Halil İbrahim Ceylan - 390709073
halilibrahimceylan2@posta.mu.edu.tr
Fatih Emre YILDIZ - 180702004
fatihemreyildiz@posta.mu.edu.tr

Final Project
CENG 3521: Data Mining

22-01-2021

Project's Topic

Electronic Sports - League of Legends

Project's Goal

Try to guess the winner team

Project's Description

We are going to use almost every data in the game. We don't know which parameter will be really effective to guess the end of the game. Our most known parameters are The gold which teams earn during the game, towers, inhibitors, wards, monsters killed for both teams, wards replacement, firstblood firstower and so on. We will explain the parameters better on our report. Our dataset is this: https://www.kaggle.com/gyejr95/league-of-legends-challenger-ranked-games2020 There are almost 200.000 games. We do not know about which loss function we are going to use. We will decide it later on while we are doing the project.

Abstraction

Electronic sports (eSports) is a type of sport competition which uses video games. It's popularity has increased in recent years, has gradually become a part of modern sports. Electronic sports, which has achieved progress since the early 2000s, It has great potential to show a similar development as modern sports in a shorter time.

Recently, one of the popular games in the field of eSports has become Multiplayer Online Battle Arena (MOBA). MOBA type games currently occupy a large place in the eSports industry. The most popular game of the genre is "League of Legends" debuted on 2009. League of Legends, which currently has 30 million players worldwide and is claimed to be the most played game, It has hundreds of thousands of players and millions of viewers.

In this study, with the data of the one of the popular game of today League of Legends, we have we aimed to predict which team will win.

1 Introduction

1.1 Description Of The Game

League of Legends is a Multiplayer Online Battle Arena game. Which is shorten as MOBA.

It is a team-based strategy game. There are five players for each team. To win the game, teams have to destroy the other's base which is called Nexus. To win the game players have to take down enemies, get the vision, earn gold, get stronger, and take down towers as they battle their way to victory.

On the playground, there are many parameters as, towers, minions, inhibitors, jungle camps, wards etc.

From each team's Nexus, Minions will spawn till the end of the game and go through the three lanes. The three lanes are known as top lane, mid lane, and bot lane. In all three lanes, there are turrets and they will attack enemy champions and minions. Each lane has three turrets and one inhibitor, Each Inhibitor is protected by a Turret. When an Inhibitor destroyed, super minions will spawn in that lane for several minutes, then it will re-spawn and Super Minions will stop spawning.

In between the lanes, there is the jungle. There are neutral monsters in the jungle. There are two important monsters in the jungle these are Dragon and Baron. which gives advantages to the teams when destroyed.

Each of these objects returns players as experience and gold. With experience players get higher levels, with gold player gets more stronger, can buy wards and get vision over the map. With more vision, teams can control the map and the enemy team.

Thus, almost every data in our data-set can affect the end of the game.

To inspect the data and try to predict the end of the game we have selected to use Data Preprocessing, Linear Classification, Neural Network-based Classification and Clustering.

1.2 Dataset

We got our data on the website called kaggle. We have 50 columns. They were.

gameId: Unique ID to identify the game.
gameDuraton: Game Duration in seconds.
blueWins-redWins: Winner Team
blueFirstBlood-redFirstBlood: Team who has the first score.
blueFirstTower-redFirstTower: Team who has destroyed the first tower.
blueFirstBaron-redFirstBaron: Team who has got the first Baron.
blueFirstDragon-redFirstDragon: Team who has got the first Dragon.
blueFirstInhibitor-redFirstInhibitor: Team who has got the first Inhibitor.
blueDragonKills-redDragonKills: Number of the Dragons killed by teams.
blueBaronKills-redBaronKills: Number of the Barons killed by teams.
blueTowerKills-redTowerKills: Number of the Towers taken by teams.
blueInhibitorKills-redInhibitorKills: Number of the Inhibitors taken by teams.
blueWardPlaced-redWardPlaced: Number of the Wards placed by teams.
blueWardkills-redWardkills: Number of the Wards killed by teams.
blueKills-redKills: Number of the Scores taken by teams.
blueDeath-redDeath: Number of the Scores given by teams.
blueAssist-redAssist: Number of the Assistance taken by teams.
blueChampionDamageDealt-redChampionDamageDealt: Damage Dealt by champions to enemy team.
blueTotalGold-redTotalGold: Number of the Gold gained by teams.
blueTotalMinionKills-redTotalMinionKills: Number of the Minions killed by teams.
blueTotalLevel-redTotalLevel: Total Level of the teams.
blueAvgLevel-redAvgLevel: Average Level of the teams.
blueJungleMinionKills-redJungleMinionKills: Number of the Jungle Minion killed by teams.
blueKillingSpree-redKillingSpree: Number of total kills for a champion without getting died.
blueTotalHeal-redTotalHeal: Number of heal gained for the teams.
blueObjectDamageDealt-redObjectDamageDealt: Damage Dealt by champions to objects.

2 Data Preprocessing

In this section, we have looked at our data, Since we have huge amount of data we did some pre-processing to make our data more suitable for the tasks. We have applied Sampling and Feature Subset Selection.

2.1 Sampling

Sampling is the selecting the certain data to analyze.

In a game, If a player disconnects from the game and His/Her team doesn't want to play the game anymore, they can start voting to surrender. If the majority of the team doesn't want to play the game and votes for it. They can give up and lose the game by voting. This affects our data in a bad way. In our dataset, some games are ended by voting. This made noise in our dataset. To avoid that we have deleted games that ended before 20 minutes. It gave us a more suitable dataset to work on.

2.2 Feature Subset Selection

Feature Subset Selection is a way to reduce the dimension of the data by deleting the columns that we have. We have applied it to the Redundant Features and Irrelevant Features.

2.2.1 Redundant Features

Redundant Features are the duplicated data in the dataset.

Since we have data for both teams some of our columns are related to each other. These columns represent the binary data for the first happened in the game. For example, the column 'redFirstBlood' is 1, It means that the Red team got the first score in the game, then 'blueFirstBlood' can not be other than 0. There are few conditions like these, these are 'redWins', 'redFirstInhibitor', 'redFirstBaron', 'redFirstDragon', 'redFirstTower'.

2 columns are related to other columns. These are 'redAvgLevel', 'blueAvgLevel' represents the Average level of the teams. Since we have other columns that are multiplied by the team's Average Level with the number of player '5' for both teams 'redTotalLevel', 'blueTotalLevel'. We have deleted the columns related to the average levels of the teams.

