Warframe

EKR

2023-05-11

Warframe is an online game available on all consoles. It allows 1-4 players to play as the titular 'Warframes'(colloquially named 'space ninjas' or simply 'Frames'), blazing through hordes of enemies from multiple factions, on multiple planets with an assortment of over 1000 weapons, and devastating abilities.

Warframe is a rather unique game as it is entirely *free to play*(FTP), paying money can help skip some of the grind, but even that can be done in game for free via trading with other players. In fact the only items unobtainable to FTP players are a small handful of cosmetics. Despite the minimal monetization, Digital Extremes have managed to keep this game relevant and profitable for 10 years.

Research Question:

My aim is investigate whether there is an association between the active monthly playercount in the game Warframe and the release of new 'frames'(Classes) within the game.

Data Origins:

Using **rvest** and the selectorgadget chrome extension, I have managed to scrape monthly playercount data from steamcharts to create the following dataframe:

```
#designating the URL to scrape from:

link = "https://steamcharts.com/app/230410"
page = read_html(link)

#Converting webpage tags into variable lists:

date = page %>% html_nodes(".left") %>%  html_text()
avg = page %>% html_nodes(".left+ .right") %>%  html_text()
gain = page %>% html_nodes(".right:nth-child(3)") %>%  html_text()
pctgain = page %>% html_nodes(".right:nth-child(4)") %>%  html_text()
peak = page %>% html_nodes(".right:nth-child(5)") %>%  html_text()

#combining lists as columns to create the 'playercount' dataframe:

playercount = data.frame(date, avg, gain, pctgain, peak, stringsAsFactors = FALSE)

#Here is a sample of the raw playercount dataframe:
head(playercount)
```

```
##                                 date          avg     gain pctgain
```

```
## 1                                Month Avg. Players     Gain  % Gain
## 2                          Last 30 Days     53405.70  +5050.6 +10.44%
## 3    \n\t\t\t\t\tApril 2023\n\t\t\t\t     48355.07  7648.58 +18.79%
## 4    \n\t\t\t\t\tMarch 2023\n\t\t\t\t     40706.49 -1223.89  -2.92%
## 5 \n\t\t\t\t\tFebruary 2023\n\t\t\t\t     41930.39  1259.47  +3.10%
## 6  \n\t\t\t\t\tJanuary 2023\n\t\t\t\t     40670.91  1568.74  +4.01%
##           peak
## 1 Peak Players
## 2       115322
## 3       115322
## 4        63635
## 5        59850
## 6        58816
```

```
#Creating an archived backup CSV of data frame in case URL is compromised:
#write.csv(playercount,"CSVs\\playercount.csv")

#If URL does get compromised the following code will retrieve the archived dataframe from 09/05/23:

#playercount <- read_csv("CSVs/playercount.csv")
```

I was able to do this by stating the relevant URL for the website -in which the data was held-, identifying the relevant website tags -via selectorgadget- for each desired dataframe column and running a script to convert the html nodes (tags) into string variables. I then combined the variables as columns within the dataframe.

I repeated this process for the dataframe below depicting the release dates of new frames (classes) within Warframe:

```
##            frame           release
## 1       Warframe      Release Date
## 2   Baruuk Prime  14 December 2022
## 3         Voruna  30 November 2022
## 4 Revenant Prime   05 October 2022
## 5        Styanax 07 September 2022
## 6    Khora Prime      16 July 2022
```

Data Preparation:

