Predicting the Kabbadi Match Result Using Random Forest.

Abstract:
In this project, we use data mining techniques using R software to answers the questions related to kabaddi dataset. We are aiming to answer the questions mentioned in the pdf we have received along with the dataset. Dataset was manually curated which consists of 66 entries over 31 variables from 33 matches. We provides a quantitative approach to profile an entire tournament to understand a generalized area of strengths of multiple teams.We also fit predictive model and hyphothesis Test to test which Team Has measure Strength And Weakness.

Introduction:
Kabaddi is a contact sport that originated in ancient India. The standard style Kabaddi World Cup, is an indoor international kabaddi competition conducted by the International Kabaddi Federation (IKF), contested by men's and women's national teams. The competition has been previously contested in 2004, 2007 and 2016. All the tournaments have been won by India. The 2016 Kabaddi World Cup, the third standard-style Kabaddi World Cup, was an internationalkabaddi tournament governed by the International Kabaddi Federation, contested from 7 to 22 October 2016 in Ahmedabad, India. Twelve countries had competed in the tournament. 30 league matches were played between the teams. Teams were divided in two pools with six teams in each pool. Top two teams from each pool qualifed for the seminals and winners from seminals played in the _nals. The Kabaddi dataset contains data for all 33 matches at granualirity level of attack, defense, allout and extra points. Data set also includes toss results, super tackle count and all out count along with match results.

A majority of Indians either knows or plays kabaddi. However, the use of technology to better the game is non-existent so far. Kabaddi produces data which is under-utilised today, and in the best form used for showing descriptive statistics. This research then ventures into predictive modelling of the game outcome using supervised learning method.

## Data Preparation
Now, we will split the dataset into train and validation set in the ratio 70:30. We can also create a test dataset, but for the time being we will just keep train and validation set.

```
> Data=read.csv("C:\\Users\\Sharad Deshmukh\\Desktop\\MSC=3\\Data Mining\\Kabaddi.csv")
> LM=Data[1:60,-1]  ##excluding serial number
> library(caret)
> set.seed(100)
> train <- sample(nrow(LM), 0.7*nrow(LM), replace = FALSE)
> TrainSet <- LM[train,]
> ValidSet <- LM[-train,]
```

## MOdel fit
Now, we will create a Random Forest model with default parameters and then we will fine tune the model by changing ‘mtry’. We can tune the random forest model by changing the number of trees (ntree) and the number of variables randomly sampled .

```
> library(randomForest)
> # Create a Random Forest model with default parameters
> model1 = randomForest(as.factor(TrainSet$matchResult)~ ., data = TrainSet, importance = TRUE)
> model1
```

```
Call:
 randomForest(formula = as.factor(TrainSet$matchResult) ~ ., data = TrainSet,      importance = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 5

        OOB estimate of  error rate: 2.38%
Confusion matrix:
         0          1       class.error
0     18          1      0.05263158
1        0        23       0.00000000
```

By default, number of trees is 500 and number of variables tried at each split is 5 in this case. Error rate is 2.38%.

```
> # Find tuning parameters of Random Forest model
> model2 = randomForest(as.factor(TrainSet$matchResult)~ ., data = TrainSet, ntree = 400, mtry = 3, importance = TRUE)
> model2

Call:
 randomForest(formula = as.factor(TrainSet$matchResult) ~ ., data = TrainSet,        ntree = 400, mtry = 3, importance = TRUE)
               Type of random forest: classification
                     Number of trees: 400
No. of variables tried at each split: 3

        OOB estimate of  error rate: 2.38%
Confusion matrix:
     0    1    class.error
0 18  1  0.05263158
1  0 23  0.00000000
```

When we have reduced the mtry to 3 from 5, error rate has not changed. value ,We will now predict on the train dataset first and then predict on validation dataset.

```
> # Predicting on train set
> predTrain <- predict(model2, TrainSet, type = "class")
> # Checking classification accuracy
> table(predTrain, TrainSet$matchResult)

predTrain     0    1
        0          19   0
        1          0   23
> # Predicting on Validation set
> predValid <- predict(model2, ValidSet, type = "class")
> # Checking classification accuracy
> mean(predValid == ValidSet$matchResult)
[1] 1
> table(predValid,ValidSet$matchResult)
```

```
predValid    0   1
       0    11   0
       1     0   7
```

In case of prediction on train dataset, there is zero misclassification; however, in the case of validation dataset we also see zero misclassification and accuracy is about 100% .
We can also use function to check important variables. The below functions show the drop in mean accuracy for each of the variables.

```
> # To check important variables
> (importance(model2))
```

