Exploratory Data Analysis of Starcraft Dataset

By: Summer Long

Importing dataset

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

sc = pd.read_csv('data/starcraft_player_data.csv')
```

Examining columns for missing values

```
# checking for null and abnormal values
for col in sc.columns:
    print(sc.loc[:, col].sort_values(ascending=False,
na_position='first').head(5))
```

```
3394    10095
3393    10094
3392    10092
3391    10090
3390    10089
Name: GameID, dtype: int64
3394    8
3369    8
3367    8
3366    8
3365    8
Name: LeagueIndex, dtype: int64
3394    ?
3369    ?
3367    ?
3366    ?
3365    ?
Name: Age, dtype: object
3394    ?
3372    ?
3365    ?
3341    ?
3366    ?
Name: HoursPerWeek, dtype: object
3394    ?
3354    ?
3368    ?
3367    ?
3366    ?
Name: TotalHours, dtype: object
734     389.8314
3393    375.8664
3277    372.6426
```

```
3373    364.8504
3372    355.3518
Name: APM, dtype: float64
2746    0.043088
3373    0.042576
3277    0.042258
1810    0.038439
734     0.038416
Name: SelectByHotkeys, dtype: float64
3349    0.001752
3354    0.001750
2870    0.001648
3384    0.001627
3378    0.001569
Name: AssignToHotkeys, dtype: float64
110     10
1884    10
1674    10
3214    10
1654    10
Name: UniqueHotkeys, dtype: int64
197     0.003019
807     0.001974
1109    0.001576
877     0.001530
921     0.001489
Name: MinimapAttacks, dtype: float64
3362    0.004041
1370    0.003688
3344    0.003552
3370    0.003328
2807    0.003030
Name: MinimapRightClicks, dtype: float64
687     0.007971
3362    0.007780
3355    0.007569
1846    0.007191
3343    0.007111
Name: NumberOfPACs, dtype: float64
1358    237.1429
2950    160.9535
1502    156.6234
455     154.8000
1542    150.2857
Name: GapBetweenPACs, dtype: float64
1542    176.3721
638     173.5556
2950    168.9249
597     165.5686
3155    165.1613
```

```
Name: ActionLatency, dtype: float64
2555    18.5581
308     17.7619
2536    17.7059
3180    14.6071
1661    14.3433
Name: ActionsInPAC, dtype: float64
911     58
2561    56
2588    55
925     53
2901    53
Name: TotalMapExplored, dtype: int64
2190    0.005149
3033    0.004307
2158    0.004120
958     0.004025
1762    0.003845
Name: WorkersMade, dtype: float64
925     13
1226    13
2114    13
964     13
2551    12
Name: UniqueUnitsMade, dtype: int64
918     0.000902
3009    0.000786
1927    0.000781
467     0.000677
1567    0.000673
Name: ComplexUnitsMade, dtype: float64
2546    0.003084
2594    0.002685
3144    0.002664
922     0.002443
2459    0.002351
Name: ComplexAbilitiesUsed, dtype: float64
```

```
question_mark_counts = (sc == '?').sum()
print(question_mark_counts)
```

```
GameID                   0
LeagueIndex              0
Age                     55
HoursPerWeek            56
TotalHours              57
APM                      0
SelectByHotkeys          0
AssignToHotkeys          0
UniqueHotkeys            0
MinimapAttacks           0
```

```
MinimapRightClicks       0
NumberOfPACs             0
GapBetweenPACs           0
ActionLatency            0
ActionsInPAC             0
TotalMapExplored         0
WorkersMade              0
UniqueUnitsMade          0
ComplexUnitsMade         0
ComplexAbilitiesUsed     0
Rank                     0
dtype: int64
```

Age, HoursPerWeek, and TotalHours all have '?' values. These are treated as 'missing' or 'unknown'.

```
# All rows that are missing age are also missing HPW & TH

print(sc.loc[:, ['Age', 'HoursPerWeek',
'TotalHours']].eq('?').all(axis=1).sum())
```

```
55
```

```
# One row is missing both, one is missing only total hours
# both players are Diamond

sc.loc[(sc.loc[:, 'Age'] != '?') & ((sc.loc[:, 'HoursPerWeek'] == '?')
| (sc.loc[:, 'TotalHours'] == '?')), :]
```

```
      GameID  LeagueIndex Age HoursPerWeek TotalHours       APM  \
358     1064            5  17           20          ?   94.4724
1841    5255            5  18            ?          ?  122.2470

      SelectByHotkeys  AssignToHotkeys  UniqueHotkeys  MinimapAttacks
...  \
358          0.003846         0.000783              3        0.000010
...
1841         0.006357         0.000433              3        0.000014
...

      NumberOfPACs  GapBetweenPACs  ActionLatency  ActionsInPAC  \
358       0.004474         50.5455        54.9287        3.0972
1841      0.003043         30.8929        62.2933        5.3822

      TotalMapExplored  WorkersMade  UniqueUnitsMade  ComplexUnitsMade
\
358                 31     0.000763                7          0.000106

1841                23     0.001055                5          0.000000
```

```
      ComplexAbilitiesUsed     Rank
358               0.000116  Diamond
1841              0.000338  Diamond

[2 rows x 21 columns]
```

```
sc.loc[(sc.loc[:, 'Age'] == '?') & (sc.loc[:, 'HoursPerWeek'] == '?')
& (sc.loc[:, 'TotalHours'] == '?'), :]
```

