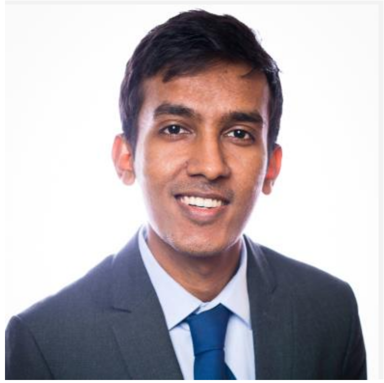
BIG DATA PROJECT – GROUP 7

ANALYZING PUBG GAME DATA

Our Team



Harsh Gupta
HXG171830

Siddharth Oza
SRO170030

Devarsh Patel
DXP173530

Adit Kansara
ASK172030

Player Unknown's Battle Grounds (PUBG)

100 players are dropped onto an island empty-handed and must explore, scavenge, and eliminate other players until only one is left standing, all while the play zone continues to shrink.

The Game

TRENDING

Released worldwide in December 2017.

MASS USERBASE

227 million monthly players, 87 million daily players, 400 million players till date

WORLD RECORD

World record for most simultaneous players at once

REVENUE

113 million monthly revenue, ~700k earning from daily user spending

MULTI PLATFORM

Available on Windows, Android, iOS, and Xbox

VIDEO

2.03 billion minutes of viewing on Twitch

Features of Our Data

4.5 MILLION ROWS

26 COLUMNS

47k+ MATCH DATA

1.88 MILLION GROUPS

3 STRING VARIABLES,
23 INTEGER VARIABLES

DATA SCOURCE: KAGGLE

METADATA

MATCHID

ID to identify match.

GROUPID

ID to identify a group within a match.

NUMGROUPS

Number of groups we have data for in the match.

MATCHTYPE

String identifying the game mode that the data comes from. The standard modes are “solo”, “duo”, “squad”

KILLS

Number of enemy players killed.

HEADSHOTKILLS

Number of enemy players killed with headshots.

VEHICLEDESTROYS

Number of vehicles destroyed.

WEAPONSACQUIRED

Number of weapons picked up.

HEALS

Number of healing items used.

REVIVES

Number of times this player revived teammates.

SWIMDISTANCE

Total distance traveled by swimming measured in meters.

WINPLACEPERC

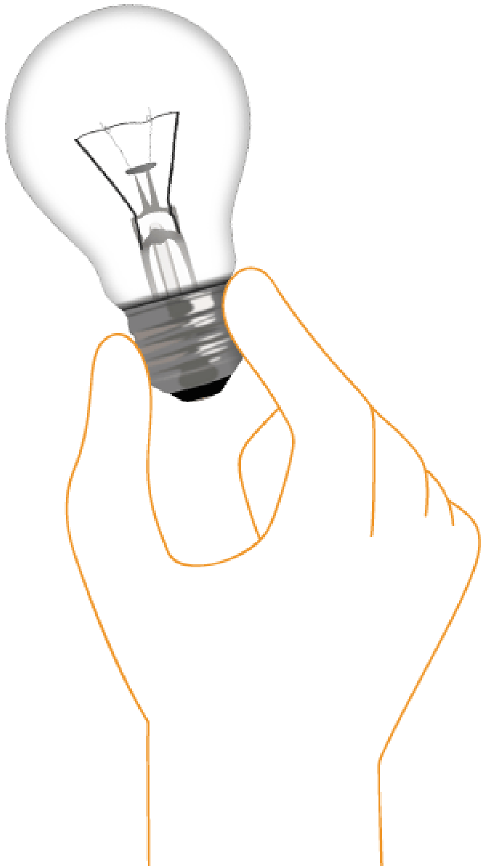
This is a percentile winning placement, where 1 corresponds to 1st place, and 0 corresponds to last place in the match.

METADATA

- DBNOs - Number of enemy players knocked.
- assists - Number of enemy players this player damaged that were killed by teammates.
- boosts - Number of boost items used.
- damageDealt - Total damage dealt. Note: Self inflicted damage is subtracted.
- headshotKills - Number of enemy players killed with headshots.
- heals - Number of healing items used.
- Id - Player's Id
- killPlace - Ranking in match of number of enemy players killed.
- killPoints - Kills-based external ranking of player. (Think of this as an Elo ranking where only kills matter.) If there is a value other than -1 in rankPoints, then any 0 in killPoints should be treated as a "None".
- killStreaks - Max number of enemy players killed in a short amount of time.
- kills - Number of enemy players killed.
- longestKill - Longest distance between player and player killed at time of death. This may be misleading, as downing a player and driving away may lead to a large longestKill stat.
- matchDuration - Duration of match in seconds.
- matchId - ID to identify match. There are no matches that are in both the training and testing set.
- matchType - String identifying the game mode that the data comes from. The standard modes are "solo", "duo", "squad", "solo-fpp", "duo-fpp", and "squad-fpp"; other modes are from events or custom matches.
- rankPoints - Elo-like ranking of player. This ranking is inconsistent and is being deprecated in the API's next version, so use with caution. Value of -1 takes place of "None".
- revives - Number of times this player revived teammates.
- rideDistance - Total distance traveled in vehicles measured in meters.
- roadKills - Number of kills while in a vehicle.
- swimDistance - Total distance traveled by swimming measured in meters.
- teamKills - Number of times this player killed a teammate.
- vehicleDestroys - Number of vehicles destroyed.
- walkDistance - Total distance traveled on foot measured in meters.
- weaponsAcquired - Number of weapons picked up.
- winPoints - Win-based external ranking of player. (Think of this as an Elo ranking where only winning matters.) If there is a value other than -1 in rankPoints, then any 0 in winPoints should be treated as a "None".
- groupId - ID to identify a group within a match. If the same group of players plays in different matches, they will have a different groupId each time.
- numGroups - Number of groups we have data for in the match.
- maxPlace - Worst placement we have data for in the match. This may not match with numGroups, as sometimes the data skips over placements.
- winPlacePerc - The target of prediction. This is a percentile winning placement, where 1 corresponds to 1st place, and 0 corresponds to last place in the match. It is calculated off of maxPlace, not numGroups, so it is possible to have missing chunks in a match.

Our Approach

Enhancing the game experience using **insights**



“Does killing more people increases the chance of winning the game?”

APPROACH

USING THE CORRELATION BETWEEN THE MATCH WINNING PERCENTAGE AND NUMBER OF KILLS TO DETERMINE THE RELATIONSHIP BETWEEN THE TWO.

