



MCAST

Evaluating Machine Learning Algorithms for Predicting Top 3 Finishers in Virtual Motorsport Racing Using Real-Time Telemetry Data and Driver Performance History

Antoine Spiteri

Supervisor: Christopher Farrugia

June - 2025

**A dissertation submitted to the Institute of Information and Communication
Technology in partial fulfilment of the requirements for the degree of BSc (Hons)
Multimedia in Software Development**

Authorship Statement

This dissertation is based on the results of research carried out by myself, is my own composition, and has not been previously presented for any other certified or uncertified qualification.

The research was carried out under the supervision of Christopher Farrugia

.....

Date

.....

Signature

Copyright Statement

In submitting this dissertation to the MCAST Institute of Information and Communication Technology, I understand that I am giving permission for it to be made available for use in accordance with the regulations of MCAST and the Library and Learning Resource Centre. I accept that my dissertation may be made publicly available at MCAST's discretion.

.....

Date

.....

Signature

Acknowledgements

I want to express my heartfelt gratitude to those who have supported me throughout this journey.

To my mother, thank you for your unwavering guidance, encouragement, and love, which have been the foundation of my life throughout this process. To my two brothers, Daniel (In spirit) and Matthew, your support during difficult times has been a source of strength. To my grandmother, your emotional warmth and kindness have been a comfort beyond measure. To my late grandfather, your memory has fueled my determination and inspired me to persevere. To my father, your wisdom and perspective have been invaluable in shaping my outlook, as well as that of all my friends and family members who supported me throughout.

Finally, I am deeply grateful to my supervisor and mentor, whose expert guidance and encouragement have been instrumental in the completion of this dissertation.

To all of you, this achievement is as much yours as it is mine. Thank you.

Abstract

Competitive online motorsports has evolved into a prominent esports platform, where performance hinges on driving skill, strategic decisions, unpredictable events, and dynamic track conditions. However, accurately forecasting podium finishers mid-race remains a technical challenge due to the complexity and real-time variability of these factors. This dissertation explores whether machine learning models can effectively analyse a blend of historical and real-time telemetry data to predict the top three finishers in live iRacing competitions, aiming to enhance both strategic decision-making and spectator engagement.

A hybrid pipeline was developed that integrates long-term driver statistics, fetched via the iRacing REST API, with real-time telemetry from the iRacing SDK. This yielded a comprehensive dataset exceeding 800 records, each combining structured past data with decayed and bucketed live telemetry signals. Feature engineering methods such as decay-weighted bucket windows, lap sampling, and the custom target “RankscoreAdjusted” metric were applied to transform the raw telemetry into predictive signals.

Three models were evaluated: XGBoost (a gradient-boosted tree model), Random Forest (a bagged decision tree model), and RankNet (a pairwise neural ranking model). These models were tested on 18 unseen races and compared using podium accuracy, Spearman rank correlation, and supplementary metrics. RankNet exhibited the best overall ranking performance and generalisation ability, achieving 95% top-3 accuracy with minimal overfitting and due to its ability to adjust prediction mid-lap. XGBoost attained perfect accuracy on certain race snapshots but showed higher variance between training and test results. Random Forest proved helpful in estimating point-based positions, though its inference time and rank-order consistency were less suitable for real-time deployment.

Results confirm the value of combining historical and live data to enhance predictive reliability and highlight the strengths of rank-based models, such as RankNet, for ordinal prediction tasks in motorsports. Interpretability was supported through SHAP analysis, which identified key factors such as NormDriverGapToLeader, consistency, and stable front-runner rankings as strong indicators of podium outcomes. The asynchronous live prediction loop achieved a 3-second update cycle, demonstrating potential for integration into live overlays and strategic dashboards.

While data limitations, such as restricted weather diversity and missing telemetry on tyre wear and brake temperatures, constrain generalisation, the research establishes a solid foundation for future work. It supports the development of data-driven coaching tools, strategic simulators, and real-time esports analytics that enhance both competitive fairness and viewer experience.

Keywords: Virtual Motorsports, iRacing, Real-Time Prediction, RankNet, XGBoost, Random Forest, Telemetry Analytics, Feature Engineering, Esports AI, Motorsports

Table of Contents

Authorship Statement	i
Copyright Statement	ii
Acknowledgements	iii
Abstract	iv
List of Figures	ix
List of Abbreviations	x
1 Introduction	1
1.0.1 Research aim	1
1.0.2 Objectives	2
1.0.3 Research questions	2
1.0.4 Hypothesis	3
1.0.5 Dissertation roadmap	3
2 Literature Review	4
2.1 Background	4
2.1.1 Overview of Predictive Modelling	4
2.1.2 Importance in Competitive Virtual Environments	4
2.2 Introduction to Predictive Modelling in Virtual Motorsports	5
2.2.1 Supervised vs. Unsupervised Learning	5
2.2.2 Unified Metrics for Supervised and Unsupervised Learning	9
2.2.3 Popular Algorithms in Predictive Modelling	10
2.3 Data Sources and Feature Engineering	13
2.3.1 Historical vs. Real-Time Data	13
2.3.2 Key Features in Racing Data	15
2.3.3 Feature Engineering Techniques	16
2.4 Challenges in Virtual Data Modelling	18
2.4.1 Data Limitations and Quality	18
2.4.2 Handling Complexities in Real-Time Data	19
2.4.3 Model Interpretability and Bias	20
2.5 Comparative Studies in Predictive Modelling for Virtual Sports	21
2.5.1 Existing Research in Esports and Simulation	22
2.5.2 Identified Gaps and Research Direction	25
2.6 Applications of Predictive Modelling in Real Racing	26
2.6.1 Case Studies in Real-World Motorsports	27
2.6.2 Comparison to Virtual Racing Predictions	27