I found the trickiest aspect of data wrangling to be the transformation of dates. for both dataframes, I separated the individual aspects of the date column, created a new variable list assigning the numerical value 1-12 to the relevant months and then ran a script to add a new column to each dataframe containing the relevant numerical value for each month. I then ran a script to combine the month and year values for each dataframe into a single date column which was now identical across dataframes, allowing for dataframe merging later. I then reformatted the columns of my dataframes to just relevant data for ease of use:

```
#splitting dates:
framerel <-  framerel %>% separate(release, c('Day', 'Month', 'Year'))

playercount <- playercount %>% separate(date, c('pmonth', 'pyear'))

#converting month names to factor:
playercount[, 1] <- sapply(playercount[, 1], as.factor)
```

```
framerel[, 3] <-  sapply(framerel[, 3], as.factor)

#adding new column for month number:
months_number<-list(January=1,February=2,March=3,April=4,May=5,June=6, July=7,August=8,September=9,Octo

playercount <- playercount %>% mutate(pmonthnum=recode(pmonth,!!!months_number))

framerel <- framerel %>% mutate(monthnum=recode(Month,!!!months_number))

#converting variables to numeric (and tidying):
playercount <- playercount %>%
  mutate_at("pctgain", str_replace, "%", "")

playercount[, c(2:7)] <- sapply(playercount[, c(2:7)], as.numeric)

#recombining dates:
playercount$date <- as.yearmon(paste(playercount$pmonthnum, playercount$pyear), "%m %Y")

framerel$date <- as.yearmon(paste(framerel$monthnum, framerel$Year), "%m %Y")

#removing non-relevant columns & reordering column order:
playercount <- playercount[, c(8, 6, 3)]

framerel <- framerel[, c(6,1)]

#Should now have 2 organised dataframes:
```

```
##       date   peak      avg
## 3 Apr 2023 115322 48355.07
## 4 Mar 2023  63635 40706.49
## 5 Feb 2023  59850 41930.39
## 6 Jan 2023  58816 40670.91
## 7 Dec 2022  64073 39102.17
## 8 Nov 2022  58261 29968.36
```

```
##       date          frame
## 2 Dec 2022   Baruuk Prime
## 3 Nov 2022         Voruna
## 4 Oct 2022 Revenant Prime
## 5 Sep 2022        Styanax
## 6 Jul 2022    Khora Prime
## 7 Apr 2022           Gyre
```

Combining datasets: I ran a code to merge my datasets, using the - now identically formatted - dates as row references. I included the additional argument "all.x = TRUE" as the framerel dataframe included repeats of months where multiple frames had been released in the same month. I created a line of code using "ifelse(frame!="NA" " to create a new column within the merged dataframe containing the values of the peak column, only for rows that contained the release of a frame. This new column is used later to add geom points to the graph, depicting when frames were released. In this state, geom points would overlap in instances where multiple frames were released in the same month. After experimenting with different options- including the use of geom_growth instead of point, and the use of jitter-, I opted to run a 'For Loop' to add a value of 5000 to any value within the frpeak (geom point) column that had the same value as the previous row. This allowed for all geom points to be visible.

```
#Create combined dataframe:

merged <- merge(playercount, framerel, by.x = "date", all.x = TRUE)
#adding column to allow for geom points overlaying only the relevent timepoints:
merged$frpeak <- with(merged, ifelse(frame!="NA", peak))

#for loop to add 5000 to Y value for geom points if y value is equal:
for (i in 2:nrow(merged)) {if (!is.na(merged$frpeak[i]) & !is.na(merged$frpeak[i-1]) & merged$frpeak[i]
```

I then implemented a custom image for my geom points by taking an image via URL and adding it to all rows as a new column in the dataframe.

```
#adding image column for custom points:

#creating variable list of image URL repeated 138 times (no. of merged dataframe rows).
image <- rep("https://warframe-school.com/wp-content/uploads/2018/07/Warframe-Logo.png", times=138)

#merging the image column to the merged dataframe:
merged <- cbind(merged, image)


head(merged)
```

```
##         date  peak      avg   frame frpeak merged_date
## 1 2013-01-01     0     0.00   Frost      0    Jan 2013
## 2 2013-01-01     0     0.00     Nyx   5000    Jan 2013
## 3 2013-02-01     1     0.00    <NA>     NA    Feb 2013
## 4 2013-03-01 19099  1960.13   Saryn  19099    Mar 2013
## 5 2013-03-01 19099  1960.13 Banshee  24099    Mar 2013
## 6 2013-04-01 21157 12329.65    <NA>     NA    Apr 2013
##                                                                      image
## 1 https://warframe-school.com/wp-content/uploads/2018/07/Warframe-Logo.png
## 2 https://warframe-school.com/wp-content/uploads/2018/07/Warframe-Logo.png
## 3 https://warframe-school.com/wp-content/uploads/2018/07/Warframe-Logo.png
## 4 https://warframe-school.com/wp-content/uploads/2018/07/Warframe-Logo.png
## 5 https://warframe-school.com/wp-content/uploads/2018/07/Warframe-Logo.png
## 6 https://warframe-school.com/wp-content/uploads/2018/07/Warframe-Logo.png
```

I downloaded a custom font into the project folder to use for the axis labels and title:

```
font_add(family = "Bruno_SC_Ace", "Fonts\\BrunoAceSC-Regular.ttf")

showtext_auto()
#adding this prevents microscopic font sizes on axis when exporting plots via ggsave:
showtext_opts(dpi=300)
```

