Competitiveness of 2022 League of Legends World Championship Matches

Name(s): Brighten Hayama

Website Link: https://brighyama.github.io/LoL-data-analysis/

Code

In [1]:
```
import pandas as pd
import numpy as np
import os

import plotly.express as px
import plotly.graph_objects as go
pd.options.plotting.backend = 'plotly'
```

Introduction

Question: Are 2022 League of Legends World Championship Games More Competitive Than Regular Season Games?

Cleaning and EDA

In [2]:
```
# Importing raw league data
league_path = os.path.join('data', '2022_LoL_esports_match_data_from_OraclesElixir.csv')
league = pd.read_csv(league_path)
league
```

```
C:\Users\brigh\anaconda3\envs\dsc80\lib\site-packages\IPython\core\interactiveshell.py:3505: DtypeWarning: Column
s (2) have mixed types.Specify dtype option on import or set low_memory=False.
  exec(code_obj, self.user_global_ns, self.user_ns)
```

Out[2]:

	gameid	datacompleteness	url	league	year	split	playoffs	dat
0	ESPORTSTMNT01_2690210	complete	NaN	LCK CL	2022	Spring	0	2022-01-1 07:44:0
1	ESPORTSTMNT01_2690210	complete	NaN	LCK CL	2022	Spring	0	2022-01-1 07:44:0
2	ESPORTSTMNT01_2690210	complete	NaN	LCK CL	2022	Spring	0	2022-01-1 07:44:0
3	ESPORTSTMNT01_2690210	complete	NaN	LCK CL	2022	Spring	0	2022-01-1 07:44:0
4	ESPORTSTMNT01_2690210	complete	NaN	LCK CL	2022	Spring	0	2022-01-1 07:44:0
...	...	...	...	...	...	...	...	
149395	9687-9687_game_5	partial	https://lpl.qq.com/es/stats.shtml?bmid=9687	DC	2022	NaN	0	2022-12-2 12:43:4
149396	9687-9687_game_5	partial	https://lpl.qq.com/es/stats.shtml?bmid=9687	DC	2022	NaN	0	2022-12-2 12:43:4
149397	9687-9687_game_5	partial	https://lpl.qq.com/es/stats.shtml?bmid=9687	DC	2022	NaN	0	2022-12-2 12:43:4
149398	9687-9687_game_5	partial	https://lpl.qq.com/es/stats.shtml?bmid=9687	DC	2022	NaN	0	2022-12-2 12:43:4
149399	9687-9687_game_5	partial	https://lpl.qq.com/es/stats.shtml?bmid=9687	DC	2022	NaN	0	2022-12-2 12:43:4

149400 rows × 123 columns

Data Cleaning

In [3]:

```
# Choosing relevant columns from the raw dataset
columns = ['gameid','league','patch','teamname','result','gamelength','golddiffat15']

# Subsetting league data to entire team stats with relevant columns
df = league[league['position']=='team'][columns].copy()
df['abs_golddiffat15'] = df['golddiffat15'].abs()
df['is_wcs'] = df['league']=='WCS'
df[df['golddiffat15'].notna()]
```

Out[3]:

	gameid	league	patch	teamname	result	gamelength	golddiffat15	abs_golddiffat15	is_wcs
10	ESPORTSTMNT01_2690210	LCK CL	12.01	Fredit BRION Challengers	0	1713	107.0	107.0	False
11	ESPORTSTMNT01_2690210	LCK CL	12.01	Nongshim RedForce Challengers	1	1713	-107.0	107.0	False
22	ESPORTSTMNT01_2690219	LCK CL	12.01	T1 Challengers	0	2114	-1763.0	1763.0	False
23	ESPORTSTMNT01_2690219	LCK CL	12.01	Liiv SANDBOX Challengers	1	2114	1763.0	1763.0	False
46	ESPORTSTMNT01_2690227	LCK CL	12.01	KT Rolster Challengers	1	1972	1191.0	1191.0	False
...	...	...	...	...	...	...	...	...	...
149111	ESPORTSTMNT01_3268686	NEXO	12.23	unknown team	1	2325	-853.0	853.0	False
149122	ESPORTSTMNT01_3269631	NEXO	12.23	unknown team	1	2076	2063.0	2063.0	False
149123	ESPORTSTMNT01_3269631	NEXO	12.23	Córdoba Patrimonio eSports	0	2076	-2063.0	2063.0	False
149134	ESPORTSTMNT01_3268705	NEXO	12.23	unknown team	1	1680	2213.0	2213.0	False
149135	ESPORTSTMNT01_3268705	NEXO	12.23	Córdoba Patrimonio eSports	0	1680	-2213.0	2213.0	False

