# Similarity Part 1: Regression

Made by: Jonathan Blade

Data set can be found here: https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset (https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset)

The goal of using this data set is to calculate the number of bike rentals during a one hour period.

```
bikeData <- read.csv("C:\\Users\\18327\\Desktop\\Academics\\Academics Fall 2022\\Machine Lear
ning\\Portfolio Similarity\\hour.csv", header=TRUE)

#Treat columns as factors for regression
bikeData$season <- as.factor(bikeData$season)
bikeData$yr <- as.factor(bikeData$yr)
bikeData$weathersit <- as.factor(bikeData$weathersit)
bikeData$weekday <- as.factor(bikeData$weekday)
bikeData$workingday <- as.factor(bikeData$workingday)
bikeData$mnth <- as.factor(bikeData$mnth)
bikeData$hr <- as.factor(bikeData$hr)

#Remove bad columns
bikeData <- bikeData[,-16]
bikeData <- bikeData[,-15]
bikeData <- bikeData[,-2]
bikeData <- bikeData[,-1]

sapply(bikeData, function(x) sum(is.na(x)))
```

```
##     season         yr       mnth         hr    holiday    weekday workingday
##          0          0          0          0          0          0          0
## weathersit       temp      atemp        hum  windspeed        cnt
##          0          0          0          0          0          0
```

```
bikeData <- bikeData[(complete.cases(bikeData)),]
sum(is.na(bikeData))
```

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

```
bikeData <- bikeData[(complete.cases(bikeData)),]
sum(is.na(bikeData))
```

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

```
set.seed(12345)

sample <- sample(c(TRUE,FALSE), nrow(bikeData), replace=TRUE, prob=c(0.8,0.2))
train <- bikeData[sample, ]
test <- bikeData[!sample, ]

summary(train)
```

```
##  season   yr            mnth            hr           holiday        weekday
##  1:3387   0:6929   8      :1214   17     :  605   Min.   :0.00000   0:1968
##  2:3534   1:7020   7      :1205   21     :  605   1st Qu.:0.00000   1:2013
##  3:3642            12     :1203   15     :  599   Median :0.00000   2:1985
##  4:3386            5      :1187   16     :  597   Mean   :0.02839   3:1965
##                    4      :1175   14     :  592   3rd Qu.:0.00000   4:2026
##                    10     :1158   23     :  590   Max.   :1.00000   5:1959
##                    (Other):6807   (Other):10361                     6:2033
##  workingday weathersit      temp            atemp             hum
##  0:4397     1:9158     Min.   :0.0200   Min.   :0.0000   Min.   :0.0000
##  1:9552     2:3641     1st Qu.:0.3400   1st Qu.:0.3333   1st Qu.:0.4800
##             3:1147     Median :0.5000   Median :0.4848   Median :0.6300
##             4:   3     Mean   :0.4982   Mean   :0.4767   Mean   :0.6275
##                        3rd Qu.:0.6600   3rd Qu.:0.6212   3rd Qu.:0.7800
##                        Max.   :0.9800   Max.   :1.0000   Max.   :1.0000
##
##    windspeed           cnt
##  Min.   :0.0000   Min.   :  1.0
##  1st Qu.:0.1045   1st Qu.: 40.0
##  Median :0.1940   Median :143.0
##  Mean   :0.1900   Mean   :189.7
##  3rd Qu.:0.2537   3rd Qu.:281.0
##  Max.   :0.8507   Max.   :977.0
##
```

```
tempCor <- cor(train$temp, train$cnt)
print(paste("Correlation between temp and cnt: ", tempCor))
```

```
## [1] "Correlation between temp and cnt:  0.404355125393318"
```