VARIABLES	LOSS	Victory	MeanDecreaseAccuracy	MeanDecreaseGini
team	-1.58113	0.561759	-0.5350718	0.097994
oppTeam	2.120543	0.838075	1.9429147	0.12692069
matchStage	0	0	0	0
tossResult	-1.00125	1.41615	0.529524	0.04268241
alloutRec	5.09838	5.164313	6.4032981	1.23359506
alloutGiv	3.532223	3.179619	4.5354336	0.90920752
sTackleRec	-1.61907	-0.20196	-1.0504892	0.05335243
sTackleGiv	-1.00125	-1.00125	-1.4111369	0.02463996
touchPntsRec	1.032033	2.252214	2.5059614	0.36557925
touchPntsGiv	2.241214	2.81245	3.2522593	0.51405164
bonusPntsRec	-0.76527	1.170172	0.4882076	0.05994589
bonusPntsGiv	1.472778	1.50484	1.7174212	0.1323507
raidPntsRec	2.349158	0.539192	1.4446432	0.49325811
raidPntsGiv	1.557991	1.421682	1.967611	0.28437359
tacklePntsRec	1.798573	2.047277	2.5957773	0.60076223
tacklePntsGiv	3.158739	2.021933	3.4936464	0.48863309
alloutPntsRec	4.360015	4.644517	5.7426743	1.01752891
alloutPntsGiv	3.122291	3.076042	4.6895865	0.89939149
extraPntsRec	0.232511	-0.04149	0.3517116	0.10770676
extraPntsGiv	0.98264	0.668488	1.0839515	0.07822819
totalPntsRec	5.993352	6.557599	7.4356339	1.37418263
totalPntsGiv	4.678876	3.50656	5.1423561	0.74013929
touchPntsDiff	6.347159	5.763676	7.3202791	1.90448476
bonusPntsDiff	0.955807	1.207021	1.2806791	0.0731421
raidPntsDiff	6.512381	5.614481	7.1095051	1.67137919
tacklePntsDiff	6.056646	4.678752	6.4856615	1.4947042
alloutPntsDiff	7.703657	7.957041	9.1018651	2.67785361
extraPntsDiff	0.83015	-0.04421	0.4653695	0.16175279
totalPntsDiff	8.556985	8.216339	8.9834935	2.67620714