```
      GameID  LeagueIndex Age HoursPerWeek TotalHours       APM  \
3340   10001            8   ?            ?          ?  189.7404
3341   10005            8   ?            ?          ?  287.8128
3342   10006            8   ?            ?          ?  294.0996
3343   10015            8   ?            ?          ?  274.2552
3344   10016            8   ?            ?          ?  274.3404
3345   10017            8   ?            ?          ?  245.8188
3346   10018            8   ?            ?          ?  211.0722
3347   10021            8   ?            ?          ?  189.5778
3348   10022            8   ?            ?          ?  210.5088
3349   10023            8   ?            ?          ?  248.0118
3350   10024            8   ?            ?          ?  299.2290
3351   10025            8   ?            ?          ?  179.9982
3352   10026            8   ?            ?          ?  340.1982
3353   10028            8   ?            ?          ?  319.7148
3354   10029            8   ?            ?          ?  290.5914
3355   10030            8   ?            ?          ?  275.8632
3356   10035            8   ?            ?          ?  298.7916
3357   10036            8   ?            ?          ?  325.1154
3358   10038            8   ?            ?          ?  146.3892
3359   10039            8   ?            ?          ?  192.4554
3360   10041            8   ?            ?          ?  315.6936
3361   10045            8   ?            ?          ?  203.7726
3362   10046            8   ?            ?          ?  334.5240
3363   10047            8   ?            ?          ?  175.5936
3364   10049            8   ?            ?          ?  252.7206
3365   10050            8   ?            ?          ?  211.9188
3366   10051            8   ?            ?          ?  269.8998
3367   10052            8   ?            ?          ?  190.2396
3368   10055            8   ?            ?          ?  212.4972
3369   10059            8   ?            ?          ?  219.3894
3370   10060            8   ?            ?          ?  230.6694
3371   10061            8   ?            ?          ?  284.2296
3372   10062            8   ?            ?          ?  355.3518
3373   10063            8   ?            ?          ?  364.8504
3374   10064            8   ?            ?          ?  256.5888
3375   10065            8   ?            ?          ?  248.4012
3376   10066            8   ?            ?          ?  251.2284
3377   10067            8   ?            ?          ?  318.3000
3378   10068            8   ?            ?          ?  288.9198
```

3379	10069	8	?	?	?	313.9080
3380	10072	8	?	?	?	243.7134
3381	10073	8	?	?	?	312.9804
3382	10074	8	?	?	?	313.5762
3383	10075	8	?	?	?	274.6194
3384	10076	8	?	?	?	225.0678
3385	10079	8	?	?	?	254.2188
3386	10081	8	?	?	?	339.1524
3387	10082	8	?	?	?	310.0416
3388	10083	8	?	?	?	288.7608
3389	10084	8	?	?	?	151.4046
3390	10089	8	?	?	?	259.6296
3391	10090	8	?	?	?	314.6700
3392	10092	8	?	?	?	299.4282
3393	10094	8	?	?	?	375.8664
3394	10095	8	?	?	?	348.3576

	SelectByHotkeys	AssignToHotkeys	UniqueHotkeys	MinimapAttacks	...	\
3340	0.004582	0.000655	4	0.000073	...	
3341	0.029040	0.001041	9	0.000231	...	
3342	0.029640	0.001076	6	0.000302	...	
3343	0.018121	0.001264	8	0.000053	...	
3344	0.023131	0.000739	8	0.000622	...	
3345	0.010471	0.000841	10	0.000657	...	
3346	0.013049	0.000940	10	0.000366	...	
3347	0.007559	0.000487	10	0.000606	...	
3348	0.007974	0.000867	7	0.000548	...	
3349	0.014722	0.001752	7	0.000375	...	
3350	0.026428	0.000951	10	0.000155	...	
3351	0.009524	0.001052	6	0.000000	...	
3352	0.028214	0.001242	8	0.000519	...	
3353	0.037130	0.000820	5	0.000403	...	
3354	0.027561	0.001750	6	0.000022	...	
3355	0.019502	0.001449	10	0.000306		

...
3356 0.023253 0.000659 4 0.000433
...
3357 0.029790 0.001338 10 0.000059
...
3358 0.006701 0.000400 10 0.000883
...
3359 0.014277 0.000466 4 0.000000
...
3360 0.028311 0.001160 10 0.001242
...
3361 0.008337 0.000573 5 0.000614
...
3362 0.017742 0.001548 6 0.000384
...
3363 0.012680 0.000934 9 0.000098
...
3364 0.019097 0.001522 6 0.000384
...
3365 0.019817 0.000633 4 0.000201
...
3366 0.024645 0.000642 10 0.000415
...
3367 0.008720 0.000879 10 0.000171
...
3368 0.014917 0.000767 10 0.000599
...
3369 0.005926 0.000741 6 0.000440
...
3370 0.010383 0.001242 10 0.000375
...
3371 0.016069 0.000711 9 0.000355
...
3372 0.037526 0.000600 7 0.001242
...
3373 0.042576 0.000996 8 0.000176
...
3374 0.019592 0.000580 8 0.000416
...
3375 0.016018 0.000874 9 0.000388
...
3376 0.022910 0.000946 5 0.001097
...
3377 0.034851 0.000933 7 0.000187
...
3378 0.029322 0.001569 6 0.000118
...
3379 0.019537 0.001214 4 0.000318
...
3380 0.017195 0.000711 6 0.000666

...
3381 0.026327 0.000266 6 0.000000
...
3382 0.030550 0.000560 5 0.000000
...
3383 0.022497 0.000707 6 0.000163
...
3384 0.014339 0.001627 7 0.000291
...
3385 0.016608 0.000788 6 0.000926
...
3386 0.033058 0.001017 10 0.000477
...
3387 0.026873 0.001278 10 0.000319
...
3388 0.024022 0.000628 6 0.000350
...
3389 0.009732 0.000949 6 0.000028
...
3390 0.020425 0.000743 9 0.000621
...
3391 0.028043 0.001157 10 0.000246
...
3392 0.028341 0.000860 7 0.000338
...
3393 0.036436 0.000594 5 0.000204
...
3394 0.029855 0.000811 4 0.000224
...

	NumberOfPACs	GapBetweenPACs	ActionLatency	ActionsInPAC	\
3340	0.006291	23.5130	32.5665	4.4451	
3341	0.005399	31.6416	36.1143	4.5893	
3342	0.006294	16.6393	36.8192	4.1850	
3343	0.007111	10.6419	24.3556	4.3870	
3344	0.005355	19.1568	36.3098	5.2811	
3345	0.005031	14.5518	36.7134	7.1943	
3346	0.003719	19.6169	38.9326	7.1320	
3347	0.005821	22.0317	36.7330	4.9050	
3348	0.006518	15.7856	30.7156	4.8058	
3349	0.004115	17.4656	34.2357	7.8973	
3350	0.005443	17.0835	33.7398	5.2703	
3351	0.003567	32.5628	39.5600	7.0050	
3352	0.006898	15.2852	26.6907	5.1293	
3353	0.005208	35.4127	44.0552	4.4282	
3354	0.005293	22.0126	36.0669	4.9540	
3355	0.007569	18.1407	24.0936	4.1723	
3356	0.005561	16.0743	29.2593	5.8444	
3357	0.005381	15.4571	40.3646	5.7652	
3358	0.003617	18.4444	47.3364	5.8341	

3359	0.003142	29.7500	35.7531	7.1975
3360	0.005076	17.7035	32.6344	6.2231
3361	0.005954	11.3597	31.1615	5.1082
3362	0.007780	13.5401	28.2243	5.6862
3363	0.005265	27.1322	43.7278	3.8371
3364	0.004090	21.6151	38.2256	6.8534
3365	0.003912	31.8222	54.5588	5.0294
3366	0.004015	25.6352	43.3856	6.4922
3367	0.004971	17.9901	35.9509	5.5872
3368	0.005648	21.6687	41.2231	4.4680
3369	0.005185	17.0456	30.5342	6.6749
3370	0.006375	13.5028	31.4044	5.0533
3371	0.006680	9.4756	29.6851	5.3326
3372	0.004541	9.2871	41.9497	6.5063
3373	0.004687	19.9499	41.1417	5.6167
3374	0.005812	17.0462	34.3734	5.0563
3375	0.005987	16.3144	30.2486	5.0973
3376	0.005411	13.7404	35.7203	4.5524
3377	0.005225	26.0987	32.4464	4.8705
3378	0.005213	23.2857	32.8026	4.7540
3379	0.005879	8.1642	26.0918	6.7885
3380	0.005594	21.8795	30.5722	5.1136
3381	0.005053	14.6118	30.7836	6.1930
3382	0.004390	19.5405	35.4094	6.4228
3383	0.004053	20.6757	32.7785	6.9262
3384	0.005281	16.3502	33.2874	5.4713
3385	0.005408	14.9191	35.9921	5.7205
3386	0.004609	21.6389	37.1862	6.7103
3387	0.005517	16.5446	33.8174	5.7350
3388	0.005580	19.0108	30.0866	5.3831
3389	0.004363	27.4658	43.8052	4.3312
3390	0.004555	18.6059	42.8342	6.2754
3391	0.004259	14.3023	36.1156	7.1965
3392	0.004439	12.4028	39.5156	6.3979
3393	0.004346	11.6910	34.8547	7.9615
3394	0.005566	20.0537	33.5142	6.3719

	TotalMapExplored	WorkersMade	UniqueUnitsMade	ComplexUnitsMade
3340	25	0.002218	6	0.000000
3341	34	0.001138	6	0.000058
3342	26	0.000987	6	0.000000
3343	28	0.001106	6	0.000000
3344	28	0.000739	6	0.000000
3345	33	0.001474	11	0.000040

3346	23	0.000898	9	0.000000
3347	28	0.000540	5	0.000000
3348	34	0.000817	6	0.000000
3349	20	0.001111	8	0.000000
3350	16	0.000697	6	0.000033
3351	13	0.000999	6	0.000000
3352	26	0.001535	8	0.000000
3353	26	0.000892	6	0.000245
3354	19	0.000642	6	0.000044
3355	15	0.001031	5	0.000000
3356	19	0.000783	6	0.000000
3357	22	0.000907	7	0.000000
3358	17	0.000950	8	0.000017
3359	11	0.001280	3	0.000000
3360	24	0.000791	6	0.000000
3361	23	0.000859	7	0.000000
3362	29	0.002161	9	0.000145
3363	24	0.000575	5	0.000000
3364	23	0.000523	5	0.000000
3365	14	0.001409	3	0.000000
3366	21	0.000478	6	0.000000
3367	21	0.000904	5	0.000000
3368	28	0.001119	9	0.000035
3369	35	0.002072	9	0.000225

3370	32	0.001512	6	0.000035
3371	25	0.002459	7	0.000000
3372	22	0.001228	8	0.000000
3373	18	0.000674	7	0.000000
3374	19	0.001308	7	0.000000
3375	21	0.001197	6	0.000000
3376	22	0.000738	5	0.000000
3377	13	0.000933	4	0.000000
3378	24	0.001130	5	0.000000
3379	26	0.001321	8	0.000106
3380	25	0.000576	8	0.000000
3381	10	0.001802	4	0.000000
3382	12	0.001296	3	0.000000
3383	15	0.000626	3	0.000000
3384	26	0.000898	8	0.000000
3385	28	0.001128	6	0.000000
3386	16	0.001049	3	0.000000
3387	22	0.000922	8	0.000000
3388	30	0.000761	5	0.000000
3389	23	0.000949	6	0.000000
3390	46	0.000877	5	0.000000
3391	16	0.000788	4	0.000000
3392	19	0.001260	4	0.000000
3393	15	0.000613	6	0.000000

3394	27	0.001566	7	0.000457	

	ComplexAbilitiesUsed	Rank
3340	0.000000	Professional leagues
3341	0.000000	Professional leagues
3342	0.000000	Professional leagues
3343	0.000000	Professional leagues
3344	0.000000	Professional leagues
3345	0.000048	Professional leagues
3346	0.000000	Professional leagues
3347	0.000000	Professional leagues
3348	0.000000	Professional leagues
3349	0.000000	Professional leagues
3350	0.000011	Professional leagues
3351	0.000000	Professional leagues
3352	0.000113	Professional leagues
3353	0.000144	Professional leagues
3354	0.000078	Professional leagues
3355	0.000000	Professional leagues
3356	0.000309	Professional leagues
3357	0.000000	Professional leagues
3358	0.000167	Professional leagues
3359	0.000000	Professional leagues
3360	0.000150	Professional leagues
3361	0.000000	Professional leagues
3362	0.000073	Professional leagues
3363	0.000000	Professional leagues
3364	0.000323	Professional leagues
3365	0.000000	Professional leagues
3366	0.000579	Professional leagues
3367	0.000000	Professional leagues
3368	0.000062	Professional leagues
3369	0.000064	Professional leagues
3370	0.000047	Professional leagues
3371	0.000000	Professional leagues
3372	0.000614	Professional leagues
3373	0.000000	Professional leagues
3374	0.000000	Professional leagues
3375	0.000000	Professional leagues
3376	0.000662	Professional leagues
3377	0.000233	Professional leagues
3378	0.000000	Professional leagues
3379	0.000443	Professional leagues
3380	0.000000	Professional leagues
3381	0.000030	Professional leagues
3382	0.000059	Professional leagues
3383	0.000000	Professional leagues
3384	0.000959	Professional leagues

```
3385              0.000000  Professional leagues
3386              0.000000  Professional leagues
3387              0.000000  Professional leagues
3388              0.000652  Professional leagues
3389              0.000099  Professional leagues
3390              0.000000  Professional leagues
3391              0.000000  Professional leagues
3392              0.000000  Professional leagues
3393              0.000631  Professional leagues
3394              0.000895  Professional leagues

[55 rows x 21 columns]
```

```
# looking at dataframe, observe that everyone who is in professional
leagues has unreported age, hrs/week, hrs/total
# confirm below
len(sc[sc['Rank'] == 'Professional leagues'])
```

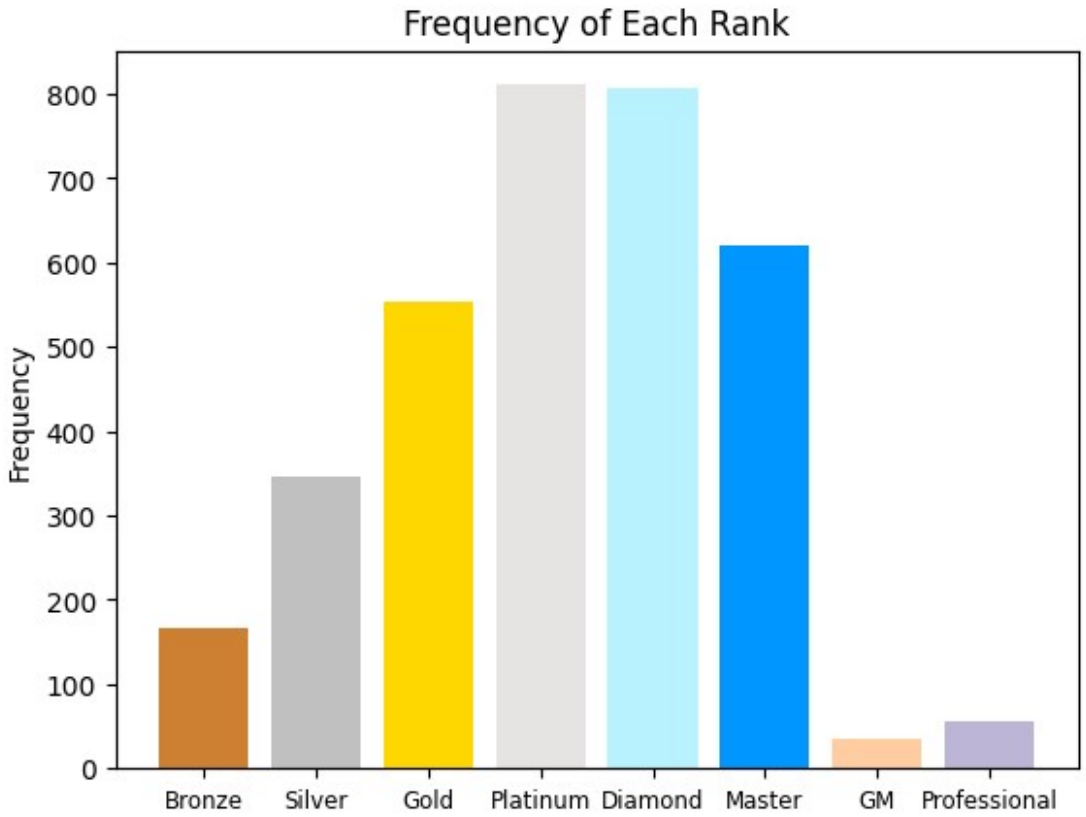
```
55
```

