Wargaming Assessment

Albert Na

November 9, 2019

Reading in Data

```
library(ggplot2)
```

```
## Registered S3 methods overwritten by 'ggplot2':
##   method         from
##   [.quosures     rlang
##   c.quosures     rlang
##   print.quosures rlang
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
##
##     date
```

```
setwd("C:/Users/Na_Al/Desktop/wargaming")
items <- read.csv("item_list.csv")
orders <- read.csv("order_list.csv")
players <- read.csv("players_list.csv")
rcamp <- read.csv("result_campaign_raw.csv")
```

Part 1: Descriptive Questions

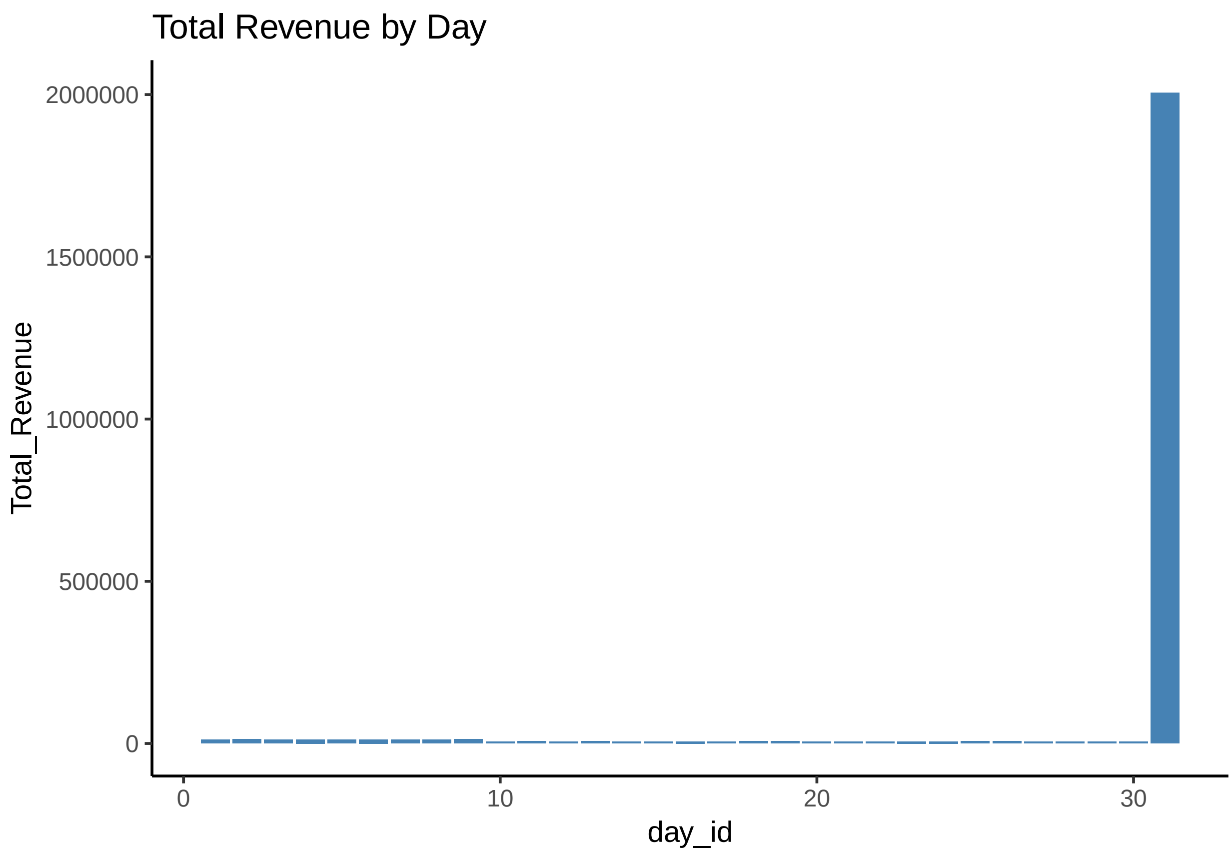
1. Plot the following chart: x_axis = day_id; y_axis = revenues; y_axis(dual axis) = paying users.

```
#Create revenues column -> I will define as item_price * item_cnt
itemorders <- merge(orders, items, by = c("item_id"))
itemorders$revenue <- itemorders$item_price * itemorders$item_cnt

#Plot total revenue by day in current quarter
totalrev <- itemorders %>% group_by(day_id) %>%  summarise(Total_Revenue = sum(revenue))
```

```r
totalrev$day_id <- sub("\\/.*", "", totalrev$day_id)
totalrev$day_id <- as.integer(totalrev$day_id)

#Table to see total revenue by each day
totalrev <- totalrev[order(totalrev$day_id), ]
totalrev
```

```
## # A tibble: 31 x 2
##    day_id Total_Revenue
##     <int>         <int>
##  1      1         11565
##  2      2         12815
##  3      3         12290
##  4      4         12535
##  5      5         11675
##  6      6         12510
##  7      7         11659
##  8      8         12197
##  9      9         13085
## 10     10          5715
## # ... with 21 more rows
```

```r
ggplot(totalrev, aes(x = day_id, y = Total_Revenue)) + geom_bar(stat = 'identity', position = 'dodge',fi
```



