Econ 143 Final Project -- TFT

Reqs: (refer to Notes)

1. Figures, Maps, Summary Tables
2. 2-3 Displays of summary or descriptive statistics (cross-tabs, stats: mean, med, std dev, min, max)
3. 2-3 figures (plots and dist.): Histograms game Duration, Box plots by Rank, violin plots of game duration by synergies, Facet Grid Quant. Reg. (important; subsets of diff ranks, quantiles of different Levels)

Research Project Background

Teamfight Tactics, or commonly known as TFT, is a popular auto-battler/autochess game that boasts 33 million players worldwide every month. Developed by gaming giant, Riot Games, players of TFT use adaptive strategies in creating their own team compositions to compete against others in a lobby of eight throughout a series of rounds. Players have to carefully maximize their gold economy each round in order to level up both themselves and their in-game units in order to create compositions with traits and synergies that boost their teams while fighting other teams. Players must also effectively maximize the utility of randomly dropped items to place onto their units in order to beat the competition and come out on top.

Project Context

In this game, there is a ranked-ladder system consisting of 9 ranks, but in our project, we focus on the top 5 which are: Platinum, Diamond, Master, Grandmaster, and Challenger (in ascending order). We focus on the top 5 ranks because in the more competitive gameplay in elo-based matchmakings, we can observe the most optimal gameplay where even the slightest differences in the min-maxing of units, synergies, game economics, and items determine the placement of player.

In-game loot drops (items) and unit shops are randomized each game, introducing a significant element of unpredictability and variation, ensuring no two games are alike.

The intuition behind this research project is that in high-elo matchmaking, randomness paradoxically increases the skill ceiling, forcing high-elo/ranked players to dynamically adapt to the RNG elements in their strategic planning and

decisions when faced with probabibilistic outcomes. The best players distinguish themselves throught their rank by their ability to consistently perform well across a wide range of scenarios, proving their strategic depth and adaptability in balancing luck and skill. Thus, lower skilled players who are stuck in their mid-tier ranks will fail to (as) consistently take control of game mechanics when facing RNG elements compared to the more skilled.

The ascending order of rank, and therefore skill, in TFT goes: Platinum > Diamond > Master > GrandMaster > Challenger. But for our analysis we will only look at Platinum and Challenger to see if there is a significant difference between mid-tier skilled players and the most skilled players.

What Are We Doing?

In the lobbies of 8, placement is important, as the top 4 placed will stand to gain LP which are points added to their rank that help the player ascend into the next rank, whereas the bottom 4 lose LP, which can jeapordize their rank as they can stand to demote into the rank below. Now, in compliance with our intuition, I believe that Challenger (best) players are the most efficient and optimally adapt to all elements of RNG in the game, therefore SHOULD have higher in-game duration times than Platinum (mid-tier) players due to the inconsistency of Platinum players not knowing how to mitigate losses and capitalize on bonuses in regards to RNG elements. To test this, I will be performing regression analyses on in-game durations of players in Platinum matchmaking vs Challenger matchmaking, taking into account each individual placement in the game. In order to show if the durations are statistically significant, I will be accounting for the interaction between the distributions of durations in each of the placements and Rank. Furthermore, I will be performing quantile regression analyses on the different quantiles in these distributions to see whether there is a difference between the best performance of games to worst performance of games and how those Durations differ in Platinum and Challenger matches.

First, we should load our data and then visualize the distribution of Durations between ranks to get a grasp of what our data looks like and what we need to do to clean it up.

In [2]:
```
import pandas as pd

# load Datasets and use Concat

#define file csv names
files = [('TFT_Challenger_MatchData.csv','Challenger'),
         ('TFT_Platinum_MatchData.csv','Platinum')]
dataframes = []
```

```
for file, rank in files:
    df = pd.read_csv(file)
    df['Rank'] = rank
    dataframes.append(df)

combined_df = pd.concat(dataframes)
print(combined_df.head())
print(combined_df.info())
print(combined_df['Rank'].value_counts())
```

```
          gameId  gameDuration  level  lastRound  Ranked  ingameDuration  \
0  KR_4247538593   2142.470703      8         35       1     2134.272217
1  KR_4247538593   2142.470703      9         35       2     2134.272217
2  KR_4247538593   2142.470703      8         34       3     2073.459229
3  KR_4247538593   2142.470703      8         33       4     1998.146729
4  KR_4247538593   2142.470703      9         33       5     1986.443237

                                         combination  \
0  {'DarkStar': 2, 'Protector': 4, 'Rebel': 1, 'S...
1  {'Blaster': 2, 'Mercenary': 1, 'Rebel': 6, 'Se...
2  {'Cybernetic': 1, 'DarkStar': 3, 'Demolitionis...
3  {'Blaster': 1, 'Cybernetic': 1, 'DarkStar': 1,...
4  {'Blaster': 2, 'Demolitionist': 2, 'Mercenary'...

                                            champion        Rank
0  {'JarvanIV': {'items': [27], 'star': 3}, 'Sona...  Challenger
1  {'Malphite': {'items': [7], 'star': 2}, 'Yasuo...  Challenger
2  {'KaiSa': {'items': [99, 2, 23], 'star': 2}, '...  Challenger
3  {'KaiSa': {'items': [44, 37], 'star': 2}, 'Ann...  Challenger
4  {'Ziggs': {'items': [], 'star': 1}, 'Yasuo': {...  Challenger
<class 'pandas.core.frame.DataFrame'>
Index: 159999 entries, 0 to 79999
Data columns (total 9 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   gameId          159999 non-null  object
 1   gameDuration    159999 non-null  float64
 2   level           159999 non-null  int64
 3   lastRound       159999 non-null  int64
 4   Ranked          159999 non-null  int64
 5   ingameDuration  159999 non-null  float64
 6   combination     159999 non-null  object
 7   champion        159999 non-null  object
 8   Rank            159999 non-null  object
dtypes: float64(2), int64(3), object(4)
memory usage: 12.2+ MB
None
Rank
Platinum      80000
Challenger    79999
Name: count, dtype: int64
```

