Regression

Jeffrey Li and Valari Graham

9/25/2022

This notebook performs linear regression on the **Bike Sharing** data set retrieved from the University of California, Irvine machine learning repository. A link to the data can be found here (https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset).

Linear Regression Overview

Linear regression is a type of analysis that tries to predict one variable based on another variable. These variables are predictor values and target values, and the goal is to find a relationship between them. This is done by finding a model of the data, also called a line of best fit, which can then predict future values. The biggest strength of linear regression is that it is very simple and easy to implement. However, this also means that the model is prone to underfitting the data, which leads to inaccuracy.

Test and train

First, load in the data and set a seed so the results are reproducible. Then, randomly sample the rows of the data to split into 80/20 train/test groups.

```
bike <- read.csv("hour.csv")
set.seed(123)
sample <- sample(1:nrow(bike), nrow(bike)*0.8, replace=FALSE)
train  <- bike[sample,]
test   <- bike[-sample,]
```

Using R functions to explore the data

Now, we use 5 R functions to explore the training data.

```
# The head function returns the column headers and first 6 rows
head(train)
```

	instant <int>	dteday <chr>	season <int>	yr <int>	m… <int>	hr <int>	holiday <int>	weekday <int>	workingday <int>
2986	2986	2011-05-09	2	0	5	7	0	1	1
1842	1842	2011-03-22	2	0	3	11	0	2	1
3371	3371	2011-05-25	2	0	5	8	0	3	1
11638	11638	2012-05-05	2	1	5	2	0	6	0
4761	4761	2011-07-22	3	0	7	6	0	5	1
6746	6746	2011-10-13	4	0	10	15	0	4	1

6 rows | 1-10 of 18 columns

```
# The summary function gives more information on the model, residuals, coefficients
summary(train)
```

```
##     instant         dteday              season            yr        
##  Min.   :    1   Length:13903       Min.   :1.000   Min.   :0.0000  
##  1st Qu.: 4354   Class :character   1st Qu.:2.000   1st Qu.:0.0000  
##  Median : 8680   Mode  :character   Median :3.000   Median :1.0000  
##  Mean   : 8688                      Mean   :2.503   Mean   :0.5018  
##  3rd Qu.:13032                      3rd Qu.:3.000   3rd Qu.:1.0000  
##  Max.   :17379                      Max.   :4.000   Max.   :1.0000  
##       mnth              hr           holiday            weekday     
##  Min.   : 1.000   Min.   : 0.00   Min.   :0.00000   Min.   :0.000  
##  1st Qu.: 4.000   1st Qu.: 6.00   1st Qu.:0.00000   1st Qu.:1.000  
##  Median : 7.000   Median :12.00   Median :0.00000   Median :3.000  
##  Mean   : 6.545   Mean   :11.54   Mean   :0.02913   Mean   :2.995  
##  3rd Qu.:10.000   3rd Qu.:18.00   3rd Qu.:0.00000   3rd Qu.:5.000  
##  Max.   :12.000   Max.   :23.00   Max.   :1.00000   Max.   :6.000  
##    workingday       weathersit         temp             atemp       
##  Min.   :0.0000   Min.   :1.000   Min.   :0.0200   Min.   :0.0000  
##  1st Qu.:0.0000   1st Qu.:1.000   1st Qu.:0.3400   1st Qu.:0.3333  
##  Median :1.0000   Median :1.000   Median :0.5000   Median :0.4848  
##  Mean   :0.6827   Mean   :1.425   Mean   :0.4978   Mean   :0.4765  
##  3rd Qu.:1.0000   3rd Qu.:2.000   3rd Qu.:0.6600   3rd Qu.:0.6212  
##  Max.   :1.0000   Max.   :4.000   Max.   :1.0000   Max.   :1.0000  
##       hum           windspeed          casual         registered   
##  Min.   :0.0000   Min.   :0.0000   Min.   :  0.00   Min.   :  0.0  
##  1st Qu.:0.4800   1st Qu.:0.1045   1st Qu.:  4.00   1st Qu.: 34.0  
##  Median :0.6300   Median :0.1940   Median : 17.00   Median :116.0  
##  Mean   :0.6275   Mean   :0.1900   Mean   : 35.83   Mean   :154.1  
##  3rd Qu.:0.7800   3rd Qu.:0.2537   3rd Qu.: 48.00   3rd Qu.:222.0  
##  Max.   :1.0000   Max.   :0.8507   Max.   :367.00   Max.   :886.0  
##       cnt       
##  Min.   :  1.0  
##  1st Qu.: 40.0  
##  Median :143.0  
##  Mean   :189.9  
##  3rd Qu.:282.0  
##  Max.   :977.0
```

```
# See if there are NA values in the training data
sum(is.na(train))
```

```
## [1] 0
```

```
# Find the mean of the total rental bike count column
mean(train$cnt)
```

```
## [1] 189.9096
```

