DATA WAREHOUSE

Final Project

Analysis of Joe Root's all International Cricket Centuries

KUHDSE22.1F – 018

KUHDSE22.1F – 023

KUHDSE22.1F – 004

KUHDSE22.1F – 019

Objective

Our dataset is Joe Root's All International Cricket Centuries. Joe Root was the captain of the England cricket team and a good performer of the team. He is a all-round cricketer in the England team. We are going to analysis specifically targeting Joe Root's international career and his all centuries of all formats.

We can use data mining algorithms to this dataset to extract meaningful insights and patterns to get many potential benefits.

- Can analyze the performance of international cricket of him. Getting to know the trends, patterns, and factors which help to score successful centuries.
- Analysis of this dataset may give valuable insights to other international teams and coaches to prepare the teams and plan the games against England team.
- Can Identify the strengths and weaknesses of Joe Root according to the century patterns.
- Coaches and selectors are able to understand the things to improve and development of skills of the Joe Root.
- Also, they can understand the countries that the Joe Root has performed well. It will help to take decisions when select the team against those countries.
- This will also be helpful to predict the matches results like the game changer of the match.

Data set description

Link to dataset :-
https://www.kaggle.com/datasets/drahulsingh/joe-root-all-international-cricket-centuries

This dataset includes Joe Root's all century details of all formats (Test, T20, and ODI). The details this dataset is included are the country that the century had been taken, position that Joe Root had played, the inning that the century has been taken, venue of the game played, the ground, date, score of Joe Root and the result of the match. Like that there are 9 columns in this dataset.

Original Data Set Features:

No	Number of the row.
Score	Score that has been taken by the Joe Root of each match that he got centuries.
Against	The country against which Joe Root scored a century.
Position	The place in the team that Joe Root has played at that match.
Innings	The Inning that the Joe Root got the century.
Venue	The place that the match had been played

Ground	The ground place where the match had been played (Home/Away/Neutral)
Date	The date that the match had been hold.
Result	Th result of the match (Won/Lost/Drawn)

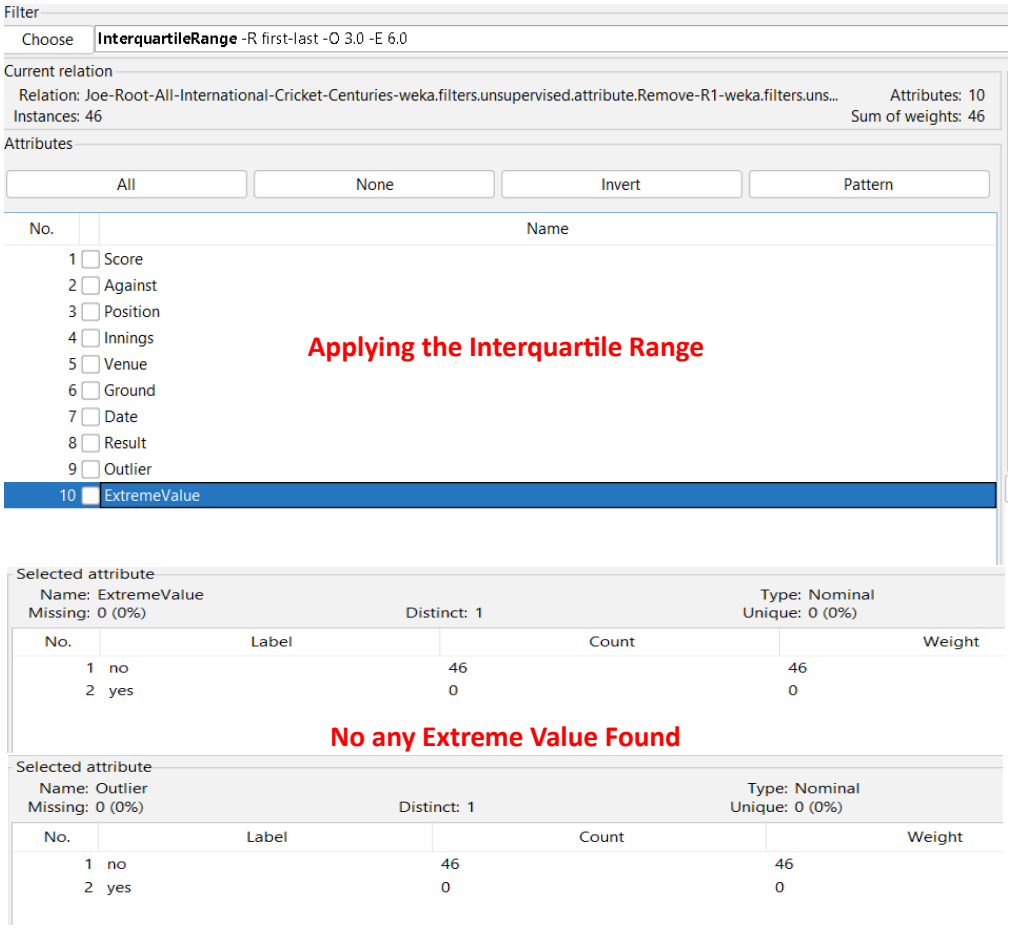
Data Preprocessing

In this dataset there are no any missing values of each attribute, the missing percentage of all attributes is 0%.

We have removed an unnecessary column that we do not need. We removed No column that column denoted the number of each row.

In this dataset we wanted to check the outliers. We used the Interquartile Range algorithm to check the outliers.

As the result of that algorithm in this dataset there is not any outliers or any extreme values found. We can keep the dataset 46 instances as it is.



Data Transformation

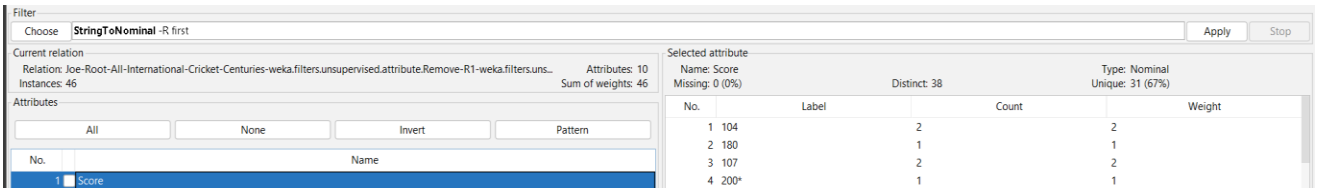
In our dataset some of attributes are in numeric type and one attribute is a string type

- Score – String
- Position – Numeric
- Innings – Numeric

```
@relation Joe-Root-All-International-Cricket-Centuries-weka.filters.unsupervised.attribute.Remove-R1

@attribute Score string
@attribute Against {' New Zealand',' Australia',' West Indies',' Sri Lanka',' India',' South Africa',' Pakistan',' Bangladesh'}
@attribute Position numeric
@attribute Innings numeric
@attribute Venue {'Headingley, Leeds','Lord\'s, London','Sir Vivian Richards Stadium, North Sound','Trent Bridge, Nottingham','The Oval, London','Pallekele
International Cricket Stadium, Kandy','Wellington Regional Stadium, Wellington','National Cricket Stadium, St. George\'s','Edgbaston, Birmingham','Sophia
Gardens, Cardiff','Wanderers Stadium, Johannesburg','SuperSport Park, Centurion','Old Trafford, Manchester','Saurashtra Cricket Association Stadium,
Rajkot','Kensington Oval, Bridgetown','Edgbaston Cricket Ground, Birmingham','University Oval, Dunedin','Daren Sammy Cricket Ground, Gros Islet','Rose Bowl,
Southampton','Seddon Park, Hamilton','Galle International Stadium, Galle','M. A. Chidambaram Stadium, Chennai','Basin Reserve, Wellington'}
@attribute Ground {Home,Away,Neutral}
@attribute Date {24-May-13,18-Jul-13,05-Mar-14,12-Jun-14,09-Jul-14,15-Aug-14,05-Sep-14,10-Dec-14,01-Mar-15,21-Apr-15,09-Jun-15,17-Jun-15,08-Jul-15,06-
Aug-15,14-Jan-16,09-Feb-16,12-Feb-16,22-Jul-16,09-Nov-16,09-Mar-17,01-Jun-17,06-Jul-17,17-Aug-17,07-Mar-18,14-Jul-18,17-Jul-18,07-Sep-18,14-Nov-18,09-
Feb-19,20-Feb-19,03-Jun-19,14-Jun-19,29-Nov-19,14-Jan-21,22-Jan-21,05-Feb-21,04-Aug-21,12-Aug-21,25-Aug-21,08-Mar-22,16-Mar-22,02-Jun-22,10-Jun-22,01-
Jul-22,24-Feb-23,16-Jun-23}
@attribute Result {Won,Drawn,Lost}
```

Here, we want to convert the numeric types to nominal and the string type to nominal.