Data cleansing and Data Engineering

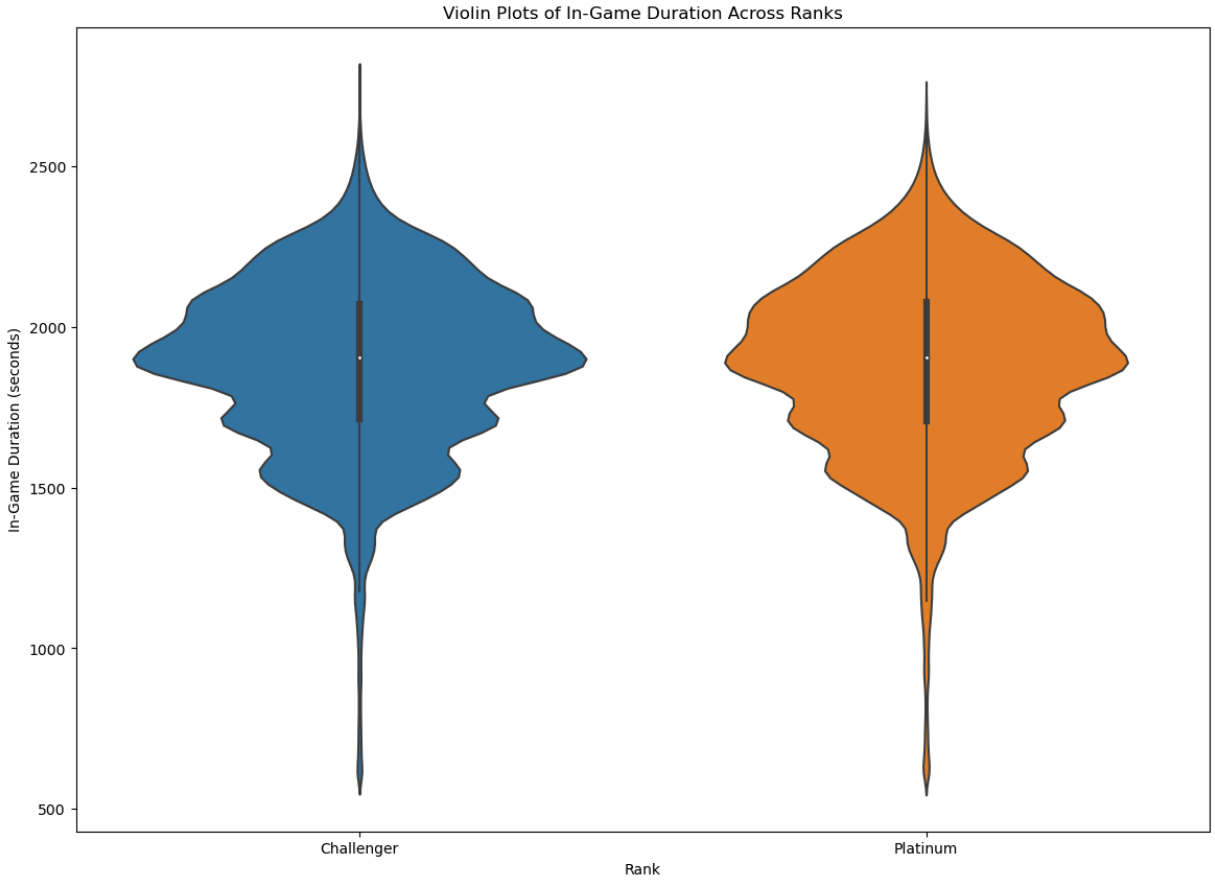
In [3]:
```
#Clean data
#Found Ranked=0 values. doesnt make sense placements are 1-8
#Looked at OG scatter; showed a lot of low 1st place durations and same inGa

combined_df = combined_df[combined_df['Ranked'] != 0]
duration_std = combined_df.groupby('gameId')['ingameDuration'].transform('st
combined_df['duration_std'] = duration_std

#Only want games where there is at least 100 second difference between place
filtered_df = combined_df[combined_df['duration_std'] > 100]
```

In [4]:
```
#Verifying dataframe changes when removing Ranked = 0 values AND disconnects
print("Original entries:", len(combined_df))
print("Filtered entries:", len(filtered_df))
```

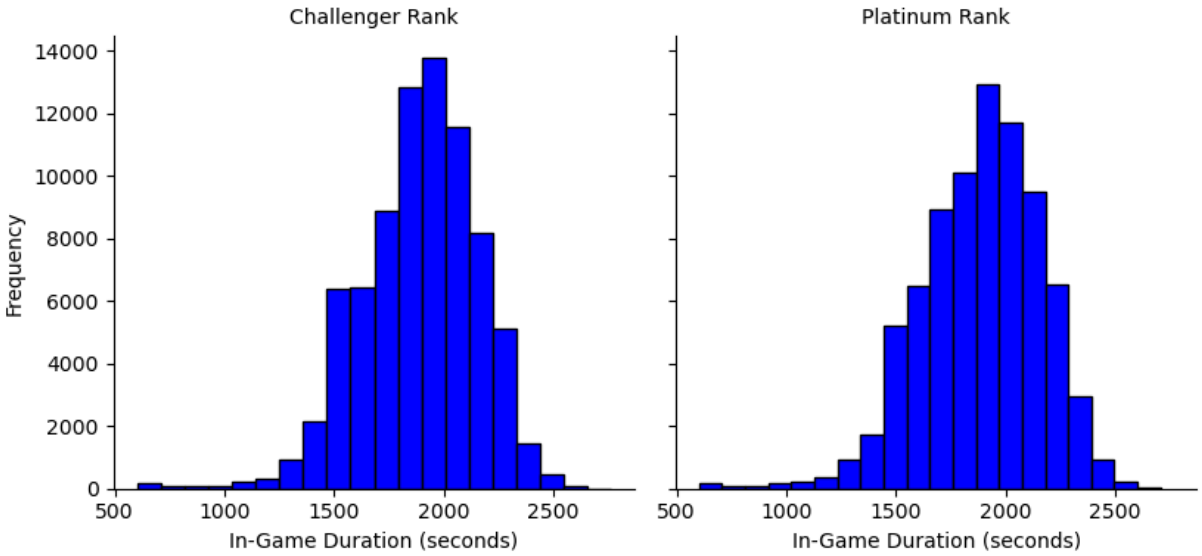
```
Original entries: 159928
Filtered entries: 158587
```

