# A Learning Algorithm for Prediction in the game of Cricket

CS – 7641 A (Machine Learning)
Sethuraman K, Parameswaran Raman, Vijay Ramakrishnan

# Problem Description

- Scenario 1:

  Team 1 scored 200 runs from their 50 overs, and then Team 2 reaches 146 for the loss of two wickets from their first 40 overs before rain stops play.

- Scenario 2:

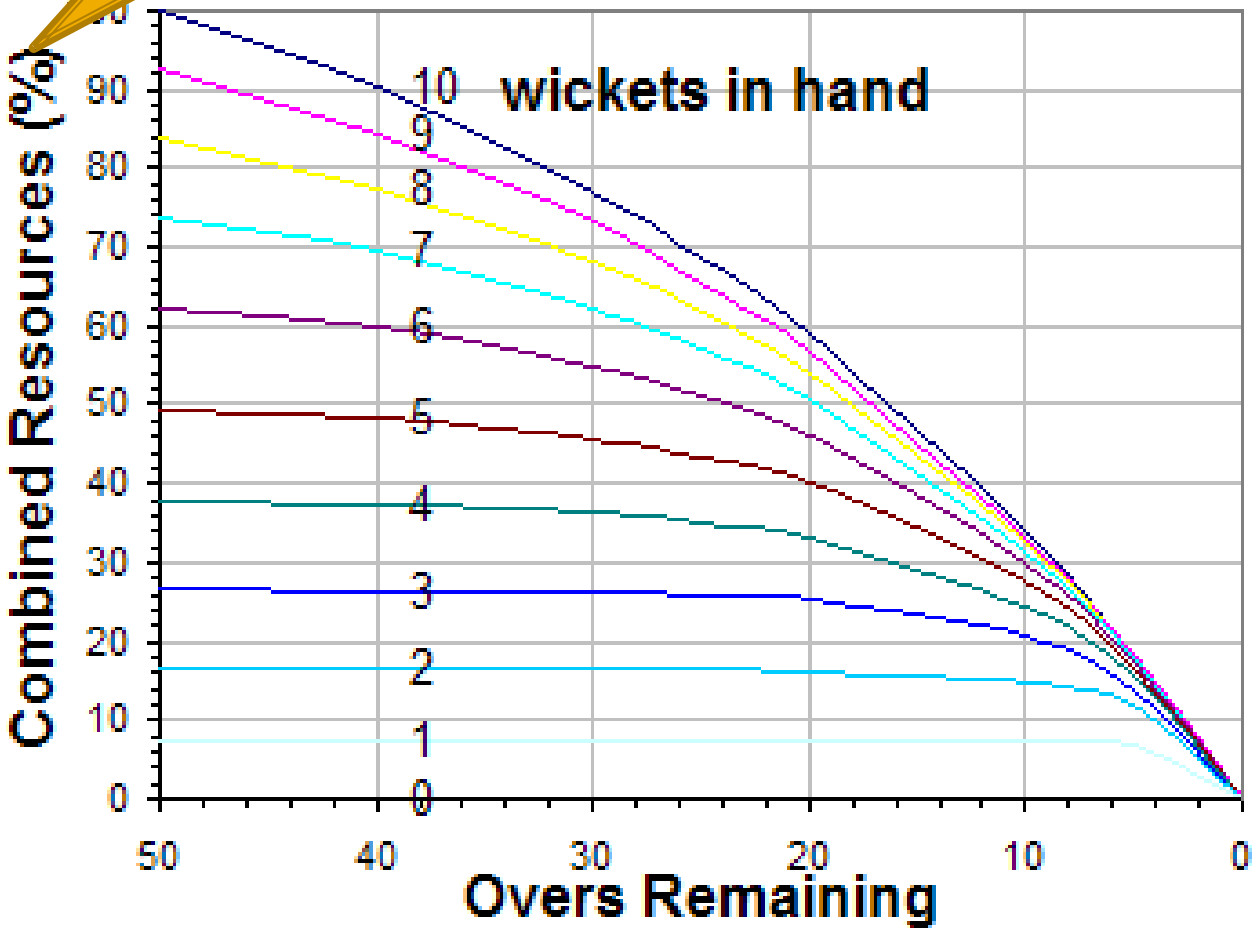
  Bad weather reduces match to 40 overs. Team 1 scores 223 from their 40 overs. During, Team 2's innings, it rains after 30 overs, by which point they are 147/5. They lose 5 overs due to rain and face the final 5 overs.

