title("Standardised Residual by Borough")
```



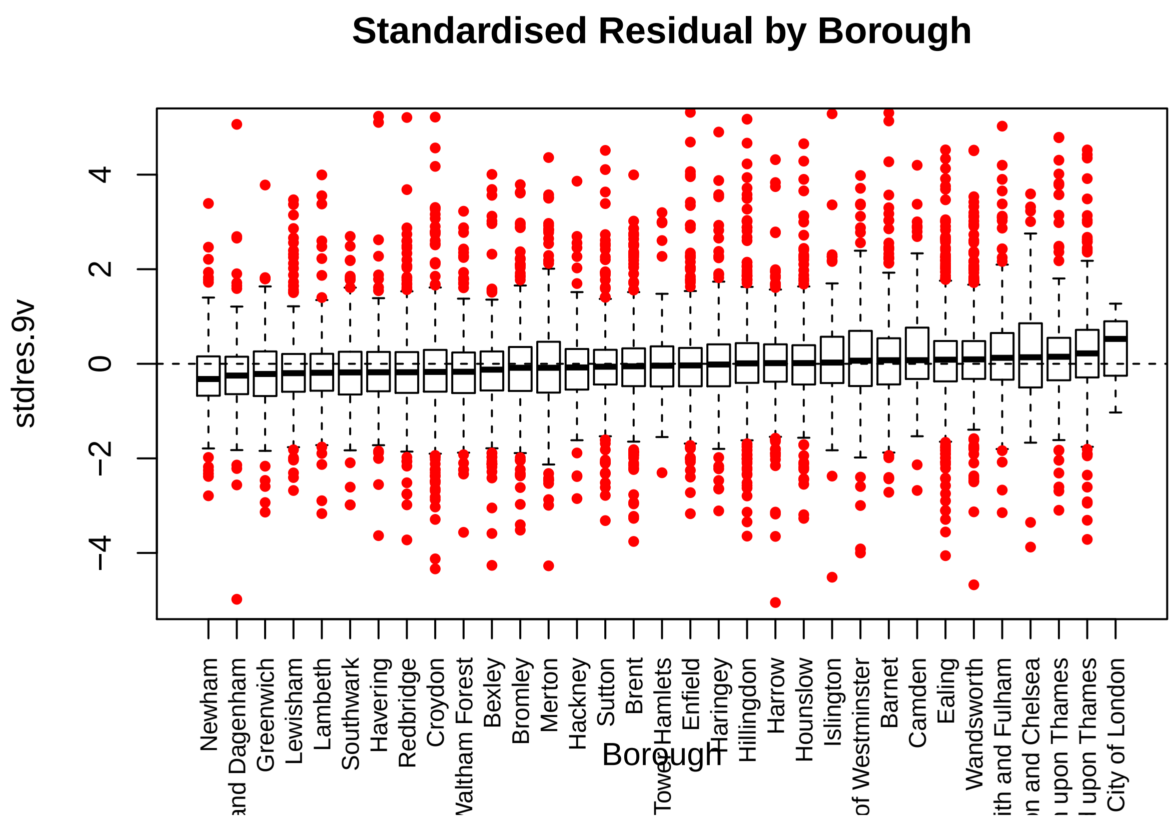

```
boxplot(stdres.9v~Borough,data=MyData,outpch=16,outcol="red",outcex=0.75,xaxt="n",ylim=c(-5,5))
axis(1,labels=Boroughs,at=1:NB,cex.axis=0.75,las=2)
title("Standardised Residual by Borough")
abline(h=0,lty=2)
```



y-yhat negative : overproediction

y-yhat positive : underprediction

