ALGOMIA

Cricket ODI Match Prediction & Betting Strategies

Sabyasachi Banik

MSc Informatics
University of Zürich UZH
sabyasachi.banik@uzh.ch

date

Table Of Content

1. Introduction
2. Dataset Selection
3. Feature Engineering- a) Success and b) Failed Attempts
4. Model Training
5. Web Scraping OddsPortal
6. Betting Strategies – a) Generalised approach, b) Kelly Criterion
7. Conclusion

Introduction

a) ML Model: We use historical data to fit a model to predict the Winner in an One Day International (ODI) match.

b) Exploring betting strategies: We next examine betting strategies by comparing our model's outcomes against the betting odds for these matches, aiming to identify patterns and strategies that could inform more effective betting decisions."

Dataset

- Finding a “good” dataset is always a challenge for any ML model

- After a hectic search a potential dataset from Kaggle could be found

- This dataset included all the match results for ODI matches in between Jan 1971- May 2023.

- Since the data is already in downloadable form, we can skip the Web Scraping part which makes our job a bit easier.

Click here for the link to the dataset

Feature Engineering- a) Failed Attempts

We basically tried out everything with the data. Here is an overview of things that didn't work out:

1. Ground Location for Home/Away:
- After a lot of search the API nominatim.
-The location results were mostly correct
- On top no need to have an access token for using it.
-Some of the data points were still wrong but needed to be corrected by roughly looking in the dataset manually.
-- But, but the model's performance have decreased which was quite unexpected.
- Numerical Venue codes
- Binary Categorization (Home-1, Away/Neutral-0)

Feature Engineering- a) Failed Attempts

2. Rolling Average:

- Tried with the recent form of teams
  - -Adding a new column with wins in last 5 matches
- Decreased accuracy again

3. ICC Team Ratings:

- Added recent ODI team ratings to better train the algo
  - didn’t help

4. NN/RF Parameters:

- Explored different combination of algorithms
- Particularly NN and RF since they were close contenders
  - RF: n_estimators, max/min_sample_split
  - NN: different hidden_layer_sizes, relu/elu/tanh activation functions, mit & ohne adam solver, alpha, batch_size 2^x

Feature Engineering- b) Success Attempts

- Keeping things rather simple worked out “better”

1. Recency of data:
   - Old data may be irrelevant in predicting recent trends.
   - Sample_weights based on date_rank
   - df with Match Date > 01-01-2000(we want a better model not model trained on many datapoints)
2. One Hot Encoding Team names:
   - Each team gets a unique column
   - High increase in total number of features
   - Nevertheless helped increasing model’s performance

Feature Engineering- b) Success Attempts

3. Keeping Important_teams only:
   - Based on current ICC rankings
     - Only top 15 teams since others are/were not part of any major tournaments

4. Others Include:
   - Keeping Rows that has a “Winner”
   - Dropping rows with “relevant” Columns, ex: Teamname

Model Training

- Test and Train Data Split:
  - Based on Cutoff_date = 01-01-2019
  - Test data only consists of datapoints after the Cutoff_date
  - Not so ideal to train the model for testing the past

- Algorithms Used:
  - Popular ML ones like Random Forest, Gradient Boosting, Neural Network And Logistic Regression
  - Usually for Binary Decision making models like Yes/No, Cancerous/Non Cancerous LR performs better
    - No exception here as well.
    - We reach an accuracy of 72 percent
    - Other algos' perform well too

Web Scraping OddsPortal
And Merging of 2 datasets

- It was the most fun and challenging part
- Odds Portal contains a set of betting odds from different betting websites (like bet365, pinnacle, 1Xbet etc.) and shows the average figure from all these sites
- Manual scraping is extremely difficult
  - requires high HTML understanding
- Used a bunch of tools like Uipath, Octoparse, ParseHub
  - All of these failed miserably.
- Finally discovered WebHarvy (free and effective)
  - Although required manual handling
- It was possible to JOIN the two datasets based on unique date and team names.
  - 72.65 percent of datapoints matched in the merged_df

Betting Strategies (1)