# Existing Work – D/L method

- Duckworth/Lewis method
  - Each team has two **resources** to use to make as many runs as possible: overs and wickets
  - At any point, score depends on combination of these two resources
  - A published table provides the % of these combined resources remaining for any number of overs left and wickets lost
  - This table helps calculate the target score

# D/L Method

Essence of D/L method



# Drawbacks

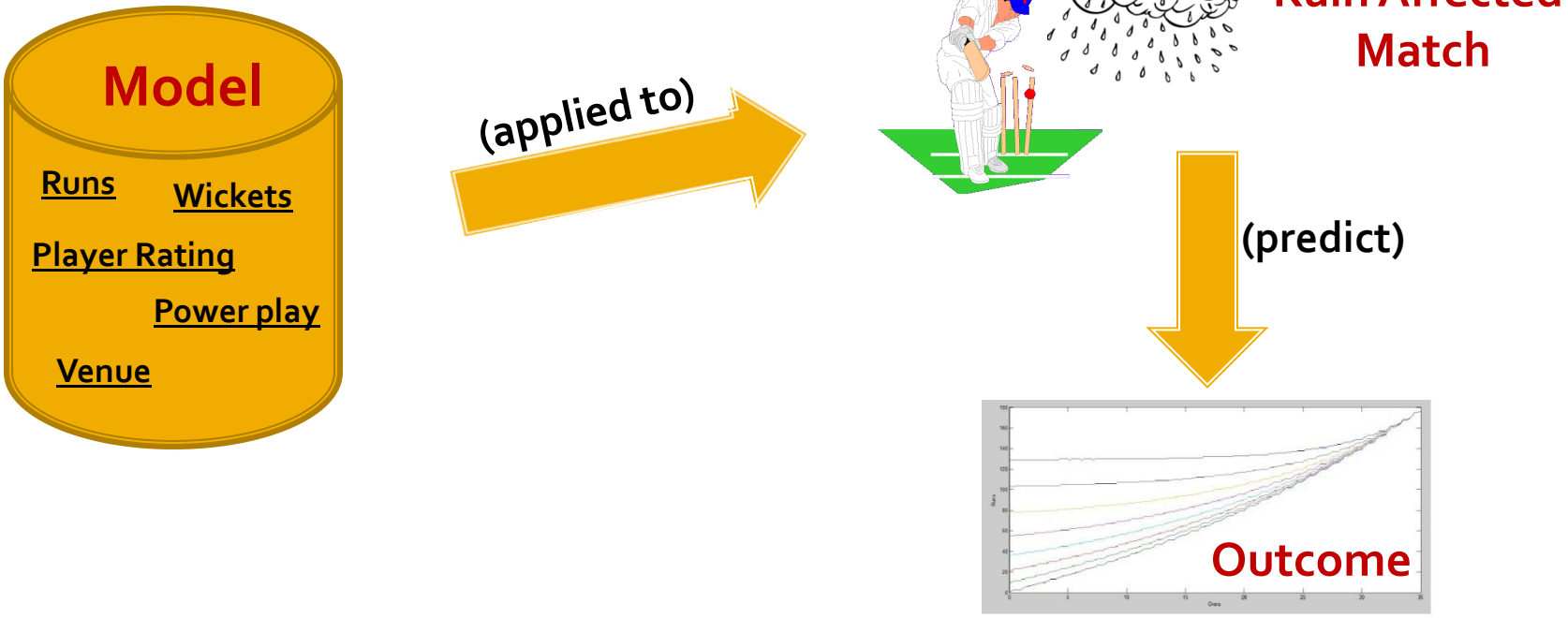
- Wickets are (necessarily) a much more weighted resource than overs
- Other factors that equally affect outcome not considered:





# Our Attempt

- Creating an efficient model to reflect as many diverse scenarios in real-time matches as possible



# Data sets

- Data sets for cricket matches were not available in usable form
- Data had to be extracted from websites that maintain cricket statistics (eg: 

)
- Ball-by-ball details of the match were extracted (commentary)
- Data was formatted based on desired attributes (over, runs, wickets, batsman, bowler, power play) and written to a file
- In all, data for 52 innings were collected

# Data sets

Commentary

0.1 Steyn to Dilshan, no run, nice bounce and carry straightaway, leaves the right hander and easily negotiated by Dilshan