We used **StringtoNominal** algorithm to convert the string value of score to nominal.



After convert to nominal

@attribute Score {104,180,107,'200* ',154*,'149* ',113,'104* ',121,'182* ',106*,134,130,110,125,109,254,124,101,'133* ',190,136,102,'113* ',100*,122,'100* ',226,228,186,218,'180* ',153,'115* ',176,142*,153*,118*}

We use the method call **discretize** to convert the numerical values to categorical values. We can use equal width binning or equal frequency binning to do this.

We took 2 bins to do the discretize the Position because he has only played as Top order and Middle order batsman and took 4 bins to discretize the Innings because there are only maximum 4 inning in a match. After that the Position and Innings looked like this.

@attribute Position {'\'(-inf-3.5]\'','\'(3.5-inf)\''}

@attribute Innings {'\'(-inf-1.75]\'','\'(1.75-2.5]\'','\'(2.5-3.25]\'','\'(3.25-inf)\''}

Then we replaced the categories names manually using notepad

@attribute Position {Top-Order,Middle-Order}

@attribute Innings {1st,2nd,3rd,4th}

After the transformation the dataset

```
@relation Joe-Root-All-International-Cricket-Centuries-weka.filters.unsupervised.attribute.Remove-R1-weka.filters.unsupervised.attribute.InterquartileRange-Rfirst-last-O3.0-E6.0-
weka.filters.unsupervised.attribute.StringToNominal-Rfirst-weka.filters.unsupervised.attribute.Discretize-B2-M-1.0-R3-precision6-weka.filters.unsupervised.attribute.Discretize-B4-M-1.0-R4-
precision6

@attribute Score {104,180,107,'200* ',154*,'149* ',113,'104* ',121,'182* ',106*,134,130,110,125,109,254,124,101,'133* ',190,136,102,'113* ',100*,122,'100* ',226,228,186,218,'180*
',153,'115* ',176,142*,153*,118*}
@attribute Against {' New Zealand',' Australia',' West Indies',' Sri Lanka',' India',' South Africa',' Pakistan',' Bangladesh'}
@attribute Position {Top-Order,Middle-Order}
@attribute Innings {1st,2nd,3rd,4th}
@attribute Venue {'Headingley, Leeds','Lord\'s, London','Sir Vivian Richards Stadium, North Sound','Trent Bridge, Nottingham','The Oval, London','Pallekele International Cricket Stadium,
Kandy','Wellington Regional Stadium, Wellington','National Cricket Stadium, St. George\'s','Edgbaston, Birmingham','Sophia Gardens, Cardiff','Wanderers Stadium, Johannesburg','SuperSport
Park, Centurion','Old Trafford, Manchester','Saurashtra Cricket Association Stadium, Rajkot','Kensington Oval, Bridgetown','Edgbaston Cricket Ground, Birmingham','University Oval,
Dunedin','Daren Sammy Cricket Ground, Gros Islet','Rose Bowl, Southampton','Seddon Park, Hamilton','Galle International Stadium, Galle','M. A. Chidambaram Stadium, Chennai','Basin Reserve,
Wellington'}
@attribute Ground {Home,Away,Neutral}
@attribute Date {24-May-13,18-Jul-13,05-Mar-14,12-Jun-14,09-Jul-14,15-Aug-14,05-Sep-14,10-Dec-14,01-Mar-15,21-Apr-15,09-Jun-15,17-Jun-15,08-Jul-15,06-Aug-15,14-Jan-16,09-Feb-16,12-
Feb-16,22-Jul-16,09-Nov-16,09-Mar-17,01-Jun-17,06-Jul-17,17-Aug-17,07-Mar-18,14-Jul-18,17-Jul-18,07-Sep-18,14-Nov-18,09-Feb-19,20-Feb-19,03-Jun-19,14-Jun-19,29-Nov-19,14-Jan-21,22-
Jan-21,05-Feb-21,04-Aug-21,12-Aug-21,25-Aug-21,08-Mar-22,16-Mar-22,02-Jun-22,10-Jun-22,01-Jul-22,24-Feb-23,16-Jun-23}
@attribute Result {Won,Drawn,Lost}
@attribute Outlier {no,yes}
@attribute ExtremeValue {no,yes}

@data
104,' New Zealand',Middle-Order,1st,'Headingley, Leeds',Home,24-May-13,Won,no,no
180,' Australia',Top-Order,3rd,'Lord\'s, London',Home,18-Jul-13,Won,no,no
107,' West Indies',Middle-Order,1st,'Sir Vivian Richards Stadium, North Sound',Away,05-Mar-14,Won,no,no
'200* ',' Sri Lanka',Middle-Order,1st,'Lord\'s, London',Home,12-Jun-14,Drawn,no,no
154*,' India',Middle-Order,2nd,'Trent Bridge, Nottingham',Home,09-Jul-14,Drawn,no,no
'149* ',' India',Middle-Order,2nd,'The Oval, London',Home,15-Aug-14,Won,no,no
113,' India',Middle-Order,1st,'Headingley, Leeds',Home,05-Sep-14,Won,no,no
'104* ',' Sri Lanka',Middle-Order,2nd,'Pallekele International Cricket Stadium, Kandy',Away,10-Dec-14,Won,no,no
121,' Sri Lanka',Middle-Order,1st,'Wellington Regional Stadium, Wellington',Neutral,01-Mar-15,Lost,no,no
```

Once all above processes were done, the preprocessing and transformation parts were completed. Now we are able use the processed data set to do any type of datamining techniques.

Data Mining Process

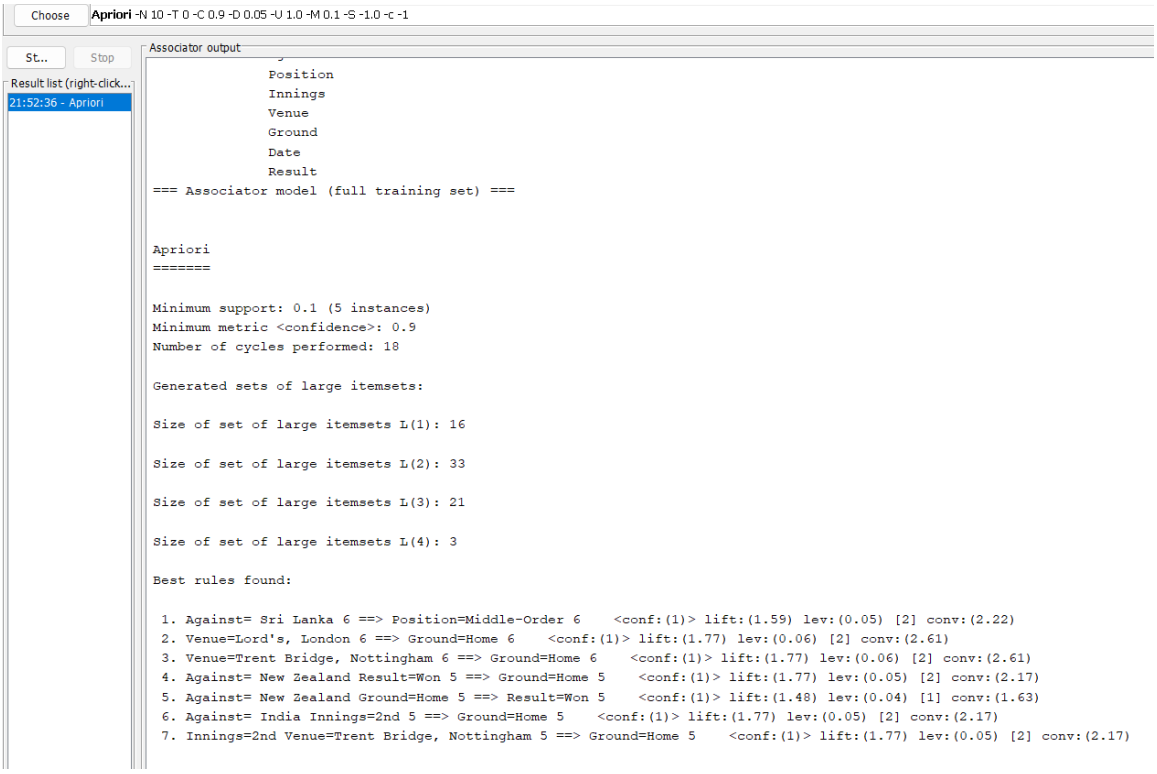
Now it can be applied some data mining algorithms to these processed dataset. Here, we are going to use Apriori, Decision tree (J48), and Naive Bayes algorithms.