```
## fill '?' with na
sc.replace('?', np.nan, inplace=True)

## cast to float since it was an object due to mix of int and str
sc.loc[:, sc.columns[2:5]] = sc.loc[:,
sc.columns[2:5]].astype('float64')

## dictionary to map string representation to LeagueIndex

rank_dict = {1: 'Bronze',
             2: 'Silver',
             3: 'Gold',
             4: 'Platinum',
             5: 'Diamond',
             6: 'Master',
             7: 'GrandMaster',
             8: 'Professional leagues'}

sc.loc[:, 'Rank'] = sc.loc[:, 'LeagueIndex'].apply(lambda x:
rank_dict.get(x))

## string representation to color for plotting
color_mapping = {
    'Bronze': '#CD7F32',
    'Silver': '#C0C0C0',
    'Gold': '#FFD700',
    'Platinum': '#e5e4e2',
    'Diamond': '#b9f2ff',
    'Master': '#0096FF',
    'GrandMaster': '#FFCDA1',
```

```
    'Professional leagues': '#BDB5D5'
    }
```

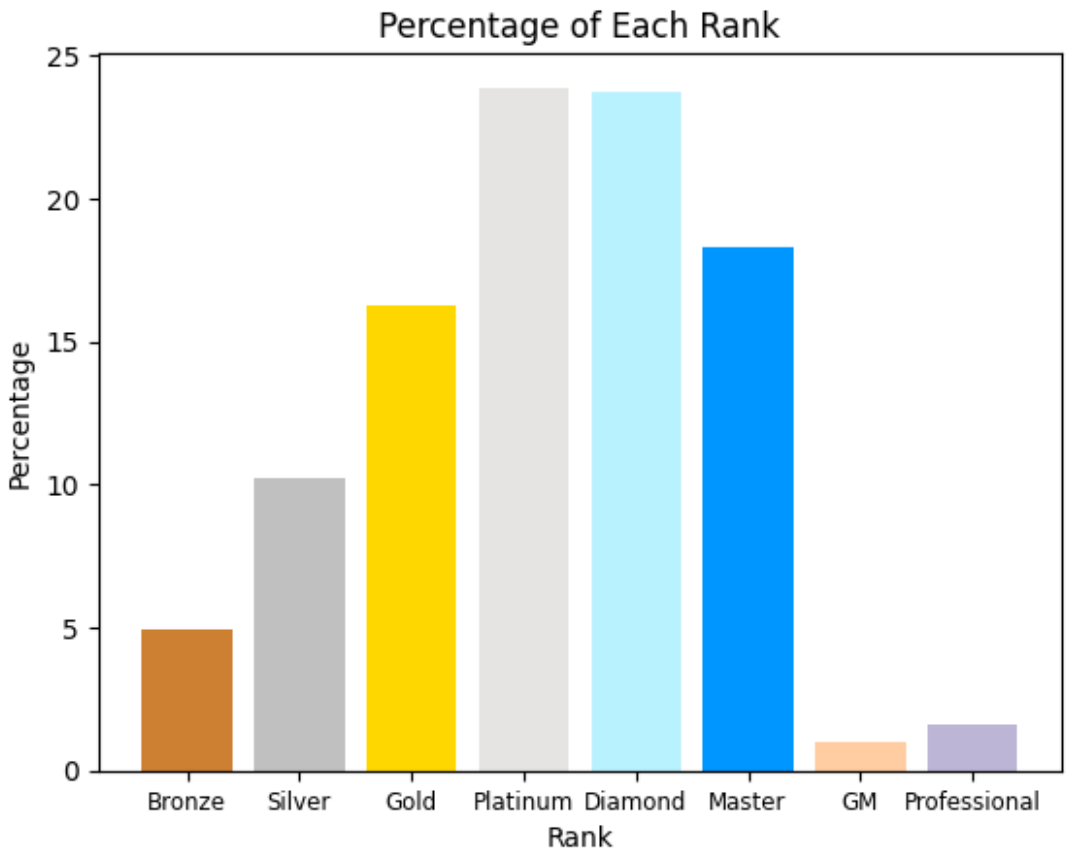
Rank distribution

```
rank_counts = sc['Rank'].value_counts()
rank_counts = rank_counts.loc[list(rank_dict.values())]
bar_colors = [color_mapping.get(label) for label in rank_counts.index]
modified_labels = ['GM' if label == 'GrandMaster' else 'Professional'
if label == 'Professional leagues' else label for label in
rank_counts.index]
plt.bar(modified_labels, rank_counts.values, color=bar_colors)

plt.xticks(fontsize=8.5)

plt.ylabel('Frequency')
plt.title('Frequency of Each Rank')
plt.show()
```




```
percentage = (rank_counts / sc.shape[0]) * 100
plt.bar(modified_labels, percentage, color=bar_colors)
plt.xticks(fontsize=8.5)

plt.xlabel('Rank')
plt.ylabel('Percentage')
plt.title('Percentage of Each Rank')
```

```
plt.show()
```




```
# Platinum is the average rank
sc.loc[:, 'LeagueIndex'].mean()
```

```
4.184094256259205
```

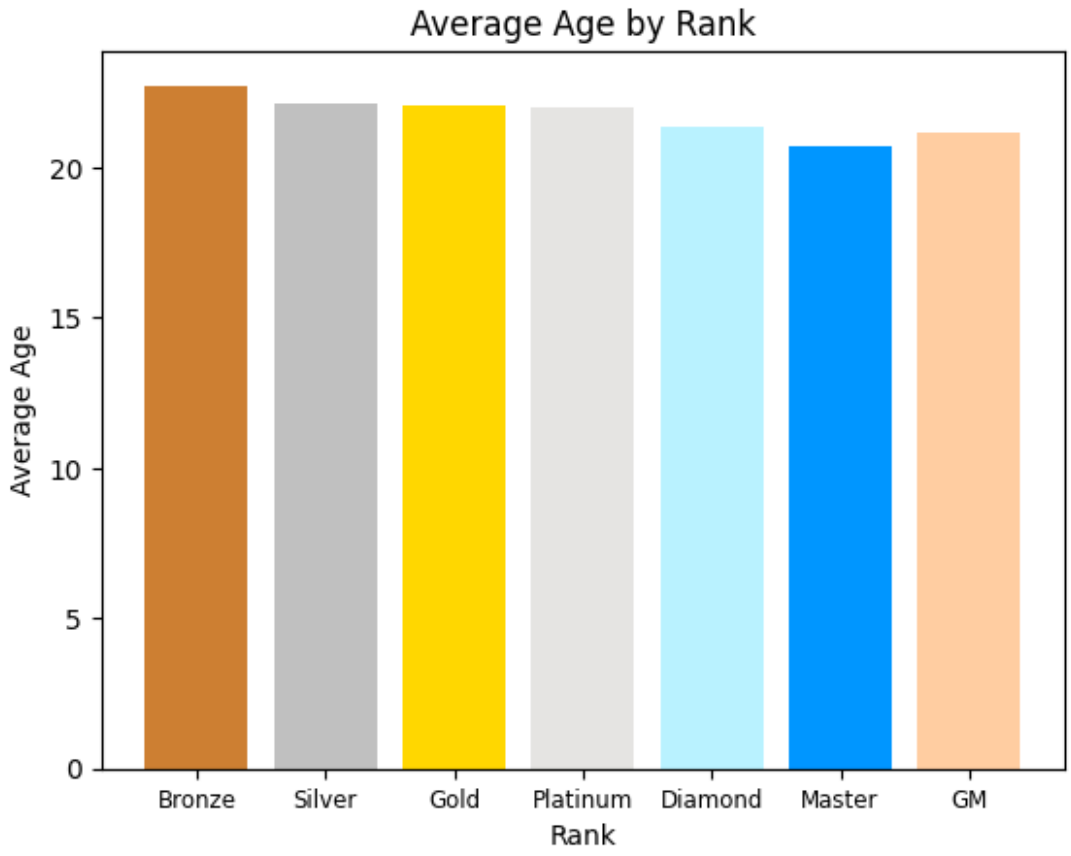
Feature by Target Variable

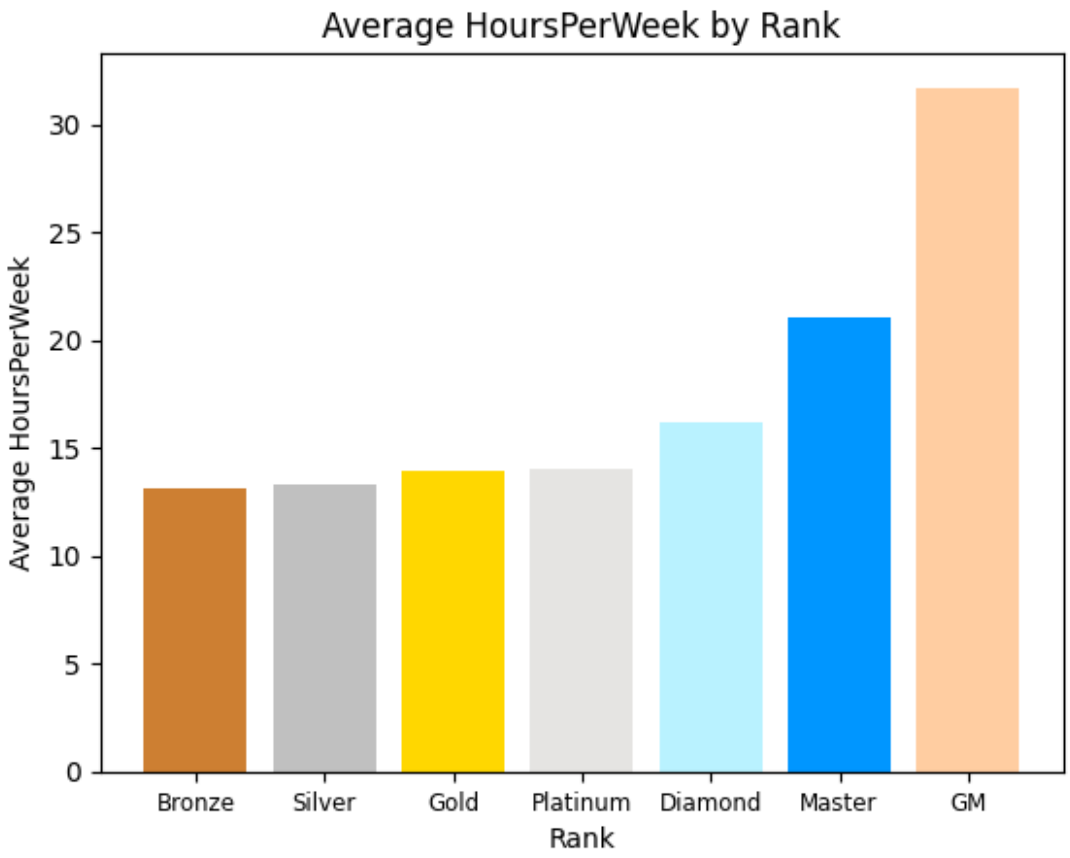
```
## function for plotting by average

