Analyzing matching strategies for gaining more Matchmaking Ranking in Dota2
(Yuxiao Ye, Bin Zhang)

I. Research Questions(Yuxiao Ye, Bin Zhang)

1. **What are some traits that winning teams share?**
   What are the main differences between the winning team and the losing team? We are going to analyze each match in the dataset and compare the statistics of the winner team and loser team and to figure out what traits are commonly shared in winner teams' features as the result we want.
   - Answer: according to our data analysis for 20000 games, winner teams tend to deal more damage to enemy teams, make more kills and assists. Also, the winning team tends to have more gold than loser teams

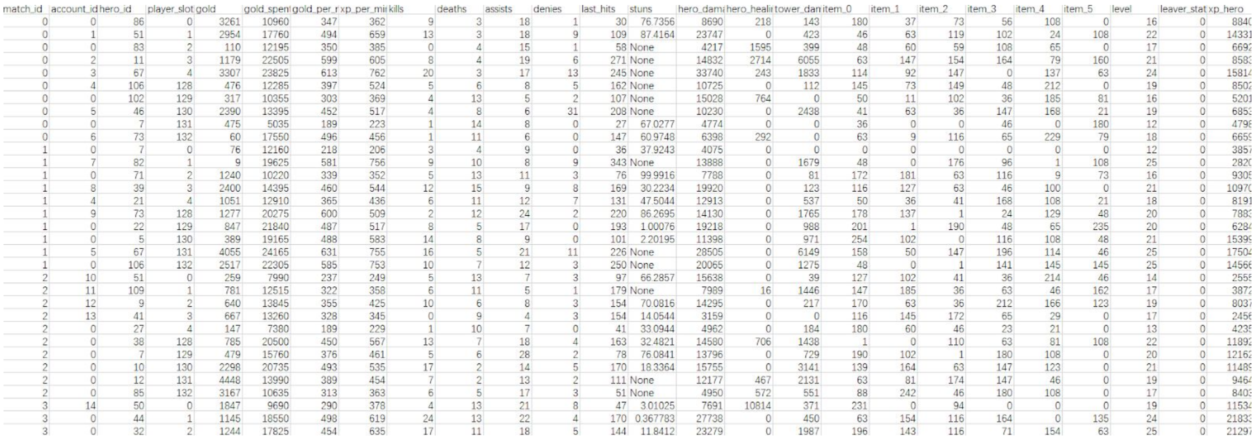
2. **How to predict the number of kills by each player?**
   Killing the enemy hero is the charm of the MOBA games like Dota2. So it is valuable to test out what are some factors that will affect the number of kills for each game for a player.
   - Answer: according to our data analysis, RandomForest forecasts the number of kills by each player with around 85% accuracy in the testing set. The most significant contributors to the results are, with descending order of feature importance, hero damages, gold per minute, last hits, assists, gold spent, etc. Hero damages account for more than 70% of the feature importance among 210 variables in terms of usefulness in the prediction.

3. **How does each hero perform on average?**
   Dota2 has more than 110 unique heroes and each game will have ten unique heroes played by ten players. We would like to find out how these heroes perform in Dota matches on average. After, we can recommend heroes for a newbie to play or find a hero to play in a ranked game
   - Answer: according to our data analysis, we categories those heroes based on the number of appearances, gold spent, kills, assists, deaths, then found out the "most" in each category. And based on this data we are able to give recommendations to players. See more in the "Result" section

II. DataSet

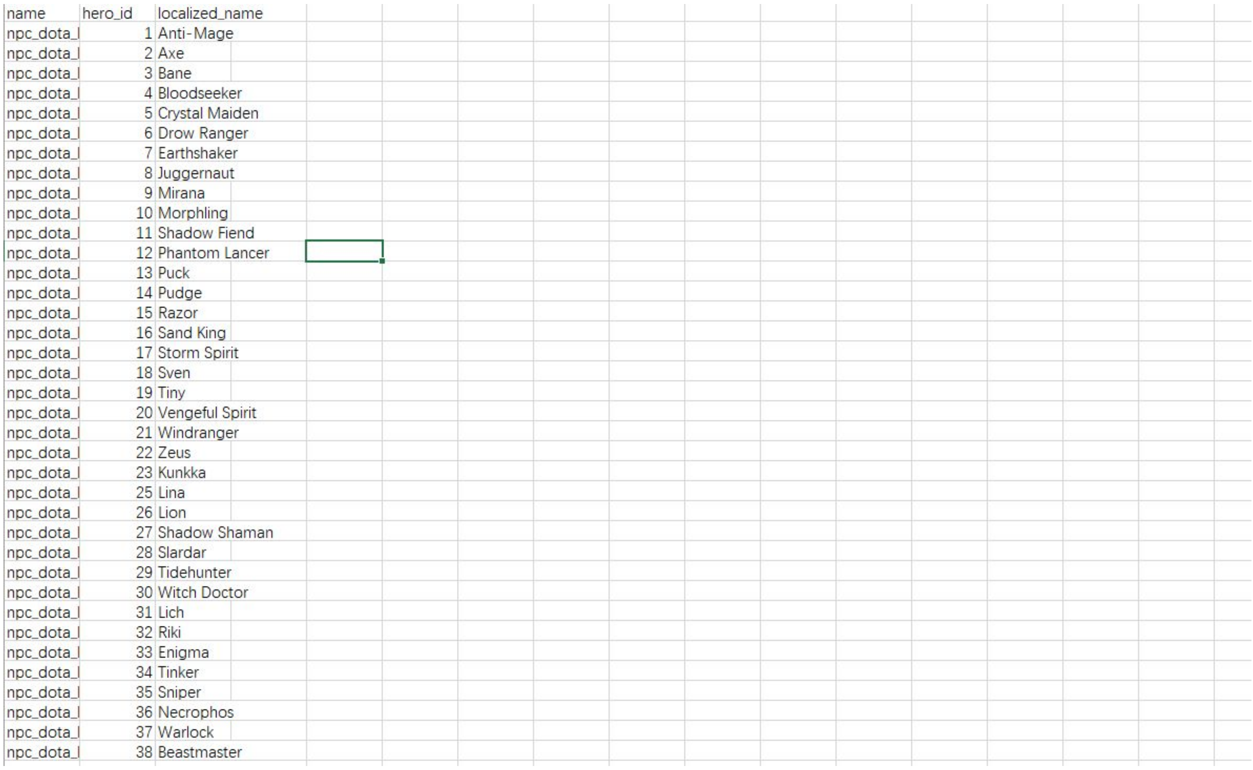
Our dataset is found on this website
The data set we selected for this project can be found here

Players.csv has the data for each game associated with its match id. Each match has 10 players and their picked heroes identified by hero_id. Every player has its statistics such as kill, death, etc.