Table 1 Variable Importants

```
> (varImpPlot(model2))
```

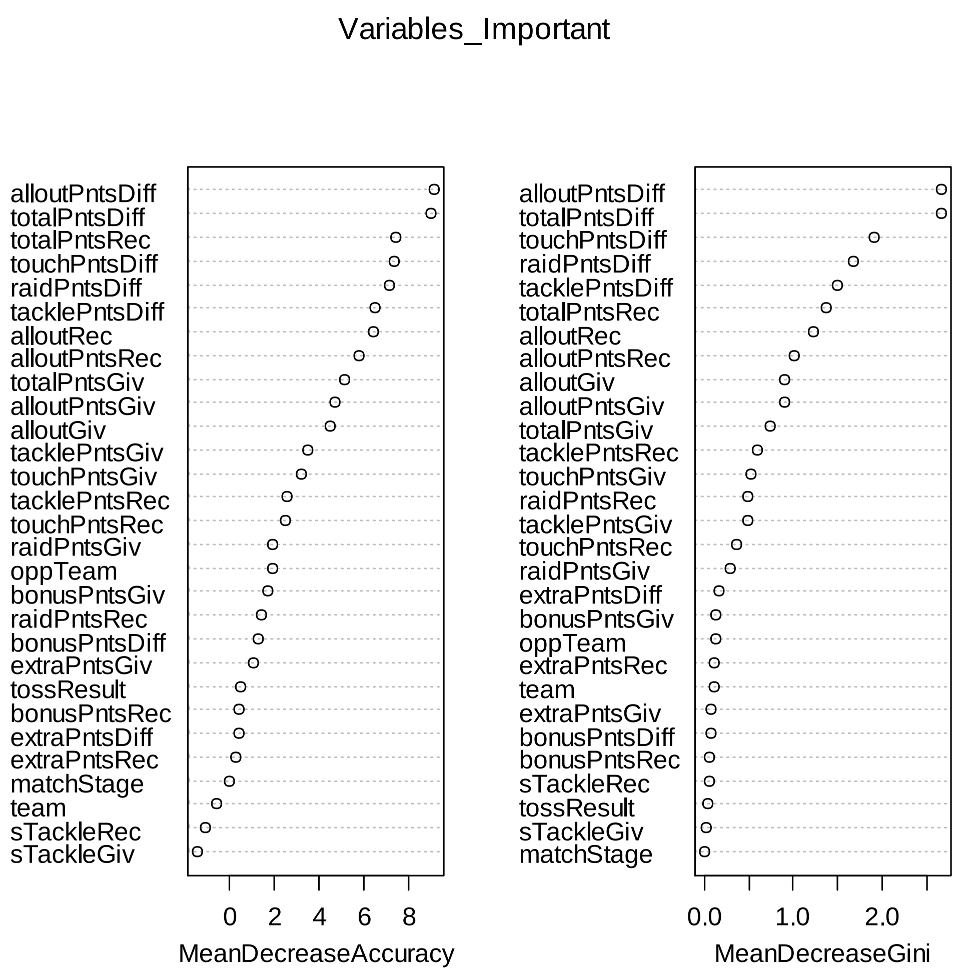

Figure 1 Random Forest Important Variable Plot

From Fig 1 we get plot a plot of importance of variables . The top 6 most Variables which are :

1) alloutPntsDiff_: Difference between the all out points received by the two teams
2) totalPntsDiff_: Difference between the total points received by the two teams
3) totalPntsRec: No. of total points received by team, sum of raid points, tackle points, all out points & extra points
4) touchPntsDiff_: Difference between the touch points received by the two teams
5) raidPntsDiff_: Difference between the raid points received by the two teams
6) tacklePntsDi_: Difference between the tackle points received by the two teams

Q1) (a) Touch, Bonus, Raid points represent strength of the attack and the Tackle, Supertackle points represent strength of the defense. Can we say that one of the two (attack or defense) has a higher role in victory?

Attack	Victory	Mean DecreaseAccuracy	MeanDecreaseGini

touchPntsRec	2.252214	2.5059614	0.36557925
bonusPntsRec	1.170172	0.4882076	0.05994589
raidPntsRec	0.539192	1.4446432	0.49325811

TABLE.2 Attack Variable Importance

Depends	Victory	MeanDecreaseAccuracy	MeanDecreaseGini
tacklePntsRec	2.04727672	2.5957773	0.6007622
sTackleRec	-0.20195839	-1.0504892	0.0533524

Table3. Depends Variables Importance

We draw above tables from Random Forest importunacy of Variables table no.1 .
From comparing the TABLE.2 Attack Variable Importance and Table3. Depends Variables Importance Which gives more weightage to Attack Variables rather than the variables of Depends We can conclude that the **Strength Of attack** is higher Role Than the **strength of the defense.**

b) Which variables contribute the most to victory?
For checking which variables contribute the most to victory we Draw a histogram with the help of excel tool.We extract data from Table No 1 For these our y-axis is points importance of Variables for Victory only and X-axis is Variable names.