2.2.2 Irrelevant Features

Irrelevant Features doesn't contain any information for our tasks.

In this case there is only one column that doesn't affect the end of the game which is 'gameId'. 'gameId' is the unique data to identify the game.

2.3 Standardization

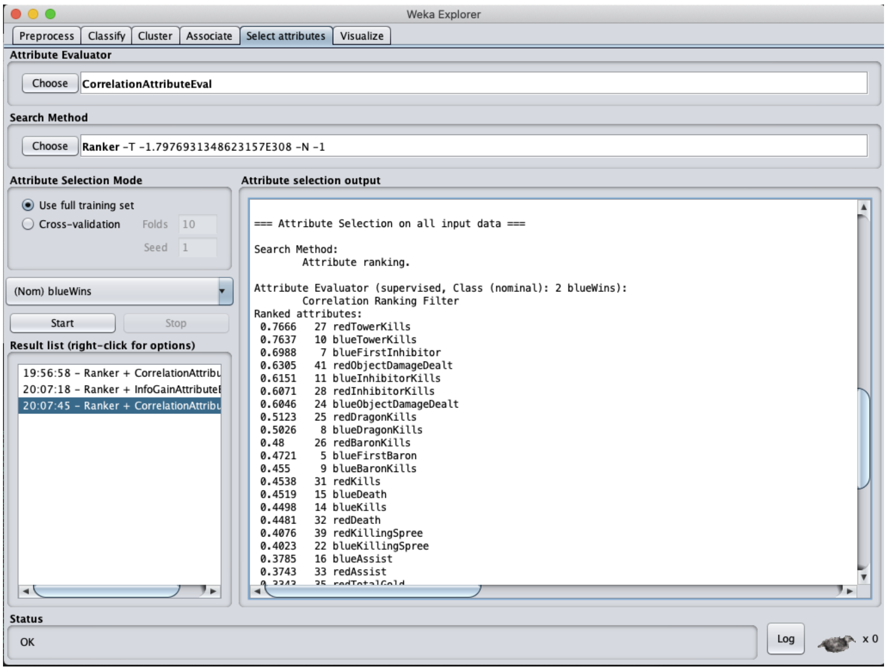
Also we applied Standardization to our dataset. We used MinMaxScaler class from scikit learn. It transforms features by scaling each feature to a given range. We have scaled all the features between zero and one.

2.4 Weka Analysis

In this section, after get reviews from supervisor we decide to analyse our data with weka tool. In this tool there are many useful data mining subtool. We put the arff file into dataset folder, so you can get same analyse from weka with opening this file. (dataset/dataset_for_weka.arff)

2.4.1 Feature Importance

In weka tool, we used to Select Attributes, CorrelationAttributeEval method for get the feature importances. So there are the results.



Attribute Evaluator: CorrelationAttributeEval

Search Method: Ranker -T -1.7976931348623157E308 -N -1

Attribute Selection Mode:
● Use full training set
○ Cross-validation Folds 10 Seed 1

(Nom) blueWins

Result list (right-click for options)
19:56:58 - Ranker + CorrelationAttribu
20:07:18 - Ranker + InfoGainAttributeE
20:07:45 - Ranker + CorrelationAttribu

Attribute selection output

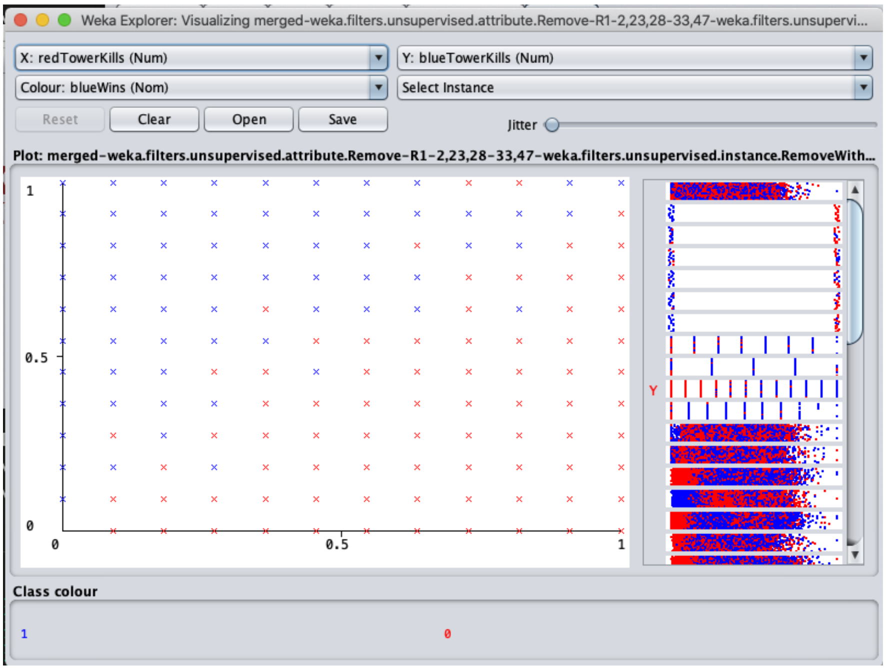
```
=== Attribute Selection on all input data ===

Search Method:
        Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 2 blueWins):
        Correlation Ranking Filter
Ranked attributes:
 0.7666   27 redTowerKills
 0.7637   10 blueTowerKills
 0.6988    7 blueFirstInhibitor
 0.6305   41 redObjectDamageDealt
 0.6151   11 blueInhibitorKills
 0.6071   28 redInhibitorKills
 0.6046   24 blueObjectDamageDealt
 0.5123   25 redDragonKills
 0.5026    8 blueDragonKills
 0.48     26 redBaronKills
 0.4721    5 blueFirstBaron
 0.455     9 blueBaronKills
 0.4538   31 redKills
 0.4519   15 blueDeath
 0.4498   14 blueKills
 0.4481   32 redDeath
 0.4076   39 redKillingSpree
 0.4023   22 blueKillingSpree
 0.3785   16 blueAssist
 0.3743   33 redAssist
 0.3343   35 redTotalGold
```

Status: OK

2.4.2 Two most importance feature graph



Weka Explorer: Visualizing merged-weka.filters.unsupervised.attribute.Remove-R1-2,23,28-33,47-weka.filters.unsupervi...