Columns Used

WINPLACEPERC, KILLS

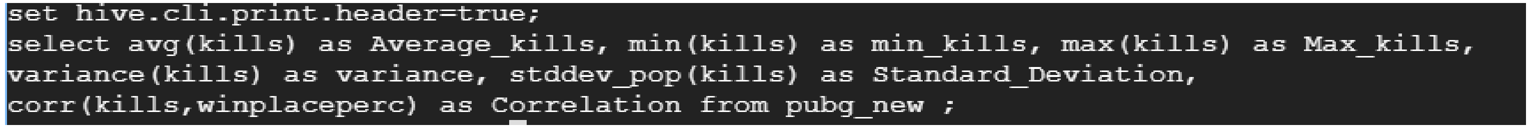
Data Pre-processing

None

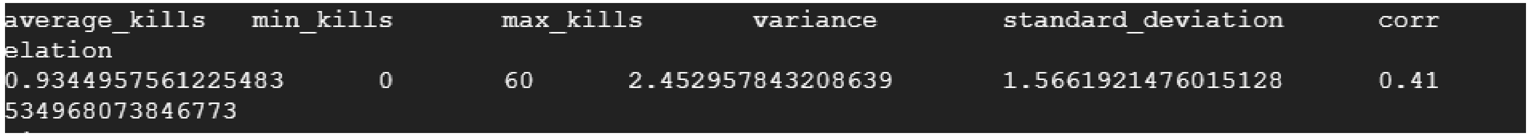
Tool Used

Hive

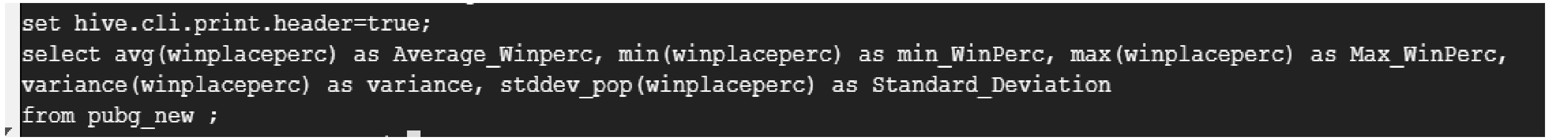
DATA ANALYSIS



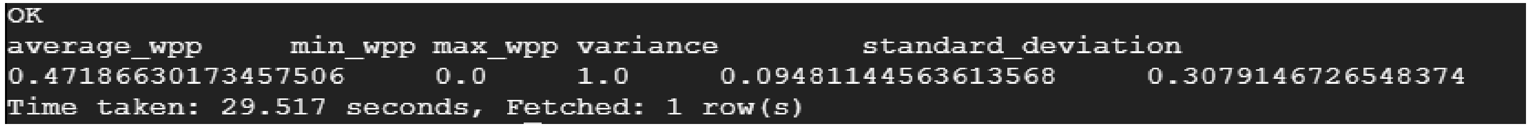
```
set hive.cli.print.header=true;
select avg(kills) as Average_kills, min(kills) as min_kills, max(kills) as Max_kills,
variance(kills) as variance, stddev_pop(kills) as Standard_Deviation,
corr(kills,winplaceperc) as Correlation from pubg_new ;
```



```
average_kills   min_kills       max_kills       variance        standard_deviation      corr
elation
0.9344957561225483      0       60      2.452957843208639       1.5661921476015128      0.41
534968073846773
```



```
set hive.cli.print.header=true;
select avg(winplaceperc) as Average_Winperc, min(winplaceperc) as min_WinPerc, max(winplaceperc) as Max_WinPerc,
variance(winplaceperc) as variance, stddev_pop(winplaceperc) as Standard_Deviation
from pubg_new ;
```



```
OK
average_wpp     min_wpp max_wpp variance        standard_deviation
0.47186630173457506     0.0     1.0     0.09481144563613568     0.3079146726548374
Time taken: 29.517 seconds, Fetched: 1 row(s)
```

	WINPLACEPERC	KILLS
Max	1	60
Min	0	0
Average	0.47	0.93
Standard Deviation	0.31	1.56
Variance	0.09	2.45
Missing Values	0	0
Correlation	**0.4153**	

“Can we predict the winner of the game?”

APPROACH

CLASSIFICATION PROBLEM: DIVIDE THE DATA INTO WINNERS AND LOSERS. DESIGN AND TEST A MODEL USING VARIOUS CLASSIFICATION ALGORITHMS TO PREDICT IF A PLAYER WILL WIN/LOSE.

Columns Used

WINORLOSE, WINPLACEPERC, All Columns

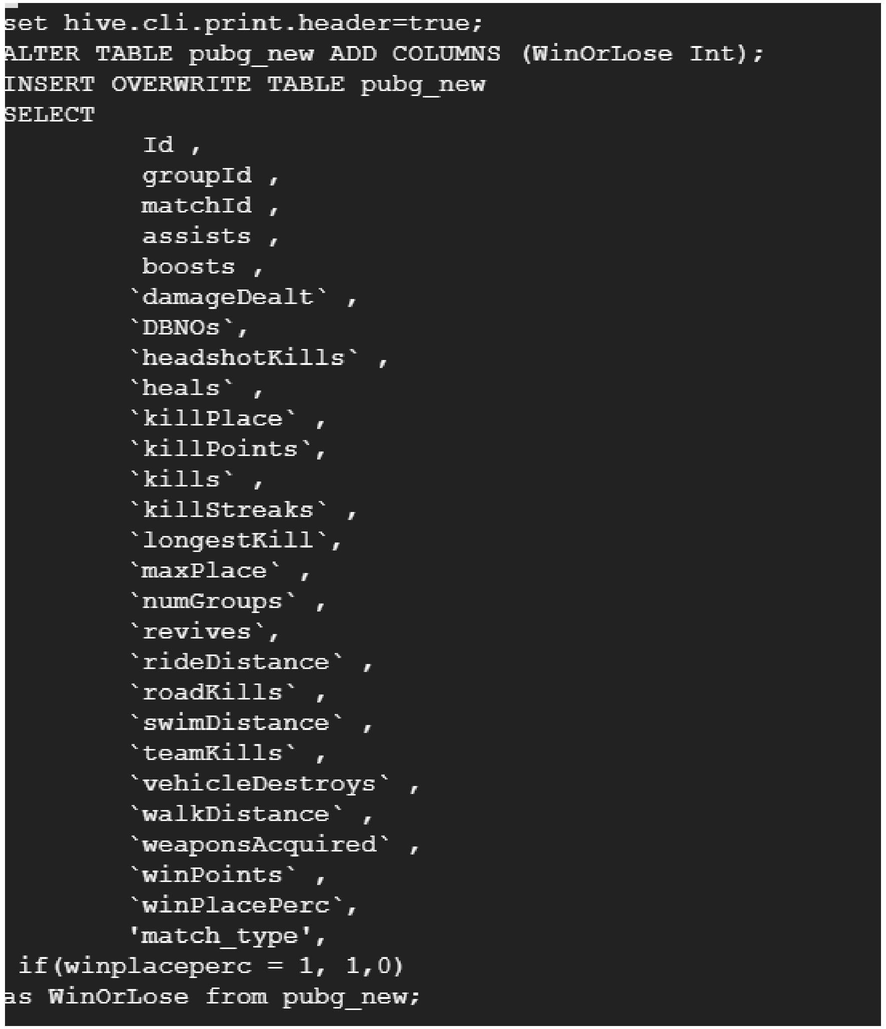
Data Pre-processing

Create new column "WINORLOSE" which will have value 1 for all the WINPLACEPERC=1 and 0 otherwise. Data Standardization.