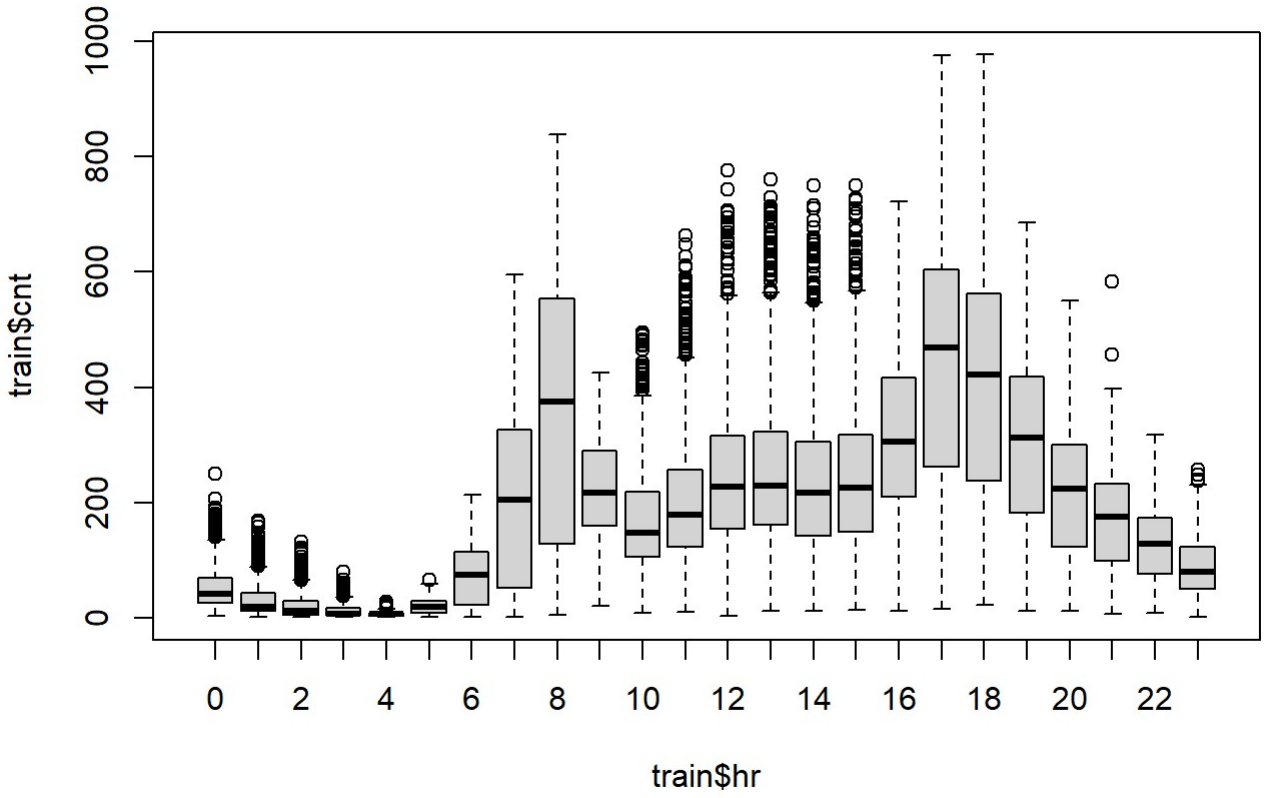
```
atempCor <- cor(train$atemp, train$cnt)
print(paste("Correlation between atemp and cnt: ", atempCor))
```

```
## [1] "Correlation between atemp and cnt:  0.399290108121706"
```