X: redTowerKills (Num)

Y: blueTowerKills (Num)

Colour: blueWins (Nom)

Select Instance

Plot: merged-weka.filters.unsupervised.attribute.Remove-R1-2,23,28-33,47-weka.filters.unsupervised.instance.RemoveWith...

2.5 Result

Before Data preprocessing section we had 50 columns of dataset. We have examined every column. After we have applied the data preprocessing techniques, we reduced features to 40 without losing any information.

3 Linear Classification

In this section, we used Linear Classification to our dataset. Linear Classification helps us to classify the data by true/false or 1 and 0. as blueWins and redWins.

3.1 Perceptron

In data mining, Perceptron is the sensor for an algorithm to the supervised learning of binary classifiers.

A binary classifier is a function that can decide whether an input represented by a vector of numbers belongs to a particular class.

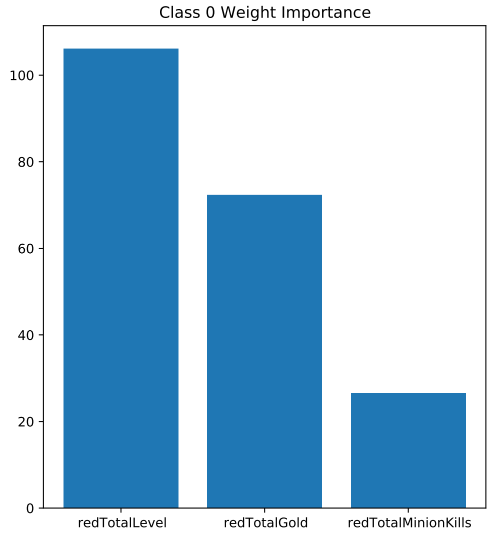
3.1.1 Performance Analyse

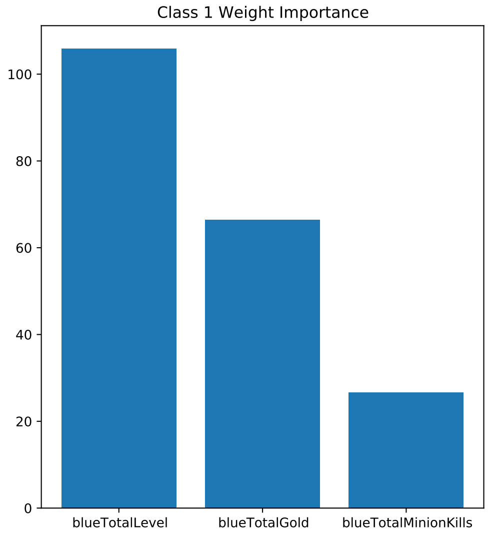
We run like 5 times perceptron model with default parameters for get average values you can inspect below table for result.

Linear Classification: Perceptron Results			
#	Training Time (ms)	Error (cost)	Score (%)
1	341.68	0.018723	0.9813
2	326.62	0.018723	0.9813
3	336.92	0.018723	0.9813
4	332.29	0.018723	0.9813
5	323.27	0.018723	0.9813
Average	332.16	0.018723	0.9813

3.1.2 Feature Weight Analyse

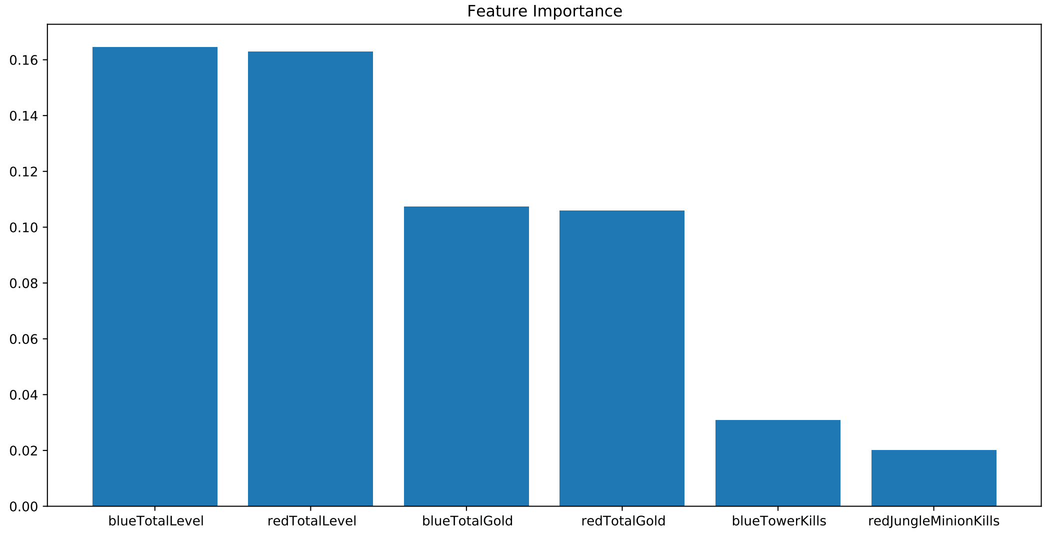
In this section, we inspect Perceptron instancee with coef_ attribute (thanks to sckit-learn). with coef values we can get weights from first layer, so we applied this values with column features. there are some negative and positive values. Positive ones are most effective for class 1 (means blue win). Negative ones are most effective for class 0 (means red wins). According to this model, 3 Significant factor for class 0 are RedTotalLevel, redTotalGold, redTotalMinionKill. for class 1, BlueTotalLevel, blueTotalGold, blueTotalMinionKill. When we search on internet for these parameters, there are really effective for winnings. So each proffessional team is traning their players for improvements these features. You can inspect graph before.