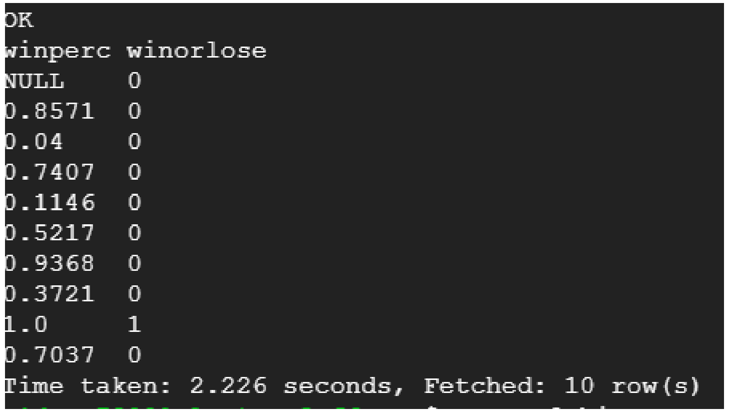
Tool Used

Hive, Spark

DATA ANALYSIS

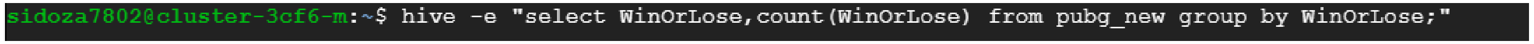


```
set hive.cli.print.header=true;
ALTER TABLE pubg_new ADD COLUMNS (WinOrLose Int);
INSERT OVERWRITE TABLE pubg_new
SELECT
         Id ,
         groupId ,
         matchId ,
         assists ,
         boosts ,
        `damageDealt` ,
        `DBNOs`,
        `headshotKills` ,
        `heals` ,
        `killPlace` ,
        `killPoints`,
        `kills` ,
        `killStreaks` ,
        `longestKill`,
        `maxPlace` ,
        `numGroups` ,
        `revives`,
        `rideDistance` ,
        `roadKills` ,
        `swimDistance` ,
        `teamKills` ,
        `vehicleDestroys` ,
        `walkDistance` ,
        `weaponsAcquired` ,
        `winPoints` ,
        `winPlacePerc`,
        'match_type',
 if(winplaceperc = 1, 1,0)
as WinOrLose from pubg_new;
```

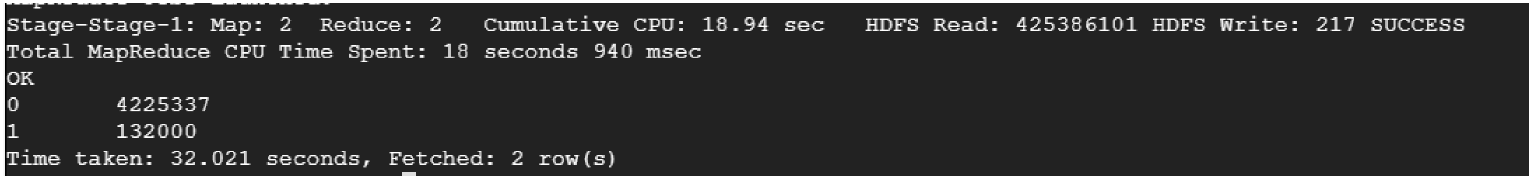


```
OK
winperc winorlose
NULL    0
0.8571  0
0.04    0
0.7407  0
0.1146  0
0.5217  0
0.9368  0
0.3721  0
1.0     1
0.7037  0
Time taken: 2.226 seconds, Fetched: 10 row(s)
```

Creating new column “WINORLOSE” and validating it.

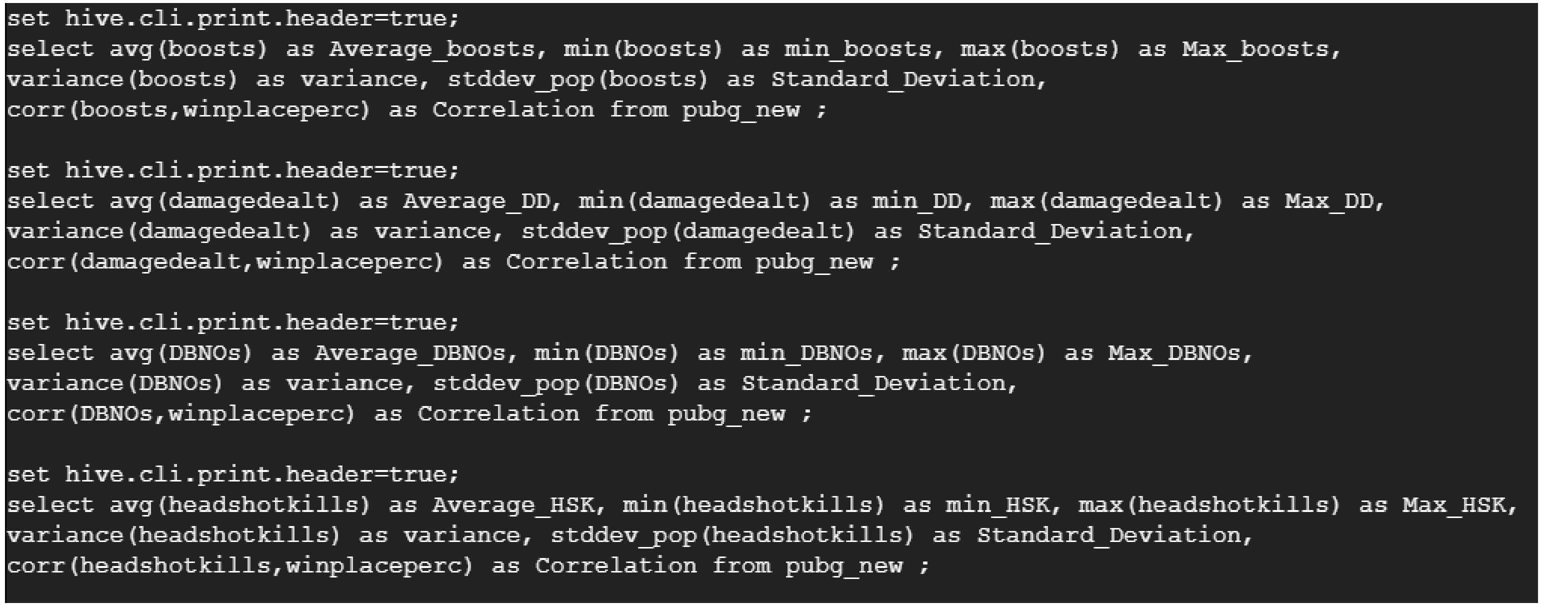


```
sidoza7802@cluster-3cf6-m:~$ hive -e "select WinOrLose,count(WinOrLose) from pubg_new group by WinOrLose;"
```

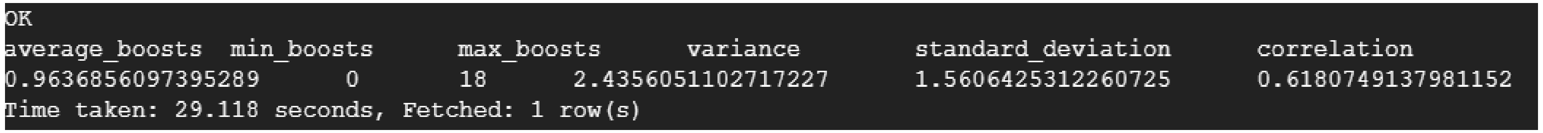


```
Stage-Stage-1: Map: 2  Reduce: 2   Cumulative CPU: 18.94 sec   HDFS Read: 425386101 HDFS Write: 217 SUCCESS
Total MapReduce CPU Time Spent: 18 seconds 940 msec
OK
0       4225337
1       132000
Time taken: 32.021 seconds, Fetched: 2 row(s)
```