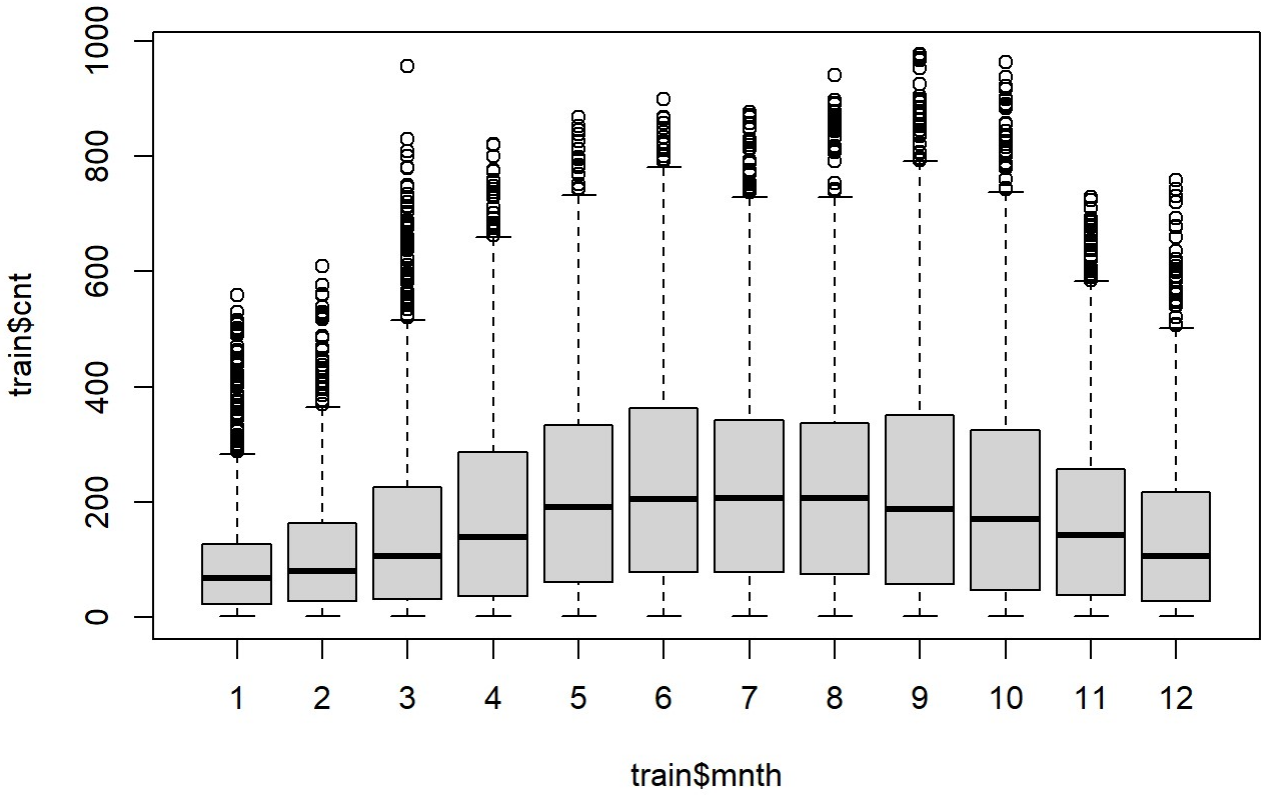
```
humCor <- cor(train$hum, train$cnt)
print(paste("Correlation between humidity and cnt: ", humCor))
```

```
## [1] "Correlation between humidity and cnt:  -0.327113264879201"
```

```
boxplot(train$cnt ~ train$hr)
```

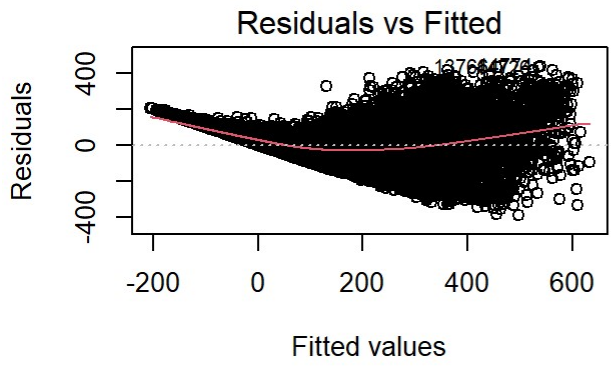


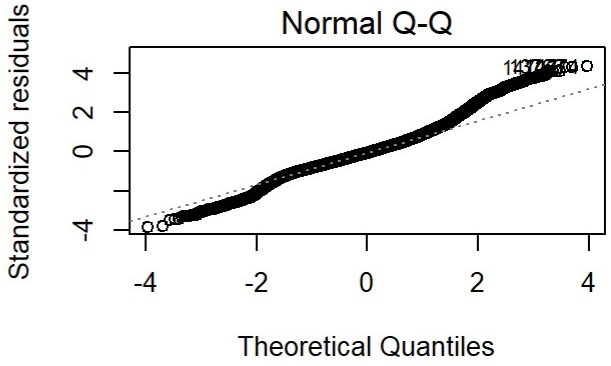

```
boxplot(train$cnt ~ train$mnth)
```