1. Generic Approach:

We follow a bit high level approach. Divide our Strategy into 4 tasks:

a) Task 1- Calculating Implied Probability (IP)

Implied_Prob_Team1 = 1 / Betting Odds for Team 1

Implied_Prob_Team2 = 1 / Betting Odds for Team 2

b) Task 2- Model's Predicted Probability using the best classifier from model training

Using the best classifier from model training (LR) we get the predicted winning probability for both teams in a match-

i) Model_Prob_Team1

ii) Model_Prob_Team2

Betting Strategies (2)

c) Task 3- Finding Value Bets

For Team 1:

if Model_Prob_Team1 > Implied_Prob_Team1

- It’s a Value Bet (safe to play) Why?

Otherwise,

For Team 2:

if Model_Prob_Team2 > Implied_Prob_Team2

- Then this one is the Value Bet

Example-

Situation 1	Situation 2
Betting Odds on Team 1 = 6.0	Betting Odds on Team 1 = 1.1
Implied_Prob_Team1 = 0.166	Implied_Prob_Team1 = 0.90
Model_Prob_Team1 = 0.2	Model_Prob_Team1 = 0.85
0.2>0.166 : Value Bet	0.85 ≠ 0.90 : Not a value Bet

Betting Strategies (3)

d) Task 4- Backtesting

- Simulating betting based on these predictions
- Bets are placed on a team only if it was identified as a “Value Bet” and if that team actually won (Winner column matches the team).
- Profit or loss for each game is calculated based on the odds. If the bet was successful (the team you bet on won), you gain based on the odds (Betting Odds – 1); if not, you lose your stake.
- We calculate the total Profit for both teams. Our model showed as a positive figure meaning the strategy was successful

<u>Key intuition is</u> to identify the Value Bets (i.e place bets when there's an edge over the market, theoretically leading to long-term profitability)

Betting Strategies (4)

2. Kelly’s Criterion:

a) We first calculate Kelly’s fraction

kelly_fraction = (b * p - q) / b

b is set as Betting Odds – 1, representing the net profit per unit of stake, excluding the original stake

p is probability of winning as predicted by our model

q = 1-p

- We calculate Kelly_Fraction_Team1 & Kelly_Fraction_Team2

b) We next calculate the Optimal Betting Amount

- Based on a fixed bankroll of say for example 100 $,
  Bet_Amount_Team1 = Kelly_Fraction_Team1 * bankroll
  Likewise for Team2

Betting Strategies (5)

Example:

Team1 vs Team2

- p for Team1 = 0.8, q = 0.2, Bet1 (Betting Odds for Team1) = 1.3
  b = 1.3-1 = 0.3
- Kelly_Fraction_Team1 = (0.3*0.8 - 0.2)/ 0.3 = 0.1333
- Thus Optimal Kelly Amount for betting on Team1 = 0.133 * 100
  = 1.33

To have "an edge over the market", we should have b> q/p

- In the above case, say Bet2 = 4
  - If we are to Bet for Team2, b = 4 – 1 = 3, q = 0.8, p = 0.2
    3 < 4

But when b = q/p, then Kelly Criterion suggests bet nothing

Betting Strategies (6)

c) Next up we calculate Profit/Loss

Profit_Team1: Based on our model predictions, if we Bet on Team1, ~~can be~~ combined with the idea of finding “Value Bets”, i.e if we find a value bet only then we go for it with our Optimal Bet Amount calculated by Kelly Criterion. Then,

Profit = (Betting Odds on Team1 – 1) * Optimal Bet Amount on Team 1

Loss = - Optimal Bet Amount on Team1

Likewise we find Profit_Team2

Total gets added up to give Net Profits from Kelly Criterion Backtesting

We had a Positive figure this time with the New Model

Conclusion

- Simplistic model but 72% accuracy is not bad indeed (also paper)

- Improvement can be considered with the inclusion of Player Statistics
    - Although finding an appropriate Dataset and merging both with minimum loss of data points will be a challenge

- Betting Strategies – Although pretty intuitive but
    - Betting odds are influenced by market factors and may change over time (dynamic)
    - Always Exists an Element of Risk. Betting involves risk, and even a well-devised strategy does not guarantee profits.