PREDICTING CRICKET MATCH OUTCOME USING MACHINE LEARNING

Design and Specification proposal
COMP702 - M.Sc. project (2024/25)

Name: Alka Manvayalar Suresh

Student ID: 201805939

Supervisor: Dr Henry Forbes

1. Statement of Ethical Compliance

This project uses only publicly available cricket match data from Cricsheet, with supplementary metadata from Kaggle, and contains no personal or sensitive information. It falls under Category A0 of the university's ethical framework, and I will comply fully with the institution's data storage, processing, and reporting policies. If any new ethical issues arise such as adding user-sourced feedback I will update this statement and consult my supervisor before proceeding.

2. Project Description

An end-to-end system will be developed to predict cricket match outcomes using publicly available ball-by-ball data. Match records will be stored in a relational database and summarized into key statistics such as run-rates, economy figures, and toss win probabilities to capture team form and conditions. These features will feed a suite of classification models whose predictive performance will be compared on unseen seasons. An interactive dashboard will then present win-probability forecasts and model insights, all packaged within a reproducible workflow.

3. Aims and Requirements

3.1 Main Aim

- ❖ **Predictive Modeling:** Develop and benchmark a suite of machine-learning classifiers (logistic regression, random forest and gradient boosting) to forecast cricket match outcomes.

- ❖ **Feature Engineering:** Design and compute match-level statistics (powerplay run-rates, death-over economies, toss win probabilities) that effectively summarize team form and playing conditions.

- ❖ **Interactive Delivery:** Deploy an interactive dashboard presenting feature importance, model comparisons and live win-probability forecasts, underpinned by a reproducible end-to-end pipeline.

3.2 Requirements

Essential

- ❖ Download and store Cricsheet T20 YAML archives and supplementary Kaggle CSV metadata.

- ❖ Implement a Python-based ingestion script to populate `matches` and *`deliveries`* tables in PostgreSQL.

- ❖ Create a *`match_features`* table combining engineered statistics with a binary win/loss label.

- ❖ Train and compare at least three classification models in Python, logging hyperparameters and metrics.

- ❖ Evaluate model performance on one season held out from training, reporting accuracy, ROC-AUC and log-loss.

- ❖ Connect Power BI Desktop to the PostgreSQL backend and publish a live, parameter-driven dashboard.

Desirable

- ❖ Deploy a lightweight Flask API to serve pre-match win-probability predictions.

- ❖ Implement a fourth classification model (e.g. support vector machine or a simple sequential LSTM-based model) to extend the comparative analysis.

4. Key Literature and Background Reading

A foundational end-to-end pipeline for cricket outcome prediction was outlined by Rehman et al. [1], in which raw ball-by-ball data were parsed, aggregated into match-level summaries, and processed through successive modeling stages to generate final forecasts. This architecture directly informs our system design by validating the use of Cricsheet YAML ingestion and feature-engineering workflows. The predictive value of granular delivery-level statistics such as dot-ball percentages and boundary rates was demonstrated by Anik et al. [3], prompting the design of our *`deliveries`* table schema and driving computation of detailed aggregates like powerplay run-rate and death-over economy.

The effectiveness of deep learning in capturing non-linear interactions among cricket features was shown by Agarwal and Bajpai [2], leading to the inclusion of an LSTM-based classifier as an optional extension to explore potential performance gains. A broad comparison of multiple machine-learning techniques across diverse T20 venues and conditions was conducted by Kaur and Sharma [4], justifying our multi-model strategy of benchmarking logistic regression, Random Forest, XGBoost, and a fourth classifier to identify the best performer on unseen seasons.

Patel and Shah [5] demonstrated significant accuracy improvements by blending contextual metadata such as toss decisions, venue characteristics, and head-to-head records with ball-by-ball features. Accordingly, Kaggle-sourced venue and team datasets will be integrated alongside Cricsheet logs to capture these external factors. Finally, Smith and Kumar [6] distilled best practices for crafting interactive dashboards emphasizing clarity, drill-down capability, and scenario-testing controls which will guide the layout and interactivity of our Power BI report, ensuring that complex model insights are accessible to both technical and non-technical users.

5. Development and Implementation Summary

A modular development environment will be established using Python 3.9 (chosen for its mature, well-supported data-science ecosystem) and PostgreSQL 13 (selected for robust SQL features window functions, JSONB support that ease feature engineering). Visual Studio Code will serve as the IDE for its seamless integration with Python linters (flake8) and formatters (Black), ensuring code quality, while Git and GitHub will provide version control and change tracking.