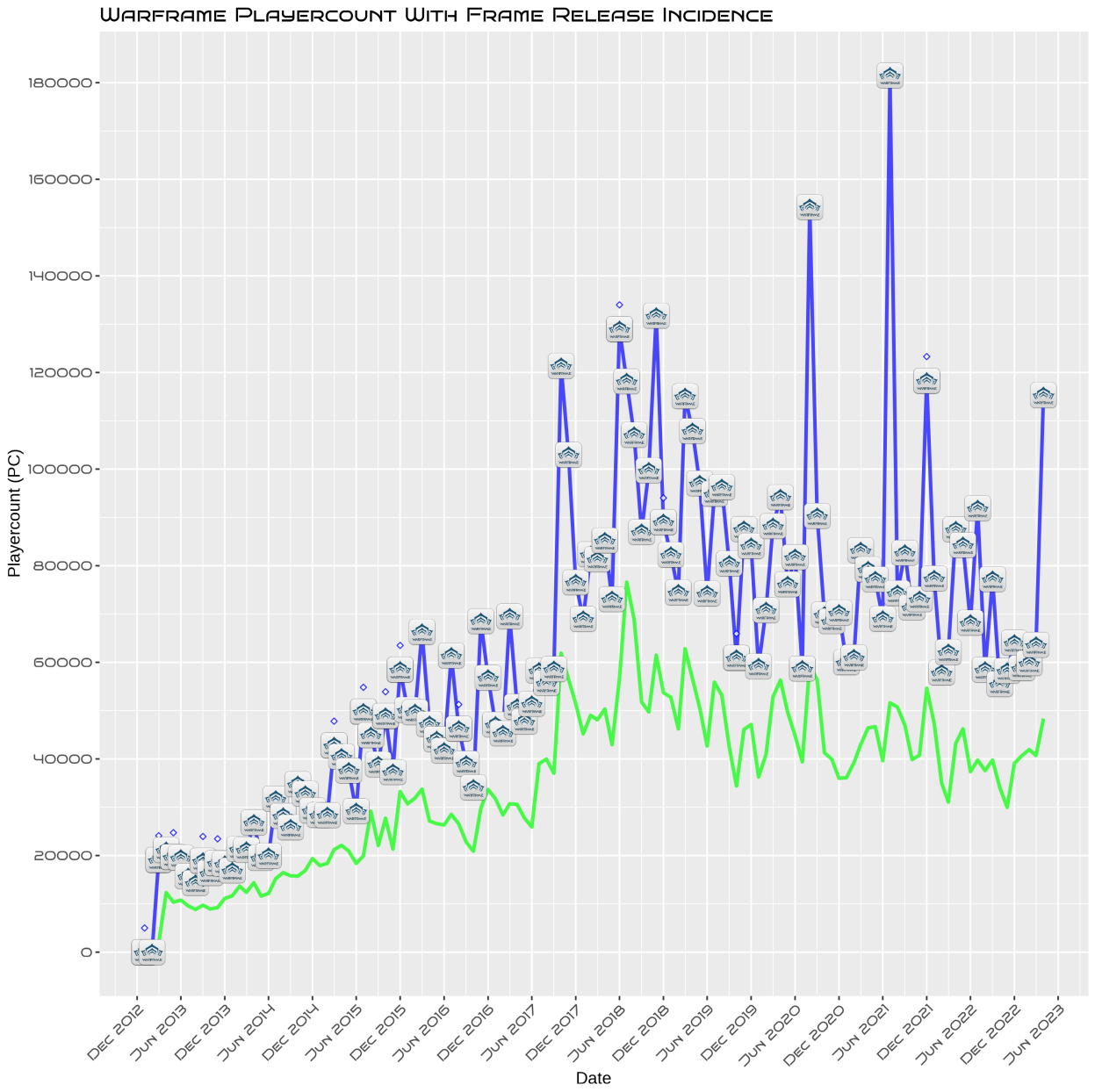
The woes of plotly customization:

By adding a line of code to the graph code (n<- m + geom_image(aes(image=image), size= 0.03)), I was able to use the custom image as points for a ggplot graph. However, when attemping to use this with ggplotly, it was revealed that this had not yet been added to the plotly package.