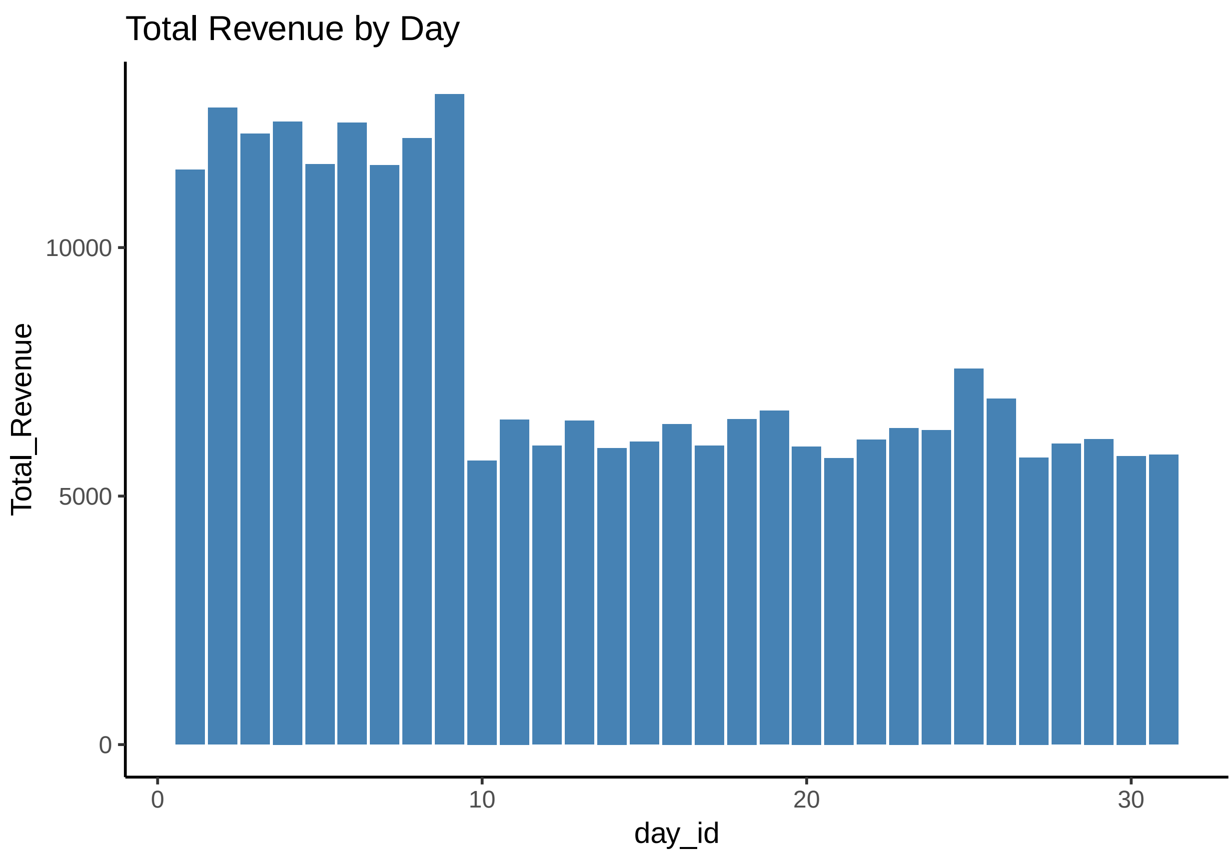
- From this plot, we can immediately see a heavy left skew of the data, with all of the revenue seemingly coming from the 31st.

```r
itemorders[which.max(itemorders$revenue), ]
```

```
##     item_id     day_id user_id item_cnt item_name item_description
## 642   24589 31/01/2016   14213    99999     TankB       mediumTank
##     item_price revenue
## 642         20 1999980
```

- We see that this skew is a result of either error in data collection, or a "whale" customer purchasing 99999 quantity of Tank B on the 31st, massively inflating revenue for this day.

```r
adjusted <- itemorders
adjusted <- itemorders[-642, ]
adjusted <- adjusted %>% group_by(day_id) %>%  summarise(Total_Revenue = sum(revenue))
adjusted$day_id <- sub("\\/.*", "", adjusted$day_id)
adjusted$day_id <- as.integer(adjusted$day_id)
ggplot(adjusted, aes(x = day_id, y = Total_Revenue)) + geom_bar(stat = 'identity', position = 'dodge',fi
```



- This is how the revenue graph would look like if we removed the outlier value. Now, we see that the points where the revenue is at its highest, are towards the beginning, possibly as a result of the beginning of an event or sale that encourages players to make purchases. After the 9th, which I would assume to be the conclusion of the event, revenue drops dramatically and plateaus in this range up until the end of the quarter.

2. What is the total revenue for the period?

```r
sum(itemorders$revenue)
```

```
## [1] 2247632
```

- Total revenue after removing the one outlier value

```
sum(itemorders[-642, 8])
```

```
## [1] 247652
```

3. How many paying players do we have?

```
#Check how many players made an order
paying <- merge(players, orders, by = "user_id")

#Since some players made multiple orders, check distinct user ids
length(unique(orders$user_id))
```

```
## [1] 5921
```

- There are 5921 paying players.

4. What is the percentage of paying players among all players?

```
5921 / 20000
```

```
## [1] 0.29605
```

- 29.6% of players are paying players.

5. What is the top item in terms of revenue?

```
itemorders %>% group_by(item_name) %>% summarise(totalrevenue = sum(revenue))
```

```
## # A tibble: 9 x 2
##   item_name totalrevenue
##   <fct>            <int>
## 1 GoldPack1        24400
## 2 GoldPack2        16860
## 3 GoldPack3        29040
## 4 GoldPack4        12122
## 5 TankA             4690
## 6 TankB          2025320
## 7 TankC            28550
## 8 TankD            47150
## 9 TankE            59500
```

- We see that the top item in terms of revenue is Tank B, with $2,025,320 in revenue

```
#Top item in terms of revenue after removing the one outlier customer
itemorders[-642, ] %>% group_by(item_name) %>% summarise(totalrevenue = sum(revenue))
```

```
## # A tibble: 9 x 2
##   item_name totalrevenue
##   <fct>            <int>
## 1 GoldPack1        24400
## 2 GoldPack2        16860
## 3 GoldPack3        29040
## 4 GoldPack4        12122
## 5 TankA             4690
## 6 TankB            25340
## 7 TankC            28550
## 8 TankD            47150
## 9 TankE            59500
```

- We see that after removing the one outlier customer, the top item in terms of revenue becomes Tank E, with $59,500 in revenue.

6. What is the top item in terms of units sold?

```
itemorders %>% group_by(item_name) %>% summarise(totalsold = sum(item_cnt))
```

```
## # A tibble: 9 x 2
##   item_name totalsold
##   <fct>         <int>
## 1 GoldPack1      2440
## 2 GoldPack2       843
## 3 GoldPack3       968
## 4 GoldPack4      1102
## 5 TankA           469
## 6 TankB        101266
## 7 TankC          1142
## 8 TankD          1886
## 9 TankE          1190
```

- Again, without adjusting for the outlier value, Tank B is the top item in terms of units sold.

```
itemorders[-642,] %>% group_by(item_name) %>% summarise(totalsold = sum(item_cnt))
```

```
## # A tibble: 9 x 2
##   item_name totalsold
##   <fct>         <int>
## 1 GoldPack1      2440
## 2 GoldPack2       843
## 3 GoldPack3       968
## 4 GoldPack4      1102
## 5 TankA           469
## 6 TankB          1267
## 7 TankC          1142
## 8 TankD          1886
## 9 TankE          1190
```

- After adjustment, we now see that GoldPack1 is the top item in terms of units sold.