Apriori algorithm

This is a algorithm that used to association rule mining. This can be used to this dataset to discover interesting associations or patterns among the attributes.

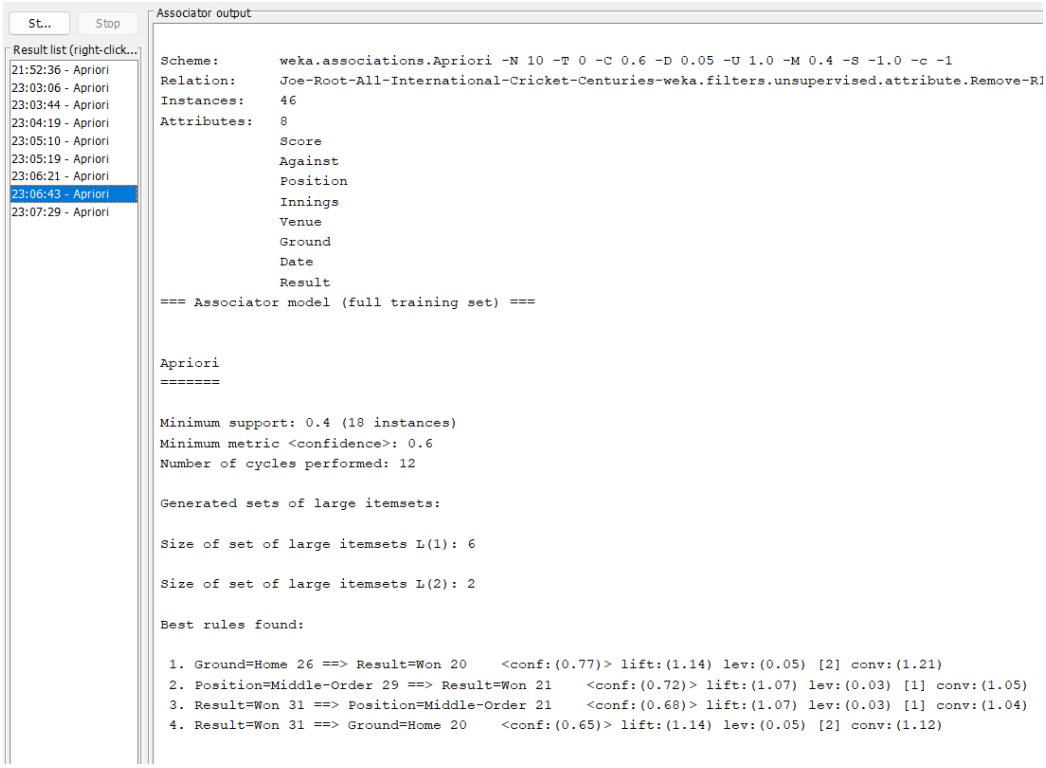
The result of this algorithm after adding it to the dataset will help to get the patterns or associations. From this it can be understood that Joe Root tends to score centuries more often against certain teams or in specific match formats.

First applied this algorithm to dataset with the min_confidence = 0.9 and min_sup = 0.1. But considering the patterns of that those are not reliable. Therefore we tried many times with changing the min_confidence and the min_sup.



Minimum support = 0.1 | Minimum confidence = 0.9 | Number of cycles performed = 18

Like this we applied the apirori algorithm so many times to the dataset.
Finally we can got patterns that can be reliable after checking the pattern.

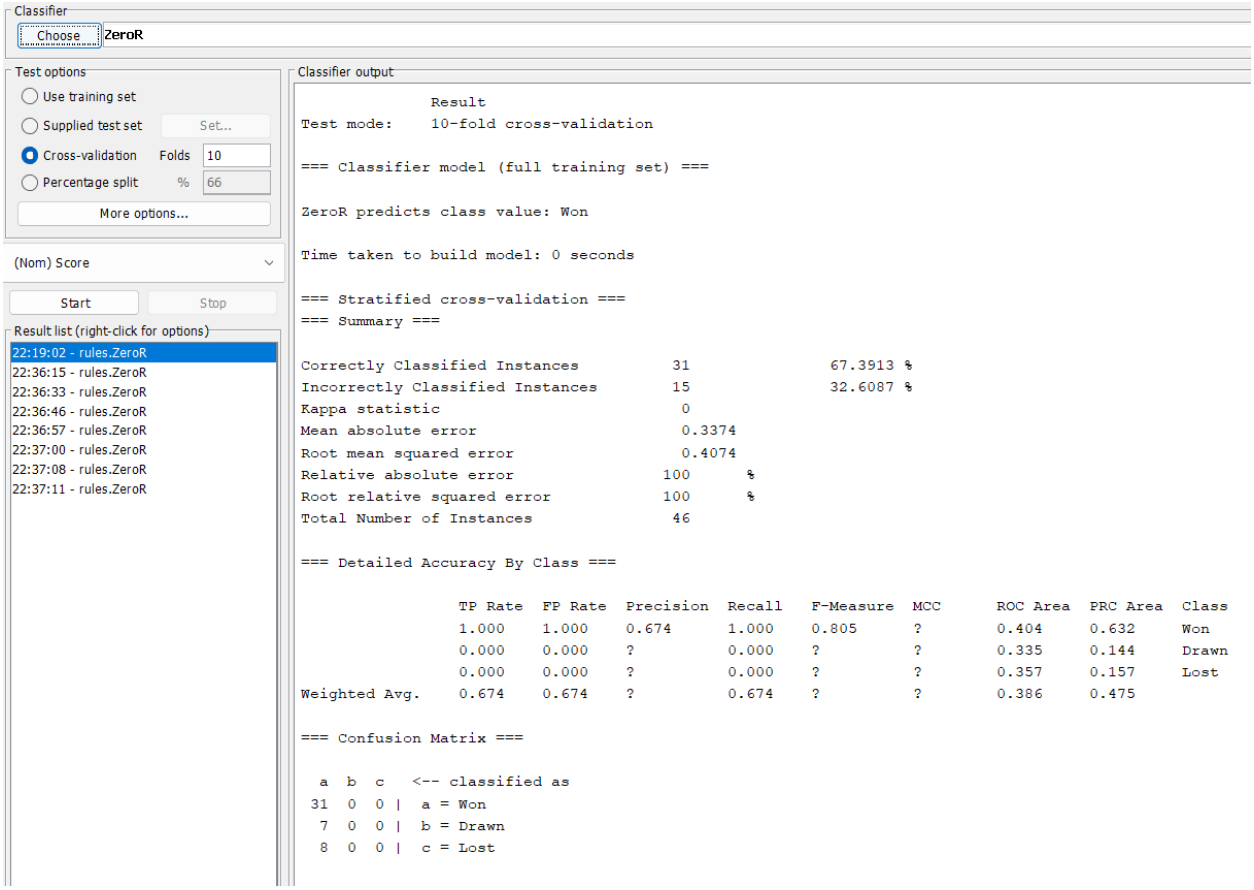


Here, we put the min_confidence = 0.6 and min_sup = 0.4. Then we can got a reliable patterns as Joe Root's century taking patterns.

