Linear Regression Coding Assignment-1

# Load essential libraries  
library(ggplot2)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

# Set ggplot theme for plotting  
My\_Theme = theme(axis.text.x = element\_text(size = 9),  
 axis.text.y = element\_text(size = 9),  
 axis.title.x = element\_text(size = 11),  
 axis.title.y = element\_text(size = 11),  
 plot.title = element\_text(size = 12, hjust = 0.5, face = "bold"))

# Load the house price dataset  
hData = read.csv("Data/houseprices.csv")  
str(hData)

## 'data.frame': 225 obs. of 8 variables:  
## $ locality : chr "BTM Layout" "BTM Layout" "BTM Layout" "BTM Layout" ...  
## $ area : int 565 1837 1280 2220 1113 1332 1815 1400 3006 1600 ...  
## $ rent : int 20060 97434 54448 117000 34388 36394 112000 41266 129000 92849 ...  
## $ price\_per\_sqft: int 6195 9254 7422 9234 5391 4767 10744 5143 7485 10125 ...  
## $ facing : chr "North-West" "East" "East" "North" ...  
## $ BHK : int 1 3 2 3 2 2 3 2 4 3 ...  
## $ bathrooms : int 1 3 2 3 2 2 2 2 5 2 ...  
## $ parking : chr "Bike" "Bike and Car" "Car" "Bike and Car" ...

hData = read.csv('Data/houseprices.csv', header = TRUE, stringsAsFactors = FALSE, na.strings = c("", "NA", "Not Available", "not available"))  
str(hData)

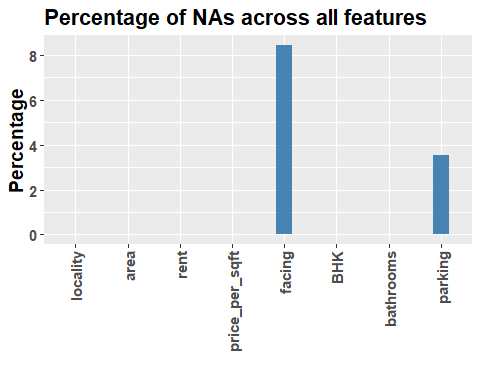
## 'data.frame': 225 obs. of 8 variables:  
## $ locality : chr "BTM Layout" "BTM Layout" "BTM Layout" "BTM Layout" ...  
## $ area : int 565 1837 1280 2220 1113 1332 1815 1400 3006 1600 ...  
## $ rent : int 20060 97434 54448 117000 34388 36394 112000 41266 129000 92849 ...  
## $ price\_per\_sqft: int 6195 9254 7422 9234 5391 4767 10744 5143 7485 10125 ...  
## $ facing : chr "North-West" "East" "East" "North" ...  
## $ BHK : int 1 3 2 3 2 2 3 2 4 3 ...  
## $ bathrooms : int 1 3 2 3 2 2 2 2 5 2 ...  
## $ parking : chr "Bike" "Bike and Car" "Car" "Bike and Car" ...

# Convert 'locality', 'facing' and 'parking' columns to factors  
categorical\_cols = c("locality", "facing", "parking")  
hData[categorical\_cols] = lapply(hData[categorical\_cols], as.factor)  
str(hData)

## 'data.frame': 225 obs. of 8 variables:  
## $ locality : Factor w/ 9 levels "Attibele","BTM Layout",..: 2 2 2 2 2 2 2 2 2 2 ...  
## $ area : int 565 1837 1280 2220 1113 1332 1815 1400 3006 1600 ...  
## $ rent : int 20060 97434 54448 117000 34388 36394 112000 41266 129000 92849 ...  
## $ price\_per\_sqft: int 6195 9254 7422 9234 5391 4767 10744 5143 7485 10125 ...  
## $ facing : Factor w/ 7 levels "East","North",..: 4 1 1 2 1 7 3 6 1 5 ...  
## $ BHK : int 1 3 2 3 2 2 3 2 4 3 ...  
## $ bathrooms : int 1 3 2 3 2 2 2 2 5 2 ...  
## $ parking : Factor w/ 3 levels "Bike","Bike and Car",..: 1 2 3 2 2 2 3 2 2 2 ...