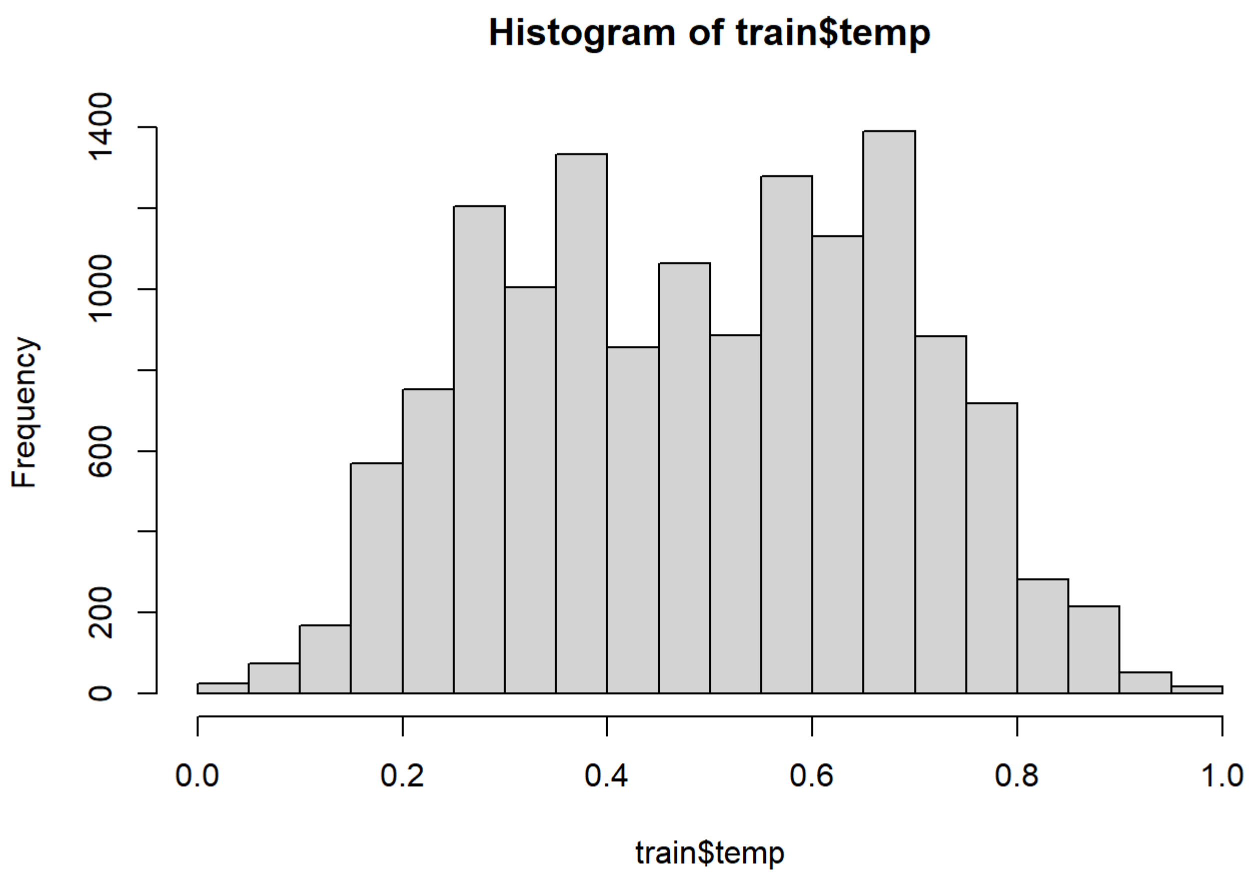
```
# Look at the correlation between columns 3-17
cor(train[3:17])
```

```
##                  season           yr          mnth            hr      holiday
## season      1.000000000 -0.0084311801  0.8297377631 -0.0061891083 -0.008372719
## yr         -0.008431180  1.0000000000 -0.0111239401 -0.0002422363  0.005805022
## mnth        0.829737763 -0.0111239401  1.0000000000 -0.0069441176  0.019950456
## hr         -0.006189108 -0.0002422363 -0.0069441176  1.0000000000  0.003331570
## holiday    -0.008372719  0.0058050217  0.0199504562  0.0033315698  1.000000000
## weekday    -0.011120214 -0.0067355961 -0.0003402332 -0.0047567619 -0.103212994
## workingday  0.012161583 -0.0039337586 -0.0060718465  0.0040083363 -0.254099105
## weathersit -0.014557084 -0.0249239957  0.0040789729 -0.0169495195 -0.024178079
## temp        0.312751294  0.0479223030  0.2023495668  0.1440827044 -0.023170994
## atemp       0.319971258  0.0454623736  0.2085446599  0.1400198126 -0.027052929
## hum         0.153651668 -0.0873548243  0.1687652170 -0.2724281163 -0.014530675
## windspeed  -0.147861704 -0.0048809412 -0.1340066974  0.1380058554  0.006179575
## casual      0.123372230  0.1457728442  0.0697015044  0.3027422260  0.038828125
## registered  0.173061716  0.2522036183  0.1215081664  0.3768766170 -0.043419142
## cnt         0.177984340  0.2501001805  0.1203502736  0.3968891938 -0.025623983
##                  weekday   workingday   weathersit         temp         atemp
## season     -0.0111202138  0.012161583 -0.014557084  0.312751294  0.3199712576
## yr         -0.0067355961 -0.003933759 -0.024923996  0.047922303  0.0454623736
## mnth       -0.0003402332 -0.006071847  0.004078973  0.202349567  0.2085446599
## hr         -0.0047567619  0.004008336 -0.016949519  0.144082704  0.1400198126
## holiday    -0.1032129936 -0.254099105 -0.024178079 -0.023170994 -0.0270529291
## weekday     1.0000000000  0.039307027  0.004893359  