**Akhil Anand**

**MATH 2140-01**

**Applied Statistics Final Project**

**IPL Winner Prediction**

**Introduction**

Growing up in India, cricket was the biggest thing around which is surprising given the national sport of India is Field Hockey. For as long as I can remember, I was outside playing all kinds of sports and never the kid who played video games. There was another problem, my parents never allowed video games, especially no violent games. I started playing cricket when I was 4 when I could get my hands on a cricket bat and ball, now this is supposed to be a professional paper so I am not going to keep dragging this on for any longer but the gist of the story is that I’ve played cricket forever.

The inspiration for this project came from one of the biggest and most exciting cricketing tournaments in the world. It was first started in 2008, I remember going for one of the first matches that were played in Delhi, the excitement for the event was exuberant. A little background on the sport of cricket, it was originally created to have one format, test cricket. This type of cricket usually lasts five days, 90 overs each (I will get into what those are), and 4 innings (2 per side). An over consists of six deliveries, think of them similar to pitches in baseball but here the bowler has a whole run-up and cannot bend their arm while bowling like in baseball. So as you can imagine, people watching this format of cricket really do need to be cricket geeks, and I am guilty of being one. Having spent hours and hours on end, maybe even days sometimes I think that qualifies me as a geek. This format also consists of regular intervals for lunch and tea each day for the players to get some food and recover. By doing some quick match test matches usually have 2700 deliveries bowled over five days, you can see that this can get boring for almost everyone. I will explain how one wins in this format during my presentation in class.

Another format of a match is one-day internationals, these are definitely a lot more exciting but sometimes can feel like forever. One-day internationals consist of two innings with either 50 overs bowled by each side, the first team trying to score as many runs as they can without losing all of their wickets (a wicket is like a strike in baseball, but every batsman gets one strike and the team gets 10 wickets). The way you win in this format is by chasing down the target set by the first team or getting the other team all out before they reach the target. I will explain this further during the presentation.

Coming to the last format in this paper, T20 matches. These consist of 20 over matches which usually last 3 hours. This last format is the most exciting and this is where IPL cricket originated. These matches are usually a lot more high-scoring and exhilarating, and this paper is mainly consisting of predicting the home team winner given all these different variables. The way you win in this format is the same as winning in an ODI match. So, let’s start looking at some data, do some exploratory data analysis, and modeling before I bore you to death with my geeked-out rant about cricket.

**Project Ideas**

I came up with three different ideas for these datasets for projects. The first one and mostly what this paper will contain is trying to predict whether the home team will win given the variables collected from the datasets. The other was trying to use what I learned about multinomial logistic regressions to predict the actual team name that is going to win. I did not have very high hopes for this project and lastly to make it more of a linear problem and do some linear regression tests, the ability to predict the number of runs scored by the home side given the different variables.

**Dataset Exploration**

I found this dataset on Kaggle, the machine learning competition platform. The initial dataset consisted of two different datasets, the “matches” dataset containing data on every single match played in every single season from 2008-2017. Sadly, this dataset did not contain the most recent data of the last few seasons. One reasoning I could give would be Covid-19 and that restricting matches being played over the last few years. The “deliveries” dataset consisted of data on every single ball bowled, anything that may have happened in that particular match was in that dataset.

Let’s do the initial exploration on the matches dataset and get a better understanding of every single variable. Variables from the matches dataset and inference:

* **Id**: No real information from this, just the id of the match
* **Season:** Again, very self-explanatory.
* **City:**  This was used to do a lot of EDA and data cleaning. This variable wasn’t expected to be a big influence but was mostly used in every single model created
* **Date:** Was removed.
* **Team1:** Used in doing a lot of EDA and modeling. This variable was information on the first team playing in the match
* **Team2:** This variable contained information on the second team playing in the match.
* **Toss\_winner:** Variable contains information on which team won the toss.
* **Toss\_decision:** Variable contains information on what the toss winner decided to do, bat or bowl first.
* **Result:** Was the result normal or a tie or no-decision.
* **Winner:** Winner of the match, team name before performing any cleaning.
* **Win\_by\_runs:** The number of runs the team won in the match.
* **Win\_by\_wickets:** The number of wickets the team won by in the match.

