CS313:Big Data

BIG DATA CLASS PROJECT

IPL ANALYSIS AND MATCH SIMULATION

SNo	Name	USN	Class/Section
1	Siddharth Itagi	01FB16ECS382	G
2	Sumanth V Rao	01FB16ECS402	G
3	Sumedh Bhasarkod	01FB16ECS403	G
4	Tanmaya Udupa	01FB16ECS416	G

Introduction

IPL (Indian Premier League) is one of the most anticipated and entertaining cricket tournament of the year. Well everybody is interested and waiting to know which team will win a particular match. Many people even tend to predict these outcomes. But how many of these outcomes actually come true? Our project is also built on the same lines of predicting the outcome of any IPL match. We just need the player info I.e the batting order and the bowling order of a particular match and there it is, the outcome of the match is in front of you.

ALGORITHM/DESIGN

PHASE 1:

This phase involved building clusters for batsmen and bowlers. These clusters had to be built considering different attributes of a batsman and a bowler. These attributes needed to be appropriately chosen as they play an important role during match simulation. The basic reason for having clusters is that suppose we encounter a batsman-bowler pair whose data isn't available to us. In this case we will try to fetch this data from the clusters which correctly represent the unknown-batsman bowler pair.

Attributes:

For batsman cluster – Innings batted, runs and the batting strike rate.

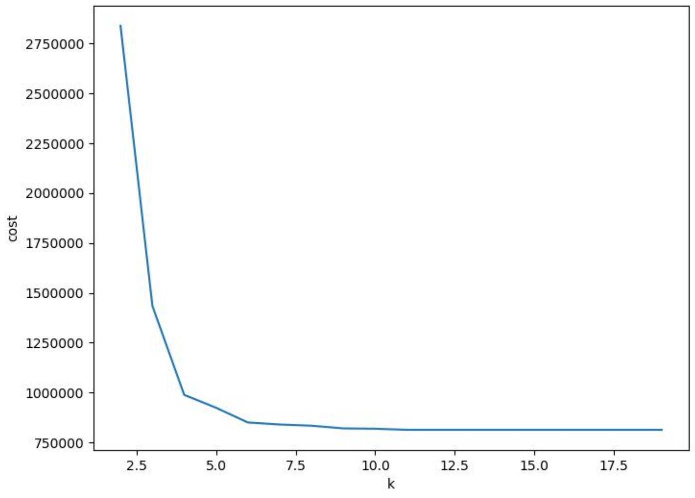
For bowler cluster - Innings bowled, Wickets, Average and Economy.

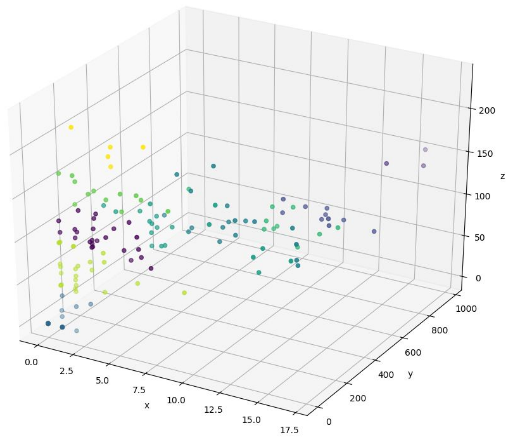
Using the plot of the number of clusters vs cost. We found the elbow point for both the batsman and bowler cluster. The elbow point basically signifies that even after increasing the number of clusters the cost remains the same after this point.

