MILESTONE-4

Member Name : VISHAL BHURANGI
Roll Number : MT2016154

1. CLUSTERING REPORT

Target User :- Captains/Bookies

Algorithm : Kmeans

1.1.Problem Formulation: To build clusters of Cricket Stadiums according to their “win probability batting first” on that particular stadium and “win probability batting second” on each stadium so that it helps captains or bookies visually see whether it is beneficial to bat first or bowl first.

1.2.Data Preparation:
1.We first calculated win probabilty batting first at each stadium and win probability bowling first at each stadium.

R script for preparation:-

```
matches <- read_csv("matches.csv")

deliveries <- read_csv("deliveries.csv")

data7<-as.data.frame(matches)
for(i in 1:length(data7$win_by_runs)){
  if(data7$win_by_runs[i]!=0)
    data7$win_by_runs[i]=1
  else
    data7$win_by_runs[i]=0
}
for(i in 1:length(data7$win_by_wickets)){
  if(data7$win_by_wickets[i]!=0)
    data7$win_by_wickets[i]=1
  else
    data7$win_by_wickets[i]=0
}

data7<-as.data.frame(data7)
```

```
df2<-sqldf("SELECT id,venue,COUNT(distinct(id)) as
number_of_matches,sum(win_by_runs) as
number_of_matches_won_by_runs ,sum(win_by_wickets) as
numberOfMatchesWonByWickets,
(sum(win_by_runs)*1.0/COUNT(distinct(id))) probabiltyOfWinBattingFirst,
((sum(win_by_wickets)*1.0/COUNT(distinct(id)))) as
probabiltyOfWinBowlingFirst FROM data7  GROUP BY venue")

df3<-sqldf("select
probabiltyOfWinBattingFirst,probabiltyOfWinBowlingFirst,venue from df2 ")
```

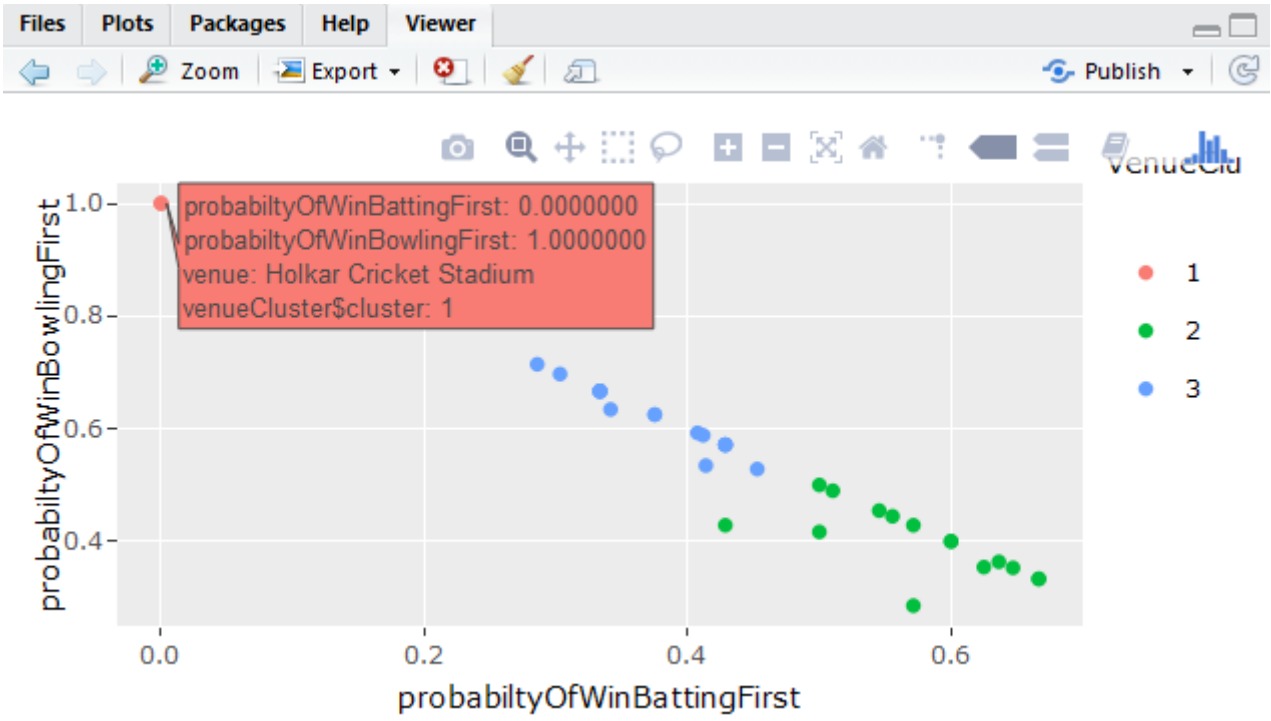
1.3. R-Code for kmeans and plotting :

```
attach(df3)
#plot(probabiltyOfWinBattingFirst,probabiltyOfWinBowlingFirst,
main="Scatterplot Example",
#xlab="winbattingfirst ", ylab="winbowlingfirst ")
#ggplot(df2, aes(probabiltyOfWinBattingFirst, probabiltyOfWinBowlingFirst,
color = venue)) + geom_point()
#set.seed(10)
venueCluster <- kmeans(df3[,1:2 ], 3, nstart = 15)
venueCluster$cluster <- as.factor(venueCluster$cluster)

library("plotly")
p <- ggplot(df3, aes(probabiltyOfWinBattingFirst,
probabiltyOfWinBowlingFirst,label=venue, color = venueCluster$cluster)) +
geom_point()
ggplotly(p)
venueCluster
```

1.4.Evaluation Of Model

```
Console ~/
We recommend that you use the dev version of ggplot2 with `ggplotly()`
Install it with: `devtools::install_github('hadley/ggplot2')`
> venueCluster
K-means clustering with 3 clusters of sizes 3, 15, 17

Cluster means:
  probabiltyOfWinBattingFirst probabiltyOfWinBowlingFirst
1                  0.06666667                   0.9333333
2                  0.57495990                   0.3990480
3                  0.37727191                   0.6171409

Clustering vector:
 [1] 2 2 2 3 3 2 3 3 3 1 2 1 3 2 3 2 3 2 3 2 2 3 3 3 2 1 3 3 3 2 3 2 3 2 2
Levels: 1 2 3

Within cluster sum of squares by cluster:
[1] 0.05333333 0.12017818 0.09313193
 (between_SS / total_SS =  85.8 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
> #venueCluster<-kmeans(df3,centers = 3)
```

Output:Clusters Of Stadiums



Explanation of Output:

Here We have made three clusters of stadiums.The left most cluster shows stadiums which have high probability of winning bowling first and low probability of winning batting first. Similarly right most cluster shows stadiums which have high probability of winning batting first and low probability of winning bowling first.The middle cluster have medium probabilty of winning either batting first or bowling first.

1.5.Target User : Team Captain/Bookies Why ? : it helps captains visually see whether it is benefitial to bat first or bowl first in that stadium.
For bookies ,it helps to bet on the team that bats first on a ground which belongs to cluster having high probability winning batting first.

2.ASSOCIATION RULES

Target User : Team Management/Captain

Algorithm : Apriori

2.1.Problem Formulation: To formulate rules that help find association between “Runs scored”,”Number of dot balls”,”Number of wickets” on a particular stadium by a particular team to win matches.

2.2.Data Preparation:
1.We calculated runs scored on particular stadium (here M Chinnaswamy) no of dot balls,no of wickets by bangalore.
2.According to above collected data we made new dataframe having columns “Runs scored greater tan 160”,”Dot balls less than 40”,”Wickets fallen <5”,”Result”.
3.We filled columns with Zero if the data satisfies column names else we fill with one.