The other variables in this dataset were not as useful in the modeling so those are not included in this paper. The next dataset, deliveries:

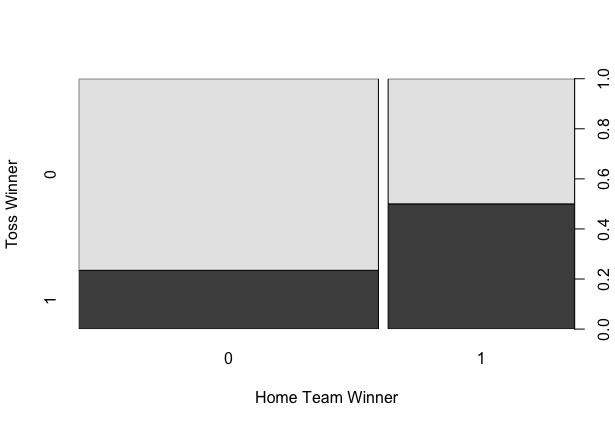
* **Inning:** Innings of the match, most matches have two innings, some go to four because of a super over.
* **Batting team:**  Self-explanatory
* **Bowling team:** Self-explanatory
* **Over:** Over number in the innings
* **Ball:** Delivery number in the over
* **Batsman:** Player facing the delivery
* **Non-striker:** Batsman on the other side of the pitch.
* **Bowler:** Player bowling the deliveries
* **Is\_super\_over:** A super over is bowled when the match is tied and an extra over per side is bowled.
* **Extra\_runs:** The number of runs scored from wides or no-balls
* **Total\_runs:** The total runs from each delivery.

There were a few other variables but those were subsets of extra runs and total runs so not included here.

**Data Cleaning and EDA**

The first and most crucial step was to figure out a way to merge the two datasets into one without losing much valuable information. I based my decisions on intuition and which variables I thought would be the highest influence and best predictors in predicting the winner of the match. A lot of the data from the “deliveries” dataset was merged into the matches dataset. Here were the 4 ideas that I came up with based on intuition and what I could get from the deliveries dataset.

* Get the total number of extras bowled in the given match
  + These come from wides and no-balls
  + The intuition was based on the fact that if there are more extra runs given in a match, there is a higher probability for a higher score for the batting team which in turn would lead to a higher chance of winning.
* Get the number of deliveries bowled in the match
* Get if the match had a super over
* Get the total runs scored in the match

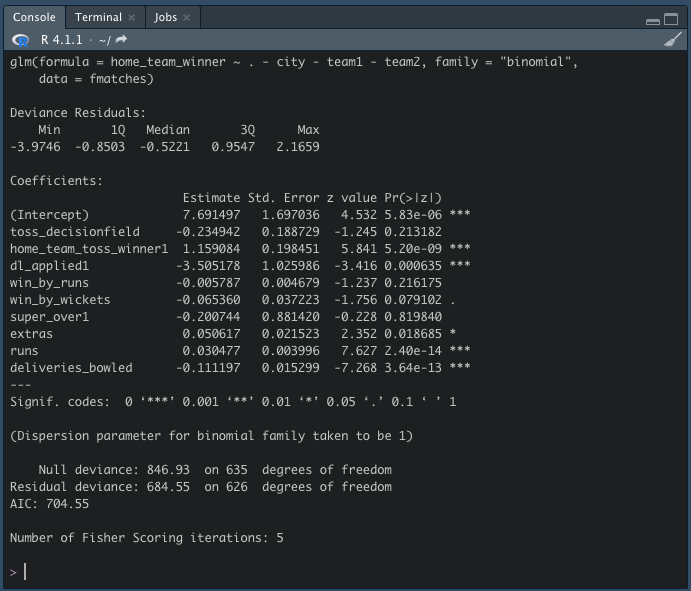
After gathering all this data, at first, the data for all these ideas were gathered per innings per match from the deliveries dataset. Since the first and main project was to predict if the home team wins, the winner column was changed into whether the home team won the match or not. At first, this column contained names of teams that won the match. The logic of this came from comparing the city the match was played in and whether the team won the match was on their home ground. The same logic was applied to the toss winner column, whether the home team won the toss or not. I have this graph as to what percentage of home team toss winner won the match. The same logic was used to get the extras, runs, and deliveries bowled for the home team in the match. The data for whether the match had a super over was also gathered and added to the dataset.

