Rlab2

Eunjin Park

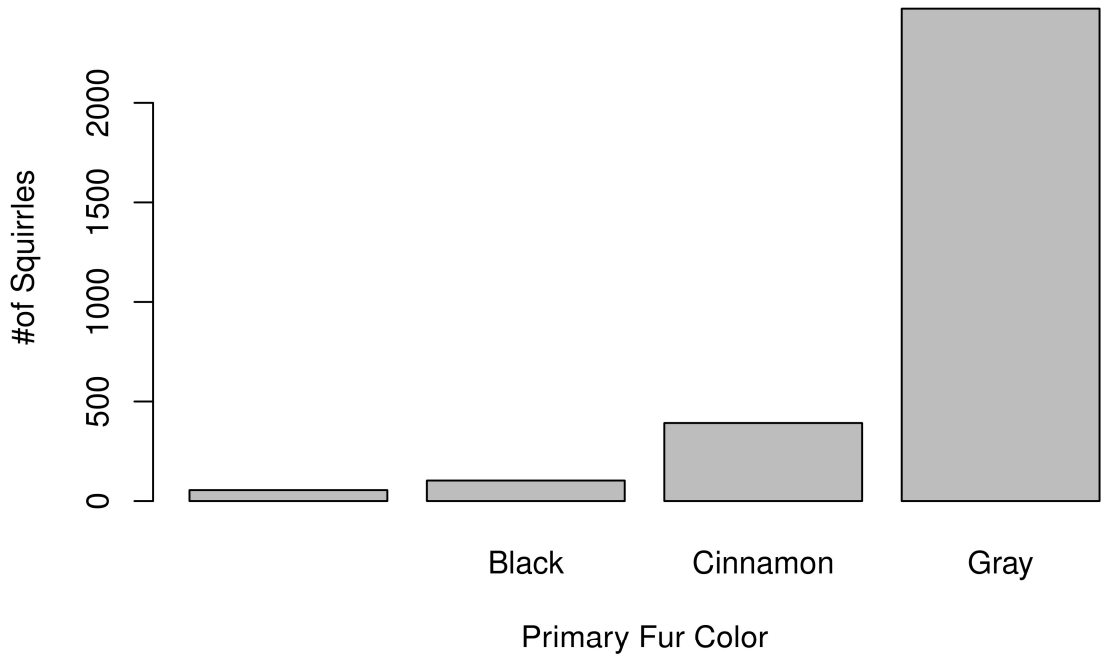
Sep,8th 2022

DELETE ANYTHING FROM THIS TEMPLATE BELOW THAT IS NOT PART OF YOUR SOLUTION.

I. Visualization

(1)Read the squirrel census data into a dataframe called squirrels in R. Make a barplot showing the number of observations for each of the possible levels in Primary.Fur.Color. Provide executable code (PEC).

```
## code for first question...
central = read.csv("2018_Central_Park_Squirrel_Census_-_Squirrel_Data (1).csv")
barplot(table(central$Primary.Fur.Color), xlab = "Primary Fur Color", ylab = "#of Squirrles")
```