7. At which day of the week (1 = Monday,. . . , 7 = Sunday) the average transaction (in EUR) is the highest?

- Note: From this point, I will just remove the outlier value since it skews the results of the data too much. Also, for this question, normally I would use an R package (lubridate) which would take in the date and accurately output the corresponding weekday on the calendar, but since we are going by the assumption that the 1st will be Monday and the 7th will be Sunday, I will just input it manually.

```
#Create weekdays column with a repeating pattern from 1-7, where 1 = Monday, 7 = Sunday.
weekdate <- data.frame(x=1:31)
x <- weekdate$x
x[] <- 1:7
```

```
## Warning in x[] <- 1:7: number of items to replace is not a multiple of
## replacement length
```

```
adjusted$weekdate <- x

adjusted$weekdate[adjusted$weekdate == 1] <- "Monday"
adjusted$weekdate[adjusted$weekdate == 2] <- "Tuesday"
```

```r
adjusted$weekdate[adjusted$weekdate == 3] <- "Wednesday"
adjusted$weekdate[adjusted$weekdate == 4] <- "Thursday"
adjusted$weekdate[adjusted$weekdate == 5] <- "Friday"
adjusted$weekdate[adjusted$weekdate == 6] <- "Saturday"
adjusted$weekdate[adjusted$weekdate == 7] <- "Sunday"

#Table with average revenue by weekday
adjusted %>% group_by(weekdate) %>% summarise(averagerevenue = mean(Total_Revenue))
```

```
## # A tibble: 7 x 2
##   weekdate  averagerevenue
##   <chr>              <dbl>
## 1 Friday             9708
## 2 Monday             8392.
## 3 Saturday           7354
## 4 Sunday             7605
## 5 Thursday           6532.
## 6 Tuesday            8518.
## 7 Wednesday          7662.
```

- The average transaction (in EUR) is highest on Friday.

8. What is the average amount of items bought for paying users?

```r
#Remove outlier value
orders[which.max(orders$item_cnt), ]
```

```
##          day_id user_id item_cnt item_id
## 6740 31/01/2016   14213    99999   24589
```

```r
ordersadj <- orders[-6740, ]

#Find amount of items purchased per player
averageitems <- ordersadj %>% group_by(user_id) %>% summarize(itemsperplayer = sum(item_cnt))
#Average items for total population
sum(averageitems[, 2])/ 5921
```

```
## [1] 1.909644
```

- The average amount of items bought for paying users is 1.91 items.

9. Fill in the table

```r
itemorders <- itemorders[-642, ]
#Paying Players per item
#For cases where users buy the same item on more than one occasion, only count that item once
paid <- itemorders
paid <- paid %>% group_by(user_id) %>% distinct(item_name)
table(paid$item_name)
```

```
##
## GoldPack1 GoldPack2 GoldPack3 GoldPack4     TankA     TankB     TankC
##      2045       741       898       976       449      1107      1035
##     TankD     TankE
##      1655      1077
```

```r
#Revenues per item
revperitem <- itemorders %>% group_by(item_name) %>% summarize(revenues = sum(revenue))
```

```
revperitem
```

```
## # A tibble: 9 x 2
##   item_name revenues
##   <fct>        <int>
## 1 GoldPack1    24400
## 2 GoldPack2    16860
## 3 GoldPack3    29040
## 4 GoldPack4    12122
## 5 TankA         4690
## 6 TankB        25340
## 7 TankC        28550
## 8 TankD        47150
## 9 TankE        59500
```

```
#Conversion Rate Per Item -> Paying players per item / Total Players
payingplayers <- table(paid$item_name)
payingplayers / 20000
```

```
##
## GoldPack1 GoldPack2 GoldPack3 GoldPack4     TankA     TankB     TankC
##   0.10225   0.03705   0.04490   0.04880   0.02245   0.05535   0.05175
##     TankD     TankE
##   0.08275   0.05385
```

```
#Create Table
x <- data.frame(payingplayers, revperitem, payingplayers / 20000)
x <- x[, c(1,2,4,6)]
colnames(x) <- c("Item Name", "Paying Players", "Revenues", "Conversion Rate")
x
```

```
##   Item Name Paying Players Revenues Conversion Rate
## 1 GoldPack1           2045    24400         0.10225
## 2 GoldPack2            741    16860         0.03705
## 3 GoldPack3            898    29040         0.04490
## 4 GoldPack4            976    12122         0.04880
## 5     TankA            449     4690         0.02245
## 6     TankB           1107    25340         0.05535
## 7     TankC           1035    28550         0.05175
## 8     TankD           1655    47150         0.08275
## 9     TankE           1077    59500         0.05385
```

Part II - Campaign Analysis

```
#Create new variable days_played by subtracting today's date by the date of player's first battle
rcamp$first_battle_dt <- as.Date(as.Date(rcamp$first_battle_dt, "%m/%d/%Y"), "%Y-%m-%d")
rcamp$days_played <- difftime(Sys.Date(),rcamp$first_battle_dt)
rcamp %>% group_by(group_player) %>% summarize(Average_Days_Played = mean(days_played))
```

```
## # A tibble: 2 x 2
##   group_player  Average_Days_Played
##   <fct>         <drtn>
## 1 control_group 307.2970 days
## 2 target_group  574.7601 days
```

- At first glance, it does seem as though the promotion is influential in making a conversion, as almost 76% of players in the target group made a transaction as opposed to only 35% of players in the control group making a transaction. However, upon inspecting the data, I noticed that the control group seemed to include more newer players while the target group included more older players.
- After further analysis, I found that on average, the target group consisted of players that have been playing for more than 2X the amount of time that the players in the control group have been playing. Due to my belief that players would be more inclined to make a transaction given a longer period of time spent playing, I would deem the analysis inconclusive.
- In order to increase the accuracy of the test, I would redo the promotion with random selection in both control and target groups and ensure that the average time played between both groups are comparable.

Data Construction Part

```
#is_payer column
players$is_payer <- 0
for(i in 1:nrow(players)){
  if((players$user_id[i] %in% orders$user_id) == TRUE){
    players$is_payer[i] <- 1
  }
  else{
    players$is_payer[i] <- 0
  }
}

#first_TankE_date column
itemorders <- merge(orders, items, by = c("item_id"))
playerorders <- merge(players,itemorders, by = c("user_id"))
#Find users that purchased Tank E
firstE <- playerorders %>% filter(item_name == "TankE")
#Sort users by purchase date and only keep the first date
firstE <- firstE[order(firstE$user_id, firstE$day_id), ]
firstE <- firstE[!duplicated(firstE$user_id), ]
firstE <- firstE[, c(1,8)]

#Add firstE column to Player data
players <- left_join(players, firstE, by = c("user_id"))
players$day_id <- as.character(players$day_id)
players$day_id[is.na(players$day_id)] <- "01/01/1900"
colnames(players)[7] <- "first_TankE_date"

head(players, 10)
```

```
##    user_id battles_cnt win_cnt total_playtime_min first_battle_dt is_payer
## 1        1        1219     646                398        2/6/2015        1
## 2        2        2113    1078                382       12/9/2015        1
## 3        3        2423    1163                551       11/7/2015        0
## 4        4        2199    1275                309      22/10/2015        1
## 5        5        2399    1343                585        9/7/2015        1
## 6        6        1228     553                549      22/10/2015        1
## 7        7        1392     807                526       2/11/2015        1
## 8        8        1157     683                384        8/9/2015        1
## 9        9        1721     809                458      15/07/2015        1
```

```
## 10      10        2078    1122               353      18/07/2015        1
##    first_TankE_date
## 1        01/01/1900
## 2        01/01/1900
## 3        01/01/1900
## 4          1/1/2016
## 5        01/01/1900
## 6        01/01/1900
## 7        01/01/1900
## 8        01/01/1900
## 9        01/01/1900
## 10       01/01/1900
```