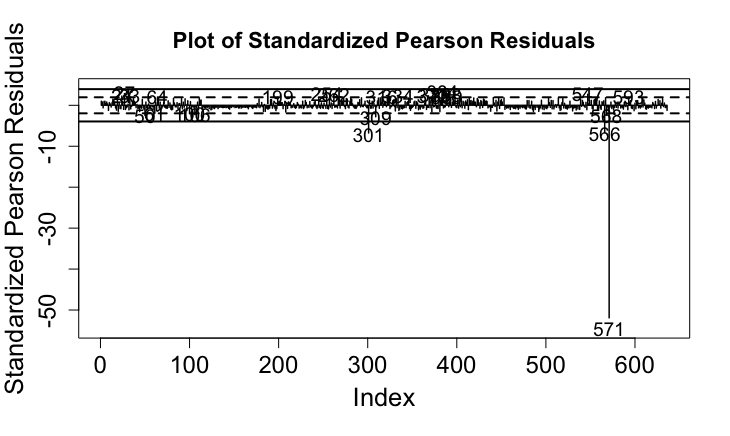
I wanted to take a look at what percentage of toss winners leads to the home team winning. 50% of the time, it the home team won the toss, the won the match, and 20% of the time, if the home team lost they won the match. This is evident from the plot to the left.

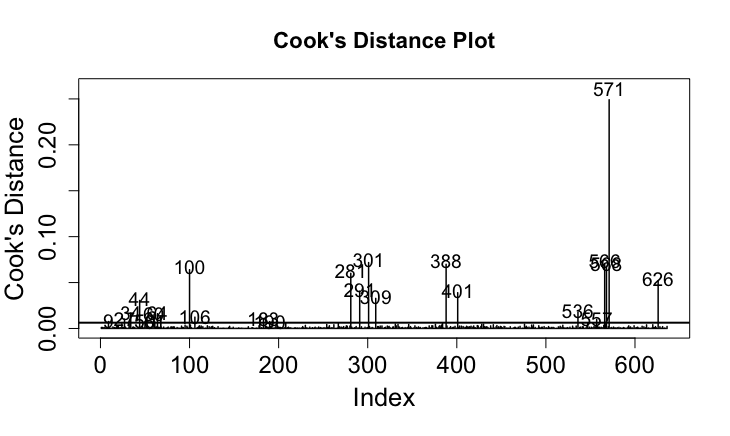
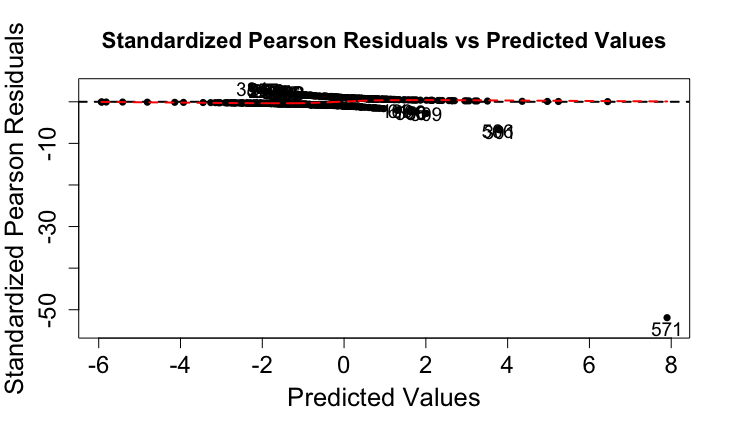
**Data Modeling**

**Logistic Regression:**

At first, I ran a full model without cities or team names and ran some diagnostics on the model. This was only to get a feel of how the variables were working in the model. Here are some diagnostic plots and the summary from the model:







Datapoint 571 seems to be having quite a lot of influence on this model so it might be a good idea to remove this from the model and run it again. To overcome this, I could have either removed the point, run the model again or split the dataset into training and testing and try getting some predictions.