With the results of the batsman and bowler graphs we got these results:

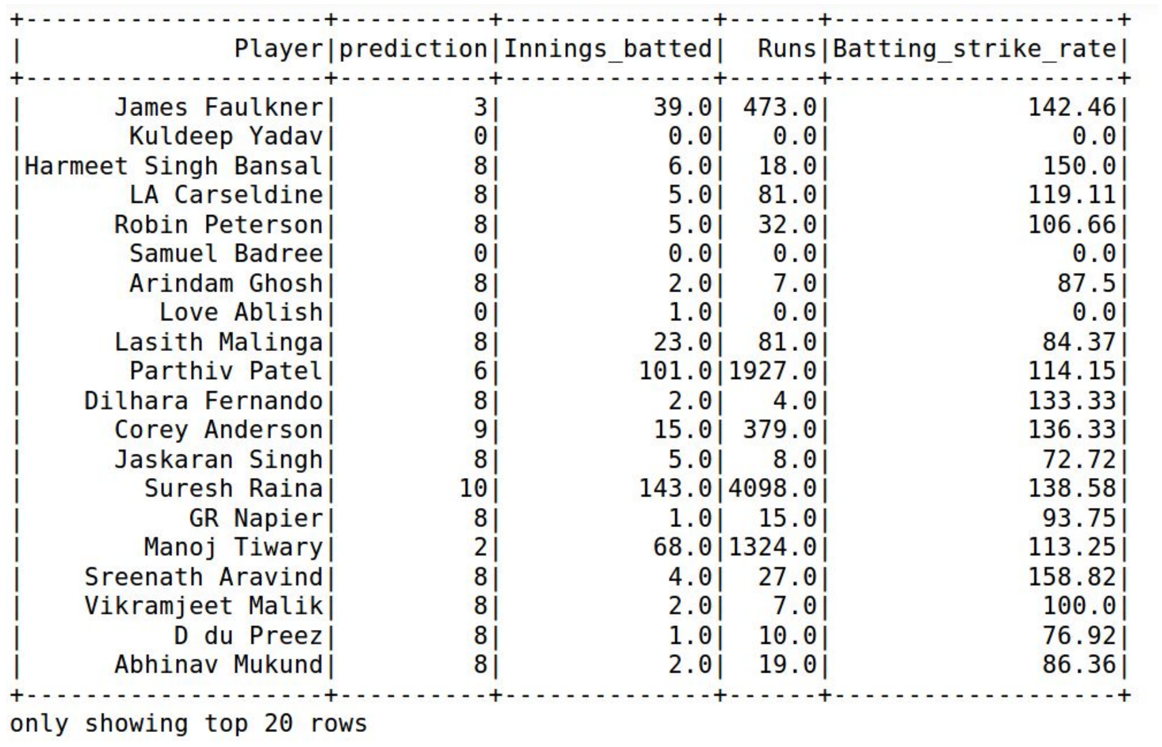
Number of batsman cluster: 10

Number of bowler clusters: 11





We also got a 3D plot for the batsman cluster as we had 3 attributes for it. Visualizing the 4 attribute bowler cluster is difficult.