def plot_mean_by_rank(df, col):
    avg = df.groupby('Rank')[col].mean()
    avg = avg.loc[list(rank_dict.values())]

    modified_labels = ['GM' if label == 'GrandMaster' else
'Professional' if label == 'Professional leagues' else label for label
in avg.index]

    plt.bar(modified_labels, avg.values,
color=[color_mapping.get(label) for label in avg.index])

    plt.xticks(fontsize=8.5)
    plt.xlabel('Rank')
```

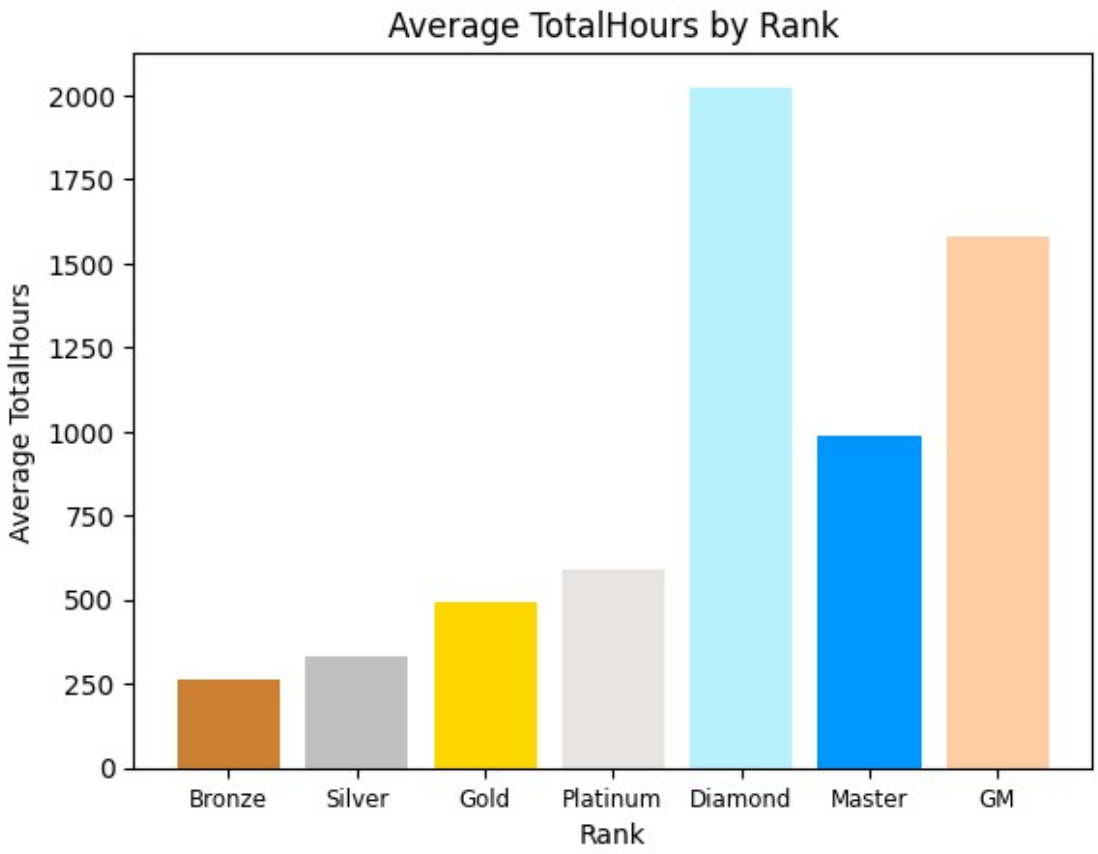
```
    plt.ylabel('Average ' + str(col))
    plt.title('Average ' + str(col) + ' by Rank')
    plt.show()