The data was split into 67% for training and 33% for testing, three different logistic regression models were run to see what kind of predictions they produce.

fit1 <- glm(home\_team\_winner~.-city-team1-team2, data = training\_data, family = "binomial") # code for actual model

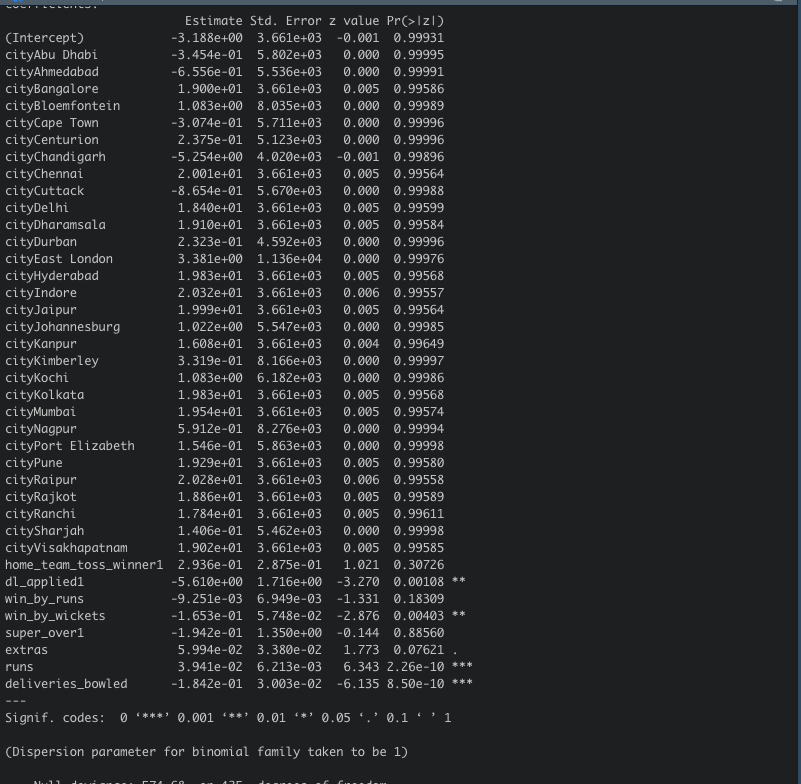
logistic.plots(fit1) # logistic plots

lin\_preds\_test <- predict(fit1,newdata=testing\_data) # predictions from model

class\_winner <- ifelse(lin\_preds\_test>0.5,1,0) # classifications from model

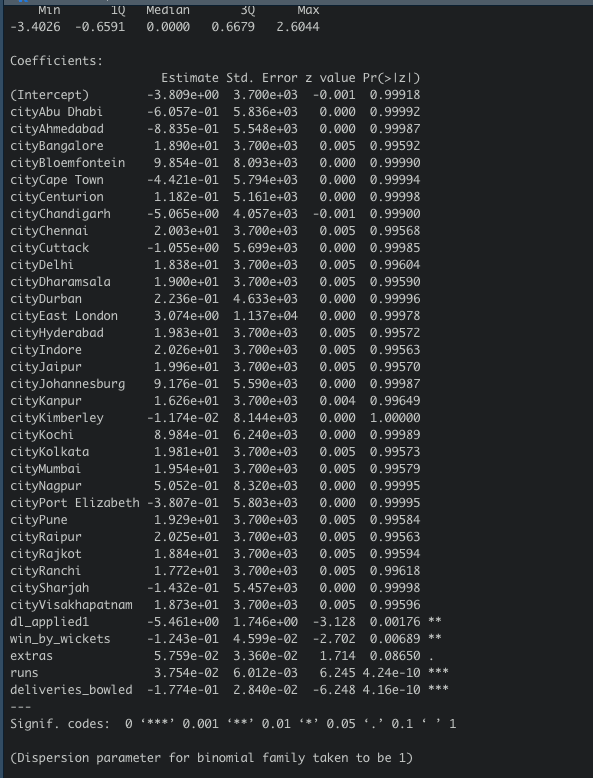
acc\_test <- # model accuracy

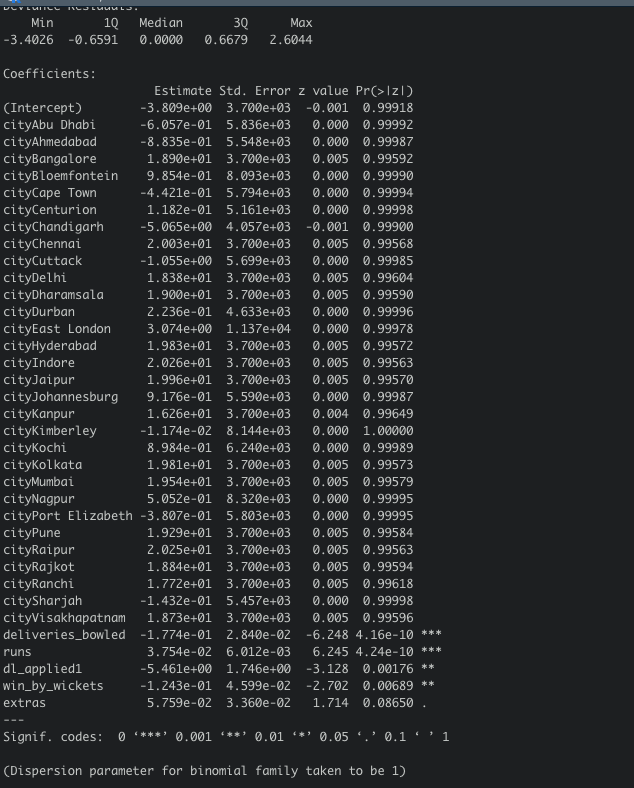
sum(testing\_data$home\_team\_winner==class\_winner)/nrow(testing\_data)

These same lines of code were run to get three different models and accuracies with some changes to the variables included in each model. The best accuracy achieved out of these models was 78.57%. Here is the summary output of that model:

It was interesting to see that with the city variable in the model better accuracy was achieved than without the city variable. From the output, one can see that there is no influence from the city variable on the model.

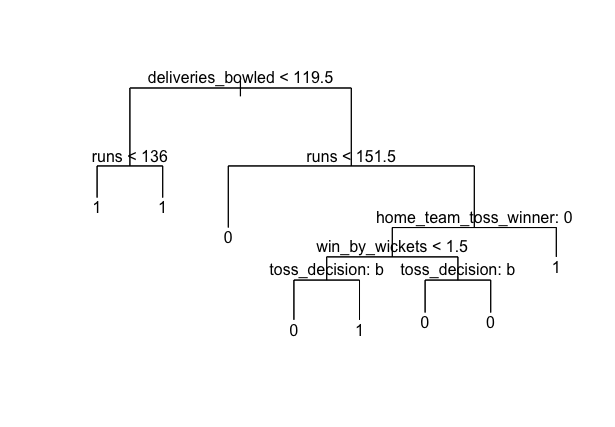
