Problem Set 7

a.

```
DayDF <-read.csv("/Users/karlhickel/Desktop/Bike-Sharing-Dataset/day.csv")
HourDF <-read.csv("/Users/karlhickel/Desktop/Bike-Sharing-Dataset/hour.csv")
helpMe <- read.delim("/Users/karlhickel/Desktop/Bike-Sharing-Dataset/Readme.txt")
```

b.

d.

We cannout utilize strings to perform an analysis because they are the the wrong data type. Having the incorrect data type will not allow us to measure the values. #e.

Instant = Integer dteday = string season = integer yr = integer month = integer holiday = integer weekday = integer workingday = integer weathersit = integer temporary = double weekday = int working day = int weathersit = int temp = double atemp = double hum = double windspeed = double casual = int registered = int cnt = int

f.

```
sapply(DayDF, class)
```

```
##    instant     dteday     season         yr       mnth    holiday
##  "integer"   "factor"  "integer"  "integer"  "integer"  "integer"
##    weekday workingday weathersit       temp      atemp        hum
##  "integer"  "integer"  "integer"  "numeric"  "numeric"  "numeric"
##  windspeed     casual registered        cnt
##  "numeric"  "integer"  "integer"  "integer"
```

g.

```
DayDF$weekday <- as.factor(DayDF$weekday)
DayDF$workingday <- as.factor(DayDF$workingday)
DayDF$weathersit <- as.factor(DayDF$weathersit)
DayDF$season <- as.factor(DayDF$season)
DayDF$yr <- as.factor(DayDF$yr)
DayDF$mnth <- as.factor(DayDF$mnth)
DayDF$holiday <- as.factor(DayDF$holiday)
```