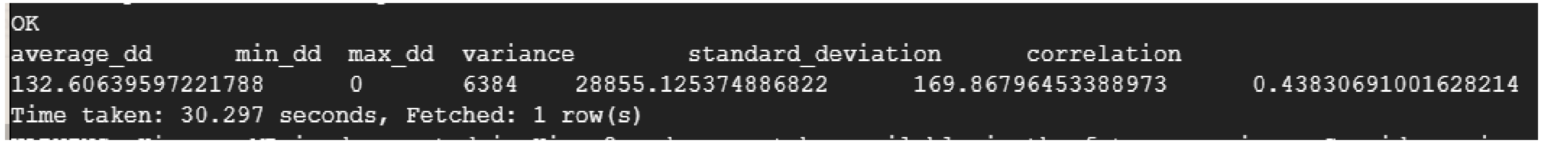
DATA ANALYSIS



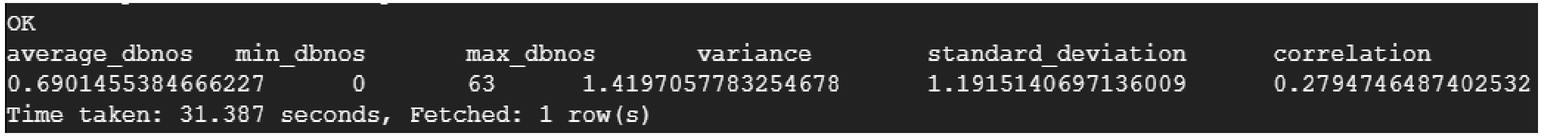
```
set hive.cli.print.header=true;
select avg(boosts) as Average_boosts, min(boosts) as min_boosts, max(boosts) as Max_boosts,
variance(boosts) as variance, stddev_pop(boosts) as Standard_Deviation,
corr(boosts,winplaceperc) as Correlation from pubg_new ;

set hive.cli.print.header=true;
select avg(damagedealt) as Average_DD, min(damagedealt) as min_DD, max(damagedealt) as Max_DD,
variance(damagedealt) as variance, stddev_pop(damagedealt) as Standard_Deviation,
corr(damagedealt,winplaceperc) as Correlation from pubg_new ;

set hive.cli.print.header=true;
select avg(DBNOs) as Average_DBNOs, min(DBNOs) as min_DBNOs, max(DBNOs) as Max_DBNOs,
variance(DBNOs) as variance, stddev_pop(DBNOs) as Standard_Deviation,
corr(DBNOs,winplaceperc) as Correlation from pubg_new ;

set hive.cli.print.header=true;
select avg(headshotkills) as Average_HSK, min(headshotkills) as min_HSK, max(headshotkills) as Max_HSK,
variance(headshotkills) as variance, stddev_pop(headshotkills) as Standard_Deviation,
corr(headshotkills,winplaceperc) as Correlation from pubg_new ;
```



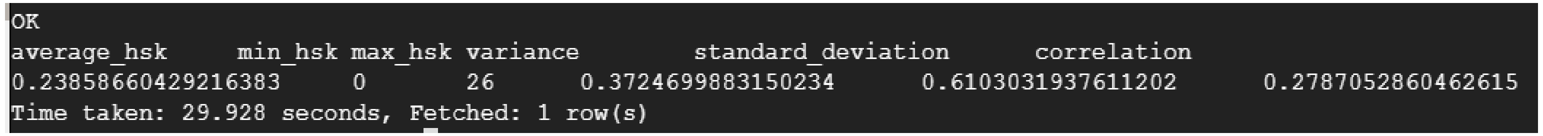
```
OK
average_boosts  min_boosts      max_boosts      variance        standard_deviation      correlation
0.9636856097395289      0       18      2.4356051102717227      1.5606425312260725      0.6180749137981152
Time taken: 29.118 seconds, Fetched: 1 row(s)
```



```
OK
average_dd      min_dd  max_dd  variance        standard_deviation      correlation
132.60639597221788      0       6384    28855.125374886822      169.86796453388973      0.43830691001628214
Time taken: 30.297 seconds, Fetched: 1 row(s)
```



```
OK
average_dbnos   min_dbnos       max_dbnos       variance        standard_deviation      correlation
0.6901455384666227      0       63      1.4197057783254678      1.1915140697136009      0.2794746487402532
Time taken: 31.387 seconds, Fetched: 1 row(s)
```



```
OK
average_hsk     min_hsk max_hsk variance        standard_deviation      correlation
0.23858660429216383     0       26      0.3724699883150234      0.6103031937611202      0.2787052860462615
Time taken: 29.928 seconds, Fetched: 1 row(s)
```

	BOOSTS	DAMAGE DEALT	DBNO's	HEADSHOT KILLS
Max	18	6384	63	26
Min	0	0	0	0
Average	0.96	132.6	0.69	0.23
Standard Deviation	1.56	169.86	1.19	0.61
Variance	2.43	28855.12	1.41	0.37
Missing Values	0	0	0	0
Correlation with win percentage	**0.61**	**0.43**	**0.27**	**0.27**

“Can we predict the finishing position of a player in the game?”

APPROACH

REGRESSION PROBLEM: DESIGN AND TEST A MODEL USING VARIOUS REGRESSION ALGORITHMS TO PREDICT THE FINAL POSITION OF THE PLAYER AT THE END OF THE GAME.

Columns Used

WINPLACEPERC, All Columns

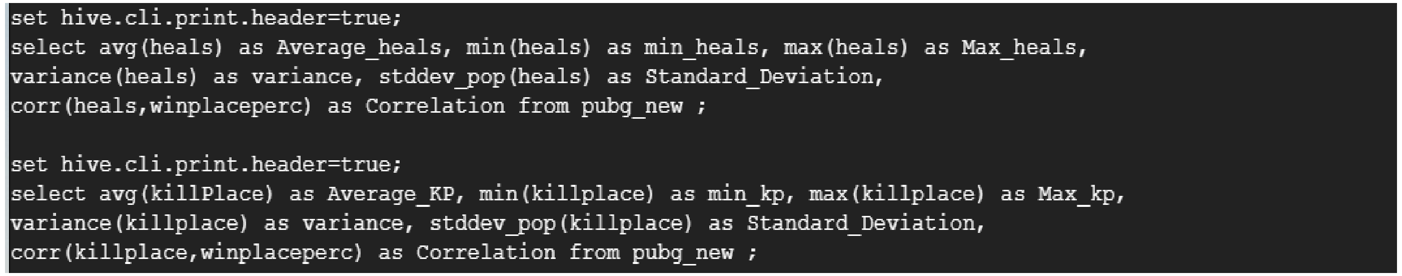
Data Pre-processing

Data standardization

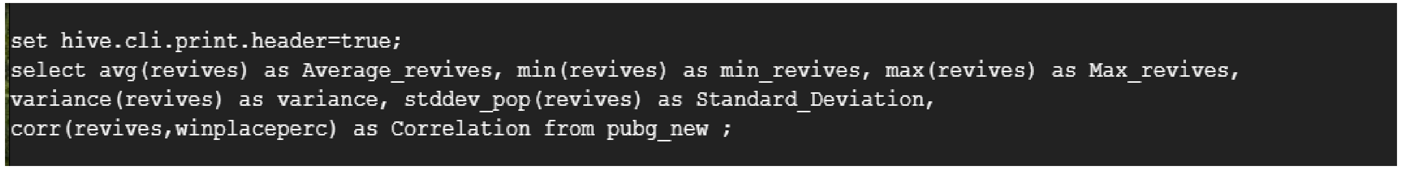
Tool Used

Hive, Spark

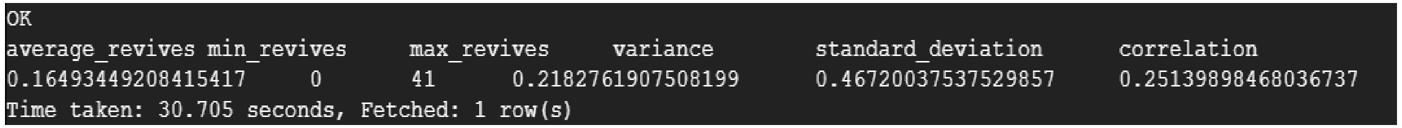
DATA ANALYSIS



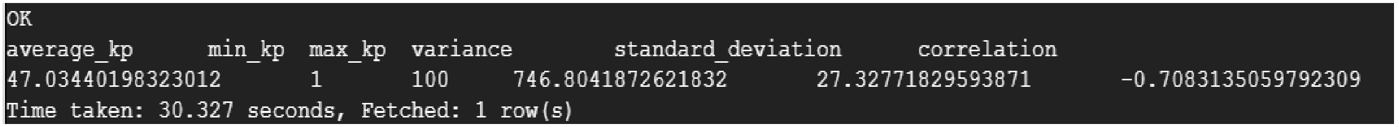
```
set hive.cli.print.header=true;
select avg(heals) as Average_heals, min(heals) as min_heals, max(heals) as Max_heals,
variance(heals) as variance, stddev_pop(heals) as Standard_Deviation,
corr(heals,winplaceperc) as Correlation from pubg_new ;

set hive.cli.print.header=true;
select avg(killPlace) as Average_KP, min(killplace) as min_kp, max(killplace) as Max_kp,
variance(killplace) as variance, stddev_pop(killplace) as Standard_Deviation,
corr(killplace,winplaceperc) as Correlation from pubg_new ;
```