2.6.3	Potential Impact on Player Experience and Strategy	28
2.7	Conclusion	28
3	Research Methodology	30
3.1	Introduction to Methodology	30
3.2	Research Design	30
3.2.1	Study type, methods and choices	31
3.2.2	Limitations of Research Design	35
3.3	Data Collection, Pre-processing, feature engineering and retrieval .	40
3.4	Model choice justification, Training and Validation	45
3.4.1	Model Choice Justification	45
3.4.2	Training and validation	46
3.5	Ethical approaches	52
3.6	Conclusions	53
4	Analysis of Results and Discussion	55
4.1	Introduction to Analysis	55
4.2	Findings and Interpretation	56
4.2.1	Results	56
4.2.2	Overall best practical predictor due to its mid-lap capability and stronger generalisation, despite XGBoost's slightly higher top3 accuracy and Spearman	68
4.2.3	Enhanced Real-Time Adaptability of RankNet	69
4.2.4	XGBoost stability and reliability across races	71
4.2.5	Random Forest: Stable Point-Estimate Specialist	72
4.3	Comparison with Previous Research	73
4.3.1	Alignment with Prior Evidence on Neural Rankers	74
4.3.2	Real-Time Adaptability versus Lap-Aggregated Stability . . .	74
4.3.3	XGBoost's Cross-Race Robustness in Structured Contexts .	75
4.3.4	The Suitability of Random Forests for Point Estimation over Ranking Tasks	76
4.3.5	Why Some Results Diverge from Earlier Studies	76
4.3.6	Contribution to Current Understanding	77
4.4	Implications of the Findings	78
4.5	Limitations of the Study	80
5	Conclusion	83
5.1	Summary of Key Findings	83
5.2	Answers to the Research Questions	83
5.3	Hypothesis	84
5.4	Recommendations for Future Work	85
5.5	Final Reflection	86
	List of References	87
A	Data Collection	95

B	Database Schema	100
C	Model Training, and Evaluation	102
D	Live Prediction	104
E	Feature engineering and database to a flat file	106
F	Multimedia Supplemental Materials	110

List of Figures

2.1	supervised learning diagram	5
2.2	unsupervised learning diagram	7
2.3	gradient boosting diagram	11
2.4	random forest diagram	12
2.5	neural network diagram	14
4.1	Comparison of model metrics across XGBoost, Random Forest, and RankNet.	57
4.2	XGBoost Model learning rate	58
4.3	Random forest Model learning rate	59
4.4	RankNet Model learning rate	60
4.5	Model differences	61
4.6	XGBoost feature importance	62
4.7	Random Forest feature importance	63
4.8	RankNet feature importance	64
4.9	XGBoost feature importance (bar graph)	65
4.10	Random Forest feature importance (bar graph)	66
4.11	RankNet feature importance (bar graph)	67
A.1	Data Collection Pipeline	96
A.2	Historical Data Gathering	97
A.3	Live Data Collection and Processing part 2	98
A.4	Live Data Collection and Processing part 2	99
B.1	Database Schema	101
C.1	Model Training Workflow and Model Performance Evaluation	103
D.1	Live Prediction	105
E.1	Data Extraction from Database	109

List of Abbreviations

API	Application Programming Interface
ARIMA	Autoregressive Integrated Moving Average
CV	Cross-Validation
DB	Database
DT	Decision Trees
FC	Fully Connected
GAN	Generative Adversarial Network
GBT	Gradient-Boosted Trees
GPU	Graphics Processing Unit
IoT	Internet of Things
JSON	JavaScript Object Notation
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MSE	Mean Squared Error
PCA	Principal Component Analysis
PyTorch	Python deep-learning framework
R²	R-squared
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
SDK	Software Development Kit
SHAP	Shapley Additive ExPlanations
SMOTE	Synthetic Minority Over-Sampling Technique
TSSTN	Two-Stage Spatial-Temporal Network
VSE	Virtual Strategy Engineer
YAML	YAML Ain't Markup Language
XGBoost	eXtreme Gradient Boosting

Chapter 1: Introduction

Competitive virtual motorsport has matured from a casual simulation into a professionally regulated esports ecosystem that attracts real-world racing teams, licensed drivers, and global audiences. Platforms such as *iRacing* deliver telemetry streams and publish comprehensive historical race archives; these data sources can generate detailed driver data regarding performance. This presents an opportunity for data-driven performance forecasting, although the academic literature on virtual racing analytics remains largely unexplored. Most studies are confined to post-race aggregate data or discuss race-outcome prediction as a binary “win/loss” distinction, without reference to the ordinal ranking that separates a first-place finish from a second or third. Most importantly, none have benchmarked gradient-boosted, bagged-tree, and neural pair-wise ranking algorithms on a dataset that combines long-term driving statistical data and real-time telemetry, as this research knowledge is lacking. This appears to be an under-explored area, although relevant work may exist undetected. This study addresses this gap.

1.0.1 Research aim

The central aim is to compare and evaluate three machine learning models—XGBoost, Random Forest, and RankNet—to rank the top three finishers (podium drivers) in ongoing *iRacing* competitions from a combined feed of static performance histories and real-time telemetry data. This work aims to demonstrate how each machine learning model best balances accuracy, real-world responsiveness, stability,

and interpretability to rank in virtual motor sports.