R Script for Data Preparation

```
parone <- sqldf("select id,sum(total_runs) as total_runs,sum(wide_runs) as total_dots,sum(is_super_over) as total_wickets from mergeddata where venue=\"M Chinnaswamy Stadium\" and batting_team=\"Royal Challengers Bangalore\" group by match_id")

#another loop for wide_runs same as the below one
```

```
for(i in 1:length(mergeddata$player_dismissed)){
  if(is.na(mergeddata$player_dismissed[i]))
    mergeddata$is_super_over[i]=0
  else
    mergeddata$is_super_over[i]=1
}


partwo <- sqldf("select match_id,winner from mergeddata where  venue=\"M
Chinnaswamy Stadium\" and batting_team=\"Royal Challengers Bangalore\"
group by match_id")

for(i in 1:length(partwo$winner)){
  if(partwo$winner[i]!="Royal Challengers Bangalore")
    partwo$result[i]=0
  else
    partwo$result[i]=1
}

partwo$winner <- NULL

parthree <- sqldf("select match_id,sum(batsman_runs) as bat from mergeddata
where  venue=\"M Chinnaswamy Stadium\" and batting_team=\"Royal
Challengers Bangalore\" group by match_id,batsman")
parfour <- sqldf("select match_id,max(bat) as highest_runs from parthree group
by match_id")

for(i in 1:length(parfour$highest_runs)){
  if(parfour$highest_runs[i]>60)
    parfour$highest_score_greater_sixty[i]=1
  else
    parfour$highest_score_greater_sixty[i]=0
}

parfour$highest_runs <- NULL

for(i in 1:length(parone$total_runs)){
  if(parone$total_runs[i]>160)
    parone$total_runs[i]=1
  else
    parone$total_runs[i]=0
```

```
  if(parone$total_dots[i]<40)
    parone$total_dots[i]=1
  else
    parone$total_dots[i]=0

  if(parone$total_wickets[i]<5)
    parone$total_wickets[i]=1
  else
    parone$total_wickets[i]=0
}

colnames(parone) <-
c("match_id","runs_scored_greater_160","dot_balls_less_40","total_wickets_le
ss_5")
colnames(parfour) <- c("match_id","highest_score_greater_60")


merge_asso_one <- sqldf("select
parone.match_id,runs_scored_greater_160,dot_balls_less_40,total_wickets_less
_5,highest_score_greater_60,result from parone,partwo,parfour where
parone.match_id=partwo.match_id and partwo.match_id=parfour.match_id")


merge_asso_one$match_id <- NULL
```

2.3. R-Code for algorithm and plotting :

```
library(arules)
#rules<-apriori(merge_asso_one)
#summary(rules)

merge_asso_one_final <- read.csv("merge_asso_one.csv",header=T,colClasses
= "factor")

rules <- apriori(merge_asso_one_final,parameter=list(supp=.2,conf=0.5),
                 appearance=list(rhs=c("result=1"), default="lhs"))
#we are looking for rules that has result as winning,thats why result=1.
rules = sort(rules,by="lift")

inspect(rules)
```

2.4.Evaluation Model

Association Rules

```
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [10 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
> rules = sort(rules,by="lift")
>
> inspect(rules)
     lhs                                rhs          support confidence      lift count
[1]  {dot_balls_less_40=1,
      total_wickets_less_5=1}     => {result=1} 0.2264151  1.0000000 1.8928571    12
[2]  {total_wickets_less_5=1,
      highest_score_greater_60=1} => {result=1} 0.2075472  0.9166667 1.7351190    11
[3]  {total_wickets_less_5=1}     => {result=1} 0.3207547  0.8947368 1.6936090    17
[4]  {dot_balls_less_40=1}        => {result=1} 0.3018868  0.7272727 1.3766234    16
[5]  {runs_scored_greater_160=1,
      highest_score_greater_60=1} => {result=1} 0.2641509  0.7000000 1.3250000    14
[6]  {highest_score_greater_60=1} => {result=1} 0.3773585  0.6666667 1.2619048    20
[7]  {dot_balls_less_40=0,
      highest_score_greater_60=1} => {result=1} 0.2075472  0.5789474 1.0958647    11
[8]  {runs_scored_greater_160=1}  => {result=1} 0.2830189  0.5555556 1.0515873    15
[9]  {}                           => {result=1} 0.5283019  0.5283019 1.0000000    28
[10] {runs_scored_greater_160=0}  => {result=1} 0.2452830  0.5000000 0.9464286    13
>
```

Explanation: Here we get 10 association rules .The first rule which has highest “confidence” and “lift” says that if “dot ball less than 40”is true and “total wickets less than 5” is true then their is greater chance of bangalore winning that match at that(M Chinnaswamy) stadium.
Similarly other rules suggest different association between columns for winning.

2.5.Target User : Team Management/Team Captain Why ? : As they would want to know what are the relevant areas that they would need to improve upon to win matches at a particular stadium.As they now know what aspects of game contribute more to winning match at that stadium.

Milestone-4

Member Name : SIDDHARTH DEWANGAN
Roll Number : MT2016133

1. CLUSTERING REPORT

Target User :- Team Management/Captain/Owner

Algorithm : Kmeans

1.Problem Formulation:
To build clusters of all the Bowlers that have bowled during the Death overs (16-20) according to their average ,strike-rate,economy. Three of the different graphs have been plotted taking two at a time these three features. Also on hovering on each of the data point, information is shown such as the value of the feature and the name of the bowler as shown below in figure.

2.Data Preparation:
1.We first created a data frame clus_one that contains total runs given, total wickets taken and total balls bowled at the death over.
2. Then created data frame clus_two that has features of average,economy and strike-rate calculated from clus_one.

R script for data preparation:-

```
matches <- read_csv("matches.csv")

deliveries <- read_csv("deliveries.csv")

clus_one <- sqldf("select sum(total_runs) as tr,sum(is_super_over) as
tw,bowler,count(ball) as tb from mergeddata where over>15 group by bowler")

clus_two <-sqldf("select tr*1.0/tw as Average,tb*1.0/tw as Strike_Rate,(tr*6)/tb
as Economy,Bowler from clus_one")

for(i in 1:length(clus_one$bowler)){
  if(is.na(clus_two$Average[i]))
    clus_two$Average[i]=clus_one$tr[i]
  if(is.na(clus_two$Strike_Rate[i]))
    clus_two$Strike_Rate[i]=clus_one$tb[i]
}
```

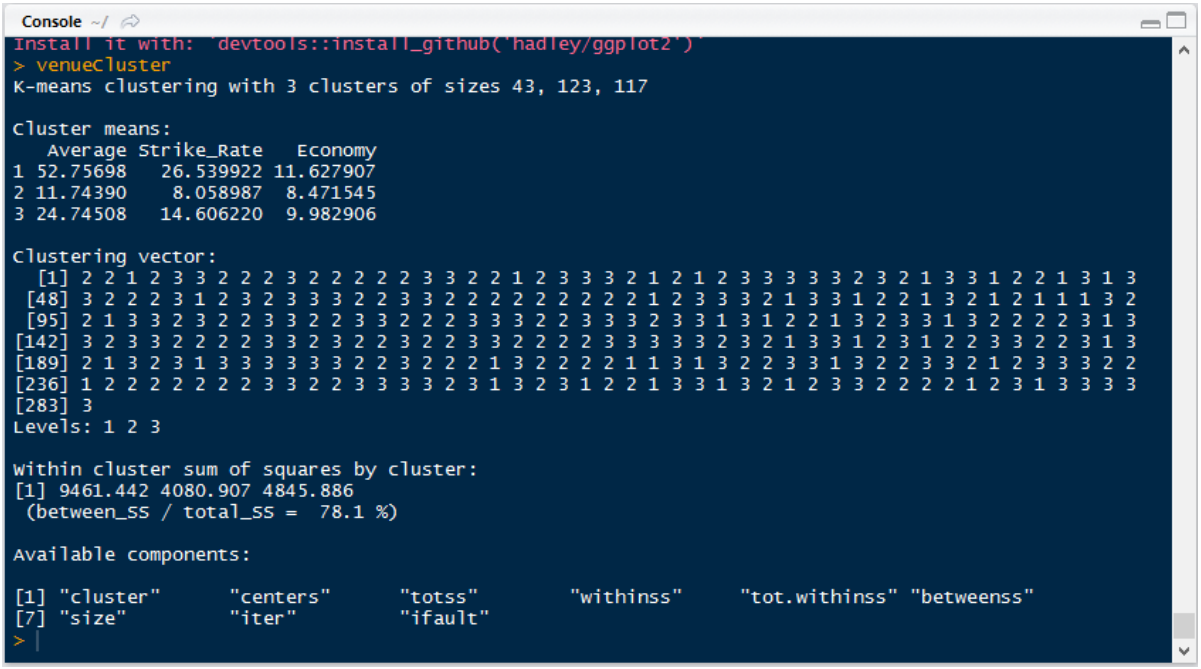
3. R-Code for clustering :