Here, we can say Joe Root took many centuries in home and most of the matches were won. And he has best performance as a middle-order batsman. He is performing well at home grounds rather than away from the England.

Classifying

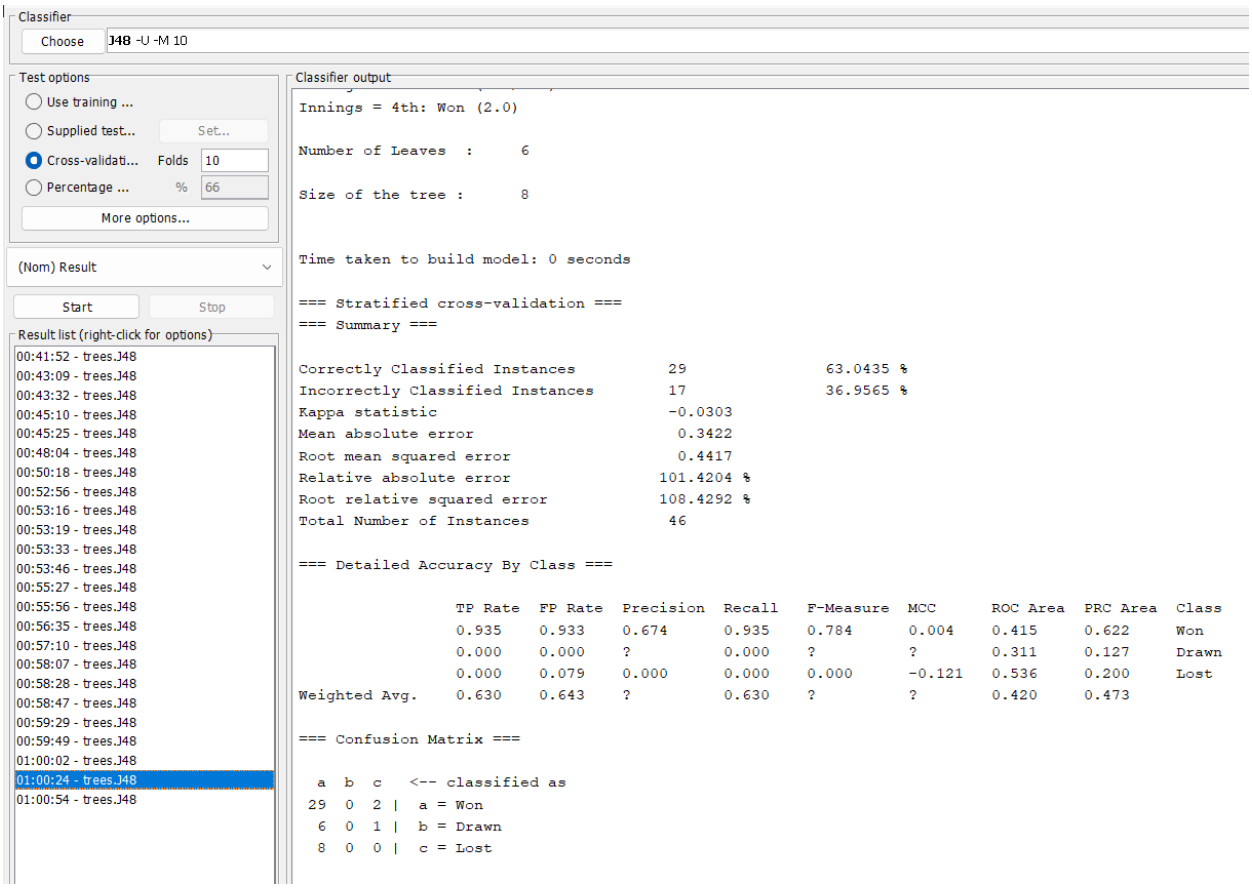
Before adding the J48 we will applied zeroR algorithm to the dataset. It is a very simple algorithm and we can compare the accuracy of other algorithms with **zeroR** algorithm.



After applying zeroR algorithm we can see there are 31 correctly classified instances and 15 of incorrectly classified instances. Correctly classified instances as a percentage 67.39% and incorrectly classified instances 32.6%. Accuracy by class the target variable we have used is Result. According to this accuracy we can compare this with other algorithms.

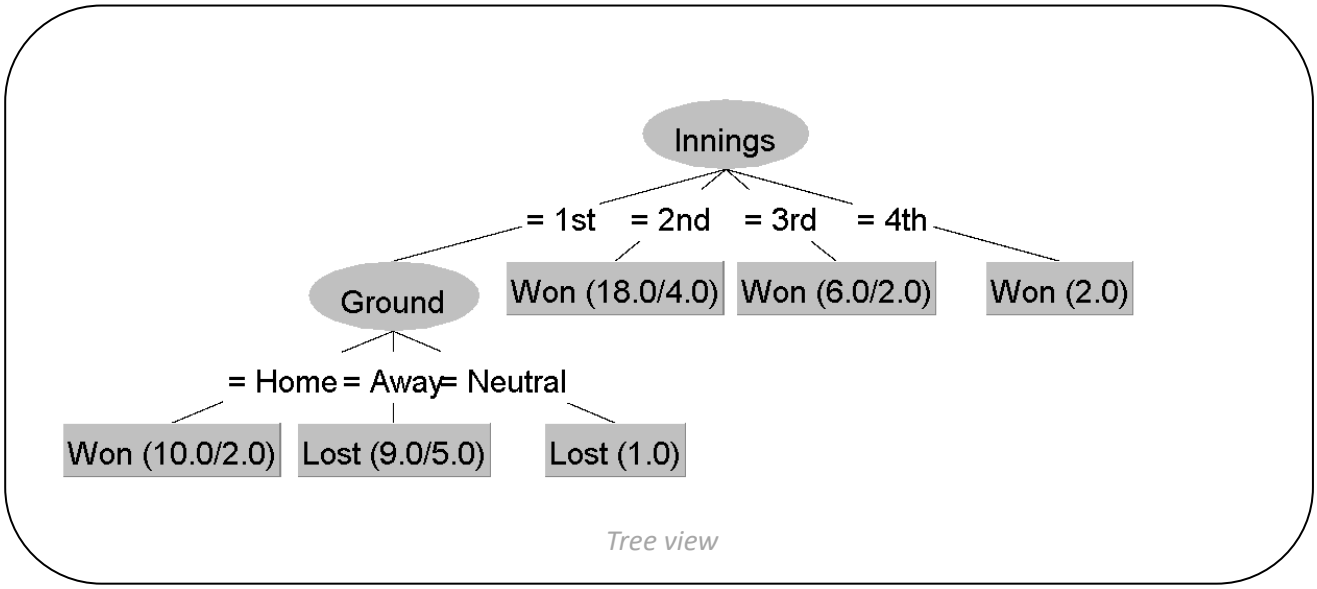
Decision tree (J48)

Now we are going to use J48 algorithm to this dataset. It may classify whether a given match which Joe Root scored century had won or not.



After applied the J48 algorithm among 46 of instances the correctly classified instances are 29 and the percentage is 63.04%. The incorrectly classified instances are 17 and the percentage is 36.95%.

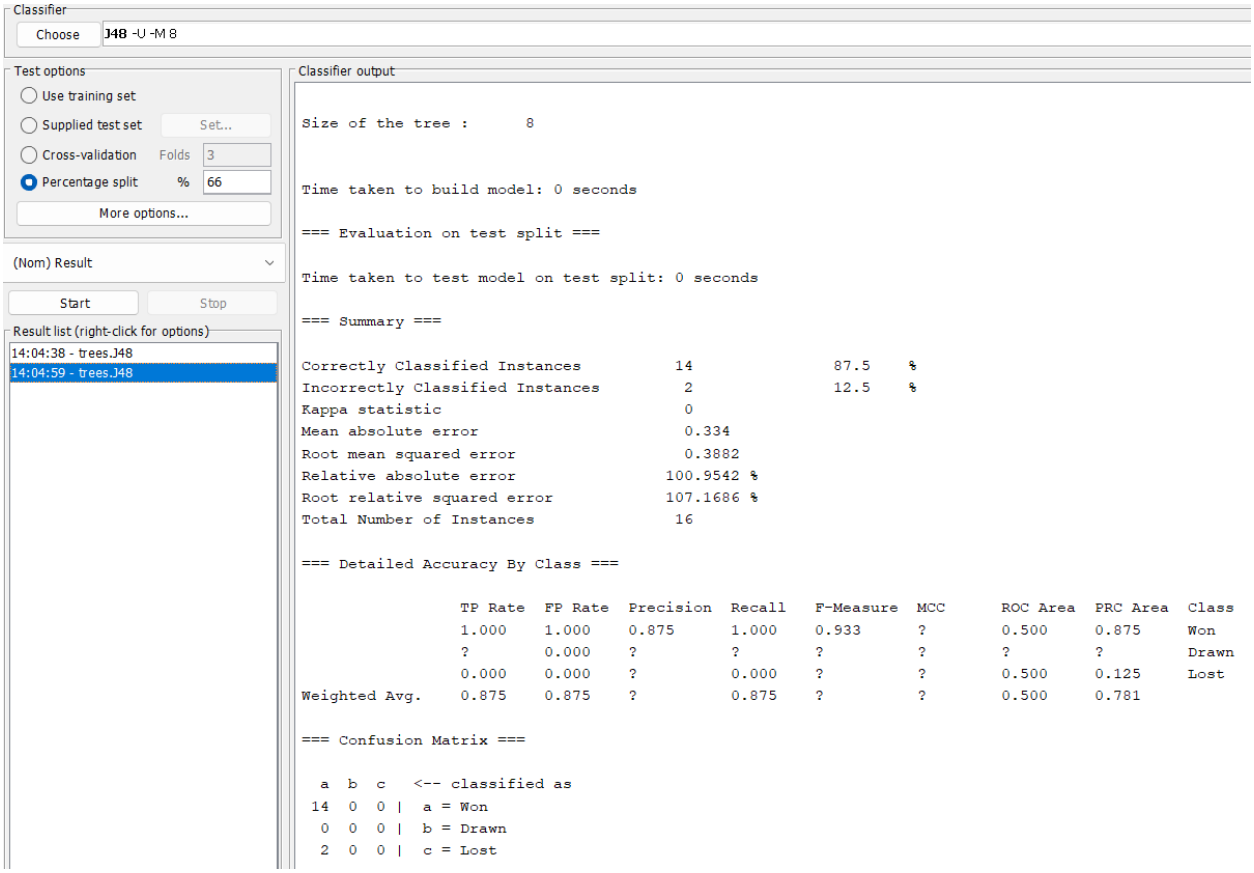
The decision tree view is helps to understand how the algorithm partitions the dataset based on different attributes and creates decision rules for classification or predication tasks.



Tree view

J48 Percentage split

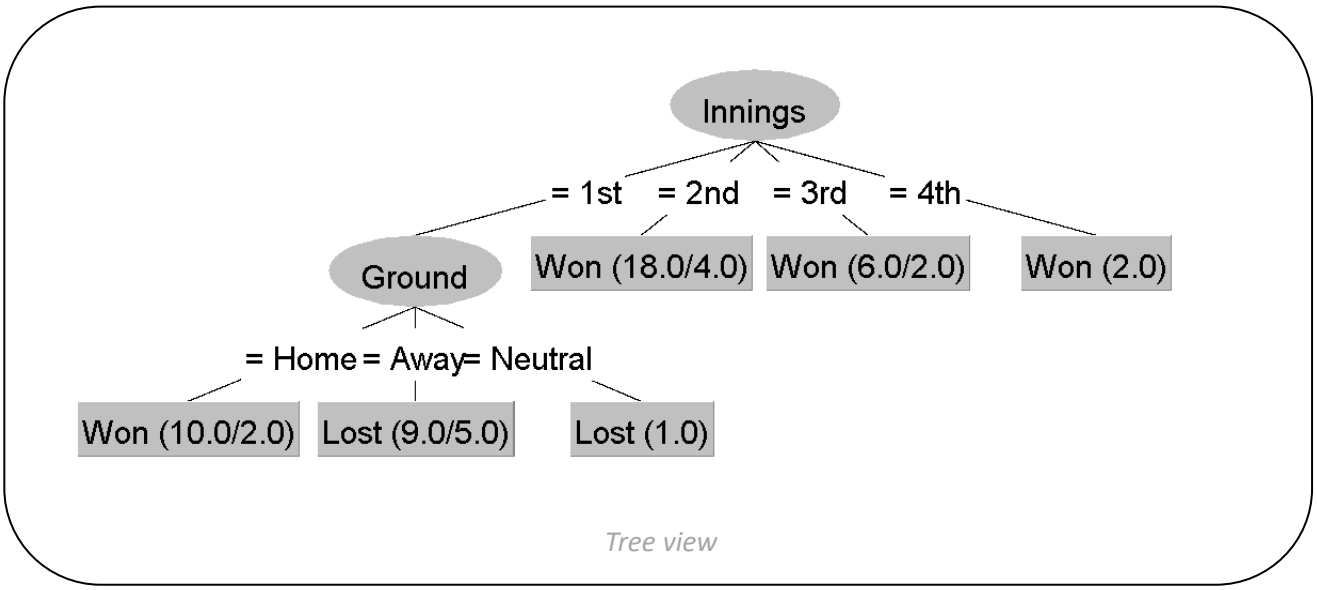
Now, to the dataset again going to be applied the J48. Here, it will be applied with percentage split. It means this algorithm will be applied to 66% percentage of data from this dataset. 66% is the training and 34% is testing split. The dataset is partitioned 70% of the instances will be using to training the J48 model remaining 34% is used to evaluate the model's performance.



Here, we can see the correctly classified instances are 14 and the percentage is 87.5%. Incorrectly classified instances are 2 the percentage is 12.5%. Total of instances that has been taken to applied the J48 is 16 from 46.

When the accuracy of this result is compared to the previous result the percentage split accuracy has increased than previous one.

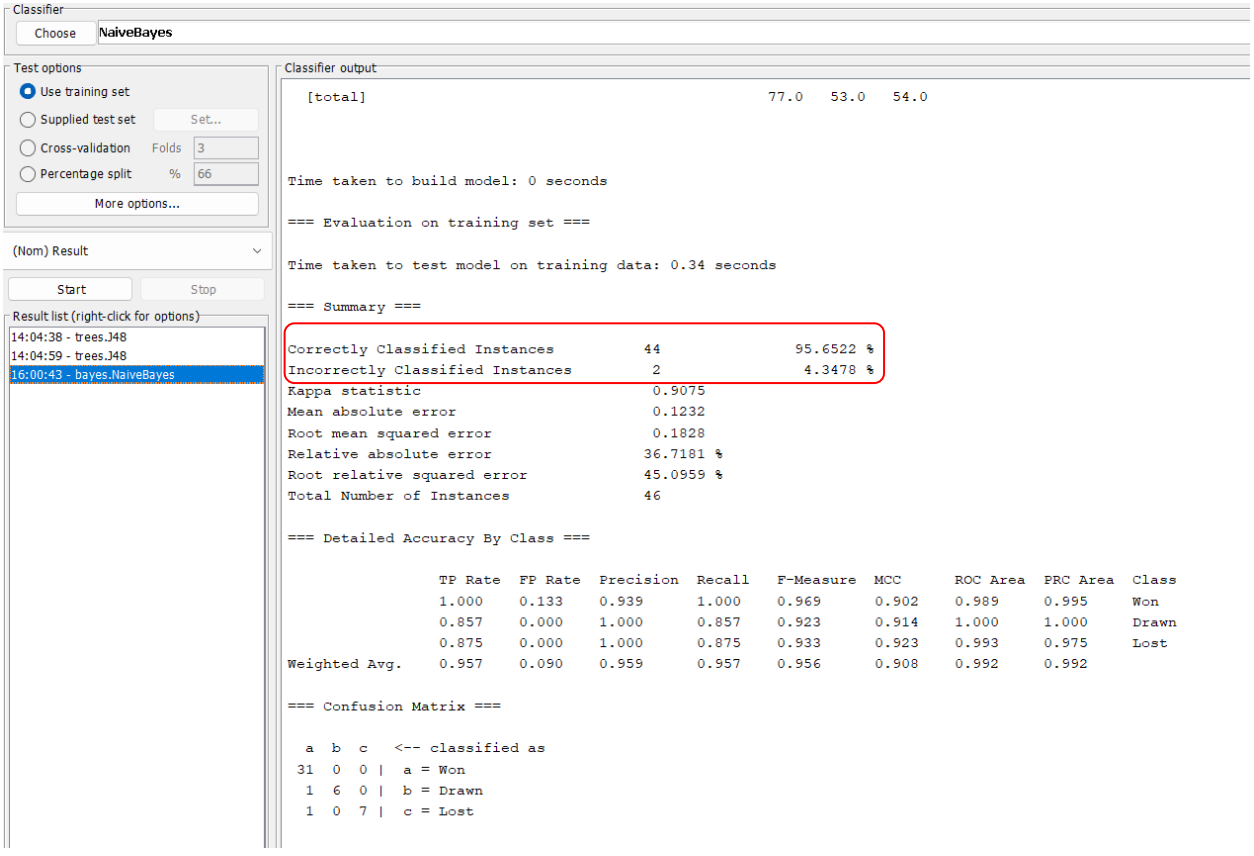
The decision tree view will give a clear idea about how the algorithm has worked. Below is the decision tree view of the percentage split J48



Tree view

Naive bayes algorithm

After applying J48 algorithm we applied the naive bayes algorithm to the dataset using training set.



Classifier

Choose NaiveBayes

Test options

Use training set
Supplied test set Set...
Cross-validation Folds 3
Percentage split % 66
More options...

(Nom) Result

Start Stop

Result list (right-click for options)

14:04:38 - trees.J48
14:04:59 - trees.J48
16:00:43 - bayes.NaiveBayes

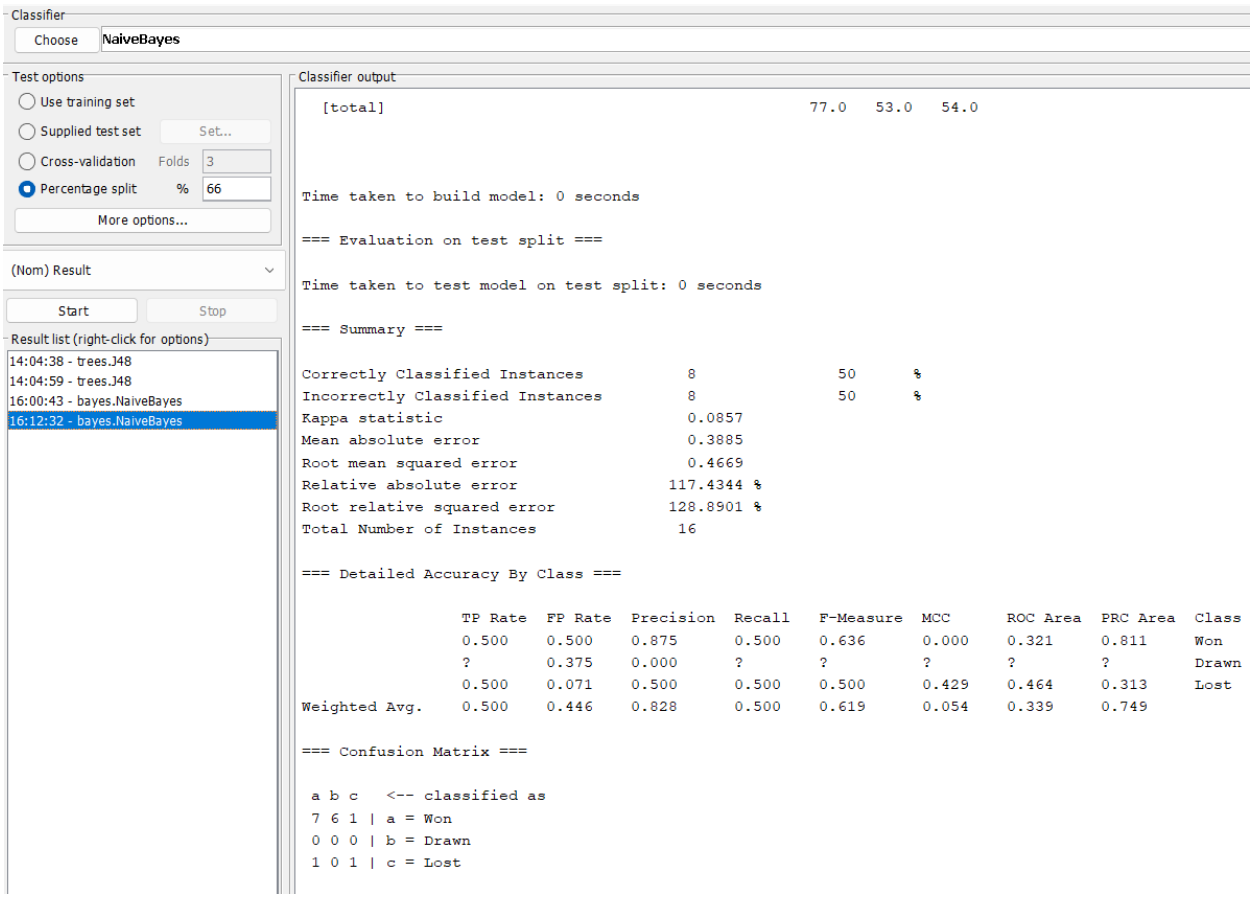
Classifier output

```
  [total]                                          77.0    53.0    54.0



Time taken to build model: 0 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.34 seconds

=== Summary ===

Correctly Classified Instances          44               95.6522 %
Incorrectly Classified Instances         2                4.3478 %
Kappa statistic                          0.9075
Mean absolute error                      0.1232
Root mean squared error                  0.1828
Relative absolute error                 36.7181 %
Root relative squared error             45.0959 %
Total Number of Instances               46

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.133    0.939      1.000    0.969      0.902    0.989     0.995     Won
                 0.857    0.000    1.000      0.857    0.923      0.914    1.000     1.000     Drawn
                 0.875    0.000    1.000      0.875    0.933      0.923    0.993     0.975     Lost
Weighted Avg.    0.957    0.090    0.959      0.957    0.956      0.908    0.992     0.992

=== Confusion Matrix ===

  a  b  c   <-- classified as
 31  0  0 |  a = Won
  1  6  0 |  b = Drawn
  1  0  7 |  c = Lost
```

Here the correctly classified instances are 44 and the percentage of it is 95.6522%. The incorrectly classified instances is 2 and the percentage of it is 4.3478%. We can say this is the highest accurate algorithm among the other algorithms that we have applied so far.

Naive bayes using percentage split

Next we applied the naïve bayes algorithms using 66% percentage of training data and the remaining 34% testing data.



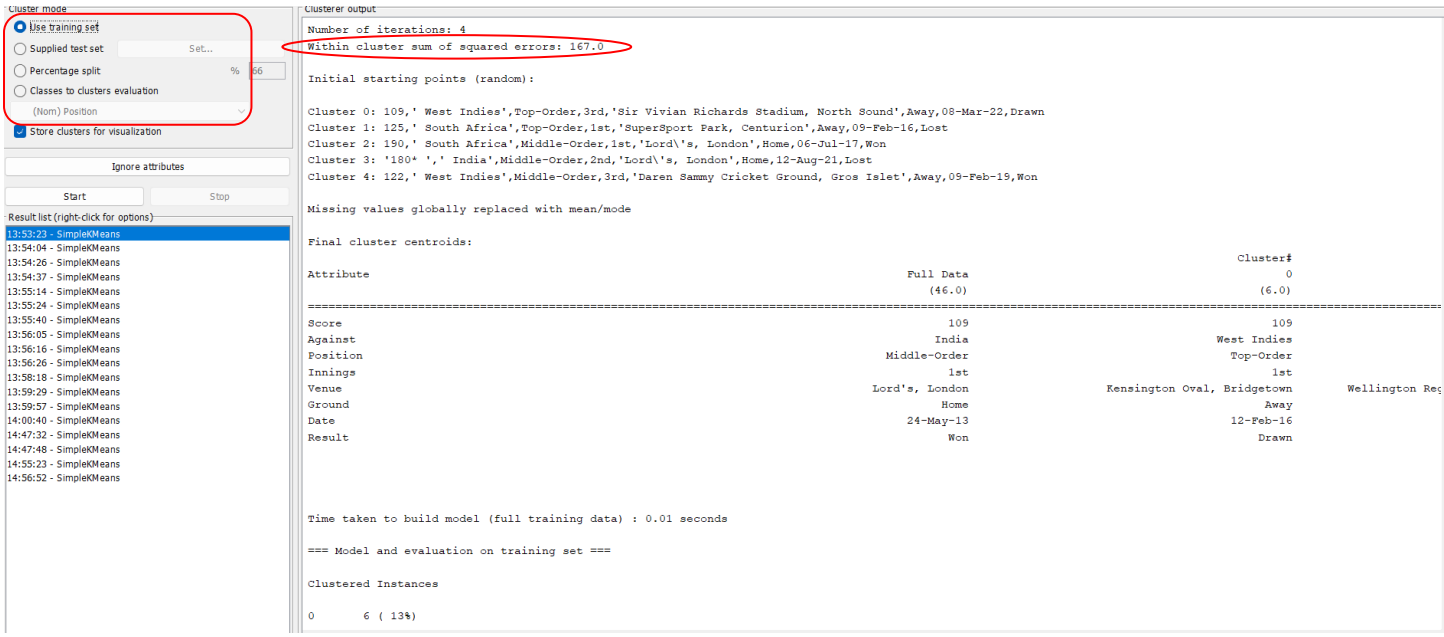
Here, the both correctly classified instances and incorrectly classified instances are the same 50% and 50%. It has been used only 16 instances from 46. We can see the accuracy of naïve bayes using percentage split is very lower than naïve bayes with training set.

Clustering

To this dataset we are going to apply the clustering algorithm. Here, we will use **SimpleKmeans** algorithm to do the clustering. This will give data exploration by organizing the data into meaningful groups. It will be easy to understand the distribution and relationships between this dataset`s attributes.

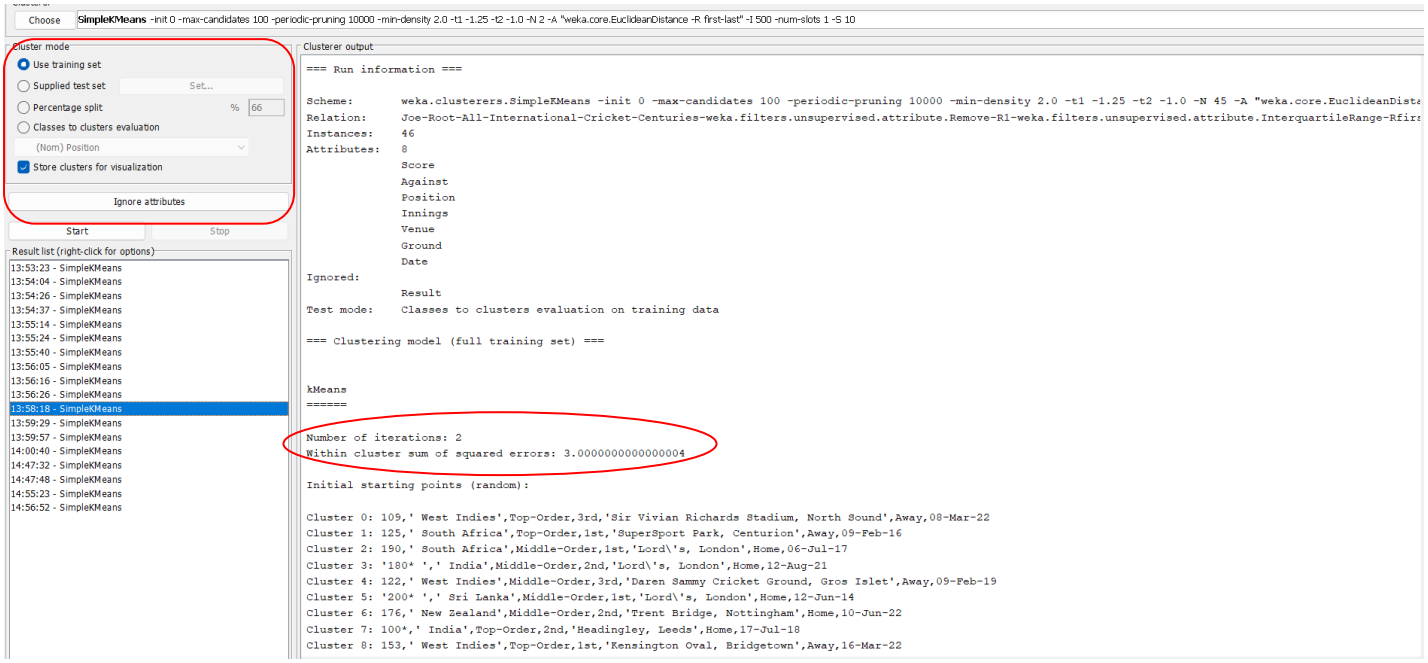
Firstly we applied the Kmeans to the dataset having 3 clusters. But the squared error was high. And the number of iteration is 4.

Within cluster sum of squared errors: 167.0



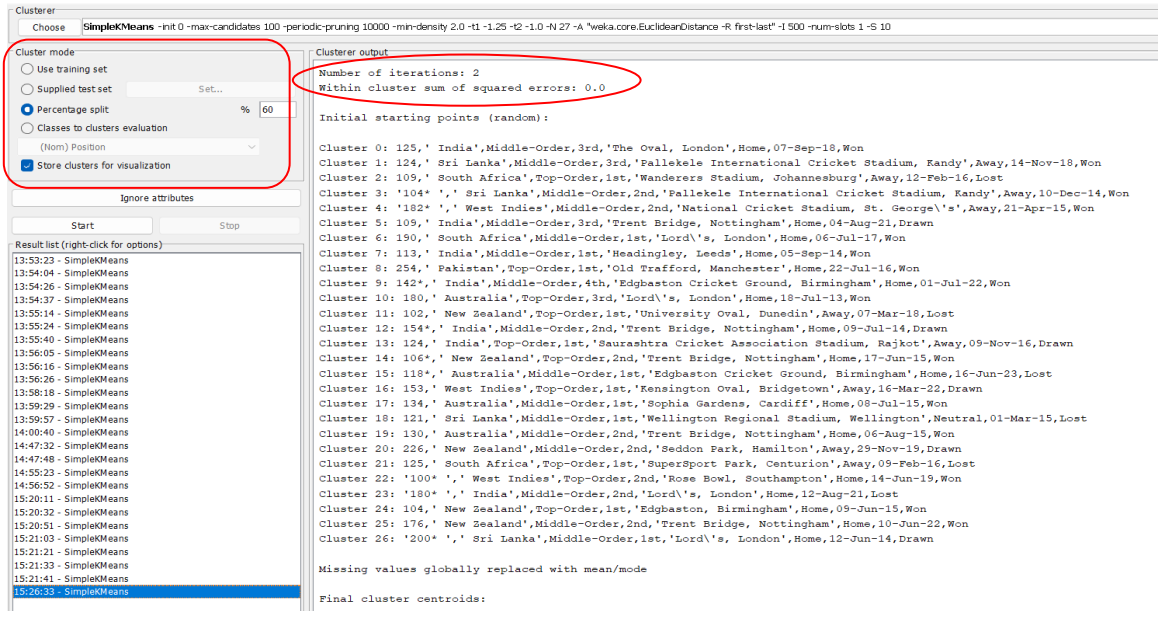
Then we generated many models. After that we could found a model with low squared error. Here we needed to put 45 clusters to reduce the squared error. Number of iterations is 2.