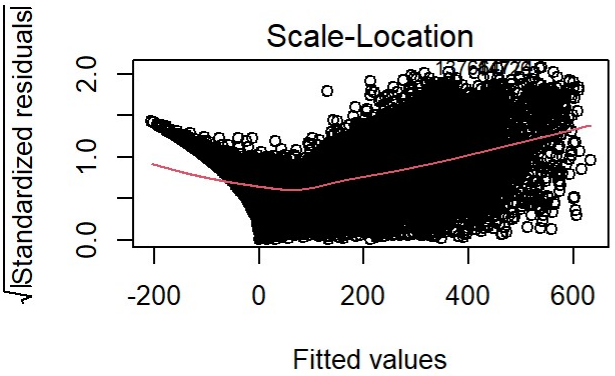


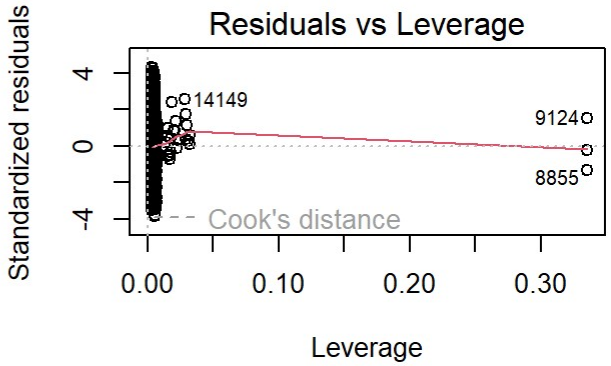

```
lm1 <- lm(cnt ~ ., data = train)

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




```
par(mfrow=c(1,1))
summary(lm1)
```

```
##
## Call:
## lm(formula = cnt ~ ., data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -389.41  -60.44   -7.74   50.53  438.40
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -85.1383     7.4246 -11.467  < 2e-16 ***
## season2      42.3377     5.4441   7.777 7.96e-15 ***
## season3      36.4430     6.4107   5.685 1.34e-08 ***
## season4      68.4190     5.3987  12.673  < 2e-16 ***
## yr1          85.4861     1.7399  49.132  < 2e-16 ***
## mnth2         2.5743     4.3710   0.589 0.555911
## mnth3        12.0865     4.9672   2.433 0.014976 *
## mnth4         2.5956     7.3191   0.355 0.722866
## mnth5        14.9953     7.8531   1.909 0.056220 .
## mnth6        -0.8851     8.0766  -0.110 0.912737
## mnth7       -22.3877     9.0314  -2.479 0.013191 *
## mnth8        -3.1753     8.7918  -0.361 0.717976
## mnth9        24.3307     7.8086   3.116 0.001838 **
## mnth10       12.0586     7.2149   1.671 0.094676 .
## mnth11      -12.0653     6.9301  -1.741 0.081705 .
## mnth12       -7.2357     5.4841  -1.319 0.187057
## hr1         -13.2564     5.9760  -2.218 0.026553 *
## hr2         -25.3666     5.9523  -4.262 2.04e-05 ***
## hr3         -34.2409     6.0111  -5.696 1.25e-08 ***
## hr4         -39.1518     6.0191  -6.505 8.06e-11 ***
## hr5         -22.0011     5.9813  -3.678 0.000236 ***
## hr6          37.6424     5.9766   6.298 3.10e-10 ***
## hr7         171.7399     6.0087  28.582  < 2e-16 ***
## hr8         309.8964     5.9360  52.206  < 2e-16 ***
## hr9         164.4305     5.9474  27.647  < 2e-16 ***
## hr10        110.6261     5.9774  18.508  < 2e-16 ***
## hr11        135.4424     6.0249  22.480  < 2e-16 ***
## hr12        171.0592     6.1083  28.005  < 2e-16 ***
## hr13        172.0642     6.1683  27.895  < 2e-16 ***
## hr14        155.3305     6.1291  25.343  < 2e-16 ***
## hr15        162.0514     6.1277  26.446  < 2e-16 ***
## hr16        225.2757     6.1201  36.809  < 2e-16 ***
## hr17        379.1401     6.0650  62.513  < 2e-16 ***
## hr18        350.5662     6.0883  57.580  < 2e-16 ***
## hr19        240.3978     6.0242  39.905  < 2e-16 ***
## hr20        157.8064     5.9968  26.315  < 2e-16 ***
## hr21        112.1793     5.9072  18.990  < 2e-16 ***
## hr22         71.8345     5.9718  12.029  < 2e-16 ***
## hr23         37.2181     5.9276   6.279 3.51e-10 ***
## holiday     -30.9385     5.4540  -5.673 1.43e-08 ***
## weekday1      9.3008     3.3116   2.809 0.004984 **
```