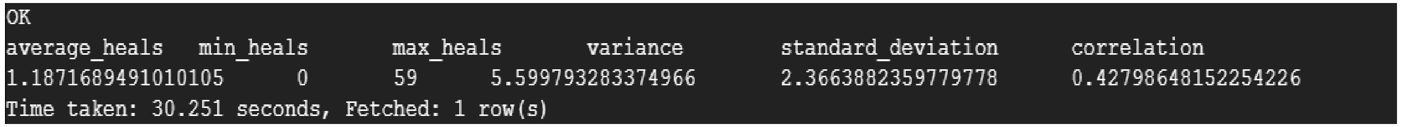
```
set hive.cli.print.header=true;
select avg(revives) as Average_revives, min(revives) as min_revives, max(revives) as Max_revives,
variance(revives) as variance, stddev_pop(revives) as Standard_Deviation,
corr(revives,winplaceperc) as Correlation from pubg_new ;
```



```
OK
average_revives min_revives     max_revives     variance        standard_deviation      correlation
0.16493449208415417     0       41      0.2182761907508199      0.46720037537529857     0.25139898468036737
Time taken: 30.705 seconds, Fetched: 1 row(s)
```



```
OK
average_kp      min_kp  max_kp  variance        standard_deviation      correlation
47.03440198323012       1       100     746.8041872621832       27.32771829593871       -0.7083135059792309
Time taken: 30.327 seconds, Fetched: 1 row(s)
```



```
OK
average_heals   min_heals       max_heals       variance        standard_deviation      correlation
1.1871689491010105      0       59      5.599793283374966       2.3663882359779778      0.42798648152254226
Time taken: 30.251 seconds, Fetched: 1 row(s)
```

	HEALS	KILLPLACE	REVIVES
Max	59	100	41
Min	0	1	0
Average	1.18	47.03	0.16
Standard Deviation	2.36	27.32	0.47
Variance	5.59	746.80	0.22
Missing Values	0	0	0
Correlation with win percentage	**0.43**	**-0.71**	**0.25**

“How different/similar are the strategies required to win the game when playing solo, duo, or in a group?”

APPROACH

Divide the data on the basis of match type. Run regression analysis on the these three types independently, to determine the coefficients affecting each match type.

Columns Used

Major: NUMGROUPS, Derived: Match_Type, All other columns

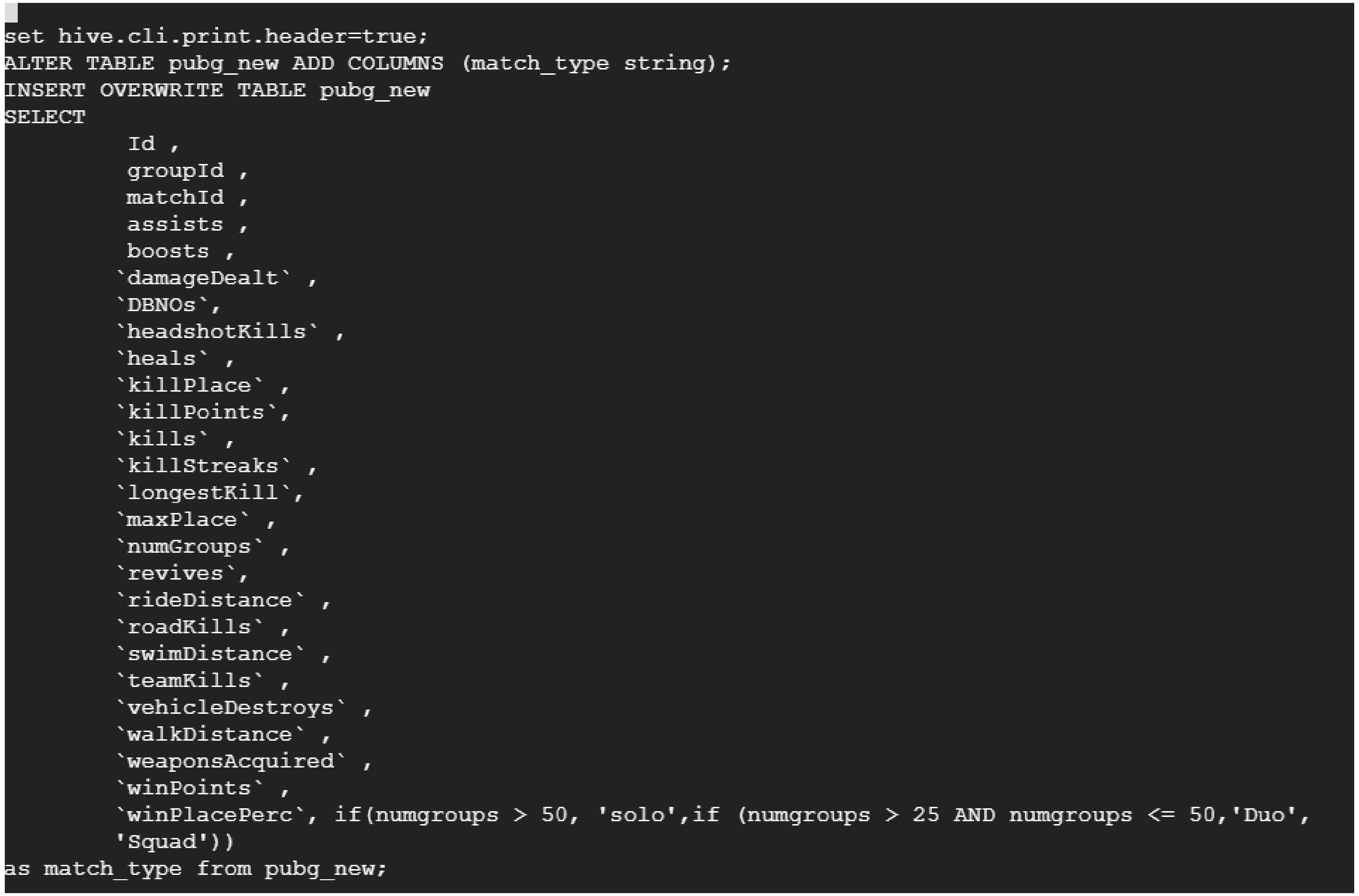
Data Pre-processing

Create a new column from the no of groups column which will act as a filter. Data Standardization.