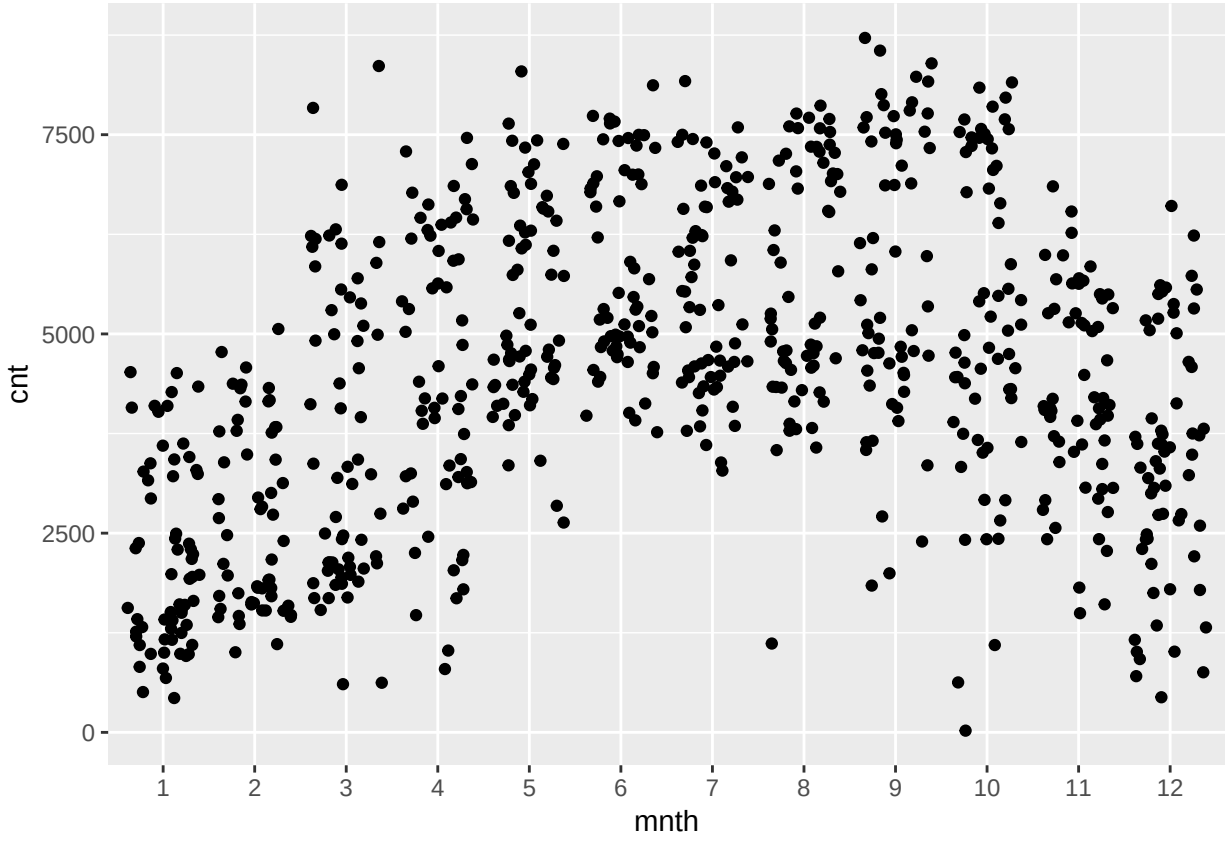
h.

```
sapply(DayDF, class)
```

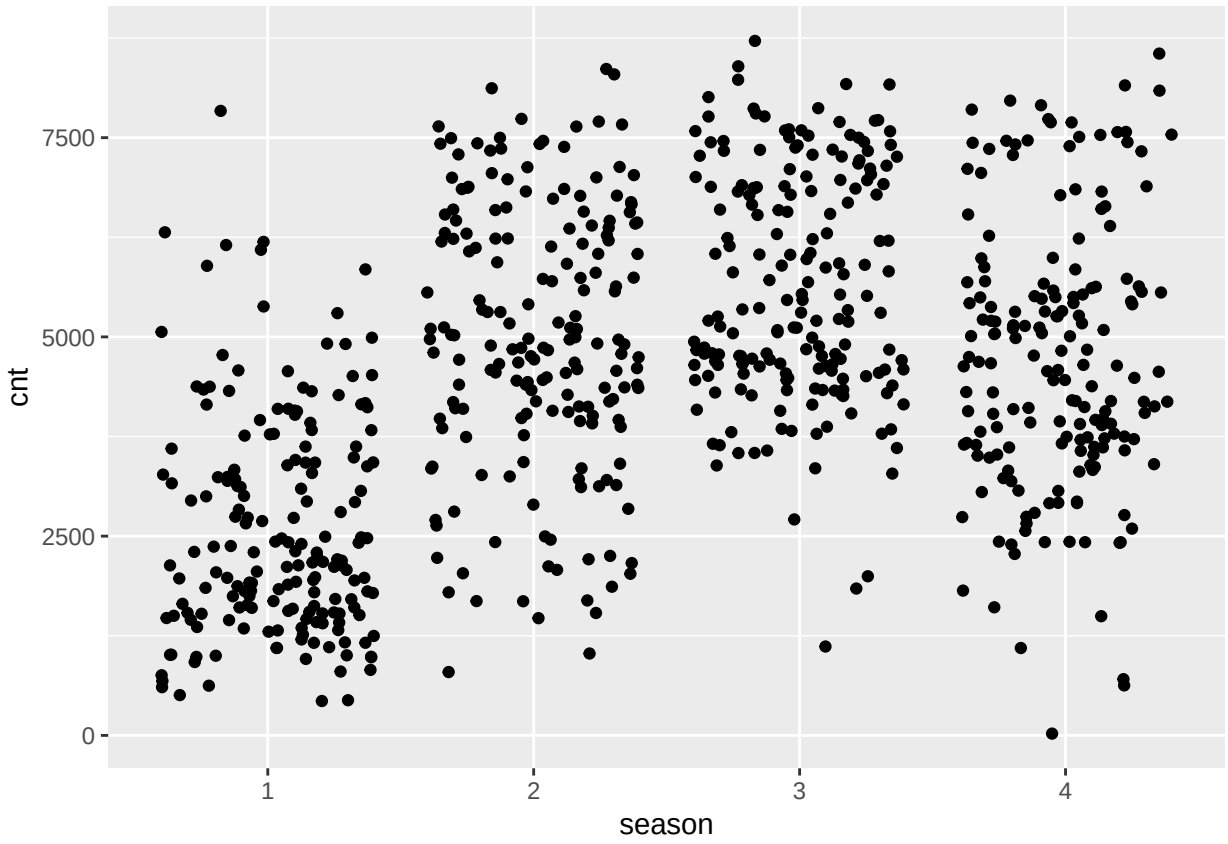
```
##    instant     dteday     season         yr       mnth    holiday
##  "integer"   "factor"   "factor"   "factor"   "factor"   "factor"
##    weekday workingday weathersit       temp      atemp        hum
##   "factor"   "factor"   "factor"  "numeric"  "numeric"  "numeric"
##  windspeed     casual registered        cnt
##  "numeric"  "integer"  "integer"  "integer"
```

i.

```
library(ggplot2)
ggplot(DayDF, aes(x = mnth, y = cnt)) + geom_jitter()
```




```
ggplot(DayDF, aes(x = season, y = cnt)) + geom_jitter()
```