```
## weekday2      9.9909     3.2427   3.081 0.002067 **
## weekday3     11.9950     3.2516   3.689 0.000226 ***
## weekday4     13.4868     3.2281   4.178 2.96e-05 ***
## weekday5     16.2459     3.2507   4.998 5.88e-07 ***
## weekday6     16.7495     3.2127   5.213 1.88e-07 ***
## workingday1       NA         NA      NA       NA
## weathersit2 -11.5788     2.1418  -5.406 6.55e-08 ***
## weathersit3 -65.9315     3.5988 -18.320  < 2e-16 ***
## weathersit4 -66.3520    58.7596  -1.129 0.258828
## temp        148.5978    31.7889   4.675 2.97e-06 ***
## atemp        97.0027    32.8956   2.949 0.003195 **
## hum         -79.9487     6.1788 -12.939  < 2e-16 ***
## windspeed   -32.4710     7.8342  -4.145 3.42e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 101.4 on 13896 degrees of freedom
## Multiple R-squared:  0.6876, Adjusted R-squared:  0.6864
## F-statistic: 588.2 on 52 and 13896 DF,  p-value: < 2.2e-16
```

```
pred1 <- predict(lm1, newdata=test)
```

```
## Warning in predict.lm(lm1, newdata = test): prediction from a rank-deficient fit
## may be misleading
```

```
correlation1 <- cor(pred1, test$cnt)
print("Model 1: ")
```

```
## [1] "Model 1: "
```

```
print(paste("Correlation: ", correlation1))
```

```
## [1] "Correlation:  0.824804758347274"
```

```
mse1 <- mean((pred1 - test$cnt)^2)
print(paste("MSE: ", mse1))
```

```
## [1] "MSE:  10631.5009722608"
```

```
rmse1 <- sqrt(mse1)
print(paste("RMSE: ", rmse1))
```

```
## [1] "RMSE:  103.109170165707"
```

The above linear regression model has reasonable correlation and accuracy. This will provide a baseline to

compare the results of our next two algorithms.

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
#Convert factors back to numerics for kNN
train$yr <- as.integer(train$yr)
train$weathersit <- as.integer(train$weathersit)
train$weekday <- as.integer(train$weekday)
train$workingday <- as.integer(train$workingday)
train$mnth <- as.integer(train$mnth)
train$hr <- as.integer(train$hr)

test$yr <- as.integer(test$yr)
test$weathersit <- as.integer(test$weathersit)
test$weekday <- as.integer(test$weekday)
test$workingday <- as.integer(test$workingday)
test$mnth <- as.integer(test$mnth)
test$hr <- as.integer(test$hr)

fit1 <- knnreg(train[,1:12], train[,13], k=8)
summary(fit1)
```

```
##         Length Class  Mode
## learn   2      -none- list
## k       1      -none- numeric
## theDots 0      -none- list
```

```
pred2 <- predict(fit1, test[,1:12])
cor_knn1 <- cor(pred2, test$cnt)
mse_knn1 <- mean((pred2 - test$cnt)^2)
rmse_knn1 <- sqrt(mse_knn1)
print(paste("Cor = ", cor_knn1))
```

```
## [1] "Cor =  0.953965954598516"
```

```
print(paste("MSE = ", mse_knn1))
```

```
## [1] "MSE =  3082.61122510843"
```

```
print(paste("RMSE = ", rmse_knn1))
```

```
## [1] "RMSE =  55.5212682231632"
```

