STAT 847: Analysis Assignment 1

Chris Binoi Verghese ID:21092999

Q1. (10 marks) Currently in each of three "scraped data" datasets, one row represents one horse. Make a new dataset for the Woodbine dataset where one line represents one race instead. Keep all the variables pertaining to the race and drop the ones pertaining to the horse (that is, drop `horse_number`,`horse_name`,`horse_sire`,`horse_trainer`,`horse_jockey`,`horse_odds`,`horse_odds_decimal`, and `horse_place`.)

Use the mini case study that uses one large `ddply()` function as a basis for your code.

Show your code and the first 3 rows of the new dataset.

Since this question makes other, later questions easier, you may use the "afterQ1" datasets for Q2 onward.

```
# Load the dataset
woodbine_data <- read.csv("HRN woodbine scraped data 2023-12-04.csv")

library(plyr)
library(tidyverse)
#Use ddply to summarize based on racecount and only display information that are
#consistent per race
woodbine_race <- woodbine_data %>% ddply("racecount", summarize,
                                         racecount = first(racecount),
                                         meet_location = first(meet_location),
                                         meet_wday = first(meet_wday),
                                         meet_mday = first(meet_mday),
                                         meet_year = first(meet_year),
                                         purse = first(purse),
                                         time_frac1 = first(time_frac1),
                                         time_frac2 = first(time_frac2),
                                         time_frac3 = first(time_frac3),
                                         time_frac4 = first(time_frac4),
                                         time_frac5 = first(time_frac5),
                                         time_final = first(time_final),
                                         track_length = first(track_length),
                                         track_type = first(track_type),
                                         race_class = first(race_class),
                                         dist_frac1 = first(dist_frac1),
                                         dist_frac2 = first(dist_frac2),
                                         dist_frac3 = first(dist_frac3),
                                         dist_frac4 = first(dist_frac4),
                                         dist_frac5 = first(dist_frac5))


# Display the first 3 rows of the new dataset
woodbine_race[1:3,]
```

```
##   racecount meet_location meet_wday meet_mday meet_year  purse time_frac1
## 1         1      Woodbine    Sunday August 21      2022  64300      23.40
```

```
## 2         2      Woodbine    Sunday August 21      2022 123200      23.34
## 3         3      Woodbine    Sunday August 21      2022 125000      23.05
##   time_frac2 time_frac3 time_frac4 time_frac5 time_final track_length
## 1      47.04         NA         NA         NA      59.48           5F
## 2      46.99      71.82      96.58         NA     103.13      1 1/16M
## 3      46.25      69.88         NA         NA      76.14       6 1/2F
##          track_type                       race_class dist_frac1 dist_frac2
## 1        Inner turf $40,000 Maiden Optional Claiming        1/4        1/2
## 2              Turf            Maiden Special Weight        1/4        1/2
## 3 All Weather Track               Sweet Briar Too S.        1/4        1/2
##   dist_frac3 dist_frac4 dist_frac5
## 1
## 2        3/4       MILE
## 3        3/4
```

Q2. (5 marks) At Woodbine, calculate the average time it takes for the winning horse to complete a race of each available length (hint: use the by() command). Present your answer as a table like the following, and round average times to two decimal places.

Event length	Average Time
3F (3 Furlongs)	53.25
6F	101.42
1M (1 Mile, 8 Furlongs)	

```
#Average Race final times per Track Length
by(woodbine_race$time_final, woodbine_race$track_length, function(x) round(mean(x), 2)) %>%
  knitr::kable(col.names = c("Event length", "Average Time"))
```

Event length	Average Time
1 1/16M	86.21
1 1/2M	91.74
1 1/4M	87.33
1 1/8M	84.54
1 3/4M	92.31
1 3/8M	88.10
1M	88.01
1M 70Y	98.69
4 1/2F	65.19
5 1/2F	77.81
5F	77.02
6 1/2F	82.01
6F	82.87
7 1/2F	84.23
7F	84.02

Q3. (5 marks) At Woodbine, find the probability of a horse coming in second place as a function of the decimal odds, rounded to the nearest whole number. Present your answer as a table like the following, and round probabilties to three decimal places. You may use the provided EDA code for the first place probabilities as a starting point.