0.008365177  0.0001888062
## workingday  0.0393070271  1.000000000  0.048614809  0.059011498  0.0578951642
## weathersit  0.0048933592  0.048614809  1.000000000 -0.101631466 -0.1042193426
## temp        0.0083651774  0.059011498 -0.101631466  1.000000000  0.9872491567
## atemp       0.0001888062  0.057895164 -0.104219343  0.987249157  1.0000000000
## hum        -0.0378275449  0.020360199  0.418664879 -0.072574669 -0.0544043117
## windspeed   0.0118109128 -0.014006412  0.024070878 -0.018993681 -0.0597425817
## casual      0.0317060248 -0.302056077 -0.150707204  0.461427143  0.4554642371
## registered  0.0215929296  0.134272830 -0.118084503  0.340715469  0.3377710049
## cnt         0.0266558700  0.029612930 -0.139586214  0.410005231  0.4059232856
##                    hum    windspeed      casual  registered         cnt
## season      0.15365167 -0.147861704  0.12337223  0.17306172  0.17798434
## yr         -0.08735482 -0.004880941  0.14577284  0.25220362  0.25010018
## mnth        0.16876522 -0.134006697  0.06970150  0.12150817  0.12035027
## hr         -0.27242812  0.138005855  0.30274223  0.37687662  0.39688919
## holiday    -0.01453068  0.006179575  0.03882813 -0.04341914 -0.02562398
## weekday    -0.03782754  0.011810913  0.03170602  0.02159293  0.02665587
## workingday  0.02036020 -0.014006412 -0.30205608  0.13427283  0.02961293
## weathersit  0.41866488  0.024070878 -0.15070720 -0.11808450 -0.13958621
## temp       -0.07257467 -0.018993681  0.46142714  0.34071547  0.41000523
## atemp      -0.05440431 -0.059742582  0.45546424  0.33777100  0.40592329
## hum         1.00000000 -0.294348310 -0.34891320 -0.26907478 -0.31957048
## windspeed  -0.29434831  1.000000000  0.09157501  0.08017104  0.09183900
## casual     -0.34891320  0.091575010  1.00000000  0.50563971  0.69443366
## registered -0.26907478  0.080171038  0.50563971  1.00000000  0.97192702
## cnt        -0.31957048  0.091838995  0.69443366  0.97192702  1.00000000
```