# Continuous columns  
continuous\_cols = setdiff(colnames(hData), categorical\_cols)

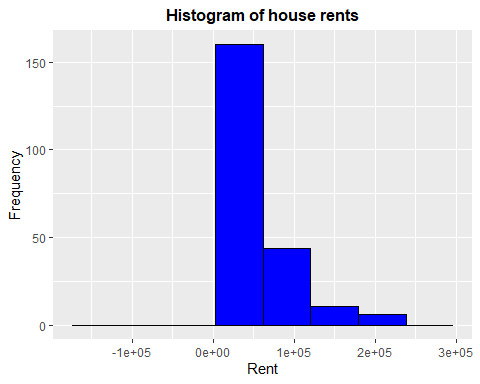
# Plot percentage of NAs in each column of the data frame  
hData\_NA = setNames(stack(sapply(hData, function(x){(sum(is.na(x))/length(x))\*100}))[2:1], c('Feature','Value'))  
p = ggplot(data = hData\_NA, aes(x = Feature, y = Value)) +  
 geom\_bar(stat = 'identity', fill = 'steelblue', width = 0.3) +  
 theme(text = element\_text(size = 14, face = 'bold'),  
 axis.text.x = element\_text(angle = 90, hjust = 1, vjust = 0.5)) +  
 xlab('') + ylab('Percentage') +  
 ggtitle('Percentage of NAs across all features')  
p



categorical\_cols1 = c('facing', 'parking')  
# Add NA as a factor level for categorical columns facing and parking only  
hData[categorical\_cols1] = lapply(hData[categorical\_cols1], addNA)  
str(hData)

## 'data.frame': 225 obs. of 8 variables:  
## $ locality : Factor w/ 9 levels "Attibele","BTM Layout",..: 2 2 2 2 2 2 2 2 2 2 ...  
## $ area : int 565 1837 1280 2220 1113 1332 1815 1400 3006 1600 ...  
## $ rent : int 20060 97434 54448 117000 34388 36394 112000 41266 129000 92849 ...  
## $ price\_per\_sqft: int 6195 9254 7422 9234 5391 4767 10744 5143 7485 10125 ...  
## $ facing : Factor w/ 8 levels "East","North",..: 4 1 1 2 1 7 3 6 1 5 ...  
## $ BHK : int 1 3 2 3 2 2 3 2 4 3 ...  
## $ bathrooms : int 1 3 2 3 2 2 2 2 5 2 ...  
## $ parking : Factor w/ 4 levels "Bike","Bike and Car",..: 1 2 3 2 2 2 3 2 2 2 ...

# Calculate the mean and standard deviation of rent  
mean\_rent <- mean(hData$rent, na.rm = TRUE)  
sd\_rent <- sd(hData$rent, na.rm = TRUE)  
  
# Create histogram  
p = ggplot(data = hData) +  
 geom\_histogram(aes(x = rent, y = after\_stat(count)),   
 breaks = seq(mean\_rent - 4 \* sd\_rent, mean\_rent + 4 \* sd\_rent, by = sd\_rent),   
 color = 'black', fill = 'blue') +  
 labs(x = 'Rent', y = 'Frequency') +  
 ggtitle('Histogram of house rents') +  
 My\_Theme  
  
# Display the plot  
p

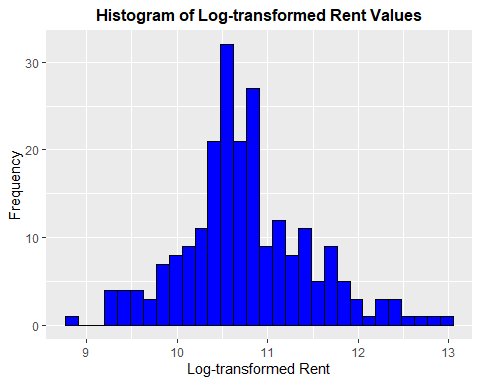


# Build a linear model to predict price per square feet as a function of rent. How accurate is the model?  
model = lm(data=hData, price\_per\_sqft ~ rent)  
summary(model)

