Cricket-Insights Project Deliverable Predictive Analytics using Classification

Abhinav Chawla(IMT2013002)

Aditya Naidu(IMT2013003)

Shivam Kumar(IMT2013042)

Siddartha Padhi(IMT2013043)

1 Introduction

This document presents the results that we obtained for predictive analysis using classification on our dataset. We used three main classification algorithms on our dataset with several parameters and analyzed the accuracy for each using a confusion matrix. The algorithms that we experimented with are as follows:

- Random Forest
- Decision Trees
- Support Vector Machine

Overall, average accuracy that we received for all the algorithms was around 75% with SVM reaching the maximum accuracy of 82.69%. In the subsequent sections, we present the details of the dataset used for training and testing these algorithms and the results that we obtained.

2 Dataset

Our raw data consists of 577 yaml files where each file contains ball-by-ball details of IPL matches. The data was scraped from CricInfo for all the IPL seasons i.e. 2008-2016. We decided to divide the data in every match into three

sections namely segment-one(0-6 overs), segment-two(7-13 overs) and segment-three(14-20) overs. For each segment, one csv file was maintained containing the following data extracted using python scripts:

- Match_ID: Unique identifier for a match
- Venue: Location of play
- Team1: Team which had to bat first
- Team2: Team which had to bat second
- MR: Runs scored by Team1 that segment
- OR: Runs scored by Team2 in that segment
- MRN: Run rate of Team1 in that segment
- ORN: Run rate of Team2 in that segment
- MW: Wickets lost by Team1 in that segment
- OW: Wickets lost by Team2 in that segment
- Toss_win: Winner of toss for the match
- Win_Bat_First: A binary column, where a 1 corresponds to victory of team batting first and 0 to victory of that batting second

For our classification and prediction purposes, we used the third segment and used all the above mentioned attributes.

2.1 Training Data

We start by randomly shuffling the data of third segment in R. Then we pick top 70% of the records for training and the rest 30% is reserved for testing. Before using any of the algorithms mentioned in the previous section, we have scaled the quantitative attributes MR,OR,MRN,ORN,MW,OW using scale function in R. Scale function in R by default uses Z-score method to normalize each column. After normalization, we assigned the following weights to attributes:

- MR: 0.65
- OR: 0.65
- MW: 0.35
- OW: 0.35
- MRN: 0.65
- ORN: 0.65

By assigning weight for example of 0.65 to MR, we mean that we replaced the column MR by 0.65*MR in the data frame. We arrived at these weights by a series of random trials to check which combination of parameters gives better accuracy.

2.2 Test Data

The 30% of the data reserved for testing the model is stripped off the Win_Bat_First column because that is the one that we want to predict. The attributes which were normalized in training data are also normalized in test data before giving it as input to the model for prediction. In the subsequent section, we describe the results that we received for three different models.

Aim Our aim for predictive analysis was to predict which team won the match based on their performance in the third segment i.e. 14-20 overs.

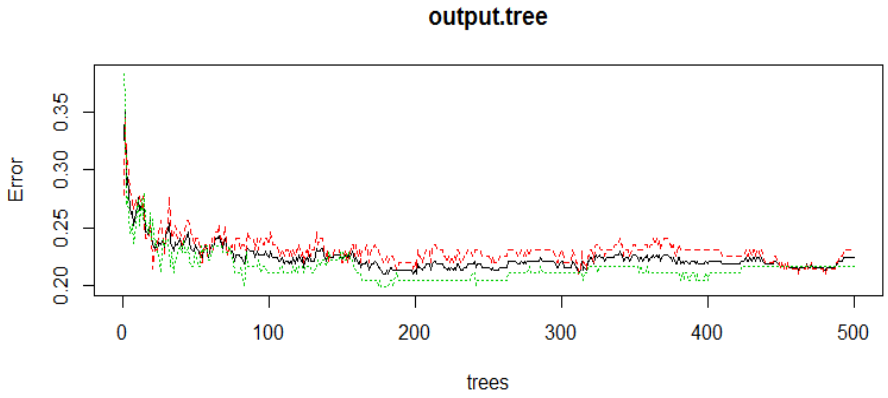
3 Results

In this section, we describe the results and the plots that we got for each of the algorithms. We mention the accuracy(Number of correctly classified rows/Total rows in test data) and the model plot for each.

3.1 Random Forest

- Accuracy: 76.92%
- Confusion Matrix

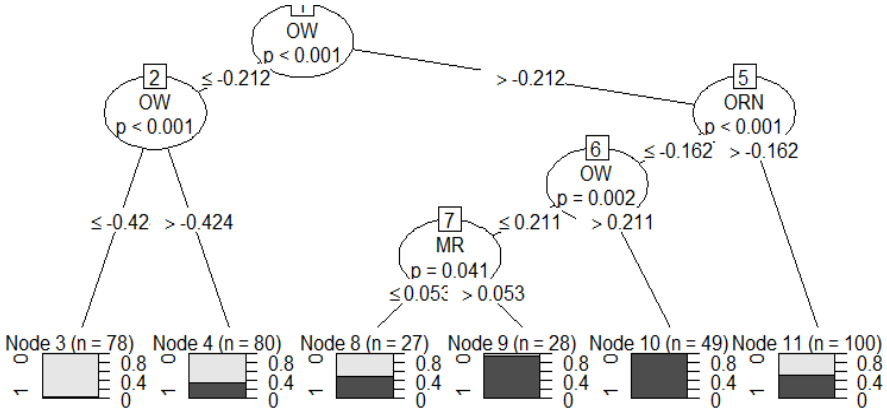
	Reference 0	Reference 1
Predicted 0	59	16
Predicted 1	20	61



3.2 Decision Trees

- Accuracy: 77.5641%
- Confusion Matrix

	Reference 0	Reference 1
Predicted 0	57	13
Predicted 1	22	64



3.3 Support Vector Machines

- Accuracy: 82.692%
- Confusion Matrix

	Reference 0	Reference 1
Predicted 0	67	15
Predicted 1	12	62

4 Conclusion

Support vector machine algorithm gave us the highest accuracy our test dataset. On an average, the accuracy for different training and test data sets obtained by random shuffling vary from 72%-83%. When we increase our train data ratio from 70% onwards, the accuracy for all the models was noticed to fall down. Scaling the attributes and assigning them weights played a crucial role in improving the model accuracy as without normalization, the average accuracy was noticed to be around 61%.