21262 rows × 9 columns

In [4]:

```
wcs = df.loc[(df['patch']==12.18) & (df['league']=='WCS')]
wcs
```

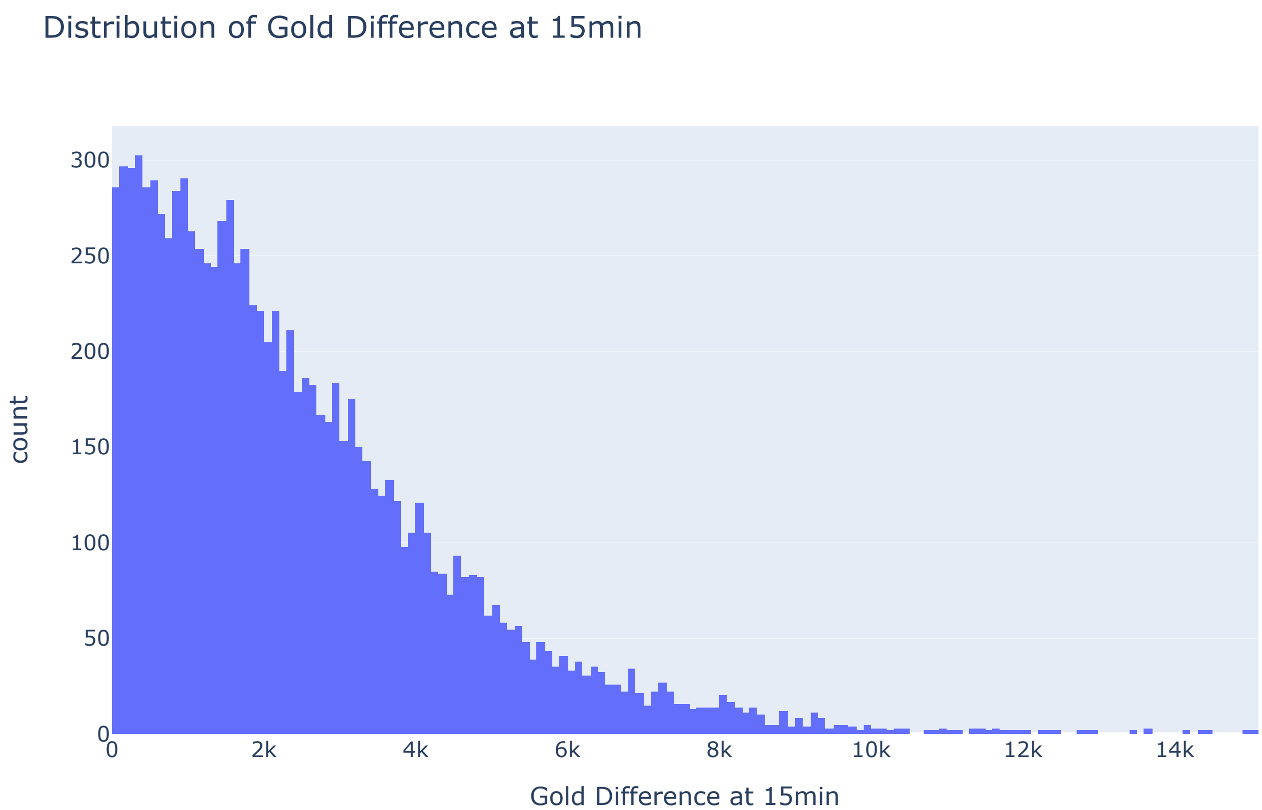
Out[4]:

	gameid	league	patch	teamname	result	gamelength	golddiffat15	abs_golddiffat15	is_wcs
136342	ESPORTSTMNT02_3041846	WCS	12.18	Isurus	0	2278	-2243.0	2243.0	True
136343	ESPORTSTMNT02_3041846	WCS	12.18	MAD Lions	1	2278	2243.0	2243.0	True
136354	ESPORTSTMNT02_3041862	WCS	12.18	Fnatic	1	1767	2961.0	2961.0	True
136355	ESPORTSTMNT02_3041862	WCS	12.18	Evil Geniuses	0	1767	-2961.0	2961.0	True
136366	ESPORTSTMNT02_3041903	WCS	12.18	LOUD	0	1723	-1680.0	1680.0	True
...	...	...	...	...	...	...	...	...	...
146219	ESPORTSTMNT02_3080871	WCS	12.18	DRX	0	1931	-796.0	796.0	True
146230	ESPORTSTMNT02_3080872	WCS	12.18	DRX	1	1724	1289.0	1289.0	True
146231	ESPORTSTMNT02_3080872	WCS	12.18	T1	0	1724	-1289.0	1289.0	True
146242	ESPORTSTMNT02_3080905	WCS	12.18	T1	0	2529	433.0	433.0	True
146243	ESPORTSTMNT02_3080905	WCS	12.18	DRX	1	2529	-433.0	433.0	True

254 rows × 9 columns

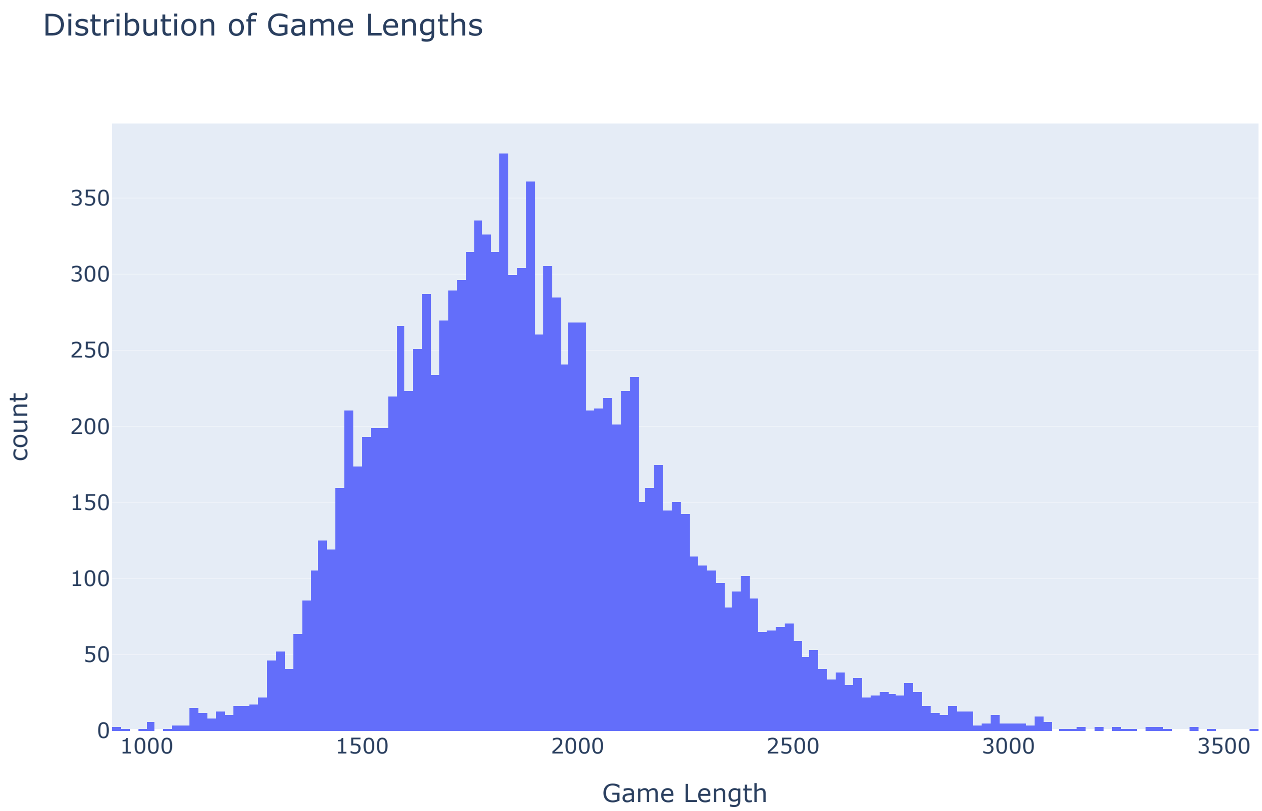
Univariate Analysis (plot 1)

