Cricket Match Outcome Prediction Using AutoGluon: A Machine Learning Approach

# Abstract

This study presents an end-to-end machine learning pipeline designed to predict the winner of One Day International (ODI) cricket matches using only pre-match data. Leveraging AutoGluon, an AutoML framework, we developed an ensemble-based classifier to make informed predictions using contextual features such as team rankings, weather conditions, toss outcomes, and match venues. This model serves as a proof of concept for integrating AI into sports analytics, enabling the optimization of pre-game strategies and enhancing fan engagement.

# Objectives

- To forecast ODI match outcomes using only pre-match metadata.

- To apply AutoML techniques to optimize model performance without manual tuning.

- To ensure generalizable and explainable performance using cross-validation and feature engineering.

# Methodology

The methodology encompasses data acquisition, preprocessing, feature engineering, model training, and evaluation.

## 1. Data Preprocessing

The dataset was sourced from historical ODI matches, enriched with weather attributes and team-related details. Initial cleaning included the removal of rows containing missing values in critical columns such as match winner, team names, and invalid temperature values. Columns were renamed to standardized formats for clarity.

## 🌦️ Weather Data Integration Methodology

### Objective

The goal was to augment historical One Day International (ODI) cricket match records with relevant weather information for each match day and venue. This enrichment was necessary to explore how meteorological conditions such as temperature and humidity might influence match outcomes.

### Data Sources

1. **ODI Match Dataset** (odi\_Matches\_Data.xlsx)  
    Contains match details including:  
   * Match Date
   * Venue (City and Country)
   * Team1, Team2, Winner, etc.
2. **External Weather Dataset** Sourced from a public weather API (e.g., OpenWeatherMap, Visual Crossing) or a historical climate archive.  
    This dataset included:  
   * Date
   * Location (city-level or coordinates)
   * Avg\_Temp\_C, Humidity, Wind Speed, etc.

### Matching Logic

To integrate weather data with each ODI match, the following strategy was applied:

#### 1. Date Normalization

* The Match Date field in the cricket dataset was standardized to YYYY-MM-DD format to ensure exact date matches with the weather dataset.
* Any matches with invalid or missing dates were excluded from enrichment.

#### 2. Venue Normalization

* Match venue strings (e.g., "Melbourne Cricket Ground, Australia") were parsed to extract the city name (e.g., "Melbourne").
* City names were standardized (e.g., lowercased, stripped of whitespace and suffixes like "City" or "Ground") to align with weather dataset location entries.

#### 3. Merge Criteria

A **left join** was performed using:  
  
 match\_df.merge(weather\_df, left\_on=['Match Date', 'Venue City'], right\_on=['Date', 'City'])

* In pandas: pd.merge() on ['date', 'city'] or using fuzzywuzzy for approximate city matching if needed.

#### 4. Missing Weather Handling

* For matches where exact weather data was not available:  
  + The nearest available date was used (±1 day tolerance), if applicable.
  + Otherwise, the match was flagged as having missing weather and excluded from weather-related modeling.

### Result

The final merged dataset included new weather-related features per match:

| **Feature** | **Description** |
| --- | --- |
| Avg\_Temp\_C | Average temperature on match day (°C) |
|  |  |
|  |  |

These features were later used in predictive modeling to evaluate their influence on the likelihood of Team1 winning.

### Benefit

Integrating weather data allowed the model to simulate real-world match conditions more accurately and enabled deeper insight into how external environmental factors impact cricket match outcomes.

## 2. Feature Engineering

Derived features were introduced to enhance model signal strength. These included binary encodings for home advantage and toss influence, manually assigned ICC team rankings, and their difference (`rank\_diff`). Additionally, a binary target was defined: 1 if Team1 won, else 0.

## 3. Data Transformation

Numerical features were scaled using `StandardScaler` for SVM-based models. One-hot encoding was applied to categorical features for XGBoost, and string labels were label-encoded for compatibility.

## 4. Addressing Class Imbalance

To handle imbalance in match outcomes, class weights were inversely scaled based on class frequency. These weights were passed to training routines.

## 5. Model Training

The following models were implemented:

- TabPFN: A transformer-based probabilistic model trained using scaled features. Auto-selects GPU when available.

- Support Vector Machine (SVM): RBF kernel-based classifier optimized via GridSearchCV over parameters C and gamma.

- XGBoost: Both binary and multiclass classifiers were implemented with `RandomizedSearchCV` over hyperparameters like depth, learning rate, and regularization values.

## 6. AutoML with AutoGluon

AutoGluon Tabular was employed to streamline model selection, tuning, and ensembling. The configuration included two levels of stacking and five-fold bagging. The time budget was constrained to 900 seconds.

# Results

Model performance was evaluated via cross-validation and full-dataset evaluation.

## Evaluation Metrics (Final Stacked Model):

- Accuracy: 0.6512

- Balanced Accuracy: 0.6509

- MCC: 0.3102

- ROC AUC: 0.7204

- F1 Score: 0.6611

- Precision: 0.6258

- Recall: 0.7024

Best performing model: LightGBMLarge\_BAG\_L1

## Model Performance Comparison

| Model | Accuracy | Precision | Recall |
| --- | --- | --- | --- |
| SVM | 0.605 | - | - |
| XGBoost | 0.637 | - | - |
| XGBoost (Alt) | 0.644 | - | - |
| Random Forest | 0.662 | - | - |
| Final Stacked Model | 0.651 | 0.626 | 0.702 |

# Conclusion

The study successfully demonstrates the feasibility of using AutoML techniques like AutoGluon to automate model development in cricket match outcome prediction. By leveraging engineered features and ensemble learning, the model achieved over 65% accuracy on pre-match data alone. Future improvements could involve incorporating dynamic player statistics, match history, and real-time updates.