Data Mining Part

a) What makes players spend money?

```r
#Logistic Regression model to determine variables that make players spend money
#Get rid of correlated/unecessary columns: User id, Tank E first purchase date
players1 <- players[, -c(1, 7)]
model <- glm(is_payer ~ ., data = players1, family = binomial)
summary(model)
```

```
##
## Call:
## glm(formula = is_payer ~ ., family = binomial, data = players1)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2018  -0.7542  -0.4414   0.7905   2.8198
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)                 -5.444e+00  2.776e-01 -19.608   <2e-16 ***
## battles_cnt                  1.504e-03  1.127e-04  13.353   <2e-16 ***
## win_cnt                     -5.294e-05  2.135e-04  -0.248   0.8041
## total_playtime_min           7.101e-03  1.733e-04  40.968   <2e-16 ***
## first_battle_dt1/11/2015    -3.069e-01  3.671e-01  -0.836   0.4032
## first_battle_dt1/12/2015    -3.890e-01  3.826e-01  -1.017   0.3093
## first_battle_dt1/6/2015      1.996e-01  3.626e-01   0.551   0.5820
## first_battle_dt1/7/2015     -2.261e-02  3.714e-01  -0.061   0.9515
## first_battle_dt1/8/2015      8.212e-02  3.682e-01   0.223   0.8235
## first_battle_dt1/9/2015     -2.113e-01  3.705e-01  -0.570   0.5684
## first_battle_dt10/10/2015 -5.825e-01  3.713e-01  -1.569   0.1166
## first_battle_dt10/11/2015  6.344e-02  3.769e-01   0.168   0.8663
## first_battle_dt10/12/2015 -3.246e-01  3.708e-01  -0.875   0.3813
## first_battle_dt10/6/2015    -3.002e-01  3.761e-01  -0.798   0.4247
## first_battle_dt10/7/2015    -7.040e-02  3.610e-01  -0.195   0.8454
## first_battle_dt10/8/2015    -4.533e-01  3.691e-01  -1.228   0.2194
## first_battle_dt10/9/2015    -1.753e-02  3.733e-01  -0.047   0.9625
## first_battle_dt11/10/2015 -2.914e-01  3.734e-01  -0.781   0.4351
## first_battle_dt11/11/2015  7.167e-02  3.796e-01   0.189   0.8502
## first_battle_dt11/12/2015  1.538e-01  3.831e-01   0.402   0.6880
```