1.0.2 Objectives

This project has multiple objectives to fulfil: a data pipeline to extract telemetry and historical driver data for data collection into the database (DB). Additionally, feature engineering involves creating composite variables that more effectively represent race dynamics than individual features. Furthermore, model implementation also requires fine-tuning and validation metrics, including group k-fold cross-validation, bootstrapping, and learning.

Furthermore, assessing each model's performance entails looking at metrics, such as validation, live predictions, and behavioural observations. Finally, this research will synthesise how each algorithm's strengths are implemented for effective use in practical race-strategy cases, developing recommendations for future research and esports development.

1.0.3 Research questions

- **Research question 1:** *Which evaluated algorithms (RankNet, XGBoost, or Random Forest) achieve the highest predictive accuracy for podium finishers when leveraging integrated real-time telemetry and historical driver performance data from the iRacing platform?*
- **Research question 2:** *How robust and consistent are each algorithm's predictions across varying race scenarios, including track layouts, driver skill levels, and dynamic race incidents?*

- **Research question 3:** *What are the most critical predictive features influencing podium placements, and to what extent do RankNet, XGBoost, and Random Forest offer transparent interpretability for practical strategic applications?*

1.0.4 Hypothesis

Integrating real-time telematic data and historic driver statistics into rank-aware models (e.g., RankNet) will produce more prediction accuracy for podium rankings than traditional regression and classification methods.

1.0.5 Dissertation roadmap

This dissertation begins with Chapter 2, which provides a review of previous research in the literature, including motorsports analytics, ranking theory, and real-time esports forecasting. This is preceded by Chapter 3, which outlines the methodology, describing data collection, feature engineering, model configuration, validation procedures, and ethical protections. In addition, Chapter 4 presents the findings and discussion, providing experimental results, referencing them against earlier studies, and examining the implications for both practice and theory.

Finally, Chapter 5 summarises the contributions, covers the research questions, recognises limitations, and suggests future work. The appendices will also include high-level flowcharts of code and formulas for feature engineering, providing further reference information.

Chapter 2: Literature Review

2.1 Background

2.1.1 *Overview of Predictive Modelling*

Virtual Motorsports have evolved substantially in recent decades, transforming from entertaining video games into complex, highly realistic racing simulations utilised by professional drivers and esports competitions. Advances in game development, such as realistic physics engines, sophisticated data analytics and integration with real-world motorsport telemetry in games like iRacing, Assetto Corsa and more, have bridged the gap between virtual realism and professional driving [1] [2] [3].

2.1.2 *Importance in Competitive Virtual Environments*

Predictive modelling can enhance realism and competitiveness by combining data-driven racing strategies with predictive modelling that refines realism and racing strategies. It can simulate racing strategies and refine them in real-time by analysing race data, just as in professional settings. Additionally, it can enhance fan engagement by making it more immersive and interactive. This improvement stems from the subsequent machine-learning techniques: random forest, gradient boosting, and neural networks. Their capacity to manage high-frequency data facilitates timely decision-making and enhances accuracy through continuous refinement [1] [2] [3].

2.2 Introduction to Predictive Modelling in Virtual Motorsports

2.2.1 Supervised vs. Unsupervised Learning

Supervised and unsupervised learning are two distinct machine learning techniques that differ in their approach to data processing, learning methods, and objectives. Both approaches serve critical roles in data analysis, with different strengths and weaknesses depending on the scenario [4] [5].

1. Supervised Learning:

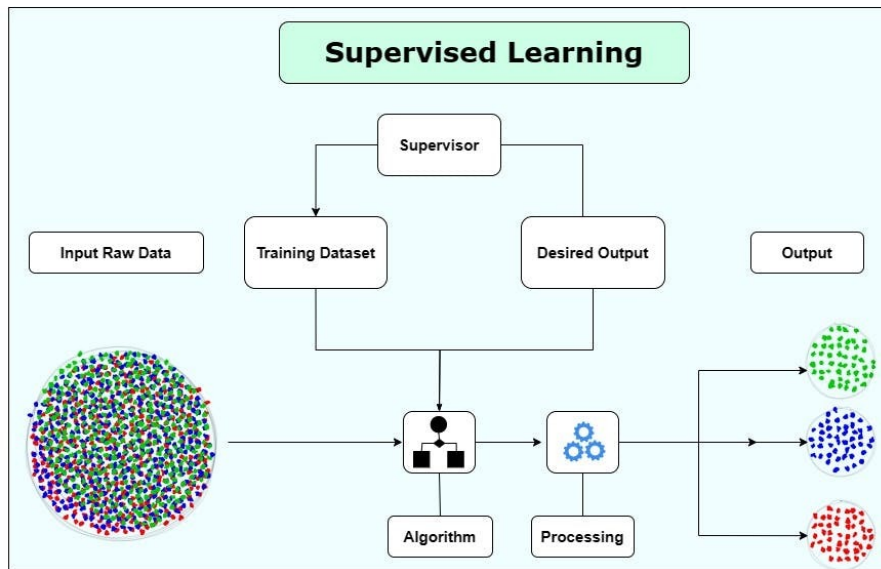


Figure 2.1: Supervised learning diagram. Source: [6]

The objective is to analyse and examine the data to discover patterns in the labelled data, where the inputs (features) in the training data are utilised to correspond to the labelled data. Labelled data is the target data for the expected output; each data point in the training is exposed to analyse its relationship or pattern with the labelled data. During training, the model

updates its parameters based on previously learned information to minimise prediction errors on the training set. This is achieved through metrics such as precision, accuracy, recall, and F1-score for classification, as well as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R^2 for regression. Moreover, cross-validation ensures generalisation for unseen data [4] [5].