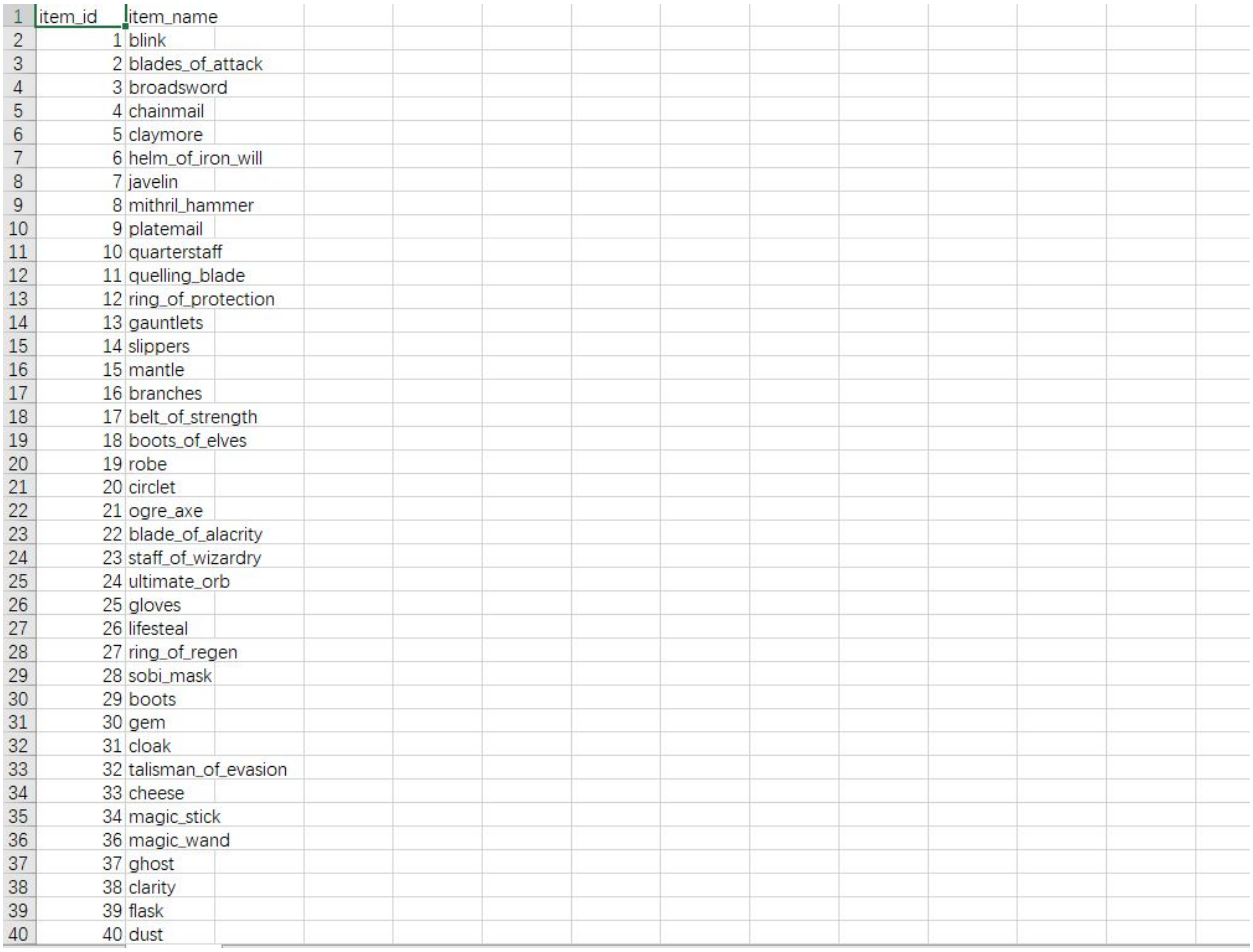
match_id	account_id	hero_id	player_slot	gold	gold_spent	gold_per_r	xp_per_mi	kills	deaths	assists	denies	last_hits	stuns	hero_dam	hero_heali	tower_dan	item_0	item_1	item_2	item_3	item_4	item_5	level	leaver_stat	xp_hero
0	0	86	0	3261	10960	347	362	9	3	18	1	30	76.7356	8690	218	143	180	37	73	56	108	0	16	0	8840
0	1	51	1	2954	17760	494	659	13	3	18	9	109	87.4164	23747	0	423	46	63	119	102	24	108	22	0	14331
0	0	83	2	110	12195	350	385	0	4	15	1	58	None	4217	1595	399	48	60	59	108	65	0	17	0	6692
0	2	11	3	1179	22505	599	605	8	4	19	6	271	None	14832	2714	6055	63	147	154	164	79	160	21	0	8583
0	3	67	4	3307	23825	613	762	20	3	17	13	245	None	33740	243	1833	114	92	147	0	137	63	24	0	15814
0	4	106	128	476	12285	397	524	5	6	8	5	162	None	10725	0	112	145	73	149	48	212	0	19	0	8502
0	0	102	129	317	10355	303	369	4	13	5	2	107	None	15028	764	0	50	11	102	36	185	81	16	0	5201
0	5	46	130	2390	13395	452	517	4	8	6	31	208	None	10230	0	2438	41	63	36	147	168	21	19	0	6853
0	0	7	131	475	5035	189	223	1	14	8	0	27	67.0277	4774	0	0	36	0	0	46	0	180	12	0	4798
0	6	73	132	60	17550	496	456	1	11	6	0	147	60.9748	6398	292	0	63	9	116	65	229	79	18	0	6659
1	0	7	0	76	12160	218	206	3	4	9	0	36	37.9243	4075	0	0	0	0	0	0	0	0	12	0	3857
1	7	82	1	9	19625	581	756	9	10	8	9	343	None	13888	0	1679	48	0	176	96	1	108	25	0	2820
1	0	71	2	1240	10220	339	352	5	13	11	3	76	99.9916	7788	0	81	172	181	63	116	9	73	16	0	9305
1	8	39	3	2400	14395	460	544	12	15	9	8	169	30.2234	19920	0	123	116	127	63	46	100	0	21	0	10970
1	4	21	4	1051	12910	365	436	6	11	12	7	131	47.5044	12913	0	537	50	36	41	168	108	21	18	0	8191
1	9	73	128	1277	20275	600	509	2	12	24	2	220	86.2695	14130	0	1765	178	137	1	24	129	48	20	0	7883
1	0	22	129	847	21840	487	517	8	5	17	0	193	1.00076	19218	0	988	201	1	190	48	65	235	20	0	6284
1	0	5	130	389	19165	488	583	14	8	9	0	101	2.20195	11398	0	971	254	102	0	116	108	48	21	0	15399
1	5	67	131	4055	24165	631	755	16	5	21	11	226	None	28505	0	6149	158	50	147	196	114	46	25	0	17504
1	0	106	132	2517	22305	585	753	10	7	12	3	250	None	20065	0	1275	48	0	1	141	145	145	25	0	14566
2	10	51	0	259	7990	237	249	5	13	7	3	97	66.2857	15638	0	39	127	102	41	36	214	46	14	0	2555
2	11	109	1	781	12515	322	358	6	11	5	1	179	None	7989	16	1446	147	185	36	63	46	162	17	0	3872
2	12	9	2	640	13845	355	425	10	6	8	3	154	70.0816	14295	0	217	170	63	36	212	166	123	19	0	8037
2	13	41	3	667	13260	328	345	0	9	4	3	154	14.0544	3159	0	0	116	145	172	65	29	0	17	0	2456
2	0	27	4	147	7380	189	229	1	10	7	0	41	33.0944	4962	0	184	180	60	46	23	21	0	13	0	4235
2	0	38	128	785	20500	450	567	13	7	18	4	163	32.4821	14580	706	1438	1	0	110	63	81	108	22	0	11892
2	0	7	129	479	15760	376	461	5	6	28	2	78	76.0841	13796	0	729	190	102	1	180	108	0	20	0	12162
2	0	10	130	2298	20735	493	535	17	2	14	5	170	18.3364	15755	0	3141	139	164	63	147	123	0	21	0	11489
2	0	12	131	4448	13990	389	454	7	2	13	2	111	None	12177	467	2131	63	81	174	147	46	0	19	0	9464
2	0	85	132	3167	10635	313	363	6	5	17	3	51	None	4950	572	551	88	242	46	180	108	0	17	0	8403
3	14	50	0	1847	9690	290	378	4	13	21	8	47	3.01025	7691	10814	371	231	0	94	0	0	0	19	0	11534
3	0	44	1	1145	18550	498	619	24	13	22	4	170	0.367783	27738	0	450	63	154	116	164	0	135	24	0	21833
3	0	32	2	1244	17825	454	635	17	11	18	5	144	11.8412	23279	0	1987	196	143	116	71	154	63	25	0	21297

