Linear Regression Coding Assignment-1

Code

Hide

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

Hide

```
# Load the price per square feet dataset
hData = read.csv('Data/houseprices_cleaned.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" ...
```

Hide

```
# 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)
```

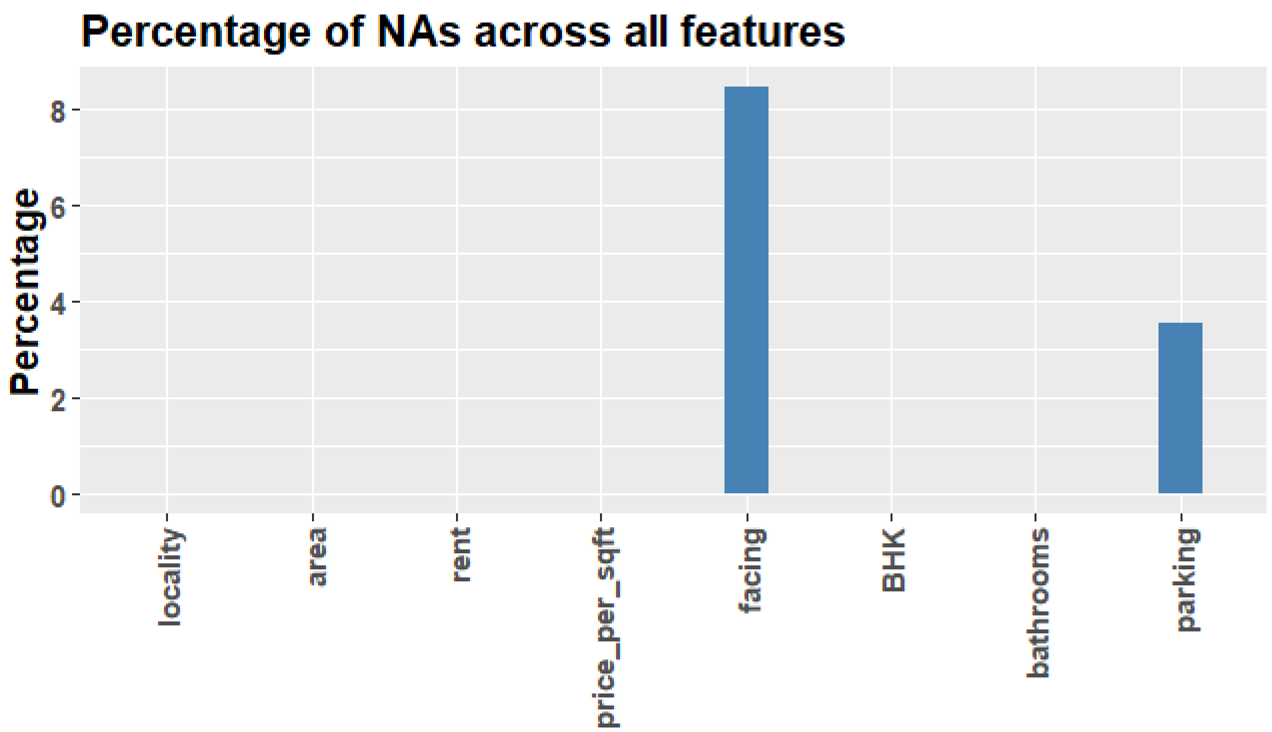
```
'data.frame':	225 obs. of  8 variables:
 $ locality      : Factor w/ 9 levels "Attibele","BTM Layout",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ area          : int  565 1837 1280 2220 1113 1332 1815 1400 3006 1600 ...
 $ rent          : int  20060 97434 54448 117000 34388 36394 112000 41266 129000 92849 ...
 $ 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 ...
```

Hide

```
# Continuous columns
continuous_cols = c('area', 'rent', 'price_per_sqft', 'BHK', 'bathrooms')
```

Hide

```
# Plot percentage of Nans 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
```