# see list of colunns to determine slicing
sc.columns
```

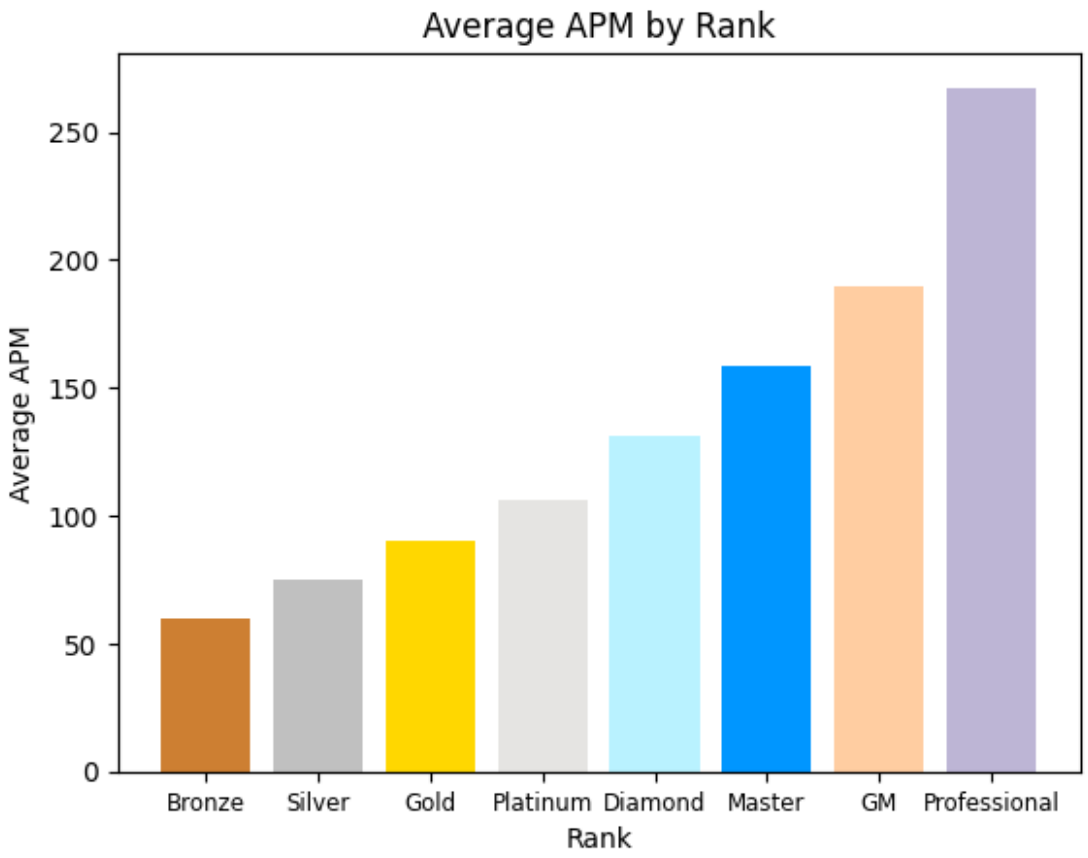
```
Index(['GameID', 'LeagueIndex', 'Age', 'HoursPerWeek', 'TotalHours',
'APM',
       'SelectByHotkeys', 'AssignToHotkeys', 'UniqueHotkeys',
'MinimapAttacks',
       'MinimapRightClicks', 'NumberOfPACs', 'GapBetweenPACs',
'ActionLatency',
       'ActionsInPAC', 'TotalMapExplored', 'WorkersMade',
'UniqueUnitsMade',
       'ComplexUnitsMade', 'ComplexAbilitiesUsed', 'Rank'],
      dtype='object')
```

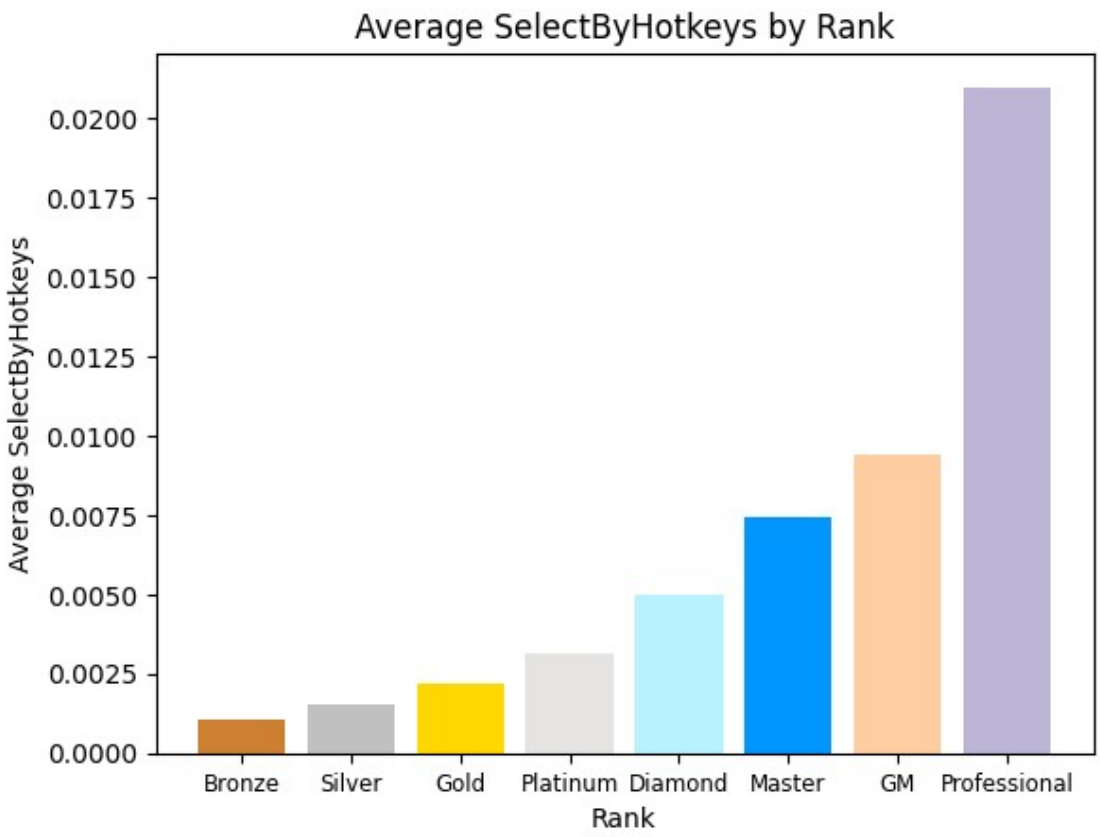
```
for col in sc.columns[2:-1]:
    plot_mean_by_rank(sc, col)
```

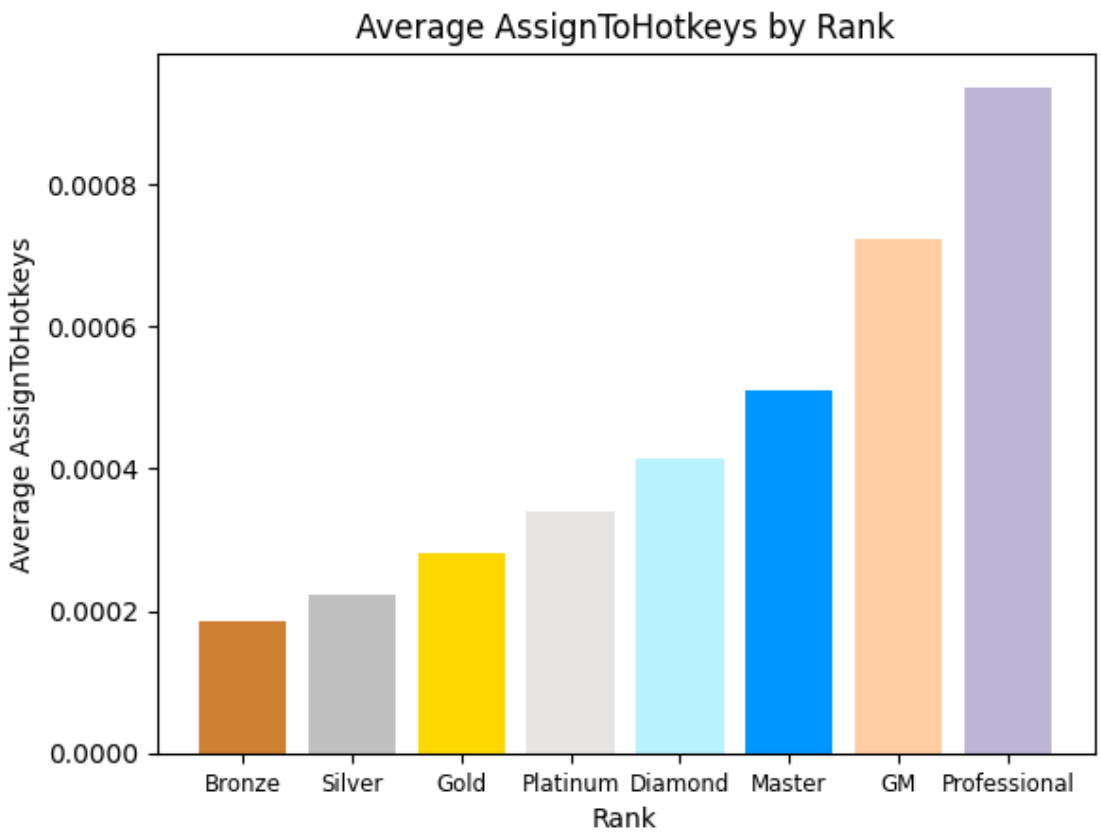


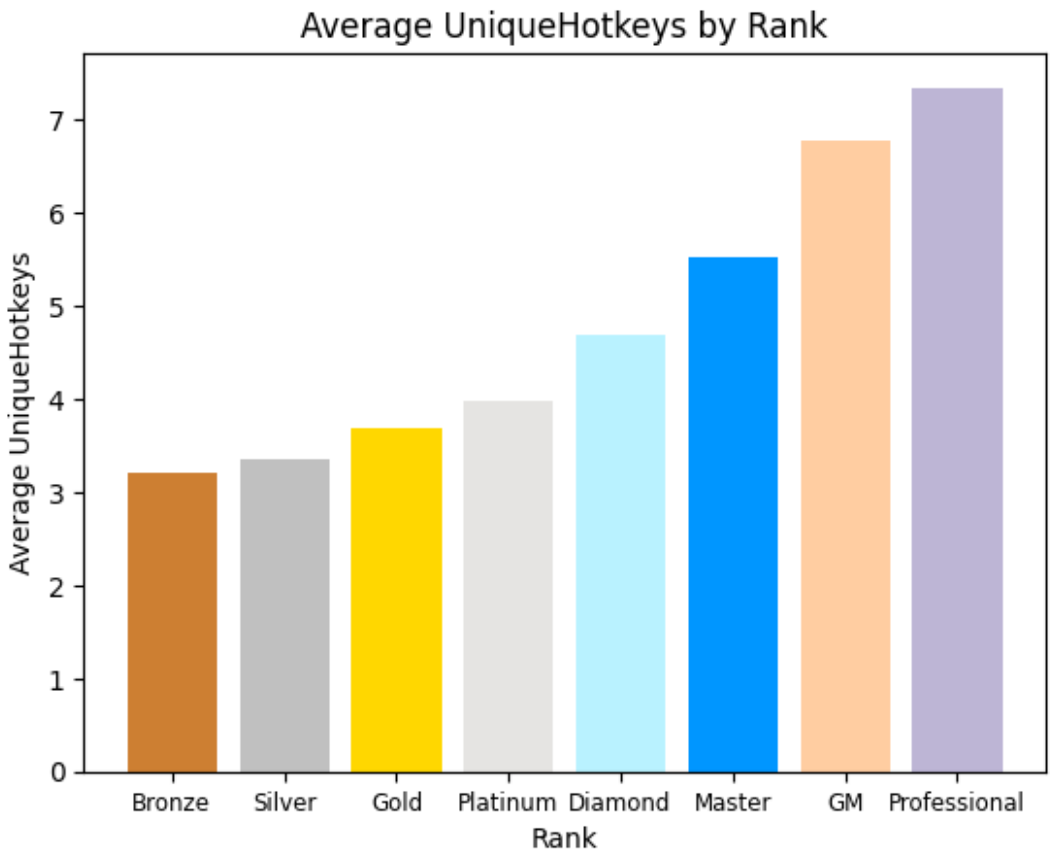

Average HoursPerWeek by Rank
Average HoursPerWeek
0
5
10
15
20
25
30
Bronze
Silver
Gold
Platinum
Diamond
Master
GM
Rank

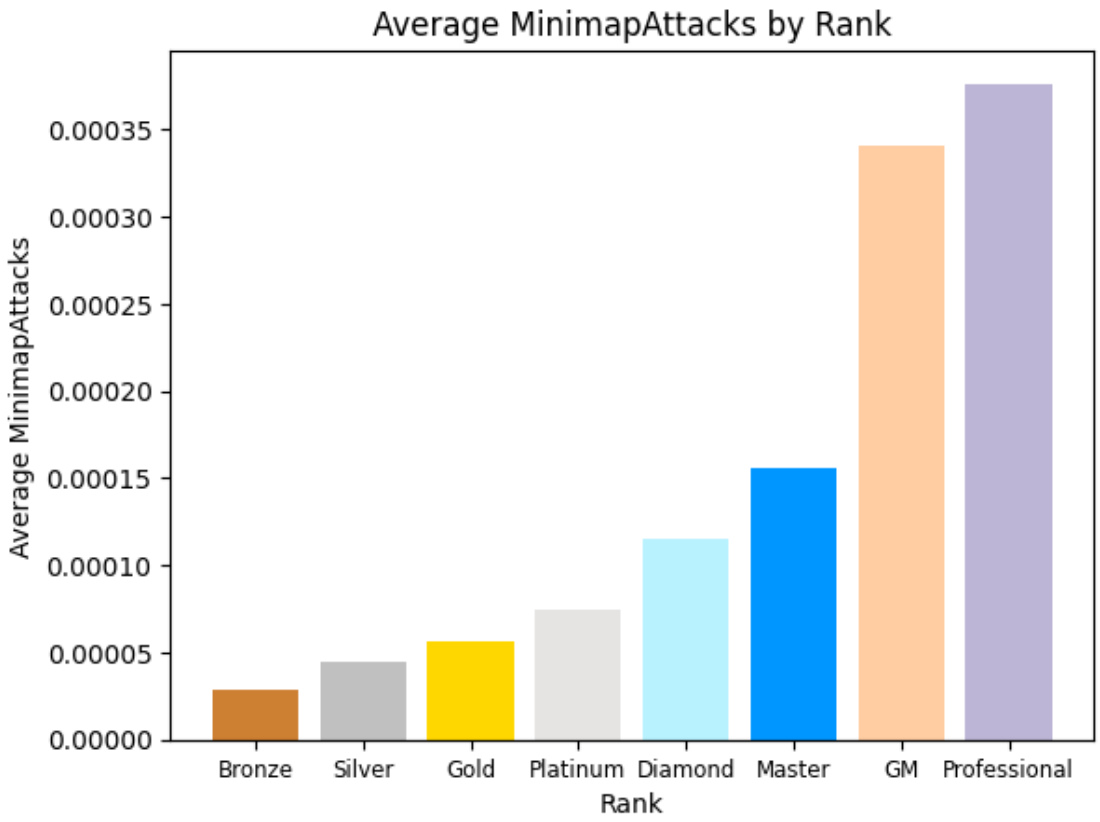

Average TotalHours by Rank
Average TotalHours
2000
1750