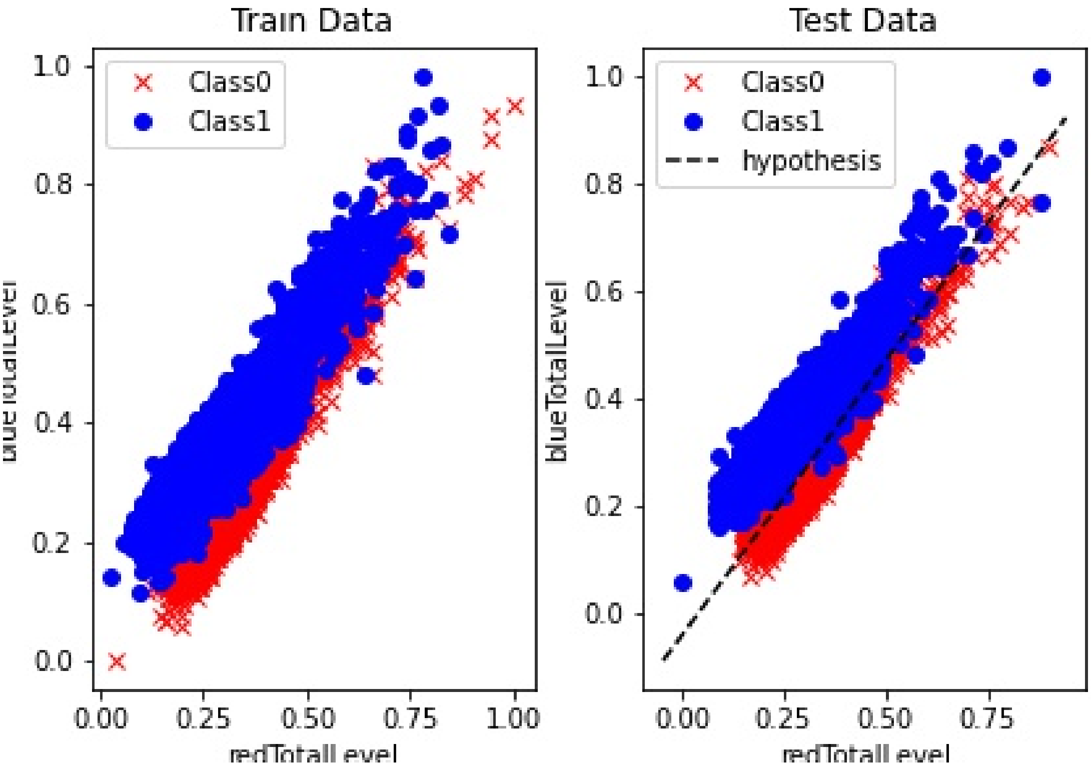
3.1.3 Feature Analyse with BruteForce

In this section, we analyze all values with randomly suffling, and these values are calculated to model-feature relation. when we suffle a feature, we look the model score, when its drop dramaticly. we decide to this feautre is important than others. These values very close to Feature Weight Analyse.



3.1.4 Decision Boundary Graph

We draw graph to with two importance features (according to model) and we draw line from first layer. You can inspect result below.



3.2 Logistic Regression

Logistic regression is a statistical method used to analyze a data set with one or more independent variables that determine a result. The result is measured by a binary variable. In logistic regression, the dependent variable contains data coded as binary or binary, i.e. only 1 or 0. The purpose of logistic regression is to find the most suitable model to describe the relationship between a two-way characteristic and a set of independent variables. Logistic regression generates the coefficients of a formula to estimate the probability of the existence of the characteristics of interest for the transformation.

3.2.1 Performance Analyse

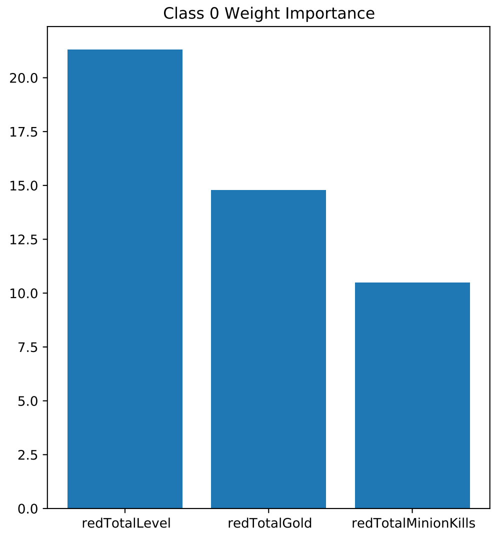
We run like 5 times for get average values you can inspect below table for result.

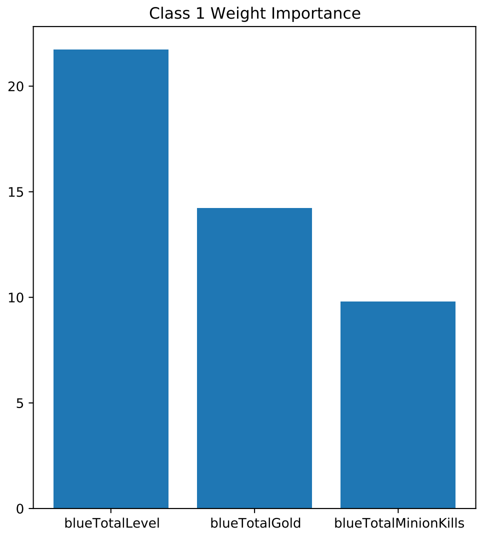
Linear Classification : LogisticRegression Results

#	Training Time (ms)	Error (cost)	Score (%)
1	2048.80	0.014602	0.9854
2	2056.79	0.014602	0.9854
3	1871.26	0.014602	0.9854
4	2162.40	0.014602	0.9854
5	2640.21	0.014602	0.9854
Average	2155.89	0.014602	0.9854

3.2.2 Feature Weight Analyse

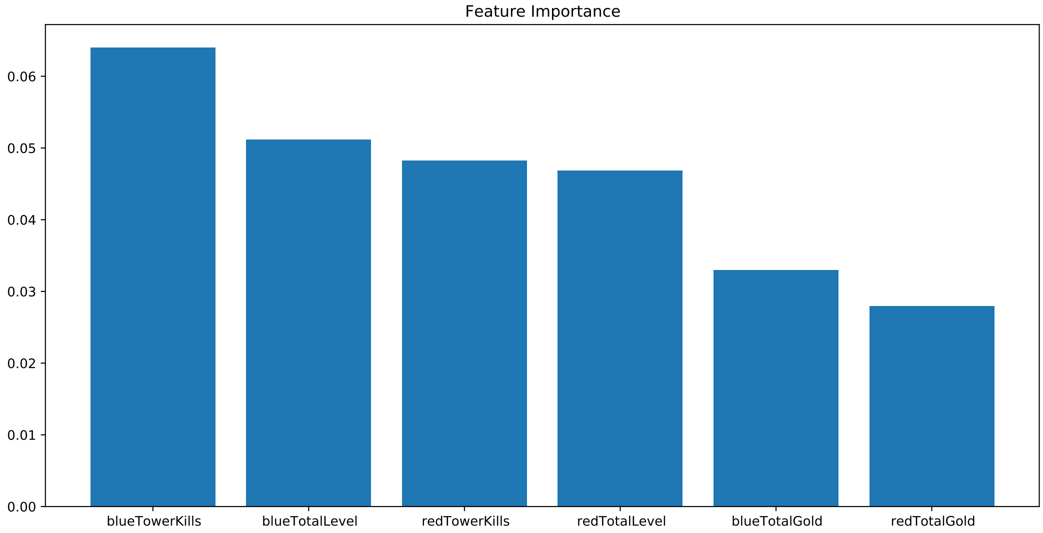
In this section, we applied same technique which we used before. According to this model, 3 Significant factor for class 0 are RedTotalLevel, redTotalGold, redTotalMinionKill. for class 1, BlueTotalLevel, blueTotalGold, blueTotalMinionKill. Which like we mentioned before, These values are very important. You can inspect Perceptron Model Section and below graphs.