In [5]:
```
df_vis = df[df['result']==1].copy()
fig1 = px.histogram(df_vis,x='abs_golddiffat15',labels={'abs_golddiffat15':'Gold Difference at 15min'},
                    title='Distribution of Gold Difference at 15min')
fig1.show()
```



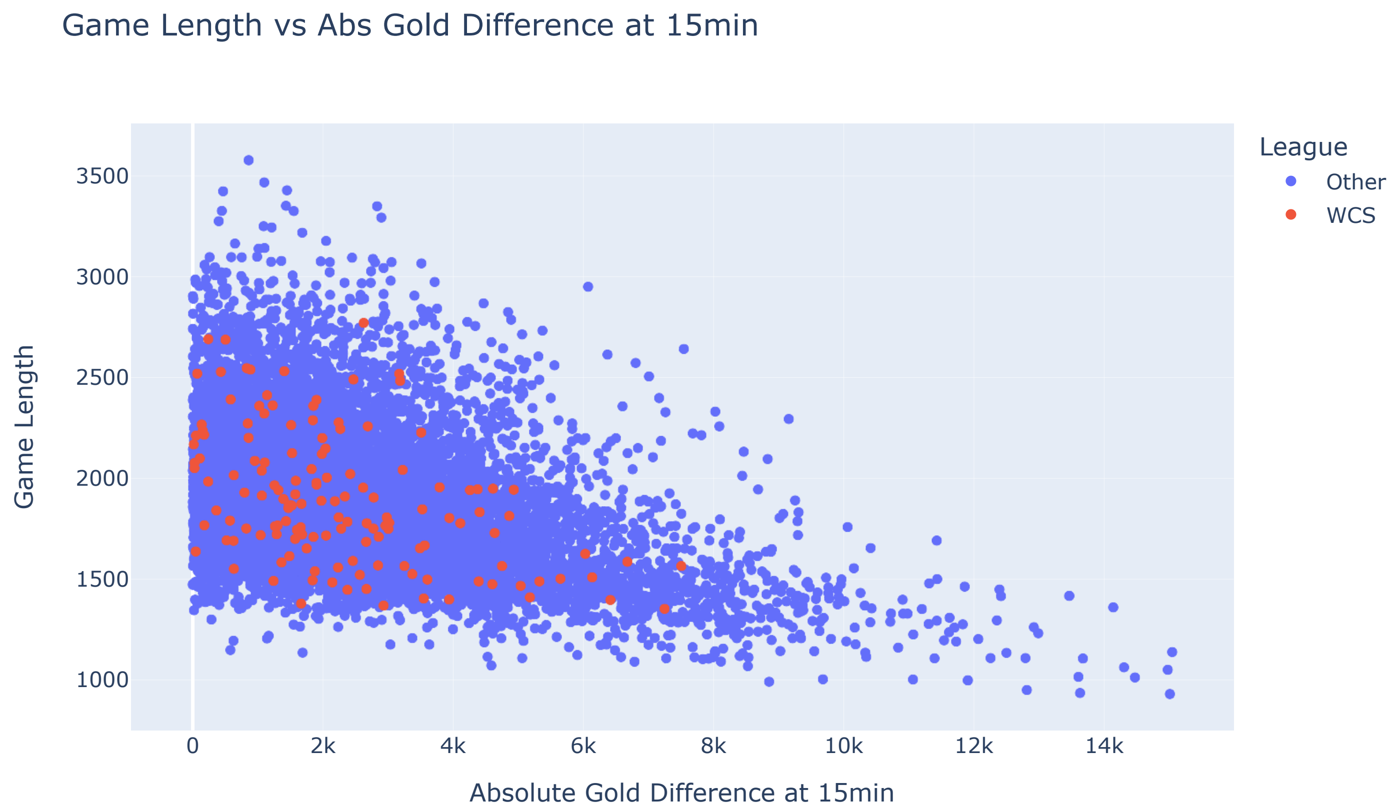
Univariate Analysis (plot 2)

In [6]:
```
fig2 = px.histogram(df_vis,x='gamelength',labels={'gamelength':'Game Length'},
                    title='Distribution of Game Lengths')
fig2.show()
```



Bivariate Analysis (plot 1)

In [7]:
```
df_vis['is_wcs'] = df_vis['is_wcs'].replace({True:'WCS',False:'Other'})
fig3 = px.scatter(df_vis,title='Game Length vs Abs Gold Difference at 15min',
                  x='abs_golddiffat15',y='gamelength',color='is_wcs',
                  labels={'abs_golddiffat15':'Absolute Gold Difference at 15min','gamelength':'Game Length','is_
fig3.show()
```