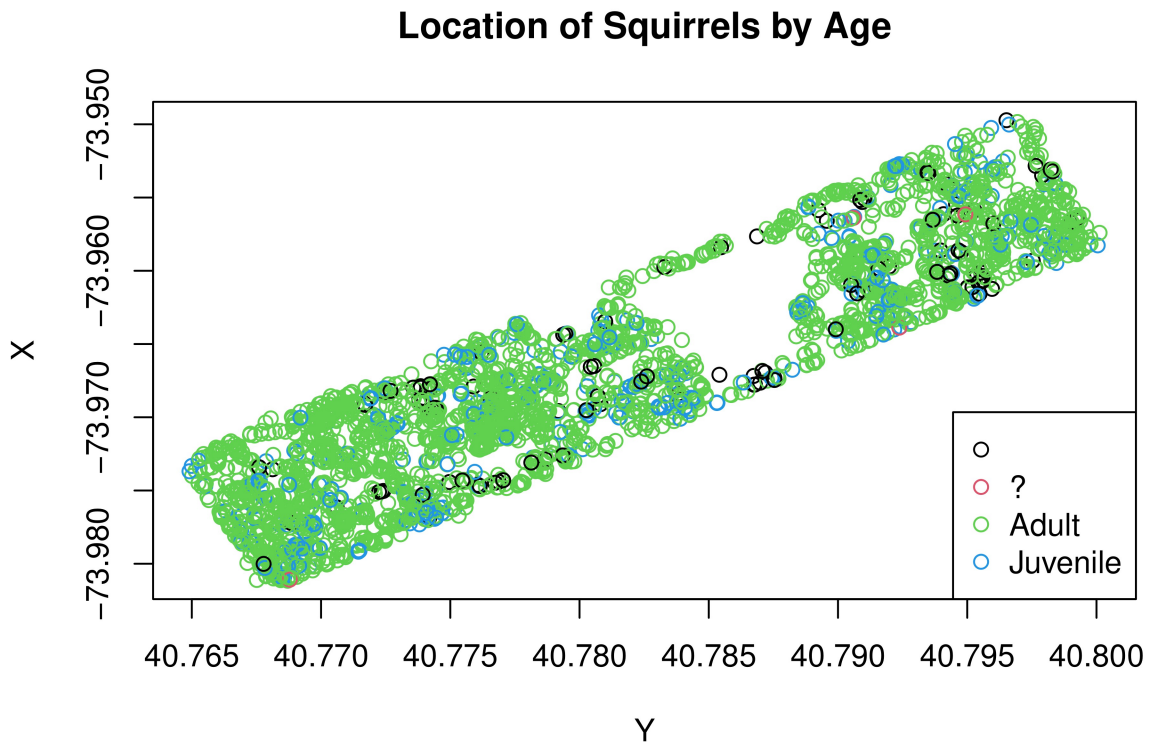


(2) Make a scatterplot showing the locations of each observation. Color each point based on the age of the observed squirrel, and add a legend the explains what color matches with each age type (hint: you may find it helpful to convert the Age variable into a factor). PEC.

```
central$Age = as.factor(central$Age)
nlevels(central$Age)
```

```
## [1] 4
```

```
plot(X~ Y, data= central, main= "Location of Squirrels by Age",  col = Age)
legend("bottomright", col = 1:4, pch = 1, legend = levels(central$Age))
```