1. Using kmeans algorithm with 3 parameters of Average, Strike-rate, Economy and 3 clusters taken.
2. Using ggplot for plotting the clusters and label=bowler that is used for hovering and shows bowler names.

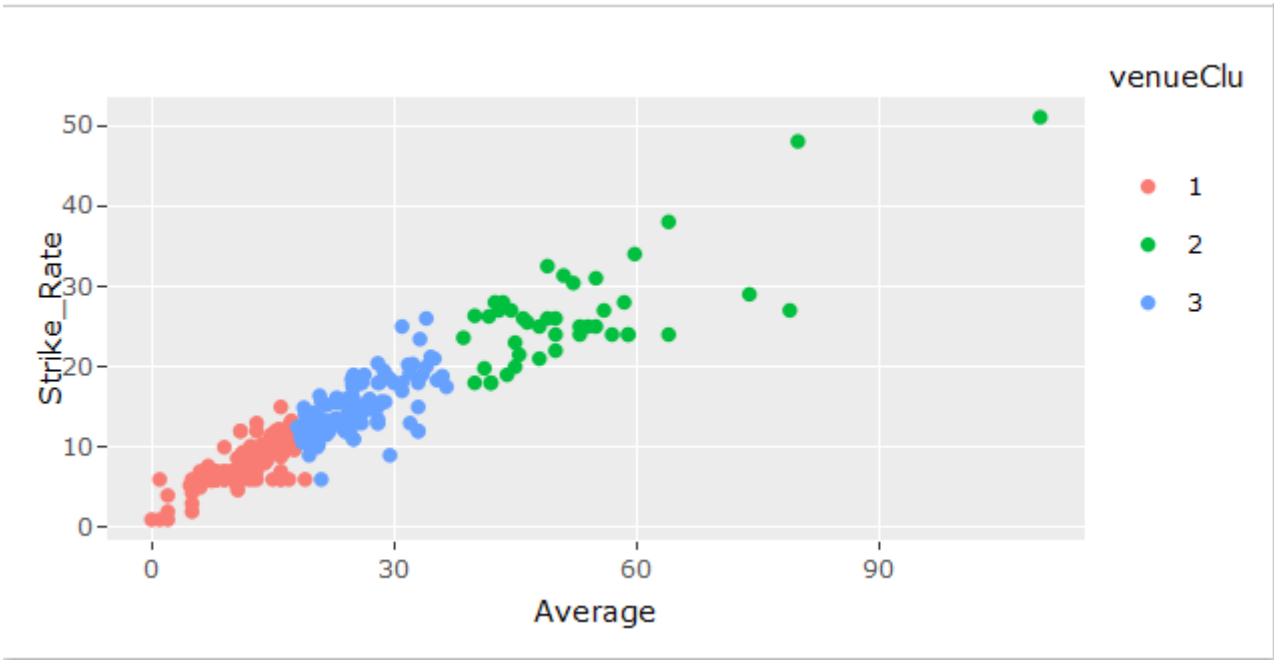
```
library(ggplot2)
attach(clus_two)
venueCluster <- kmeans(clus_two[,1:3 ], 3, nstart = 15)
venueCluster$cluster <- as.factor(venueCluster$cluster)
#irisCluster$centers
#nrow(df3)
library("plotly")
p <- ggplot(clus_two, aes(Average,Strike_Rate,label = bowler ,color =
venueCluster$cluster)) + geom_point()
ggplotly(p)
venueCluster
```

4.Evaluation Of Model

1. In the below figure various means are given for all the 3 clusters that we are using.
2. Clustering vector is also shown with each bowler assigned any of 3 clusters.
3 There are total of 283 bowlers out of which each cluster has size : 43, 123, 117.



Above fig: Clustering of bowlers for Economy and Average



Above fig: Clustering of bowlers for Strike-rate and Average

5.Target User : Team Management/Captain/Owner :

By the help of this clustering figures the team management knows the potential of each and every bowler of their team on the features of average, economy and strike-rate. They can choose which bowler to select by sacrificing in one feature and gaining in another.

2.ASSOCIATION REPORT

Target User :- Team Management/Captain

Algorithm : Apriori

1.Problem Formulation: According to the parameters such as wide balls, leg-byes and no balls, we find out whether the team looses a match. With the combination of different parameters we figure out which combination has its major contribution towards losing of a team.

2.**Data Preparation:**

1. We first formed different data frames according to the total wide runs given, total byes given and total no balls given.
2. Then we merge all these data frames with the result of the match as well.
3. We assign either 0/1 to the data frames according to the fact as, if the team has conceded wide balls greater than 4 then assign 1 otherwise 0.
4. Similarly repeat step 3 for all the parameters and form a single data frame.

R Script for data preparation:

```
library("sqldf")

mergeddata2 <- sqldf("select * from matches,deliveries where
matches.id=deliveries.match_id")

asso_one <- sqldf("select match_id,sum(wide_runs) as total_wide from
mergeddata2 where bowling_team=\"Mumbai Indians\" group by match_id")

for(i in 1:length(asso_one$total_wide)){
  if(asso_one$total_wide[i]>=5)
    asso_one$total_wide[i]=1
```

```
  else
    asso_one$total_wide[i]=0
}

asso_two <- sqldf("select match_id,sum(bye_runs) as total_bye from
mergeddata2 where bowling_team=\"Mumbai Indians\" group by match_id")

for(i in 1:length(asso_two$total_bye)){
  if(asso_two$total_bye[i]>=1)
    asso_two$total_bye[i]=1
  else
    asso_two$total_bye[i]=0
}



asso_three <- sqldf("select match_id,sum(noball_runs) as total_no_ball from
mergeddata2 where bowling_team=\"Mumbai Indians\" group by match_id")

for(i in 1:length(asso_three$total_no_ball)){
  if(asso_three$total_no_ball[i]>=1)
    asso_three$total_no_ball[i]=1
  else
    asso_three$total_no_ball[i]=0
}

asso_four <- sqldf("select match_id,winner as result from mergeddata2 where
bowling_team=\"Mumbai Indians\" group by match_id")

for(i in 1:length(asso_four$result)){
  if(asso_four$result[i]=="Mumbai Indians")
    asso_four$result[i]=1
  else
    asso_four$result[i]=0
}

merge_asso_two <- sqldf("select asso_one.match_id,total_wide as
total_wide_greater_four,total_bye as total_bye_greater_zero, total_no_ball as
total_no_ball_greater_zero, result from asso_one natural join asso_two natural
join asso_three natural join asso_four")

merge_asso_two$match_id <- NULL
```

```
write.csv(merge_asso_two,file="merge_asso_two.csv")
```