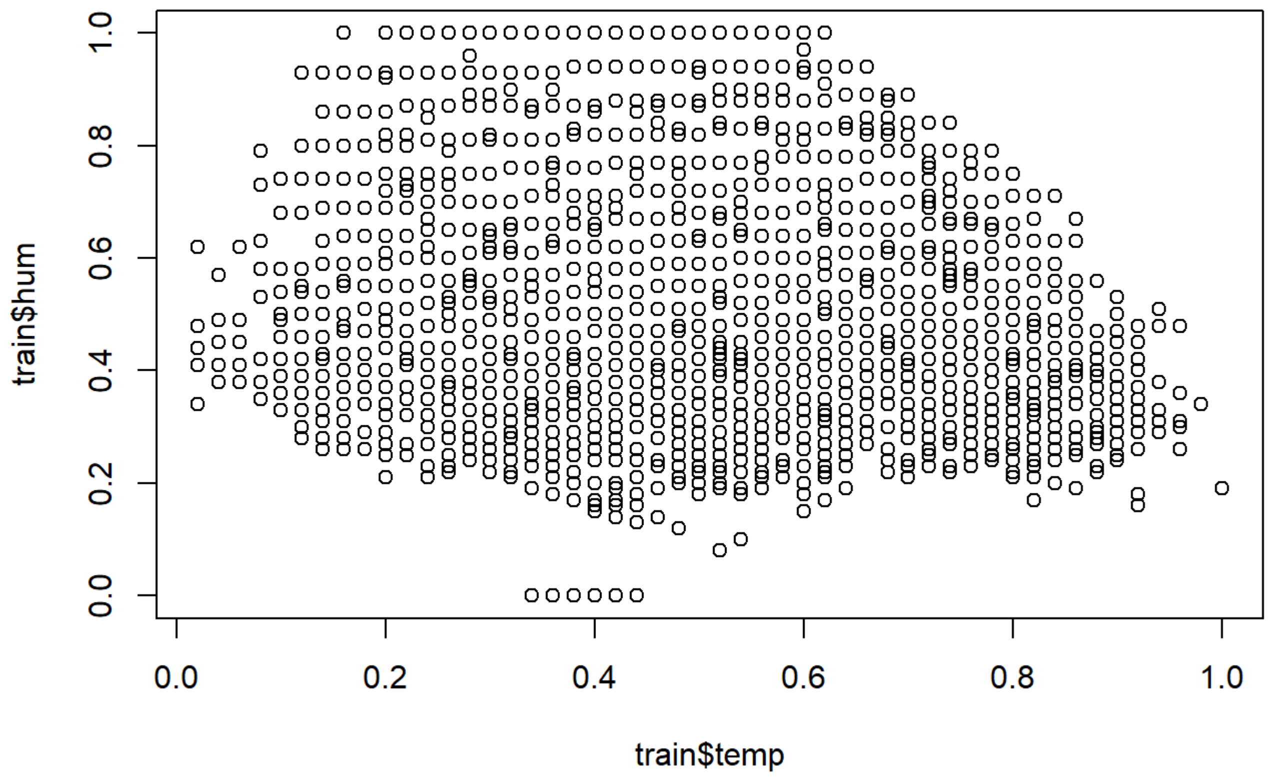
Creating graphs

We create two graphs of the training data. The first graph is a histogram of the temperature and the second graph is a plot of humidity versus temperature.

```
hist(train$temp)
```



```
plot(train$temp, train$hum)
```