1500
1250
1000
750
500
250
0
Bronze
Silver
Gold
Platinum
Diamond
Master
GM
Rank

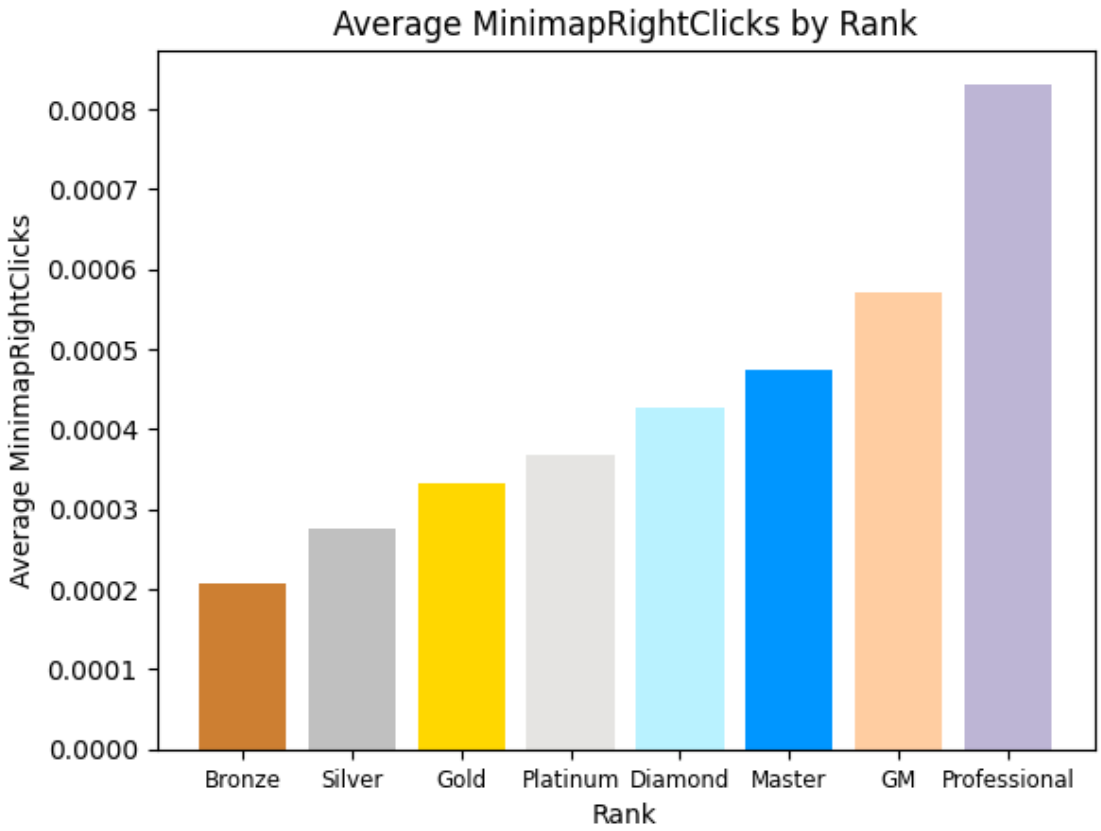

Average APM by Rank
Average APM
0
50
100
150
200
250
Bronze
Silver
Gold
Platinum
Diamond
Master
GM
Professional
Rank

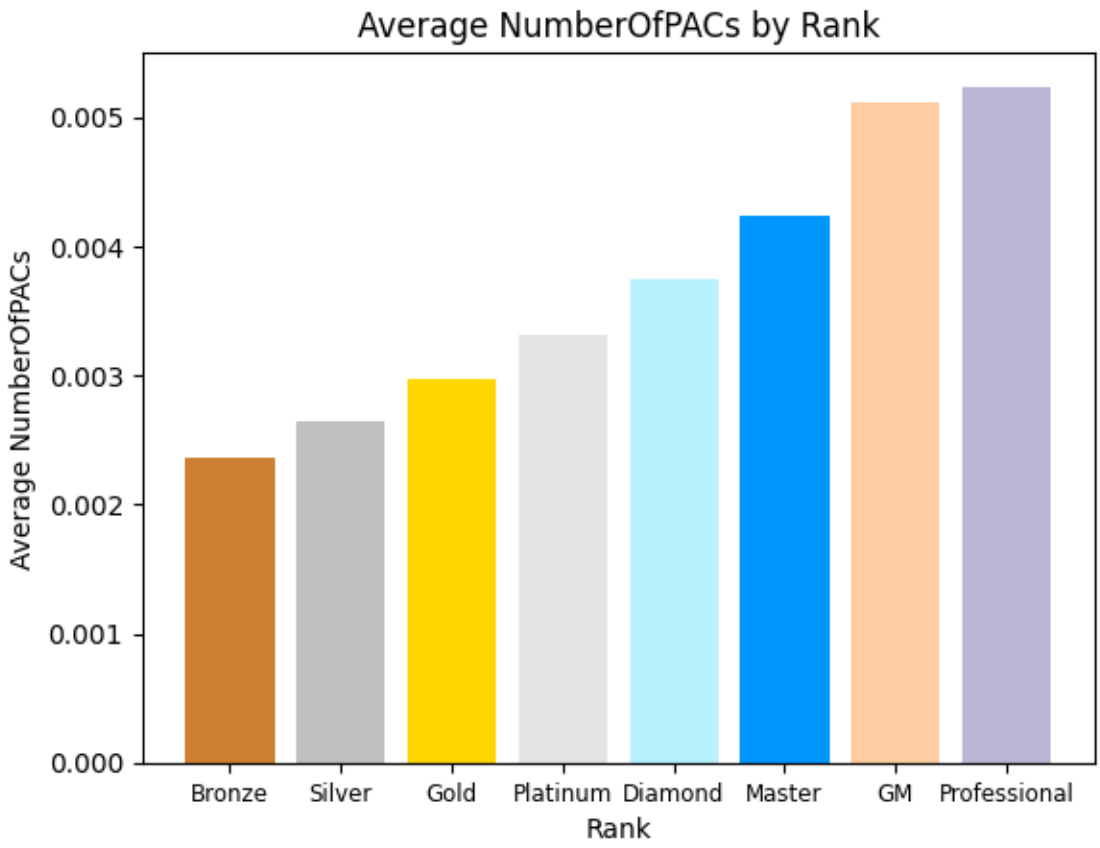

Average SelectByHotkeys by Rank
Average SelectByHotkeys
0.0200
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
Bronze
Silver
Gold
Platinum
Diamond
Master
GM
Professional
Rank

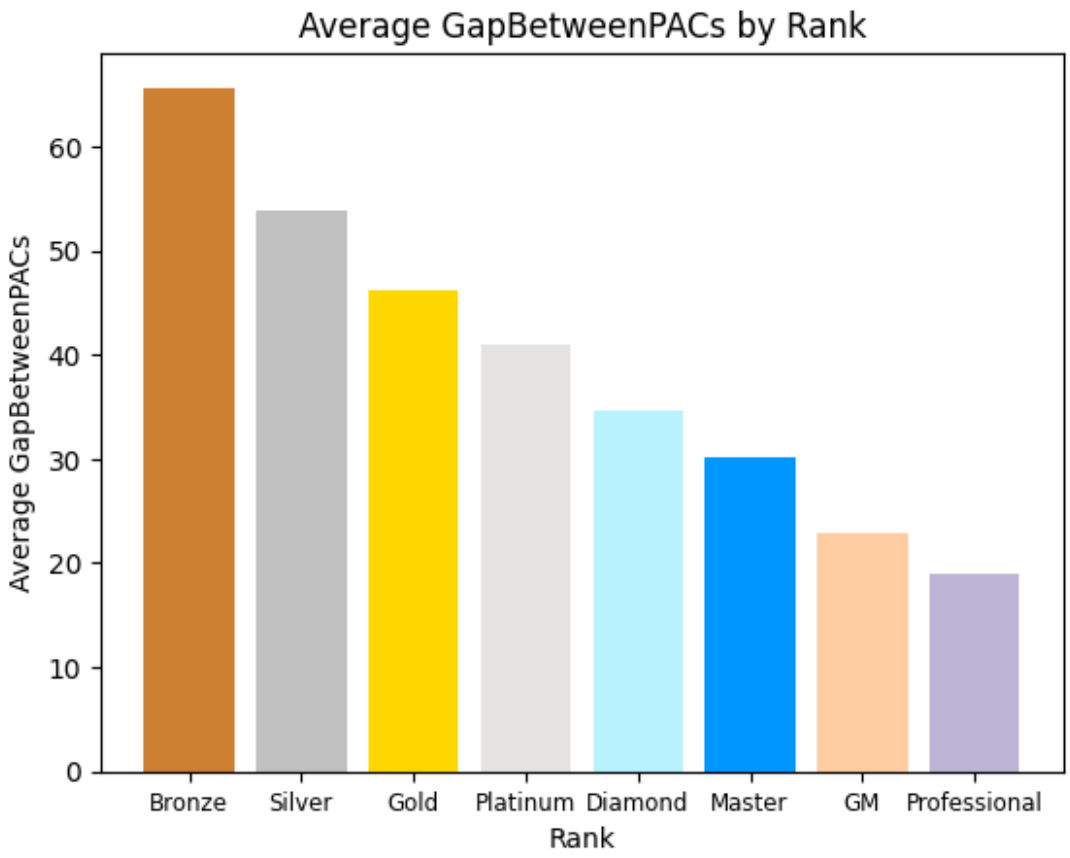

Average AssignToHotkeys by Rank
Average AssignToHotkeys
0.0000
0.0002
0.0004
0.0006
0.0008
Bronze
Silver
Gold
Platinum
Diamond
Master
GM
Professional
Rank

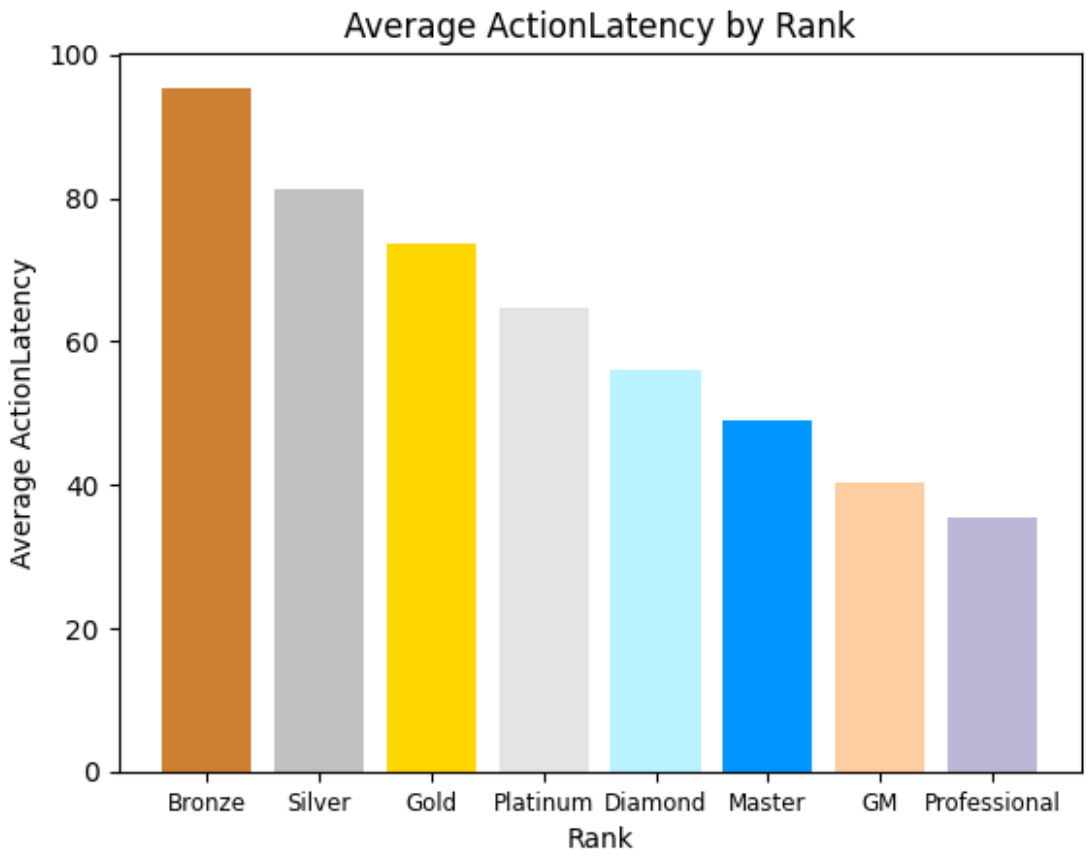

Average UniqueHotkeys by Rank
Average UniqueHotkeys
0
1
2
3
4
5
6
7
Bronze
Silver
Gold
Platinum
Diamond
Master
GM
Professional
Rank

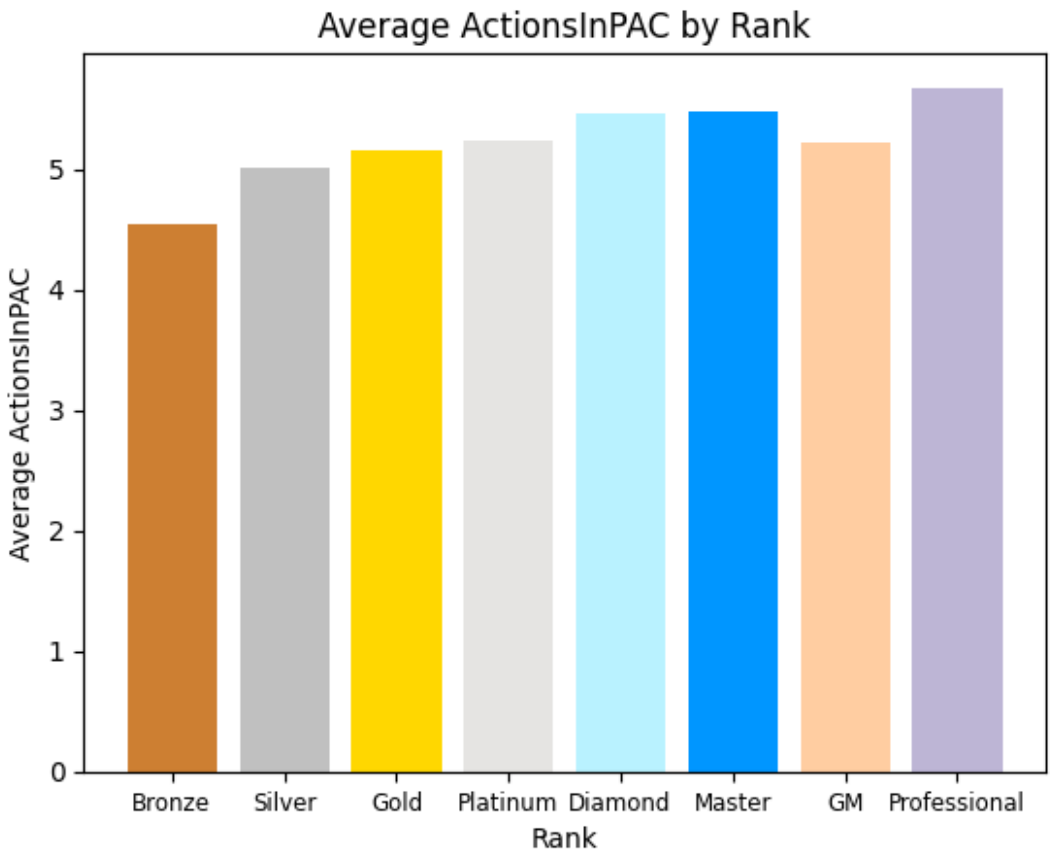

Average MinimapAttacks by Rank