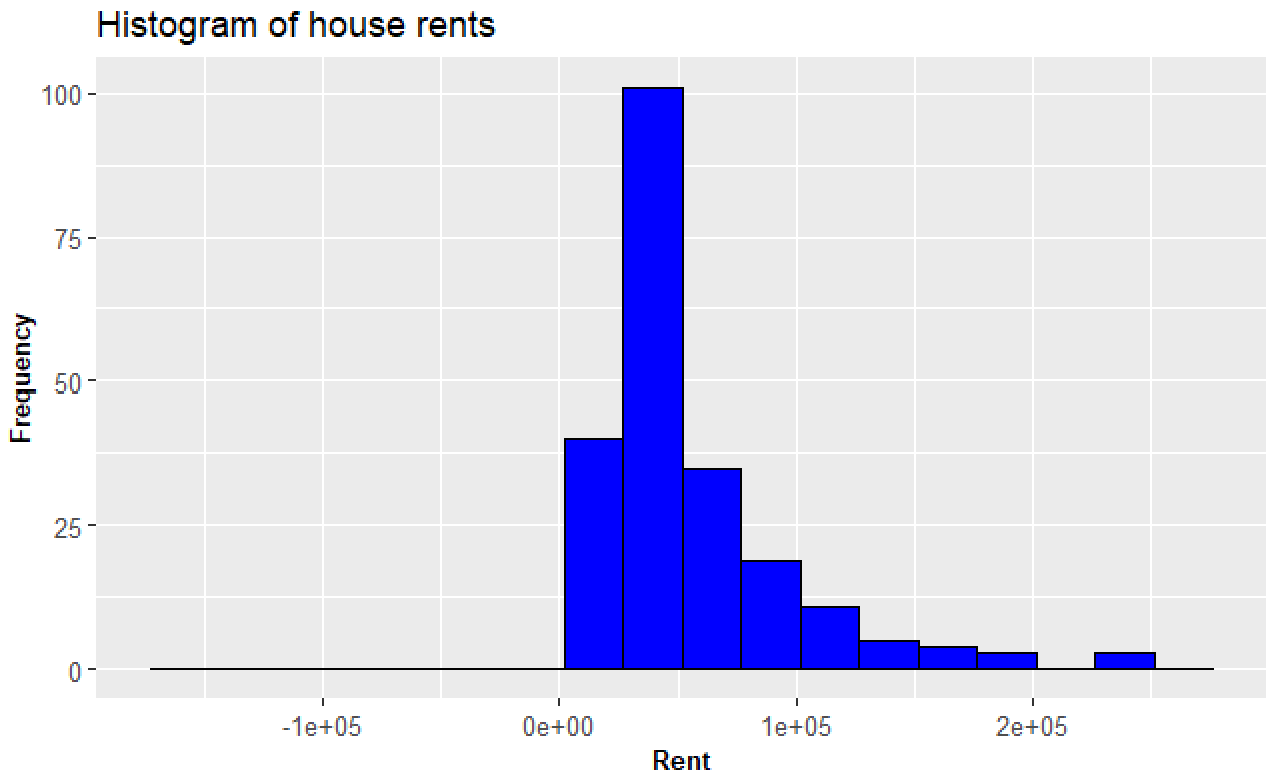
Hide

```
# Add NA as a factor level for categorical columns
hData[categorical_cols] = lapply(hData[categorical_cols], addNA)
str(hData)
```

```
'data.frame':	225 obs. of  8 variables:
 $ locality      : Factor w/ 10 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 ...
```

Hide

```
# Make a histogram of rent values
p = ggplot(data = hData) +
  geom_histogram(aes(x = rent, y = after_stat(count)), breaks = seq(mean(hData$rent)-4*sd(hDa
ta$rent), mean(hData$rent)+4*sd(hData$rent), by = 25000), color = 'black', fill = 'blue') +
  labs(x = 'Rent', y = 'Frequency')  +
   theme(axis.text = element_text(size = 8),
   axis.text.x = element_text(size = 10),
   axis.text.y = element_text(size = 10),
   axis.title = element_text(size = 10, face = "bold"))  +
  ggtitle('Histogram of house rents')
p
```



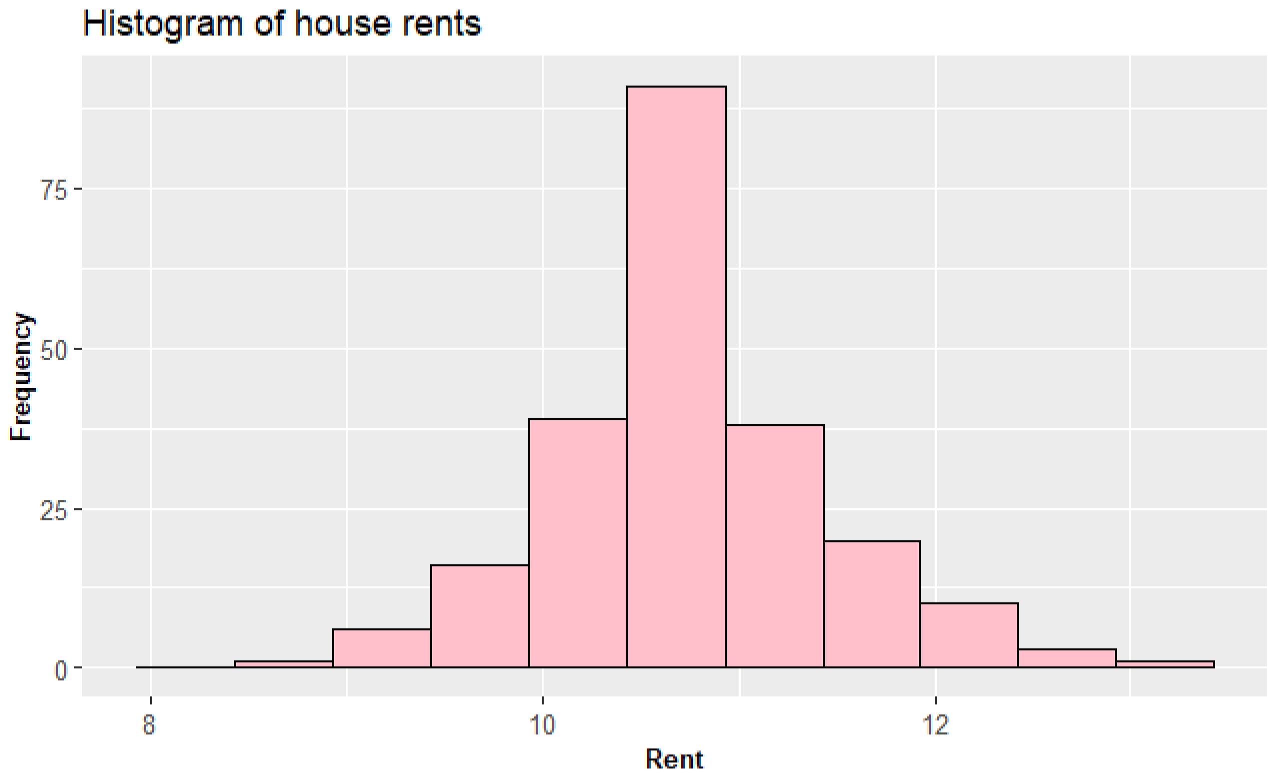
Hide

```
# Build a linear model to predict price per square feet as a function of rent. How accurate i
s 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
```