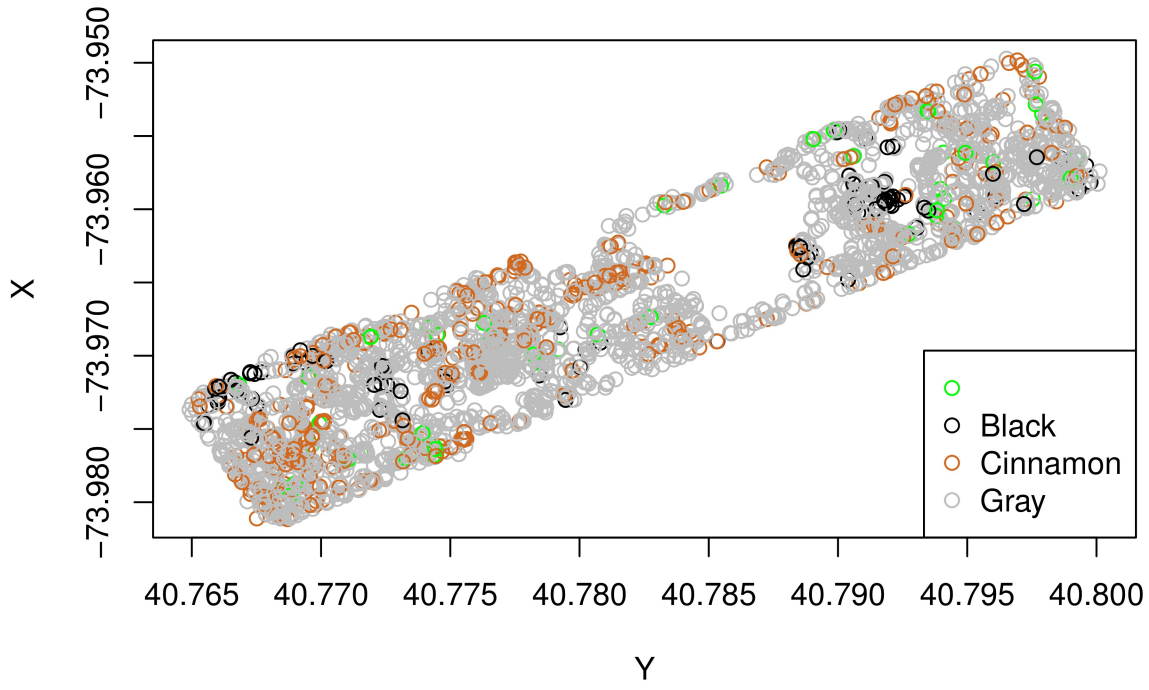


(3)Make another scatterplot that shows the observed points, now colored by the animal's primary fur color. Choose colors that approximately match the names of the possible colors. PEC.

```
color = c("Green", "Black","Chocolate", "Grey")
central$Primary.Fur.Color = as.factor(central$Primary.Fur.Color)
nlevels(central$Primary.Fur.Color)
```

```
## [1] 4
```

```
plot(X~Y, data=central, col=color[Primary.Fur.Color])
legend("bottomright", col=color, pch = 1, legend = c(levels(central$Primary.Fur.Color)))
```

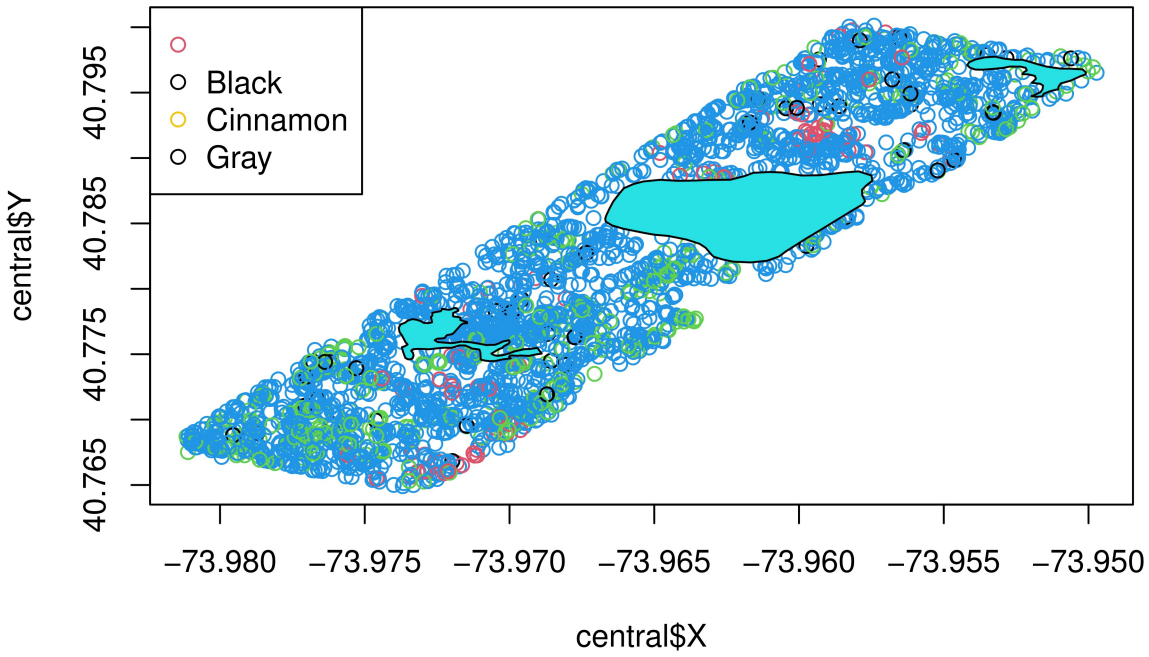


There are some big holes in the scatterplot corresponding to bodies of water in Central Park. Load the CentralParkWater.RData file into R to access a dataframe called water_boundaries. There are three variables in the dataframe: Longitude and Latitude, which give coordinates describing the boundaries of three lakes, and Lake, which gives the name of the body of water.

```
load("CentralParkWater.RData")
```