Build a simple linear regression model

We build a simple linear regression model with only one predictor. This model is looking at the impact of temperature on the total rental bike count.

```
lm1 <- lm(cnt~temp, data=train)
summary(lm1)
```

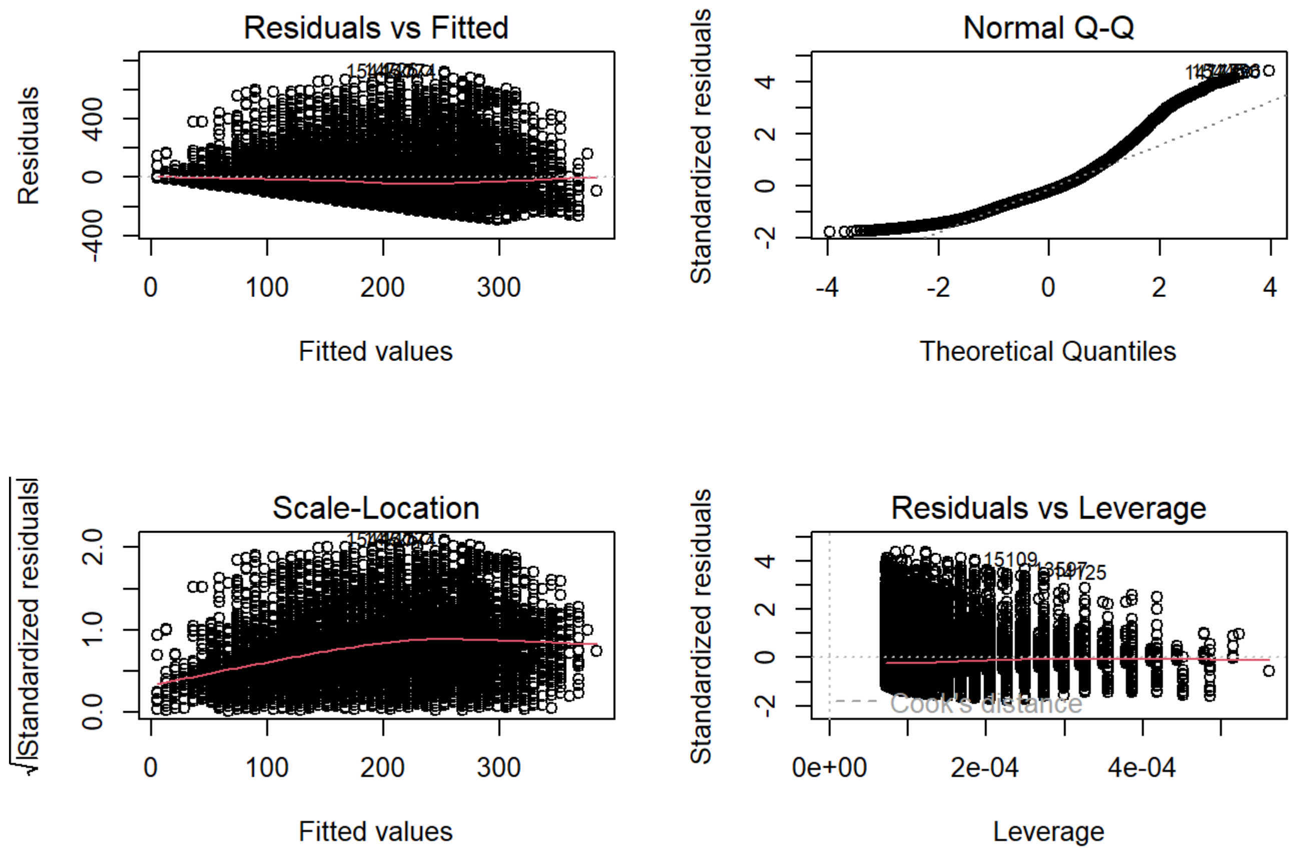
```
##
## Call:
## lm(formula = cnt ~ temp, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -293.01 -110.10  -32.88   76.84  730.85
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -2.543      3.893  -0.653    0.514
## temp         386.602      7.294  53.000   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 165.6 on 13901 degrees of freedom
## Multiple R-squared:  0.1681, Adjusted R-squared:  0.168
## F-statistic:  2809 on 1 and 13901 DF,  p-value: < 2.2e-16
```