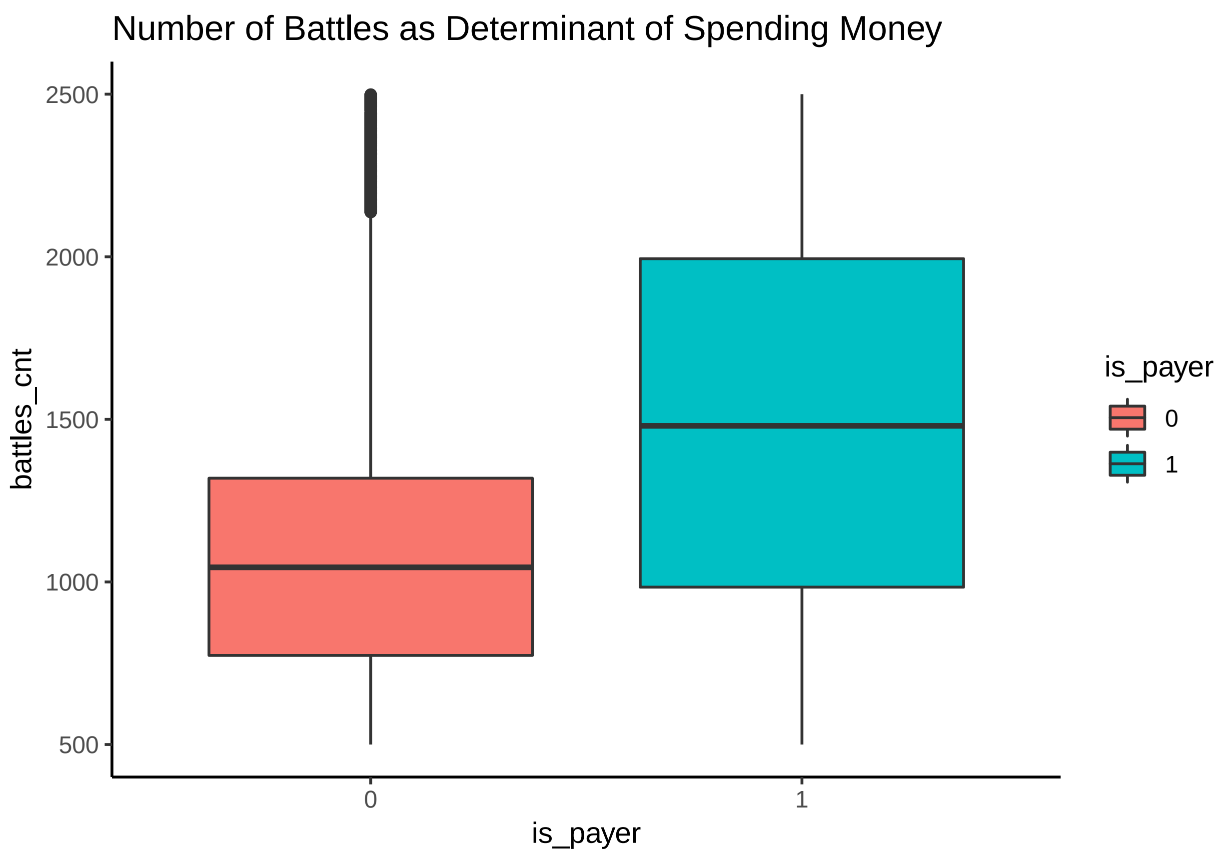
```
## first_battle_dt11/6/2015   4.418e-02  3.547e-01   0.125   0.9009
## first_battle_dt11/7/2015   1.651e-01  3.558e-01   0.464   0.6426
## first_battle_dt11/8/2015  -3.745e-01  3.616e-01  -1.036   0.3004
## first_battle_dt11/9/2015   1.392e-01  3.552e-01   0.392   0.6951
## first_battle_dt12/10/2015 -9.590e-02  3.909e-01  -0.245   0.8062
## first_battle_dt12/11/2015 -3.527e-01  3.984e-01  -0.885   0.3760
## first_battle_dt12/12/2015  1.577e-01  3.556e-01   0.443   0.6575
## first_battle_dt12/6/2015  -1.870e-01  3.818e-01  -0.490   0.6242
## first_battle_dt12/7/2015  -5.493e-01  3.972e-01  -1.383   0.1667
## first_battle_dt12/8/2015  -2.308e-01  3.674e-01  -0.628   0.5299
## first_battle_dt12/9/2015   3.952e-01  3.718e-01   1.063   0.2879
## first_battle_dt13/06/2015 -2.422e-01  3.623e-01  -0.668   0.5039
## first_battle_dt13/07/2015  9.679e-02  3.672e-01   0.264   0.7921
## first_battle_dt13/08/2015 -1.908e-02  3.745e-01  -0.051   0.9594
## first_battle_dt13/09/2015 -7.878e-02  3.800e-01  -0.207   0.8358
## first_battle_dt13/10/2015  1.514e-01  3.697e-01   0.410   0.6822
## first_battle_dt13/11/2015 -6.128e-03  3.679e-01  -0.017   0.9867
## first_battle_dt13/12/2015  1.810e-01  3.804e-01   0.476   0.6342
## first_battle_dt14/06/2015  3.386e-01  3.818e-01   0.887   0.3752
## first_battle_dt14/07/2015  8.916e-03  3.692e-01   0.024   0.9807
## first_battle_dt14/08/2015 -1.011e-01  3.768e-01  -0.268   0.7884
## first_battle_dt14/09/2015 -3.827e-01  3.838e-01  -0.997   0.3187
## first_battle_dt14/10/2015  7.950e-02  3.727e-01   0.213   0.8311
## first_battle_dt14/11/2015 -7.120e-02  3.646e-01  -0.195   0.8452
## first_battle_dt14/12/2015  1.869e-01  3.726e-01   0.502   0.6159
## first_battle_dt15/06/2015 -8.851e-02  3.481e-01  -0.254   0.7993
## first_battle_dt15/07/2015 -2.298e-02  3.735e-01  -0.062   0.9509
## first_battle_dt15/08/2015 -1.012e-01  3.686e-01  -0.275   0.7837
## first_battle_dt15/09/2015  8.348e-02  3.683e-01   0.227   0.8207
## first_battle_dt15/10/2015 -4.897e-01  3.919e-01  -1.249   0.2115
## first_battle_dt15/11/2015 -2.117e-02  3.692e-01  -0.057   0.9543
## first_battle_dt15/12/2015 -2.030e-01  3.681e-01  -0.551   0.5814
## first_battle_dt16/06/2015 -3.503e-01  3.900e-01  -0.898   0.3691
## first_battle_dt16/07/2015 -2.413e-01  3.863e-01  -0.625   0.5322
## first_battle_dt16/08/2015 -1.075e-01  3.592e-01  -0.299   0.7647
## first_battle_dt16/09/2015 -2.538e-01  3.852e-01  -0.659   0.5099
## first_battle_dt16/10/2015 -3.129e-01  3.703e-01  -0.845   0.3981
## first_battle_dt16/11/2015 -9.826e-02  3.743e-01  -0.263   0.7929
## first_battle_dt16/12/2015 -1.985e-01  3.722e-01  -0.533   0.5938
## first_battle_dt17/06/2015  1.247e-01  3.455e-01   0.361   0.7181
## first_battle_dt17/07/2015  3.210e-01  3.525e-01   0.911   0.3624
## first_battle_dt17/08/2015  8.366e-02  3.954e-01   0.212   0.8324
## first_battle_dt17/09/2015 -9.977e-02  3.727e-01  -0.268   0.7889
## first_battle_dt17/10/2015  2.061e-01  3.680e-01   0.560   0.5755
## first_battle_dt17/11/2015  3.294e-01  3.521e-01   0.936   0.3494
## first_battle_dt17/12/2015 -4.172e-01  3.728e-01  -1.119   0.2631
## first_battle_dt18/06/2015 -4.746e-03  3.763e-01  -0.013   0.9899
## first_battle_dt18/07/2015  1.340e-01  3.576e-01   0.375   0.7078
## first_battle_dt18/08/2015 -6.238e-02  3.607e-01  -0.173   0.8627
## first_battle_dt18/09/2015 -2.374e-01  3.668e-01  -0.647   0.5175
## first_battle_dt18/10/2015 -3.442e-01  3.682e-01  -0.935   0.3500
## first_battle_dt18/11/2015  1.915e-01  3.631e-01   0.527   0.5978
## first_battle_dt18/12/2015 -4.653e-01  3.852e-01  -1.208   0.2270
## first_battle_dt19/06/2015 -1.962e-01  3.769e-01  -0.521   0.6027
```