##   
## Call:  
## lm(formula = price\_per\_sqft ~ rent, data = hData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -6415.5 -1116.9 -340.6 1193.6 5270.1   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 4.591e+03 1.960e+02 23.42 <2e-16 \*\*\*  
## rent 3.844e-02 2.305e-03 16.68 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2026 on 223 degrees of freedom  
## Multiple R-squared: 0.5551, Adjusted R-squared: 0.5531   
## F-statistic: 278.2 on 1 and 223 DF, p-value: < 2.2e-16

The model is explaining 0.5531 of varinace for this given dataset the model is about 55% accurate

# Make a histogram of log-transformed rent values  
hData['logrent'] = log(hData$rent)  
p = ggplot(data = hData) +  
 geom\_histogram(aes(x = logrent), bins = 30, color = 'black', fill = 'blue') +  
 labs(x = 'Log-transformed Rent', y = 'Frequency') +  
 ggtitle('Histogram of Log-transformed Rent Values') +  
 My\_Theme # You can replace this with your custom theme (e.g., My\_Theme)  
  
# Display the plot  
p



# Build a linear model to predict price per square feet as a function of logrent. Did log-transforming rent help improve the model accuracy?  
model = lm(data=hData, price\_per\_sqft ~ logrent)  
summary(model)

##   
## Call:  
## lm(formula = price\_per\_sqft ~ logrent, data = hData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -7406.1 -966.0 -325.3 968.0 5970.3   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -31058.9 1752.8 -17.72 <2e-16 \*\*\*  
## logrent 3535.5 162.6 21.74 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1720 on 223 degrees of freedom  
## Multiple R-squared: 0.6794, Adjusted R-squared: 0.6779   
## F-statistic: 472.5 on 1 and 223 DF, p-value: < 2.2e-16

The model accuracy has imporved by 12 % the accuracy of the model obtained after using logrent as predictor is about 67%

# Build a linear model to predict log of price per square feet as a function of logrent. Did log-transforming the response variable price per square feet improve the model accuracy?  
hData['logprice\_per\_sqft'] = log(hData$price\_per\_sqft)  
model = lm(logprice\_per\_sqft ~ logrent, data = hData)  
summary(model)

##   
## Call:  
## lm(formula = logprice\_per\_sqft ~ logrent, data = hData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.21981 -0.12244 -0.00241 0.17319 0.56131   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 3.49328 0.24805 14.08 <2e-16 \*\*\*  
## logrent 0.48973 0.02302 21.28 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.2434 on 223 degrees of freedom  
## Multiple R-squared: 0.67, Adjusted R-squared: 0.6685   
## F-statistic: 452.7 on 1 and 223 DF, p-value: < 2.2e-16

The model accuracy has decreased by 1 % the accuracy of the model obtained after using log-transforming the response variable price per square feet as response is about 66%, SO the models accuracy is decreased.

# Build a linear model to predict sqrt of price per square feet as a function of logrent. Did sqrt-transforming the response variable price per square feet improve the model accuracy?  
hData['sqrtprice\_per\_sqft'] = sqrt(hData$price\_per\_sqft)  
model = lm(sqrtprice\_per\_sqft ~ logrent, data = hData)  
summary(model)

##   
## Call:  
## lm(formula = sqrtprice\_per\_sqft ~ logrent, data = hData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -46.536 -5.489 -1.030 6.830 24.025   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -137.769 9.882 -13.94 <2e-16 \*\*\*  
## logrent 20.401 0.917 22.25 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 9.696 on 223 degrees of freedom  
## Multiple R-squared: 0.6894, Adjusted R-squared: 0.688   
## F-statistic: 494.9 on 1 and 223 DF, p-value: < 2.2e-16

The model accuracy has imporved by 1 % the accuracy of the model obtained after using sqrt-transforming the response variable price per square feet as response is about 68%, SO the models accuracy is Increased.