We can see that there were indeed a significant amount of data entries that did not belong in our analysis because we do not want any 'Ranked' = 0 values as that does not make sense with placements being #1-#8. Also maintenance happens weekly and those disconnected players will have similar if not the same in-game duration, so we do not want to count those either.

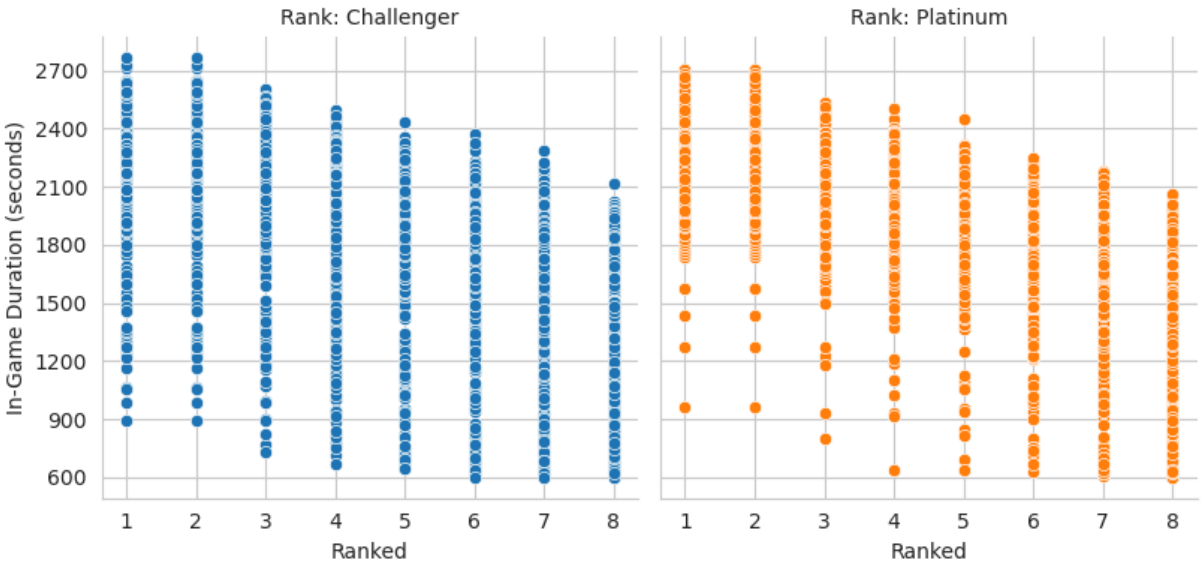
In [5]:
```
# Visualize ingameDuration as ViolinPlots to see distribution in different f
import seaborn as sns
import matplotlib.pyplot as plt
plt.figure(figsize=(14, 10))
sns.violinplot(x='Rank', y='ingameDuration', data=filtered_df)
plt.title('Violin Plots of In-Game Duration Across Ranks')
plt.xlabel('Rank')
plt.ylabel('In-Game Duration (seconds)')
plt.show()
```



In [6]:
```
#histograms in facet grid form
g_hist = sns.FacetGrid(filtered_df, col="Rank", col_wrap=3, height=4,hue = '

# Histogram to the grid for the 'ingameDuration' variable
g_hist.map_dataframe(plt.hist, 'ingameDuration', bins=20, color='b', edgecol

# Title
g_hist.set_titles("{col_name} Rank")

# Label
g_hist.set_axis_labels('In-Game Duration (seconds)', 'Frequency')
plt.show()
```



In [7]:
```
#Scatter based on Ranked place in match

sns.set_style('whitegrid')
grid_scat = sns.FacetGrid(filtered_df, col = 'Rank',col_wrap =3, height = 4,
y_min = 0
y_max = 2700 #Set max a little higher to show that upper tails of 1st and 2r

#Initiate For Loop to create both scatterplots (Ranked vs inGameDuration)
for ax in grid_scat.axes.flat:
    ax.set_yticks(range(y_min, y_max + 1, 300))
grid_scat.map_dataframe(sns.scatterplot, 'Ranked', 'ingameDuration')

grid_scat.set_titles('Rank: {col_name}')
grid_scat.set_axis_labels('Ranked','In-Game Duration (seconds)')

plt.show()
```



Descriptive Stats

Write Descriptive stats of the distributions. Write any observations and differences seen in any of the distributions and plots.

From our code, we can see that for Challenger: Max game Duration: 2772.365 Mean Duration: 2162.786, Median: 2158.34, Std.Dev = 138.11 Platinum: Max Duration: 2714.2837 Mean Duration: 2172.5, Median: 2165.624, Std.Dev = 137.75

The Max game duration for Challenger games in our dataset is higher than the max game duration for Platinum games in our dataset. The mean and median for platinum games are higher than the mean and median durations of Challenger games. The standard deviations are very similar (not even 1 second apart).