I then ran into the exact same problem for the use of custom text. There is supposedly a way to allow for custom text with plotly, however this requires the custom text to have been pre-downloaded onto the

computer (not just within project) running the code - crippling code replication and portability-. I found a comment on This page, by cttnguyen, detailing a workaround, however the required code for this fix was perplexing so I avoided it.

This is my what my plot could have been if plotly was compatible with custom fonts and geom images. Due to the lack of interaction, and legibility issues with the custom geom image, I decided it was less informative than an interactive plot:



Final Graph:

Below is my final plot, demonstrating the trends of playercount over the 10 year period in which Warframe has been released. The geom points demonstrate when new frames were released and provide information when hovered over.

```
#defining plot basis & indicating text for ggplotly hover text box:
g <- ggplot(data = merged, aes(x = date, group=1,
  text = paste("Date: ", merged_date, "<br>Peak: ", peak, "<br>Average: ", avg))) +
  #line for peak daily playercount, averaged over a month:
   geom_line(size=1.1, alpha=0.7,aes(y = peak, colour = "Peak", group= 1))+
  #line for mean average daily playercount, averaged over a month:
   geom_line(size=1.1, alpha=0.9, aes(y= avg, colour = "Average", group= 1)) +
  #points to depict frame release aong the peak geom_line. Shape and colours customized:
   geom_point(shape=23,  fill="white", alpha=0.8, size= 1.2, aes(x=date, y=frpeak,colour= "New Frame Re
  #adding altered text for ggplotly hover text box with name of released frame:
  text = paste("Date: ", merged_date, "<br>Peak: ", peak, "<br>Average: ", avg, "<br>New Warframe:", fra
  #using scale colour manual to select line colours and add legend to graph:
  scale_colour_manual("", values = c("Average"="deeppink", "Peak"="blue", "New Frame Released"="blue2"))

#adding labels and scaling:

  #expanding limits to use playercount df date column (no repeated values):
g<- g + expand_limits(x = playercount$date) +
  #breaking x axis into 6 month intervals:
  scale_x_date(date_breaks = "6 months", date_labels = "%b %Y")+
  #adjusting label angle:
  theme(axis.text.x = element_text(angle = 45, hjust = 1))+
  #breaking y axis into 20,000 (minor intervals every 10,000 are not compatible with plotly):
  scale_y_continuous(breaks = round(seq(min(merged$avg), max(merged$peak), by = 10000))) +
  #adding title:
  labs(title= "Warframe Playercount With Frame Release Incidence", x= "Date", y= "Playercount (PC)")
#making background a softer blue:
g<- g + theme(panel.background = element_rect(fill="azure2"))
#completed graph as interactive plotly:
ggplotly(g, tooltip = "text")
```

I believe this graph demonstrates that the release of a new frame is definitely associated with an increase in playercount averages, with the majority of geom points landing either at the top of a line summit or just before an increase in playerbase. However, the non uniform trend in playerbase indicates multiple other factors that interact with playercount. This is expected as new frames are not the only content released to the game. Other content that likely affects playercount includes reworks of existing frames, major/minor gameplay expansions such as new storyline quests, game modes, etc, and real world events such as the Covid-19 lockdowns and Warframes annual convention: Tennocon.

Summary:

Whilst I am still disappointed that I cannot use the custom font (that I spent way too long choosing), or custom geom images with ggplotly, I am still happy with my graph. I believe it provides an intuitive visualization to answer my research question.

If I had more time, I would love to find a way to include playercount numbers from other devices such as consoles (Xbox, Playstation, etc). I would have also liked the time to add other factors that may influence playercount such as expansions (as discussed above). I also attempted to utilise RENV as a package manager, however ran into a plethora of problems on short notice, so disabaed it. It would be nice if I could get that to work in the future.

I am very happy with what I have learnt through this course. Prior to this module, coding seemed to be an entirely new language that I would never be able to learn. Now I am proud to say I've used one coding language and definitely want to learn and make more with it or another code language.