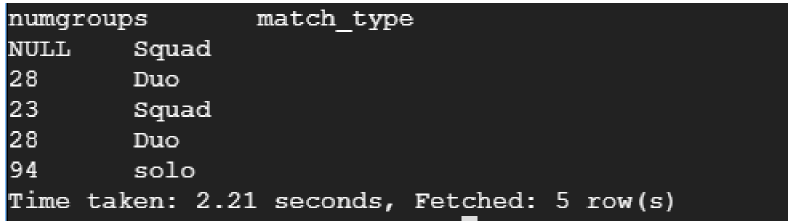
Tool Used

Hive, Spark

DATA ANALYSIS

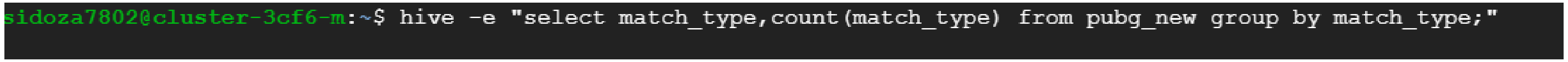


```
set hive.cli.print.header=true;
ALTER TABLE pubg_new ADD COLUMNS (match_type string);
INSERT OVERWRITE TABLE pubg_new
SELECT
         Id ,
         groupId ,
         matchId ,
         assists ,
         boosts ,
        `damageDealt` ,
        `DBNOs`,
        `headshotKills` ,
        `heals` ,
        `killPlace` ,
        `killPoints`,
        `kills` ,
        `killStreaks` ,
        `longestKill`,
        `maxPlace` ,
        `numGroups` ,
        `revives`,
        `rideDistance` ,
        `roadKills` ,
        `swimDistance` ,
        `teamKills` ,
        `vehicleDestroys` ,
        `walkDistance` ,
        `weaponsAcquired` ,
        `winPoints` ,
        `winPlacePerc`, if(numgroups > 50, 'solo',if (numgroups > 25 AND numgroups <= 50,'Duo',
        'Squad'))
as match_type from pubg_new;
```

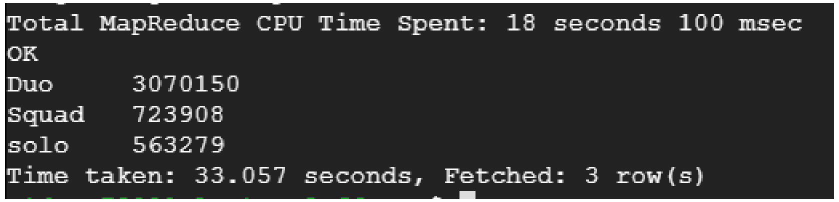


```
numgroups       match_type
NULL    Squad
28      Duo
23      Squad
28      Duo
94      solo
Time taken: 2.21 seconds, Fetched: 5 row(s)
```

Creating new column "MATCH_TYPE" and validating it.



```
sidoza7802@cluster-3cf6-m:~$ hive -e "select match_type,count(match_type) from pubg_new group by match_type;"
```



```
Total MapReduce CPU Time Spent: 18 seconds 100 msec
OK
Duo     3070150
Squad   723908
solo    563279
Time taken: 33.057 seconds, Fetched: 3 row(s)
```

“How do we catch the cheaters in the game?”

APPROACH

USING VARIOUS LOGICAL CONDITIONS BASED ON GAME KNOWLEDGE TO DETERMINE CHEATERS IN THE GAME.