From the summary, we can see that the minimum residual was -293.01 and the maximum residual was 730.85. The residuals are the difference between the observed values and predicted values. Next, we see the coefficients of the model, which can give us the following equation predicting rental bike count: count = -2.543 + 386.602temp. From the significance codes, we can tell that temperature is a good predictor from the three asterisks. The residual standard error tells us that the model is off around 166 in units of y. The R-squared statistic is 0.1681, which is bad since we are looking for a value close to 1. Finally, the p-value is very low, indicating a good model since it fits the data better than a model without predictors.

Plot the residuals

Now we plot the residuals.

```
par(mfrow=c(2,2))
plot(lm1)
```



The first plot, Residuals vs Fitted, shows if the residuals have non-linear patterns. In the plot above, the lack of a distinct pattern lets us know that there aren't non-linear relationships. The Normal Q-Q plot shows if the residuals are normally distributed. In this case, they deviate a lot, which is cause for concern. The Scale-Location plot shows if the residuals are spread evenly among the predictor range. A horizontal line is a good sign, but the above plot has an angle near the middle which levels out. The last plot, Residuals vs. Leverage, shows if there are any outliers that are influential to the linear regression. Any values that show up outside of the dashed line, called Cook's distance, will be influential to the results. In this case, there do not seem to be any such values.

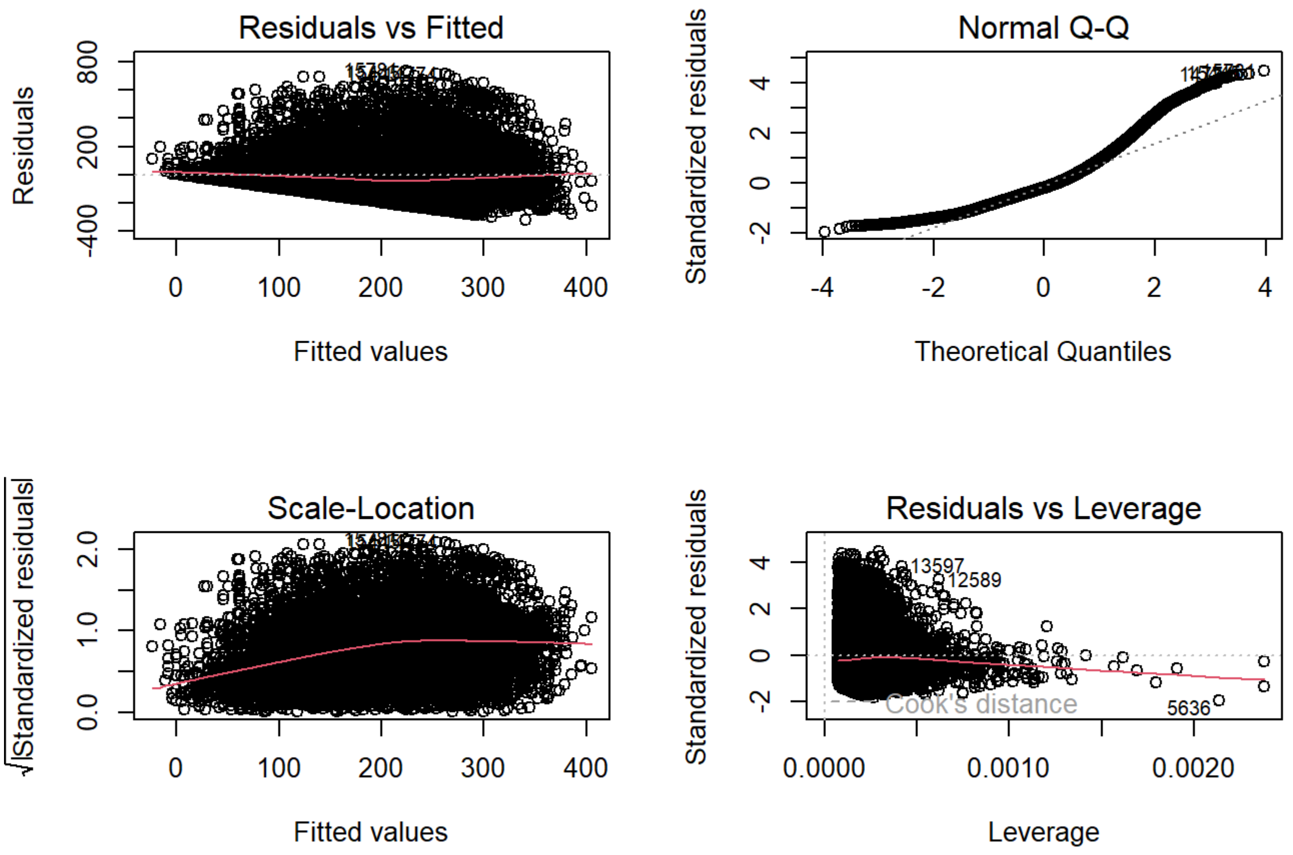
Build a multiple linear regression model