codebook:

```
## playercount
##
##  3  Variables      124  Observations
## --------------------------------------------------------------------------------
## date : Date
##          n    missing   distinct       Info       Mean        Gmd        .05
##        124          0        124          1 2018-02-14       1268 2013-07-05
##        .10        .25        .50        .75        .90        .95
## 2014-01-10 2015-07-24 2018-02-15 2020-09-08 2022-03-22 2022-09-26
##
## lowest : 2013-01-01 2013-02-01 2013-03-01 2013-04-01 2013-05-01
## highest: 2022-12-01 2023-01-01 2023-02-01 2023-03-01 2023-04-01
## --------------------------------------------------------------------------------
## peak : Highest number of players online at the same time each day, averaged monthly
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      124        0      124        1    62327    35527    18302    19796
##      .25      .50      .75      .90      .95
##    39004    60475    81915    98935   117761
##
## lowest :      0      1  14349  15714  16381, highest: 121377 129002 131766 154246 181509
## --------------------------------------------------------------------------------
## avg : Number of players online at the same time, averaged daily, then monthly
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      124        0      123        1    34731    18466     9622    11807
##      .25      .50      .75      .90      .95
##    21308    36646    46755    54374    56616
##
## lowest : 0       1960.13 8831.78 8937.68 9220.75
## highest: 61486.6 61908.8 62800.5 68891.7 76594.9
## --------------------------------------------------------------------------------

## framerel
##
##  2  Variables      90  Observations
## --------------------------------------------------------------------------------
## date : Month when a new frame (class) was released
##        n  missing distinct
##       90        0       69
##
## lowest : Oct 2012 Dec 2012 Jan 2013 Mar 2013 May 2013
## highest: Jul 2022 Sep 2022 Oct 2022 Nov 2022 Dec 2022
## --------------------------------------------------------------------------------
## frame : Name of frame released
##        n  missing distinct
##       90        0       90
##
## lowest : Ash          Ash Prime    Atlas        Atlas Prime  Banshee
## highest: Wukong Prime Xaku         Yareli       Zephyr       Zephyr Prime
## --------------------------------------------------------------------------------

## merged
```

```
##
##  7  Variables      138  Observations
## ---------------------------------------------------------------------------
## date : Dates (repeated if more than 1 frame was released in 1 month)
##          n    missing   distinct       Info       Mean        Gmd        .05
##        138          0        124          1 2017-11-30       1281 2013-05-01
##        .10        .25        .50        .75        .90        .95
## 2013-10-22 2015-05-08 2017-11-16 2020-06-23 2022-02-09 2022-09-05
##
## lowest : 2013-01-01 2013-02-01 2013-03-01 2013-04-01 2013-05-01
## highest: 2022-12-01 2023-01-01 2023-02-01 2023-03-01 2023-04-01
## ---------------------------------------------------------------------------
## peak : Highest number of players online at the same time each day, averaged monthly
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      138        0      124        1    61220    36419    18072    19405
##      .25      .50      .75      .90      .95
##    37922    59001    81683   100704   118297
##
## lowest :      0      1  14349  15714  16381, highest: 121377 129002 131766 154246 181509
## ---------------------------------------------------------------------------
## avg : Number of players online at the same time, averaged daily, then monthly
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      138        0      123        1    33812    19060     9178    10644
##      .25      .50      .75      .90      .95
##    20899    34718    46617    54651    56623
##
## lowest : 0       1960.13 8831.78 8937.68 9220.75
## highest: 61486.6 61908.8 62800.5 68891.7 76594.9
## ---------------------------------------------------------------------------
## frame : Name of frame released
##        n  missing distinct
##       81       57       81
##
## lowest : Ash Prime     Atlas         Atlas Prime   Banshee       Banshee Prime
## highest: Wukong Prime  Xaku          Yareli        Zephyr        Zephyr Prime
## ---------------------------------------------------------------------------
## frpeak : peak value from column 2, but only if a frame was released that month, if more than 1 frame
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       81       57       81        1    63792    40501    18459    19939
##      .25      .50      .75      .90      .95
##    34785    58719    85227   115102   129002
##
## lowest :      0   5000  15714  18274  18459, highest: 129002 131766 134002 154246 181509
## ---------------------------------------------------------------------------
## merged_date : date in aesthetic form
##        n  missing distinct
##      138        0      124
##
## lowest : Apr 2013 Apr 2014 Apr 2015 Apr 2016 Apr 2017
## highest: Sep 2018 Sep 2019 Sep 2020 Sep 2021 Sep 2022
## ---------------------------------------------------------------------------
## image : URL for custom geom image
##                                                                          n
##                                                                        138
```

```
##                                                                      missing
##                                                                            0
##                                                                     distinct
##                                                                            1
##                                                                        value
## https://warframe-school.com/wp-content/uploads/2018/07/Warframe-Logo.png
##
## Value      https://warframe-school.com/wp-content/uploads/2018/07/Warframe-Logo.png
## Frequency                                                                138
## Proportion                                                               1
## ---------------------------------------------------------------------------
```