This is important because it is an iterative learning process that refines the model parameters to improve performance metrics. The Primary type of supervised learning is regression, which is used to predict continuous numerical values. Classification categorises inputs to represent a binary output variable, ranking ranks its inputs based on relevance, and sequence prediction predicts a sequence or the following item of a sequence. Applications span esports analytics, IoT-security anomaly detection and healthcare diagnostics, each leveraging labelled data for precise, actionable predictions, [5,7].

Among its strengths, one of the most notable is its high accuracy, due to the sufficient labelled data, which enables robust predictive performance. Another is the ability to deliver findings into features that influence outcomes, as well as the broad applicability discussed previously. While effective in specific scenarios, it struggles with the need for large amounts of labelled data, which can be time-consuming. Data without proper validation may overfit the model, and the inability to discover patterns without predefined labels [8] [9].

2. Unsupervised Learning:

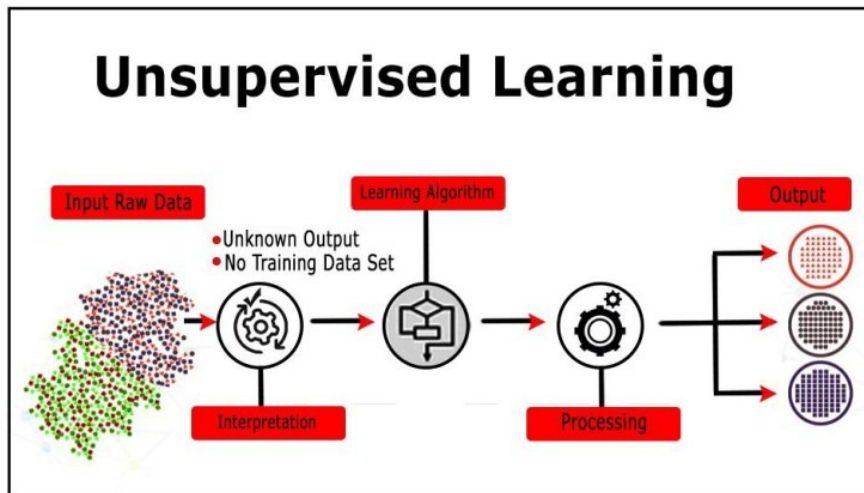


Figure 2.2: Unsupervised learning diagram. Source: [10]

In contrast, its goal is to analyse the data without any target outputs and to acquire observations of its structure and relationships within the data. It provides perspectives on the data organisation instead of predicting outcomes in contexts where labelled data is unavailable or to explore and understand the data structures it is used in applications in Cluster Analysis, like in healthcare to group patient data to identify health patterns, thermal imaging to detect anomalies in heat distribution in industrial machinery and many more [4] [5]. **Unsupervised learning is categorised into distinct types**, each with unique characteristics to reveal underlying patterns in data. Each has its methods to measure the model performance due to the lack of labelled data for straightforward performance metrics like accuracy, f1 score and more [5]:

- **Clustering:** To group data points into clusters based on similarity. Examples include K-Means, hierarchical clustering, and others. Typical applications include social network analysis, customer segmentation, and

more [4] [5].

- Dimensionality Reduction reduces the number of features in the dataset while retaining the most critical information. Techniques like Principal Component Analysis (PCA) and t-SNE are used for data visualisation, noise reduction, and feature extraction [4] [5].
- Association Rules: Identifies relationships between variables in large datasets using Apriori and Eclat algorithms, such as market basket analysis, recommendation systems, and more [5].
- Anomaly Detection: identifying data points that deviate significantly from the norm. Examples include Isolation Forest and One-Class SVM applied in fraud detection, network security, and other domains [4].

One of the key strengths of this approach is no label dependence; it can function without labelled data, making it ideal for specific scenarios where labelled datasets are scarce, functionality to provide insights into the data structure and the versatility of domains without labelled data [4] [5]. However, this method also has limitations, such as increased difficulty in interpretability due to a lack of labelled data and predefined objectives, a lack of standardised performance metrics like accuracy, recall and more, and sensitivity to parameters in specific unsupervised algorithms that may affect results [4] [5].

2.2.2 *Unified Metrics for Supervised and Unsupervised Learning*

Machine learning tasks, whether supervised or unsupervised, rely on appropriate metrics to evaluate performance. Supervised metrics focus on predictive accuracy using labelled data, while unsupervised metrics assess how well the model identifies patterns in unlabelled data [4,5].

Standard supervised metrics include Accuracy, Precision, Recall, F1 Score, Mean Squared Error (MSE), R-squared, and Cross-Validation. Accuracy provides a direct measure of correct versus incorrect predictions. Precision, Recall, and F1 Score are crucial for imbalanced datasets: Precision reduces false alarms, Recall minimises missed cases, and the F1 Score balances both [4,5]. MSE and R-squared are applied in regression tasks to measure error magnitude and model fit, respectively [11,12]. Cross-validation ensures consistent performance on unseen data, supporting generalisability and reducing overfitting.