Rounded Odds	Probability of 2nd
1	0.142
2	0.241
3	

```
#Rounded odds probability of a horse coming second
woodbine_data %>% mutate(rounded_odds = round(horse_odds_decimal)) %>%
  ddply("rounded_odds", summarize,
        Mean_Time = round(mean(horse_place == 2, na.rm = TRUE), 3)) %>%
  knitr::kable(col.names = c("Rounded Odds", "Probability of 2nd"))
```

Rounded Odds	Probability of 2nd
0	0.500
1	0.250
2	0.216
3	0.196
4	0.147
5	0.120
6	0.111
8	0.107
10	0.094
12	0.099
15	0.065
20	0.051
30	0.000
50	0.000

Q4. (6 marks) At Woodbine, conduct a two-sample t-test to see if the finish times differ on average between turf tracks and the all weather track for 6F (6 furlong) length races. For this question, assume that 'inner turf' and 'turf' are both turf tracks that belong in the same group. Use alpha = 0.05 as your cut-off for significance.

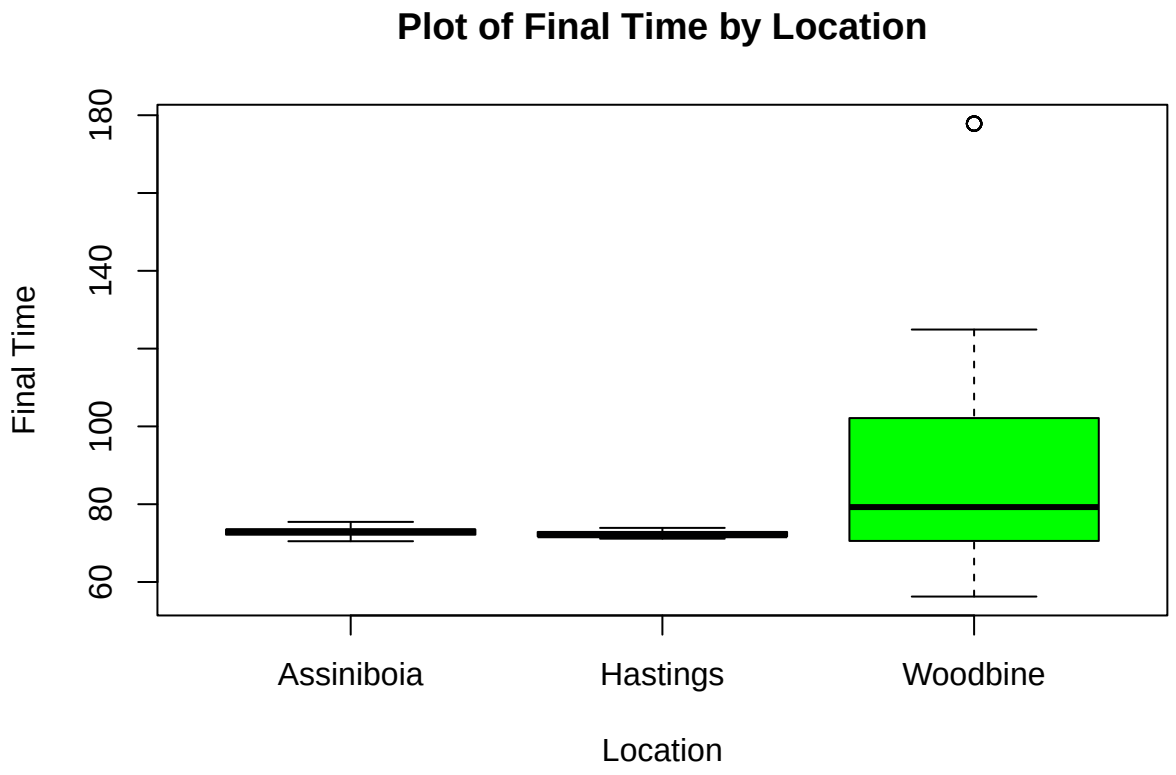
```
sample1 <- woodbine_race %>%
  filter(track_type != "All Weather Track" & track_length == "6F")

# Create sample2
sample2 <- woodbine_race %>%
  filter(track_type == "All Weather Track" & track_length == "6F")

# Perform t-test
t.test(sample1$time_final, sample2$time_final)
```

```
##
##  Welch Two Sample t-test
##
## data:  sample1$time_final and sample2$time_final
## t = -0.043617, df = 95.02, p-value = 0.9653
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -4.862445  4.653376
## sample estimates:
## mean of x mean of y
##  82.78309  82.88763
```

The obtained p-value of approximately 0.9653 which is much greater than 0.05 suggesting that there is insufficient evidence to reject the null hypothesis that track type affects the time to finish the race. This indicates that there is no statistically significant difference in the mean finish times between the different track types as the calculated means of the two samples are 82.78309 and 82.88763 .