```
## first_battle_dt19/07/2015 -4.627e-01  3.744e-01  -1.236   0.2165
## first_battle_dt19/08/2015 -1.444e-02  3.699e-01  -0.039   0.9688
## first_battle_dt19/09/2015 -5.015e-03  3.568e-01  -0.014   0.9888
## first_battle_dt19/10/2015  5.771e-02  3.697e-01   0.156   0.8760
## first_battle_dt19/11/2015 -7.355e-02  3.554e-01  -0.207   0.8361
## first_battle_dt19/12/2015 -1.148e-01  3.655e-01  -0.314   0.7534
## first_battle_dt2/10/2015  -9.620e-02  3.695e-01  -0.260   0.7946
## first_battle_dt2/11/2015  -2.117e-01  3.606e-01  -0.587   0.5571
## first_battle_dt2/12/2015   4.668e-03  3.703e-01   0.013   0.9899
## first_battle_dt2/6/2015    2.432e-01  3.597e-01   0.676   0.4990
## first_battle_dt2/7/2015   -1.685e-03  3.623e-01  -0.005   0.9963
## first_battle_dt2/8/2015   -1.605e-02  3.551e-01  -0.045   0.9640
## first_battle_dt2/9/2015   -6.647e-01  3.700e-01  -1.796   0.0724 .
## first_battle_dt20/06/2015  2.569e-02  3.630e-01   0.071   0.9436
## first_battle_dt20/07/2015  2.594e-01  3.487e-01   0.744   0.4570
## first_battle_dt20/08/2015 -4.892e-01  3.907e-01  -1.252   0.2105
## first_battle_dt20/09/2015 -4.096e-01  3.772e-01  -1.086   0.2775
## first_battle_dt20/10/2015  1.991e-01  3.571e-01   0.558   0.5771
## first_battle_dt20/11/2015  8.537e-03  3.791e-01   0.023   0.9820
## first_battle_dt20/12/2015 -1.228e-01  3.785e-01  -0.324   0.7456
## first_battle_dt21/06/2015 -2.200e-01  3.581e-01  -0.614   0.5389
## first_battle_dt21/07/2015 -3.752e-01  3.822e-01  -0.982   0.3262
## first_battle_dt21/08/2015  2.522e-01  3.680e-01   0.685   0.4931
## first_battle_dt21/09/2015  3.920e-03  3.702e-01   0.011   0.9916
## first_battle_dt21/10/2015 -8.441e-02  3.616e-01  -0.233   0.8154
## first_battle_dt21/11/2015 -1.374e-01  3.627e-01  -0.379   0.7048
## first_battle_dt21/12/2015 -1.769e-01  3.727e-01  -0.475   0.6350
## first_battle_dt22/06/2015 -2.936e-01  3.677e-01  -0.799   0.4246
## first_battle_dt22/07/2015 -8.224e-02  3.746e-01  -0.220   0.8262
## first_battle_dt22/08/2015 -2.975e-01  3.721e-01  -0.800   0.4239
## first_battle_dt22/09/2015 -2.201e-01  3.806e-01  -0.578   0.5630
## first_battle_dt22/10/2015 -2.017e-01  3.811e-01  -0.529   0.5967
## first_battle_dt22/11/2015 -5.446e-01  4.010e-01  -1.358   0.1744
## first_battle_dt22/12/2015 -1.160e-01  3.626e-01  -0.320   0.7491
## first_battle_dt23/06/2015  1.810e-01  3.647e-01   0.496   0.6197
## first_battle_dt23/07/2015 -1.922e-01  3.710e-01  -0.518   0.6043
## first_battle_dt23/08/2015  5.434e-02  3.558e-01   0.153   0.8786
## first_battle_dt23/09/2015 -3.783e-02  3.801e-01  -0.100   0.9207
## first_battle_dt23/10/2015  1.326e-01  3.545e-01   0.374   0.7084
## first_battle_dt23/11/2015 -6.997e-02  3.709e-01  -0.189   0.8504
## first_battle_dt23/12/2015  1.575e-01  3.782e-01   0.416   0.6771
## first_battle_dt24/06/2015  1.058e-01  3.618e-01   0.292   0.7701
## first_battle_dt24/07/2015 -1.685e-01  3.663e-01  -0.460   0.6456
## first_battle_dt24/08/2015 -5.035e-01  3.767e-01  -1.336   0.1814
## first_battle_dt24/09/2015 -3.504e-02  3.672e-01  -0.095   0.9240
## first_battle_dt24/10/2015 -6.593e-01  3.986e-01  -1.654   0.0981 .
## first_battle_dt24/11/2015 -6.540e-01  3.943e-01  -1.659   0.0972 .
## first_battle_dt24/12/2015 -8.898e-01  3.958e-01  -2.248   0.0246 *
## first_battle_dt25/06/2015 -1.650e-01  3.735e-01  -0.442   0.6587
## first_battle_dt25/07/2015 -1.583e-01  3.603e-01  -0.439   0.6604
## first_battle_dt25/08/2015  8.533e-02  3.542e-01   0.241   0.8097
## first_battle_dt25/09/2015 -7.460e-02  3.818e-01  -0.195   0.8451
## first_battle_dt25/10/2015  2.314e-01  3.754e-01   0.617   0.5375
## first_battle_dt25/11/2015  1.244e-03  3.642e-01   0.003   0.9973
```