# Build a linear model to predict price per sqft as a function of area and rent. Did adding area as an additional predictor improve model accuracy (compared to only rent as the predictor)? Also, interpret the coefficient estimates for area and rent practically.  
model = lm(price\_per\_sqft ~ area + rent, data = hData)  
summary(model)

##   
## Call:  
## lm(formula = price\_per\_sqft ~ area + rent, data = hData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -7500.7 -751.5 -221.9 849.9 6367.8   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.455e+03 2.164e+02 29.82 <2e-16 \*\*\*  
## area -2.521e+00 2.079e-01 -12.13 <2e-16 \*\*\*  
## rent 6.653e-02 2.928e-03 22.72 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1575 on 222 degrees of freedom  
## Multiple R-squared: 0.7324, Adjusted R-squared: 0.73   
## F-statistic: 303.8 on 2 and 222 DF, p-value: < 2.2e-16

Only rent as the predictor the models accuracy was 55% after using the area as a new predictor the models accuracy is increased to 73%.

the coefficent estimates for area beta1 is that the change in the price\_per\_sqft of a house when the area of the house is increased by 1 unit (1 sqr feet) and by keeping the rent predictor fixed.

the coefficent estimates for area beta2 is that the change in the price per sqft of a house when the rent of the house is increased by 1 unit ($ increase) and by keep the area predictor fixed.

# Build a linear model to predict sqrt of price per sqft as a function of area and logrent. Did adding area as an additional predictor improve model accuracy (compared to only logrent as the predictor)? Also, interpret the coefficient estimates for area and logrent practically.  
model = lm(sqrtprice\_per\_sqft ~ area + logrent, data = hData)  
summary(model)

##   
## Call:  
## lm(formula = sqrtprice\_per\_sqft ~ area + logrent, data = hData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -10.297 -4.238 -1.777 3.361 17.935   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -2.382e+02 8.414e+00 -28.31 <2e-16 \*\*\*  
## area -1.307e-02 7.243e-04 -18.04 <2e-16 \*\*\*  
## logrent 3.147e+01 8.482e-01 37.11 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 6.189 on 222 degrees of freedom  
## Multiple R-squared: 0.874, Adjusted R-squared: 0.8729   
## F-statistic: 770.2 on 2 and 222 DF, p-value: < 2.2e-16

the model accuracy is increased when compared to only logrent as the predictor.

the coefficent estimates for area beta1 is that the change in the sqrt\_price\_per\_sqft of a house when the area of the house is increased by 1 unit (1 sqr feet) and by keeping the logrent predictor fixed.

the coefficent estimates for area beta2 is that the change in the sqrt\_price per\_sqft of a house when the logrent of the house is increased by 1 unit ($ increase) and by keep the area predictor fixed.

# Build a linear model to predict sqrt of price per sqft as a function of logarea and logrent. Did log-transforming area improve model accuracy?  
hData['logarea'] = log(hData$area)  
model = lm(sqrt(price\_per\_sqft) ~ logarea + logrent, data = hData)  
summary(model)

##   
## Call:  
## lm(formula = sqrt(price\_per\_sqft) ~ logarea + logrent, data = hData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.8882 -1.4545 -0.9082 0.7440 19.6434   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -73.5869 2.8088 -26.20 <2e-16 \*\*\*  
## logarea -38.4642 0.6911 -55.66 <2e-16 \*\*\*  
## logrent 40.0275 0.4252 94.13 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.513 on 222 degrees of freedom  
## Multiple R-squared: 0.9792, Adjusted R-squared: 0.979   
## F-statistic: 5233 on 2 and 222 DF, p-value: < 2.2e-16

The model accuracy is improved to 97%.

