**Project Final Report**

**Team member:**

Yue Han V00785781

Huiying Wang V00794342

Jianzhao Liu V00812054

**Membership changement:**

Tony has dropped the class, so we only have four teammates in the team. Li has been removed from our team.

**Description:**

The topic we choose to develop is a game named PlayerUnknown’s Battlegrounds (PUBG).

***PlayerUnknown's Battlegrounds*** (***PUBG***) is a multiplayer online battle royale video game developed by PUBG crop. Battleground is an action game in which up to 100 players fight in a battle royale, a type of last man standing deathmatch game where players try to be the last alive to win the game.

Each match starts with players parachuting from a plane onto a map area approximately 8 by 8 kilometers. Players need to determine the best time to eject and parachute to the ground. After landing, players need to scavenge for weapons and other gear to survive. Players can hide into “shelters” to avoid getting killed by other players. As time pass by, the available safe area randomly decreases in size and players who are not entered in time will be continue to drop health.

We will attempt to use the dataset we found online and analyze the factor in their gameplay data to predict the possibility of winning in three different match modes.

**Solo mode:**

Players are all by themsleves.  
**Duo mode:**

Players will play in two as a team. If one of your teammates are the last alive player, you will win the game as your team will win the game.

**Squad mode:**

Players will play in four as a team. You will win the game, if one of your teammates alive in the end.

For this project, we will analyse Single mode as the only situation for predicting.

There are some specific factor we will use in our prediction:

|  |  |
| --- | --- |
| Attribute | Description |
| Movedistance | Movedistance include Walkdsitance and Ridedistance. Players need to keep moving as the safe zone randomly shrinks and search buildings to acquire weapons and other gears. |
| DPG(Damage per Game) | Players with higher hit rate have higher DPG. Higher DPG aso means higher possibility of knocking out. Basically, in solo mode, higher DPG means higher KPG. |
| KPG(Kills per game) | KPG will be affected by the number of killings. Normally, DPG increases, KPG also increases. |
| HPG(Heals per game) | Players can be healed by patching medicate-kits, drinking energy drink and taking pain-killer. Players with higher tend to survival time longer in game. |

We will analyze these factors and combined the knowledge we learned in data-mining to predict players winning rates.

**Project Goal：**

Only 1 player could win the game among 100 players. In this project, we would like to analyse player’s current personal gameplay data (Movedistance, DPG, KPG, HPG), and depending on it, we can predict the player’s chance of becoming the last surviving player in the game.

**Project Domain:**

The dataset is already sufficiently complicated and complex to work.Thus, we will collect the each player's gameplay data in the solo mode and predict which feature would most affect the winning rate.

**Tools and Data sets**

* **Dataset**

The dataset we use contains player statistics for approximately 85,000 of the top ranked PUBG players. It provides players’ personal gamaplay data including the following parameters in 3 mode(solo, duo, squad): tracker\_id, time\_survived, wins, damage\_done, headshots, move distance, kills, suicides, etc.

Dataset Link: <https://www.kaggle.com/lazyjustin/pubgplayerstats/data>

* **Tools**

Weka

* **Classifier**
* Naive Bayes Classifier
* Logistic

Parameters will take into algorithm:

* Player’s information: player\_name, tracker\_id
* performance : solo\_KillDeathratio, solo\_Winratio, solo\_TimeSurvived, solo\_Roundsplayed
* Distance: solo\_WalkDistance, solo\_RideDistance, solo\_MoveDistance
* Support: solo\_Heals, solo\_Boosts, solo\_DamageDealt
* Player’s at different Rank: solo\_Rating, solo\_BestRating
* Player’s skill: solo\_KPG, solo\_Assists, solo\_Suicides, solo\_DPG

Overfitting or underfitting the data need to be avoided.

**Milestones:**

|  |  |  |
| --- | --- | --- |
| Timeline | Task | Progress |
| Oct 15 | Check similar projects on Kaggle and brain storm ideas | Completed |
| Oct 19 | Hand in Project Specification, clarify the topic and data sets | Completed |
| Oct 20 | Investigate the domain | Completed, we decide to chose one mode to test(solo mode) our data |
| Oct 26 | Collect data from mutiple webstire in order to ensure the accuracy, Refine dataset | Completed, we collect the data from kaggle and PUBG.Me. We found the database is huge and refine the domain |
| Oct 28 | Data transformation | Completed, Use CSVLoader and ArffSaver to transfer csv dataset into arff format.  using the following command: java -cp /Applications/weka/weka.jar weka.core.converters.CSVLoader PUBG\_Player\_Statistics.csv > PUBG.arff |
| Oct 31 | Prepare training, Test dataset | Completed, the original database has 85,000 players. We decided to extract 1000 players data as the training dataset, and the other 1000 players’ data will play as the test data |
| Nov 15 | Begin development of the algorithm | Completed, we imported the dataset and run it on Weka with different classifiers. We use Naive bayes as the classifier and test the data to compare the accuracy. |
| Nov 19 | Initial algorithm test | Completed, we ran the dataset in Weka with Naive Bayes. |
| Nov 21 | Tune algorithm, Final test | Completed, we got some result for Naive Bayes. |
| Nov 23 | Classifier performance evaluation | Evaluated the result of each classifier and create an output. |
| Nov 24 | Views | The result of each classifier are presented in the information viewer prior analysis |
| Nov 25 | Start making PPT with existing data and graphs  Prepare presentation | Conclude the result and prepare the presentation |
| Nov 29 | Final presentation | Completed |
| Dec 4 | Final report | Completed |