0.2 Steyn to Dilshan, no run, gets bat on ball and thumps it hard down to mid-off

0.3 Steyn to Dilshan, FOUR, the first thwack of the tournament, full and outside the off stump, he gets enough power to push the ball past the covers, quick outfield

0.4 Steyn to Dilshan, 1 leg bye, strays on the pads and the ball drops to the on side, now Jayasuriya on strike

0.5 Steyn to Jayasuriya, no run, kicks up a bit off the surface and importantly the batsman drops his glove and fends it off

0.6 Steyn to Jayasuriya, 1 run, just opens the face of the bat and angles it down to third man

Here's a stat. The last four day-night games here all won by the team batting first.

**End of over 1** (6 runs) **Sri Lanka 6/0** (RR: 6.00)

| | | | |
|---|---|---|---|
| **ST Jayasuriya** | 1* (2b) | **DW Steyn** | 1-0-5-0 |
| **TM Dilshan** | 4* (4b 1x4) | | |

# Feature Extraction

- D/L method uses only two features – **runs** and **overs**
- New features were added – **venues**, **teams** taking part in the game, data about the **batsman/bowler**, **nature of the over** with field restriction (ie, **power play** or not)
- Ran some feature extraction algorithms & finally selected **Correlation Based Subset Feature Selection (with fast correlation based filter search strategy)**

# CBF Algorithm

- **Principle:**
  *A good feature subset is one that contains features highly correlated with (predictive of) the class, yet uncorrelated with (not predictive of) each other.*
- **Results:**
  *Showed that most important attributes were runs scored, wickets, venue & power play*

**Also validated by other feature estimators like PCA, Latent Semantic Analysis**

# CBF Features

| Feature | Merit of the Feature |
|---|---|
| Run | 0.418 |
| Wicket | 0.418 |
| Venue | 0.095 |
| Power play | 0.076 |
| Batsman | 0.007 |

# Benchmarking

- Almost all target prediction algorithms make use of function approximation
- Standard basic classifiers were used to benchmark the effort of the data sets
- Classification  rate is currently lower than the D/L method mainly because of the lesser number of data sets available for training. This can be increased with more data samples
- kNN and Linear Regression provided high classification rates

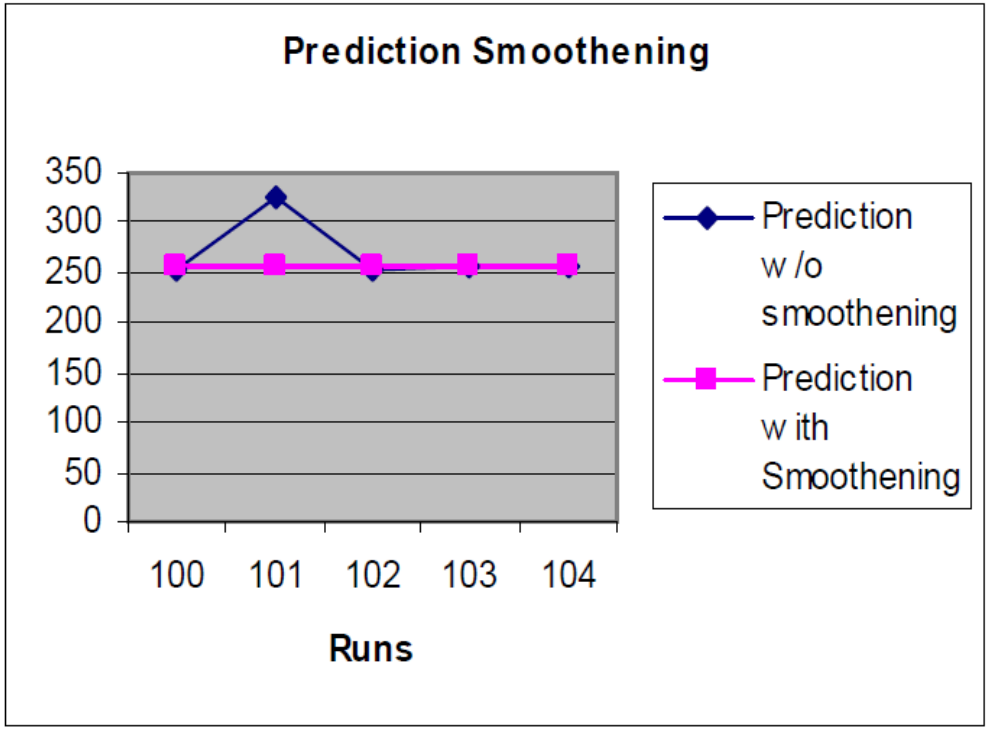
# Classification Rates of Algorithms

| Algorithm | Error Rate |
|---|---|
| Neural Network | 51 % |
| Linear Regression | 19 % |
| kNN | 16 % |
| REPTree | 23 % |