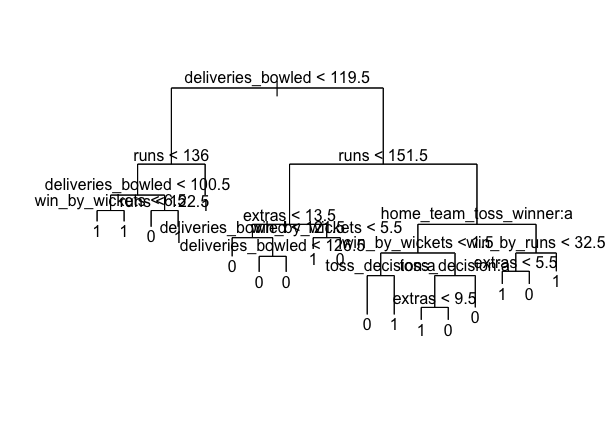
Another interesting observation was that dl\_applied, win\_by\_wickets, extras, runs, and deliveries bowled seemed to be significant in predicting home team wins.

**Stepwise Logistic Regression:**

To get a better understanding and some confidence in the logistic models run earlier, there was a need to run some stepwise procedures. This was to check if my intuition was in line with what the R thinks would be a good model. Both a full model and an intercept model were created, a backward stepwise procedure was run first, here is the output to the right. The model is very similar to the best logistic model above with an accuracy of **77.14%.** On the left is the model output from the forward stepwise procedure. This model also had an accuracy of **77.14%.** It was very interesting to see that both the forward and backward models landed up on the same final model with different significance for the variables. This gave a lot more backing to the final logistic model. Finally, the model was run in both directions and again, the same final model as the backward and forward procedures with an accuracy of **77.14%** was obtained.