**Problem we face**

1. Two of our team members removed from our team, more works for other team members.
2. We found that it is difficult to schedule group meetings outside class. Everyone has different time schedules.
3. We originally thought that we could take all parameters into the algorithm, but we shortly realized it will make our algorithm extremely complicated.
4. We start by analysing all three modes, but the dataset is so large and complex. We finally decide to analyse solo mode instead of all three modes because players can only rely on their own skills on this mode.
5. The raw data has too many attributes and values, so it is difficult to filter out the conditional attributes and find the break points. (i.e. we delete the avg\_walkdistance because there is too many noisy data and has less relationship to the win ratio than we thought)

**Data Preprocessing**

Data preprocessing is an important step in the data-mining process. It aims to filter out the representation and high-quality data because using data that has not been carefully screened for such problem can produce misleading results. Thus, we need to clean the original data set and get the data as the training set. Here are the steps:

1. **Analyzing raw data:** To identify the original data set and analyze the attributes of the set.
2. **Data reduction:** To reduce the number of attributes and remain the information that related to the Win-ratio prediction.
3. **Splitting data:** To choose the split that minimizes information and calculate information in the possible splits.
4. **Training and Testing data generation:** To generate the files that required for the data mining process.

The details of each step are explained below:

**3.1 Analyzing Raw data:**

The original dataset comes from the player statistics for approximately 85000 of the top PUBG players that provided by the Kaggle websites. The data set is an ARFF file, which means we should transfer the file into CSV later. The original data set has 85000 player’s information (columns) and 125 attributes, which related to each player modes (solo, duo, and squad).

**3.2 Data reduction:**

In the raw data set, all statistics were gathered using aggregate region filters, and features labels are divided by server types: solo, duo, and squad. Solo is the only mode that players don’t have teammates and only relate to players own performance and skills (win-ratio depends more on their statics in each solo-attributes). Thus, we choose the solo mode as the region to predict our win ratio, to increase the accuracy.