# Build a linear model to predict price per sqft as a function of area, rent, and parking (compared to just using area and rent as predictors). Did adding parking as an additional predictor improve model accuracy?  
model = lm(price\_per\_sqft ~ area + rent, data = hData)  
summary(model)

##   
## Call:  
## lm(formula = price\_per\_sqft ~ area + rent, data = hData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -7500.7 -751.5 -221.9 849.9 6367.8   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.455e+03 2.164e+02 29.82 <2e-16 \*\*\*  
## area -2.521e+00 2.079e-01 -12.13 <2e-16 \*\*\*  
## rent 6.653e-02 2.928e-03 22.72 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1575 on 222 degrees of freedom  
## Multiple R-squared: 0.7324, Adjusted R-squared: 0.73   
## F-statistic: 303.8 on 2 and 222 DF, p-value: < 2.2e-16

model1 = lm(price\_per\_sqft ~ area + rent + parking, data = hData)  
summary(model1)

##   
## Call:  
## lm(formula = price\_per\_sqft ~ area + rent + parking, data = hData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -7465.5 -752.6 -208.9 842.4 6565.3   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 5.860e+03 5.393e+02 10.866 <2e-16 \*\*\*  
## area -2.453e+00 2.170e-01 -11.301 <2e-16 \*\*\*  
## rent 6.578e-02 3.008e-03 21.867 <2e-16 \*\*\*  
## parkingBike and Car 5.319e+02 4.865e+02 1.093 0.275   
## parkingCar 8.863e+02 5.468e+02 1.621 0.106   
## parkingNA 2.724e+02 7.223e+02 0.377 0.706   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1575 on 219 degrees of freedom  
## Multiple R-squared: 0.736, Adjusted R-squared: 0.73   
## F-statistic: 122.1 on 5 and 219 DF, p-value: < 2.2e-16

The model accuracy is the same even after add the parking as a new predictor.

# Build a linear model to predict sqrt of price per sqft as a function of logarea, logrent, and locality. Did adding locality as an additional predictor improve model accuracy (compared to just using logarea and logrent as predictors)?  
model = lm(sqrt(price\_per\_sqft) ~ logarea + logrent + locality, data = hData)  
summary(model)

##   
## Call:  
## lm(formula = sqrt(price\_per\_sqft) ~ logarea + logrent + locality,   
## data = hData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -4.5577 -1.1073 -0.2527 0.4398 16.6760   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -70.01549 2.95936 -23.659 < 2e-16 \*\*\*  
## logarea -37.69954 0.74724 -50.451 < 2e-16 \*\*\*  
## logrent 39.35270 0.56700 69.405 < 2e-16 \*\*\*  
## localityBTM Layout -2.92678 0.71814 -4.076 6.47e-05 \*\*\*  
## localityElectronic City -2.77473 0.67493 -4.111 5.61e-05 \*\*\*  
## localityIndiranagar -1.17372 0.80139 -1.465 0.14449   
## localityJayanagar 0.02791 0.87628 0.032 0.97462   
## localityK R Puram -3.32188 0.67817 -4.898 1.90e-06 \*\*\*  
## localityMalleshwaram -0.96970 0.83368 -1.163 0.24606   
## localityMarathahalli -3.09626 0.67094 -4.615 6.78e-06 \*\*\*  
## localityYalahanka -1.84366 0.66641 -2.767 0.00616 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.238 on 214 degrees of freedom  
## Multiple R-squared: 0.9841, Adjusted R-squared: 0.9834   
## F-statistic: 1326 on 10 and 214 DF, p-value: < 2.2e-16

The model accuracy is improved to 98%.