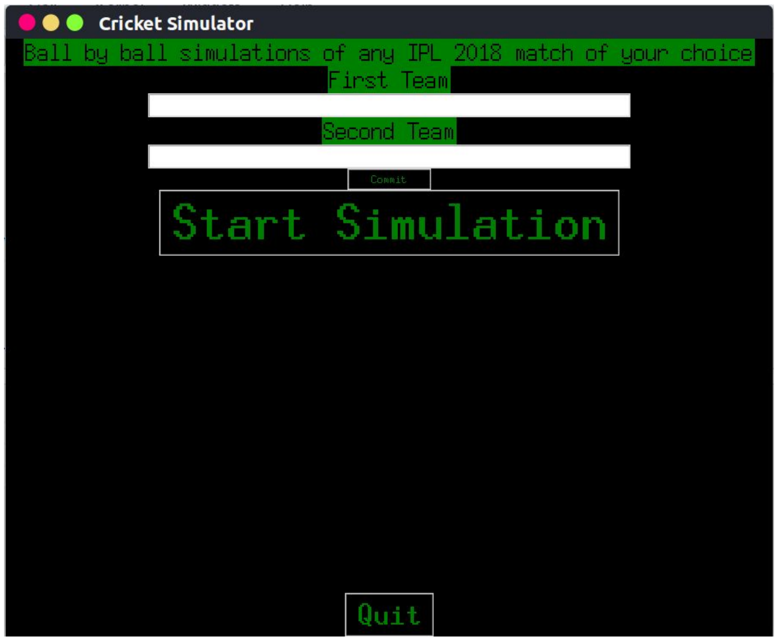
```
+-------------------+----------+-------------+------+-------------------+
|             Player|prediction|Innings_batted|  Runs|Batting_strike_rate|
+-------------------+----------+-------------+------+-------------------+
|     James Faulkner|         3|         39.0| 473.0|             142.46|
|      Kuldeep Yadav|         0|          0.0|   0.0|                0.0|
|Harmeet Singh Bansal|         8|          6.0|  18.0|              150.0|
|      LA Carseldine|         8|          5.0|  81.0|             119.11|
|     Robin Peterson|         8|          5.0|  32.0|             106.66|
|      Samuel Badree|         0|          0.0|   0.0|                0.0|
|      Arindam Ghosh|         8|          2.0|   7.0|               87.5|
|        Love Ablish|         0|          1.0|   0.0|                0.0|
|     Lasith Malinga|         8|         23.0|  81.0|              84.37|
|      Parthiv Patel|         6|        101.0|1927.0|             114.15|
|    Dilhara Fernando|         8|          2.0|   4.0|             133.33|
|     Corey Anderson|         9|         15.0| 379.0|             136.33|
|      Jaskaran Singh|         8|          5.0|   8.0|              72.72|
|       Suresh Raina|        10|        143.0|4098.0|             138.58|
|          GR Napier|         8|          1.0|  15.0|              93.75|
|       Manoj Tiwary|         2|         68.0|1324.0|             113.25|
|    Sreenath Aravind|         8|          4.0|  27.0|             158.82|
|    Vikramjeet Malik|         8|          2.0|   7.0|              100.0|
|         D du Preez|         8|          1.0|  10.0|              76.92|
|     Abhinav Mukund|         8|          2.0|  19.0|              86.36|
+-------------------+----------+-------------+------+-------------------+
only showing top 20 rows
```

PHASE 2:

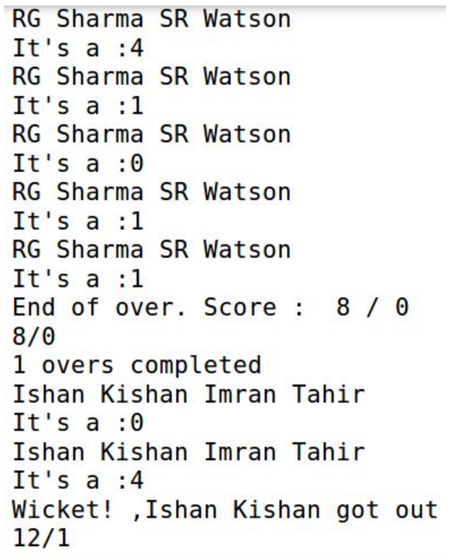
Once we were ready with the batsman and bowling clusters the next part of our project involved match simulation.

The first step involved calculating probabilities of a batsman scoring a 0,1,2,3,4 or 6 runs. We then calculated the cumulative probabilities for each ball.

We basically generate a random number between 0 and 1 and then check under which category does this probability fall under. Also suppose we don't have sufficient data about a particular batsman-bowler pair (Note: Here we have kept a threshold of 5 below which we need clustering) we end up clustering and getting the probabilities.



For wicket probability we started with a probability of 0.9 and then calculated the probability such that this probability decreases after each ball and once this probability falls below 0.5 we say that a wicket is taken.