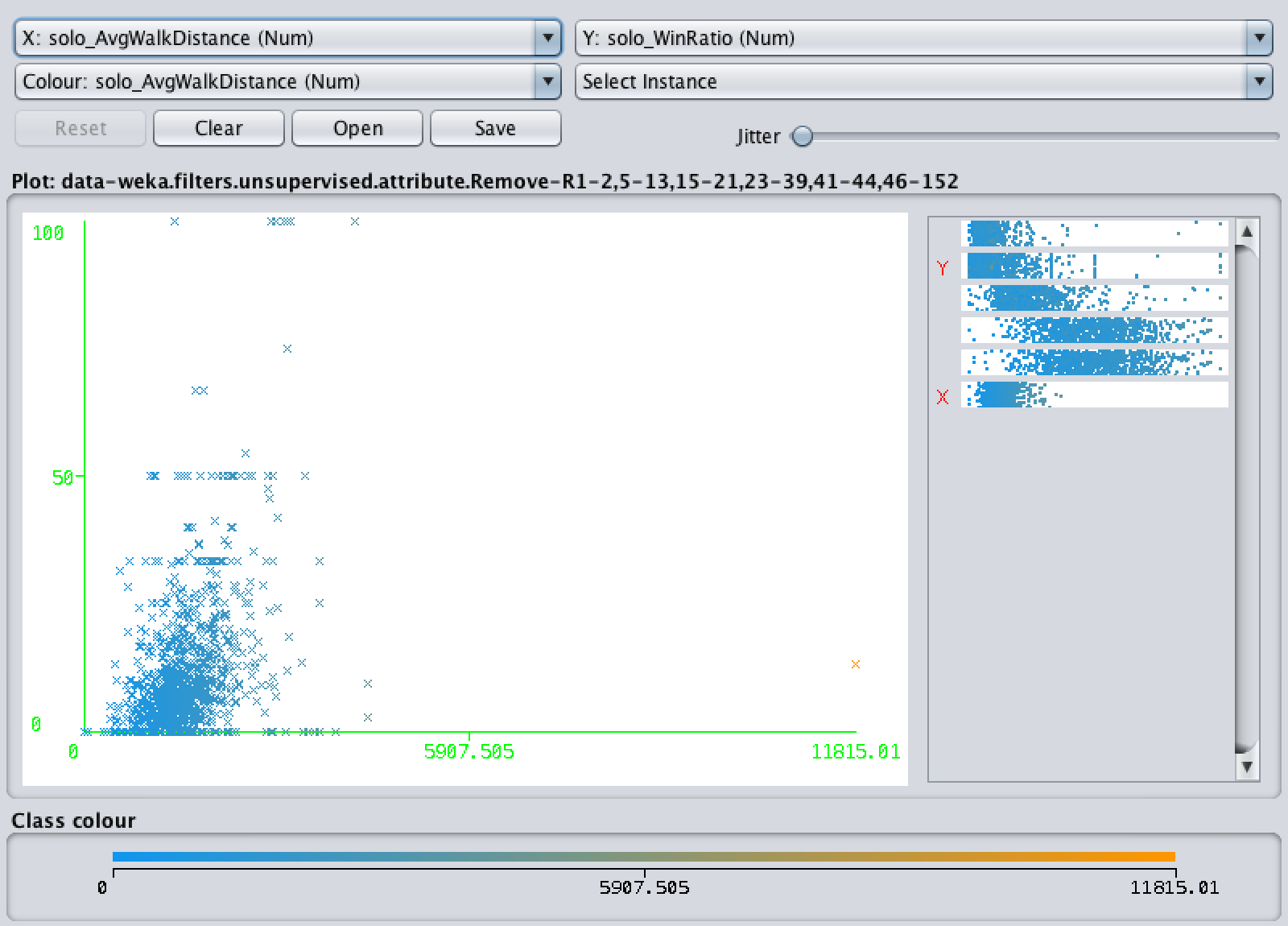
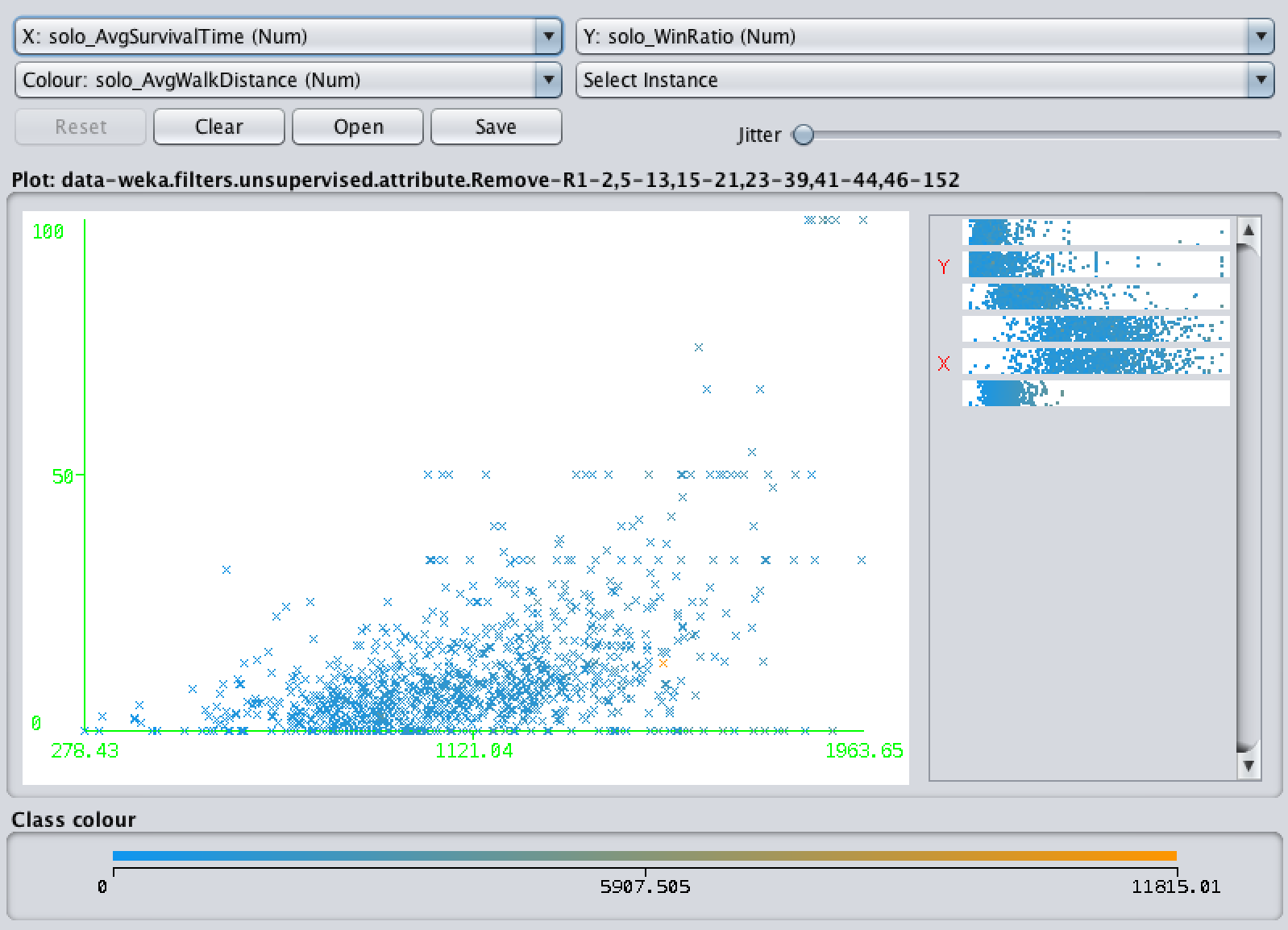
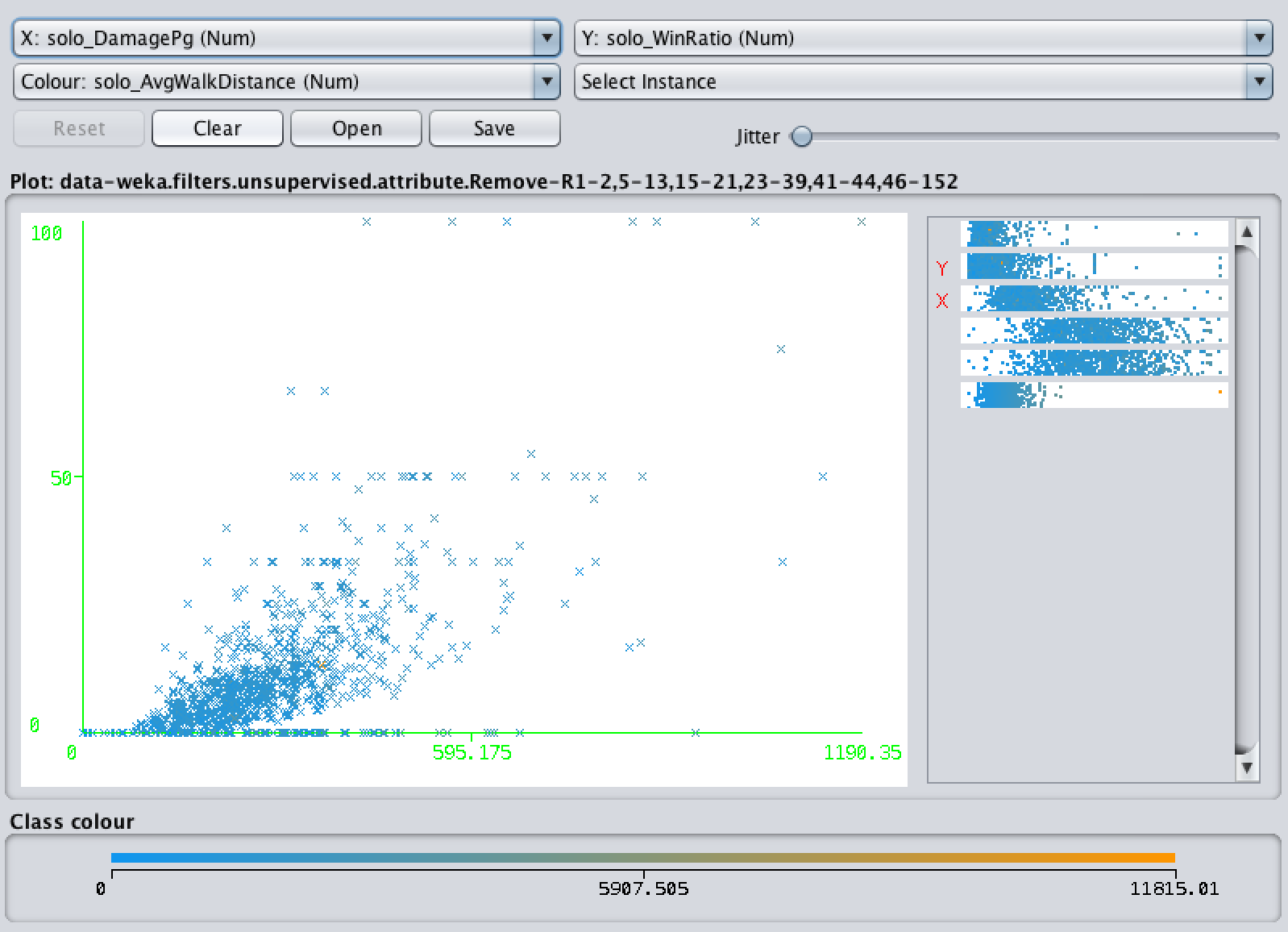
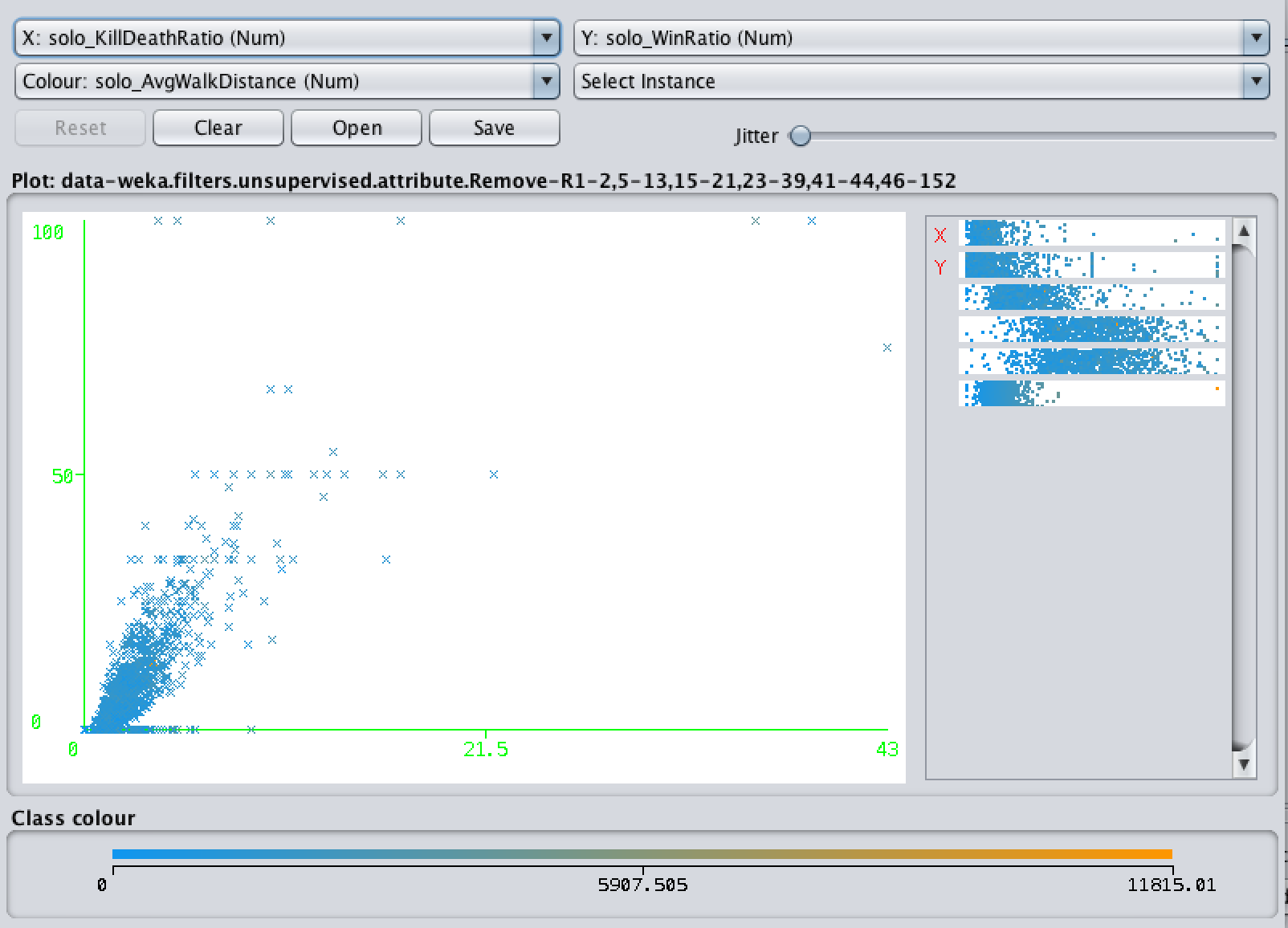
After removing attributes not related to the solo mode, we still have 35 of them. To reduce the attributes one step closer, we import the ARFF file into the Weka and use the visualize function to view the relationship between each attribute and solo\_winratio. As we see in the graph below, we can remove the attributes, which are too scattered (i.e. bar chart) and remain the value that can mostly represent the performance. Then, we choose four attributes: solo\_killdeathratio, solo\_damagepg, solo\_timesurvived, and solo\_Avgwalkdistance.