```
b.order.9v <- rank(tapply(MyData$stdres.9v+runif(nrow(MyData))*0.0001,MyData$Borough,median))
boxplot(stdres.9v~Borough,data=MyData,outpch=16,outcol="red",outcex=0.75,xaxt="n",at=b.order.9v,ylim=c(
axis(1,labels=Boroughs,at=b.order.9v,cex.axis=0.75,las=2)
title("Standardised Residual by Borough")
abline(h=0,lty=2)
```

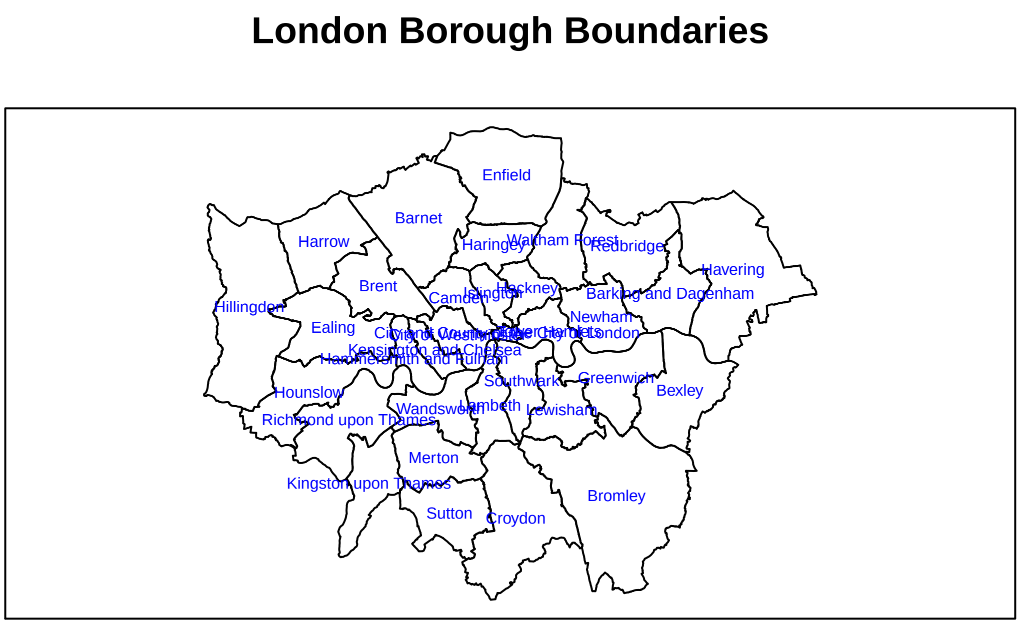


Map of Boroughs with names

```
head(LB$NAME)
```

```
## [1] "Camden London Boro"        "Tower Hamlets London Boro"
## [3] "Islington London Boro"     "Hackney London Boro"
## [5] "Haringey London Boro"      "Newham London Boro"
```

```
Bname <- gsub(" London Boro","",LB$NAME)
xy <- coordinates(LB)
plot(LB)
text(xy[,1],xy[,2],Bname,col="blue",cex=0.5)
box()
title("London Borough Boundaries")
```




```r
quickMap <- function(Var,nClass=10){
   require(classInt)
   require(RColorBrewer)
   Classes <- classIntervals(Var,nClass,method="quantile")
   Palette <- brewer.pal(nClass,"Reds")
   Colours <- findColours(Classes,Palette)
   plot(y)
   points(x.sdf2,cex=0.5,pch=16,col=Colours)
   }
```

How about some borough specific models

```r
data.frame(Bname,LB$NAME)                              # check ordering of names
```

```
##                                       Bname                                 LB.NAME
## 1                                    Camden                      Camden London Boro
## 2                             Tower Hamlets               Tower Hamlets London Boro
## 3                                 Islington                   Islington London Boro
## 4                                   Hackney                     Hackney London Boro
## 5                                  Haringey                    Haringey London Boro
## 6                                    Newham                      Newham London Boro
## 7                      Barking and Dagenham        Barking and Dagenham London Boro
## 8  City and County of the City of London City and County of the City of London
## 9                      Kingston upon Thames        Kingston upon Thames London Boro
## 10                                  Croydon                     Croydon London Boro
## 11                                  Bromley                     Bromley London Boro
## 12                                 Hounslow                    Hounslow London Boro
## 13                                   Ealing                      Ealing London Boro
## 14                                 Havering                    Havering London Boro
## 15                               Hillingdon                  Hillingdon London Boro
## 16                                   Harrow                      Harrow London Boro
## 17                                    Brent                       Brent London Boro
## 18                                   Barnet                      Barnet London Boro
```