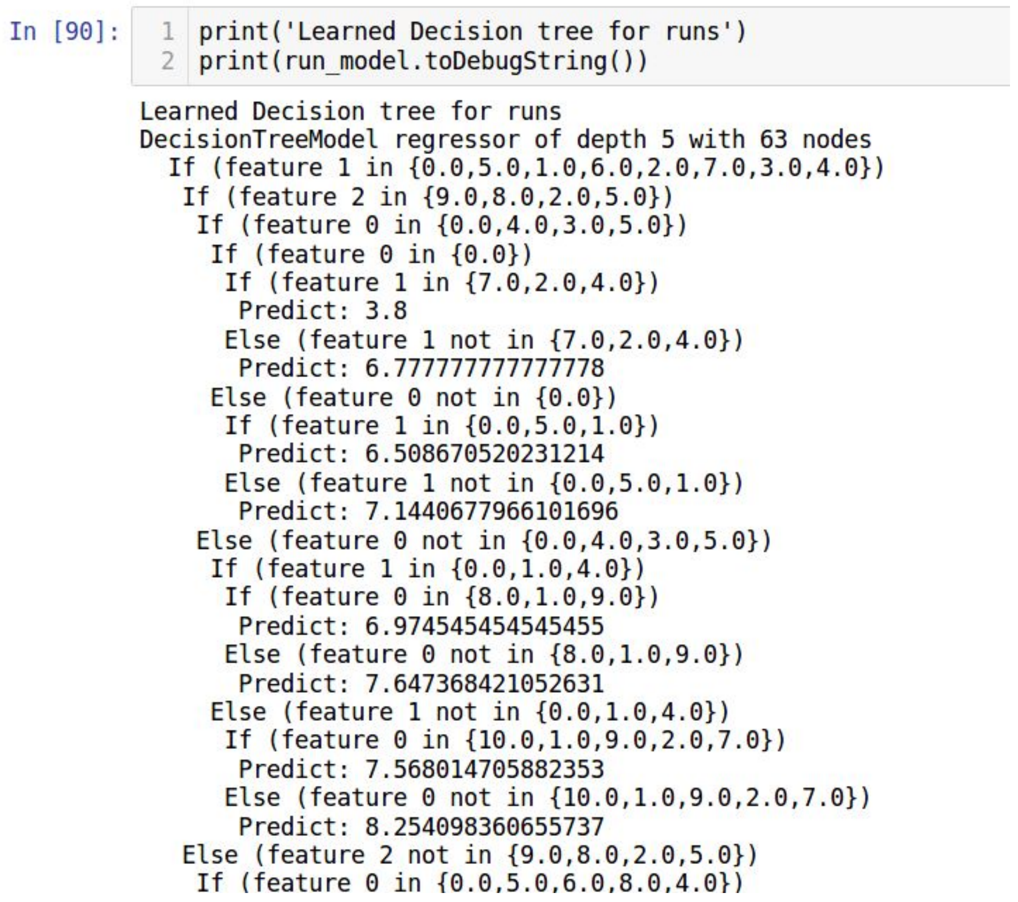


```
RG Sharma SR Watson
It's a :4
RG Sharma SR Watson
It's a :1
RG Sharma SR Watson
It's a :0
RG Sharma SR Watson
It's a :1
RG Sharma SR Watson
It's a :1
End of over. Score :  8 / 0
8/0
1 overs completed
Ishan Kishan Imran Tahir
It's a :0
Ishan Kishan Imran Tahir
It's a :4
Wicket! ,Ishan Kishan got out
12/1
```

We then simulated this for 6 matches and obtained the results for these.