Q5. (8 marks) Make a side-by-side boxplot of the finish times for 6F races between the three locations. That is, make a boxplot where each of the three boxes shows the distribution of times from Woodbine, Assiniboia, or Hastings. Either base R or ggplot is acceptable.

```
#Read the Hastings and Assiniboia Race data
hastings_data <- read.csv("HRN hastings scraped data 2023-12-04.csv")
assiniboia_data <- read.csv("HRN assiniboia scraped data 2023-12-04.csv")
#Only use the final time for races where track length is 6F for all three locations
assiniboia_race <- assiniboia_data[assiniboia_data$track_length == "6F", "time_final"]
hastings_race <- hastings_data[hastings_data$track_length == "6F", "time_final"]
woodbine_race <- woodbine_data[woodbine_data$track_length == "6F", "time_final"]
#Create a list containing all three locations
all_data <- list(assiniboia_race, hastings_race, woodbine_race)
#Plot these finish times using a box plot
boxplot(all_data, col = c("red", "blue", "green"), main = "Plot of Final Time by Location",
        xlab = "Location", ylab = "Final Time", names = c("Assiniboia", "Hastings", "Woodbine"))
```



Q6. (8 marks) Find the names of the five horses that have won the most evnets at Woodbine (in 2022 and 2023 combined) and their total number of wins. Present their results in a table like so.

Horse	Wins
Rainbow Dash	11
Twilight Sparkle	7
The cowboy one	6

```
#Create a dataframe consisting of the name of horses and the number of wins
winning_horse <- woodbine_data[woodbine_data$horse_place == 1
                               & !is.na(woodbine_data$horse_place),] %>%
  ddply("horse_name",summarize, wins=length(racecount)) %>%
  arrange(desc(wins))
#Top 5 winning horses need to be added into a table
head(winning_horse, 5) %>%
  knitr::kable(col.names = c("Horse", "Wins"))
```

Horse	Wins
Canadiansweetheart	8
Patches O'Houlihan	8
Hallie's Hero	6
Wentru	6
C C's Kingdom	5

Q7. (8 marks) Typically, a purse is divided so that 60% goes to the winner, 20% goes to 2nd place, 10% goes to 3rd place, and the remaining 10% is split among all the other horses that finish. Assume that this purse payout system is used at Woodbine. Find the names of the five horses that have won the most money at Woodbine (in 2022 and 2023 combined) and their total winnings during these two years.

Horse	Prize Money
Rainbow Dash	654,000
Twilight Sparkle	321,000
The cowboy one	

```
#Create dataframe with horse information and new column with number of horses
modified_data <- woodbine_data[, c("racecount","horse_name","horse_place","purse")] %>%
  drop_na(horse_place)%>%
  group_by(racecount) %>%
  mutate(num_horses = sum(!horse_place %in% c(1, 2, 3)))

#Add new column (prizes) which calculates each horse's prize money
prizes <- modified_data %>%
  mutate(prize = ifelse(horse_place == 1, 0.6 * purse,
                        ifelse(horse_place == 2, 0.2 * purse,
                               ifelse(horse_place == 3, 0.1 * purse,
                                      0.1 * purse/num_horses))) ,)

#Calculate cumulative prizes for each horse and arrange in descending order
complete_prizes <- prizes %>% ddply("horse_name", summarize,Prize_Money = sum(prize)) %>%
  arrange(desc(Prize_Money))
#Print 5 largest earned prizes in a table
head(complete_prizes, 5) %>%
  knitr::kable(col.names = c("Horse", "Prize Money"))
```