The solo-Avgwalkdistance is the most scattered one, so I view the value of each row in this attributes and found it contains so many noisy data which makes this attributes contribute less relation to the win ratio prediction. Our team decides to delete this attribute.

We also reduce the number of attributes by grouping similar attributes into intervals.(i.e. solo\_Avgtimesurvived and solo\_timesurvived; solo\_damagepg and solo\_Avgdamagepg)

After all, we have three attributes: solo\_killdeathratio, solo\_damagepg, and solo\_timesurvived

Here to view their relation graph to the win ratio:

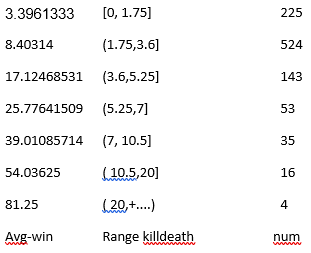


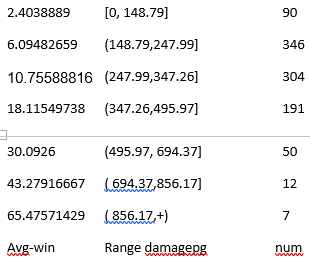
*(from left to right, the order is solo\_killdeathratio, solo\_damagepg, solo\_Avgwalkdistance and solo\_timesurvived )* *Weka Graph 1.*

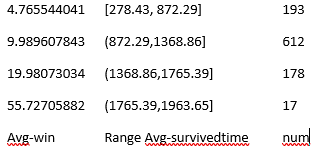
**3.3 Spliting date:**

To prepare the data for Naive-Bayes, we need to regroup the data based on the values of the three chosen attributes. To apply the supervised discretization method, we sort the data from small to big in each attribute. Then, I place breakpoints between values based on the Weka’s graph of each attribute (***see in weka graph 2***). There are too many intervals for each attribute, so I merge intervals with equal or similar class distributions.

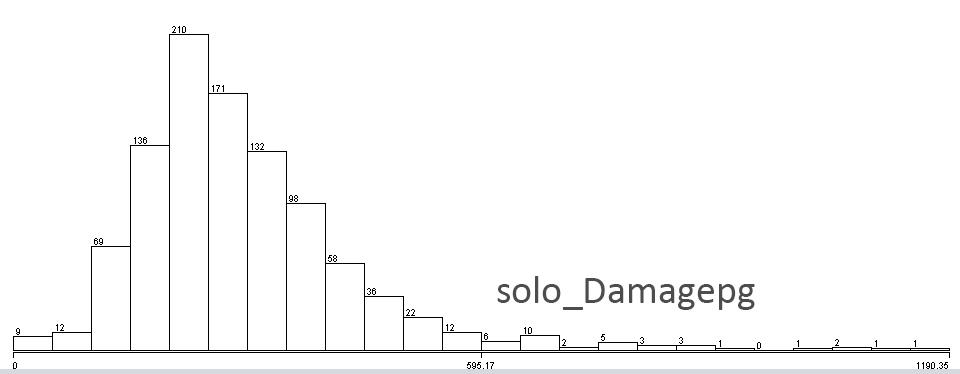
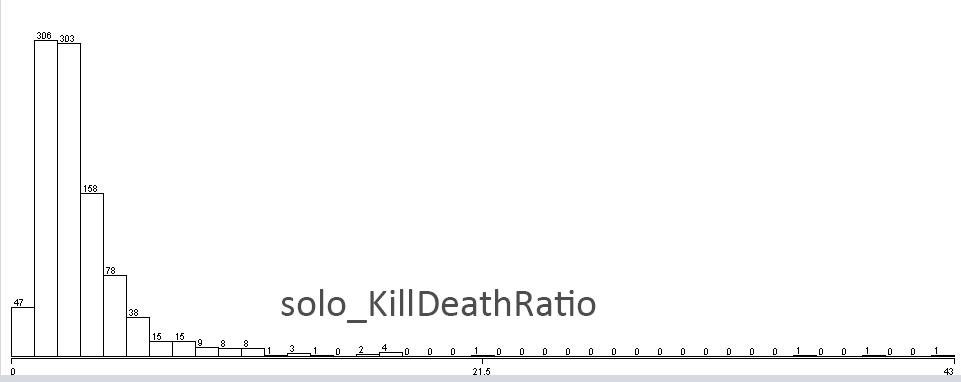
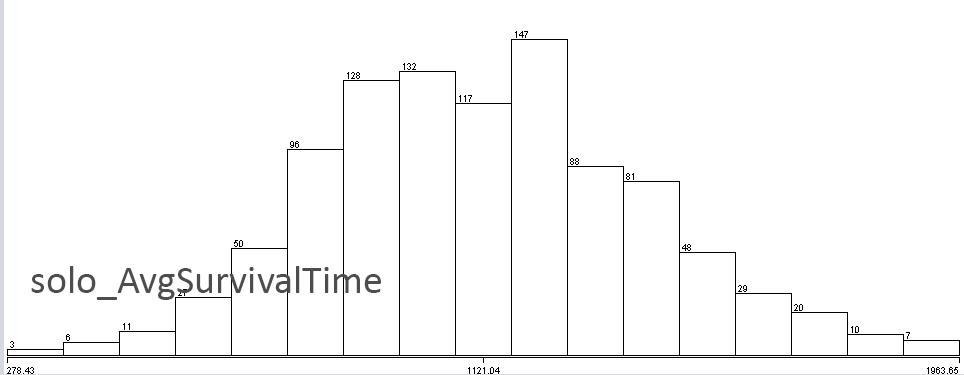
Here is the result:

solo\_killdeath ratio:

solo\_damagepg 

solo\_survivedtime 

*Figure 2.*



*(show the relationship between each attributes and win ratio) Weka Graph 2.*

**3.4 Tranning and Testing data generation:**

Finally, we filtered out our training data and corresponded testing data according to the analysis from Weka. I will generate and display the conditional attributes in Excel table.