Hide

```
# Make a histogram of log-transformed rent values
hData['logrent'] = log(hData['rent'])
p = ggplot(data = hData) +
  geom_histogram(aes(x = logrent, y = after_stat(count)), breaks = seq(mean(hData$logrent)-4*sd(hData$logrent), mean(hData$logrent)+4*sd(hData$logrent), by = 0.5), color = 'black', fill = 'pink') +
  labs(x = 'Rent', y = 'Frequency')  +
   theme(axis.text = element_text(size = 8),
   axis.text.x = element_text(size = 10),
   axis.text.y = element_text(size = 10),
   axis.title = element_text(size = 10, face = "bold"))  +
  ggtitle('Histogram of house rents')
p
```



Hide

```
# Build a linear model to predict price per square feet as a function of logrent. Did log-tra
nsforming 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
```

Hide

```
# 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(data = hData,logprice_per_sqft ~ logrent)
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
```

Hide

```
# 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(data = hData,sqrtprice_per_sqft ~ logrent)
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
```

Hide

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

Hide

```
# Build a linear model to predict sqrt of price per sqft as a function of area and logrent. D
id 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(data = hData,sqrtprice_per_sqft ~ area + logrent)
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
```

Hide

```
# Build a linear model to predict sqrt of price per sqft as a function of logarea and logren
t. Did log-transforming area improve model accuracy?
hData['logarea'] = log(hData['area'])
model = lm(data = hData,sqrtprice_per_sqft ~ logarea + logrent )
summary(model)
```

```
Call:
lm(formula = sqrtprice_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
```

Hide

```
# Build a linear model to predict price per sqft as a function of area, rent, and parking (co
mpared to just using area and rent as predictors). Did adding parking as an additional predic
tor improve model accuracy?
model = lm(data = hData,price_per_sqft ~ area + rent + parking )
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
```

Hide

```
# Build a linear model to predict sqrt of price per sqft as a function of logarea, logrent, a
nd locality. Did adding locality as an additional predictor improve model accuracy (compared
to just using logarea and logrent as predictors)?
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
```

Hide

```
# Build a linear model to predict price per sqft as a function of area, rent, and parking. Ho
w many levels does the categorical feature parking have? How many new variables are introduce
d for the categorical variable parking? Interpret all regression coefficient estimates except
the intercept coefficient estimate beta0 practically. Do the p-values suggest any insignifica
nt features (that is, features which probably don't have a linear relationship with the respo
nse variable?
model = lm(data = hData,price_per_sqft ~ area + rent + parking )
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
```

Hide

```
# Create new columns corresponding to scaled versions of the continuous columns
hData[paste0('scaled_', continuous_cols)] = lapply(hData[continuous_cols], scale)
str(hData)
```

```
'data.frame':	225 obs. of  17 variables:
 $ locality            : Factor w/ 10 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
 $ scaled_BHK          : num [1:225, 1] -1.993 0.741 -0.626 0.741 -0.626 ...
  ..- attr(*, "scaled:center")= num 2.46
  ..- attr(*, "scaled:scale")= num 0.731
 $ scaled_bathrooms    : num [1:225, 1] -0.686 0.209 -0.239 0.209 -0.239 ...
  ..- attr(*, "scaled:center")= num 2.53
  ..- attr(*, "scaled:scale")= num 2.23
```

Hide

```
# Build a linear model to predict scaled price per sqft as a function of scaled area and scal
ed rent. Compare this with the model built using unscaled data: that is, predict price per sq
ft as a function of area and rent. Does scaling help?
model_scaled = lm(data = hData, scaled_price_per_sqft ~ scaled_area + scaled_rent)
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
```

Hide

```
# 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, sqrt(price_per_sqft) ~ logarea + logrent + locality)
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
```

Hide

```
# 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 tes
t sets and report the model performance on the resulting train and test sets. Is there much v
ariability 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, ]

# 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(train_error)
```

```
[1] 7408.737
```

Hide

```
print(test_error)
```

```
[1] 7891.224
```