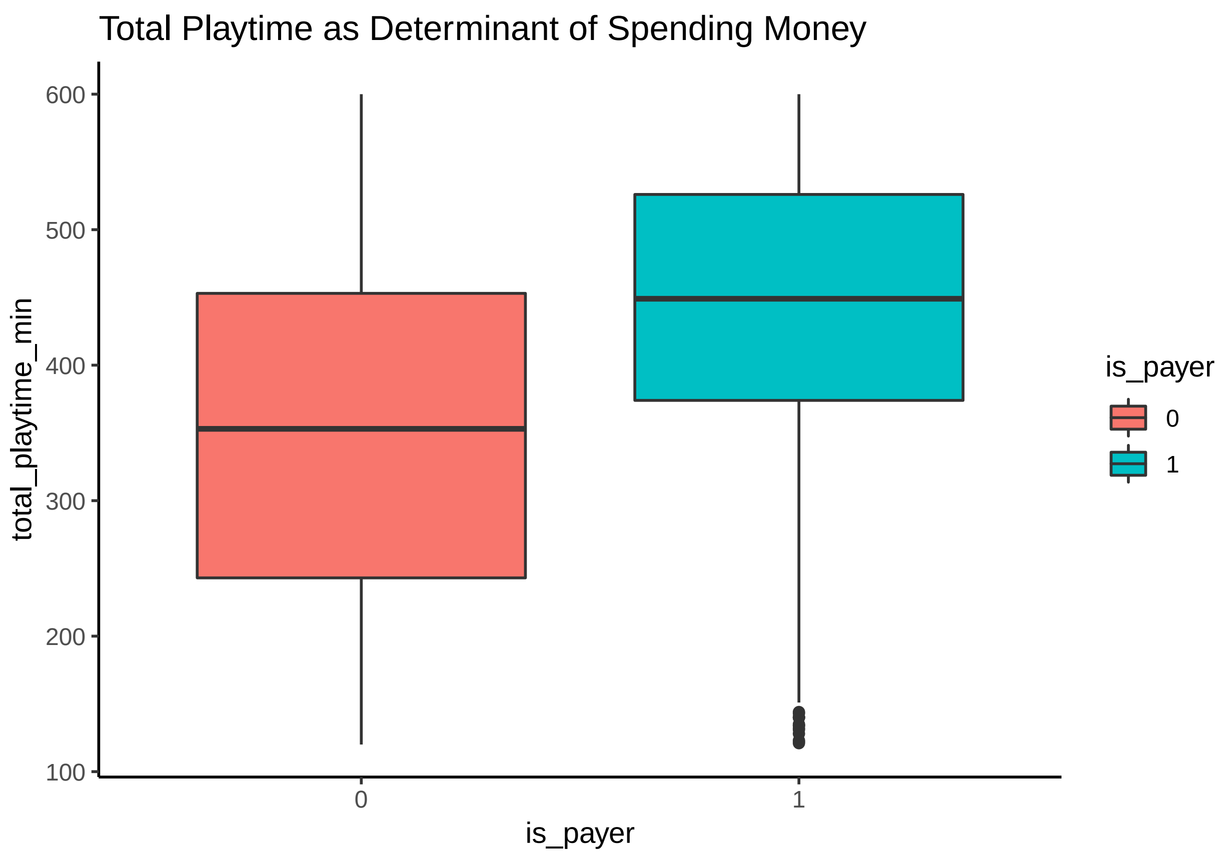
```
## first_battle_dt25/12/2015 -1.012e-01  3.651e-01  -0.277   0.7816  
## first_battle_dt26/06/2015 -7.694e-01  3.743e-01  -2.055   0.0398 *
## first_battle_dt26/07/2015 -5.410e-01  3.728e-01  -1.451   0.1466  
## first_battle_dt26/08/2015 -5.135e-02  3.612e-01  -0.142   0.8870  
## first_battle_dt26/09/2015 -3.321e-01  3.569e-01  -0.930   0.3522  
## first_battle_dt26/10/2015  2.627e-01  3.694e-01   0.711   0.4770  
## first_battle_dt26/11/2015  1.092e-01  3.565e-01   0.306   0.7594  
## first_battle_dt26/12/2015 -1.634e-02  3.822e-01  -0.043   0.9659  
## first_battle_dt27/06/2015 -2.742e-01  3.966e-01  -0.691   0.4894  
## first_battle_dt27/07/2015 -7.365e-01  3.771e-01  -1.953   0.0508 .
## first_battle_dt27/08/2015 -6.319e-01  3.951e-01  -1.599   0.1098  
## first_battle_dt27/09/2015 -2.481e-01  3.801e-01  -0.653   0.5139  
## first_battle_dt27/10/2015  2.459e-01  3.515e-01   0.699   0.4843  
## first_battle_dt27/11/2015 -9.325e-02  3.737e-01  -0.250   0.8029  
## first_battle_dt27/12/2015 -3.666e-01  3.714e-01  -0.987   0.3235  
## first_battle_dt28/06/2015 -6.828e-01  3.781e-01  -1.806   0.0710 .
## first_battle_dt28/07/2015 -2.898e-01  3.768e-01  -0.769   0.4418  
## first_battle_dt28/08/2015 -3.510e-01  3.788e-01  -0.926   0.3542  
## first_battle_dt28/09/2015 -3.551e-01  3.788e-01  -0.937   0.3485  
## first_battle_dt28/10/2015  1.956e-01  3.870e-01   0.505   0.6133  
## first_battle_dt28/11/2015 -5.485e-01  3.946e-01  -1.390   0.1645  
## first_battle_dt28/12/2015 -2.169e-01  3.697e-01  -0.587   0.5574  
## first_battle_dt29/06/2015 -2.714e-01  3.748e-01  -0.724   0.4689  
## first_battle_dt29/07/2015 -5.401e-01  4.112e-01  -1.314   0.1890  
## first_battle_dt29/08/2015 -2.629e-01  3.714e-01  -0.708   0.4791  
## first_battle_dt29/09/2015  1.539e-01  3.740e-01   0.412   0.6807  
## first_battle_dt29/10/2015  5.487e-02  3.753e-01   0.146   0.8837  
## first_battle_dt29/11/2015 -1.398e-01  3.613e-01  -0.387   0.6988  
## first_battle_dt29/12/2015 -3.422e-01  3.805e-01  -0.899   0.3685  
## first_battle_dt3/10/2015  -1.854e-01  3.634e-01  -0.510   0.6098  
## first_battle_dt3/11/2015  -8.157e-01  4.102e-01  -1.989   0.0467 *
## first_battle_dt3/12/2015  -4.452e-01  3.875e-01  -1.149   0.2506  
## first_battle_dt3/6/2015    9.185e-02  3.636e-01   0.253   0.8006  
## first_battle_dt3/7/2015   -9.447e-02  3.821e-01  -0.247   0.8048  
## first_battle_dt3/8/2015   -5.745e-01  3.832e-01  -1.499   0.1338  
## first_battle_dt3/9/2015   -6.863e-02  3.679e-01  -0.187   0.8520  
## first_battle_dt30/06/2015 -5.226e-01  3.858e-01  -1.355   0.1755  
## first_battle_dt30/07/2015 -3.493e-01  3.781e-01  -0.924   0.3557  
## first_battle_dt30/08/2015  8.657e-02  3.646e-01   0.237   0.8123  
## first_battle_dt30/09/2015 -1.944e-01  3.786e-01  -0.514   0.6076  
## first_battle_dt30/10/2015  1.759e-01  3.572e-01   0.492   0.6224  
## first_battle_dt30/11/2015 -3.560e-01  3.809e-01  -0.935   0.3500  
## first_battle_dt30/12/2015 -3.697e-01  3.763e-01  -0.982   0.3259  
## first_battle_dt31/07/2015 -7.016e-01  3.979e-01  -1.763   0.0778 .
## first_battle_dt31/08/2015  1.241e-01  3.699e-01   0.336   0.7371  
## first_battle_dt31/10/2015 -2.445e-01  3.564e-01  -0.686   0.4927  
## first_battle_dt31/12/2015 -1.807e-01  3.594e-01  -0.503   0.6151  
## first_battle_dt4/10/2015  -3.939e-01  3.798e-01  -1.037   0.2998  
## first_battle_dt4/11/2015  -5.670e-01  3.723e-01  -1.523   0.1278  
## first_battle_dt4/12/2015   1.869e-01  3.716e-01   0.503   0.6151  
## first_battle_dt4/6/2015    1.292e-02  3.589e-01   0.036   0.9713  
## first_battle_dt4/7/2015   -4.112e-01  3.732e-01  -1.102   0.2705  
## first_battle_dt4/8/2015   -1.223e-01  3.798e-01  -0.322   0.7475  
## first_battle_dt4/9/2015   -1.406e-01  3.543e-01  -0.397   0.6914  
```

```
## first_battle_dt5/10/2015  -3.008e-01  3.890e-01  -0.773   0.4393
## first_battle_dt5/11/2015  -1.506e-01  3.804e-01  -0.396   0.6921
## first_battle_dt5/12/2015  -3.545e-01  3.735e-01  -0.949   0.3426
## first_battle_dt5/6/2015   -1.314e-01  4.022e-01  -0.327   0.7438
## first_battle_dt5/7/2015   -1.189e-01  3.775e-01  -0.315   0.7528
## first_battle_dt5/8/2015   -5.360e-01  3.921e-01  -1.367   0.1716
## first_battle_dt5/9/2015   -1.867e-01  3.750e-01  -0.498   0.6187
## first_battle_dt6/10/2015   2.442e-01  3.592e-01   0.680   0.4967
## first_battle_dt6/11/2015   4.019e-02  3.619e-01   0.111   0.9116
## first_battle_dt6/12/2015   1.578e-01  3.596e-01   0.439   0.6607
## first_battle_dt6/6/2015   -4.480e-02  3.714e-01  -0.121   0.9040
## first_battle_dt6/7/2015   -8.210e-01  3.826e-01  -2.146   0.0319 *
## first_battle_dt6/8/2015    7.344e-02  3.722e-01   0.197   0.8436
## first_battle_dt6/9/2015   -4.113e-01  3.625e-01  -1.135   0.2566
## first_battle_dt7/10/2015   2.782e-02  3.639e-01   0.076   0.9391
## first_battle_dt7/11/2015   2.299e-01  3.663e-01   0.628   0.5302
## first_battle_dt7/12/2015  -4.816e-01  3.790e-01  -1.271   0.2038
## first_battle_dt7/6/2015   -5.411e-01  3.875e-01  -1.396   0.1626
## first_battle_dt7/7/2015   -2.079e-01  3.778e-01  -0.550   0.5821
## first_battle_dt7/8/2015    9.319e-02  3.806e-01   0.245   0.8066
## first_battle_dt7/9/2015   -4.091e-01  3.940e-01  -1.038   0.2991
## first_battle_dt8/10/2015  -5.977e-01  3.971e-01  -1.505   0.1323
## first_battle_dt8/11/2015  -2.312e-01  3.726e-01  -0.620   0.5350
## first_battle_dt8/12/2015  -6.216e-01  3.848e-01  -1.615   0.1063
## first_battle_dt8/6/2015   -5.754e-01  3.782e-01  -1.521   0.1282
## first_battle_dt8/7/2015    1.820e-01  3.868e-01   0.470   0.6380
## first_battle_dt8/8/2015   -1.491e-01  3.548e-01  -0.420   0.6744
## first_battle_dt8/9/2015   -1.469e-01  3.599e-01  -0.408   0.6832
## first_battle_dt9/10/2015  -1.637e-01  3.727e-01  -0.439   0.6605
## first_battle_dt9/11/2015  -5.596e-01  3.848e-01  -1.454   0.1459
## first_battle_dt9/12/2015  -2.096e-03  3.654e-01  -0.006   0.9954
## first_battle_dt9/6/2015   -1.593e-01  3.667e-01  -0.434   0.6639
## first_battle_dt9/7/2015    2.818e-03  3.928e-01   0.007   0.9943
## first_battle_dt9/8/2015    2.337e-01  3.609e-01   0.648   0.5172
## first_battle_dt9/9/2015   -3.630e-01  3.726e-01  -0.974   0.3299
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 24299  on 19999  degrees of freedom
## Residual deviance: 19244  on 19783  degrees of freedom
## AIC: 19678
##
## Number of Fisher Scoring iterations: 5
```

```
players1$is_payer <- as.factor(players1$is_payer)
battles <- ggplot(data = players1, aes(x = is_payer, y = battles_cnt, fill = is_payer)) + geom_boxplot()
battles
```




```
playtime <- ggplot(data = players1, aes(x = is_payer, y = total_playtime_min, fill = is_payer)) + geom_b
playtime
```