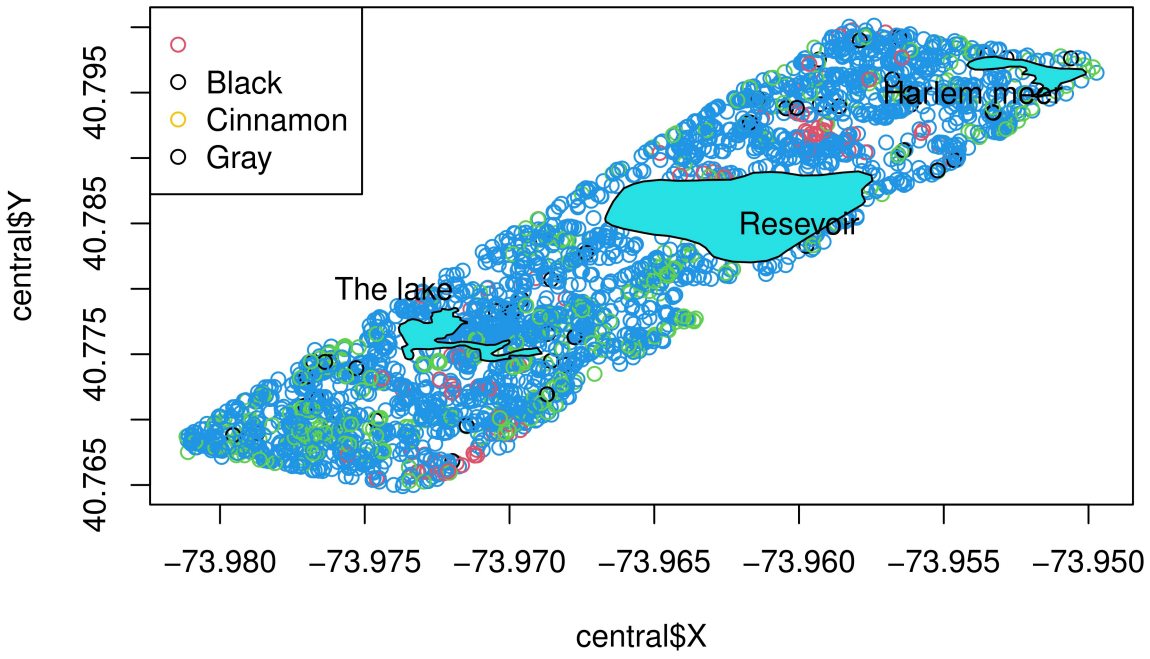
(4)Use the polygon() function to add three separate light blue polygons to your second scatter plot, one for each body of water. PEC.

```
central$Primary.Fur.Color.A = as.factor(central$Primary.Fur.Color)
plot(central$X, central$Y, col = central$Primary.Fur.Color.A)
legend("topleft", col = c(2,1,7,9), pch =1, legend = levels(central$Primary.Fur.Color.A))

polygon(water_boundaries$Longitude[water_boundaries$Lake == "resevoir"], water_boundaries$Latitude[wate
polygon(water_boundaries$Longitude[water_boundaries$Lake == "the lake"], water_boundaries$Latitude[wate
polygon(water_boundaries$Longitude[water_boundaries$Lake == "harlem meer"], water_boundaries$Latitude[w
```