# Build a linear model to predict price per sqft as a function of area, rent, and parking. How many levels does the categorical feature parking have? How many new variables are introduced for the categorical variable parking? Interpret all regression coefficient estimates except the intercept coefficient estimate beta0 practically. Do the p-values suggest any insignificant features (that is, features which probably don't have a linear relationship with the response variable?  
levels\_parking = levels(as.factor(hData$parking))  
num\_levels = length(levels\_parking)  
model = lm(price\_per\_sqft ~ area + rent + parking, data = hData)  
summary(model)

##   
## Call:  
## lm(formula = price\_per\_sqft ~ area + rent + parking, data = hData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -7465.5 -752.6 -208.9 842.4 6565.3   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 5.860e+03 5.393e+02 10.866 <2e-16 \*\*\*  
## area -2.453e+00 2.170e-01 -11.301 <2e-16 \*\*\*  
## rent 6.578e-02 3.008e-03 21.867 <2e-16 \*\*\*  
## parkingBike and Car 5.319e+02 4.865e+02 1.093 0.275   
## parkingCar 8.863e+02 5.468e+02 1.621 0.106   
## parkingNA 2.724e+02 7.223e+02 0.377 0.706   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1575 on 219 degrees of freedom  
## Multiple R-squared: 0.736, Adjusted R-squared: 0.73   
## F-statistic: 122.1 on 5 and 219 DF, p-value: < 2.2e-16

parking have 4 levels parkingBike is the reference. parking 3 new variables are introduced.

the coefficent estimates for area beta1 is that the change in the price\_per\_sqft of a house when the area of the house is increased by 1 unit (1 sqr feet) and by keeping the all other predictor fixed.

the coefficent estimates for area beta2 is that the change in the price\_per\_sqft of a house when the rent of the house is increased by 1 unit ($ increased) and by keeping the all other predictor fixed.

the coefficent estimates for area beta3 is that the change in the price\_per\_sqft of a house when the house parking as Bike and Car parking availabity and by keeping the all other predictor fixed.

the coefficent estimates for area beta4 is that the change in the price\_per\_sqft of a house when the house parking as only Car parking availabity and by keeping the all other predictor fixed.

the coefficent estimates for area beta4 is that the change in the price\_per\_sqft of a house when the house as no parking availabity and by keeping the all other predictor fixed.

Yes parking perdictor as less significant to the model.

# Create new columns corresponding to scaled versions of the continuous columns  
continuous\_cols = c("area", "rent", "price\_per\_sqft")  
hData[paste0('scaled\_', continuous\_cols)] = lapply(hData[continuous\_cols], scale)  
str(hData)

## 'data.frame': 225 obs. of 15 variables:  
## $ locality : Factor w/ 9 levels "Attibele","BTM Layout",..: 2 2 2 2 2 2 2 2 2 2 ...  
## $ area : int 565 1837 1280 2220 1113 1332 1815 1400 3006 1600 ...  
## $ rent : int 20060 97434 54448 117000 34388 36394 112000 41266 129000 92849 ...  
## $ price\_per\_sqft : int 6195 9254 7422 9234 5391 4767 10744 5143 7485 10125 ...  
## $ facing : Factor w/ 8 levels "East","North",..: 4 1 1 2 1 7 3 6 1 5 ...  
## $ BHK : int 1 3 2 3 2 2 3 2 4 3 ...  
## $ bathrooms : int 1 3 2 3 2 2 2 2 5 2 ...  
## $ parking : Factor w/ 4 levels "Bike","Bike and Car",..: 1 2 3 2 2 2 3 2 2 2 ...  
## $ logrent : num 9.91 11.49 10.91 11.67 10.45 ...  
## $ logprice\_per\_sqft : num 8.73 9.13 8.91 9.13 8.59 ...  
## $ sqrtprice\_per\_sqft : num 78.7 96.2 86.2 96.1 73.4 ...  
## $ logarea : num 6.34 7.52 7.15 7.71 7.01 ...  
## $ scaled\_area : num [1:225, 1] -1.041 0.496 -0.177 0.959 -0.379 ...  
## ..- attr(\*, "scaled:center")= num 1426  
## ..- attr(\*, "scaled:scale")= num 827  
## $ scaled\_rent : num [1:225, 1] -0.708 0.609 -0.123 0.942 -0.464 ...  
## ..- attr(\*, "scaled:center")= num 61652  
## ..- attr(\*, "scaled:scale")= num 58729  
## $ scaled\_price\_per\_sqft: num [1:225, 1] -0.253 0.757 0.152 0.75 -0.518 ...  
## ..- attr(\*, "scaled:center")= num 6961  
## ..- attr(\*, "scaled:scale")= num 3030