```
## 19                          Lambeth                     Lambeth London Boro
## 20                        Southwark                   Southwark London Boro
## 21                         Lewisham                    Lewisham London Boro
## 22                        Greenwich                   Greenwich London Boro
## 23                           Bexley                      Bexley London Boro
## 24                          Enfield                     Enfield London Boro
## 25                   Waltham Forest              Waltham Forest London Boro
## 26                        Redbridge                   Redbridge London Boro
## 27                           Sutton                      Sutton London Boro
## 28             Richmond upon Thames        Richmond upon Thames London Boro
## 29                           Merton                      Merton London Boro
## 30                       Wandsworth                  Wandsworth London Boro
## 31           Hammersmith and Fulham      Hammersmith and Fulham London Boro
## 32           Kensington and Chelsea      Kensington and Chelsea London Boro
## 33            City of Westminster         City of Westminster London Boro
```

```
head(MyData)                                  # and MyData
```

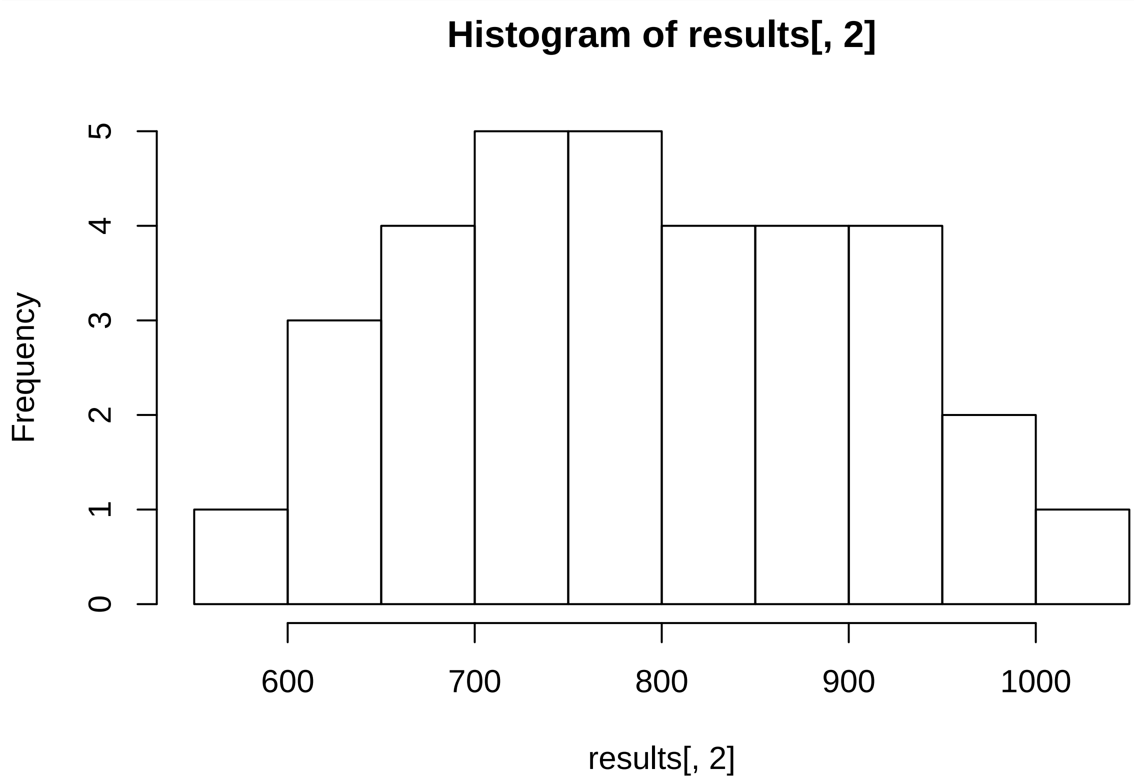
```
##   Easting Northing Purprice Tenfree CenHeat BathTwo NewPropD  FlorArea ProfPct
## 1  545500   173000    85000     yes     yes      no       no  76.16146  0.0000
## 2  525000   177800    71000     yes     yes      no       no  98.45262  6.2500
## 3  531100   183400    60000     yes     yes     yes       no 124.73761  0.0000
## 4  538500   169400    64000     yes     yes      no      yes 127.00000  0.0000
## 5  534000   168400   260000     yes     yes     yes       no 190.40366  9.0909
## 6  528700   168800    48500     yes     yes      no       no  87.00000 16.6667
##      Age     Type   Garage Bedrooms                Borough  stdres.9v
## 1 Bld60s TypDetch GarSingl BedThree                 Bexley  0.5498566
## 2 Bld80s TypDetch GarSingl BedThree Hammersmith and Fulham -0.8386380
## 3 PreWW1 TypSemiD HardStnd  BedFour              Islington -2.3747931
## 4 Bld80s TypDetch GarSingl BedThree                Bromley -1.7998290
## 5 Bld80s TypDetch GarDoubl  BedFour                Croydon  2.5149597
## 6 PreWW1  TypFlat HardStnd BedThree                 Merton -0.5885948
```