|  |  |  |
| --- | --- | --- |
| Attribute | Description | Trying to measure |
| solo-killdeathratio | the average rate of the amounts of deaths  you need per kill | Experience |
| solo-damagepg | the damage each player produce per game | Experience |
| solo-timesurvived | the average time players surviving per game | Experience |
| solo-winratio | the average rate of the possibility of winning the game | Experience/information delivery |

**Data Mining**

As the dataset has been reduced to 4 attributes, the predictor trains 2000 instances, rearranges the attributes into ranges, takes the ranges of win ratio as classes, compute the probabilities for each win ratio range, for each player predict the win ratio by choose the highest probability of win ratio range, and calculate the accuracy of the prediction. The project uses Naive Bayes as the classifier.

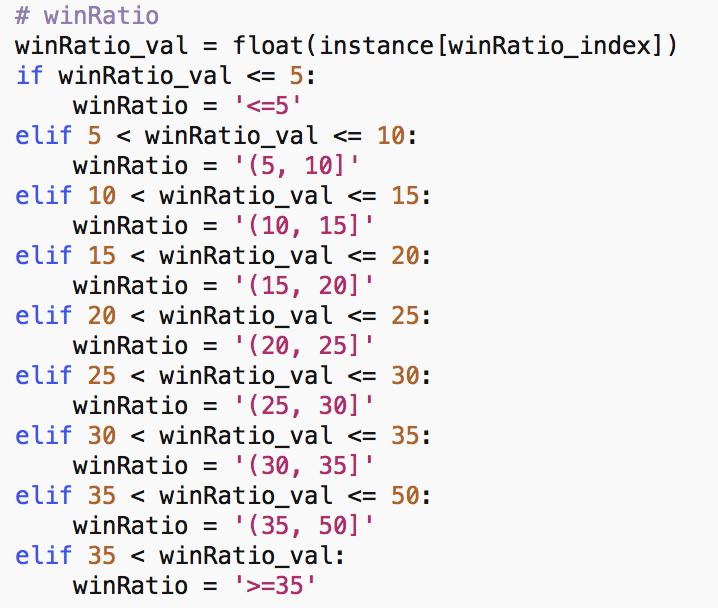


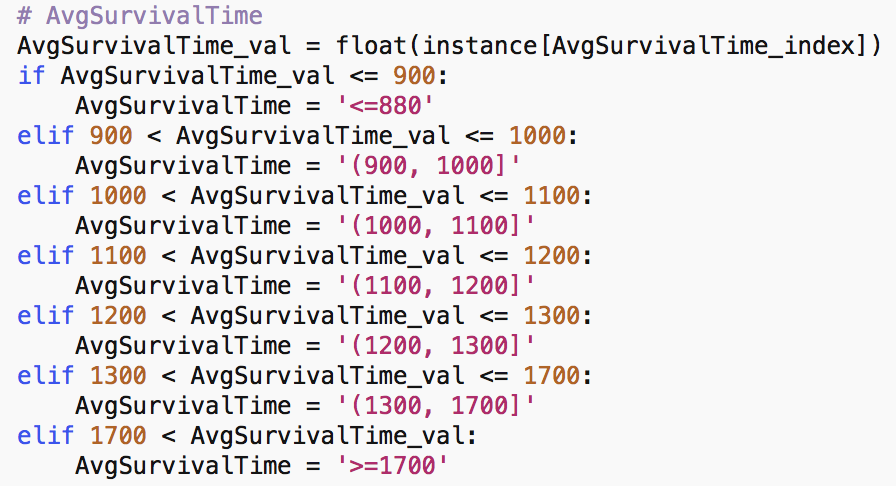
*Figure 3. Attributes*

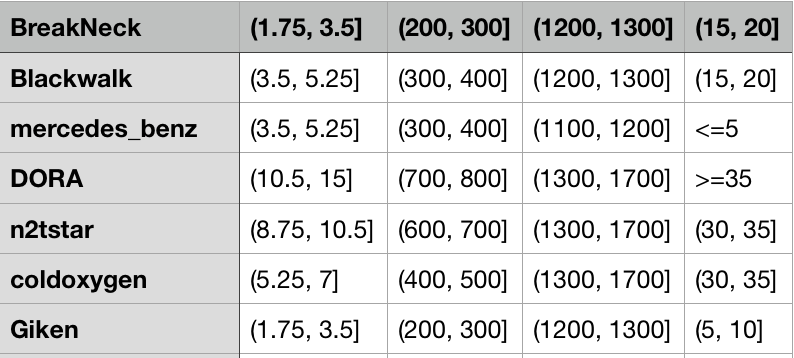
At the beginning, we included average walk distance in the attributes. After building the predictor and some experiments, we found that average walk distance does not actually affect the win ratio. As a result, we deleted average walk distance as an attribute, and it lead an 1% increase in accuracy.