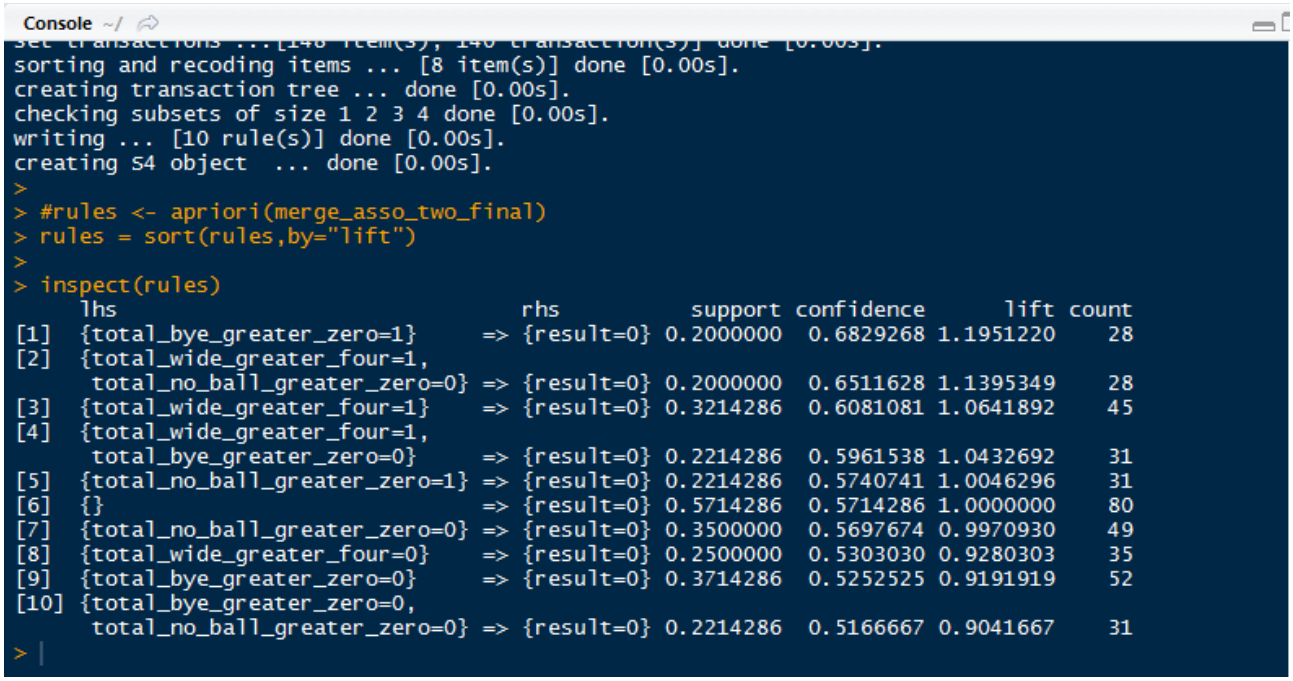
3. R-Code for performing Association:

1. Included library for arules.
2. Performed apriori function on the data frame.
3. Used only those rules which had result as loss(0) on the rhs.

```
library(arules)
#rules<-apriori(merge_asso_one)
#summary(rules)

merge_asso_two_final <- read.csv("merge_asso_two.csv",header=T,colClasses
= "factor")

rules <- apriori(merge_asso_two_final,parameter=list(supp=.2,conf=0.5),
              appearance=list(rhs=c("result=0"), default="lhs"))

#rules <- apriori(merge_asso_two_final)
rules = sort(rules,by="lift")

inspect(rules)
```

4.Evaluation Model

1. In the below figure we notice that we have 10 rules associated with the data frame that we formulated.
2. We sorted the rules according to the lift value and we get that whenever total bye runs are greater than equal to one teams lose with lift value of 1.19 and confidence .68.



5.Target User : Team Management/Captain

With the data above, the management and team can strategies on the various factors that can lead to loss of their team, avoid them and lead them to success.

Milestone-4

Member Name : RAJU RAGHUWANSHI & SUBHAM KANDOI & VIJAY AGARWAL

1. CLUSTERING REPORT

Target User :- Team Management/Captain/Owner

Algorithm : Kmeans

1.1 Problem Formulation:

To build clusters of total wicket and total sum of each season cross ponding to particular venue (stadium).we take two column (venue , season) to calculate the cluster value.

1.2.Data Preparation:

First we find the total wicket and total run for each season for particular venue.

R script for data preparation:-

```
matches <- read.csv('matches.csv')
deliveries <- read.csv('deliveries.csv')

venue_data <- matches[,c("id","season","venue")]

str(deliveries$player_dismissed)
player_out <- as.numeric(deliveries$player_dismissed)

## wicket-1 else-0
match_data <- cbind(deliveries[,c("match_id","total_runs")], player_out)
match_data$player_out <- ifelse(match_data$player_out > 1, 1, 0)

colnames(venue_data)[1] <- 'match_id'

venue_stats <- merge(venue_data, match_data, by='match_id')
```

1.3. R-Code for clustering :

1. Using kmeans algorithm with 4 parameters of , Session, total wickets ,total run, and venue taken.

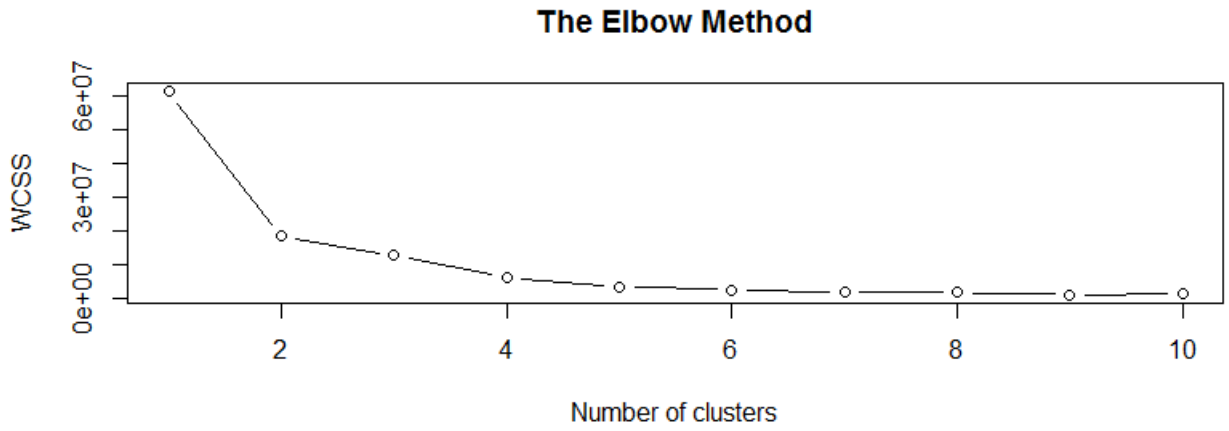
```
## cluster the venue
venue_total_runs <- aggregate(venue_stats$total_runs, list(venue_stats$venue, venue_stats$season), sum)
colnames(venue_total_runs)[1] <- 'venue'
colnames(venue_total_runs)[2] <- 'season'
colnames(venue_total_runs)[3] <- 'total_runs'

total_wickets <- aggregate(venue_stats$player_out, list(venue_stats$venue, venue_stats$season), sum)
colnames(total_wickets)[1] <- 'venue'
colnames(total_wickets)[2] <- 'season'
colnames(total_wickets)[3] <- 'total_wickets'

venue_season_clustering <- merge(venue_total_runs, total_wickets, by=c('venue','season'))

dataset = venue_season_clustering[3:4]
```

1.4.Evaluation Of Model

```
> kmeans
K-means clustering with 3 clusters of sizes 33, 24, 45

Cluster means:
  total_runs total_wickets
1  1522.7576      58.90909
2   693.7083      27.70833
3  2413.9556      91.51111

Clustering vector:
  [1] 2 2 2 3 2 1 2 2 1 1 3 2 2 1 3 1 3 1 3 3 1 3 3 1 3 1 3 3 1
 [30] 1 3 2 2 2 2 2 2 2 1 2 3 3 3 1 3 3 3 3 3 3 3 3 3 3 1 1 1
 [59] 3 1 2 3 3 1 1 1 1 2 2 3 3 1 3 1 1 3 1 1 1 1 3 2 1 3 3 2 2
 [88] 2 1 1 3 3 3 3 2 1 3 3 3 3 3 1

Within cluster sum of squares by cluster:
[1] 2619302.8  951571.9 9222553.2
 (between_SS / total_SS =  79.1 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"
[5] "tot.withinss" "betweenss"    "size"         "iter"
[9] "ifault"
```

1.5. ELBOW METHOD

```
#Elbow Method
wcss = vector()
for (i in 1:10) wcss[i] = sum(kmeans(dataset, i)$withinss)
plot(1:10,
    wcss,
    type = 'b',
    main = paste('The Elbow Method'),
    xlab = 'Number of clusters',
    ylab = 'WCSS')
```



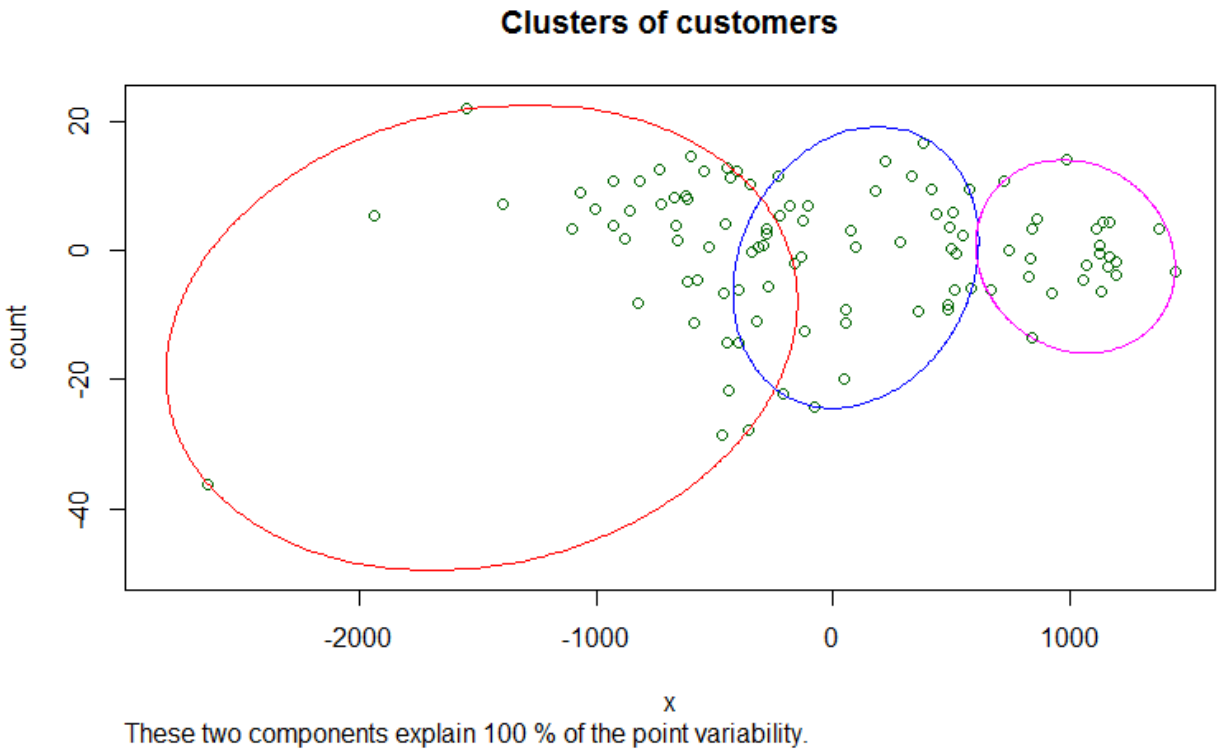
1.6 TABLE

Filter

	venue	season	total_runs	total_wickets	cluster
1	Barabati Stadium	2010	666	31	2
2	Barabati Stadium	2012	612	21	2
3	Barabati Stadium	2014	1000	28	2
4	Brabourne Stadium	2010	2345	81	3
5	Brabourne Stadium	2014	349	11	2
6	Brabourne Stadium	2015	1148	35	1
7	Buffalo Park	2015	799	38	2
8	De Beers Diamond Oval	2009	897	39	2
9	Dr DY Patil Sports Academy	2008	1138	50	1
10	Dr DY Patil Sports Academy	2010	1675	84	1
11	Dr DY Patil Sports Academy	2011	1997	82	3
12	Dr. Y.S. Rajasekhara Reddy ACA-VDCA Cricket Stadium	2012	592	30	2
13	Dr. Y.S. Rajasekhara Reddy ACA-VDCA Cricket Stadium	2015	865	29	2
14	Dr. Y.S. Rajasekhara Reddy ACA-VDCA Cricket Stadium	2016	1664	75	1
15	Dubai International Cricket Stadium	2014	2064	79	3
16	Eden Gardens	2008	1843	83	1
17	Eden Gardens	2010	2167	70	3
18	Eden Gardens	2011	1854	72	1
19	Eden Gardens	2012	2012	76	3
20	Eden Gardens	2013	2304	99	3
21	Eden Gardens	2014	1289	44	1
22	Eden Gardens	2015	2386	87	3

owing 1 to 23 of 102 entries

1.7 CLUSTERING



1.8. Target User : Team Management/Captain/Owner :
By the help of this clustering figures the team management knows whether any stadium is batting or bowling and according to that they select their team that number of bowler is more or batsman is more.

2.ASSOCIATION REPORT

Target User :- Team Management/Captain

Algorithm : Apriori

2.1. Problem Formulation: According to the parameters such as Toss_decision, Toss_winner and total_runs, we find out whether the team (Chennai super Kings) win a match or not . With the combination of different parameters we figure out which combination has winning chance is more.

2.**Data Preparation:**

1. We first formed different data frames according to the total runs given, toss_winner given and toss_decision given.
2. Then we merge all these data frames with the result of the match as well.
3. We assign either 0/1 to the data frames according to the fact as, if the team has win the match then assign 1 otherwise 0.
4. Similarly repeat step 3 for all the parameters and form a single data frame.

R Script for data preparation:

```
matches <- read.csv('matches.csv')
deliveries <- read.csv('deliveries.csv')

df1 <- subset(matches, team1 == 'Chennai Super Kings' | team2 == 'Chennai Super Kings')
#dataframe1 preparaion

dataframe1 <- df1[,c('id','team1', 'team2','toss_winner', 'winner')]
dataframe1$winner <- ifelse(dataframe1$winner == 'Chennai Super Kings', 1, 0)
dataframe1$toss_winner <- ifelse(dataframe1$toss_winner == 'Chennai Super Kings', 1, 0)
dataframe1 <- dataframe1[,c('id','toss_winner', 'winner')]

#dataframe2 prepration
dataframe2 <- df1[,c('id','team1', 'team2','toss_decision','toss_winner', 'winner')]
dataframe2$winner <- ifelse(dataframe2$winner == 'Chennai Super Kings', 1, 0)
dataframe2$toss_winner <- ifelse(dataframe2$toss_winner == 'Chennai Super Kings', 1, 0)
dataframe2$toss_decision <- ifelse(dataframe2$toss_decision == 'bat'& dataframe2$toss_winner==1, 1, 0)
dataframe2 <- subset(dataframe2, toss_winner == 1)
dataframe2 <- dataframe2[,c('id','toss_decision', 'winner')]
```

```
#dataframe3 preparation
df2 <- subset(deliveries, batting_team == 'Chennai Super Kings')
total <- aggregate(df2$total_runs, list(df2$match_id), sum)
total1 <-dataframe2[,c('id','winner')]
colnames(total)[1] <- 'id'
colnames(total)[2] <- 'total_run'
dataframe3<- merge(total,total1,by ="id")
dataframefinal<- merge(dataframe3,dataframe2,by ="id")
dataframefinal <- dataframefinal[,c('id','toss_decision','total_run',
'winner.y')]
dataframefinal<- merge(dataframefinal,dataframe1,by ="id")
dataframefinal <- dataframefinal[,c('id','toss_decision','total_run',
'winner','toss_winner')]

cat_runs <- ifelse(dataframefinal$total_run>180, 'high',
ifelse(dataframefinal$total_run > 145 & dataframefinal$total_run<=180,
'medium', 'low'))
dataframefinal <- cbind(dataframefinal, cat_runs)

dataframefinal <- dataframefinal[,c(1,2,5,6,4)]

rownames(dataframefinal) <- dataframefinal$id

dataframefinal <- dataframefinal[,-1]

dataframefinal$toss_decision <- as.factor(dataframefinal$toss_decision)
dataframefinal$toss_winner <- as.factor(dataframefinal$toss_winner)
dataframefinal$cat_runs <- as.factor(dataframefinal$cat_runs)
dataframefinal$winner <- as.factor(dataframefinal$winner)

transactions <- as(dataframefinal, "transactions")
```

3. R-Code for performing Association:

1. Included library for arules.
2. Performed apriori function on the data frame.
3. Used only those rules which had result as loss(0) on the rhs.

```
library(arules)
rules = apriori(transactions, parameter=list(support=0.06, confidence=0.8),
appearance = list(default="lhs",rhs="0"))
```

```
rules<-sort(rules, decreasing=TRUE,by="lift")
inspect(rules)

rules_frame <- as(rules, "data.frame")
write.csv(rules_frame, 'winning Association Rules.csv')
```

4.Evaluation Model

In the below figure we notice that we have 8 rules associated with the data frame that we formulated.

	rules	support	confidenc
5	{toss decision=1,cat runs=medium}=>{winner=1}	0.272727	0.818182
8	{toss decision=1,toss winner=1,cat runs=medium}=>{winner=1}	0.272727	0.818182
2	{cat runs=medium}=>{winner=1}	0.363636	0.75
3	{toss decision=1,cat runs=high}=>{winner=1}	0.136364	0.75
6	{toss winner=1,cat runs=medium}=>{winner=1}	0.363636	0.75
7	{toss decision=1,toss winner=1,cat runs=high}=>{winner=1}	0.136364	0.75
1	{cat runs=high}=>{winner=1}	0.181818	0.705882
4	{toss winner=1,cat runs=high}=>{winner=1}	0.181818	0.705882

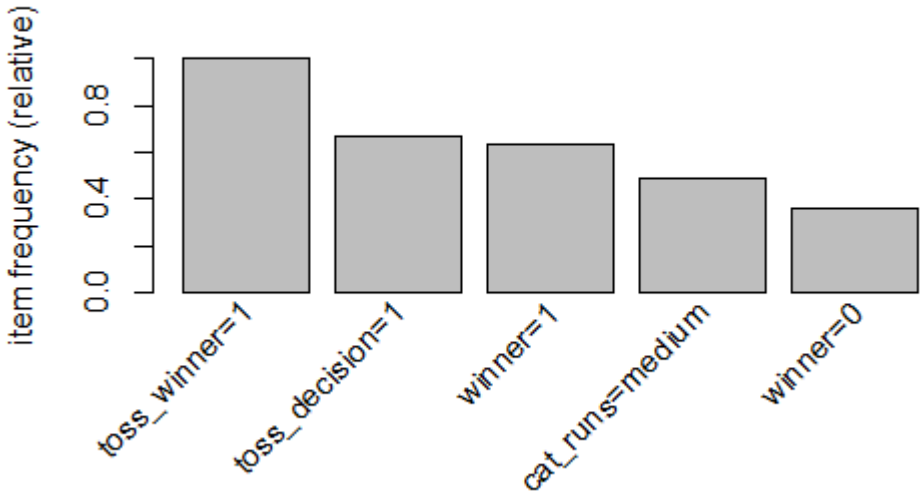
5.Target User : Team Management/Captain

With the data above, the management and team can strategies on the various factors that can lead them to success.

6. Frequency Plot

##Item Frequencies and plot

itemFrequency(transactions)
itemFrequencyPlot(transactions, topN = 5)



MILESTONE-4

Member Name : SUBHAM KANDOI & RAJU RAGHUWANSHI & VIJAY AGARWAL

1. CLUSTERING REPORT

Target User :- Captains/Bookies

Algorithm : Kmeans

1.1.Problem Formulation: To build clusters according to average run and average wicket for each over and also have one column that show the cluster value for each over cross ponding to average wicket and average run.

1.2. Data Preparation:
1.We first calculate the total number of wickets for each over by putting zero and one if wicket is fall then put one otherwise put zero for each ball of each over.

R script for preparation:-

```
matches <- read_csv("matches.csv")
deliveries <- read_csv("deliveries.csv")

str(deliveries$player_dismissed)
player_out <- as.numeric(deliveries$player_dismissed)


## wicket-1 else-0
match_data <-
cbind(deliveries[,c("match_id","inning","over","ball","total_runs")],
player_out)
match_data$player_out <- ifelse(match_data$player_out > 1, 1, 0)

## cluster the overs
avg_runs <- aggregate(match_data$total_runs, list(match_data$over),
mean)
colnames(avg_runs)[1] <- 'over'
colnames(avg_runs)[2] <- 'avg_runs'
```

avg_wickets <- aggregate(match_data$player_out, list(match_data$over), mean)
colnames(avg_wickets)[1] <- 'over'
colnames(avg_wickets)[2] <- 'avg_wickets'

over_data <- merge(avg_runs, avg_wickets, by='over')

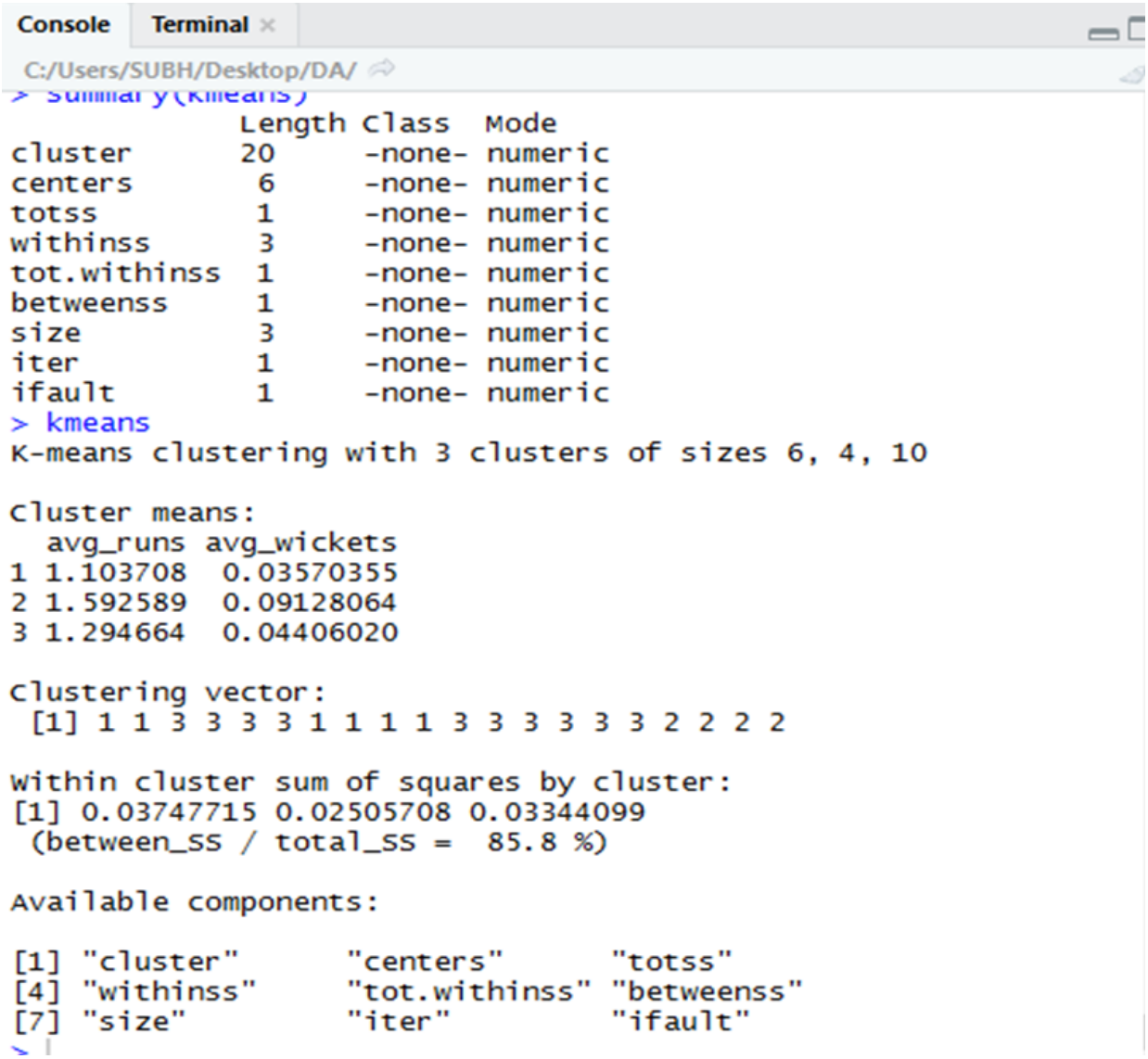
1.2. Elbow method

```
wcss = vector()
for (i in 1:10) wcss[i] = sum(kmeans(dataset, i)$withinss)
plot(1:10,
    wcss,
    type = 'b',
    main = paste('The Elbow Method'),
    xlab = 'Number of clusters',
    ylab = 'WCSS')
```

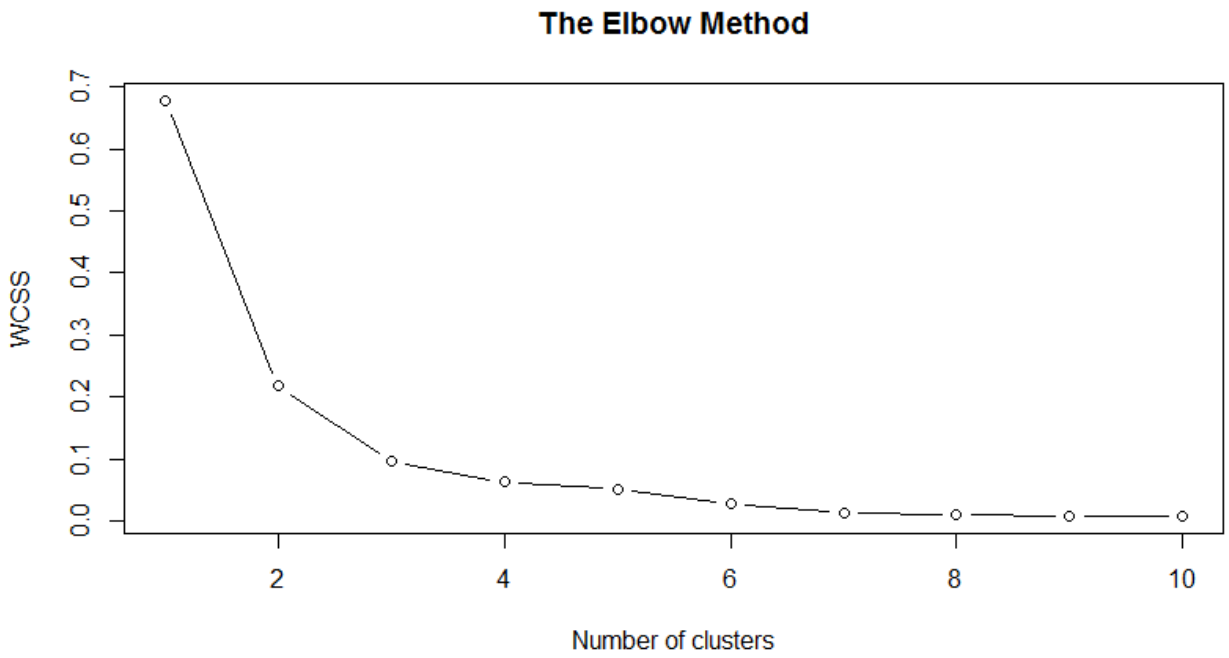
1.3. R-Code for kmeans and plotting :

```
# Perform KMeans
kmeans = kmeans(x = dataset, centers = 3)
y_kmeans = kmeans$cluster

# Plot KMeans result
library(cluster)
clusplot(dataset,
        y_kmeans,
        lines = 0,
        shade = TRUE,
        color = TRUE,
        labels = 2,
        plotchar = FALSE,
        span = TRUE,
        main = paste('Clusters of customers'),
        xlab = 'x',
        ylab = 'count')
```