For unsupervised learning, common metrics include Silhouette Score, Inertia, Reconstruction Error, and Entropy-Based Metrics. Silhouette Score evaluates the cohesion and separation of clusters [5]. Inertia measures compactness via Within-Cluster Sum of Squares, while Reconstruction Error assesses how well models like PCA retain essential information after dimensionality reduction. Entropy-Based Metrics evaluate the randomness and clarity of cluster assignments, with lower entropy indicating more meaningful groupings [4].

In summary, these metrics help assess model success or failure in both labelled and unlabelled contexts and guide further optimisation and development [5].

2.2.3 Popular Algorithms in Predictive Modelling

Algorithms can be broadly classified into several types based on their functionality and application. Furthermore, there is no universal solution, as algorithms are specifically designed to handle various challenges, including data type variability, relationship complexity, scalability, resistance to overfitting and underfitting, interpretability and accuracy trade-offs, and different goals. Ultimately, these different algorithms ensure a tailored solution is available for each unique problem [5] [4].

1. Gradient Boosting: The core mechanics of this model build sequentially, where each new model corrects errors made by the previous ones. It minimises the loss function by utilising gradient descent with weak learners, such as decision trees, trained on the mistakes made by the model. These predictions are updated regularly, scaled by the learning rate, and regularised to prevent overfitting. It is widely used for regression, classification, and ranking tasks for Esports, IoT security, and many more applications [11] [12] [4].

Its strength lies in its high accuracy and flexibility with data types, as well as its capacity to manage complex data relationships. It also includes features essential for interpretability [11] [5]. In contrast, it is computationally expensive and sensitive to parameter tuning, prone to overfitting without proper regularisation, and interpreting individual predictions becomes chal-

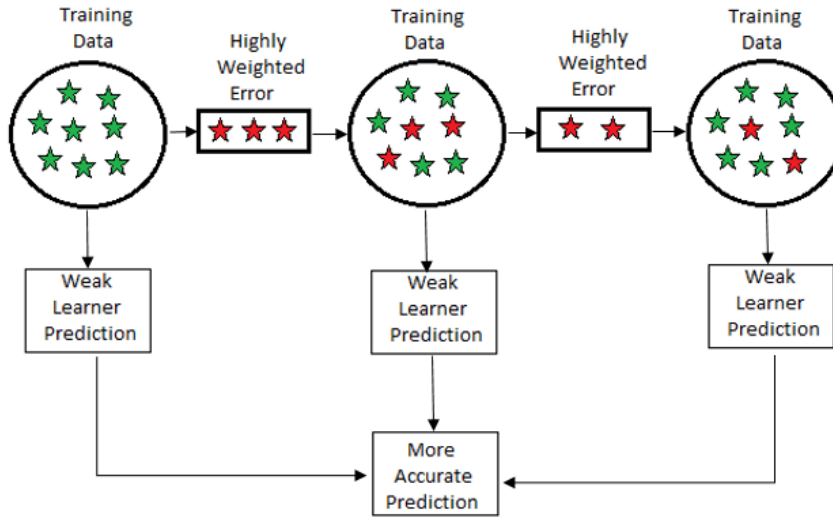


Figure 2.3: gradient boosting diagram. Source: [13]

lenging in large datasets. Additionally, it is slower compared to simpler models, especially in real-time predictions [12].

2. Random Forest: Consists of numerous decision trees trained independently, each singular tree being a weak learner but providing a stronger model combined. During training, each tree is built by randomly selecting a sample from the dataset, reducing variance by averaging predictions across many trees. A random subset of features is considered whenever a tree splits to reduce the correlation among trees. In applications for classification, the majority vote of the tree outputs determines the final prediction; for regression, the average of the tree outputs is used as the final prediction. For ranking tasks, each decision tree predicts a score for each instance [4] [5].

The strengths lie in its robustness against noisy data, missing values, and

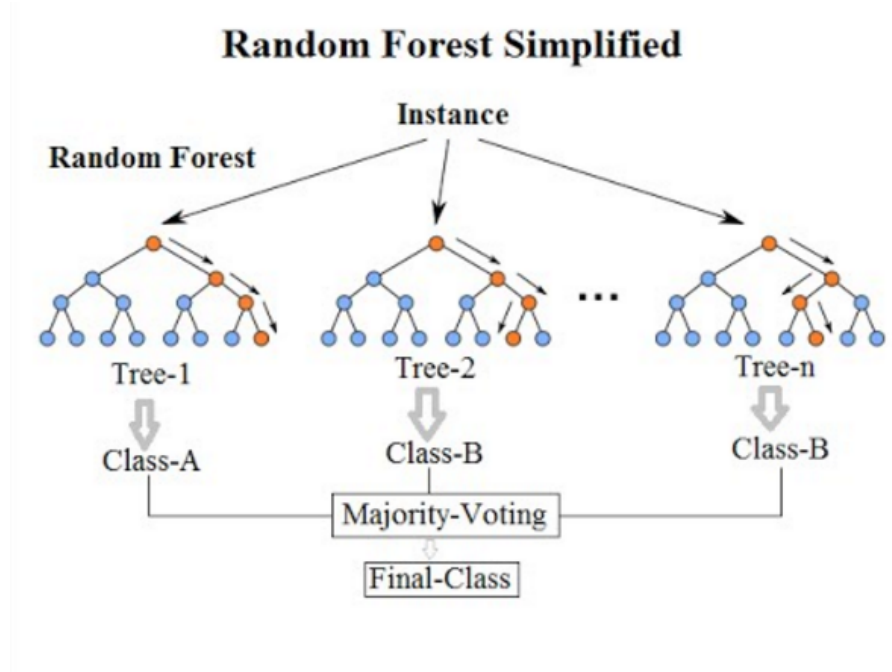


Figure 2.4: Random forest diagram. Source: [14]

the ability to handle non-linear relationships. Moreover, it demonstrates a high degree of accuracy and is very resistant to noisy data due to averaging the results of multiple trees. In contrast, its limitations are slower processing speeds in large datasets or high-dimensional input spaces, and predictions are more challenging to interpret [11] [5] [12].