For the next model, we build a linear regression model with multiple predictors, then output the summary and residual plots.

```
lm2 <- lm(cnt~temp+windspeed, data=train)
summary(lm2)
```

```
##
## Call:
## lm(formula = cnt ~ temp + windspeed, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -324.84 -111.03  -33.29   76.40  738.26
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -31.591      4.473  -7.063  1.7e-12 ***
## temp         388.387      7.252  53.554  < 2e-16 ***
## windspeed    148.196     11.437  12.958  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 164.6 on 13900 degrees of freedom
## Multiple R-squared:  0.178,  Adjusted R-squared:  0.1779
## F-statistic:  1505 on 2 and 13900 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,2))
plot(lm2)
```



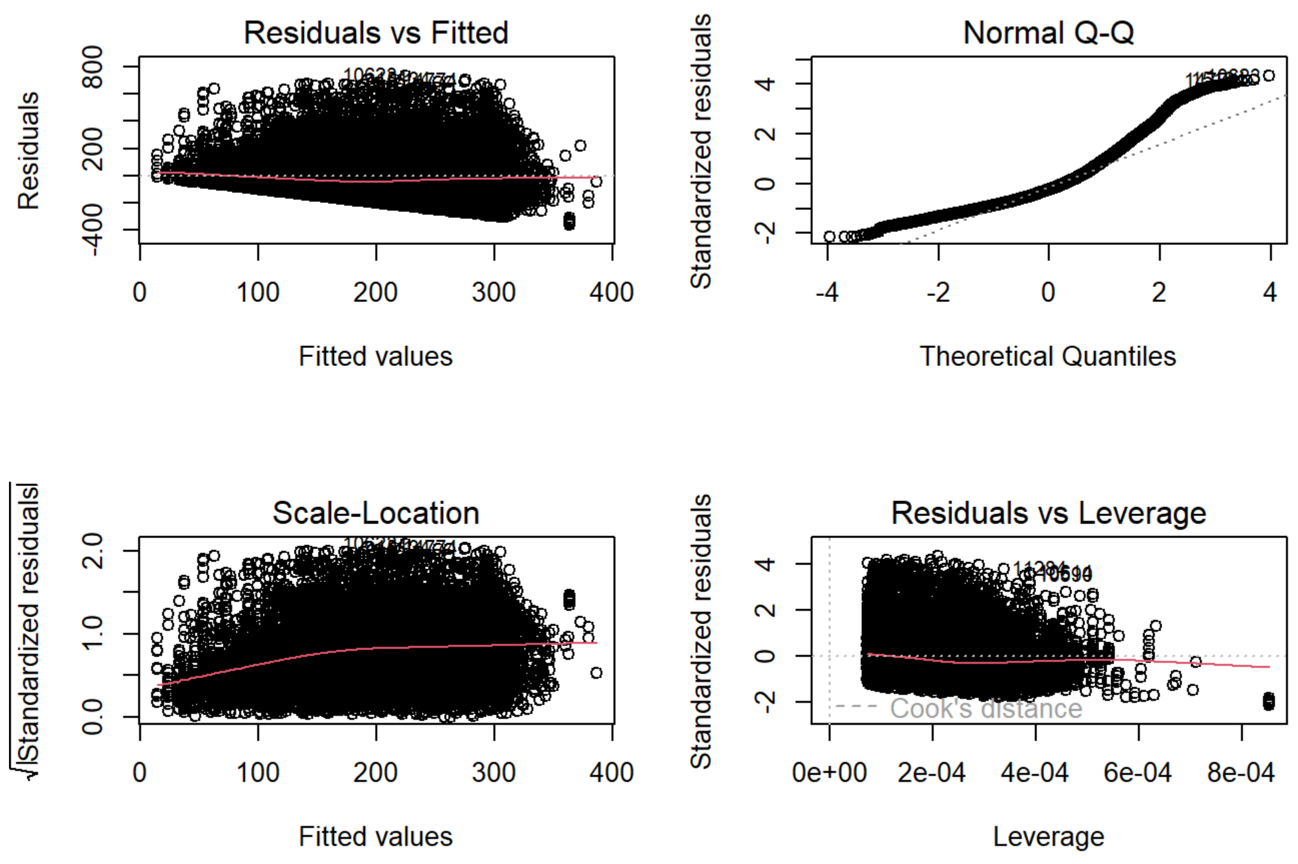
Build a third linear regression model

For our third linear regression model, we will be using the predictors of humidity and month to predict the rental bike count.

```
lm3 <- lm(cnt~hum+mnth, data=train)
summary(lm3)
```

```
##
## Call:
## lm(formula = cnt ~ hum + mnth, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -362.08 -119.31  -40.95   77.10  732.35
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  334.6088     5.2674   63.52   <2e-16 ***
## hum         -329.6110     7.5475  -43.67   <2e-16 ***
## mnth           9.4910     0.4238   22.39   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 169 on 13900 degrees of freedom
## Multiple R-squared:  0.1334, Adjusted R-squared:  0.1333
## F-statistic:  1070 on 2 and 13900 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,2))
plot(lm3)
```



Results

Between the three linear regression models, it seems like the second model, the multiple linear regression model, is the best. Looking at the residual plots for all three models, we see very similar results. The Residuals vs Fitted, Normal Q-Q, and Scale-Location plots are all extremely similar. The Residuals vs Leverage plot has the most differences between all three models, but since no values lie outside of Cook's distance, we can say all the models have the same residual plot results.