- From the results of the logistic model, it is clear that the number of battles and the total playtime in minutes are highly significant factors in determining whether a player spends money, with both of these variables having P-values close to 0. In the boxplots above, we see that for both variables, the average playtime and number of battles are vastly higher in spenders than in nonspenders.
- In addition, the date of first battle is also significant in determining whether a player would spend money but only on certain dates. I would chalk this significance up to coincidence, as the date when a player first starts playing should not be an indicator of whether they spend money or not. However, this relationship could be further explored in a larger dataset that shows dates of events or promotions, as perhaps the player could have started playing during an event or promotion that prompted them to spend money.

b) What makes players buy Gold Pack 4?

```
#Create variable buys_gp4 that is valued 1 if players purchased goldpack4 and 0 if not
goldpack4 <- playerorders
goldpack4$buys_gp4 <- 0
goldpack4[goldpack4$item_name == "GoldPack4", "buys_gp4"] <- 1
#Remove correlated/unecessary variables: userid, firstdateplayed, is_payer, itemid, itemname, itemdescr
goldpack4 <- goldpack4[, c(-1, -5, -6, -7, -10, -11)]

#Logistic Regression model to determine variables that make people buy Gold Pack 4
model2 <- glm(buys_gp4 ~ ., data = goldpack4, family = binomial)
summary(model2)
```

```
##
## Call:
## glm(formula = buys_gp4 ~ ., family = binomial, data = goldpack4)
```

```
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.17946  -0.25461  -0.10963  -0.00005   2.11394
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -1.346e+01  4.217e+02  -0.032    0.975
## battles_cnt        -1.688e-05  2.102e-04  -0.080    0.936
## win_cnt            -6.162e-04  4.058e-04  -1.518    0.129
## total_playtime_min -2.299e-03  3.957e-04  -5.810 6.26e-09 ***
## day_id10/1/2016     1.665e+01  4.217e+02   0.039    0.969
## day_id11/1/2016     1.691e+01  4.217e+02   0.040    0.968
## day_id12/1/2016     1.713e+01  4.217e+02   0.041    0.968
## day_id13/01/2016    1.695e+01  4.217e+02   0.040    0.968
## day_id14/01/2016    1.698e+01  4.217e+02   0.040    0.968
## day_id15/01/2016    1.703e+01  4.217e+02   0.040    0.968
## day_id16/01/2016    1.680e+01  4.217e+02   0.040    0.968
## day_id17/01/2016    1.688e+01  4.217e+02   0.040    0.968
## day_id18/01/2016    1.691e+01  4.217e+02   0.040    0.968
## day_id19/01/2016    1.667e+01  4.217e+02   0.040    0.968
## day_id2/1/2016      1.803e-01  5.841e+02   0.000    1.000
## day_id20/01/2016    1.668e+01  4.217e+02   0.040    0.968
## day_id21/01/2016    1.664e+01  4.217e+02   0.039    0.969
## day_id22/01/2016    1.679e+01  4.217e+02   0.040    0.968
## day_id23/01/2016    1.680e+01  4.217e+02   0.040    0.968
## day_id24/01/2016    1.662e+01  4.217e+02   0.039    0.969
## day_id25/01/2016    1.704e+01  4.217e+02   0.040    0.968
## day_id26/01/2016    1.675e+01  4.217e+02   0.040    0.968
## day_id27/01/2016    1.691e+01  4.217e+02   0.040    0.968
## day_id28/01/2016    1.698e+01  4.217e+02   0.040    0.968
## day_id29/01/2016    1.663e+01  4.217e+02   0.039    0.969
## day_id3/1/2016      3.394e-01  5.903e+02   0.001    1.000
## day_id30/01/2016    1.658e+01  4.217e+02   0.039    0.969
## day_id31/01/2016    1.697e+01  4.217e+02   0.040    0.968
## day_id4/1/2016      2.036e-02  5.865e+02   0.000    1.000
## day_id5/1/2016      5.542e-02  5.955e+02   0.000    1.000
## day_id6/1/2016      2.271e-01  5.871e+02   0.000    1.000
## day_id7/1/2016      1.643e+01  4.217e+02   0.039    0.969
## day_id8/1/2016      1.672e+01  4.217e+02   0.040    0.968
## day_id9/1/2016      1.683e+01  4.217e+02   0.040    0.968
## item_cnt           -4.723e-05  4.430e-04  -0.107    0.915
## item_price         -2.730e-01  1.135e-02 -24.056  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6702.2  on 11000  degrees of freedom
## Residual deviance: 4506.4  on 10965  degrees of freedom
## AIC: 4578.4
##
## Number of Fisher Scoring iterations: 18
```

- The two variables that determine what makes players buy Gold Pack 4 are Total Playtime in Minutes and Item Price, with P-Values close 0, making these variables highly significant. With a p-value of 0.129, win count is also a factor that can be looked at. Interestingly, despite the similarity between the number of battles and total playtime in minutes, the number of battles was far from being a predictor in determining what makes players buy Gold Pack 4, with a p-value of 0.9375. The date of purchase was also not significant at all for any date during this period.

SQL Part

SELECT DISTINCT USER_ID, MAX(PROFILE_DT), FAVORITE_CLASS FROM user_profile WHERE FAVORITE_CLASS = “Medium” Group By USER_ID Order By USER_ID, MAX(PROFILE_DT) DESC