3. **Neural Networks:** Inspired by the mechanisms of the human brain, neural networks process data through layers of interconnected nodes, similar to neurons, to extract patterns and relationships in complex datasets. The core mechanics involve forward propagation, backpropagation, and iterative training. During forward propagation, data moves sequentially through the input layer, the hidden layers, and finally, the output layer, where each node performs weighted calculations followed by an activation function. Moreover,

backpropagation calculates the error between predictions and actual outputs, tracing back this error through the network to adjust the weights through optimisation, utilising algorithms like Stochastic Gradient Descent (SGD). In conclusion, this methodology ensures optimal efficiency for the model parameters to mitigate the loss function, such as Mean Squared Error for regression, Cross-Entropy for classification, and pairwise losses for ranking [11] [4] [12].

.

Their primary benefits are highlighted in their high adaptability, as they can model complex, non-linear relationships in data; automation, as they can extract and learn features without manual feature engineering; and high scalability, making them suitable for large datasets. Conversely, they are highly computationally demanding, requiring high-end hardware during training, substantial amounts of labelled data for practical training, complex interpretability challenges in individual predictions, and a higher risk of overfitting without proper regularization techniques [5] [4].

.

2.3 Data Sources and Feature Engineering

2.3.1 Historical vs. Real-Time Data

Predictive modelling relies on historical and real-time data, each serving distinct purposes. Historical data, such as past race results and driver statistics, form the foundation for training machine learning models, enabling the recognition

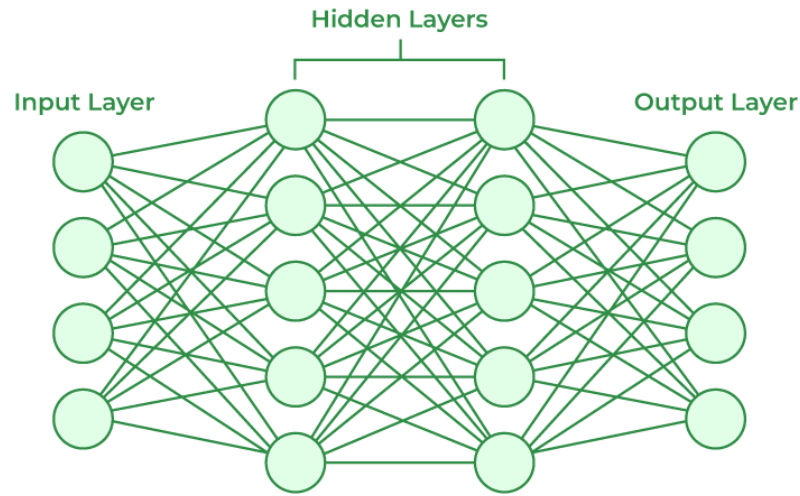


Figure 2.5: neural network diagram. Source: [15]

of patterns and trend analysis. For instance, [16] demonstrated how historical data guides tyre compound and pit-stop strategies in the Virtual Strategy Engineer (VSE). Similarly, [17] showed that historical traffic data enhances real-time deep reinforcement learning systems, improving adaptation and route optimisation in dynamic conditions.

Historical and real-time data are interdependent, with each addressing specific challenges. Historical data provides context for real-time insights, while real-time data adds adaptability to evolving conditions. However, real-time data introduces challenges such as latency and noisy streams, requiring techniques like filtering and anomaly detection to maintain accuracy, as demonstrated by [18]. Scalability is another issue, as processing large telemetry streams demands advanced computational systems.

The synergy between these data types significantly enhances predictive accuracy. Studies like [16] demonstrate how integrating historical data with real-time telemetry enhances decision-making in dynamic racing scenarios. Similarly, [19]

showed that hybrid models combining CNNs and RNNs leverage the strengths of both data types, boosting adaptability and precision in rapidly changing conditions. By integrating these approaches, predictive models achieve superior performance in motorsport analytics.

2.3.2 Key Features in Racing Data

Effective predictive modelling in racing hinges on leveraging key features, such as historical data, real-time metrics, and external factors. Historical data, including past race outcomes and driver profiles, reveal performance patterns and environmental impacts. Metrics such as average lap time variance quantify driver consistency, a strong predictor of reliability. [16] highlighted the importance of experience, starting position, and team dynamics in predicting NASCAR outcomes.

Lap times capture performance variability influenced by driver skill, tyre wear, fuel load, and track conditions. [16] demonstrated their value in optimising pit stop strategies. Track conditions, weather and surface grip directly affect strategies, as seen in rain-altered races like the Monaco Grand Prix. [17] emphasised environmental factors' impact on decision-making [17].

Tire degradation, caused by friction and heat, reduces grip and braking efficiency, influencing lap times and pit stops. Predictive algorithms estimate tyre wear to optimise timing, as [16] illustrated. Fuel levels, while format-dependent, impact weight and performance, and strategic fuel use is often decisive in endurance races.