Average MinimapAttacks
0.00035
0.00030
0.00025
0.00020
0.00015
0.00010
0.00005
0.00000
Bronze
Silver
Gold
Platinum
Diamond
Master
GM
Professional
Rank

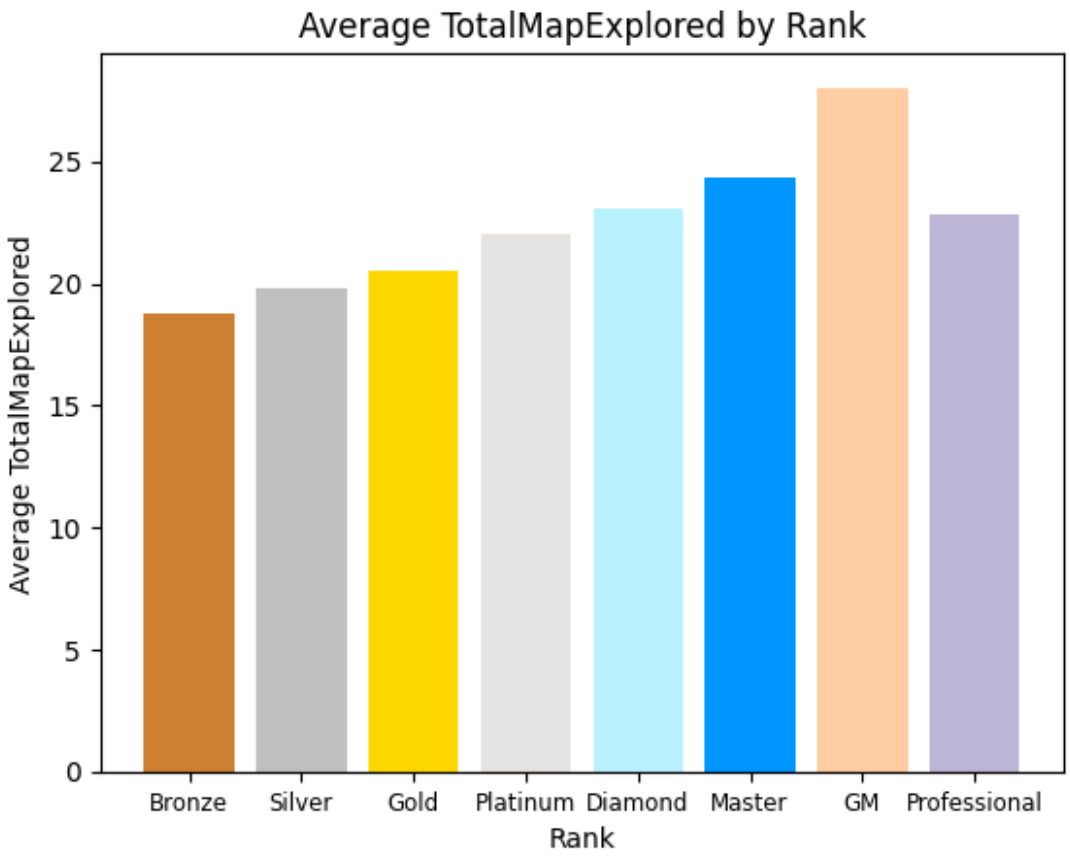

Average MinimapRightClicks by Rank
Average MinimapRightClicks
0.0008
0.0007
0.0006
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
Bronze
Silver
Gold
Platinum
Diamond
Master
GM
Professional
Rank

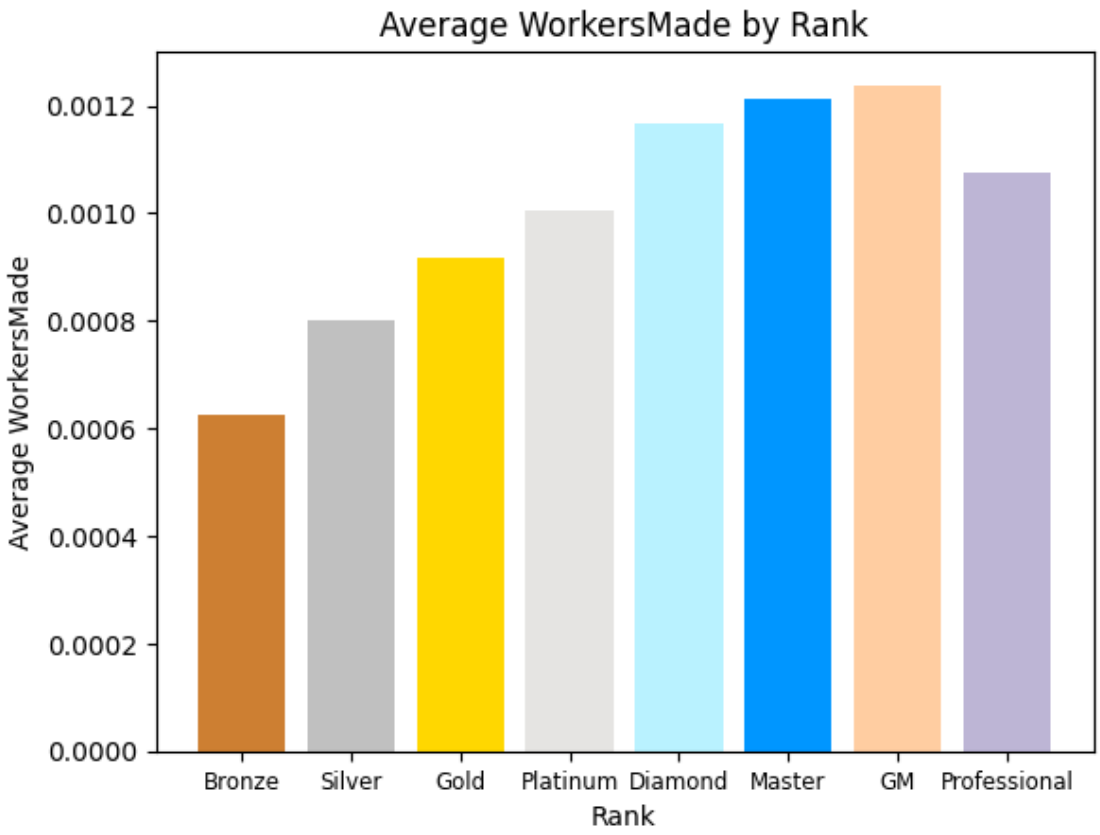

Average NumberOfPACs by Rank
Average NumberOfPACs
0.000
0.001
0.002
0.003
0.004
0.005
Bronze
Silver
Gold
Platinum
Diamond
Master
GM
Professional
Rank

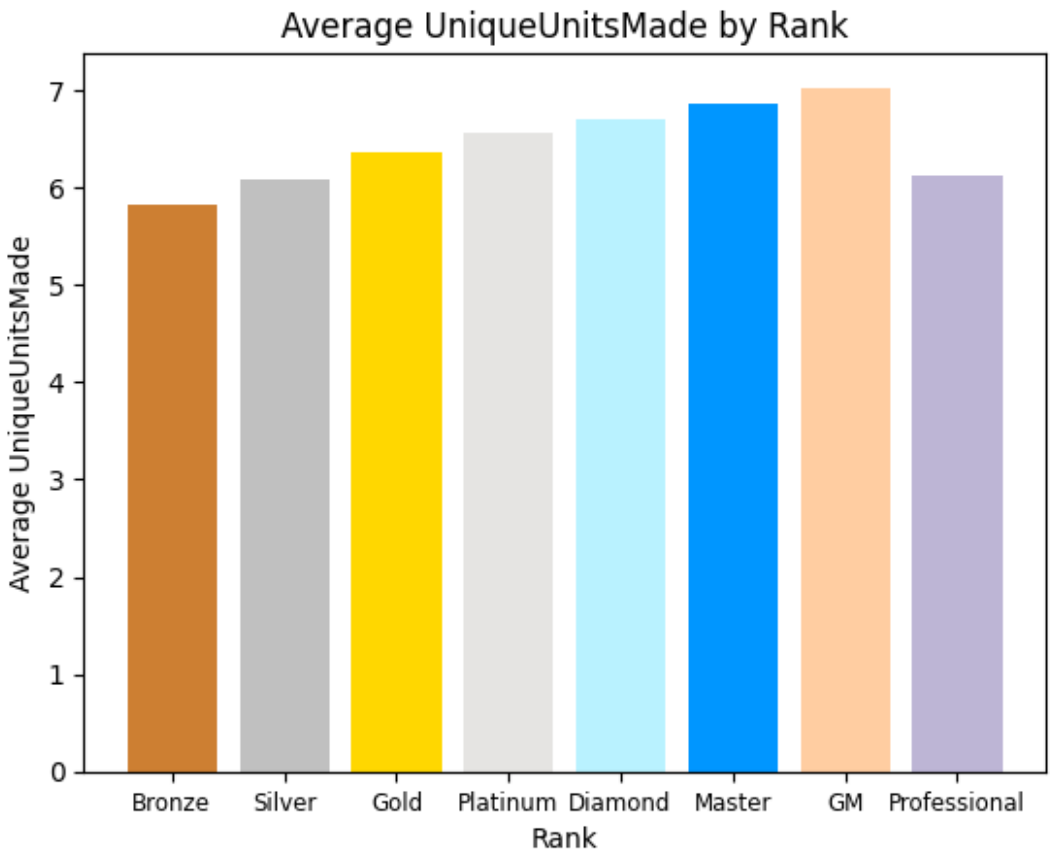

Average GapBetweenPACs by Rank
Average GapBetweenPACs
0
10
20
30
40
50
60
Bronze
Silver
Gold
Platinum
Diamond
Master
GM
Professional
Rank

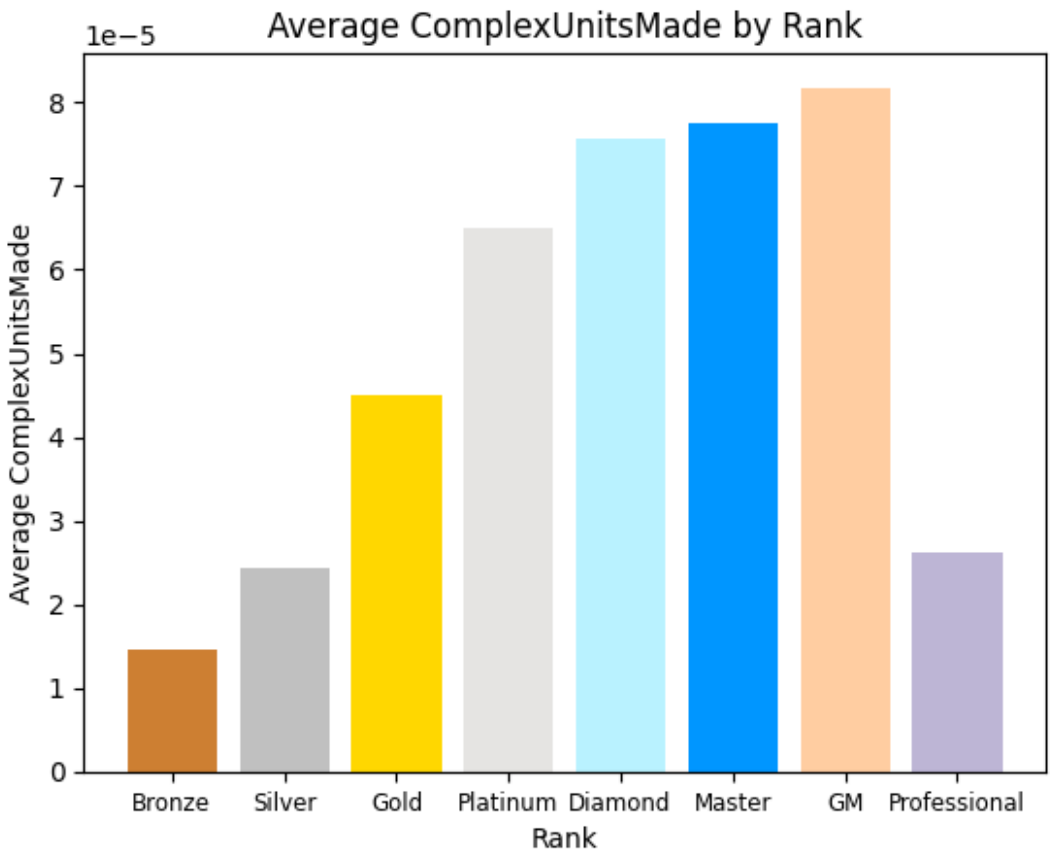

Average ActionLatency by Rank
Average ActionLatency
100
80
60
40
20
0
Bronze
Silver
Gold
Platinum
Diamond
Master
GM