In [12]:
```
grouped = filtered_df.groupby('Rank')
descriptive_stats = grouped['gameDuration'].agg(['max','mean','median','std'
descriptive_stats.columns = ['Rank','Max Duration','Mean Game Duraation', 'M
print(descriptive_stats)
```

```
         Rank  Max Duration  Mean Game Duraation  Median Game Duration  \
0  Challenger   2772.365479          2162.785743           2158.343018
1    Platinum   2714.283691          2172.499714           2165.624268

   Standard Deviation
0          138.105257
1          137.749369
```

Baseline OLS regression w/o interaction

In [8]:
```
import statsmodels.api as sm
import statsmodels.formula.api as smf

#Convert 'Rank' to a categorical variable if its not already
combined_df['Rank'] = combined_df['Rank'].astype('category')
model = smf.ols('ingameDuration ~(Rank) + Ranked', data = combined_df).fit()

print((model.summary()))
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:         ingameDuration   R-squared:                       0.6
53
Model:                            OLS   Adj. R-squared:                  0.6
53
Method:                 Least Squares   F-statistic:                 1.504e+
05
Date:                Sun, 05 May 2024   Prob (F-statistic):               0.
00
Time:                        02:49:22   Log-Likelihood:            -1.0381e+
06
No. Observations:              159928   AIC:                          2.076e+
06
Df Residuals:                  159925   BIC:                          2.076e+
06
Df Model:                           2
Covariance Type:            nonrobust
================================================================================
========
                       coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
--------
Intercept         2314.7764      0.965   2397.598      0.000    2312.884
2316.669
Rank[T.Platinum]    -1.5557      0.798     -1.950      0.051      -3.120
0.008
Ranked             -95.5041      0.174   -548.505      0.000     -95.845
-95.163
==============================================================================
==
Omnibus:                    65692.223   Durbin-Watson:                   0.8
67
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           954553.6
90
Skew:                          -1.585   Prob(JB):                         0.
00
Kurtosis:                      14.541   Cond. No.                         1
3.8
==============================================================================
==

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is corre
ctly specified.
```

Interpretation of OLS (Level, Rank, ingameDuration)

Our dependent variable here is 'ingameDuration' and our intercept here is a baseline that we decided to be Challenger rank in-game duration. Our intercept

is 2314.7764 with std.error 0.965 and is statistically significant. This is the average game duration for challenger players when their ranked/placement is 1st. Rank[T.Platinum] coefficient = -1.5557 with p-value of 0.051 very close to being statistically significant tells us that rank 1 placement of Platinum has average game duration -1.5557 seconds less than the baseline. Ranked coefficient = -95.5041 with std. error 0.174 (statistically significant) tells us the reduction of in-game duration for every placement descending from 1st. R-sq is 0.653 which indicates 65.3% of the variance in in-game duration is explained by this model. This is a moderate value, suggesting a decent fit but still there's variability unexplained. Adjusted R-sq is same value, confirming that the model is not overfitting, but I believe it is underfitting.

This simple OLS does not really support our hypothesis too much, as we know that with every placement in descending order, there will of course a reduction of in-game Duration of the player. Also, we want to know not only about the overall game durations of challenger vs platinum but the differences in game durations of Challenger vs Platinum ACROSS each placement ('ranked')

To strengthen our model, we introduce interaction terms for Ranked and Rank (placement and Rank) to see the regression when facing the combination of rank and placement.

In [9]:
```
#Interaction Terms
#We use sample because kernel kept crashing. I think memory overload for Jup
sampled_df = filtered_df.sample(frac=0.5, random_state=1)  # Adjust fraction

# Fit the model on the sampled dataset
model_sampled = smf.ols('ingameDuration ~ C(Rank) * C(Ranked)', data=sampled
print(model_sampled.summary())
```

OLS Regression Results

Dep. Variable:	ingameDuration	R-squared:	0.691
Model:	OLS	Adj. R-squared:	0.691
Method:	Least Squares	F-statistic:	1.181e+04
Date:	Sun, 05 May 2024	Prob (F-statistic):	0.00
Time:	02:49:25	Log-Likelihood:	-5.0952e+05
No. Observations:	79294	AIC:	1.019e+06
Df Residuals:	79278	BIC:	1.019e+06
Df Model:	15		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	2149.1558	2.120	1013.762	0.000	2145.001	2153.311
C(Rank)[T.Platinum]	16.2480	2.985	5.444	0.000	10.398	22.098
C(Ranked)[T.2]	4.8679	2.993	1.626	0.104	-0.999	10.734
C(Ranked)[T.3]	-111.0051	2.999	-37.018	0.000	-116.883	-105.128
C(Ranked)[T.4]	-205.8093	3.010	-68.385	0.000	-211.708	-199.911
C(Ranked)[T.5]	-288.3661	3.002	-96.052	0.000	-294.250	-282.482
C(Ranked)[T.6]	-379.3171	3.003	-126.334	0.000	-385.202	-373.432
C(Ranked)[T.7]	-486.7801	2.994	-162.607	0.000	-492.648	-480.913
C(Ranked)[T.8]	-643.5313	3.003	-214.299	0.000	-649.417	-637.646
C(Rank)[T.Platinum]:C(Ranked)[T.2]	-4.3419	4.233	-1.026	0.305	-12.638	3.954
C(Rank)[T.Platinum]:C(Ranked)[T.3]	-9.5636	4.235	-2.258	0.024	-17.864	-1.263
C(Rank)[T.Platinum]:C(Ranked)[T.4]	-12.8131	4.259	-3.008	0.003	-21.161	-4.465
C(Rank)[T.Platinum]:C(Ranked)[T.5]	-20.0476	4.237	-4.732	0.000	-28.351	-11.744
C(Rank)[T.Platinum]:C(Ranked)[T.6]	-22.9088	4.230	-5.416	0.000	-31.200	-14.618
C(Rank)[T.Platinum]:C(Ranked)[T.7]	-25.6047	4.222	-6.065	0.000	-33.879	-17.330
C(Rank)[T.Platinum]:C(Ranked)[T.8]	-39.2027	4.231	-9.265	0.000		

```
00     -47.496     -30.910
==============================================================================
==
Omnibus:                    18268.161   Durbin-Watson:                   2.0
17
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           129168.4
32
Skew:                          -0.927   Prob(JB):                         0.
00
Kurtosis:                       8.971   Cond. No.                         2
3.2
==============================================================================
==

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is corre
ctly specified.
```