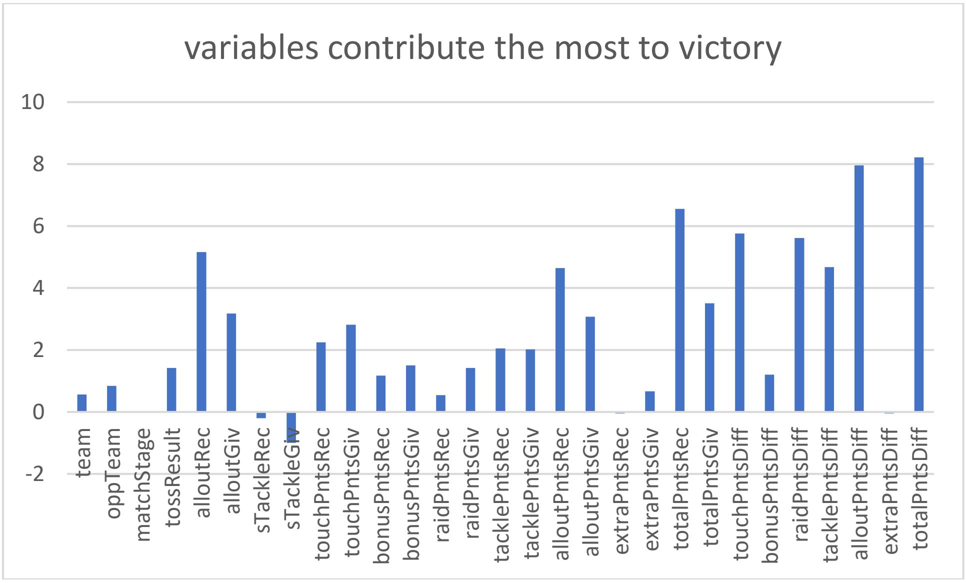

Fig 2 Variables Important to Victory

Here we Observe That the **TotalPntDiff** is the highest Important Variable to contribute for victory then **alloutPntDiff , totalPntsDiff,touchPntsDiff,raidPntsDiff** are respectively higher importants.
sTackleGiv variables shows the negative impact it means it reduces the chances of Victory.

Q2) Do your results change if you use only one row per match (i.e., only 30 observations)? Examine and justify.

```
> #Q2)Data Preparation
> ##taking observation of 30
> odd=seq_len(nrow(LM))%%2      ##create Row indicator
> new_df=LM[odd==1,]            ##Subset of odd rows
> set.seed(100)
> train = sample(nrow(new_df), 0.7*nrow(new_df), replace = FALSE)
> TrainSet = new_df[train,]
> ValidSet = new_df[-train,]
> dim(new_df)
[1] 30 30
> # Predicting on train set
> predTrain = predict(model2, TrainSet, type = "class")
> # Checking classification accuracy
> table(predTrain, TrainSet$matchResult)

predTrain    0   1
       0     9   0
       1     0  12
> # Predicting on Validation set
> predValid = predict(model2, ValidSet, type = "class")
> # Checking classification accuracy
> mean(predValid == ValidSet$matchResult)
[1] 1
> table(predValid,ValidSet$matchResult)

predValid     0   1
       0      3   0
       1      0   6
```

Again For 30 Observation taken only odd data set from original data set we conclude the following
In case of prediction on train dataset, there is zero misclassification; however, in the case of validation dataset we also see zero misclassification and accuracy is about 100% .
We can also use function to check important variables. The below functions show the drop in mean accuracy for each of the variables.

```
>(importance(model2))          ##For 30 observation dataset
```

Variables	Loss	Victory	MeanDecreaseAccuracy	MeanDecreaseGini
team	-1.37523	-1.34467	-1.3560475	0.070344265
oppTeam	1.409048	0.577591	1.1044887	0.064117188
matchStage	0	0	0	0
tossResult	0	0.372169	0.2773768	0.004128788
alloutRec	3.635956	1.448326	3.8592552	0.373628696
alloutGiv	5.235496	4.198017	5.2654368	0.651758274
sTackleRec	0	-1.00125	-1.0012523	0.027962093
sTackleGiv	-1.00125	-1.00125	-1.4119109	0.018961835
touchPntsRec	0.580626	3.433501	3.0960156	0.562195854
touchPntsGiv	1.921168	2.041543	2.3737539	0.2069637
bonusPntsRec	-1.00125	-0.66263	-0.8971594	0.081555542
bonusPntsGiv	-1.72133	-1.38128	-1.6087468	0.03058708
raidPntsRec	3.795457	3.630177	4.441581	0.445525172
raidPntsGiv	2.423275	1.92947	2.561062	0.2353191
tacklePntsRec	1.74514	2.974465	2.834737	0.335252373
tacklePntsGiv	0.906608	2.34287	2.3203746	0.338634539
alloutPntsRec	2.784022	2.207826	3.1318738	0.220095779
alloutPntsGiv	4.831558	4.122511	4.902295	0.639476371

extraPntsRec	1.390096	0.867737	1.3686745	0.037595058
extraPntsGiv	0.820994	0.447325	0.9188318	0.110366023
totalPntsRec	3.188946	3.978404	4.1638438	0.500447969
totalPntsGiv	3.818679	3.522309	4.3548169	0.412268554
touchPntsDiff	6.707845	6.720366	7.1403138	1.14705081
bonusPntsDiff	0.600555	-0.31804	0.1829209	0.063809524
raidPntsDiff	5.094228	4.839414	5.8412388	0.793788337
tacklePntsDiff	3.669482	3.851648	4.1399874	0.541681085
alloutPntsDiff	6.493634	6.428303	6.9306687	0.999863289
extraPntsDiff	0.156179	1.001252	1.0012523	0.023566139
totalPntsDiff	6.323257	6.031031	6.5622226	0.886389899

Table 4 Importance Of Variables For 30 Observation

```
>(varImpPlot(model2))          ##For 30 observation dataset
```