**Trees:**

Now to the more exciting part of the analysis, trees seem to be the best predictors of data in my experience, even in this project the highest prediction accuracy came from running trees on the data. They seem to be the easiest to visualize and interpret as well. The first tree run was a simple tree without team and city names, adding city names to trees was making them hard to interpret and the accuracy scores were quite low. Both the main tree and pruned tree are shown below, as one can see, these are much easier to predict. Pruning the tree keeps the best model with a number of decisions one specifies. If we look closely, these are the same as the step and logistic models with the same accuracy of **77.14%**



Summary from 1st tree:

Classification tree:

tree(formula = home\_team\_winner ~ . - city - team1 - team2, data = training\_data)

Variables actually used in tree construction:

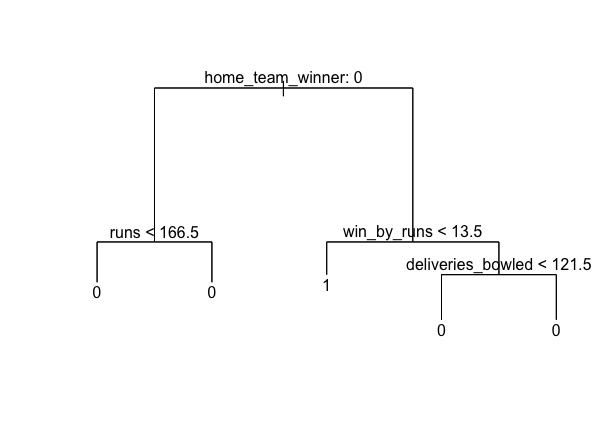
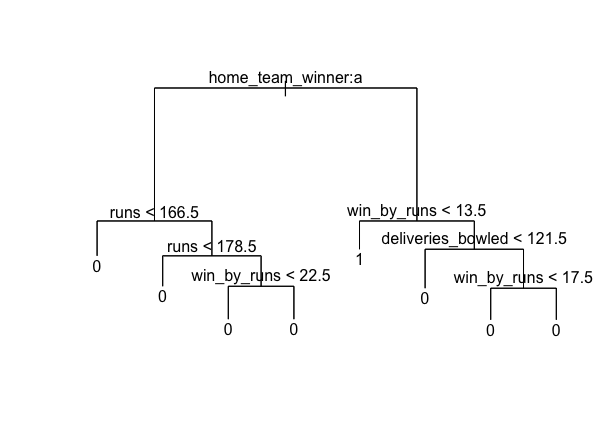
[1] "deliveries\_bowled" "runs" "win\_by\_wickets" "extras"

[5] "home\_team\_toss\_winner" "toss\_decision" "win\_by\_runs"

Number of terminal nodes: 18

Residual mean deviance: 0.6812 = 277.9 / 408

Misclassification error rate: 0.1808 = 77 / 426

The next iteration of the tree model was running a tree without toss\_decision from the model, this was done because the accuracies from the logistic models earlier seemed to benefit without having toss\_decision in the model. This tree model, when pruned did substantially better in predicting whether the home team is going to win or not. Below are the plots of the trees, the tree model had an accuracy of **85.71%** which was substantially better than the logistic models earlier**.**​​ I will explain more of what the plots look like during the presentation.

Summary from 2nd tree:

Classification tree:

snip.tree(tree = tree\_fit2, nodes = c(15L, 5L))

Variables actually used in tree construction:

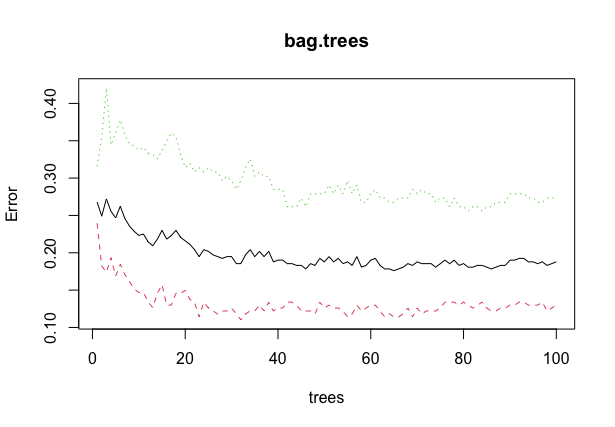
[1] "home\_team\_winner" "runs" "win\_by\_runs" "deliveries\_bowled"