Class 0 Weight Importance


Class 1 Weight Importance

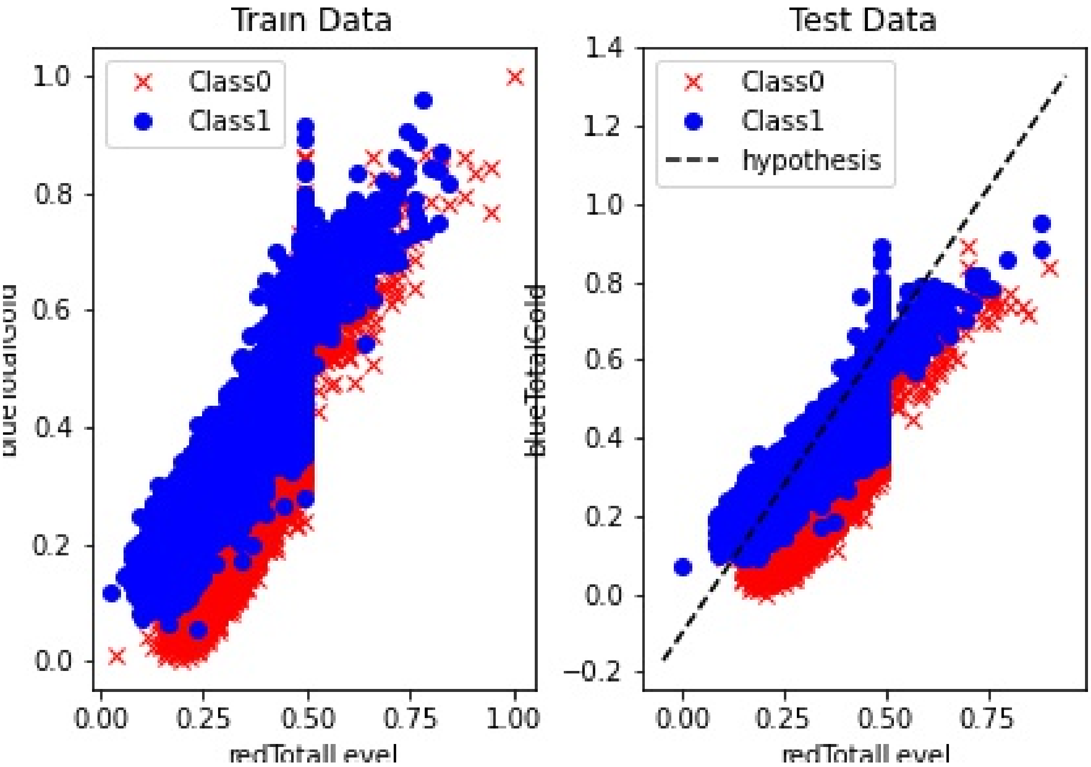
3.2.3 Feature Analyse with BruteForce

In this section, we analyze all values with randomly suffling, and these values are calculated to model-feature relation. when we suffle a feature, we look the model score, when its drop dramaticly. we decide to this feautre is important than others. These values very close to Feature Weight Analyse except that we realized here two new paramters which do not mentioned above, blueTowerKill and redTowerkill these values are very familiar from weka Feature Analyse.



3.2.4 Decision Boundary Graph

We draw graph to with two importance features (according to model) and we draw line from first layer. You can inspect result below.



3.3 SGD Classifier

Stochastic Gradient Descent shorten as SGD is very effective function for the learning of linear classifiers under loss functions such as Support Vector Machines and Logistic Regression. While SGD has been around for a long time in the data mining space, it has recently became popular in the of large-scale learning.

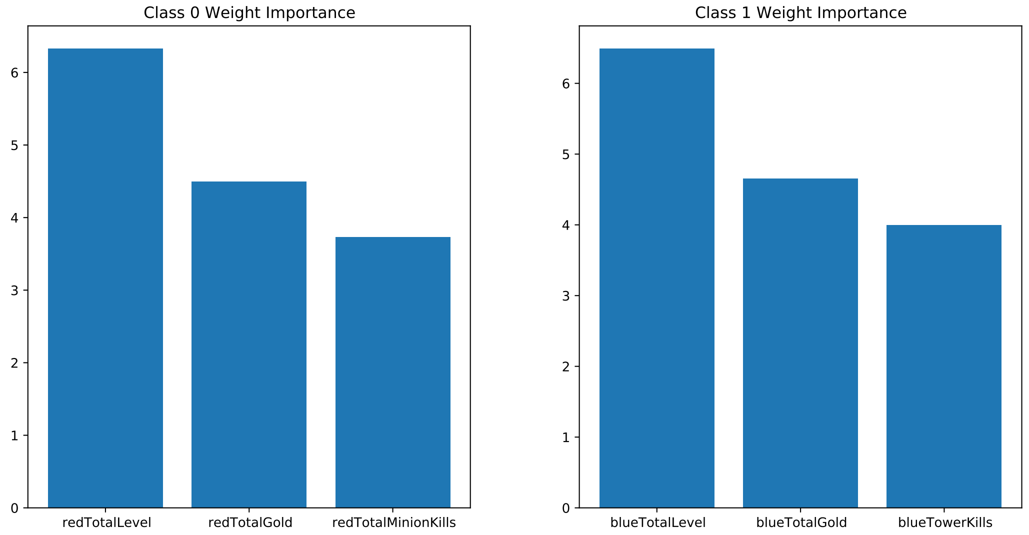
3.3.1 Performance Analyse

We run like 5 times for get average values you can inspect below table for result.