PHASE 3:

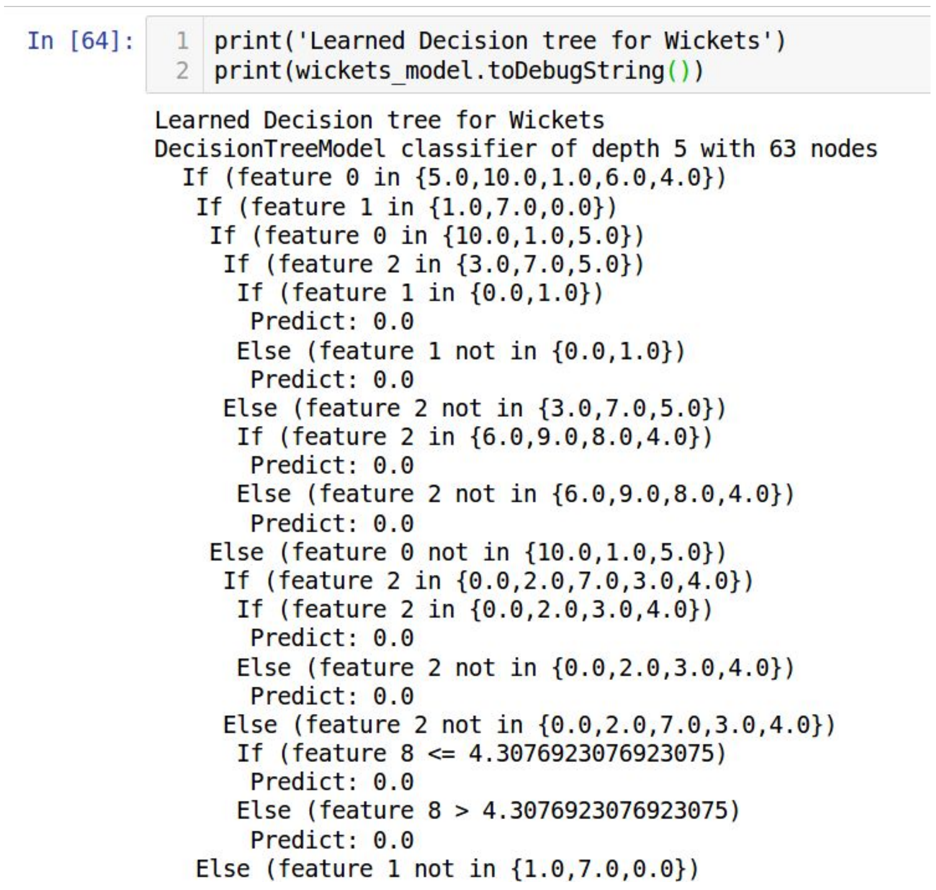
This phase involves the building of a decision tree for match simulation. The important part in building a decision tree is to make the input suitable to be passed into the decision tree.



```
In [90]:  print('Learned Decision tree for runs')
           print(run_model.toDebugString())
```

```
Learned Decision tree for runs
DecisionTreeModel regressor of depth 5 with 63 nodes
  If (feature 1 in {0.0,5.0,1.0,6.0,2.0,7.0,3.0,4.0})
   If (feature 2 in {9.0,8.0,2.0,5.0})
    If (feature 0 in {0.0,4.0,3.0,5.0})
     If (feature 0 in {0.0})
      If (feature 1 in {7.0,2.0,4.0})
       Predict: 3.8
      Else (feature 1 not in {7.0,2.0,4.0})
       Predict: 6.777777777777778
     Else (feature 0 not in {0.0})
      If (feature 1 in {0.0,5.0,1.0})
       Predict: 6.508670520231214
      Else (feature 1 not in {0.0,5.0,1.0})
       Predict: 7.1440677966101696
    Else (feature 0 not in {0.0,4.0,3.0,5.0})
     If (feature 1 in {0.0,1.0,4.0})
      If (feature 0 in {8.0,1.0,9.0})
       Predict: 6.974545454545455
      Else (feature 0 not in {8.0,1.0,9.0})
       Predict: 7.647368421052631
     Else (feature 1 not in {0.0,1.0,4.0})
      If (feature 0 in {10.0,1.0,9.0,2.0,7.0})
       Predict: 7.568014705882353
      Else (feature 0 not in {10.0,1.0,9.0,2.0,7.0})
       Predict: 8.254098360655737
   Else (feature 2 not in {9.0,8.0,2.0,5.0})
    If (feature 0 in {0.0,5.0,6.0,8.0,4.0})
```

We calculated the average strike rate, average economy for every cluster so as to be able to obtain the data relevant for this training. Here are the 2 visualizations for Decision trees. The top one indicates the Decision tree for runs and the bottom one indicates the decision tree for the wickets.