Professional
Rank

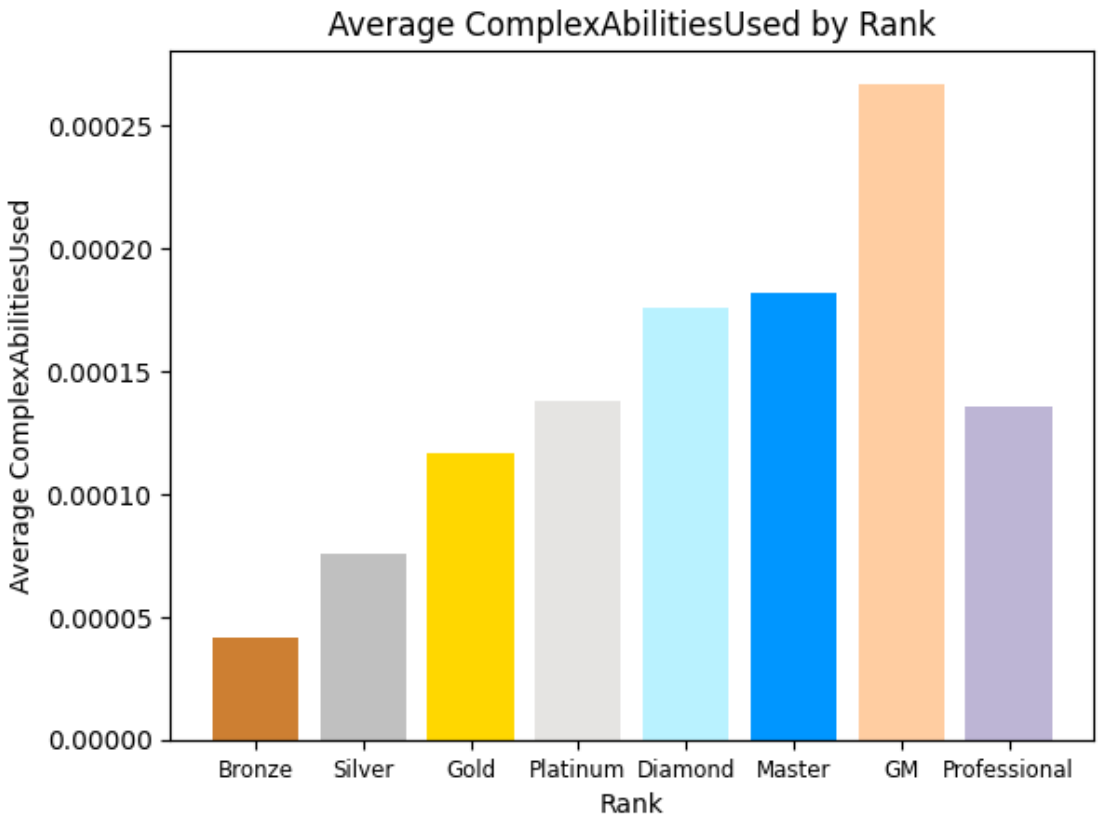

Average ActionsInPAC by Rank
Average ActionsInPAC
0
1
2
3
4
5
Bronze
Silver
Gold
Platinum
Diamond
Master
GM
Professional
Rank


Average TotalMapExplored by Rank
Average TotalMapExplored
0
5
10
15
20
25
Bronze
Silver
Gold
Platinum
Diamond
Master
GM
Professional
Rank


Average WorkersMade by Rank
Average WorkersMade
0.0000
0.0002
0.0004
0.0006
0.0008
0.0010
0.0012
Bronze
Silver
Gold
Platinum
Diamond
Master
GM
Professional
Rank


Average UniqueUnitsMade by Rank
Average UniqueUnitsMade
0
1
2
3
4
5
6
7
Bronze
Silver
Gold
Platinum
Diamond
Master
GM
Professional
Rank


Average ComplexUnitsMade by Rank
1e−5
Average ComplexUnitsMade
0
1
2
3
4
5
6
7
8
Bronze
Silver
Gold
Platinum
Diamond
Master
GM
Professional
Rank



Preliminarily, the following features will be excluded when building the model:

- Age
- Hours
- Total Hours
- Unique Units Made
- Actions in PAC