Hero_names.csv has hero_id corresponds with its name



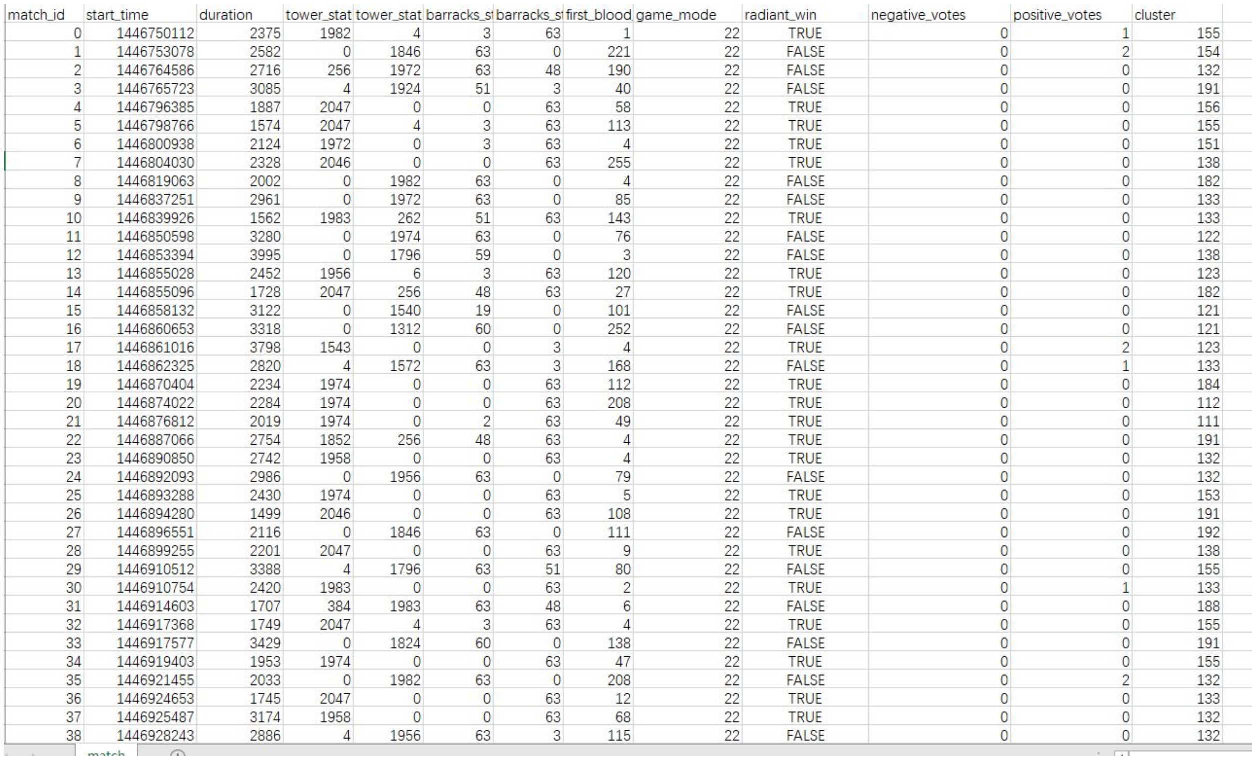
name	hero_id	localized_name
npc_dota_l	1	Anti-Mage
npc_dota_l	2	Axe
npc_dota_l	3	Bane
npc_dota_l	4	Bloodseeker
npc_dota_l	5	Crystal Maiden
npc_dota_l	6	Drow Ranger
npc_dota_l	7	Earthshaker
npc_dota_l	8	Juggernaut
npc_dota_l	9	Mirana
npc_dota_l	10	Morphling
npc_dota_l	11	Shadow Fiend
npc_dota_l	12	Phantom Lancer
npc_dota_l	13	Puck
npc_dota_l	14	Pudge
npc_dota_l	15	Razor
npc_dota_l	16	Sand King
npc_dota_l	17	Storm Spirit
npc_dota_l	18	Sven
npc_dota_l	19	Tiny
npc_dota_l	20	Vengeful Spirit
npc_dota_l	21	Windranger
npc_dota_l	22	Zeus
npc_dota_l	23	Kunkka
npc_dota_l	25	Lina
npc_dota_l	26	Lion
npc_dota_l	27	Shadow Shaman
npc_dota_l	28	Slardar
npc_dota_l	29	Tidehunter
npc_dota_l	30	Witch Doctor
npc_dota_l	31	Lich
npc_dota_l	32	Riki
npc_dota_l	33	Enigma
npc_dota_l	34	Tinker
npc_dota_l	35	Sniper
npc_dota_l	36	Necrophos
npc_dota_l	37	Warlock
npc_dota_l	38	Beastmaster