Aggregation

In [34]:
```
# Pivoting on each team in WCS and observing their gold differences at 15min
wcs_piv = pd.pivot_table(wcs,values='golddiffat15',columns='teamname',index='gameid')
wcs_piv
```

Out[34]:

teamname	100 Thieves	Beyond Gaming	CTBC Flying Oyster	Chiefs Esports Club	Cloud9	DRX	DWG KIA	DetonatioN FocusMe	EDward Gaming	Evil Geniuses	...
gameid											
ESPORTSTMNT02_3041846	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
ESPORTSTMNT02_3041862	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	-2961.0	...
ESPORTSTMNT02_3041903	NaN	1680.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
ESPORTSTMNT02_3041937	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
ESPORTSTMNT02_3041983	NaN	NaN	NaN	-573.0	NaN	NaN	NaN	NaN	NaN	NaN	...
...	...	...	...	...	...	...	...	...	...	...	...
ESPORTSTMNT02_3080864	NaN	NaN	NaN	NaN	NaN	-1475.0	NaN	NaN	NaN	NaN	...
ESPORTSTMNT02_3080870	NaN	NaN	NaN	NaN	NaN	-2626.0	NaN	NaN	NaN	NaN	...
ESPORTSTMNT02_3080871	NaN	NaN	NaN	NaN	NaN	-796.0	NaN	NaN	NaN	NaN	...
ESPORTSTMNT02_3080872	NaN	NaN	NaN	NaN	NaN	1289.0	NaN	NaN	NaN	NaN	...
ESPORTSTMNT02_3080905	NaN	NaN	NaN	NaN	NaN	-433.0	NaN	NaN	NaN	NaN	...

127 rows × 24 columns

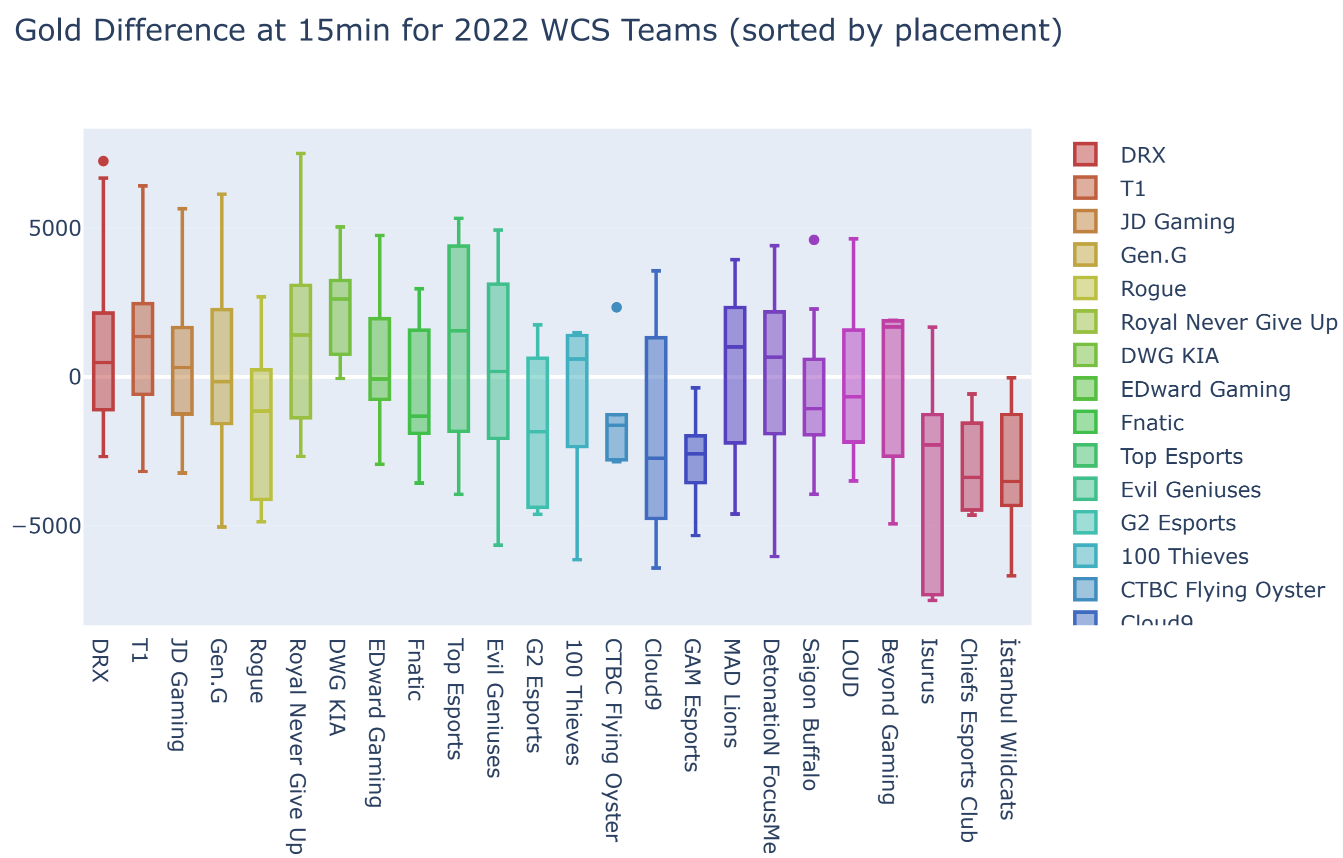
In [35]:
```python
wcs_group = wcs.groupby('teamname')[['golddiffat15']].agg(np.mean)
wcs_group
```