# Build a linear model to predict scaled price per sqft as a function of scaled area and scaled rent. Compare this with the model built using unscaled data: that is, predict price per sqft as a function of area and rent. Does scaling help?  
model = lm(price\_per\_sqft ~ area + rent, data= hData)  
summary(model)

##   
## Call:  
## lm(formula = price\_per\_sqft ~ area + rent, data = hData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -7500.7 -751.5 -221.9 849.9 6367.8   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.455e+03 2.164e+02 29.82 <2e-16 \*\*\*  
## area -2.521e+00 2.079e-01 -12.13 <2e-16 \*\*\*  
## rent 6.653e-02 2.928e-03 22.72 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1575 on 222 degrees of freedom  
## Multiple R-squared: 0.7324, Adjusted R-squared: 0.73   
## F-statistic: 303.8 on 2 and 222 DF, p-value: < 2.2e-16

model\_scaled = lm(scaled\_price\_per\_sqft ~ scaled\_area + scaled\_rent, data = hData)  
summary(model\_scaled)

##   
## Call:  
## lm(formula = scaled\_price\_per\_sqft ~ scaled\_area + scaled\_rent,   
## data = hData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.47520 -0.24798 -0.07323 0.28045 2.10132   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 7.534e-17 3.464e-02 0.00 1   
## scaled\_area -6.882e-01 5.674e-02 -12.13 <2e-16 \*\*\*  
## scaled\_rent 1.289e+00 5.674e-02 22.72 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.5196 on 222 degrees of freedom  
## Multiple R-squared: 0.7324, Adjusted R-squared: 0.73   
## F-statistic: 303.8 on 2 and 222 DF, p-value: < 2.2e-16

the scaling of the predictors haven’t imporved the model’s accuracy.

# Rebuild a linear model to predict sqrt of price per sqft as a function of logarea, logrent, and locality which we will evaluate using a train-test split of the dataset  
model = lm(data = hData, sqrtprice\_per\_sqft ~ logarea + logrent + locality)  
summary(model)

##   
## Call:  
## lm(formula = sqrtprice\_per\_sqft ~ logarea + logrent + locality,   
## data = hData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -4.5577 -1.1073 -0.2527 0.4398 16.6760   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -70.01549 2.95936 -23.659 < 2e-16 \*\*\*  
## logarea -37.69954 0.74724 -50.451 < 2e-16 \*\*\*  
## logrent 39.35270 0.56700 69.405 < 2e-16 \*\*\*  
## localityBTM Layout -2.92678 0.71814 -4.076 6.47e-05 \*\*\*  
## localityElectronic City -2.77473 0.67493 -4.111 5.61e-05 \*\*\*  
## localityIndiranagar -1.17372 0.80139 -1.465 0.14449   
## localityJayanagar 0.02791 0.87628 0.032 0.97462   
## localityK R Puram -3.32188 0.67817 -4.898 1.90e-06 \*\*\*  
## localityMalleshwaram -0.96970 0.83368 -1.163 0.24606   
## localityMarathahalli -3.09626 0.67094 -4.615 6.78e-06 \*\*\*  
## localityYalahanka -1.84366 0.66641 -2.767 0.00616 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.238 on 214 degrees of freedom  
## Multiple R-squared: 0.9841, Adjusted R-squared: 0.9834   
## F-statistic: 1326 on 10 and 214 DF, p-value: < 2.2e-16

# Split data into train (80%) and test (20%) sets and evaluate model performance on train and test sets. Run this cell multiple times for a random splitting of the data into train and test sets and report the model performance on the resulting train and test sets. Is there much variability in the model performance across different test sets? If that is the case, then the model is not generalizing well and is overfitting the train set. Is it the case here?  
ind = sample(nrow(hData), size = floor(0.8 \* nrow(hData)), replace = FALSE)  
hData\_train = hData[ind, ]  
hData\_test = hData[-ind, ]  
  