Implementation will proceed in four stages:

1.1 Data Ingestion

- ❖ A Python script will parse Cricsheet's YAML files and insert match metadata into a *`matches`* table and delivery events into a *`deliveries`* table using SQLAlchemy's bulk-insert (to simplify database interactions and benefit from built-in connection pooling).

- ❖ Supplemental Kaggle CSVs will be loaded via *`pandas.read_csv()`* and *`df.to_sql()`* (leveraging pandas' ease of data cleaning) to incorporate contextual metadata.

1.2 Feature Engineering

- ❖ SQL views will compute core aggregates (powerplay run-rate, death-over economy, toss advantage) because pushing computations into the database reduces Python-side load and improves performance.

- For more complex aggregations (e.g. rolling averages), pandas scripts will be used, then results will be materialized into a `match_features` table with a binary win/loss label for straightforward modeling.

1.3 Model Training & Evaluation

- Jupyter notebooks will orchestrate data retrieval via *`pandas.read_sql()`*, model fitting with scikit-learn and XGBoost, and a fourth optional classifier (e.g. SVM or LSTM). Notebooks were chosen for their interactivity and ease of documenting experiments.

- Hyperparameter tuning will be carried out via simple grid or randomized search, because these methods balance exploring parameter space against time constraints.

- Models will be compared on hold-out seasons, with accuracy, ROC-AUC, and log-loss recorded to ensure a fair, quantitative evaluation.

1.4 Visualization & Delivery

- Power BI Desktop will connect via DirectQuery to the `match_features` and prediction tables, allowing live dashboards that reflect the latest model outputs without redundant data exports.

- Dashboard files and supporting scripts will be versioned alongside the codebase, ensuring the entire pipeline from ingestion through to interactive reporting can be reproduced or extended with minimal effort.

6. Data Sources

Cricsheet T20 YAML archives (2009–2025): downloaded from https://cricsheet.org/downloads/, licensed for public use. These files supply detailed ball-by-ball information.

Kaggle T20 metadata: supplementary CSVs containing team standings and venue characteristics, obtained under Kaggle's open-data license.

All data are public and contain no personal or sensitive information. Files will be stored on the university's secure network drive, with automated scripts ensuring read-only access and no external sharing.

7. Testing and Evaluation

A multi-layered approach will be used to verify both the technical integrity of the pipeline and its ability to meet the project's goals. First, unit tests will be written for each parsing and transformation function using *pytest* to feed known YAML and CSV snippets into the ingestion scripts so that any changes in file format or unexpected values are detected immediately. Once individual components are validated, integration tests will be run by passing a small, controlled dataset (with pre-computed aggregates) through the entire ingestion, feature engineering, and storage pipeline; the outputs (match counts, delivery counts, and key statistics) will be compared against expected results to confirm the pieces work together seamlessly.

For model evaluation, a hold-out season strategy will be employed: algorithms will be trained on data through 2018 and tested on 2019–2020 data. Performance will be measured using accuracy, ROC-AUC, and log-loss, ensuring that the models generalize beyond their training data. In addition, calibration plots will be generated by binning predicted win probabilities (e.g. 60–70%) and verifying that the actual win rates in those bins closely match the model's forecasts this step is crucial for assessing whether probability outputs are reliable.

Finally, to evaluate real-world usability, beta testing sessions will be organized with two peers. Testers will be given concrete tasks such as applying filters to the Power BI dashboard and interpreting feature-importance charts and asked to rate clarity, responsiveness, and overall user experience. Their anonymized feedback will guide final adjustments to the dashboard layout and interactive controls, ensuring that the finished system not only performs correctly but also delivers actionable insights in an intuitive format.

8. Project Ethics and Human Participants

No human-subject data will be collected or processed; all inputs are public sports records. If informal usability testing is conducted, participants will be briefed, and their comments will be anonymized and aggregated. This falls within Category A0 (no personal data). Should any future evaluation require personal or feedback data, ethical approval will be sought and documented in consultation with the supervisor before proceeding.