Out[35]:

teamname	golddiffat15
100 Thieves	-731.833333
Beyond Gaming	-274.800000
CTBC Flying Oyster	-1299.666667
Chiefs Esports Club	-2973.400000
Cloud9	-1957.166667
DRX	1006.846154
DWG KIA	2261.666667
DetonatioN FocusMe	-9.000000
EDward Gaming	672.909091
Evil Geniuses	364.125000
Fnatic	-555.090909
G2 Esports	-1714.166667
GAM Esports	-2728.833333
Gen.G	116.437500
Isurus	-3520.200000
JD Gaming	484.785714
LOUD	-151.200000
MAD Lions	154.833333
Rogue	-1452.400000
Royal Never Give Up	1279.473684
Saigon Buffalo	-538.555556
T1	1030.000000
Top Esports	1175.500000
İstanbul Wildcats	-3082.000000

Bivariate Analysis (plot 2)

In [9]:
```python
# List of teams in WCS sorted by placement (best to worst)
teams_sorted = ['DRX','T1','JD Gaming','Gen.G','Rogue','Royal Never Give Up','DWG KIA','EDward Gaming','Fnatic',
                'Evil Geniuses','G2 Esports','100 Thieves','CTBC Flying Oyster','Cloud9','GAM Esports','MAD Lior
                'DetonatioN FocusMe','Saigon Buffalo','LOUD','Beyond Gaming','Isurus','Chiefs Esports Club','İst
# Creating list of colors for box plots
c = ['hsl('+str(h)+',50%'+',50%)' for h in np.linspace(0, 360, len(wcs_piv.columns))]
# Creating box plot for each team in WCS
fig4 = go.Figure(data=[go.Box(y=wcs_piv[teams_sorted[i]],
                              name=teams_sorted[i],
                              marker_color=c[i])
                       for i in range(wcs_piv.shape[1])],
                 layout={'title':'Gold Difference at 15min for 2022 WCS Teams (sorted by placement)'})
fig4.show()
```



Assessment of Missingness

In [10]:
```
# Observing missingness in golddiffat15 column, compared to patch and gamelength
columns_missingness = ['patch','result','golddiffat15']
missingness = df[columns_missingness].copy()
missingness['golddiffat15_missing'] = missingness['golddiffat15'].isna()

missingness
```

Out[10]:

	patch	result	golddiffat15	golddiffat15_missing
10	12.01	0	107.0	False
11	12.01	1	-107.0	False
22	12.01	0	-1763.0	False
23	12.01	1	1763.0	False
34	12.01	1	NaN	True
...	...	...	...	...
149375	12.23	0	NaN	True
149386	12.23	1	NaN	True
149387	12.23	0	NaN	True
149398	12.23	0	NaN	True
149399	12.23	1	NaN	True

24900 rows × 4 columns

Independent Missingness (MCAR)

In [11]:
```
# Observing proportion of missing data from wins/losses
result_dist = (
    missingness
    .assign(golddiffat15_missing=missingness['golddiffat15'].isna())
    .pivot_table(index='result', columns='golddiffat15_missing', aggfunc='size')
)
result_dist = result_dist / result_dist.sum()
result_dist
```

Out[11]:

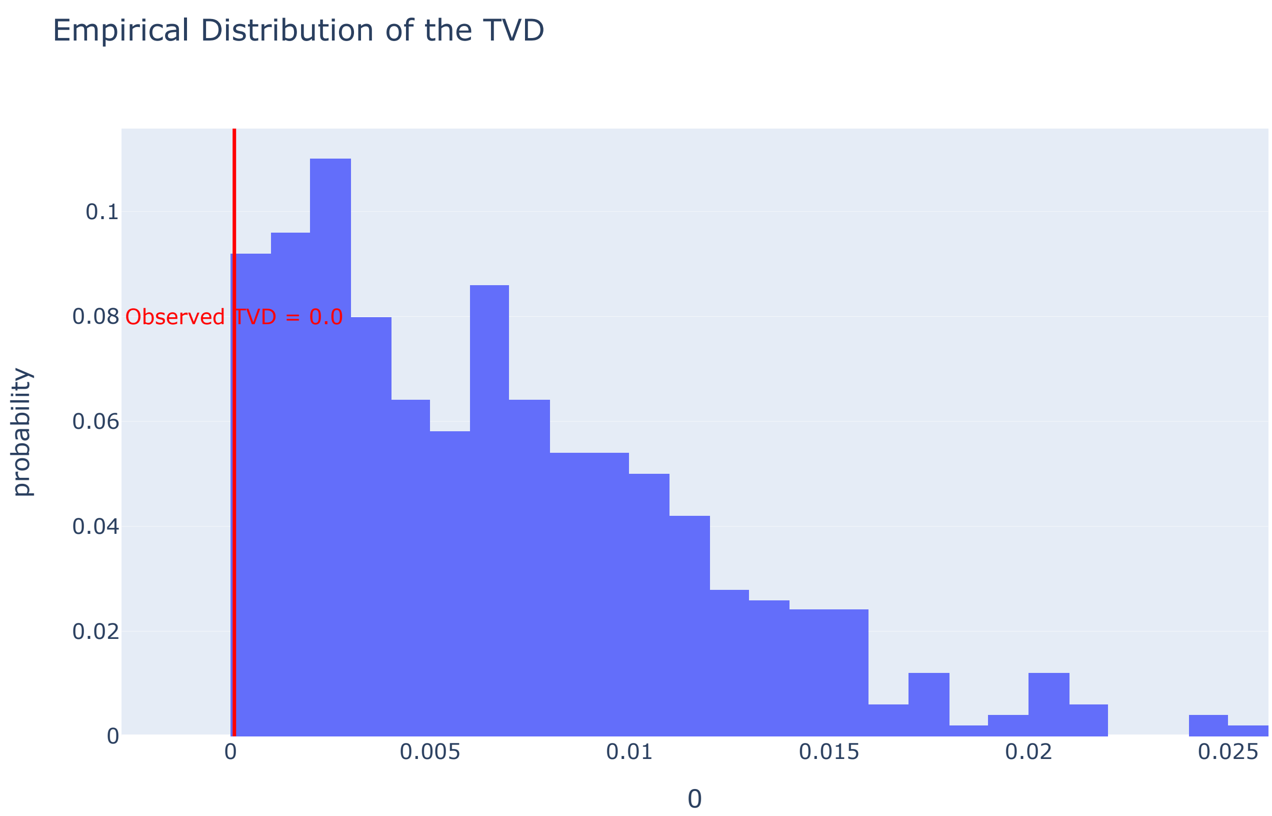
golddiffat15_missing	False	True
result		
0	0.500094	0.5
1	0.499906	0.5