```
NB <- length(LB)                              # number of boroughs
results <- matrix(0,NB,2)                     # storage for borough legfel coefficients
for(i in 1:NB) {
   m.x <- lm(Purprice~FlorArea,data=MyData[MyData$Borough == Bname[i],])
   results[i,] <- coef(m.x)
}
rownames(results) <- Bname                    # add in names
colnames(results) <- c("Intercept","FlorArea")
print(results)
```

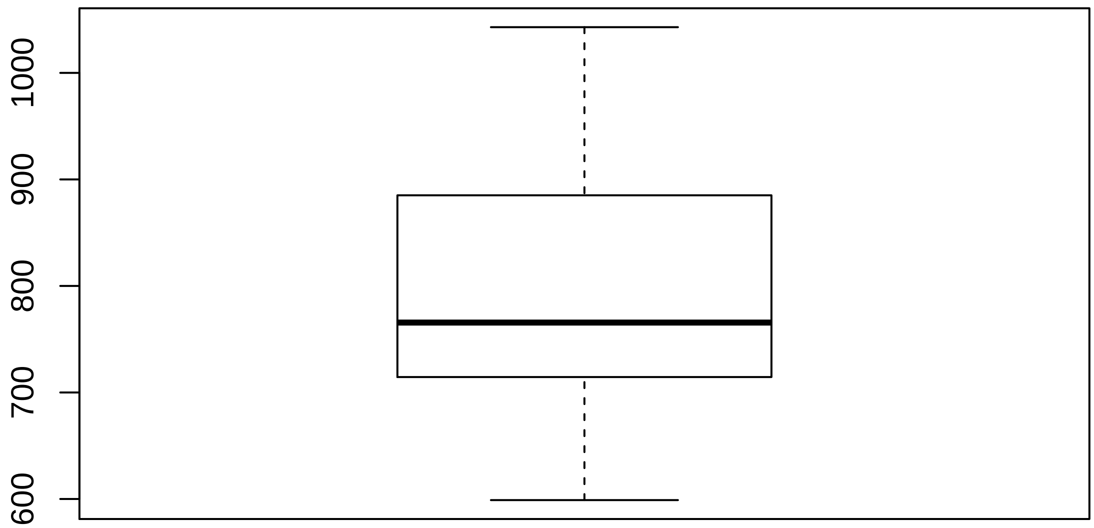
```
##                                     Intercept  FlorArea
## Camden                               4193.591  912.5144
## Tower Hamlets                      -22055.905 1042.9070
## Islington                           -9756.782  976.7416
## Hackney                             -6427.270  888.3199
## Haringey                           -10165.200  941.1180
## Newham                               8392.221  639.8260
## Barking and Dagenham                 2833.098  714.4698
## City and County of the City of London -8934.581  926.4475
## Kingston upon Thames                -7486.608  970.7183
## Croydon                              2511.360  765.5992
## Bromley                             -1299.960  838.6432
```