Item_ids.csv has item_id correspond with its name



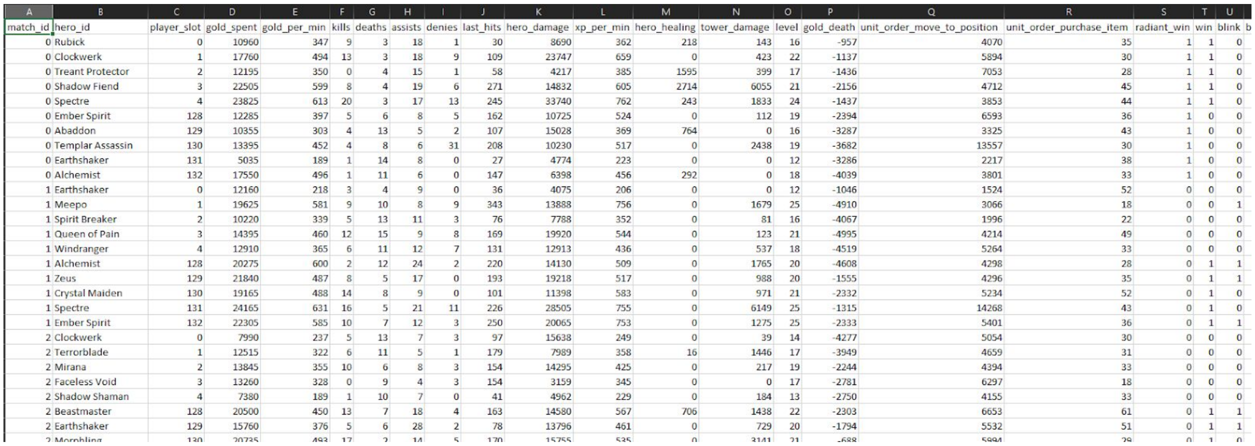
1	item_id	item_name
2	1	blink
3	2	blades_of_attack
4	3	broadsword
5	4	chainmail
6	5	claymore
7	6	helm_of_iron_will
8	7	javelin
9	8	mithril_hammer
10	9	platemail
11	10	quarterstaff
12	11	quelling_blade
13	12	ring_of_protection
14	13	gauntlets
15	14	slippers
16	15	mantle
17	16	branches
18	17	belt_of_strength
19	18	boots_of_elves
20	19	robe
21	20	circlet
22	21	ogre_axe
23	22	blade_of_alacrity
24	23	staff_of_wizardry
25	24	ultimate_orb
26	25	gloves
27	26	lifesteal
28	27	ring_of_regen
29	28	sobi_mask
30	29	boots
31	30	gem
32	31	cloak
33	32	talisman_of_evasion
34	33	cheese
35	34	magic_stick
36	36	magic_wand
37	37	ghost
38	38	clarity
39	39	flask
40	40	dust

Match.csv tells the result of each match by column radiant_win



match_id	start_time	duration	tower_stat	tower_stat	barracks_s	barracks_s	first_blood	game_mode	radiant_win	negative_votes	positive_votes	cluster
0	1446750112	2375	1982	4	3	63	1	22	TRUE	0	1	155
1	1446753078	2582	0	1846	63	0	221	22	FALSE	0	2	154
2	1446764586	2716	256	1972	63	48	190	22	FALSE	0	0	132
3	1446765723	3085	4	1924	51	3	40	22	FALSE	0	0	191
4	1446796385	1887	2047	0	0	63	58	22	TRUE	0	0	156
5	1446798766	1574	2047	4	3	63	113	22	TRUE	0	0	155
6	1446800938	2124	1972	0	3	63	4	22	TRUE	0	0	151
7	1446804030	2328	2046	0	0	63	255	22	TRUE	0	0	138
8	1446819063	2002	0	1982	63	0	4	22	FALSE	0	0	182
9	1446837251	2961	0	1972	63	0	85	22	FALSE	0	0	133
10	1446839926	1562	1983	262	51	63	143	22	TRUE	0	0	133
11	1446850598	3280	0	1974	63	0	76	22	FALSE	0	0	122
12	1446853394	3995	0	1796	59	0	3	22	FALSE	0	0	138
13	1446855028	2452	1956	6	3	63	120	22	TRUE	0	0	123
14	1446855096	1728	2047	256	48	63	27	22	TRUE	0	0	182
15	1446858132	3122	0	1540	19	0	101	22	FALSE	0	0	121
16	1446860653	3318	0	1312	60	0	252	22	FALSE	0	0	121
17	1446861016	3798	1543	0	0	3	4	22	TRUE	0	2	123
18	1446862325	2820	4	1572	63	3	168	22	FALSE	0	1	133
19	1446870404	2234	1974	0	0	63	112	22	TRUE	0	0	184
20	1446874022	2284	1974	0	0	63	208	22	TRUE	0	0	112
21	1446876812	2019	1974	0	2	63	49	22	TRUE	0	0	111
22	1446887066	2754	1852	256	48	63	4	22	TRUE	0	0	191
23	1446890850	2742	1958	0	0	63	4	22	TRUE	0	0	132
24	1446892093	2986	0	1956	63	0	79	22	FALSE	0	0	132
25	1446893288	2430	1974	0	0	63	5	22	TRUE	0	0	153
26	1446894280	1499	2046	0	0	63	108	22	TRUE	0	0	191
27	1446896551	2116	0	1846	63	0	111	22	FALSE	0	0	192
28	1446899255	2201	2047	0	0	63	9	22	TRUE	0	0	138
29	1446910512	3388	4	1796	63	51	80	22	FALSE	0	0	155
30	1446910754	2420	1983	0	0	63	2	22	TRUE	0	1	133
31	1446914603	1707	384	1983	63	48	6	22	FALSE	0	0	188
32	1446917368	1749	2047	4	3	63	4	22	TRUE	0	0	155
33	1446917577	3429	0	1824	60	0	138	22	FALSE	0	0	191
34	1446919403	1953	1974	0	0	63	47	22	TRUE	0	0	155
35	1446921455	2033	0	1982	63	0	208	22	FALSE	0	2	132
36	1446924653	1745	2047	0	0	63	12	22	TRUE	0	0	133
37	1446925487	3174	1958	0	0	63	68	22	TRUE	0	0	132
38	1446928243	2886	4	1956	63	3	115	22	FALSE	0	0	132

Final dataset.csv: this is the final datasets constructed by our python code and used by all research questions as input data frames.



match_id	hero_id	player_slot	gold_spent	gold_per_min	kills	deaths	assists	denies	last_hits	hero_damage	xp_per_min	hero_healing	tower_damage	level	gold_death	unit_order_move_to_position	unit_order_purchase_item	radiant_win	win	blink	b
0	Rubick	0	10960	347	9	3	18	1	30	8690	362	218	143	16	-957	4070	35	1	1	0	
0	Clockwerk	1	17760	494	13	3	18	9	109	23747	659	0	423	22	-1137	5894	30	1	1	0	
0	Treant Protector	2	12195	350	0	4	15	1	58	4217	385	1595	399	17	-1436	7053	28	1	1	0	
0	Shadow Fiend	3	22505	599	8	4	19	6	271	14832	605	2714	6055	21	-2156	4712	45	1	1	0	
0	Spectre	4	23825	613	20	3	17	13	245	33740	762	243	1833	24	-1437	3853	44	1	1	0	
0	Ember Spirit	128	12285	397	5	6	8	5	162	10725	524	0	112	19	-2394	6593	36	1	0	0	
0	Abaddon	129	10355	303	4	13	5	2	107	15028	369	764	0	16	-3287	3325	43	1	0	0	
0	Templar Assassin	130	13395	452	4	8	6	31	208	10230	517	0	2438	19	-3682	13557	30	1	0	0	
0	Earthshaker	131	5035	189	1	14	8	0	27	4774	223	0	0	12	-3286	2217	38	1	0	0	
0	Alchemist	132	17550	496	1	11	6	0	147	6398	456	292	0	18	-4039	3801	33	1	0	0	
1	Earthshaker	0	12160	218	3	4	9	0	36	4075	206	0	0	12	-1046	1524	52	0	0	0	
1	Meepo	1	19625	581	9	10	8	9	343	13888	756	0	1679	25	-4910	3066	18	0	0	1	
1	Spirit Breaker	2	10220	339	5	13	11	3	76	7788	352	0	81	16	-4067	1996	22	0	0	0	
1	Queen of Pain	3	14395	460	12	15	9	8	169	19920	544	0	123	21	-4995	4214	49	0	0	0	
1	Windranger	4	12910	365	6	11	12	7	131	12913	436	0	537	18	-4519	5264	33	0	0	0	
1	Alchemist	128	20275	600	2	12	24	2	220	14130	509	0	1765	20	-4608	4298	28	0	1	1	
1	Zeus	129	21840	487	8	5	17	0	193	19218	517	0	988	20	-1555	4296	35	0	1	1	
1	Crystal Maiden	130	19165	488	14	8	9	0	101	11398	583	0	971	21	-2332	5234	52	0	1	0	
1	Spectre	131	24165	631	16	5	21	11	226	28505	755	0	6149	25	-1315	14268	43	0	1	0	
1	Ember Spirit	132	22305	585	10	7	12	3	250	20065	753	0	1275	25	-2333	5401	36	0	1	1	
2	Clockwerk	0	7990	237	5	13	7	3	97	15638	249	0	39	14	-4277	5054	30	0	0	0	
2	Terrorblade	1	12515	322	6	11	5	1	179	7989	358	16	1446	17	-3949	4659	31	0	0	0	
2	Mirana	2	13845	355	10	6	8	3	154	14295	425	0	217	19	-2244	4394	33	0	0	0	
2	Faceless Void	3	13260	328	0	9	4	3	154	3159	345	0	0	17	-2781	6297	18	0	0	0	
2	Shadow Shaman	4	7380	189	1	10	7	0	41	4962	229	0	184	13	-2750	4155	33	0	0	0	
2	Beastmaster	128	20500	450	13	7	18	4	163	14580	567	706	1438	22	-2303	6653	61	0	1	1	
2	Earthshaker	129	15760	376	5	6	28	2	78	13796	461	0	729	20	-1794	5532	51	0	1	1	
2	Morphling	130	20735	493	17	2	14	5	170	15755	535	0	3141	21	-688	5994	29	0	1	0	

III. Challenge Goal:

1. Multiple Datasets:
    - Since some match statistics relevant to our data processing are in separate CSV files, we have to merge the data frames and replace the values corresponding to the ids. Sometimes, values need to be converted into other types in order to make boolean statements work.
2. Machine Learning:

    - Question 2 implements new Machine Learning called RandomForest
    - RandomForest is used to make predictions of the number of kills of a player and find out the factor importance of the features.
3. Visualization:
    - Graphical representation of the data requires additional steps to convert the input data into desired types that are required by matplotlib and seaborn
    - New plotting method named horizontal bar plot is implemented

IV. Methodology:

a. Terminologies:

- **Gold Spent:** how much gold the player's hero spent in one match to buy items. Gold is gained by farming creeps, killing enemy heroes, and destroying enemy towers
- **Kills**: number of enemy heroes killed by the player in one match.
- **Deaths:** number of times the player's hero is killed by enemy heroes.
- **Assists:** number of enemy heroes killed by the teammates which the player's hero also contribute to that kill
- **Hero Damage:** amount of damage the player's hero deals to the enemy heroes

b. Procedures:

The First step for each research question is to find out the required information. Since different categories of data of all matches are stored in different datasets in CSV format, we should look for the datasets and join them together so that data processing can be performed correspondingly.

Second Step, the selected algorithm would be performed based on the research question:

1. What are some traits that winning teams share?

In order to accomplish this goal, we have to find all the winning teams and losing teams in our data set. Next, we can separately do the statistics work such as average gold spent, kills, deaths, etc for both winning teams and losing teams.

2. How to predict the number of kills by each player?

The algorithm we are going to use to predict the number of kills is called random forest. This machine learning program is designed to create many decision trees with data selected randomly. Therefore, it could allow a more accurate prediction than implementing a single decision tree.

3. How does each hero perform on average?
The performance is observed by many features available from the datasets. The algorithm for calculating the performance is taking the available heroes' information and average them accordingly.

Third Step, the analysis of the result is performed according to the type of information the applied algorithm generates.

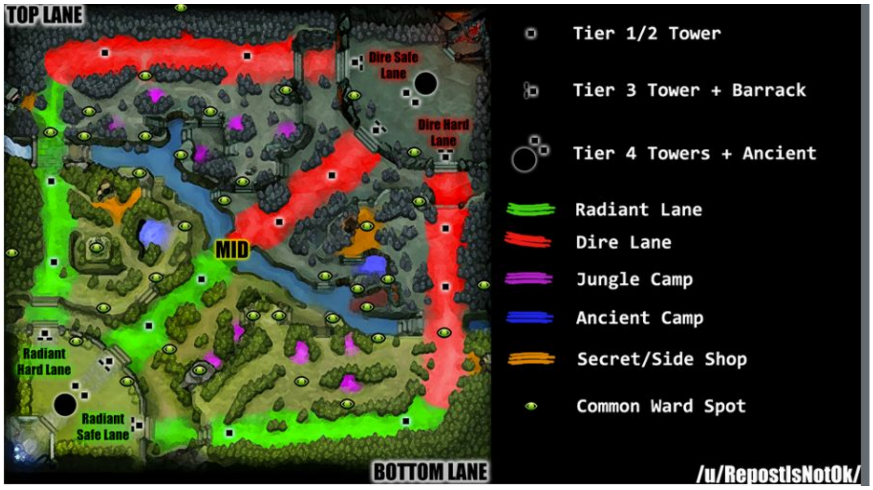
Fourth Step, each research question would have a corresponding graph as data visualization for better presentation on the results. Each question might need a slightly different way of visualization.

V. Motivation and Background:

The motivation behind this project is for us, as players, to acquire the insight of MOBA (Multiplayer Online Battle Arena) games. Dota 2 is one of the MOBA games that acquires lots of game experiences. It is quite similar to how AlphaGo manages to beat humans on Go-tons of game experience. Normal players are usually playing games for fun, and professionals are seeking the strategies to win a game of any kind. It, then, leads to the question of whether or not the in-game statistics could determine one's success in that particular game.

This project focuses on the statistical analysis of the Dota 2 datasets with various aspects. Importantly, a brief introduction to the game's mechanism is important.

In a normal match, each team of 5 players is responsible for defending their Ancient building located on the opposite side of the map, if Ancient falls, the game ends. Towers are one of the defense systems from the attack of their enemies which compose of creeps-the None Player Character which push the attack towards the enemy's towers in three lanes-and player-controlling heroes. The illustration of the map is shown below:

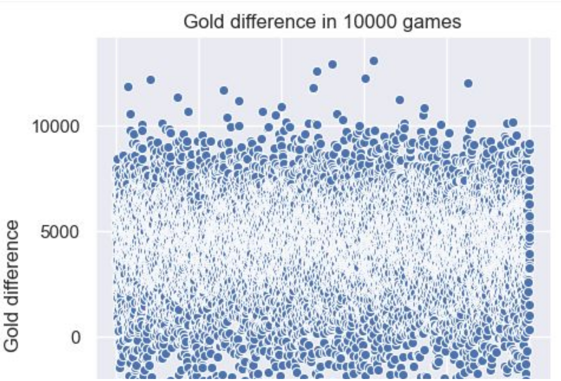


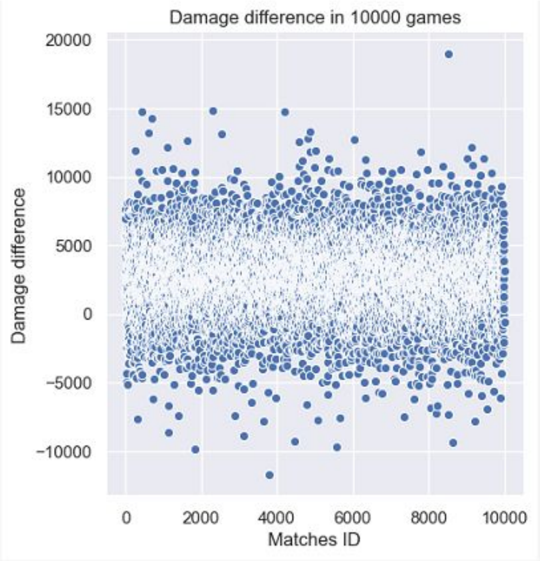
The heroes summoned by each player are unique in a typical game, and each hero has unique abilities or spells they could use to attack the enemy. Their ultimate goal is to destroy the enemy's base or Ancient shown in this graph. To win a match, it is necessary to not only master a hero but also cooperate with the teammates with strategies in fighting their enemy heroes. The heroes could learn their spells by leveling up to gain strength and also purchase the items as in-game equipment with gold. Gold is the in-game currency that could be obtained in many ways: killing the creeps, killing or assisting in an enemy kill, destroying the enemy towers, etc.

VI. Result:

1. What are some traits that winning teams share?

- By calculating the difference statistics of 20,000 winning games and 20,000 losing games, we found out, on average, the winning team spent 4219 gold more, made 3 more kills, made 3 more assists, dealt 2737 more hero damage than the losing team. Those pieces of data are very consistent to our expectation and intuition since in order to win the game, you have to control most of the map resources(Creep gold) and kill more enemy heroes to get gold/item advantage and finally result in a triumph. Therefore, we recommend players play actively in the match, controlling maps, gaining more gold, and creating opportunities for killing, and win the game in the end.

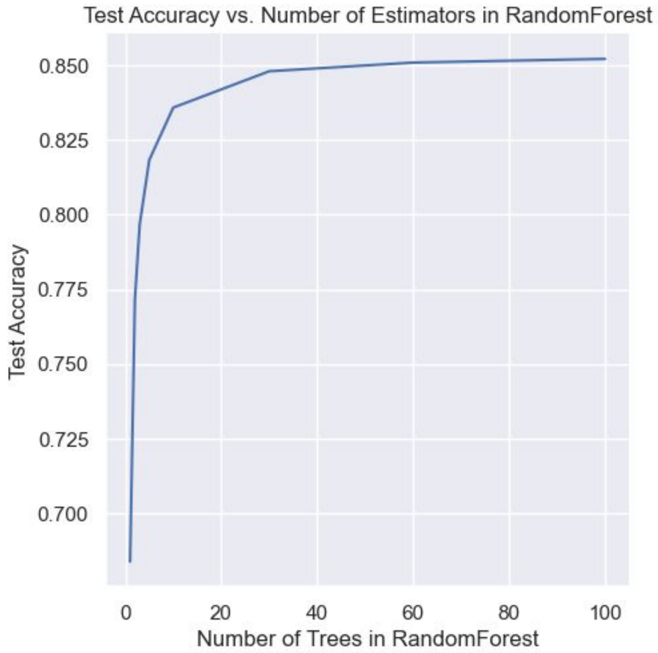




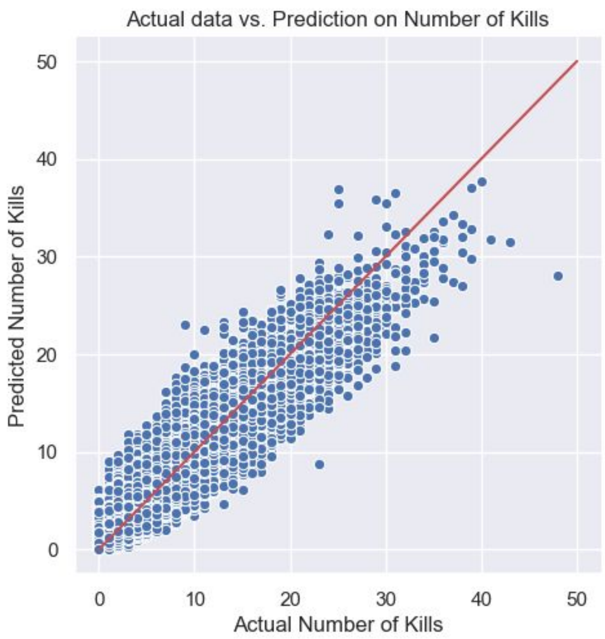
- Since 20000 games will be too large for the graph and the difference in kills/assists numbers is not significant, we picked 10000 match games and depicted Damage difference and Gold difference for winning and losing games. As illustrated in the graphs, the white spot meaning there are so many scatter points on top of each other. In the Damage graph, most points located in the range of damage difference are around 2500 +-2500 which is pretty consistent with the average difference of 2737 as indicated above. For the Gold graph, most points located in the range of gold differences are around 5000+-2500 which is unanimous to the value of 4219 indicated above. One thing we want to point out is the outlier in both graphs, especially in the negative range. An extreme negative value, for example, -5000 or below in both graphs, demonstrates the winning team made a "comeback" game meaning that in most of the match time, the team is in a disadvantageous position(less kills, gold, items...), but won the match in the end. Although they are only a few examples of "comeback" games, we should never give up until the fall of the Ancient in Dota2!

2. **How to predict the number of kills by each player?**
    - Through the machine learning model, RandomForest, we could predict the kills by each player using the relevant variables. From 100 trees built inside the RandomForest, the prediction accuracy on the training model is around 97%, and which drops to around 85% on the testing model. It is significantly improved as more trees are built inside the RandomForest which renders an increase of the test accuracy from 68% to 85%.

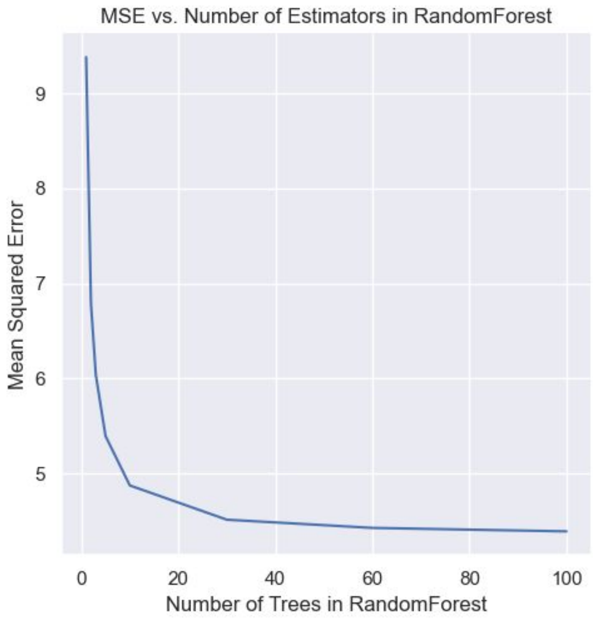
- The graph below shows Test Accuracy vs. Number of Estimators graph:



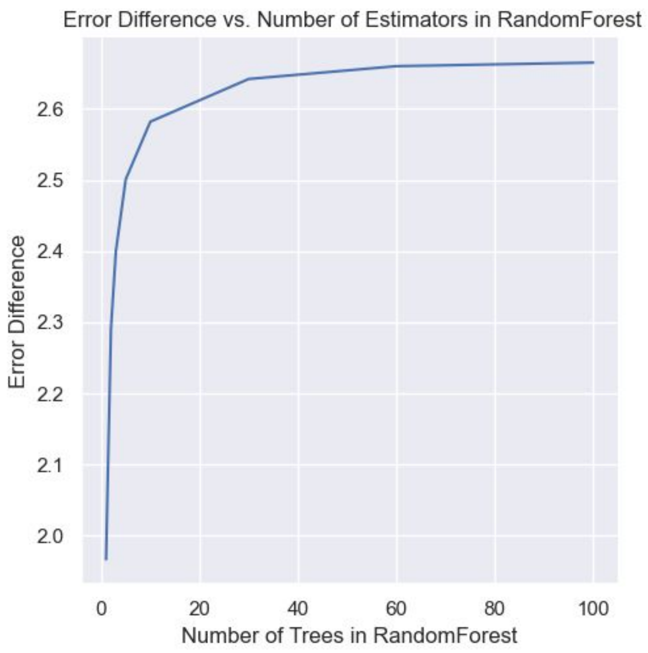
-
    - This graph shows that the increasing number of trees in the RandomForest dramatically increases the test accuracy when the number of trees approaches 10. Then, the test accuracy increases decreasingly afterward, and it reaches the highest point when 100 trees are built. Therefore, the increase in the number of trees would slowly increase the test accuracy as the number of trees increases
- The graph of actual values vs. predictions is shown below:



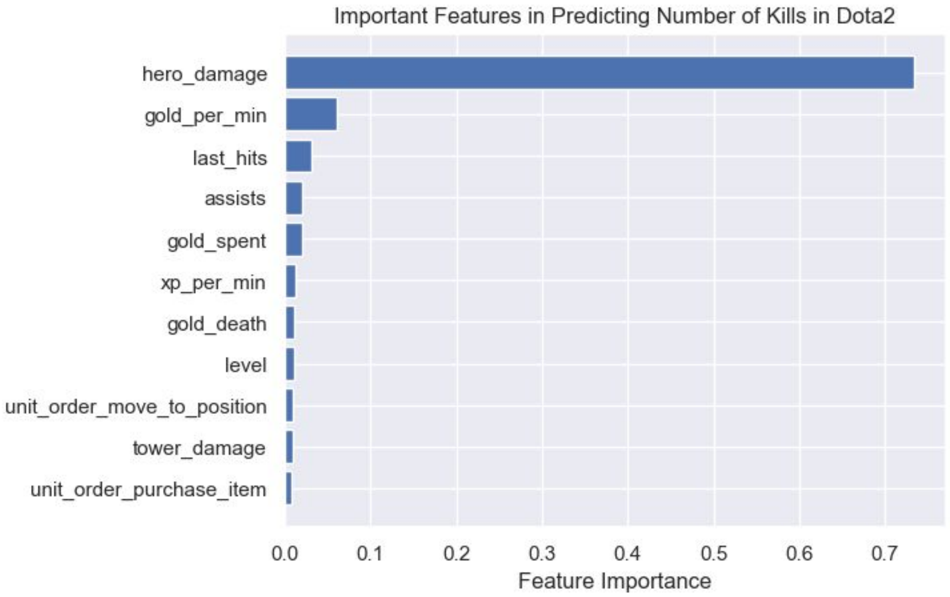
-
  - This graph uses 100 trees in the RandomForest as a parameter. The red line dictates a 100% accuracy from the prediction. When the dots are deviating from the red line, it indicates the prediction errors.



-
  - This graph indicates a decreasing trend in residual errors as the number of trees increases in the RandomForest model, and which is opposite to the result from test accuracy.



-
    - The difference between baseline errors and mean absolute errors are increasing which indicates the decrease in prediction error on kills per player. This shows that the RandomForest model is improving as the number of trees increases as a parameter.
- The features that are contributing the most to the prediction of kills is shown below:



-
    - This graph shows the top 10 contributing factors in predicting the # of kills for each player. As one of the factors, hero damages

contribute more than 70% of the weights from all factors in predicting the # of kills per player from all features.

- In conclusion, the model of RandomForest is improving as the number of trees or estimators increases and therefore improving the prediction on the number of kills per player.

4. **How does each hero perform on average?**
According to our data analysis. We found out that in 20000 games, the most popular hero is **Windranger**, with 8314 games played. The least popular hero is **Chen**, with only 215 games played. The richest hero is **Alchemist**, with average gold of 24461 each game. The poorest hero is **IO**, with an average gold of 8689 each game. The most violent hero is **Zeus**, dealing an average of 23689 damage each game. The slayer hero is **Riki**, killing an average of 13 heroes each game. The burden hero is **Techies**, dying an average of 10 times each game. Therefore, based on our result. If you like the feel of killing enemies, you probably should pick the hero, Riki. If you like farming gold, you would like to play Alchemist. Also, we advise avoiding playing Techies since it has the highest average deaths, and death will add gold and XP to the enemy team. It makes sense that Windranger is the most popular hero since her ability contains stun, AOE, and escape.

VII. Work Plan:

We will be using VScode for code sharing and Google doc for info sharing.

a. Machine Learning:

Planned: Both authors will learn more machine learning knowledge next week for about 5 to 10 hours. Since one of the research questions involves prediction. We have to dig deep into machine learning knowledge.

Reality:

- Implementation of RandomForest in Python requires more time and effort from me to understand more on the data types rather than the concepts of this machine learning technique. For instance, conversion from data frame into Numpy array to implement machine learning; conversion of list a feature importance data to series to successfully implement horizontal bar plot; build a list of tuples using zip. The syntax problem is frustrating in the sense that I have to not only view the documentations but also examples that could help me understand how they worked. (**More than days**)
- Dummy variables are created incorrectly using the Pandas built-in function. It is not right in the sense that items in different slots do not contribute to the

effect on prediction. As 6 independent item slots exist, I need to eliminate the duplicate columns. Therefore, manually creating dummy variables is necessary. Due to my coding capability, long-running time is not avoidable after several attempts to reduce it. (**Around a day**)

b. Data Management :
Planned: Author 1 will take care of most of the data merging process since many statistical values are stored in separate CSV files. Also, he will exclude the column that we are not using to analyze. **(~3 hours)**
Reality: Data management is indeed an onerous task in data analysis. It takes time to find out which column to merge as well as adding new columns to the existing CSV files. However, eliminating the columns we don't want is not that difficult(4 hours).

c. Data Visualization :
Planned: Author 2 will take care of most of the data visualization for each of the research questions including types of plot. **(~3 hours)**
Reality: Data Visualization is not hard, but we have to determine which kind of graph works best for our research questions. Also, the size, font, etc of the graph took us much time to adjust. (**2 hours**)

d. Final Output:
Planed: Both authors will work together to use the data set to answer each research question and produce the final output for part 2.**(~6 hours)**
Reality: To make the summary of the final output is indeed onerous as we expected. We have to group the result with its explanatory graph and make a deep analysis of every single research question. **(5 hours)**

VIII. Testing:

We implement the "Assert_equal()" function we normally use for the Homework and a small data file called "test_file.csv" to test out if my calculations in question1 and question3 are correct. The first parameter of the "Assert_equal()" function is calculated by hand. For the machine learning part on question2, the prediction of the number of kills per player cannot be tested and rather accuracy of the test data can be provided through visualization.

IX. Collaboration:

For the machine learning, we viewed tons of websites and videos which included in the following links:

Looking for MSE: https://www.geeksforgeeks.org/python-mean-squared-error/
Plot regression line: https://seaborn.pydata.org/generated/seaborn.lmplot.html
Graph abline:
https://scriptverse.academy/tutorials/python-matplotlib-plot-straight-line.html
Plot horizontal bar graph:https://matplotlib.org/api/_as_gen/matplotlib.pyplot.barh.html
Horizontal bar graph example:https://matplotlib.org/gallery/lines_bars_and_markers/barh.html
Property of series of pandas:https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.Series.html
Importance in RandomForest:https://www.datacamp.com/community/tutorials/random-forests-classifier-python
RandomForest Documentation:https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
Fundamental coding of RandomForest in Python:https://towardsdatascience.com/improving-random-forest-in-python-part-1-893916666cd
Understanding RandomForest:https://towardsdatascience.com/an-implementation-and-explanation-of-the-random-forest-in-python-77bf308a9b76
Abandoned Model from this project in implementing RandomForest using k-fold Cross-Validation(take too long to run):https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74
https://scikit-learn.org/stable/modules/cross_validation.html
Reference of RandomForest Through other courses:http://www.jacoblariviere.com/DS_pricing.html
Setting up Visual Studio Code for this project:https://www.youtube.com/watch?v=-nh9rCzPJ20
Seaborn.relplot documentation:https://seaborn.pydata.org/generated/seaborn.relplot.html
OS library documentation:https://www.geeksforgeeks.org/os-module-python-examples/

OS library functions example:https://careerkarma.com/blog/python-check-if-file-exists/#:~:text=Checking%20If%20a%20Certain%20File,that%20file%20can%20be%20found.

DataFrame documentation:https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html