Positional data and driver gaps provide real-time insights. Driver rank and gaps guide overtaking or conservative strategies, with smaller gaps encouraging

aggressive manoeuvres. [17] demonstrated the utility of such data for predictive models.

Incidents like crashes or safety cars add unpredictability, disrupting race flow and requiring quick adjustments. Safety cars, for example, can enable advantageous pit stops. [19] noted the importance of such unpredictable, random events, referred to as stochastic events, in dynamic modelling.

Incorporating key features like driver history, lap times, track conditions, tyre degradation, fuel levels, positional data, and safety factors creates a comprehensive understanding of race dynamics. Historical data provides context, while real-time metrics refine predictions during races. Machine learning algorithms integrate these features into adaptive models for unpredictable motorsport scenarios.

Cross-referencing studies, such as [16], highlight the synergy between research and practical applications. Leveraging these features enhances strategy optimisation, decision-making, and performance predictability in high-stakes racing.

2.3.3 Feature Engineering Techniques

Feature engineering is crucial for transforming raw racing data into actionable inputs for predictive models. It reveals more profound insights into racing dynamics, enabling more innovative strategies and accurate race predictions in dynamic, multidimensional environments.

Creating relevant metrics, such as average lap time improvement or tyre degradation rates, highlights performance trends that raw data might obscure. For instance, [16] demonstrated how combining tyre degradation with lap trends optimised pit-stop strategies. Adjusting lap times for track difficulty normalises data,

ensuring fair driver comparisons while minimising the impact of outliers.

Temporal segmentation divides races into distinct phases to track strategic shifts like pre-pit and post-pit segments. [16] highlighted how this improved predictions during critical race moments. Techniques like rolling windows reveal short-term trends, such as average times over recent laps, offering a clearer view of evolving performance.

Dimensionality reduction simplifies telemetry data, focusing on key variables like speed, braking, and tyre wear while reducing the risk of overfitting. Techniques like Principal Component Analysis (PCA) and feature selection algorithms, such as recursive feature elimination, streamline data processing while preserving predictive power [17].

Motorsport-specific metrics, such as tyre degradation rates or performance consistency, provide vital context for decision-making. Metrics like lap time variance offer insights into driver reliability, as [16] demonstrated in trajectory tracking. Similarly, real-time aggregation techniques, such as dynamic braking, acceleration, and cornering data summaries, reveal driver behaviour and track conditions [17].

Feature encoding ensures categorical data is compatible with machine learning models. For instance, one-hot encoding works well for weather conditions, while ordinal encoding preserves ranking order for features like grid positions or driver standings [17].

Handling missing or noisy data ensures the accuracy of the model. Techniques such as mean or median imputation address data gaps, while Kalman filters predict missing values in real-time. Detecting and correcting outliers, such as

erroneous speed readings, further maintains data integrity [20].

By leveraging these techniques, feature engineering transforms raw data into actionable insights, enabling the development of robust predictive models tailored to the complexities of motorsport.

2.4 Challenges in Virtual Data Modelling

2.4.1 Data Limitations and Quality

In data-driven modelling, the accuracy and reliability of predictions depend heavily on the quality and availability of input data. Key challenges include data sparsity, inconsistencies, and biases, all of which significantly impact model performance.

Data sparsity occurs when datasets are insufficient or incomplete, standard in specialised fields such as medical imaging or niche consumer behaviour. This limitation restricts a model's generalisation ability, leading to problems like overfitting or poor performance on unseen cases. For example, models trained on limited customer transaction data often fail to accurately predict purchasing behaviour. Addressing this issue involves data augmentation, which generates synthetic examples to enhance training datasets, and transfer learning, where pre-trained models from related domains are adapted to compensate for sparse data [21,22].

Data inconsistencies, such as errors, noise, and imprecision, also undermine the predictive power of models. Challenges include label noise, where incorrect labels confuse the model during training, sampling bias, where datasets fail to

represent the diversity of the target population, and data poisoning, where malicious data is introduced to degrade model accuracy. These issues can be mitigated through rigorous preprocessing methods, including data cleaning, validation, and integration [21,22]. Tools such as Generative Adversarial Networks (GANs) and Synthetic Minority Over-Sampling Techniques (SMOTE) help balance datasets and reduce bias [21].

Bias and poor data quality can exacerbate societal inequities and erode trust, particularly in high-stakes applications such as healthcare and finance. For instance, facial recognition systems trained predominantly on lighter-skinned individuals often misclassify darker-skinned faces [23,24]. These challenges underscore the need for robust training techniques and enhanced validation processes that are tailored to specific domains and data challenges.

While modern approaches address many of these limitations, their effectiveness remains dependent on domain-specific factors and the scale of the underlying data issues [21].

2.4.2 Handling Complexities in Real-Time Data

Due to its dynamic, continuous nature, real-time data processing introduces unique challenges in predictive modelling. Unlike static datasets, real-time data streams require models to adapt quickly and accurately, making them prone to errors if not handled effectively. These challenges include latency, dynamic data streams, and robustness to noise and outliers.

Data processing or transmission latency may lead to outdated predictions concerning time-sensitive domains such as autonomous driving and financial trading.

For instance, a fraction-of-a-second delay in processing sensor data could result in a life-threatening error for autonomous vehicles, such as self-driving cars. Strategies to mitigate latency include streamlined architectures that utilise lightweight models and optimised pipelines to reduce computational overhead and edge computing to process data locally and minimise transmission delays [21,24].