Interpretation of OLS with Interaction Term (Ranked Rank)

Coefficients:

1. Intercept still baseline: 2149.1558, value of in-game duration when 1st place and Challenger.
2. C(Rank)[T.Platinum]: 16.248, std. error 2.985, in-game duration increases by approx this value if 'Platinum' rank.
3. C(Ranked)[T.2] ~ [T.8]: We choose to not look at T.2 because this shows second place game-duration correlation, but 2nd place value should be 0 since when 2nd place dies off, 1st place wins, therefore both end game at same time. Hence why p-value not statistically significant. But for T-3 ~ T-8, all negative values and statistically significant because this is Ranked (placement) without interaction. The coefficient values indicate the in-game duration decreases for each placement after 1st place.
4. C(Rank)[T.Platinum]:C(Ranked)[T.2] ~ [T.8]: All negative values with only T-2 not statistically significant because of prior reason stated above. These are the interaction terms. Here since they're all negative, this value indicates the decrease and difference in in-game duration of Platinum vs Challenger games in the respective placements.

R-squared and adjusted R-squared has improved to .691 meaning this is a better fitted model and explains more of the variability. Also lower AIC and BIC indicates this model is a better fit for regression analysis.

Quantile Regression

To further improve our analyses on the relationship between Platinum and Challenger in-game durations, we will perform quantile regression to take a deeper dive on the better/longer and worse/shorter performances in the distributions and then regressing. We do this because we want to see how placement and rank affect in-game duration at different quantiles and not just the mean of the distribution. This helps to provide insight into very quick defeats or prolonged games (analyzing outcomes at extremes).

In [14]:
```
#QUANTILE REGRESSION
#Import libraries and tools to quantile regress
#Define Quantiles we want and create a dict. to store all models from the fo
#Also store results in empty dict to print dataframe later

import statsmodels.formula.api as smf
import statsmodels.regression.quantile_regression as smqr
quantiles = [0.1, 0.25, 0.5, 0.75, 0.9]  # Different quantiles for detailed
models = {}
mod = smf.quantreg('ingameDuration ~ C(Rank) * C(Ranked)', filtered_df) #Qua
regression_results = {}
for qt in quantiles:
    res = mod.fit(q=qt)
    models[qt] = res
    regression_results[qt] = res.params
    print(f'\nSummary for the {qt*100}th percentile')
    print(res.summary())

results_df = pd.DataFrame(regression_results)
```

Summary for the 10.0th percentile

QuantReg Regression Results

Dep. Variable:	ingameDuration	Pseudo R-squared:	0.4763
Model:	QuantReg	Bandwidth:	9.999
Method:	Least Squares	Sparsity:	695.5
Date:	Sun, 05 May 2024	No. Observations:	158587
Time:	02:54:59	Df Residuals:	158571
		Df Model:	15

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1998.1881	2.097	952.913	0.000	1994.078	2002.298
C(Rank)[T.Platinum]	-1.5699	2.964	-0.530	0.596	-7.380	4.240
C(Ranked)[T.2]	3.092e-11	2.966	1.04e-11	1.000	-5.812	5.812
C(Ranked)[T.3]	-103.8882	2.966	-35.032	0.000	-109.701	-98.076
C(Ranked)[T.4]	-175.2770	2.966	-59.105	0.000	-181.089	-169.465
C(Ranked)[T.5]	-286.5540	2.966	-96.629	0.000	-292.366	-280.742
C(Ranked)[T.6]	-391.8873	2.966	-132.142	0.000	-397.700	-386.075
C(Ranked)[T.7]	-503.9404	2.966	-169.934	0.000	-509.753	-498.128
C(Ranked)[T.8]	-704.0024	2.966	-237.397	0.000	-709.815	-698.190
C(Rank)[T.Platinum]:C(Ranked)[T.2]	-2.547e-11	4.192	-6.07e-12	1.000	-8.217	8.217
C(Rank)[T.Platinum]:C(Ranked)[T.3]	-2.9385	4.192	-0.701	0.483	-11.155	5.278
C(Rank)[T.Platinum]:C(Ranked)[T.4]	-13.3303	4.192	-3.180	0.001	-21.547	-5.114
C(Rank)[T.Platinum]:C(Ranked)[T.5]	-11.3475	4.192	-2.707	0.007	-19.564	-3.131
C(Rank)[T.Platinum]:C(Ranked)[T.6]	-19.6117	4.192	-4.678	0.000	-27.829	-11.395
C(Rank)[T.Platinum]:C(Ranked)[T.7]	-10.9520	4.192	-2.612	0.009	-19.169	-2.735
C(Rank)[T.Platinum]:C(Ranked)[T.8]	-63.2061	4.192	-15.077	0.000	-71.423	-54.989