Columns Used

WINPLACEPERC, KILLS, RIDE DISTANCE, WALK DISTANCE

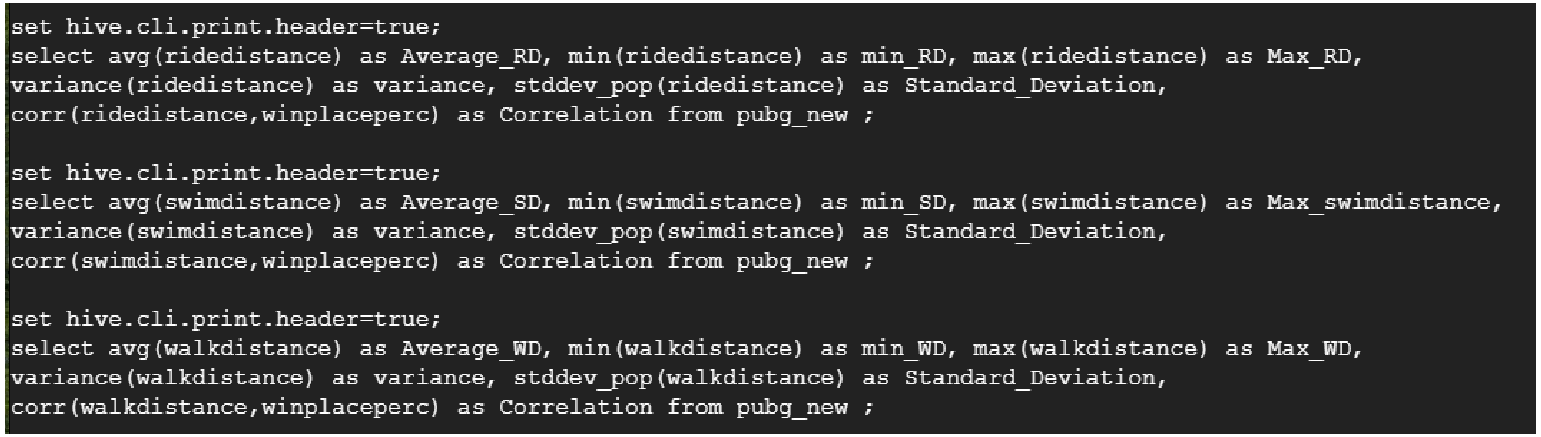
Data Pre-processing

None

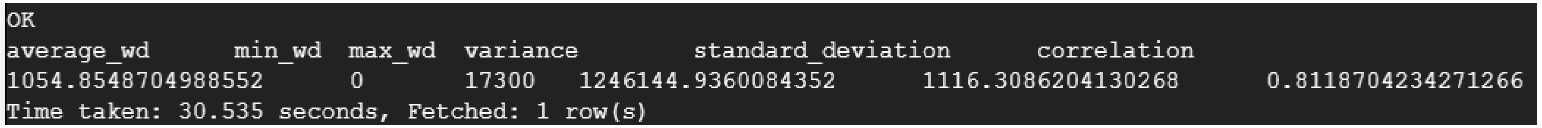
Tool Used

Hive

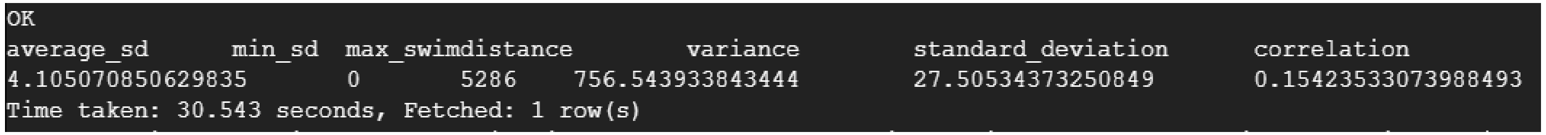
DATA ANALYSIS



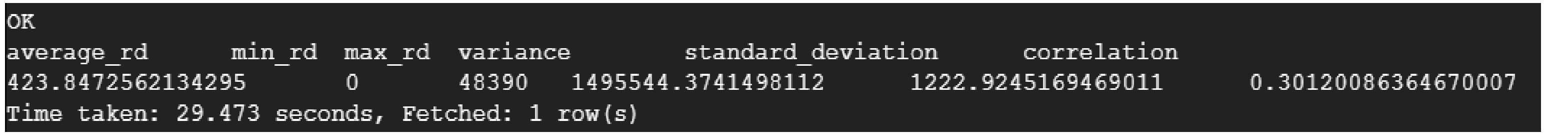
```
set hive.cli.print.header=true;
select avg(ridedistance) as Average_RD, min(ridedistance) as min_RD, max(ridedistance) as Max_RD,
variance(ridedistance) as variance, stddev_pop(ridedistance) as Standard_Deviation,
corr(ridedistance,winplaceperc) as Correlation from pubg_new ;

set hive.cli.print.header=true;
select avg(swimdistance) as Average_SD, min(swimdistance) as min_SD, max(swimdistance) as Max_swimdistance,
variance(swimdistance) as variance, stddev_pop(swimdistance) as Standard_Deviation,
corr(swimdistance,winplaceperc) as Correlation from pubg_new ;

set hive.cli.print.header=true;
select avg(walkdistance) as Average_WD, min(walkdistance) as min_WD, max(walkdistance) as Max_WD,
variance(walkdistance) as variance, stddev_pop(walkdistance) as Standard_Deviation,
corr(walkdistance,winplaceperc) as Correlation from pubg_new ;
```



```
OK
average_wd      min_wd  max_wd  variance        standard_deviation      correlation
1054.8548704988552      0       17300   1246144.9360084352      1116.3086204130268      0.8118704234271266
Time taken: 30.535 seconds, Fetched: 1 row(s)
```



```
OK
average_sd      min_sd  max_swimdistance        variance        standard_deviation      correlation
4.105070850629835       0       5286    756.543933843444        27.50534373250849       0.15423533073988493
Time taken: 30.543 seconds, Fetched: 1 row(s)
```



```
OK
average_rd      min_rd  max_rd  variance        standard_deviation      correlation
423.8472562134295       0       48390   1495544.3741498112      1222.9245169469011      0.30120086364670007
Time taken: 29.473 seconds, Fetched: 1 row(s)
```

	RIDE DISTANCE	SWIM DISTANCE	WALK DISTANCE
Max	48390	5286	17300
Min	0	0	0
Average	423.84	4.11	1054.85
Standard Deviation	1222.92	27.50	1116.30
Variance	1495544	756.54	1246144
Missing Values	0	0	0
Correlation with win percentage	**0.30**	**0.15**	**0.81**

“How does the weapon acquisition strategy differ for players in different clusters?”

APPROACH

Form clusters of data using clustering algorithm/logical division. Run ANOVA to determine if the weapon acquisition differs significantly in different clusters of the data.

Columns Used

All Columns

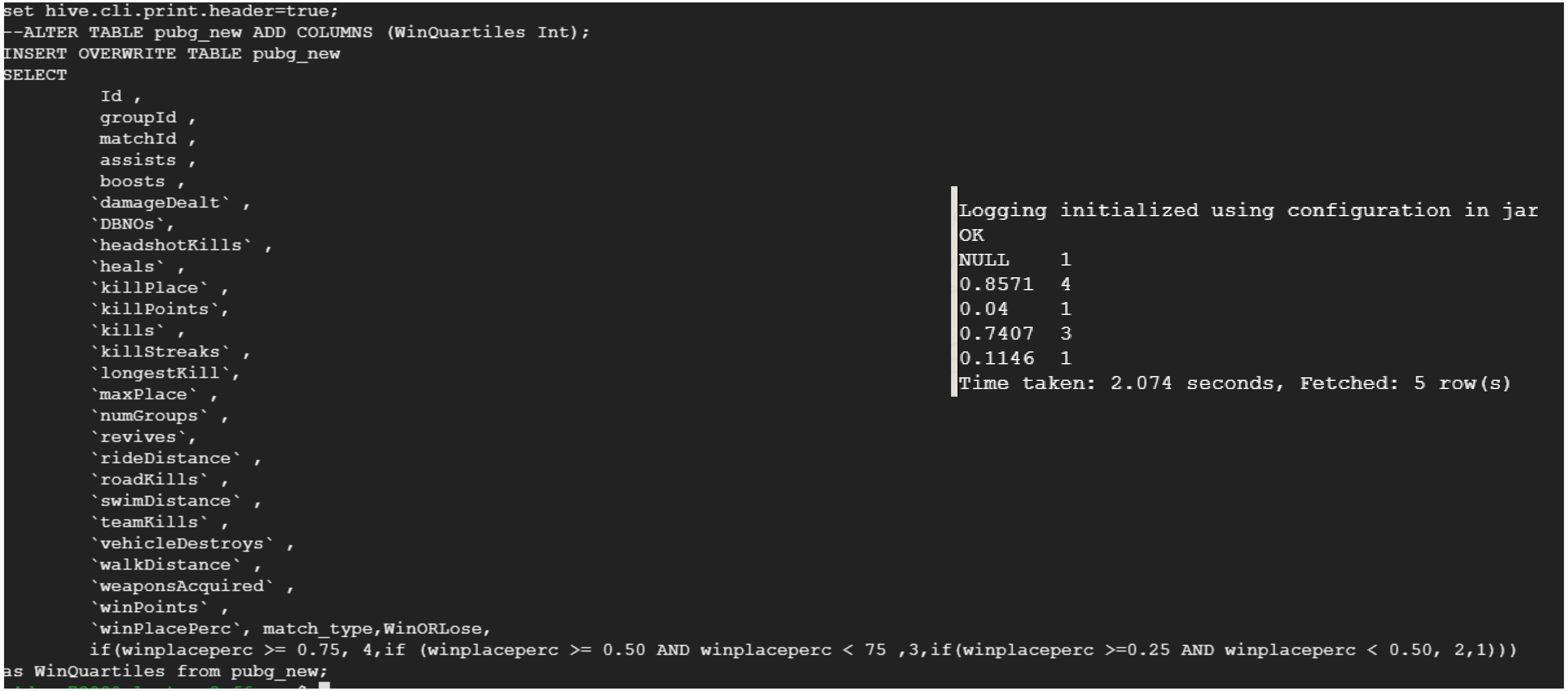
Data Pre-processing

Create data clusters.

Tool Used

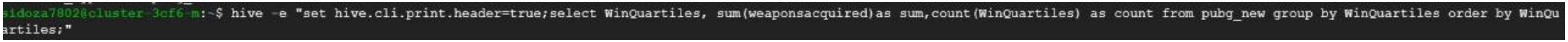
Hive, Spark

DATA ANALYSIS

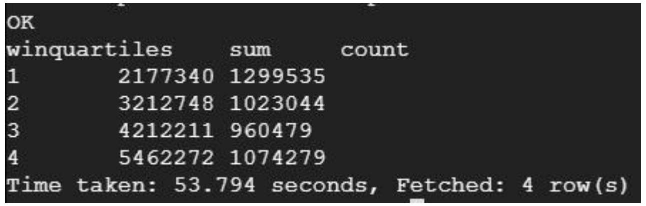


```
set hive.cli.print.header=true;
--ALTER TABLE pubg_new ADD COLUMNS (WinQuartiles Int);
INSERT OVERWRITE TABLE pubg_new
SELECT
         Id ,
         groupId ,
         matchId ,
         assists ,
         boosts ,
        `damageDealt` ,
        `DBNOs`,
        `headshotKills` ,
        `heals` ,
        `killPlace` ,
        `killPoints`,
        `kills` ,
        `killStreaks` ,
        `longestKill`,
        `maxPlace` ,
        `numGroups` ,
        `revives`,
        `rideDistance` ,
        `roadKills` ,
        `swimDistance` ,
        `teamKills` ,
        `vehicleDestroys` ,
        `walkDistance` ,
        `weaponsAcquired` ,
        `winPoints` ,
        `winPlacePerc`, match_type,WinORLose,
        if(winplaceperc >= 0.75, 4,if (winplaceperc >= 0.50 AND winplaceperc < 75 ,3,if(winplaceperc >=0.25 AND winplaceperc < 0.50, 2,1)))
as WinQuartiles from pubg_new;
```

```
Logging initialized using configuration in jar
OK
NULL    1
0.8571  4
0.04    1
0.7407  3
0.1146  1
Time taken: 2.074 seconds, Fetched: 5 row(s)
```