*Figure 4.1 Rearranging killDeathRatio*

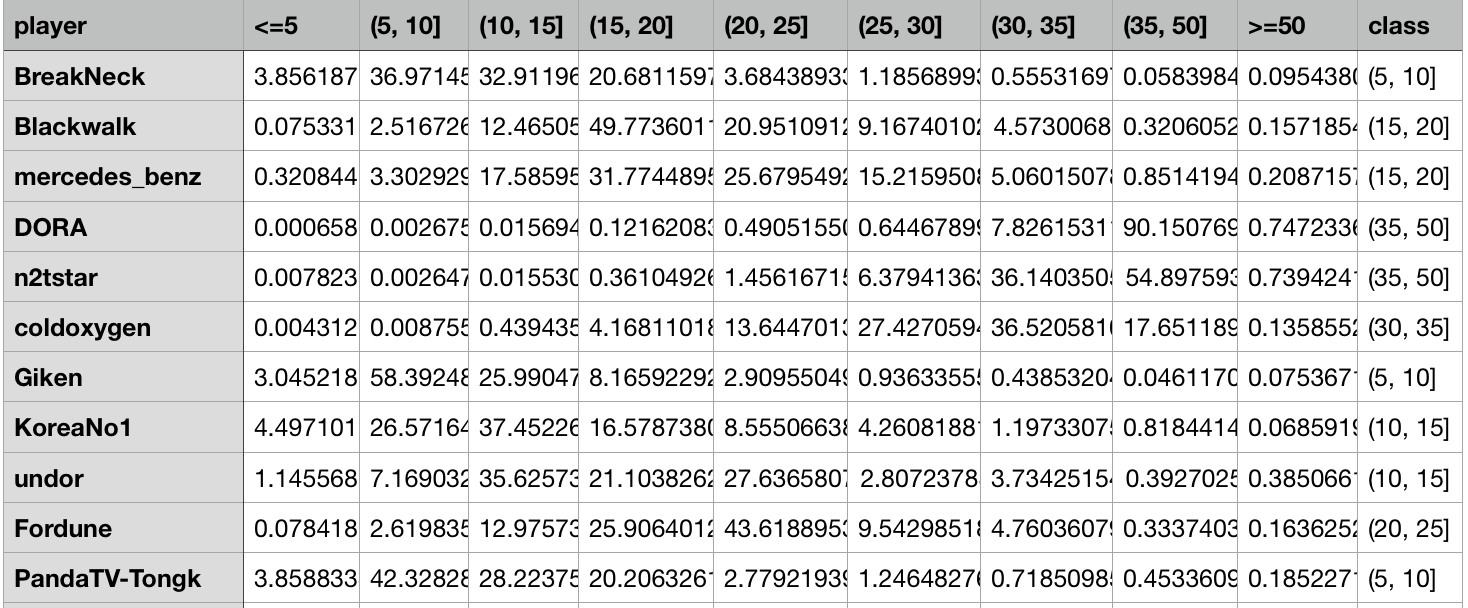
*Figure 4.2 Rearranging Data damagePg*



*Figure 4.3 Rearranging winRatio Figure 4.4 Rearranging AvgSurvivalTime*

*Figure 5. Re-formed Data (Planyer name, killDeathRatio, damagePg, AvgSurvivalTime, Win ratio)*

After rearranging the dataset, we take the ranges of attributes into probability calculation using the Bayes‘ rule, and output an csv file to hold the probability data.

*Figure 6. Result Data File*

Each column except the last one shows the probability of possible win ratio range. The last column “class” shown in Figure 6 is the prediction of win ratio for the specific player. Then, we compare the prediction to the original win ratio, and find the accuracy. The accuracy is 65.57%.

**Conclusion**

In conclusion, our group put effort on predicting the possibility of winning in PUBG’s solo match mode using the dataset we found online. We found our dataset on Kaggle, and we use Weka as a tool for data analysis. The original dataset contains approximately 85000 player’s data and 35 attributes for solo match mode, we finally decided to take 4 attributes - which has the best performance- and 2000 instances into our Naive Bayes’ algorithm. The result we got by running our algorithm is close to result we were expected. There were different challenges along the way: One of our team member decided to drop the course after the midterm, another team member never participate in any group work. Fortunately, the rest of our team cooperate very well. Overall, this is really a fun project to work with, and we deeply understood every concept we talked in class through the process of finishing this project.