In [12]:
```
# Permutation test on the gamelength column
n_repetitions = 500
shuffled = missingness.copy()

tvds = []
for _ in range(n_repetitions):
    shuffled['result'] = np.random.permutation(shuffled['result'])

    # Computing and storing the TVD.
    pivoted = (
        shuffled
        .pivot_table(index='result', columns='golddiffat15_missing', aggfunc='size')
        .apply(lambda x: x / x.sum())
    )

    tvd = pivoted.diff(axis=1).iloc[:, -1].abs().sum() / 2
    tvds.append(tvd)
```

In [13]:
```
observed_tvd = result_dist.diff(axis=1).iloc[:, -1].abs().sum() / 2
observed_tvd
```

Out[13]: 9.406452826640765e-05

In [14]:
```
fig = px.histogram(pd.DataFrame(tvds), x=0, nbins=50, histnorm='probability',
                   title='Empirical Distribution of the TVD')
fig.add_vline(x=observed_tvd, line_color='red')
fig.add_annotation(text=f'<span style="color:red">Observed TVD = {round(observed_tvd, 2)}</span>',
                   x=observed_tvd, showarrow=False, y=0.08)
```



In [15]:
```
p_value = np.mean(np.array(tvds) >= observed_tvd)
p_value
```

Out[15]:
```
1.0
```

- We fail to reject the null with significance level 0.01.
- Recall, the null stated that the distribution of `'result'` when `'golddiffat15'` is missing is the same as the distribution of `'result'` when `'golddiffat15'` is not missing.
- Hence, we conclude that the missingness in the `'golddiffat15'` column **is not dependent** on `'result'`. So the missingness can be classified as **MCAR**.

Dependent Missingness (MAR)

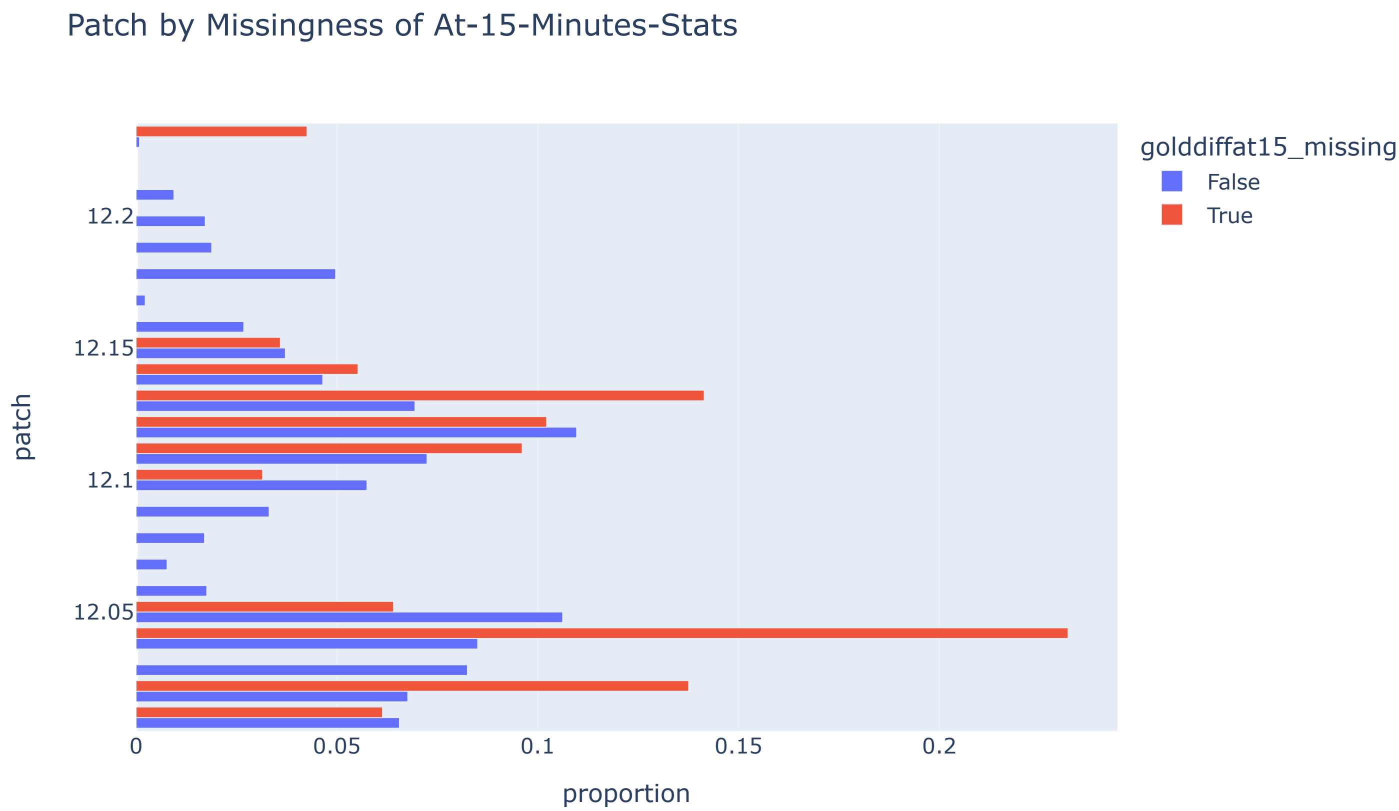
In [16]:
```
# Observing proportion of missing data from each patch
result_dist = (
    missingness
    .assign(golddiffat15_missing=missingness['golddiffat15'].isna())
    .pivot_table(index='patch', columns='golddiffat15_missing', aggfunc='size')
)

result_dist = result_dist / result_dist.sum()
result_dist
```