# Build the model using the training data  
model = lm(sqrtprice\_per\_sqft ~ logarea + logrent + locality, data = hData\_train)  
  
# Calculate RMSE (Root Mean Squared Error) on train data  
train\_error = sqrt(mean((hData\_train$price\_per\_sqft - predict(model, hData\_train))^2))  
  
# Calculate RMSE (Root Mean Squared Error) on test data  
test\_error = sqrt(mean((hData\_test$price\_per\_sqft - predict(model, hData\_test))^2))  
  
# Print RMSE for train and test sets  
print(paste("Train RMSE: ", train\_error))

## [1] "Train RMSE: 7556.40043529722"

print(paste("Test RMSE: ", test\_error))

## [1] "Test RMSE: 7310.13159653239"

there is no much variability in the model performance across different test sets setting the test error +/- 10 % to the train error.

# Set a seed for reproducibility (optional, you can remove or change the seed each time for different splits)  
#set.seed(123)  
  
# Initialize vectors to store RMSE values  
train\_errors <- numeric(10)  
test\_errors <- numeric(10)  
  
# Repeat the model training and error calculation 10 times  
for (i in 1:10) {  
   
 # Generate a new random sample for training data each time  
 ind <- sample(nrow(hData), size = floor(0.8 \* nrow(hData)), replace = FALSE)  
   
 # Split the data into training and test sets  
 hData\_train <- hData[ind, ]  
 hData\_test <- hData[-ind, ]  
   
 # Build the model using the training data  
 model <- lm(sqrtprice\_per\_sqft ~ logarea + logrent + locality, data = hData\_train)  
   
 # Calculate RMSE (Root Mean Squared Error) on train data  
 train\_error <- sqrt(mean((hData\_train$price\_per\_sqft - predict(model, hData\_train))^2))  
   
 # Calculate RMSE (Root Mean Squared Error) on test data  
 test\_error <- sqrt(mean((hData\_test$price\_per\_sqft - predict(model, hData\_test))^2))  
   
 # Store the RMSE results in the vectors  
 train\_errors[i] <- train\_error  
 test\_errors[i] <- test\_error  
}  
  
# Print the RMSE results for each iteration  
for (i in 1:10) {  
 print(paste("Iteration ", i, " - Train RMSE: ", train\_errors[i], " Test RMSE: ", test\_errors[i]))  
}

## [1] "Iteration 1 - Train RMSE: 7467.13577292787 Test RMSE: 7668.30982182556"  
## [1] "Iteration 2 - Train RMSE: 7591.4558820058 Test RMSE: 7162.7415076687"  
## [1] "Iteration 3 - Train RMSE: 7568.83514114604 Test RMSE: 7257.9556613336"  
## [1] "Iteration 4 - Train RMSE: 7508.07007922319 Test RMSE: 7506.75361521479"  
## [1] "Iteration 5 - Train RMSE: 7681.43805985378 Test RMSE: 6767.65409711568"  
## [1] "Iteration 6 - Train RMSE: 7451.72467995349 Test RMSE: 7728.19643212943"  
## [1] "Iteration 7 - Train RMSE: 7329.14596299674 Test RMSE: 8183.98293627179"  
## [1] "Iteration 8 - Train RMSE: 7487.50165305858 Test RMSE: 7587.67384120418"  
## [1] "Iteration 9 - Train RMSE: 7492.90410610574 Test RMSE: 7566.92331448936"  
## [1] "Iteration 10 - Train RMSE: 7365.86345223714 Test RMSE: 8050.91726223204"

# Optionally, you can calculate the average RMSE across all iterations  
avg\_train\_error <- mean(train\_errors)  
avg\_test\_error <- mean(test\_errors)  
  
print(paste("Average Train RMSE: ", avg\_train\_error))

## [1] "Average Train RMSE: 7494.40747895084"

print(paste("Average Test RMSE: ", avg\_test\_error))

## [1] "Average Test RMSE: 7548.11084894851"