Number of terminal nodes: 5

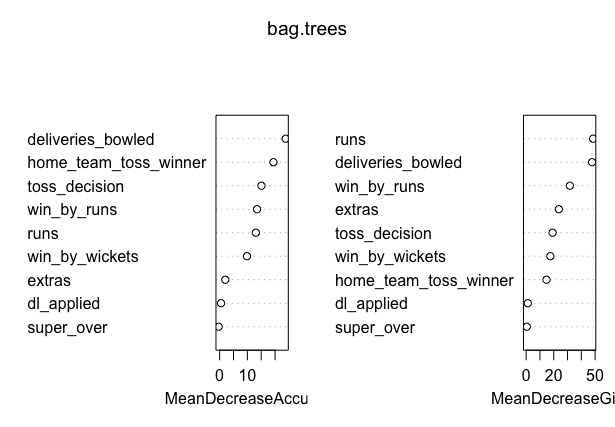
Residual mean deviance: 1.165 = 490.7 / 421

Misclassification error rate: 0.3052 = 130 / 426

**Random Forests:**

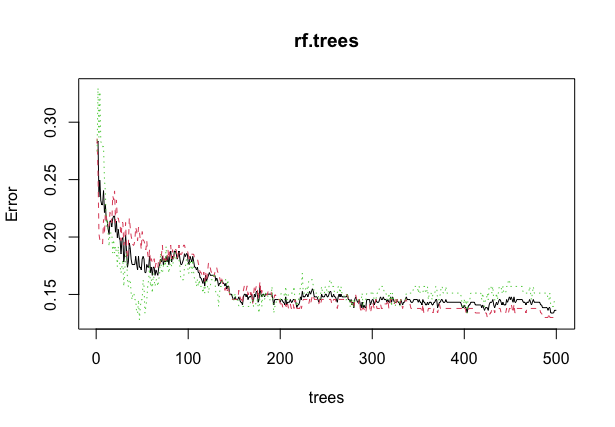
Random forests have always fascinated me, first learning about it in Data Analytics and now being able to put it into practice has been quite educational. Here is the first iteration of the random forests model, this model had everything but team and city names. The model tries 7 decision nodes and runs 100 trees and this was the output. This model had an accuracy of **83.81%**

This shows how the error in predictions is going down as more trees are run.

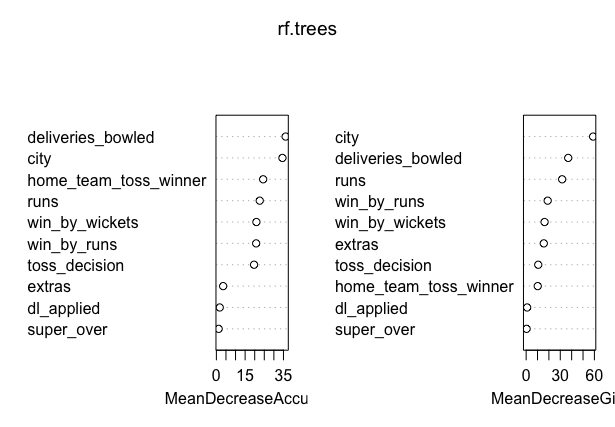


This is the variable importance plot which gives some key information about what the variables are doing.

The last model that was run for this analysis was another random forest model with 500 trees, given below are both the variable importance plots and the error reducing plot of the random forest. The model below had an **89.05%** accuracy in predicting home team wins.



As you can see, the error rate drops drastically in the beginning and then stabilizes in the end



This is the variable importance plot, one can see how important the city is in predicting the home team wins. This gives a lot more confidence in earlier models

**Conclusion**

The results from this analysis have been amazing, being able to predict with **89%** accuracy as to whether the home team will win is a win in my book. All the diagnostics and tests (likelihood ratio and wald tests) led to a boost in confidence with the variables chosen in the models. Having such a high accuracy in predicting whether the home team will win or not would give me the confidence to bet on these models with some real money. Maybe I make my millions doing that, I would like to thank Dr. Brown for pushing the students to get some real work done in these projects!