Linear Classification : SGDClassifier Results			
#	Training Time (ms)	Error (cost)	Score (%)
1	211.66	0.017651	0.9823
2	203.04	0.017342	0.9827
3	215.69	0.017365	0.9826
4	218.15	0.018128	0.9819
5	202.84	0.017532	0.9825
Average	210.27	0.017604	0.9824

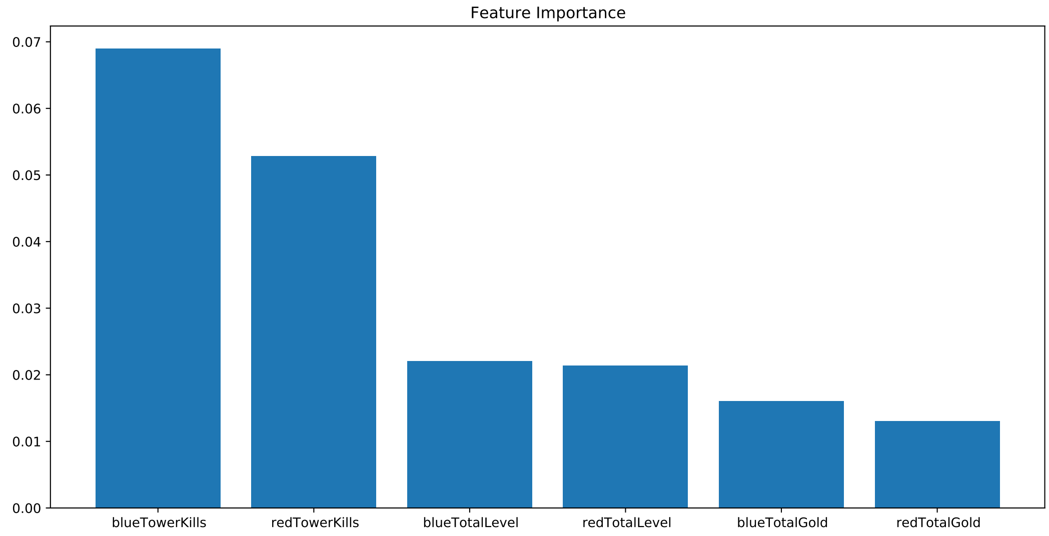
3.3.2 Feature Weight Analyse

In this section, we applied same technique which we used before. According to this model, 3 Significant factor for class 0 are RedTotalLevel, redTotalGold, redTotalMinionKill. for class 1, BlueTotalLevel, blueTotalGold, blueTotalMinionKill. Which like we mentioned before, These values are very important. You can inspect Perceptron Model Section and below graphs.



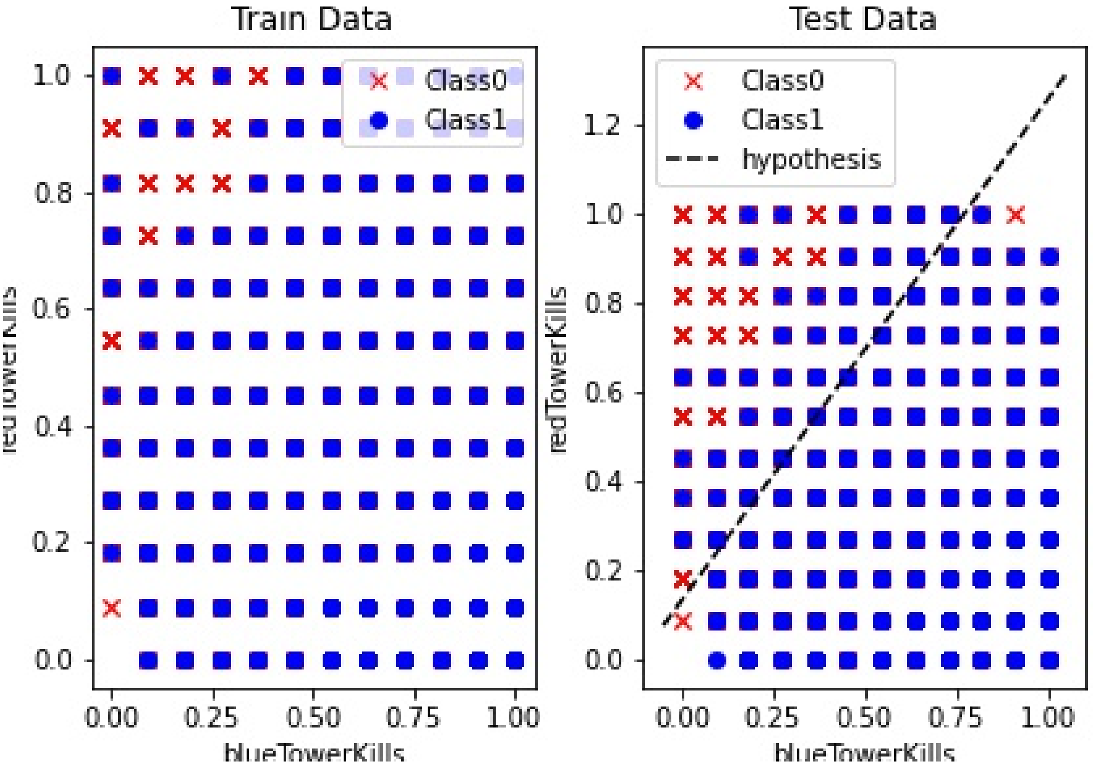
3.3.3 Feature Analyse with BruteForce

In this section, we analyze all values with randomly suffling, and these values are calculated to model-feature relation. when we suffle a feature, we look the model score, when its drop dramaticly. we decide to this feautre is important than others. These values very close to Feature Weight Analyse except that we realized here two new paramters which do not mentioned above, blueTowerKill and redTowerkill these values are very familiar from weka Feature Analyse. (same like in LogisticRegression)



3.3.4 Decision Boundary Graph

We draw graph to with two importance features (according to model) and we draw line from first layer. You can inspect result below.



4 Neural Network-based Classification

4.1 MLPClassifier

There are many machine learning models available today. Neural Network is a machine learning model. It is a model inspired by the human brain and nervous system. We can also consider it as a Multi-layer Perceptron classifier. Neural Network is a structure in the form of layers. The first layer is called input and the last layer is called exit. The layers in the middle are called ”Hidden Layers”. Its layer contains a certain deemed ’Neuron’. These neurons are connected by ”Synapse”. Synapses contain a coefficient. These coefficients tell how important the information in the neuron to which it is attached is. Synapse Coefficients The value of a neuron is found by multiplying the inputs to that neuron by the coefficients and adding them together. This result found is put into an activation. According to the result from the function, it is decided that that neuron will not fire.

4.1.1 Performance Analyse

We run like 5 times for get average values you can inspect below table for result. Compared to other models, it took longer than before. Also score’s are higher than before.

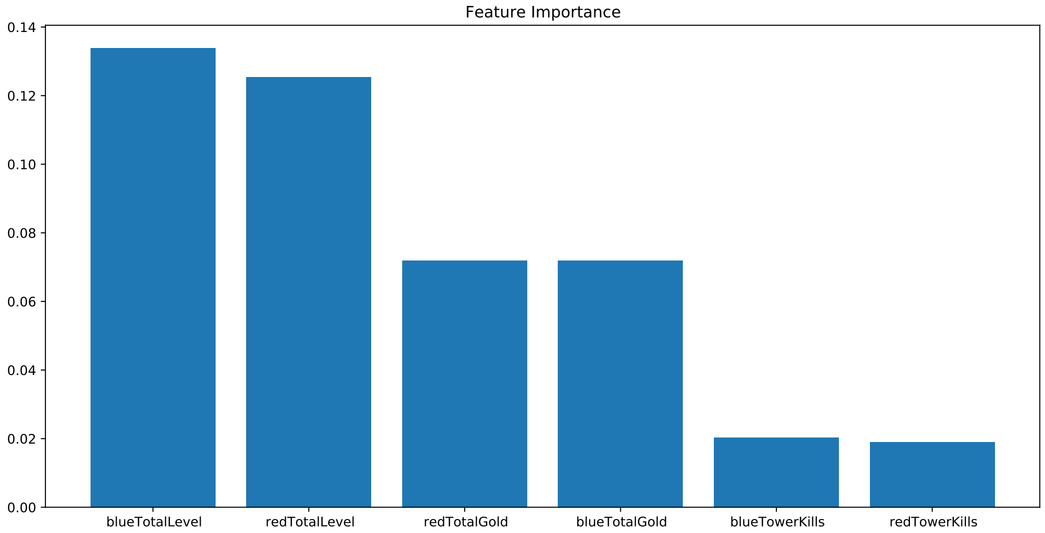
Neural Network-based Classification : MLPClassifier Results			
#	Training Time (ms)	Error (cost)	Score (%)
1	78774.47	0.008028	0.9920
2	50935.68	0.008814	0.9912
3	100431.51	0.008385	0.9916
4	62506.91	0.009552	0.9904
5	93354.18	0.008028	0.9920
Average	77200.55	0.008561	0.9914

4.1.2 Feature Weight Analyse

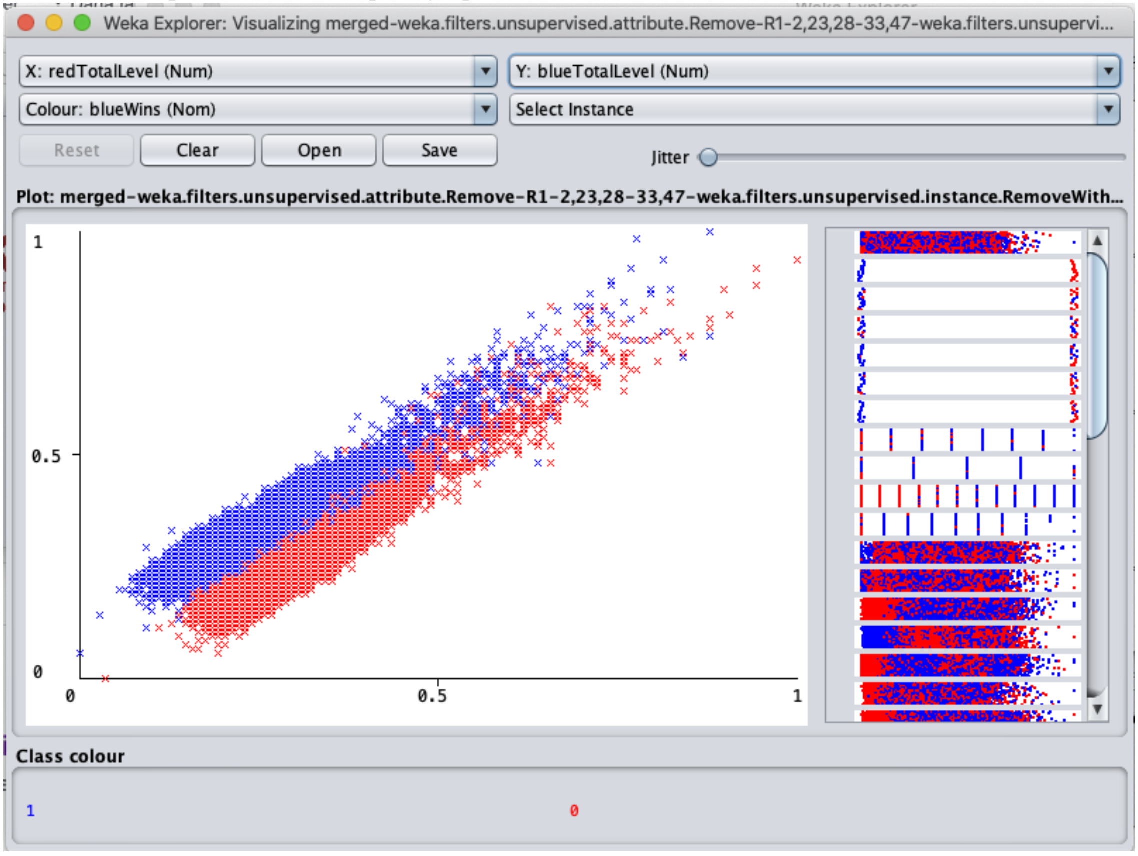
For this model manuel weight analyse is not possible, because there are a lot of layer and they are very complex like human brain. So we can not easily inspect this model so we do not have any data in this section.

4.1.3 Feature Analyse with BruteForce

for this model, this option is highly acceptable. like we mentioned before, in this technique we simply trying the brute-forcing. So this method we can figure out which features are very importants for our model. You can inspect graph below.



Also in this section we do not have decision boundary because of complexity bu we simply show the data values on 2d space. You can inspect graph below.



Weka Explorer: Visualizing merged-weka.filters.unsupervised.attribute.Remove-R1-2,23,28-33,47-weka.filters.unsupervi...

X: redTotalLevel (Num)

Y: blueTotalLevel (Num)

Colour: blueWins (Nom)

Select Instance

Plot: merged-weka.filters.unsupervised.attribute.Remove-R1-2,23,28-33,47-weka.filters.unsupervised.instance.RemoveWith...

5 Clustering

5.1 KMeans

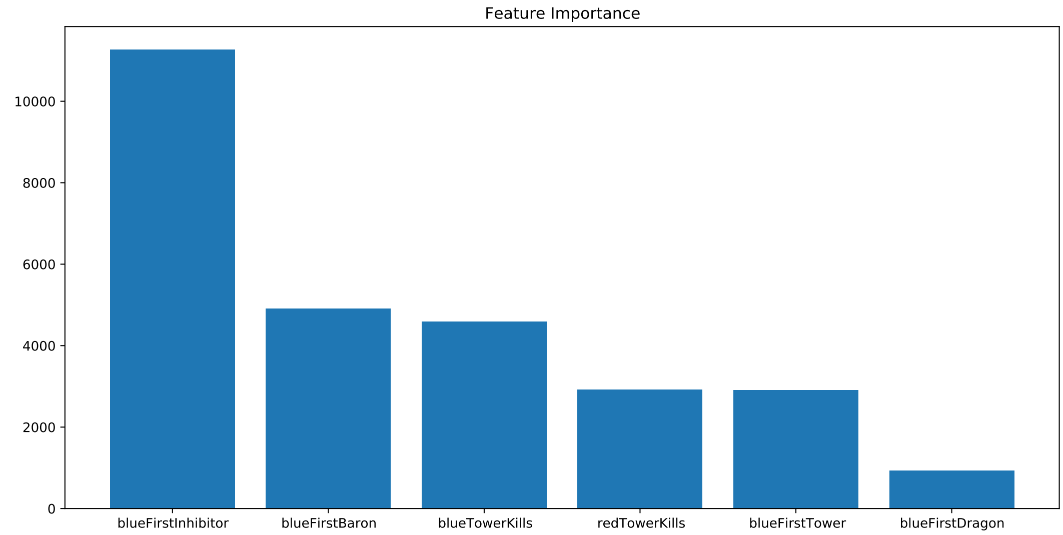
Clustering is the division into groups that show similar characteristics in a data set. There are many similarities within the same cluster and fewer similarities between clusters. There is Unsupervised Learning, meaning no prior information is given. K-Means and Hierarchical Segmentation are among the commonly used clustering algorithms. These algorithms are frequently used such as customer segmentation, market segmentation, and computer vision.

5.1.1 Performance Analyse

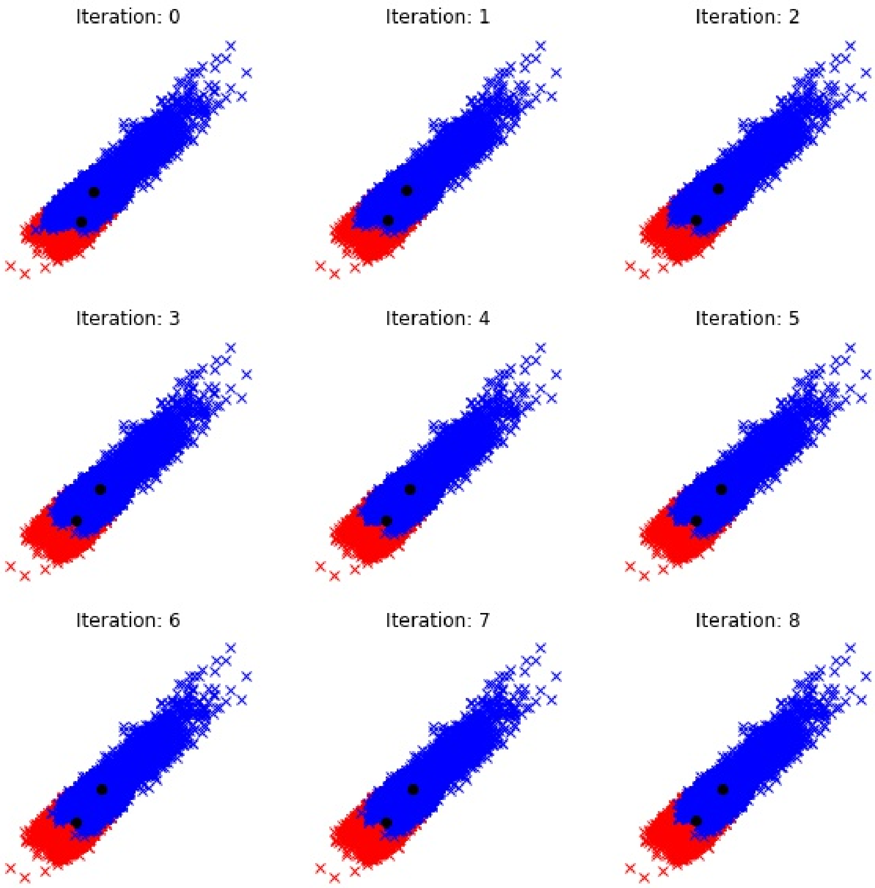
We run like 5 times for get average values you can inspect below table for result. Compared to other models, it have worst accurracy, after some research we decide to our dataset not compatible for clustering. You can look feature importance and step by step clustering graphs below.

Clustering : KMeans Results			
#	Training Time (ms)	Error (cost)	Score (%)
1	530.31	0.123487	-57026.2996
2	518.64	0.876560	-57026.2900
3	545.30	0.876346	-57026.2258
4	518.68	0.876536	-57026.2771
5	522.59	0.123464	-57026.2112
Average	527.10	0.575279	-57026.2607

5.1.2 Feature Importance with BruteForce



5.1.3 Clustering Step by Step



Conclusion

As a conclusion, In this project we experience a lot of different technique and models as a team. We research this problem under 4 main topic and we use 8 different tool and model. Each model select almos same parameters and scores. We get to best result with mlpclassifier. Also clustering section we get worst results and we can not easily fix this, after some research we decide to our dataset not compatible with clustering. In this assigment we learned debug and know how to about models.