The kNN algorithm provides both a higher correlation and accuracy than the linear regression model.

```
library(tree)

tree_bike <- tree(cnt ~ ., data=train)

summary(tree_bike)
```

```
##
## Regression tree:
## tree(formula = cnt ~ ., data = train)
## Variables actually used in tree construction:
## [1] "hr"         "temp"       "yr"         "season"     "workingday"
## Number of terminal nodes:  14
## Residual mean deviance:  11100 = 154700000 / 13940
## Distribution of residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -632.20  -57.40  -18.09    0.00   43.91  526.60
```

```
pred3 <- predict(tree_bike, newdata=test)
cor_tree <- cor(pred3, test$cnt)
print(paste("Cor: ", cor_tree))
```

```
## [1] "Cor:  0.806925391860312"
```

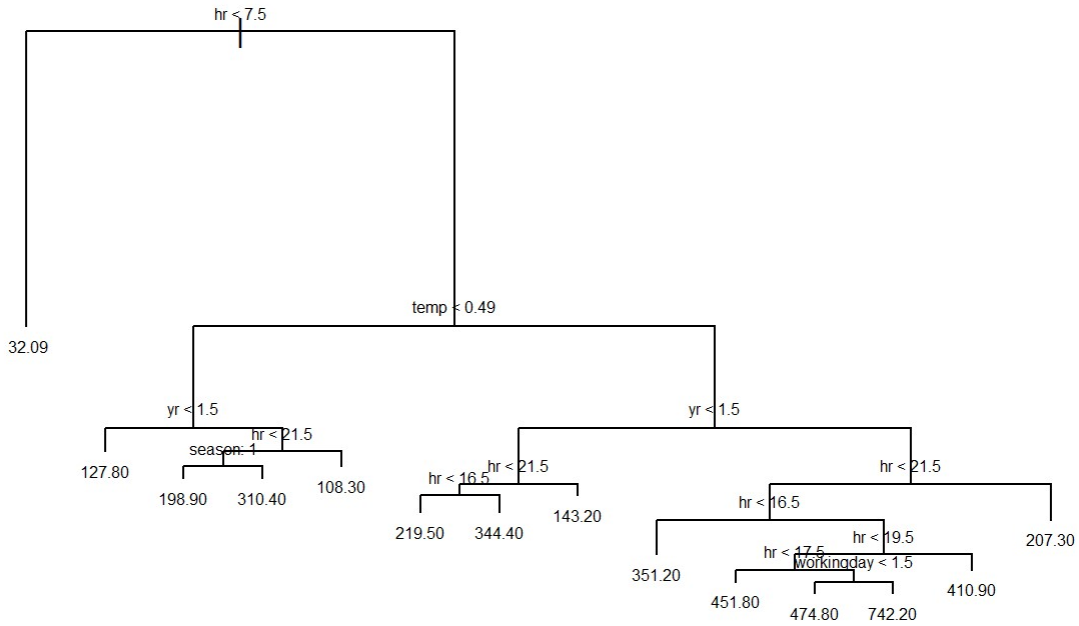
```
mse_tree <- mean((pred3 - test$cnt)^2)
rmse_tree <- sqrt(mse_tree)
print(paste("MSE: ", mse_tree))
```

```
## [1] "MSE:  11604.9279490952"
```

```
print(paste("RMSE: ", rmse_tree))
```

```
## [1] "RMSE:  107.726171142834"
```

```
plot(tree_bike)
text(tree_bike, cex=0.5, pretty=0)
```




```
cv_tree <- cv.tree(tree_bike)
```

The decision tree is less accurate than both the Linear Regression model and the kNN algorithm. But, the decision tree's greatest strength is how easy it is to interpret.