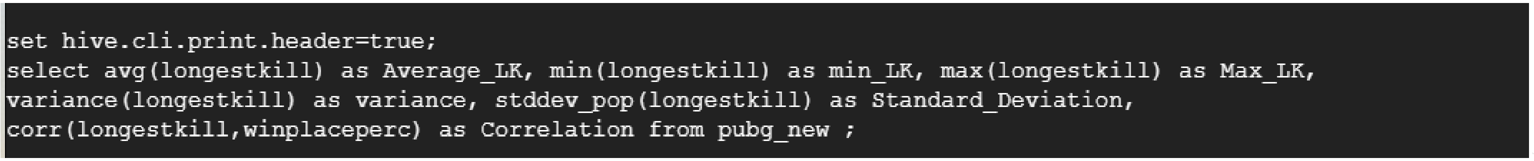
```
sidoza7802@cluster-3cf6-m:~$ hive -e "set hive.cli.print.header=true;select WinQuartiles, sum(weaponsacquired)as sum,count(WinQuartiles) as count from pubg_new group by WinQuartiles order by WinQu
artiles;"
```



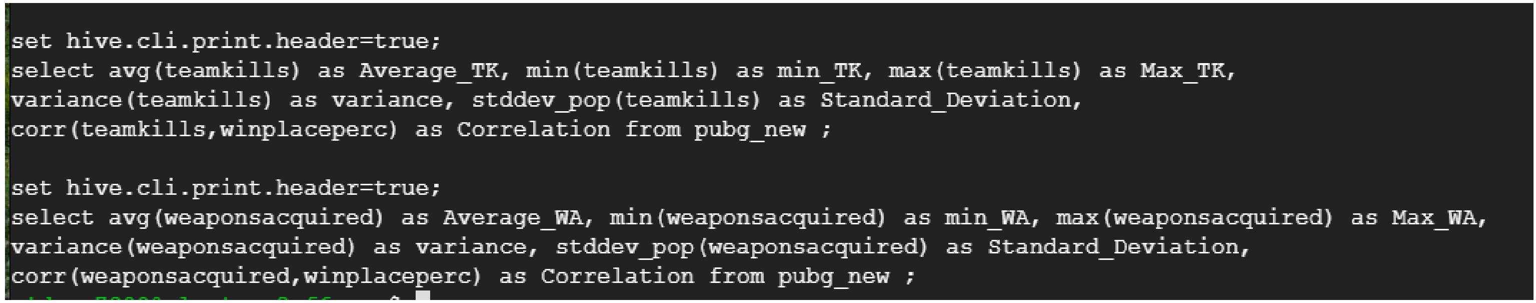
```
OK
winquartiles    sum     count
1       2177340 1299535
2       3212748 1023044
3       4212211 960479
4       5462272 1074279
Time taken: 53.794 seconds, Fetched: 4 row(s)
```

Creating new column "WINQUARTILES" and validating it.

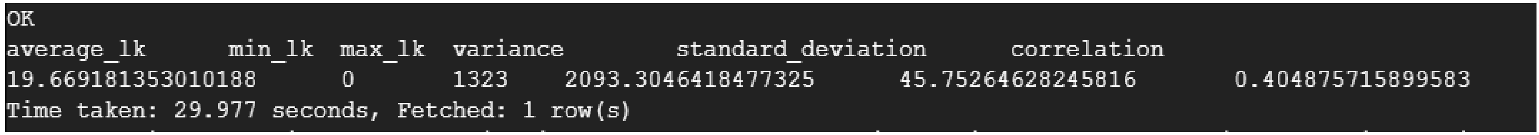
DATA ANALYSIS



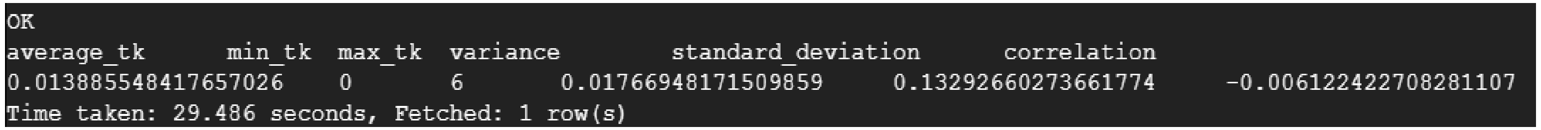
```
set hive.cli.print.header=true;
select avg(longestkill) as Average_LK, min(longestkill) as min_LK, max(longestkill) as Max_LK,
variance(longestkill) as variance, stddev_pop(longestkill) as Standard_Deviation,
corr(longestkill,winplaceperc) as Correlation from pubg_new ;
```



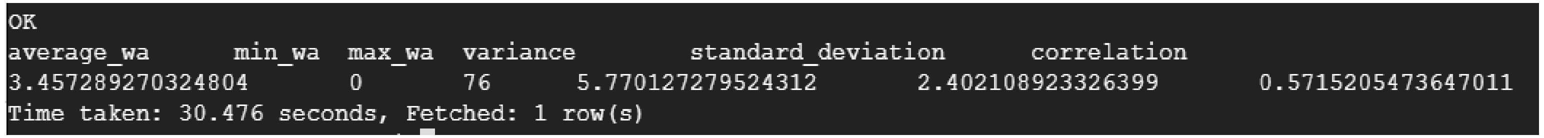
```
set hive.cli.print.header=true;
select avg(teamkills) as Average_TK, min(teamkills) as min_TK, max(teamkills) as Max_TK,
variance(teamkills) as variance, stddev_pop(teamkills) as Standard_Deviation,
corr(teamkills,winplaceperc) as Correlation from pubg_new ;

set hive.cli.print.header=true;
select avg(weaponsacquired) as Average_WA, min(weaponsacquired) as min_WA, max(weaponsacquired) as Max_WA,
variance(weaponsacquired) as variance, stddev_pop(weaponsacquired) as Standard_Deviation,
corr(weaponsacquired,winplaceperc) as Correlation from pubg_new ;
```



```
OK
average_lk      min_lk  max_lk  variance        standard_deviation      correlation
19.669181353010188      0       1323    2093.3046418477325      45.75264628245816       0.404875715899583
Time taken: 29.977 seconds, Fetched: 1 row(s)
```



```
OK
average_tk      min_tk  max_tk  variance        standard_deviation      correlation
0.013885548417657026    0       6       0.01766948171509859     0.13292660273661774     -0.006122422708281107
Time taken: 29.486 seconds, Fetched: 1 row(s)
```



```
OK
average_wa      min_wa  max_wa  variance        standard_deviation      correlation
3.457289270324804       0       76      5.770127279524312       2.402108923326399       0.5715205473647011
Time taken: 30.476 seconds, Fetched: 1 row(s)
```

	LONGEST KILL	TEAM KILLS	WEAPONS ACQUIRED
Max	1323	6	76
Min	0	0	0
Average	19.66	0.013	3.45
Standard Deviation	45.75	0.13	2.40
Variance	2093.30	0.017	5.77
Missing Values	0	0	0
Correlation with win percentage	**0.40**	**-0.006**	**0.57**

Thank You.