Summary for the 25.0th percentile

QuantReg Regression Results

Dep. Variable:	ingameDuration	Pseudo R-squared:	0.4997
Model:	QuantReg	Bandwidth:	10.12
Method:	Least Squares	Sparsity:	37 3.3
Date:	Sun, 05 May 2024	No. Observations:	158587
Time:	02:55:09	Df Residuals:	158571
		Df Model:	15

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	2063.4106	1.624	1270.379	0.000	2060.227	2066.594
C(Rank)[T.Platinum]	3.0901	2.296	1.346	0.178	-1.411	7.591
C(Ranked)[T.2]	1.637e-11	2.297	7.13e-12	1.000	-4.502	4.502
C(Ranked)[T.3]	-114.4097	2.297	-49.804	0.000	-118.912	-109.907
C(Ranked)[T.4]	-188.2833	2.297	-81.968	0.000	-192.785	-183.781
C(Ranked)[T.5]	-261.5604	2.297	-113.861	0.000	-266.063	-257.058
C(Ranked)[T.6]	-375.5473	2.297	-163.485	0.000	-380.050	-371.045
C(Ranked)[T.7]	-501.6156	2.297	-218.375	0.000	-506.118	-497.113
C(Ranked)[T.8]	-623.3981	2.297	-271.393	0.000	-627.900	-618.896
C(Rank)[T.Platinum]:C(Ranked)[T.2]	-0.0012	3.248	-0.000	1.000	-6.366	6.364
C(Rank)[T.Platinum]:C(Ranked)[T.3]	-2.0946	3.248	-0.645	0.519	-8.460	4.271
C(Rank)[T.Platinum]:C(Ranked)[T.4]	-10.0022	3.247	-3.080	0.002	-16.367	-3.637
C(Rank)[T.Platinum]:C(Ranked)[T.5]	-22.9681	3.248	-7.072	0.000	-29.333	-16.603
C(Rank)[T.Platinum]:C(Ranked)[T.6]	-16.5901	3.248	-5.109	0.000	-22.955	-10.225
C(Rank)[T.Platinum]:C(Ranked)[T.7]	-12.6888	3.247	-3.907	0.000	-19.054	-6.324
C(Rank)[T.Platinum]:C(Ranked)[T.8]	-35.0043	3.247	-10.779	0.000	-41.369	-28.639

```
==========================

Summary for the 50.0th percentile
                         QuantReg Regression Results
==============================================================================
==
Dep. Variable:         ingameDuration   Pseudo R-squared:               0.48
08
Model:                       QuantReg   Bandwidth:                       11.
50
Method:                 Least Squares   Sparsity:                        31
4.8
Date:                Sun, 05 May 2024   No. Observations:               1585
87
Time:                        02:55:16   Df Residuals:                   1585
71
                                        Df Model:
15
==============================================================================
==========================
                                             coef    std err          t      P>|
t|      [0.025      0.975]
------------------------------------------------------------------------------
-------------------------
Intercept                           2150.1536      1.581   1359.737      0.0
00    2147.054    2153.253
C(Rank)[T.Platinum]                    7.0991      2.236      3.175      0.0
01       2.717      11.481
C(Ranked)[T.2]                      -2.91e-11      2.236   -1.3e-11      1.0
00      -4.383       4.383
C(Ranked)[T.3]                      -121.0737      2.236    -54.140      0.0
00    -125.457    -116.691
C(Ranked)[T.4]                      -216.9974      2.236    -97.034      0.0
00    -221.381    -212.614
C(Ranked)[T.5]                      -280.7799      2.236   -125.556      0.0
00    -285.163    -276.397
C(Ranked)[T.6]                      -375.2820      2.236   -167.810      0.0
00    -379.665    -370.899
C(Ranked)[T.7]                      -477.1133      2.236   -213.350      0.0
00    -481.496    -472.730
C(Ranked)[T.8]                      -630.1809      2.236   -281.796      0.0
00    -634.564    -625.798
C(Rank)[T.Platinum]:C(Ranked)[T.2]    2.91e-11      3.162   9.21e-12      1.0
00      -6.197       6.197
C(Rank)[T.Platinum]:C(Ranked)[T.3]     -1.9025      3.162     -0.602      0.5
47      -8.099       4.294
C(Rank)[T.Platinum]:C(Ranked)[T.4]     -5.9146      3.162     -1.871      0.0
61     -12.111       0.282
C(Rank)[T.Platinum]:C(Ranked)[T.5]    -14.1344      3.162     -4.471      0.0
00     -20.331      -7.938
C(Rank)[T.Platinum]:C(Ranked)[T.6]    -19.2125      3.162     -6.077      0.0
00     -25.409     -13.016
C(Rank)[T.Platinum]:C(Ranked)[T.7]    -21.5409      3.162     -6.813      0.0
00     -27.737     -15.344
C(Rank)[T.Platinum]:C(Ranked)[T.8]    -12.4060      3.162     -3.924      0.0
00     -18.603      -6.209
```

```
==============================================================================
==========================

Summary for the 75.0th percentile
                         QuantReg Regression Results
==============================================================================
==
Dep. Variable:         ingameDuration   Pseudo R-squared:               0.45
00
Model:                       QuantReg   Bandwidth:                       10.
06
Method:                 Least Squares   Sparsity:                        40
3.3
Date:                Sun, 05 May 2024   No. Observations:               1585
87
Time:                        02:55:25   Df Residuals:                   1585
71
                                        Df Model:
15
==============================================================================
==========================
                                             coef    std err          t      P>|
t|      [0.025      0.975]
------------------------------------------------------------------------------
--------------------------
Intercept                             2242.3584      1.755   1278.035      0.0
00    2238.920    2245.797
C(Rank)[T.Platinum]                     11.8110      2.481      4.761      0.0
00       6.949      16.673
C(Ranked)[T.2]                       -1.455e-11      2.481  -5.86e-12      1.0
00      -4.864       4.864
C(Ranked)[T.3]                        -131.5706      2.481    -53.025      0.0
00    -136.434    -126.707
C(Ranked)[T.4]                        -238.5844      2.481    -96.153      0.0
00    -243.448    -233.721
C(Ranked)[T.5]                        -312.0525      2.481   -125.754      0.0
00    -316.916    -307.189
C(Ranked)[T.6]                        -376.1615      2.481   -151.593      0.0
00    -381.025    -371.298
C(Ranked)[T.7]                        -484.7042      2.481   -195.344      0.0
00    -489.567    -479.841
C(Ranked)[T.8]                        -633.9225      2.481   -255.481      0.0
00    -638.786    -629.059
C(Rank)[T.Platinum]:C(Ranked)[T.2]   1.091e-11      3.508   3.11e-12      1.0
00      -6.876       6.876
C(Rank)[T.Platinum]:C(Ranked)[T.3]    -0.6013      3.508     -0.171      0.8
64      -7.477       6.274
C(Rank)[T.Platinum]:C(Ranked)[T.4]     3.4634      3.508      0.987      0.3
24      -3.412      10.339
C(Rank)[T.Platinum]:C(Ranked)[T.5]    -5.6830      3.508     -1.620      0.1
05     -12.559       1.193
C(Rank)[T.Platinum]:C(Ranked)[T.6]   -18.4666      3.508     -5.264      0.0
00     -25.342     -11.591
C(Rank)[T.Platinum]:C(Ranked)[T.7]   -19.5623      3.508     -5.576      0.0
00     -26.438     -12.687
C(Rank)[T.Platinum]:C(Ranked)[T.8]    -7.9718      3.508     -2.272      0.0
```

```
23     -14.847      -1.096
==============================================================================
==========================

Summary for the 90.0th percentile
                         QuantReg Regression Results
==============================================================================
==
Dep. Variable:         ingameDuration   Pseudo R-squared:               0.40
48
Model:                       QuantReg   Bandwidth:                       9.5
06
Method:                 Least Squares   Sparsity:                        84
3.7
Date:                Sun, 05 May 2024   No. Observations:               1585
87
Time:                        02:55:43   Df Residuals:                   1585
71
                                        Df Model:
15
==============================================================================
==========================
                                            coef    std err          t      P>|
t|      [0.025      0.975]
------------------------------------------------------------------------------
--------------------------
Intercept                             2317.9399      2.544    911.270      0.0
00    2312.954    2322.925
C(Rank)[T.Platinum]                     24.1643      3.596      6.720      0.0
00      17.116      31.212
C(Ranked)[T.2]                        3.638e-12      3.597   1.01e-12      1.0
00      -7.051       7.051
C(Ranked)[T.3]                        -107.2053      3.597    -29.802      0.0
00    -114.256    -100.155
C(Ranked)[T.4]                        -227.8716      3.597    -63.346      0.0
00    -234.922    -220.821
C(Ranked)[T.5]                        -319.4540      3.597    -88.805      0.0
00    -326.505    -312.403
C(Ranked)[T.6]                        -386.1031      3.597   -107.328      0.0
00    -393.154    -379.052
C(Ranked)[T.7]                        -459.2379      3.597   -127.664      0.0
00    -466.288    -452.187
C(Ranked)[T.8]                        -601.2960      3.597   -167.155      0.0
00    -608.347    -594.245
C(Rank)[T.Platinum]:C(Ranked)[T.2] -3.638e-12      5.086  -7.15e-13      1.0
00      -9.967       9.967
C(Rank)[T.Platinum]:C(Ranked)[T.3]   -16.2717      5.085     -3.200      0.0
01     -26.239      -6.305
C(Rank)[T.Platinum]:C(Ranked)[T.4]   -10.2969      5.085     -2.025      0.0
43     -20.264      -0.330
C(Rank)[T.Platinum]:C(Ranked)[T.5]    -2.5382      5.085     -0.499      0.6
18     -12.505       7.429
C(Rank)[T.Platinum]:C(Ranked)[T.6]   -23.6254      5.085     -4.646      0.0
00     -33.593     -13.658
C(Rank)[T.Platinum]:C(Ranked)[T.7]   -36.2278      5.085     -7.124      0.0
00     -46.195     -26.261
```

```
C(Rank)[T.Platinum]:C(Ranked)[T.8]   -27.4785      5.085     -5.404      0.0
00     -37.446     -17.512
==============================================================================
==========================
```