(5) [3 pts] Use the text() function to add labels showing the appropriate names of each lake. PEC

```
central$Primary.Fur.Color.A = as.factor(central$Primary.Fur.Color)
plot(central$X, central$Y, col = central$Primary.Fur.Color.A)
legend("topleft", col = c(2,1,7,9), pch =1, legend = levels(central$Primary.Fur.Color.A))

polygon(water_boundaries$Longitude[water_boundaries$Lake == "resevoir"], water_boundaries$Latitude[wate
polygon(water_boundaries$Longitude[water_boundaries$Lake == "the lake"], water_boundaries$Latitude[wate
polygon(water_boundaries$Longitude[water_boundaries$Lake == "harlem meer"], water_boundaries$Latitude[w
text(-73.974, 40.780, labels = "The lake")
text(-73.960, 40.785, labels = "Resevoir")
text(-73.954, 40.795, labels = "Harlem meer")
```



II. Functions

(6) [6 pts] Fill in the blanks to create two functions: one that computes the logit transform for a value x, and another that computes the inverse-logit. Check that they really are inverses of each other by applying them in as shown.

```
logit = function(x =NULL){out = log(x/(1-x))
return(out)}
inverlogit = function(x = NULL){out = 1/(1-exp(x))
return(out)}
```

(7) [6 pts] PEC for a function that computes the Trimean of a vector of values (hint: you will probably need to use the quantile() function). Verify your function returns the same value as mine for the given input.

```
x = c(1,3, 4.3,7,10,2,2.3,4)
trimean = function(x){return((quantile(x)[2] + 2*quantile(x)[3] + quantile(x)[4])/4)}
trimean( c(1,3, 4.3,7,10,2,2.3,4))
```

```
##   25%
## 3.55
```

III. for loops, *apply() functions [12 pts]

(8) [4 pts] PEC that runs a for loop which converts each true/false character-string behavior variable in squirrels to a logical-type variable.

```
for(i in 13:29){
  central[,i] = as.logical(central[,i])}
```

```
central$Running = as.logical(central$Running)
central$Chasing = as.logical(central$Chasing)
```

(9) [3 pts] Use an *apply() function to produce a vector giving the class of each variable in squirrels. PEC. Verify that the correct variables are now logical.

```
sapply(central, class)
```

```
##                                          X
##                                  "numeric"
##                                          Y
##                                  "numeric"
##                           Unique.Squirrel.ID
##                                  "character"
##                                     Hectare
##                                  "character"
##                                       Shift
##                                  "character"
##                                        Date
##                                  "integer"
##                      Hectare.Squirrel.Number
##                                  "integer"
##                                         Age
##                                   "factor"
##                             Primary.Fur.Color
##                                   "factor"
##                           Highlight.Fur.Color
##                                  "character"
## Combination.of.Primary.and.Highlight.Color
##                                  "character"
##                                 Color.notes
##                                  "character"
##                                    Location
##                                   "logical"
##            Above.Ground.Sighter.Measurement
##                                   "logical"
##                               Specific.Location
##                                   "logical"
##                                     Running
##                                   "logical"
##                                     Chasing
##                                   "logical"
##                                    Climbing
##                                   "logical"
##                                      Eating
##                                   "logical"
##                                    Foraging
##                                   "logical"
##                             Other.Activities
```

```
##                                   "logical"
##                                        Kuks
##                                   "logical"
##                                       Quaas
##                                   "logical"
##                                       Moans
##                                   "logical"
##                                  Tail.flags
##                                   "logical"
##                                Tail.twitches
##                                   "logical"
##                                  Approaches
##                                   "logical"
##                                 Indifferent
##                                   "logical"
##                                  Runs.from
##                                   "logical"
##                           Other.Interactions
##                                  "character"
##                                     Lat.Long
##                                  "character"
##                          Primary.Fur.Color.A
##                                    "factor"
```

(10) [5 pts] Use either a for loop or an *apply() function to compute the proportion of TRUE values for each behavior. PEC. Which behavior has the most TRUE values?

```
for(i in 1:ncol(central)){ if(is.logical(central[,i])){sum(central[,i])}}
```