2.

a.

```
set.seed(1861)
```

b.

```
sample(1:100, size = 10)
```

```
##  [1] 100  78  59  26  12  84  72  53  45  95
```

Sample 1:100 takes random set of numbers and chooses our designated

c.

```
DayDF <- DayDF[,-c(1,2,14,15)]
```

d.

```
trainSize <- 0.75
# get an indices that marks whether an observation is in the training or validation set
trainInd <- sample(1:nrow(DayDF), size = floor(nrow(DayDF) * trainSize))
dayTrain <- DayDF[trainInd,]
dayValidate <- DayDF[-trainInd,]
```

e.

```
library(leaps)
fittedReg <- regsubsets(cnt~., dayTrain, method = "forward")
```

```
## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 1 linear dependencies found
```

```
## Reordering variables and trying again:
```

```
summary(fittedReg)
```

```
## Subset selection object
## Call: regsubsets.formula(cnt ~ ., dayTrain, method = "forward")
## 29 Variables  (and intercept)
##             Forced in Forced out
## season2         FALSE      FALSE
## season3         FALSE      FALSE
## season4         FALSE      FALSE
## yr1             FALSE      FALSE
## mnth2           FALSE      FALSE
## mnth3           FALSE      FALSE
## mnth4           FALSE      FALSE
## mnth5           FALSE      FALSE
## mnth6           FALSE      FALSE
## mnth7           FALSE      FALSE
## mnth8           FALSE      FALSE
## mnth9           FALSE      FALSE
## mnth10          FALSE      FALSE
## mnth11          FALSE      FALSE
## mnth12          FALSE      FALSE
## holiday1        FALSE      FALSE
## weekday1        FALSE      FALSE
## weekday2        FALSE      FALSE
## weekday3        FALSE      FALSE
## weekday4        FALSE      FALSE
## weekday5        FALSE      FALSE
## weekday6        FALSE      FALSE
## weathersit2     FALSE      FALSE
## weathersit3     FALSE      FALSE
## temp            FALSE      FALSE
## atemp           FALSE      FALSE
## hum             FALSE      FALSE
## windspeed       FALSE      FALSE
```

```
## workingday1     FALSE      FALSE
## 1 subsets of each size up to 9
## Selection Algorithm: forward
##          season2 season3 season4 yr1 mnth2 mnth3 mnth4 mnth5 mnth6 mnth7
## 1  ( 1 ) " "     " "     " "     " " " " " "   " "   " "   " "   " "   " "
## 2  ( 1 ) " "     " "     " "     "*" " " " "   " "   " "   " "   " "   " "
## 3  ( 1 ) " "     " "     "*"     "*" " " " "   " "   " "   " "   " "   " "
## 4  ( 1 ) " "     " "     "*"     "*" " " " "   " "   " "   " "   " "   " "
## 5  ( 1 ) " "     " "     "*"     "*" " " " "   " "   " "   " "   " "   " "
## 6  ( 1 ) "*"     " "     "*"     "*" " " " "   " "   " "   " "   " "   " "
## 7  ( 1 ) "*"     " "     "*"     "*" " " " "   " "   " "   " "   " "   " "
## 8  ( 1 ) "*"     "*"     "*"     "*" " " " "   " "   " "   " "   " "   " "
## 9  ( 1 ) "*"     "*"     "*"     "*" " " " "   " "   " "   " "   " "   " "
##          mnth8 mnth9 mnth10 mnth11 mnth12 holiday1 weekday1 weekday2
## 1  ( 1 ) " "   " "   " "    " "    " "    " "      " "      " "
## 2  ( 1 ) " "   " "   " "    " "    " "    " "      " "      " "
## 3  ( 1 ) " "   " "   " "    " "    " "    " "      " "      " "
## 4  ( 1 ) " "   " "   " "    " "    " "    " "      " "      " "
## 5  ( 1 ) " "   " "   " "    " "    " "    " "      " "      " "
## 6  ( 1 ) " "   " "   " "    " "    " "    " "      " "      " "
## 7  ( 1 ) " "   "*"   " "    " "    " "    " "      " "      " "
## 8  ( 1 ) " "   "*"   " "    " "    " "    " "      " "      " "
## 9  ( 1 ) " "   "*"   " "    " "    " "    " "      " "      " "
##          weekday3 weekday4 weekday5 weekday6 workingday1 weathersit2
## 1  ( 1 ) " "      " "      " "      " "      " "         " "
## 2  ( 1 ) " "      " "      " "      " "      " "         " "
## 3  ( 1 ) " "      " "      " "      " "      " "         " "
## 4  ( 1 ) " "      " "      " "      " "      " "         " "
## 5  ( 1 ) " "      " "      " "      " "      " "         "*"
## 6  ( 1 ) " "      " "      " "      " "      " "         "*"
## 7  ( 1 ) " "      " "      " "      " "      " "         "*"
## 8  ( 1 ) " "      " "      " "      " "      " "         "*"
## 9  ( 1 ) " "      " "      " "      " "      " "         "*"
##          weathersit3 temp atemp hum windspeed
## 1  ( 1 ) " "         " "  "*"   " " " "
## 2  ( 1 ) " "         " "  "*"   " " " "
## 3  ( 1 ) " "         " "  "*"   " " " "
## 4  ( 1 ) "*"         " "  "*"   " " " "
## 5  ( 1 ) "*"         " "  "*"   " " " "
## 6  ( 1 ) "*"         " "  "*"   " " " "
## 7  ( 1 ) "*"         " "  "*"   " " " "
## 8  ( 1 ) "*"         " "  "*"   " " " "
## 9  ( 1 ) "*"         " "  "*"   " " "*"
```

```
fittedRegAdj <- lm(cnt~ temp+hum+season, data = dayTrain)
summary(fittedRegAdj)
```

```
##
## Call:
## lm(formula = cnt ~ temp + hum + season, data = dayTrain)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4249.0  -994.5  -270.1  1123.5  4350.7
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2375.1      313.1   7.585 1.46e-13 ***
## temp          6417.4      576.8  11.125  < 2e-16 ***
## hum          -2896.7      427.9  -6.769 3.37e-11 ***
## season2        908.1      220.3   4.122 4.34e-05 ***
## season3        561.3      287.7   1.951   0.0516 .
## season4       1529.1      186.5   8.199 1.76e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1370 on 542 degrees of freedom
## Multiple R-squared:  0.4882, Adjusted R-squared:  0.4835
## F-statistic: 103.4 on 5 and 542 DF,  p-value: < 2.2e-16
```