Looking at the summary results, we can see that the multiple linear regression model is the best by a slight margin. The p-values for all of the models are equally small and the residual standard error values are also very similar to each other. However, the multiple linear regression model has the largest r-squared value, barely beating the first model and surpassing the third model by a lot. Although the three models are similar, the second one is the best. This is most likely because the model has more significant predictors than the first one, and temperature outweighs month of year and humidity when it comes to biking.

Predicting and evaluating test data

Predict and evaluate on the test data for all three linear regression models.

```
pred1 <- predict(lm1, newdata=test)
cor1 <- cor(pred1, test$cnt)
mse1 <- mean((pred1-test$cnt)^2)
print(paste('correlation:', cor1))
```

```
## [1] "correlation: 0.38362749888682"
```

```
print(paste('mse:', mse1))
```

```
## [1] "mse: 27863.4705892584"
```

```
pred2 <- predict(lm2, newdata=test)
cor2 <- cor(pred2, test$cnt)
mse2 <- mean((pred2-test$cnt)^2)
print(paste('correlation:', cor2))
```

```
## [1] "correlation: 0.399775567050009"
```

```
print(paste('mse:', mse2))
```

```
## [1] "mse: 27440.7552667856"
```

```
pred3 <- predict(lm3, newdata=test)
cor3 <- cor(pred3, test$cnt)
mse3 <- mean((pred3-test$cnt)^2)
print(paste('correlation:', cor3))
```

```
## [1] "correlation: 0.378138044220551"
```

```
print(paste('mse:', mse3))
```

```
## [1] "mse: 27979.2942762334"
```

Looking at the results of the predictions and evaluations of the test data, we can see that the second model is indeed the best. It has the largest correlation value, meaning the variables do have a relationship with each other. Additionally, it has the smallest mean squared error, indicating a better model.