# Quadratic Regression

- **Thought:**
  - kNN and Linear Regression can be used together?
- **Why Quadratic Regression?**
  - Many curve fitting functions used to predict cricket scores are cubic
  - Can be an improvement over Linear Regression, which already performed well
  - Predictor variables are run, $run^2$ ,wicket, $wicket^2$
  - Can be transformed into equivalent Linear Regression and solved
  - Can be coupled with ***Smoothing by Neighbour Polling*** so that the prediction curve is smoothened

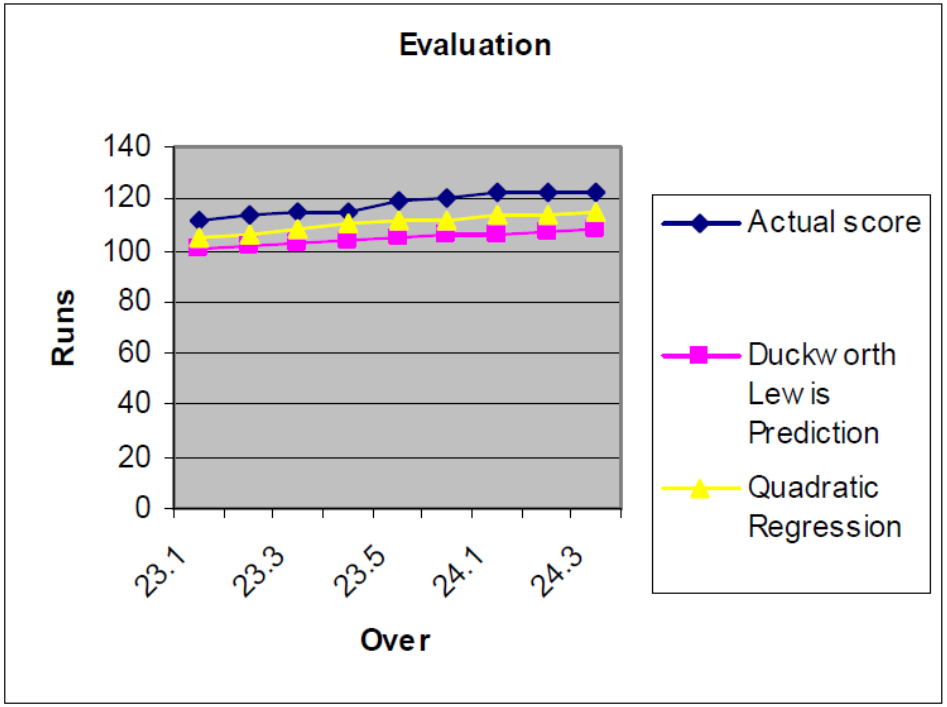
# Smoothing by Neighbour Polling




Reduces spikes in the prediction curve

# Evaluation Metrics

- **Variance Analysis:**
  - Obtain actual match curve
  - Calculate our prediction
  - Fit them over each other and obtain the variance




- This is per-match metric and hence captures accuracy locally over a continuous period
- Converges with Absolute Mean Error & Least Square Error methods (for equal samples in each match)

# Using Momentum as a factor

Match Interrupted → **100 /3** **25 overs**

- Will both the teams score at the same rate?

| After 15 overs: | Score: |
|---|---|
| Team 1 | 90/0 |
| Team 2 | 40/3 |

- Given the above history, will the answer to above question change?

# Challenges

- Data sets

# Conclusion

- Duckworth/Lewis method of score prediction was analyzed and its pitfalls were addressed
- Correlation Based Subset evaluation method was used for Feature Evaluation
- Modified data set was used to benchmark the predictions of naïve classifiers
- Hybrid approach using Quadratic Regression and kNN was used, which performed better than D/L method
- Introduced concept of prediction using Momentum