Out[16]:

golddiffat15_missing	False	True
patch		
12.01	0.065563	0.061326
12.02	0.067632	0.137569
12.03	0.082495	NaN
12.04	0.085034	0.232044
12.05	0.106199	0.064088
12.06	0.017590	NaN
12.07	0.007713	NaN
12.08	0.017026	NaN
12.09	0.033111	NaN
12.10	0.057473	0.031492
12.11	0.072430	0.096133
12.12	0.109679	0.102210
12.13	0.069420	0.141436
12.14	0.046468	0.055249
12.15	0.037155	0.035912
12.16	0.026808	NaN
12.17	0.002258	NaN
12.18	0.049666	NaN
12.19	0.018813	NaN
12.20	0.017214	NaN
12.21	0.009406	NaN
12.23	0.000847	0.042541

In [17]:

```
fig6 = result_dist.plot(kind='barh', title='Patch by Missingness of At-15-Minutes-Stats',
                 labels={'value':'proportion','variable':'Missing Gold Difference at 15 Min'},barmode='group')
fig6
```

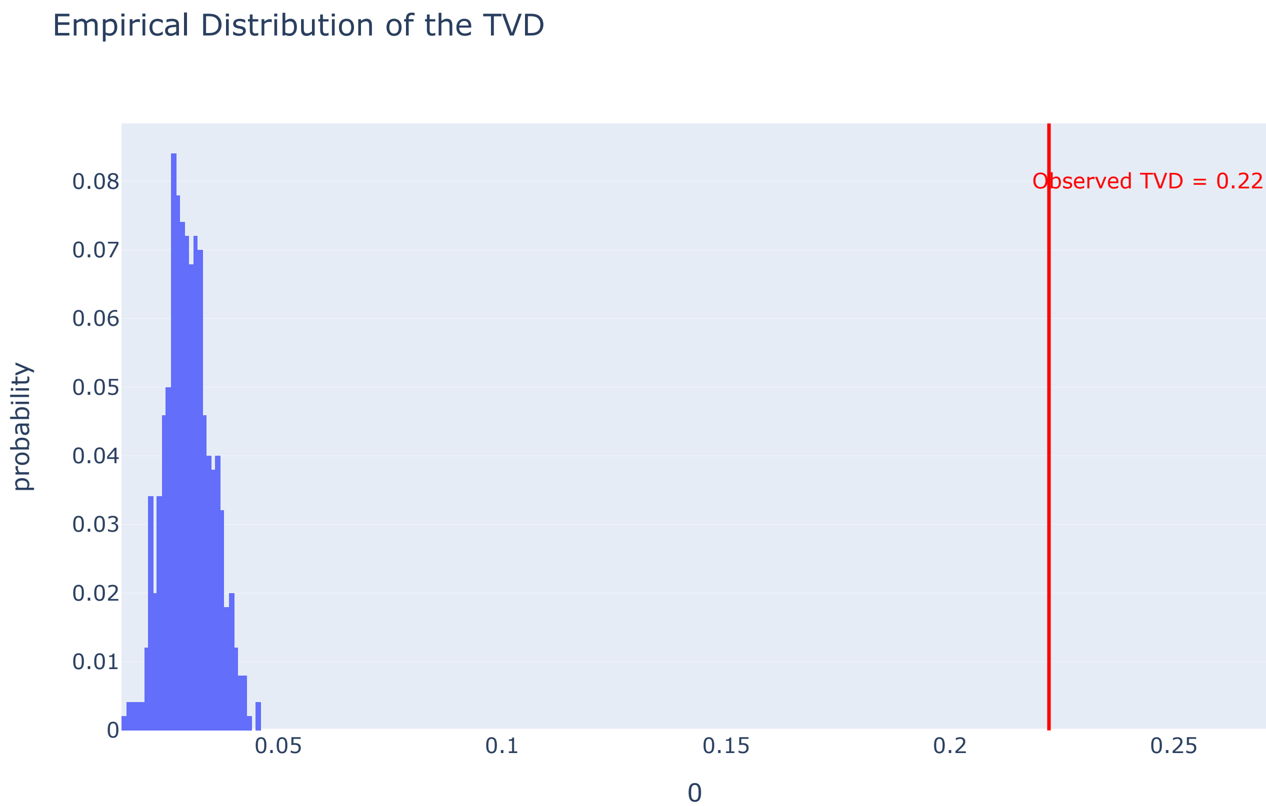


In [18]:
```
# Permutation test on the patch column
n_repetitions = 500
shuffled = missingness.copy()

tvds = []
for _ in range(n_repetitions):
    # Randomly permuting the patch column
    shuffled['patch'] = np.random.permutation(shuffled['patch'])

    # Computing and storing the TVD.
    pivoted = (
        shuffled
        .pivot_table(index='patch', columns='golddiffat15_missing', aggfunc='size')
        .apply(lambda x: x / x.sum())
    )

    tvd = pivoted.diff(axis=1).iloc[:, -1].abs().sum() / 2
    tvds.append(tvd)
```

In [19]:
```
observed_tvd = result_dist.diff(axis=1).iloc[:, -1].abs().sum() / 2
observed_tvd
```

Out[19]: 0.22209211983509086

In [20]:
```
fig = px.histogram(pd.DataFrame(tvds), x=0, nbins=50, histnorm='probability',
                   title='Empirical Distribution of the TVD')
fig.add_vline(x=observed_tvd, line_color='red')
fig.add_annotation(text=f'<span style="color:red">Observed TVD = {round(observed_tvd, 2)}</span>',
                   x=observed_tvd*1.1, showarrow=False, y=0.08)
```



In [21]:
```
p_value = np.mean(np.array(tvds) >= observed_tvd)
p_value
```

Out[21]: `0.0`

- We reject the null with significance level 0.01.
- The null stated that the distribution of `'league'` when `'golddiffat15'` is missing is the same as the distribution of `'league'` when `'golddiffat15'` is not missing.
- Hence, we conclude with 99% confidence that the missingness in the `'golddiffat15'` column **does depend** on `'patch'`. So the missingness can be classified as **MAR**.