In [15]:
```
#Print Dataframe of all coefficients at all quantiles specified
print(results_df)
```

```
                                        0.10          0.25          0.50
\
Intercept                          1.998188e+03  2.063411e+03  2.150154e+03
C(Rank)[T.Platinum]               -1.569946e+00  3.090088e+00  7.099121e+00
C(Ranked)[T.2]                     3.092282e-11  1.637090e-11 -2.910383e-11
C(Ranked)[T.3]                    -1.038882e+02 -1.144097e+02 -1.210737e+02
C(Ranked)[T.4]                    -1.752770e+02 -1.882833e+02 -2.169974e+02
C(Ranked)[T.5]                    -2.865540e+02 -2.615604e+02 -2.807799e+02
C(Ranked)[T.6]                    -3.918873e+02 -3.755473e+02 -3.752820e+02
C(Ranked)[T.7]                    -5.039404e+02 -5.016156e+02 -4.771133e+02
C(Ranked)[T.8]                    -7.040024e+02 -6.233981e+02 -6.301809e+02
C(Rank)[T.Platinum]:C(Ranked)[T.2] -2.546585e-11 -1.220703e-03  2.910383e-11
C(Rank)[T.Platinum]:C(Ranked)[T.3] -2.938476e+00 -2.094604e+00 -1.902466e+00
C(Rank)[T.Platinum]:C(Ranked)[T.4] -1.333032e+01 -1.000220e+01 -5.914551e+00
C(Rank)[T.Platinum]:C(Ranked)[T.5] -1.134753e+01 -2.296814e+01 -1.413440e+01
C(Rank)[T.Platinum]:C(Ranked)[T.6] -1.961169e+01 -1.659015e+01 -1.921252e+01
C(Rank)[T.Platinum]:C(Ranked)[T.7] -1.095202e+01 -1.268884e+01 -2.154089e+01
C(Rank)[T.Platinum]:C(Ranked)[T.8] -6.320605e+01 -3.500427e+01 -1.240601e+01

                                        0.75          0.90
Intercept                          2.242358e+03  2.317940e+03
C(Rank)[T.Platinum]                1.181104e+01  2.416431e+01
C(Ranked)[T.2]                    -1.455192e-11  3.637979e-12
C(Ranked)[T.3]                    -1.315706e+02 -1.072053e+02
C(Ranked)[T.4]                    -2.385844e+02 -2.278716e+02
C(Ranked)[T.5]                    -3.120525e+02 -3.194540e+02
C(Ranked)[T.6]                    -3.761615e+02 -3.861031e+02
C(Ranked)[T.7]                    -4.847042e+02 -4.592379e+02
C(Ranked)[T.8]                    -6.339225e+02 -6.012960e+02
C(Rank)[T.Platinum]:C(Ranked)[T.2]  1.091394e-11 -3.637979e-12
C(Rank)[T.Platinum]:C(Ranked)[T.3] -6.013190e-01 -1.627173e+01
C(Rank)[T.Platinum]:C(Ranked)[T.4]  3.463378e+00 -1.029688e+01
C(Rank)[T.Platinum]:C(Ranked)[T.5] -5.682984e+00 -2.538208e+00
C(Rank)[T.Platinum]:C(Ranked)[T.6] -1.846657e+01 -2.362537e+01
C(Rank)[T.Platinum]:C(Ranked)[T.7] -1.956226e+01 -3.622778e+01
C(Rank)[T.Platinum]:C(Ranked)[T.8] -7.971804e+00 -2.747852e+01
```