1.4. Evaluation Of Model



1.5 ELBOW METHOD



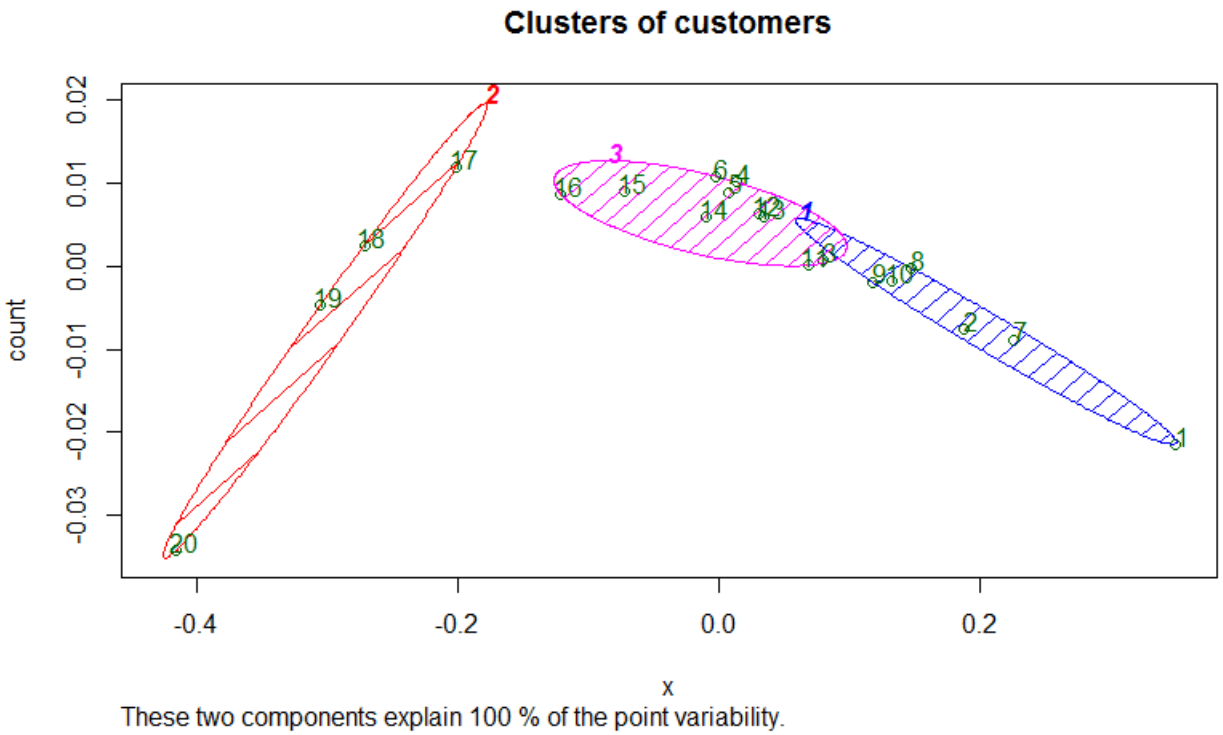
1.6. TABLE

arules.R | cluster1.R | over_data | venue_

Filter

	over	avg_runs	avg_wickets	cluster
1	1	0.9465722	0.03216574	3
2	2	1.1092483	0.03696578	3
3	3	1.2174517	0.04112825	1
4	4	1.2840798	0.03934701	1
5	5	1.2902	1.2840798 04125874	1
6	6	1.3009382	0.04061056	1
7	7	1.0715893	0.03389592	3
8	8	1.1502964	0.03429297	3
9	9	1.1798469	0.03939909	3
10	10	1.1646941	0.03750178	3
11	11	1.2289572	0.04309107	1
12	12	1.2667340	0.04111368	1
13	13	1.2632798	0.04110580	1
14	14	1.3075461	0.04635858	1
15	15	1.3695106	0.05027355	1
16	16	1.4179306	0.05631476	1
17	17	1.4982568	0.06230105	2
18	18	1.5669762	0.07964741	2
19	19	1.5994945	0.09064869	2
20	20	1.7056294	0.13252541	2

1.7. Output: plot of Clusters



Explanation of Output:
Here We have total three cluster first cluster belong between over(1-6), second cluster belong between over(7-15) and third cluster belong between over (16-20).wicket is fallen more in first cluster and third cluser and run is also more during these overs and in the middle over i:e second cluster wickets is not fallen so much and run also not so much .

1.8.Target User : Team Captain/Bookies Why ? : it helps captains visually see if number of wicket going in first cluster is less then in middle over bastman hit fast according to the situation captain send the batmans like heaters or not.

Bookies: According to the run or wickets bookie give the final target for the team .

2.ASSOCIATION RULES

Target User : Team Management/Captain

Algorithm : Apriori

2.1.Problem Formulation: To formulate rules that help find association for Royal challenger Bangalore (RCB) that which player play the match and then RCB win that match or not (Group of players ->win the match or not).

2.2.Data Preparation:

1.We find number of match played by RCB whether RCB bat first or last .
2.According to above collected data we made new dataframe having columns which player play the match in RCB according to match_id.
3.We have one column result if RCB win then that column contain 1 otherwise 0.

R Script for Data Preparation

```
matches <- read.csv('matches.csv')
deliveries <- read.csv('deliveries.csv')

matches <- matches[,c('id','team1', 'team2', 'winner')]

matches <- subset(matches, team1 == 'Royal Challengers Bangalore' | team2 == 'Royal Challengers Bangalore')

matches$winner <- ifelse(matches$winner == 'Royal Challengers Bangalore', 1, 0)

matches <- matches[,c('id','winner')]

batsman <- subset(deliveries, deliveries$batting_team=='Royal Challengers Bangalore')[,c('match_id','batsman','non_striker')]
bowler <- subset(deliveries, deliveries$bowling_team=='Royal Challengers Bangalore')[,c('match_id','bowler')]

players <- unique(bowler[c('match_id','bowler')])

players_batsman <- unique(batsman[c('match_id','batsman')])
players_non_striker <- unique(batsman[c('match_id','non_striker')])
colnames(players)[2] <-'players'
```

```
colnames(players_non_striker)[2] <-'players'
colnames(players_batsman)[2] <-'players'
mergedData <- rbind(players,players_batsman)
mergedData1 <- rbind(mergedData,players_non_striker)
FinalData <- unique(mergedData1[c('match_id','players')])
#playersnames<- FinalData[2]
#uniqueplayers <- unique(playersnames[c('players')])

colnames(matches)[1] <- 'match_id'
colnames(matches)[2] <- 'players'
```

2.3. R-Code for algorithm and plotting :

```
library(arules)
rules = apriori(transactions, parameter=list(support=0.06, confidence=0.8),
appearance = list(default="lhs",rhs="0"))
rules<-sort(rules, decreasing=TRUE,by="lift")
inspect(rules)

rules_frame <- as(rules, "data.frame")
write.csv(rules_frame, 'winning Association Rules.csv')
```

2.4. Evaluation Model

2.4.1. Winning Association Rules

rules	support	confidenc	lift	count
{CH Gayle	0.086331	0.8	1.588571	12
{CH Gayle	0.086331	0.8	1.588571	12
{A Mithun	0.064748	0.75	1.489286	9
{CH Gayle	0.107914	0.75	1.489286	15
{A Mithun	0.064748	0.75	1.489286	9
{CH Gayle	0.107914	0.75	1.489286	15
{CH Gayle	0.064748	0.75	1.489286	9
{CH Gayle	0.064748	0.75	1.489286	9
{B Akhil,R	0.064748	0.692308	1.374725	9
{S Aravinc	0.064748	0.692308	1.374725	9
{CH Gayle	0.064748	0.692308	1.374725	9
{S Aravinc	0.064748	0.692308	1.374725	9
{CH Gayle	0.064748	0.692308	1.374725	9
{TM Dilsha	0.093525	0.684211	1.358647	13
{TM Dilsha	0.093525	0.684211	1.358647	13
{TM Dilsha	0.115108	0.666667	1.32381	16
{TM Dilsha	0.115108	0.666667	1.32381	16
{B Akhil}=	0.064748	0.642857	1.276531	9
{CH Gayle	0.122302	0.62963	1.250265	17
{CH Gayle	0.122302	0.62963	1.250265	17
{CH Gayle	0.122302	0.62963	1.250265	17
{CH Gayle	0.122302	0.62963	1.250265	17
{S Aravinc	0.122302	0.607143	1.205612	17
{S Aravinc	0.122302	0.607143	1.205612	17
{CH Gayle	0.064748	0.6	1.191429	9
{CH Gayle	0.064748	0.6	1.191429	9

Explanation: Here we get 26 association rules .The first rule which has highest “confidence” and “support” says that if RCB has dilshan , zaheer khan,gayl then RCB win has more chance then loss.
Similarly other rules suggest different association between columns for winning.

2.4.1. Lossing Association Rules

	rules	support	confidenc	lift	count
3	{CL White	0.071942	0.909091	1.831357	10
2	{CL White	0.064748	0.9	1.813043	9
8	{A Kumble	0.064748	0.9	1.813043	9
1	{CL White	0.079137	0.846154	1.704571	11
4	{A Kumble	0.071942	0.833333	1.678744	10
10	{JH Kallis,	0.071942	0.833333	1.678744	10
12	{A Kumble	0.071942	0.833333	1.678744	10
5	{CL White	0.064748	0.818182	1.648221	9
6	{P Kumar,	0.064748	0.818182	1.648221	9
7	{Z Khan, R	0.064748	0.818182	1.648221	9
9	{P Kumar,	0.064748	0.818182	1.648221	9
11	{JH Kallis,	0.064748	0.818182	1.648221	9
13	{A Kumble	0.064748	0.818182	1.648221	9

Explanation: Here we get 13 association rules .The first rule which has highest “confidence” and “support” says that if RCB has CLWhite, Anil Kumble then RCB losing chance is more as compare to winning chance.
Similarly other rules suggest different association between columns for winning.

2.5.Target User : Team Management/Team Captain Why ? : As they would want to know which combination of team is good to win the match.

2.6. Rules Frequency plot