Within cluster sum of squared errors: 3.00



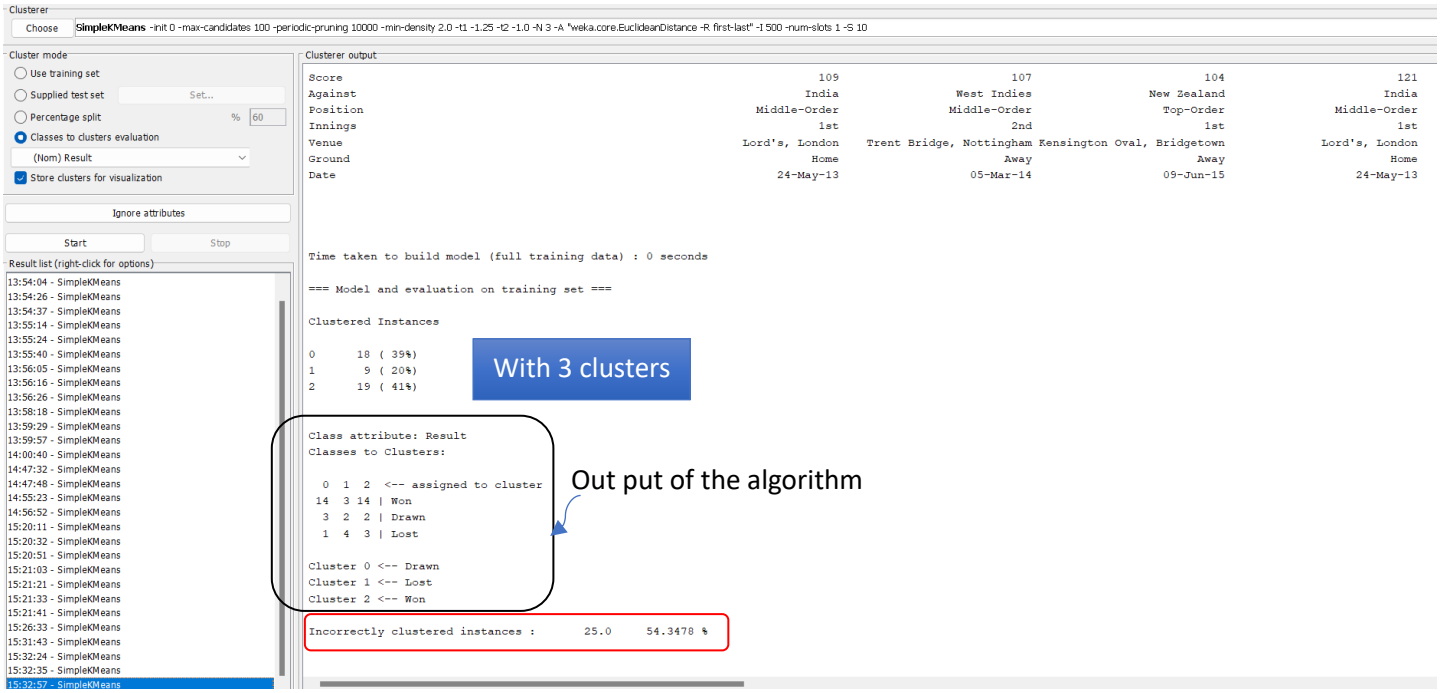
Clustering with percentage split

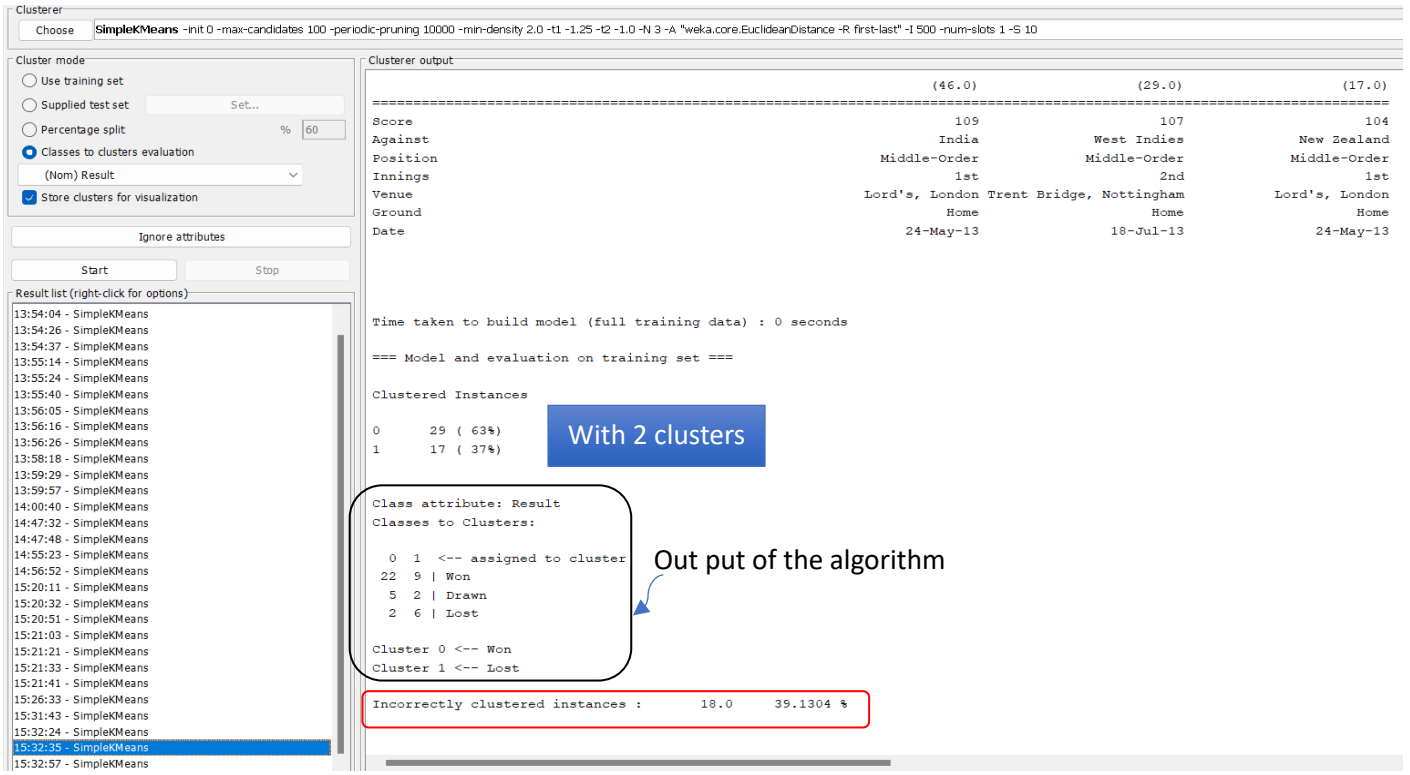
Then we applied the clustering to the dataset. We took 60% of data to train and the 40% of remaining data to test. Here, we could reduce the number of clusters to 27 and the squared error was reduced to 0.00. Number of iterations is 2.



Then we used **classes to cluster evaluation** to result attribute. We used 3 clusters initially but the incorrectly clustered instances were little bit high (25.00). As a percentage it was 54.3478 %. Then we decided to reduce clusters to 2. Then the incorrectly clustered instances were 18.0.

As a percentage it was 39.1304 %. Here the squared error was high when the number of clusters reduce but the incorrectly clustered instances became low.





Studying above algorithms we can take some most accurate algorithms that worked well for this particular dataset:

- Apriori algorithm
- J48 algorithm with percentage split a
- Naive bayes algorithm using training set
- clustering with classes to cluster evaluation (result attribute)

Results and patterns

Considering above algorithms applied to the dataset. We can identify Joe Root is performing well at home grounds. And he has taken most of the centuries at the home. He is mostly preferred and performed to play as a middle order batsman. Most of the matches that Joe Root performed well had won by the England. We can identify Joe Root has little bit low performance of matches that is playing away from home. Also, away from home some of matches were lost even Joe Root performed well. From that we can say England team is most comfortable to win the matches at home.