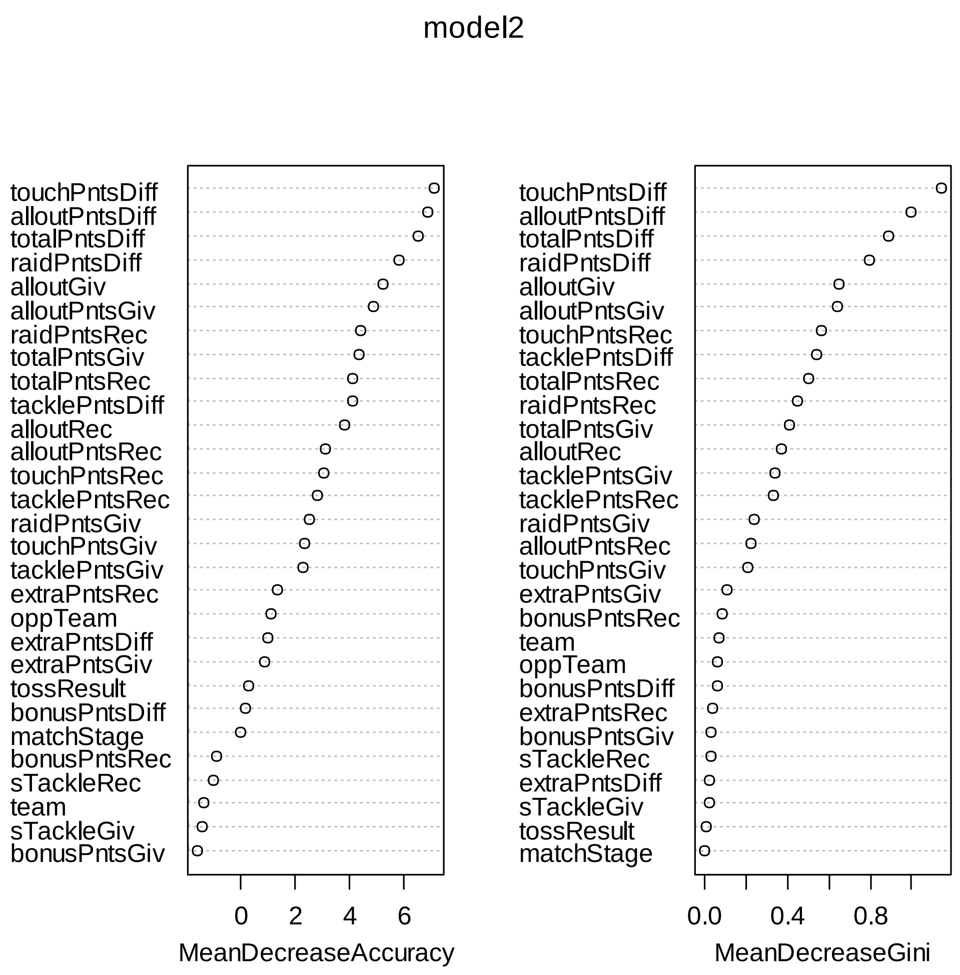

Figure 4 Importance Of Variables Plot For 30 Observation

From Fig 4 we get plot a plot of importance of variables . The top 6 most Variables which are :

1) touchPntsDiff

2) alloutPntsDiff
3) totalPntsDiff
4) raidPntsDiff
5) alloutGiv
6) alloutPntsGiv

these are the Variables which are important to the Match Result.