Interpretation of the Quantile Regression

We have quantiles .10, .25, .50, .75, .90. So we are looking regression of in game Duration of player on the interaction terms Rank and Ranked (Challenger and Platinum interaction with in-game placement 1-8). For the regressions, we have set the intercept to being a Challenger player placing 1st place (the baseline).

COEFFICIENTS in 90th Quantile:

1. Intercept = 2317.9399 meaning average time in seconds of Challenger 1st place in the 90th percentile of the distribution
2. C(Rank)[T.Platinum] = 24.1643 (stat. significant) seconds increase for Platinum 1st place vs baseline
3. C(Ranked)[T.2] - [T.8] coeffs. are relatively all negative because it is just telling us average seconds decrease from 1st to 2nd to 3rd, and so on in baseline rank (Challenger). 2nd place should be approximately zero because when 2nd place gets killed off, 1st place wins, and game is over.
4. C(Rank)[T.Platinum]:C(Ranked)[T.2] - [T.8] are all negative coefficients meaning for every rank, especially 6-8th, Platinum players have shorter in-game Duration than the placement equivalent of Challenger players. All statistically significant except for 5th place.

ALL Quantiles: We can see that in all quantiles, from the worst/shortest game performances (10th quantile) to the best/longest (90th), almost all interaction terms are negative, with values significantly lower in the 6th-8th placement than 2nd-4th, with almost all interaction terms statistically significant (all 6th - 8th place significant).

Interpretation: The coefficient of 2nd variable: C(Rank)[T.Platinum] becomes increasingly positive as we reach higher quantiles, being 'Platinum' entails longer in-game durations but even at the 90th percentile the effect is quite minimal with only 24 second difference as the max.

C(Ranked) T 2-8: are the baseline rank's (Challenger) average in-game duration decrease with every lower placement.

Our main interest lies in the coefficients of the interaction terms: C(Rank)[T.Platinum]:C(Ranked)[T.2] - [T.8] which are all negative in every quantile, tells us that for every placement, there is a statistically significant reduction in game duration between Challenger and Platinum players.

For our pseudo R-squared: we have at every quantile values from 0.4 - 0.5, which at 0.4 explains variability in outcome variable around its conditional quantile. 0.4 value is substantial for quantile regression and indicates a good fit.

DF is = 15 which is the total number of independent paramaters including interaction terms and intercept estimated by the model. The higher number increases model complexity.

Conclusion

I wanted to know if there was a statistically significant difference in game durations between middle-tier skilled players and the best in the game. I pursued this analysis because after playing this game for years, I still sit somewhere between Platinum and Challenger so this was just a topic of interest.

Throughout the creation of this project, I found mistakes in the dataset that I ultimately had to clean in order to have better foundation for regression. Then I implemented ways to find a better fitting model that gave stat. sig. coefficients. In my regression analysis, I found that there is a statistically significant difference in game durations with every player defeated (placements) between the two ranks.

I think choosing quantile regression in this scenario was optimal because it helps to understand distribution extremes. There are games where RNG just favors everyone and games where it doesn't or games where everyone is underperforming or making mistakes. It also helps to handle skewed distributions and quantile regression does not require the assumption of normality.

In a gaming context, understanding that game duration can affect rankings or player enjoyment/satisfaction, helps in exploring factors that lead to different game lengths at various quantiles. This is especially important for adjusting game balance. It is particularly useful in this field because the tails fo the distributio are just as important as the center(mean).

In [ ]: