Chicken for Dinner

A data mining project on discovering how to win in PUBG

Zijun Xu
University of Colorado Boulder
zixu6769@colorado.edu

Jonathan Young
University of Colorado Boulder
joyo2566@colorado.edu

Jianyi Chen
University of Colorado Boulder
jich2029@colorado.edu

Abstract

Playerunknown's Battleground (PUBG for short) is a popular multiplayer online battle royale game developed and published by Bluehole, a Korean publisher. There is a maximum of 100 players per game. The game starts by putting all of the players on an airplane, which goes through the map in a random direction. The players are free to choose when to jump out of the plane after the back door opens. Various resources are provided across the map, including weapons (with a large number of possible combinations of parts), armor, medicine (healing), vehicles, etc.

Additionally, after a short while when all players land on the ground, the game forces all the players to move to the “safe zone”. If a player remains outside the safe zone, they will continually lose vitality. The size of this safe zone decreases with time, picking a new circle of the same relative size inside the new circle. This is to force the player’s closer together to instigate struggle (as shown in figure 1). The game ends when there is only one team left. A team may consists of 1-4 players at the start of the game, but only one must survive till the end for the team to win.

Statistically speaking, at the start of a game a single player has only a 1 percent chance of winning based solely on number of players. In other words, it is hard to win in PUBG! As there are more and more new players joining the game everyday, our aim was to create a survival guide for PUBG newbies and help them to survive longer. Specifically, by applying several data mining techniques to our datasets, we found answers for the following interesting questions:

- When it comes to winning, does luck or skill have higher precedence?
- What is the best place to jump based on player experience and game style?
- Where are “hot spots”, or places to avoid during the early stages of the game?
- Under what situation should they use a certain type of weapons?

From what we found, skill is more important than luck when it comes to winning. We created graphics and charts to show the best places to jump, as well as seemingly high traffic areas one should avoid if they are not a more experienced player. We also developed a weapon guide giving the player-used distances per weapon. This will allow new players to look at the play styles and strategies of others while coming up with their own methods of winning.

Introduction

To help the future players of PUBG to have a better chance of winning we attempted to mine a sort of survival guide from some raw game data.

In PUBG, all resources are randomly generated across the entire map. The resources in PUBG are divided into three categories: weapons, equipment (backpacks, helmet, armor), and medicine (used to replenish hp). Each type contains items of differing strengths. The quality of a player's belongings is one of the most important factors when predicting if he/she has any chance of winning. We chose to focus on the weapons a player may find because of the sheer volume of weapons a player will find during the game. And this decision of which to take could mean victory or defeat.

Another large decision, and often the most pivotal, the player must make is where they should jump from the plane to start the game. The maps are massive, which gives many choices for where you may go. We analyzed our data to create a density graph of the busiest locations during the time of the game where players are exiting the plane. In doing so we were able to compile a graph, and scores for each sector displayed on the graph. The higher the score, the higher the chance that going to that region will result in running into other players.

After they have landed it becomes a race often to see who can get the best equipment. But as the game goes on, the size of the playzone gets smaller and smaller in order to instigate confrontation. During these shifts, a lot of players gravitate to particular locations. We looked at which places this was and again created a graph with regions and scores. The higher the score, the more popular that location is.

We hope, that with this information, we will be able to assist the masses with their upcoming victories.

Related Work

We have found some interesting prior work that is related to our project. Christoph Egerland has done an exploration on where most of the people jump out of the plane in PUBG.[1] By using Jupyter notebook, Christoph scaled both maps' coordinates to fit the map image in order to draw density points later on the image file (shown in figure 1). Since the dataset doesn't contain the positions that players jumped out but only the positions of players' deaths, Christoph restricted his program to only take death positions in the first two minutes of the game - because most of the players that die early do not usually move far away from their jump locations.

Another interesting project was done by Justin Moore and published on August 8, 2017.[2] In his project *To drive...or not to drive*, he analysed vehicle usage in depth. First of all, he showed that the mean of average ride distance grows as there are more people in a team. Justin then drew a scatter plot showing the correlation between average ride distance and wins, as shown in figure 2. The result indicates that players who have the average ride distance between 2000 and 4000 are the players with the most wins. In other words, players die more often if they don't drive at all, or drive too much.

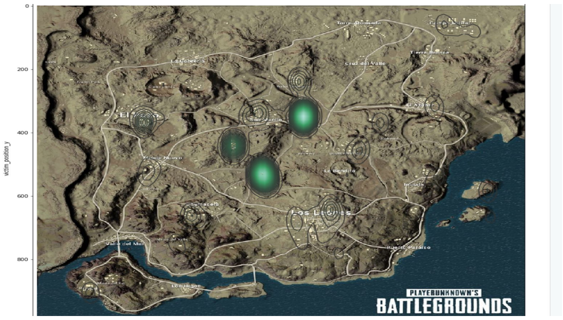


Figure 1: Density graph of where the players jump at the most

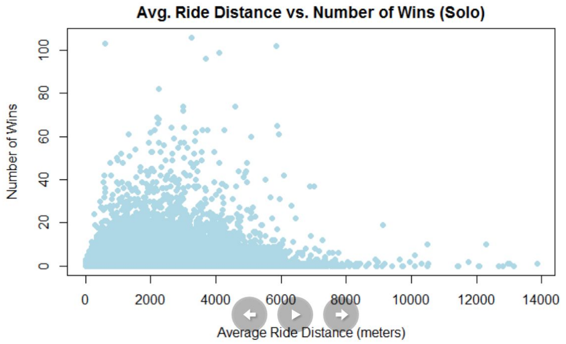


Figure 2: Average ride distance vs # of wins

Data Set

We got our dataset from Kaggle. (https://www.kaggle.com/skihikingkevin/pubg-match-deaths/data)

Our data contains two main datasets. The first dataset is called kill_match_stats_final and it contains following attributes:

- Weapon type
- Killer name
- Killer placement
- Killer position x
- Killer position y
- Map
- Match id
- Time
- Victim name
- Victim placement
- Victim position x
- Victim position y

Our second dataset is called agg_match_stats and it contains following attributes:

- Date
- Game size
- Match id
- Match mode
- Party size
- Player assists
- Player distance ride
- Player distance walk
- Player damage
- Player name
- Player survive time
- Team id
- Team placement

Tools

Most of our coding work was done in jupyter notebook. Additionally, we used several python libraries. We used sqlA for creating a database and we also used skLearn and Pandas for our clustering analysis.

Techniques Applied

Data Cleaning :

In cleaning our data, we removed all empty or incomplete data filled rows, as well as dropping columns containing naming information that was not useful to us. We then copied only our desired attributes over to a SQLAlchemy database because our data was too large to store as a Pandas DataFrame.

Since we are mining different results from our two different datasets, our data cleaning work is slightly different for the two sets as well. We used our first dataset, kill_match_stats_fina,l for weapon analysis only. Therefore, we were able deleted many unused attributes. We kept the weapon type and killer and victim position attributes. We also dropped any rows where the player was not killed by weapons, for example, the player could have been killed by the bluezone or by falling. These data are inconsequential to us.

The first data set is also the one we used to generate our graphs that tell the player which locations are most visited. For this analysis we used the victim_position and the time in game to allow us to control which part of the game we were looking at death counts for.

For our second dataset, we only selected the solo players' (party_size = 1) data. To have cleaner data, we calculated the IQR of total traveled distance and decided to drop players that have total traveled distance larger than 8400. Doing this can help us clean out outliers that caused by player using moving speed cheat. We also removed players with zero walking distance, because those people are most likely to be the people who get disconnected.

Since the second data set is used for our skill and luck, we only selected the attributes with information about kills, travel distances, and team placement.

Clustering :

We applied clustering technique in three different aspects. First of all, we applied clustering to weapon effective range. Based on our kill_match_stats_final dataset, we were able to calculate the weapon shooting distance based on killers'/victims' coordinates. Specifically, we applied euclidean distance formula to each pair of killer/victim position to get a distance. For each distance we got, we add a label indicating the weapon name, which enable us to cluster distances for each individual weapon. We had trouble finding python clustering library for one dimension arrays so we made our own version of 1 dimensional clustering code and it worked well. The result turns out to be successful and we are able to show the most used range for each weapon based on their cluster groups and number of points in each cluster group.

Second, we applied clustering to our analysis of skill vs luck. For our luck parameter, we calculated the total travel distance for each player. People who traveled less are count as lucky people. We also generated a skill score for each player. Skill score is calculated by the sum of the player's total kills, player's total knock downs and 0.5 times of player's total assists through the game. To have better results, we normalized both skill and total distance for every players by each match. Then we used the python library, sklearn, to help us do the k-means clustering with 4 clusters on our skill vs luck. See Figure 5 below.

Last but not least, in order to tell people where is a good place to start, we clustered the death locations in early game (first 100 seconds) to simulate players making "jumps". Since there would not be much chance for a player to move very far in the first 100 seconds, we can assume that these mined death locations are in fact the player's original jump location, or very close to. The reason we choose first 100 seconds is because all players will get off the airplane before 120 seconds. To avoid people that got disconnected from the game, we choose to use first 100 seconds instead of first 120 seconds. After a rough estimation, we find out the Erangel map has around 40 different areas that people can choose to jump. Therefore, we used k-means clustering with 40 clusters on player's death location in first 100 seconds. We used k-means clustering because we removed all of the excessive outliers and it is fast and allows for good results.
See Figure 10.

Key Result

Weapon range:

For most of the non-sniper weapons, their kill distances fall into the first two of the four clusters, indicating this weapon is most used when the distance between two players falls into these two clusters. For example, figure 3 is a sample output for the weapon named M416. As the output indicated, the most used range for the M416 is from 0 meters to 151 meters, since almost 80% of the distances fall into this range.

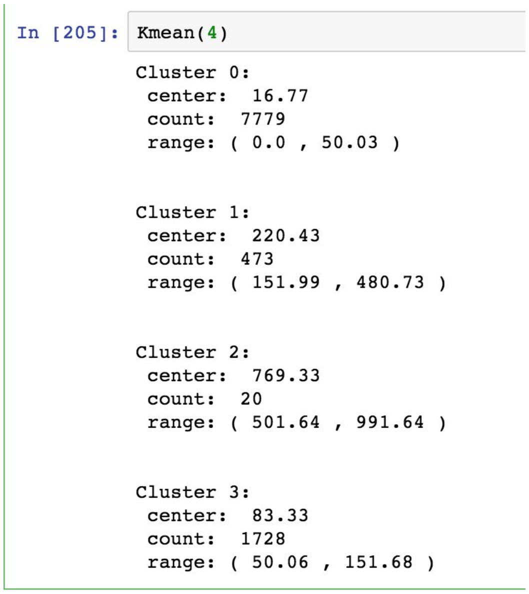
However, things are quite different for sniper type weapons. The number of points in each cluster are close unless the distance is extreme large. Figure 4 is an example showing the cluster groups for the sniper named SKS. As the table shows, the number of data points are roughly evenly distributed in the range from 0 to 381 meters. Only around one percent of the data falls into the extreme long range (382, 971), which probably indicates that those are either lucky shots or hackers or they are just very experienced players.

To cluster these distances we utilized the Euclidean distance formula and K-means clustering to group our kill distances with each type of weapon.

Based on our clustering results for each type of weapon, we are able to give a range for each weapon in meters that is the most utilized by other players who use that weapon.

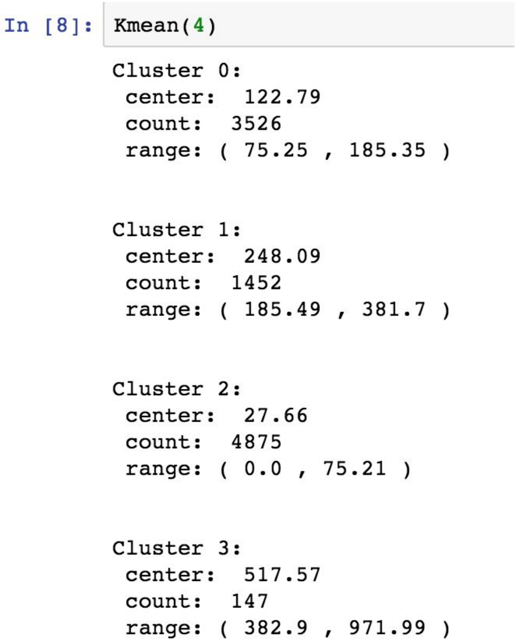
- Handguns
  - P18c 13.96
  - P1911 23.45
  - P92 25.25
- Shotguns
  - S686 13.86
  - S12K 15.05
  - S1987 9.38
- Assault Rifles
  - SCAR-L 117.83

  - M416 151.68
  - AKM 76.11
  - M16A4 209.29
  - Groza 141.07
- SMGs
  - Tommy Gun 41.88
  - UMP9 50.1
  - Micro UZI 18.22
  - Vector 61.32
- Sniper Rifles
  - Win94 96.92
  - SKS 381.7
  - Kar98k 355.52
  - Mini-14 80.75
  - Mk14 256.21
- Other
  - Grenade 44
  - Crossbow 119.3



```
In [205]: Kmean(4)

          Cluster 0:
           center:  16.77
           count:  7779
           range: ( 0.0 , 50.03 )


          Cluster 1:
           center:  220.43
           count:  473
           range: ( 151.99 , 480.73 )


          Cluster 2:
           center:  769.33
           count:  20
           range: ( 501.64 , 991.64 )


          Cluster 3:
           center:  83.33
           count:  1728
           range: ( 50.06 , 151.68 )
```

Figure 3: sample output for M416



```
In [8]: Kmean(4)

        Cluster 0:
         center:  122.79
         count:  3526
         range: ( 75.25 , 185.35 )


        Cluster 1:
         center:  248.09
         count:  1452
         range: ( 185.49 , 381.7 )


        Cluster 2:
         center:  27.66
         count:  4875
         range: ( 0.0 , 75.21 )


        Cluster 3:
         center:  517.57
         count:  147
         range: ( 382.9 , 971.99 )
```

Figure 4: sample output for SKS

Skill vs. luck:

For the skill vs. luck, we used k-means clustering with four clusters on out skill vs luck. As we mentioned before, skill is the sum of the total kills,total knock downs, and 0.5 time total assists. Luck, on the other hand, is calculated by total travel distance. People who didn’t travel a lot though the game are count as lucky people. After clustering. We have four different clusters.

As Figure 5 showing below.

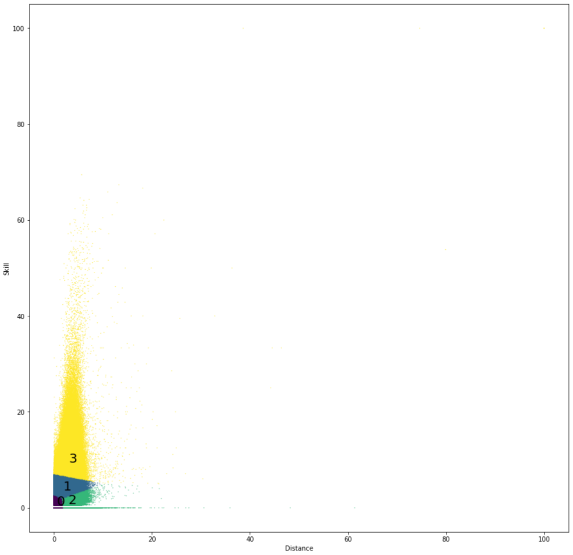


Figure 5: K-means clustering on skill vs luck

Cluster 0 contains players who didn't travel a lot and didn't have a lot kills. Those people are the lucky people. After count the frequency of rankings in

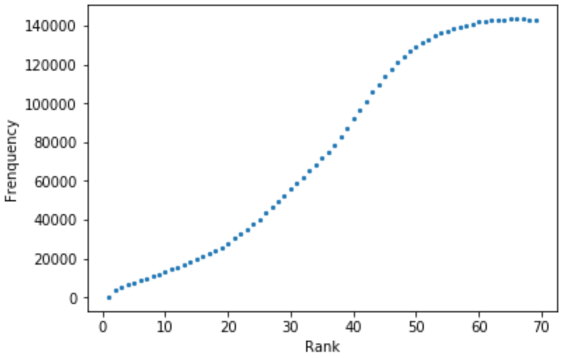


Figure 6: Rank vs frequency on cluster 0

Cluster 1 contains the players who did not get a lot of kills, but had the highest travel distances.

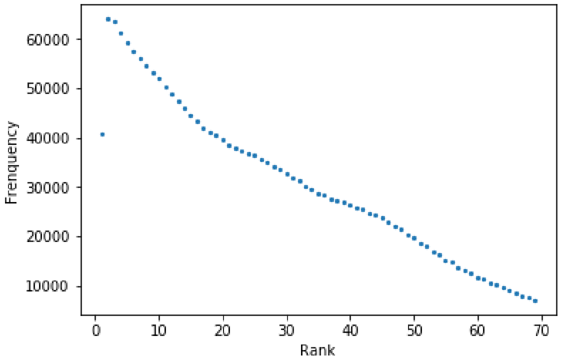


Figure 7: Rank vs frequency on cluster 1

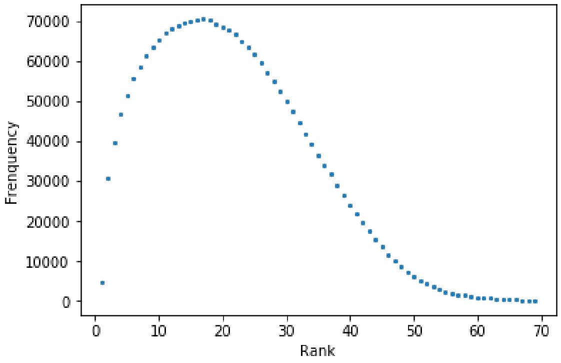


Figure 8: Rank vs frequency on cluster 2

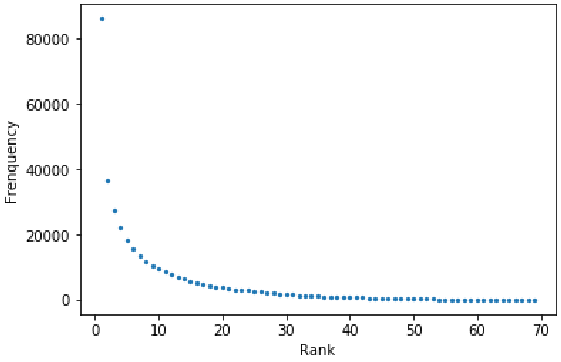
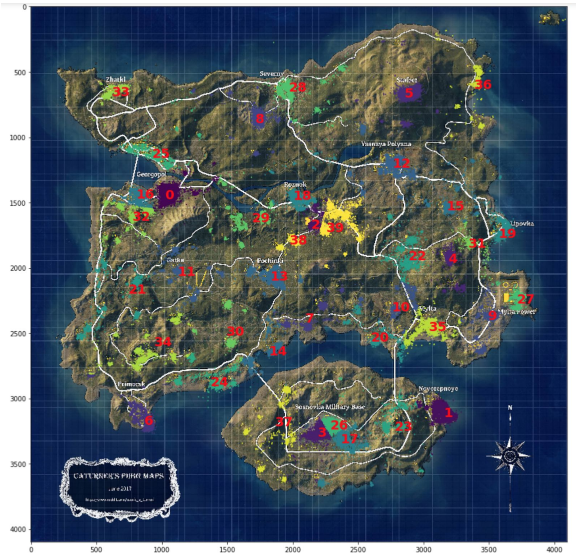


Figure 9: Rank vs frequency on cluster 3

Jump location:

In order to help players know where is a good spot to start the game, we mined the first 100 seconds of deaths. This is because the player is given the first 120 seconds to decide where to exit the plane and start the game. This allowed us to create the clustering on our map in Figure 10.

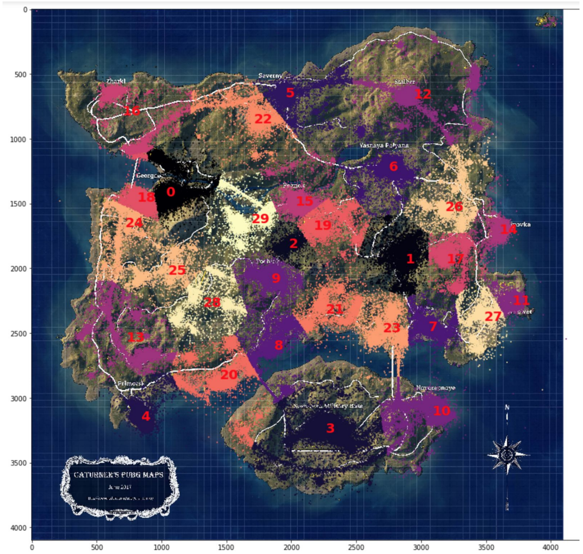
We then hoped to look at how the players' positions changed due to the reduction in play size. So we looked at the time interval that the first circle is still visible. We produced the clustering in Figure 11. We chose to look only at the first and second reductions (Figure 12), because after the these playzone size decrements, your choices become ever increasingly limited with where you can choose to go.



```
Center : Score
0 : 5.136889
1 : 2.428687
2 : 12.689553
3 : 9.601417
4 : 6.600788
5 : 4.541331
6 : 1.053509
7 : 0.928041
8 : 4.906699
9 : 4.762583
10 : 0.471083
11 : 1.179749
12 : 1.671246
13 : 2.484192
14 : 0.824643
15 : 1.703137
16 : 1.902760
17 : 5.408349
18 : 2.334448
19 : 1.062226
20 : 0.590481
21 : 1.057260
22 : 1.912250
23 : 1.463458
24 : 0.630207
25 : 0.364926
26 : 11.680405
27 : 0.665850
28 : 1.045894
29 : 0.671478
30 : 0.701272
31 : 0.419219
32 : 2.332131
33 : 0.277861
34 : 0.423964
35 : 1.270125
36 : 0.140475
37 : 0.277419
38 : 0.692444
39 : 1.691551
```

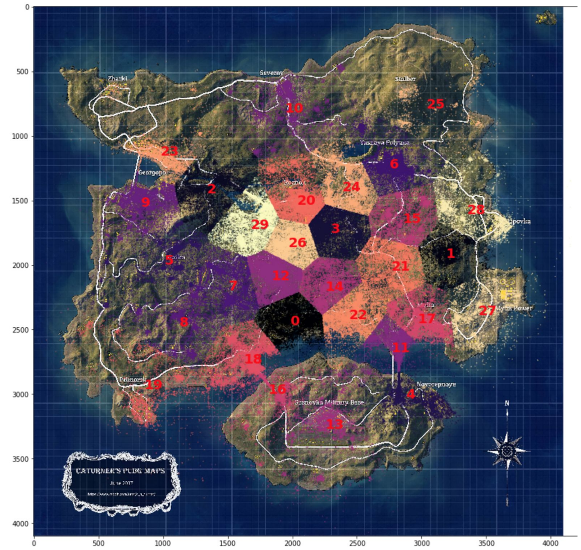
Figure 10: Map of clustering for jump locations, deaths that occurred in the first

100 seconds of a game. Also the chart showing the score for each cluster.



```
Center : Score
0 : 6.967781
1 : 4.105156
2 : 1.746695
3 : 16.547297
4 : 2.095304
5 : 2.383978
6 : 2.720461
7 : 2.850217
8 : 2.228776
9 : 11.258283
10 : 4.081673
11 : 0.903035
12 : 1.922156
13 : 0.720774
14 : 1.703162
15 : 5.299389
16 : 0.616604
17 : 4.175397
18 : 3.347580
19 : 8.690151
20 : 1.756228
21 : 1.781675
22 : 2.225481
23 : 1.407969
24 : 1.041343
25 : 1.405656
26 : 1.579014
27 : 1.979358
28 : 0.875836
29 : 1.583571
```