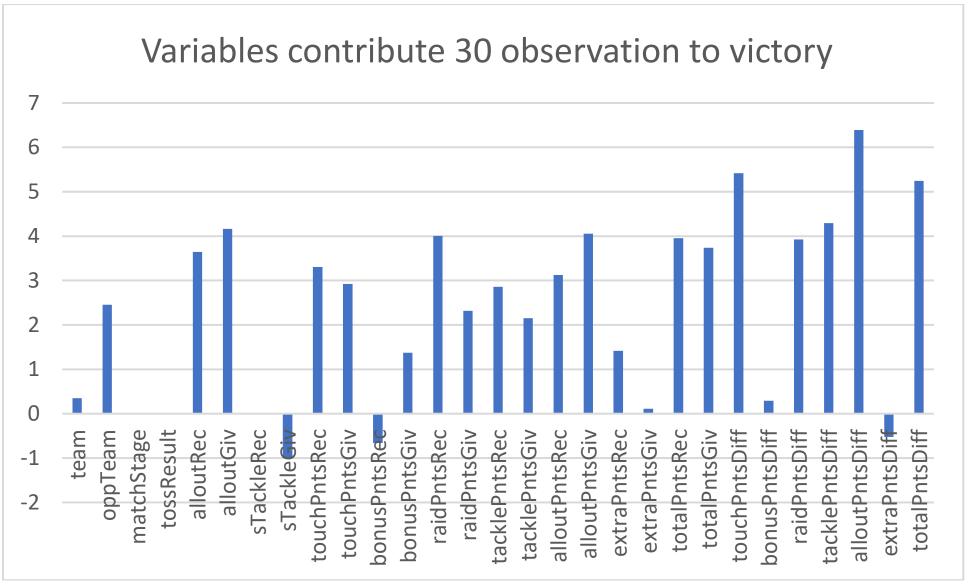

Fig no 5 importance of variables for victory for 30 observation

With the help of excel

Here we Observe That the **alloutPntDiff** is the highest Important Variable to contribute for victory then **touchPntsDiff,totalPntsDiff,tacklePntsDiff** are respectively higher importants.
sTackleGiv variables shows the negative impact it means it reduces the chances of Victory.

##Q3)

A) First predict the winners for the two semi-_finals and then predict the winner for the final match from among the predicted winners for the two semi-finals.
We do check some data processing for Semi Final

```
> ##Data Preparation for semi final
> Win=LM[LM$matchResult==1,]          ##Subset of Win team
> a=as.factor( Win$team)      ###Team winner
> sort(table(a))
a
AUS  ENG  JPN  POL  BAN  KEN  IND   IRN  THA  KOR
 1    2    2    2    3    3    4     4    4    5
> ##From this Table
> ##We choose team for semi-final IND ,IRN ,THA, KOR.
```

No of wins for each team in league matches in two pools

TEAM	WIN	LOSS
KOR	5	0
IND	4	1
BAN	3	2
ENG	2	3
AUS	1	4
ARG	0	5

TEAM	WIN	LOSS
IRN	4	1
THA	4	1
KEN	3	2
JPN	2	3
POL	2	3
USA	0	5

Table No 6.

The highlighted teams from both the pools are qualified for semifinals.
On the basic of above analysis we do some have Idea
Of Teams Vs Opponent Team we get some idea about from Table No 6

Lets have KOR VS IRN first match and second IND Vs THA from that who will win will going to the Final.

```
>korea_index=which((LM$team)=="KOR")
> korea=LM[c(korea_index),]
> summary(korea)        ###summary of Korea matches
```

##Semi Final Matches

##Iran

```
> iran_index=which((LM$team)=="IRN")
> iran=LM[c(ind_index),]
> summary(iran)        ###summary of iran matches
```

2nd Match:

#INDIA

```
> ind_index=which((LM$team)=="IND")
> ind=LM[c(ind_index),]
> summary(ind)        ###summary of indian matches
```

##THALAND

```
> ind_index=which((LM$team)=="IND")
> ind=LM[c(ind_index),]
>summary(ind)
```

team	alloutPntsDiff	totalPntsDiff	touchPntsDiff	raidPntsDiff	tacklePntsDiff	touchPntsRec
IND	7.6	34.8	12.6	15	10	23.4
Iran	7.6	34.8	12.4	15	10.8	23.4
Thailand	3.2	16.4	9.6	9.8	0.2	21
Korea	4.4	21.6	11	13	2.6	25

On the Basis of above Summary plot we also conclude
Looking at the important of mean of the above variables we can conclude that Indias average points are better than the Thailand ,on the basis of that average points we can conclude india will move to final.
Ans Also for IRAN VS Korea we can conclude Iran will go to final.

```
> predValid <- predict(model2,Data[61:64,-1], type = "class")  ##using  variables from this rows 61:64
> Data$matchResult[61:64]
[1] 0 1 1 0
```

```
> Data$team[61:64]
[1] "KOR" "IRN" "IND" "THA"
> predValid
61   62  63    64
 0    1    1      0
Levels: 0 1
```

Similary we get in R by random Forest model prediction we get same result.

B) **Your prediction is supposed to come before the matches. Hence, you cannot use the given data for semi-_nal/_nal. You have to compare the strengths of the two competing teams computed using the league matches on the basis of the model that you _tted for those league matches.?**

Similarly we get from this table and we can say
From the Average points we conclude that Iran have better chances in comparison to india for winning the league.