Hypothesis Testing

- **Null Hypothesis:** The mean absolute gold difference at 15min during WCS games is **the same** as the mean absolute gold difference at 15min during any competitive League game.

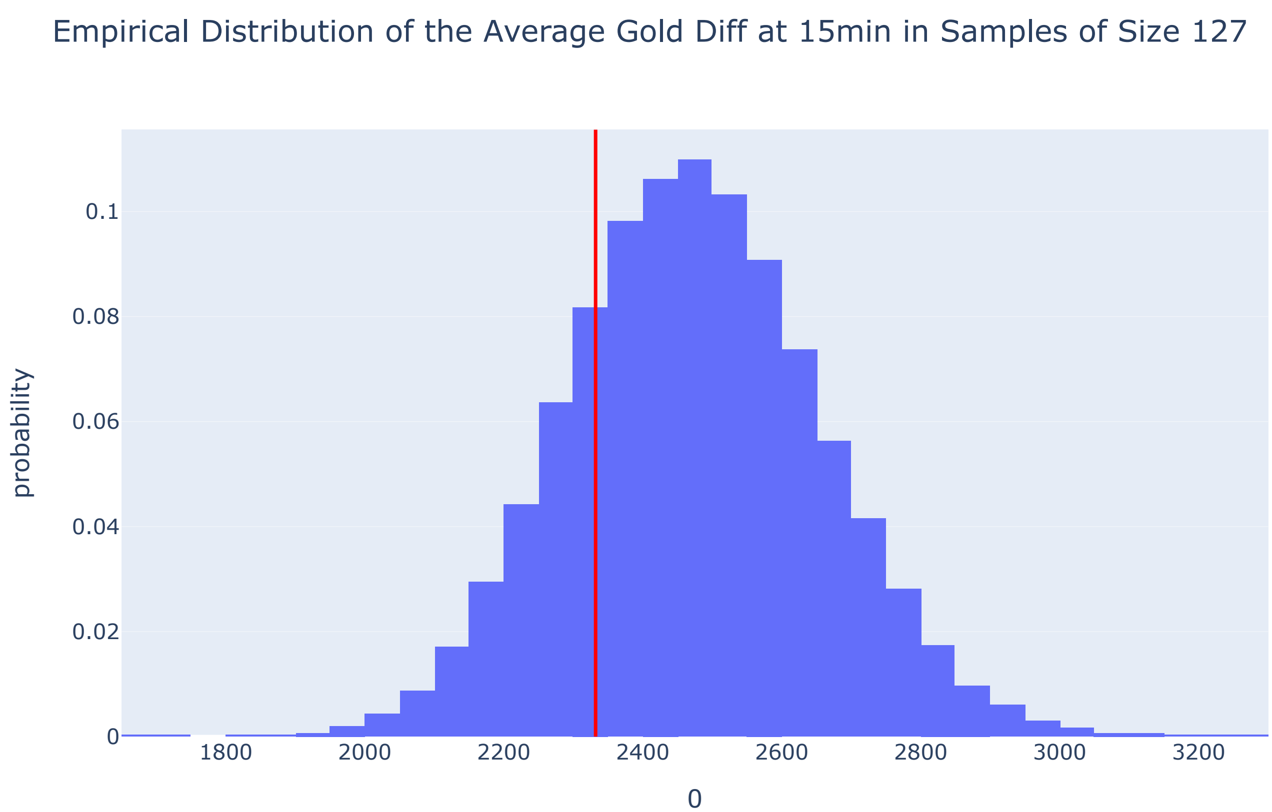
- **Alternative Hypothesis:** The mean absolute gold difference at 15min during WCS games **is less** than the mean absolute gold difference at 15min during any competitive League game.

In [22]:
```
# Observing only wins and non-null data for WCS and all of 2022
wcs_wins = wcs[wcs['result']==1 & (wcs['abs_golddiffat15'].notna())]
df_wins = df[(df['golddiffat15'].notna()) & (df['result']==1)]
```

In [37]:
```
# Hypothesis test
num_reps = 100_000
size = wcs_wins.shape[0]
averages = np.random.choice(df_wins['abs_golddiffat15'], size=(num_reps, size)).mean(axis=1)
observed = wcs_wins['abs_golddiffat15'].mean()
observed
```

```
Out[37]: 2332.0708661417325
```

```
In [25]: fig7 = px.histogram(pd.DataFrame(averages), x=0, nbins=50, histnorm='probability',
                            title='Empirical Distribution of the Average Gold Diff at 15min in Samples of Size 127')
         fig7 = fig7.add_vline(x=observed, line_color='red')
         fig7.show()

         p_value = (averages <= observed).mean()
         print(f'The p-value of our hypothesis test is {p_value}')
```




```
The p-value of our hypothesis test is 0.22147
```

- We fail to reject the null with significance level 0.01.
- We conclude with 99% confidence that the mean absolute gold difference at 15min during WCS games is **the same** as the mean absolute gold difference at 15min during any competitive League game.