Figure 11: Map of clustering for first circle duration, deaths that occurred during the time interval [420 sec, 720 sec]. Also the chart showing the score for each cluster.



```
Center : Score
0 : 4.549785
1 : 1.877927
2 : 3.057446
3 : 6.306435
4 : 3.123961
5 : 2.061969
6 : 3.995159
7 : 3.187725
8 : 1.916436
9 : 3.895386
10 : 2.299773
11 : 4.030917
12 : 8.619711
13 : 2.095727
14 : 3.726848
15 : 2.603342
16 : 3.253990
17 : 2.993681
18 : 5.194182
19 : 1.000728
20 : 6.307935
21 : 3.238237
22 : 3.909639
23 : 1.436327
24 : 3.243738
25 : 0.472107
26 : 4.760583
27 : 1.268539
28 : 1.911935
29 : 3.659833
```

Figure 12: Map of clustering for second circle duration, deaths that occurred during

the time interval [720 sec, 920 sec]. Also the chart showing the score for each cluster.

During the mining for these graphs, we came across a very interesting interaction that we had never thought we would be able to visualize. When a game of PUBG starts, all of the players are on an airplane that flies across the map. The player may choose to jump wherever they wish, however, if they remain in the plane too long, they will be automatically ejected and dropped. This often happens when a play has disconnected from the game, and thus are not moving. In Figure 13 you can see this region where the plane forcibly grounds you because when the players don't move and then die they create the sort of ring shape encompassing the majority of the map. This path is that auto drop zone. We just found this a very interesting find in our data.

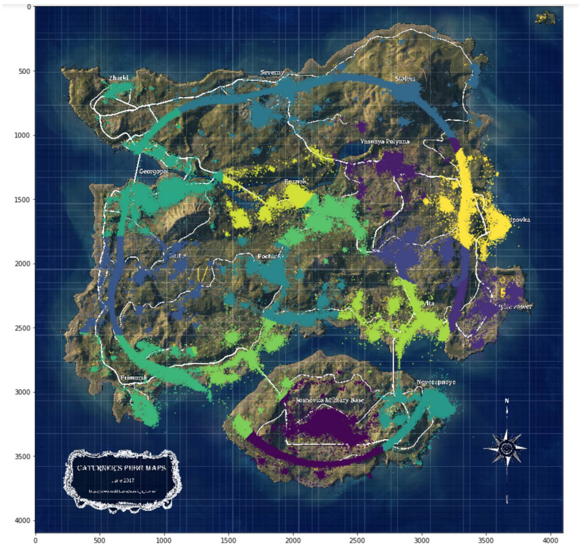


Figure 13: Interesting visual of plane auto drop zone.

Each of our graphs is given with a chart of the cluster number and the score of that given cluster. These scores are the normalized sums of the number of deaths that occurred within that cluster. In this way, we were able to determine a sort of popularity for certain locations that match up well with how the game is played. (At least in our experience.) These scores allow anyone who looks at our maps to make a more informed decision about location visitation frequency by other players when deciding to go somewhere. The way the score works is that the higher the number, the more likely there will be a larger amount of people at the location (scalable with score value).

Applications

Weapon range:
As we gain knowledge about the most used range for each weapon, we are able to help new players choose their weapon based on the distance between them and their foes. For example, if the player want to kill enemies who are close up, they may want to choose m416 over SKS because 77% of kill distance for the m416 are shorter range (50 meters) and only 48% of kill distance for the SKS are in the 75 meters range.

Skill vs luck:
From our four different clusters we got from k-means clustering. We discovered that a player's luck (total travel distance) only have a little effects on player's final ranking. However, people who have higher skills tend to have higher final ranking. Therefore, as a newbie, instead of worry about where to jump, try to improve your shooting skills. One way to do this is to jump to the high player density area like school, and airport.

Reference

1. Egerland, Christoph. “PUBG Data Analysis.” *PUBG Data Analysis | Kaggle*, Jan. 2018, www.kaggle.com/chegerland/pubg-data-analysis.
2.Moore, Justin. “To Drive... or Not to Drive...” *PUBG Data Analysis*, 8 Aug. 2017, pubganalysis.wordpress.com/2017/08/08/to-drive-or-not-to-drive/

Figure 10: https://imgur.com/6YY2ozG
Figure 11: https://imgur.com/HGTMJEa
Figure 12: https://imgur.com/KjtT8We
Figure 13: https://imgur.com/BShgUVe