My adjusted R^2 value is .53 meaning it is loosely fits the projected model. I plotted variables temp, hum and seasons.

f.

g.

K fold cross validation is a validation method that splits the testing set into k groups. Then you leave one group out and use k-1 groups to find your predictions. Once done you average out your results and compare it to the un tested group. You then rinse and repeat for each one of the groups. LOOCV is a validation mehtod where you compare the entire data set while excluding one of the instances. By doing this you compare the entire testing data set and do so for every single instance in the testing set.

h.

i.

```r
# Cross-validation
k <- 10 # number of cross validated folds
# create an index for folds
folds <- sample(1:k, nrow(dayTrain), replace = TRUE)
# this matrix will
cv.errors <- matrix(NA, k, 20, dimnames = list(NULL, paste(1:20)))
# build a function to estimate predictions from the regsubsets function
predict.regsubsets = function(object, newdata, id, ...) {
form = as.formula(object$call[[2]])
mat = model.matrix(form, newdata)
coefi = coef(object, id = id)
mat[, names(coefi)] %*% coefi
}

# for j in 1:10 folds
for (j in 1:k) {
# estimate a best fit model
```

```
best.fit <- regsubsets(cnt ~ ., data = dayTrain[folds != j, ], nvmax = 20)
# for each model from 1 to 20 variables
for (i in 1:20) {
# get the predicted values from the model with i variables
preds <- predict(best.fit, dayTrain[folds == j, ], id = i)
# estimate root mean squared error for that model
cv.errors[j, i] <- sqrt(mean((dayTrain$cnt[folds == j] - preds)^2))
}
}
```

```
## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 1 linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 1 linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 1 linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 1 linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 1 linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 1 linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 1 linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 1 linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 1 linear dependencies found

## Reordering variables and trying again:

## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 1 linear dependencies found

## Reordering variables and trying again:
```

```
# vector of errors where i indicates the number of varibles used in the
# stepwise model.
mean.cv.errors <- apply(cv.errors, 2, mean)
mean.cv.errors
```

```
##         1         2         3         4         5         6         7
## 1899.4804 1579.6397 1103.8569 1004.6854  961.2594  903.5533  907.7590
##         8         9        10        11        12        13        14
##  941.7472  891.0225  877.1243  874.7976  863.5278  860.5582  831.8530
##        15        16        17        18        19        20
##  831.1952  828.7681  818.5123  809.4218  802.7376  802.8052
```