```
## Hounslow                         2331.035  822.3698
## Ealing                          15196.698  691.0613
## Havering                        -8434.152  875.8949
## Hillingdon                       6441.099  774.5079
## Harrow                           9438.779  737.1701
## Brent                           20595.103  598.9557
## Barnet                          -1450.793  885.0956
## Lambeth                         10948.837  666.9039
## Southwark                       10211.971  689.2389
## Lewisham                        -6768.227  860.6584
## Greenwich                       16655.867  600.0343
## Bexley                           6120.194  729.3274
## Enfield                         -1612.182  844.1284
## Waltham Forest                   9317.256  669.1548
## Redbridge                        1371.890  757.0958
## Sutton                          10038.245  730.6204
## Richmond upon Thames            13743.097  752.8587
## Merton                           7064.287  753.2699
## Wandsworth                      -3590.767  919.3393
## Hammersmith and Fulham          14952.080  736.6411
## Kensington and Chelsea          24302.279  637.5807
## City of Westminster              8260.768  830.7942
```

```
hist(results[,2])                            # look at FlorArea coefficient
```

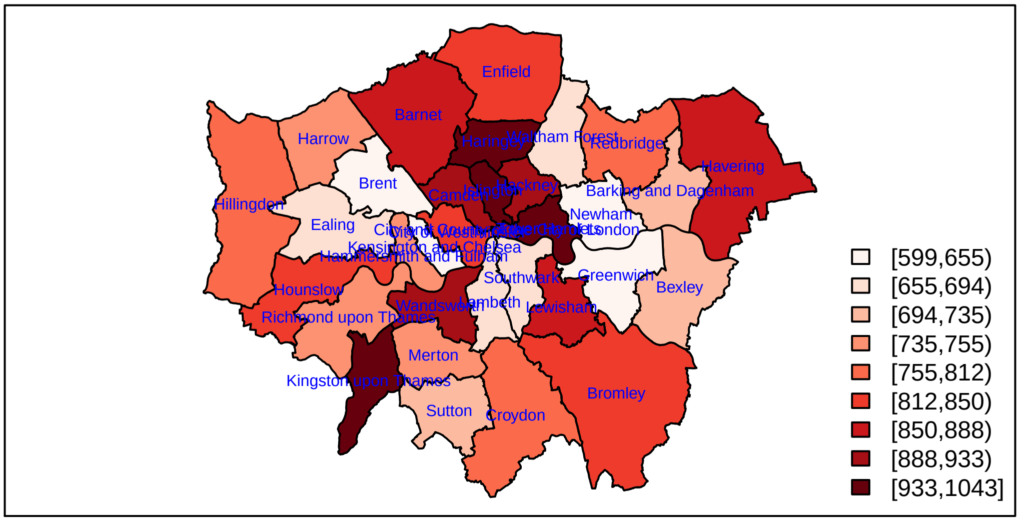



```
boxplot(results[,2])
```



borough levels plots with legend

```
quickMap2 <- function(Var,nClass=9,dp=0,plotNames=FALSE){
   require(classInt)
   require(RColorBrewer)
   Classes <- classIntervals(Var,nClass,method="quantile",dataPrecision=dp)
   Palette <- brewer.pal(nClass,"Reds")
   Colours <- findColours(Classes,Palette)
   plot(LB,col=Colours)
   legend("bottomright",
      legend=names(attr(Colours,"table")),
      fill=attr(Colours,"palette"),
      cex=0.75,bty="n")
   box()
   if(plotNames) {
      xy <- coordinates(LB)
      text(xy[,1],xy[,2],Bname,col="blue",cex=0.5)
   }
}

quickMap2(results[,2])                    # without borough names
quickMap2(results[,2],plotNames=TRUE)     # with borough names
```



and the residuals from the model? Plot the borough medians

```
quickMap2(tapply(MyData$stdres.9v,MyData$Borough,median),plotNames=TRUE,dp=3)
```