9. BCS Project Criteria Mapping

- **Practical and Analytical Skills:**The project applies hands-on database design in PostgreSQL, Python scripting for data ingestion and transformation, and the use of statistical and machine-learning libraries to real cricket data. These activities directly leverage and extend the core analytical techniques covered in the degree programme.
- **Innovation and Creativity:**Custom feature engineering separating powerplay and death-over statistics, weighting toss outcomes by venue history and the seamless integration of SQL, Python, and Power BI into one pipeline demonstrate original thinking and novel combinations of existing tools.
- **Synthesis and Evaluation**:Data pipelines, SQL-based aggregations, model training, and interactive dashboards are brought together into a cohesive workflow. Each component is evaluated through unit/integration tests and quantitative metrics (accuracy, ROC-AUC, log-loss) to ensure technical soundness and practical value.
- **Meeting a Real Need**:By producing pre-match win-probability forecasts in an accessible dashboard, the project addresses the growing demand for data-driven insights among fans, fantasy-league players, and team analysts, filling a clear gap in cricket analytics resources.
- **Self-Management:**A detailed Gantt chart, modular code structure, and Git-based version control provide a roadmap and progress-tracking mechanism. Weekly milestones and regular progress reviews ensure the project remains on schedule and scope is managed effectively.
- **Critical Self-Evaluation:**The final dissertation will include reflections on model limitations such as overfitting risks, data sparsity in certain venues, and performance trade-offs and propose concrete improvements, demonstrating a thoughtful, iterative approach to the research process.

10. Project Plan (Gantt Chart)

Task	Start Week	End Week
Literature Review	Week 1	Week 3
Data Ingestion and Setup	Week 2	Week 4
Feature Engineering	Week 3	Week 5

Dashboard development	Week 5	Week 6
Model Training	Week 6	Week 7
Evaluation Metrics	Week 8	Week 9
Dissertation Drafting	Week 9	Week 12

11. Risks and Contingency Plans

Risk	**Contingency Plan**	**Likelihood**	**Impact**
Data parsing errors (YAML/CSV)	Implement unit tests for parser functions; validate incoming files against schema; retain raw backups.	Medium	High
Model overfitting or poor generalisation	Use season-based cross-validation; apply regularisation; compare multiple algorithms; monitor validation metrics.	High	Medium
Running out of time	Prioritise essential deliverables (ingestion, baseline model, dashboard); defer optional features; review progress weekly.	Medium	High
Database or script failures	Automate nightly database backups; containerise ingestion environment; employ transaction roll-backs for safety.	Low	High
Power BI connectivity or refresh issues	Pre-aggregate heavy queries into materialised views; cache static snapshots for critical visuals; provide fallback exports.	Medium	Medium
Hardware failure or data loss	Keep code and data in Git/GitHub; store backups on university network and external drive; test recovery procedures.	Low	High
Unexpected data	Modularise ingestion logic;	Medium	Medium

schema changes	implement automated schema-drift checks; document and adapt to new formats promptly.		

12. References

[1] Z. U. Rehman, M. M. Iqbal, H. Safwan and J. Iqbal, “Predict the Match Outcome in Cricket Matches Using Machine Learning,” May 2022. [Online]. Available: https://www.researchgate.net/publication/373690614_Predict_the_Match_Outcome_in_Cricket_Matches_Using_Machine_Learning [Accessed: Jun. 19, 2025].

[2] Agarwal, S. & Bajpai, N. (2019). *A Novel Approach to Predicting Cricket Match Outcomes Using Neural Networks*. arXiv preprint. Available at: https://arxiv.org/abs/1906.04567 [Accessed 19 June 2025].

[3] Anik, A.I., Yeaser, S., Hossain, A.G.M.I. & Chakrabarty, A. (2018). *Player's Performance Prediction in ODI Cricket Using Machine Learning Algorithms*. In *2018 4th International Conference on Electrical Engineering and Information & Communication Technology (ICEEICT)* (pp. 500–505). IEEE. Available at: https://ieeexplore.ieee.org/document/8628118 [Accessed 19 June 2025].

[4] Kaur, G. & Sharma, P. (2021). *A Comparative Study of Machine Learning Techniques in Cricket Match Outcome Prediction*. In *Advances in Intelligent Systems and Computing*, Vol. 1200, pp. 225–236. Springer. Available at: https://link.springer.com/chapter/10.1007/978-3-030-54926-7_15 [Accessed 19 June 2025].

[5] Patel, R. & Shah, V. (2025). *Predictive Analysis of T20 Cricket Outcomes Using Machine Learning Techniques*. *International Research Journal of Modern Engineering and Technology Science*, January 2025 Issue. Available at: https://www.irjmets.com/uploadedfiles/paper//issue_1_january_2025/66640/final/fin_irjmets1737781673.pdf [Accessed 19 June 2025].

[6] Smith, J. & Kumar, A. (2019). *Data-Centric Prediction of Sports Outcomes: A Case Study in Cricket*. *Applied Computing and Informatics*, 17(1), pp. 120–130.

Available:https://www.emerald.com/insight/content/doi/10.1016/j.aci.2019.11.006/full/html [Accessed 19 June 2025].