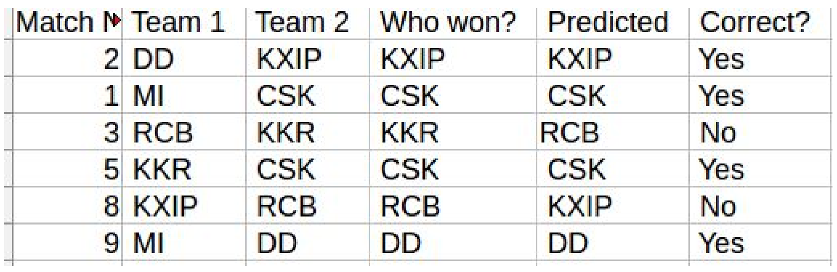
```
In [64]:  print('Learned Decision tree for Wickets')
           print(wickets_model.toDebugString())

Learned Decision tree for Wickets
DecisionTreeModel classifier of depth 5 with 63 nodes
  If (feature 0 in {5.0,10.0,1.0,6.0,4.0})
   If (feature 1 in {1.0,7.0,0.0})
    If (feature 0 in {10.0,1.0,5.0})
     If (feature 2 in {3.0,7.0,5.0})
      If (feature 1 in {0.0,1.0})
       Predict: 0.0
      Else (feature 1 not in {0.0,1.0})
       Predict: 0.0
     Else (feature 2 not in {3.0,7.0,5.0})
      If (feature 2 in {6.0,9.0,8.0,4.0})
       Predict: 0.0
      Else (feature 2 not in {6.0,9.0,8.0,4.0})
       Predict: 0.0
    Else (feature 0 not in {10.0,1.0,5.0})
     If (feature 2 in {0.0,2.0,7.0,3.0,4.0})
      If (feature 2 in {0.0,2.0,3.0,4.0})
       Predict: 0.0
      Else (feature 2 not in {0.0,2.0,3.0,4.0})
       Predict: 0.0
     Else (feature 2 not in {0.0,2.0,7.0,3.0,4.0})
      If (feature 8 <= 4.3076923076923075)
       Predict: 0.0
      Else (feature 8 > 4.3076923076923075)
       Predict: 0.0
   Else (feature 1 not in {1.0,7.0,0.0})
```

This input data was converted to LabeledPoint (an input form suitable for decision tree) and then fed into the decision tree.

We then trained this decision tree and then simulated this for 10 matches and noted the outcomes.

EXPERIMENTAL RESULTS



Match N	Team 1	Team 2	Who won?	Predicted	Correct?
2	DD	KXIP	KXIP	KXIP	Yes
1	MI	CSK	CSK	CSK	Yes
3	RCB	KKR	KKR	RCB	No
5	KKR	CSK	CSK	CSK	Yes
8	KXIP	RCB	RCB	KXIP	No
9	MI	DD	DD	DD	Yes

FUTURE ENHANCEMENTS

For any project to be actually brought into implementation it should have a good accuracy. We plan to improve the accuracy even further so that we can take this project to a higher level. Also we aim to reduce the time taken for this entire simulation and make It even more efficient. We also want to improve accuracy by applying a new model like neural net.

REFERENCES

1) Decision tree documentation

https://spark.apache.org/docs/latest/ml-classification-regression.html#decision-trees

2) Data:

https://cricsheet.org

http://www.espncricinfo.com/

EVALUATIONS (Leave this for the faculty)

Date	Evaluator	Comments	Score