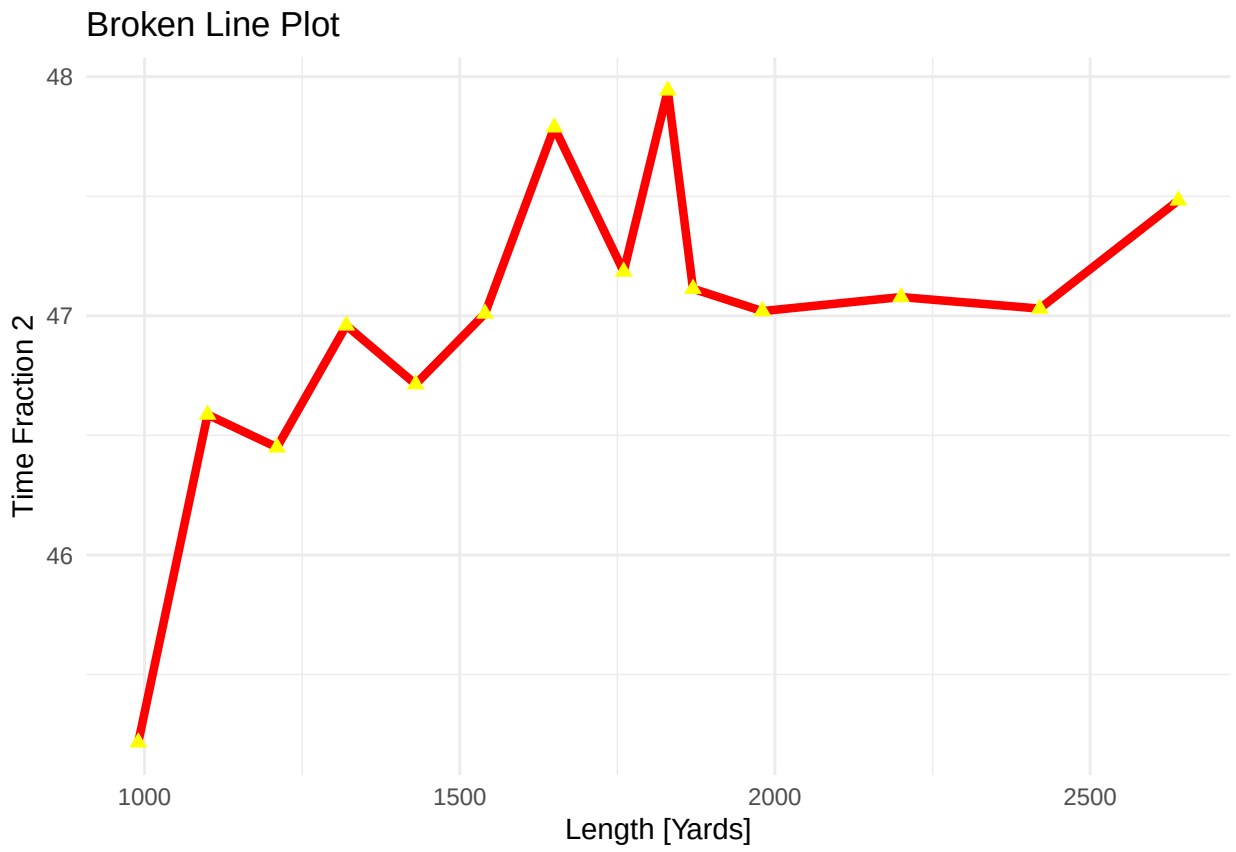
Horse	Prize Money
Last Call	919660.0
Patches O'Houlihan	703320.0
Bushido	686750.0
Malibu Secret	681609.4
Moira	680396.7

Q8. (10 marks) Every race has fractional times, which are the times when the leading horse finishes some fraction of the race. For every race that is between 4 1/2F and 1 9/16M inclusive, the second fraction (`time_frac2`) is the time that the first horse finishes 1/2 a mile (4 furlongs).

Plot as a broken line plot of `time_frac2` as a function of distance for all the distances between 4 1/2F and 1 9/16M. Be sure to convert the distances into something numeric like number of furlongs; 1 mile is 8 furlongs.

```
library(stringr)
#Function to read a distance in string and return numeric value in yards
convert_to_yards_regex <- function(distance_string) {
  #Regex to Map in format - 15M or 16 F
  match_result1 <- str_match(distance_string, "^(\\d+)([MF])$")
  #Regex to Map in format - 15 4/13M or 16 4/5F
  match_result2 <- str_match(distance_string, "^(\\d+)\\s(\\d+)/(\\d+)([MF])$")
  #Regex to Map in format - 15M 10Y or 16 F 5Y
  match_result3 <- str_match(distance_string, "^(\\d+)([MF])\\s(\\d+)[Y]$")
  if (!is.na(match_result1[,1])) {
    #Get Values if in Format 1
    numeric_value <- as.numeric(match_result1[, 2])
    unit <- as.character(match_result1[, 3])
    yards <- 0
  }
  else if (!is.na(match_result2[,1])) {
    #Get Values if in Format 2
    numerator <- as.numeric(match_result2[, 3])
    denominator <- as.numeric(match_result2[, 4])
    numeric_value <- as.numeric(match_result2[, 2]) + (numerator / denominator)
    unit <- as.character(match_result2[, 5])
    yards <- 0
  }
  else if (!is.na(match_result3[,1])) {
    #Get Values if in Format 3
    # Handle the case with a space, a '/', and a second number
    yards <- as.numeric(match_result3[, 4])
    numeric_value <- as.numeric(match_result3[, 2])
    unit <- as.character(match_result3[, 3])
  }
    #Calculate yards depending on unit
    if (unit == "F") {
      # Convert furlongs to yards (1 Furlong = 220 yards)
      return((numeric_value * 220) + yards)
    }
    else if (unit == "M") {
      # Convert miles to yards (1 Mile = 1760 yards)
      return((numeric_value * 1760)+ yards)
    }
}

#Adding a new column calculating track length in yards
woodbine_tracks <- woodbine_data %>%
  rowwise() %>%
  mutate(track_length_yards = convert_to_yards_regex(track_length)) %>%
  ungroup()
#Filter out track length that is between 4 1/2F and 1 9/16M inclusive
woodbine_tracks_filtered <- woodbine_tracks%>%
```

```
  filter(!is.na(track_length_yards) & track_length_yards >= 990 & track_length_yards <= 2750)

#Plotting filtered track_length with its mean time fractions for a broken line plot
plot = ddply(woodbine_tracks_filtered, "track_length_yards",summarise,
               AvgTime = mean(time_frac2, na.rm=T))
cplot <- ggplot(data = plot, aes(x = track_length_yards, y = AvgTime)) +
  geom_line(color = "red", size = 1.5) +
  geom_point(shape = 17, color = "yellow", size = 2) +
  theme_minimal() +
  labs(title = "Broken Line Plot",
       x = "Length [Yards]",
       y = "Time Fraction 2")
print(cplot)
```



Q9. (5 marks) Fit a quadratic model using `lm()` of `time_frac2` as a function of distance for all the distances between 4 1/2F and 1 9/16M. Be sure to convert the distances into something numeric like number of furlongs; 1 mile is 8 furlongs. Report the `summary()` of the model.

```
#Quadratic Model
model <- lm(AvgTime ~ poly(track_length_yards, degree = 2, raw = TRUE), data = plot)

cat("\n Coefficients:\n", coef(model),
    "\n Residual Standard Error: ", summary(model)$sigma,
    "\n R-squared: ", summary(model)$r.squared, "\n")
```

```
##
##  Coefficients:
##  41.55407 0.005671441 -1.360623e-06
##  Residual Standard Error:  0.4309528
##  R-squared:  0.6300253
```

```
#Summary of model
summary(model)
```

```
##
## Call:
## lm(formula = AvgTime ~ poly(track_length_yards, degree = 2, raw = TRUE),
##     data = plot)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.61893 -0.28660 -0.09414  0.40163  0.58031
##
## Coefficients:
##                                                  Estimate Std. Error t value
## (Intercept)                                     4.155e+01  1.505e+00  27.615
## poly(track_length_yards, degree = 2, raw = TRUE)1  5.671e-03  1.753e-03   3.235
## poly(track_length_yards, degree = 2, raw = TRUE)2 -1.361e-06  4.852e-07  -2.804
##                                                 Pr(>|t|)
## (Intercept)                                     1.64e-11 ***
## poly(track_length_yards, degree = 2, raw = TRUE)1  0.00794 **
## poly(track_length_yards, degree = 2, raw = TRUE)2  0.01714 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.431 on 11 degrees of freedom
## Multiple R-squared:   0.63,  Adjusted R-squared:  0.5628
## F-statistic: 9.366 on 2 and 11 DF,  p-value: 0.004216
```