The volatile nature of real-time data complicates model training and deployment. Data streams may include unexpected anomalies, missing values, or unstructured formats that challenge a model's consistency. Techniques such as on-line learning dynamically update the model to handle new data; however, this often requires significant computational resources and risks overfitting [21,24].

Real-time data is also often noisy or incomplete, leading to inaccurate predictions. For example, sudden spikes in sensor readings may be misinterpreted as valid events, causing erroneous actions in critical systems. To mitigate this, anomaly detection systems identify and isolate unusual data points, while robust training frameworks help maintain predictive stability through regularisation and ensemble methods [21,22].

2.4.3 *Model Interpretability and Bias*

As machine learning models grow more complex, their interpretability—the ability to understand and explain predictions—becomes more challenging. In high-stakes industries, such as healthcare and finance, transparency is crucial for ensuring trust and accountability.

Models such as deep neural networks function as "black boxes," producing decisions through layers of complex computations. This complexity makes it more

challenging to identify the features that influence predictions. For instance, understanding why a hiring algorithm rejects specific candidates while accepting others involves analysing which inputs affect the model's decision [24]. Techniques such as **feature attribution** help identify the most influential inputs. Simpler models such as decision trees can be used where transparency is a priority [21].

Bias, another major challenge in machine learning, can arise from imbalanced datasets or design flaws in the model. These issues can lead to unfair or discriminatory outcomes. For example, predictive models in criminal justice often show disparities against minorities due to biased training datasets [23,24]. Bias mitigation strategies are applied at three key stages: preprocessing to adjust datasets for balanced representation, in-processing to embed fairness constraints or adversarial training during model development, and post-processing to calibrate predictions for equitable outcomes across demographic groups [22,23].

2.5 Comparative Studies in Predictive Modelling for Virtual Sports

The examined studies underscore the foundational importance of predictive modelling in virtual motorsports, focusing on two key domains: esports analytics and simulated racing. These studies provide valuable insights into applying machine learning techniques, such as Random Forests, Gradient Boosting, and neural networks, emphasising their predictive accuracy, real-time feasibility, and innovations in dynamic decision-making. They also highlight challenges related to computational demands, data quality, and generalisation across broader scenarios. This comparative review lays the groundwork for identifying areas where predictive

modelling can be further enhanced, directly informing the development of models that align with the objectives of this research.

2.5.1 Existing Research in Esports and Simulation

Esports, particularly games like Dota 2 and League of Legends (LoL), have seen significant applications of machine learning in real-time performance prediction. Although these studies are not directly focused on virtual motorsports, their methodologies and insights can be transversally applied to this research.

For instance, [25] developed a win prediction model for DotA 2 using Random Forests and logistic regression, achieving an impressive 85% accuracy within five minutes of gameplay. The system relied on telemetry data extracted through Dota 2's Game State Integration, showcasing its practicality during live tournaments, such as ESL One Hamburg 2017. This model significantly enhanced live audience engagement, highlighting the potential of real-time predictive analytics. However, it faced challenges with generalizability due to its reliance on mixed datasets that included both professional and non-professional matches.

In contrast, [26] applied Gradient Boosting to League of Legends and achieved a slightly higher accuracy of 86%. While [25] focused on general win prediction, [26] explored the strategic impact of player roles. This novel approach revealed that roles had a negligible influence on overall outcomes, highlighting strategic limitations in games with defined player classes. The study also underscored the need for better integrating role-specific data, a challenge shared by both studies.

Taking a different approach, [27] introduced a Two-Stage Spatial-Temporal

Network (TSSTN) for real-time predictions in Honour of Kings, achieving a slightly lower accuracy of 76.8%. Unlike the other studies, [27] prioritised model interpretability, enabling real-time audience engagement by dynamically attributing importance to game events. However, this focus on transparency came with trade-offs, as the model struggled with multi-variable conditions, a challenge less prominent in [25] or [26].

Simulated racing studies, in contrast, leverage telemetry data to analyse performance, offering a unique testing ground for predictive modelling. For example, [28] introduced a Model Predictive Control (MPC) framework using Gaussian Processes, which improved real-time adaptability in autonomous systems and achieved a 10% reduction in lap times. Unlike esports studies, this research emphasised performance enhancements in simulation rather than numerical prediction accuracy. However, the computational demands of Gaussian Processes significantly limited scalability, particularly in large-scale simulations or competitive contexts.

Building on this, [29] compared three AI techniques—Reinforcement Learning (RL), Imitation Learning (IL), and Curriculum Learning (CL)—to train virtual racing agents. Curriculum Learning emerged as the most effective, delivering optimal lap times and reducing crash rates. It also introduced progressive task complexity, allowing agents to generalise across different race scenarios. While both [28] and [29] shared a focus on performance improvements, [29] incorporated safety as a critical factor, reflected in its crash reduction metrics. Despite these advancements, both studies faced high computational costs. In [28], these costs stemmed from the real-time adaptability requirements, whereas in [29], they

arose during model training, creating barriers for smaller-scale implementations and highlighting the need for resource-efficient algorithms.

Shifting focus, [30] explored driver performance analysis in Assetto Corsa Competizione, a popular racing simulation platform. Similar to [28] and [29], this study utilised telemetry data and machine learning, prioritising the identification of key metrics that influence driver performance, such as throttle control and acceleration, to offer actionable insights into driving patterns. Unlike the other two studies, [30] encountered scalability challenges due to the extensive data pre-processing required, limiting its applicability to broader racing contexts.

Real-time decision-making in racing and esports follows a similar approach, leveraging telemetry and historical data in predictive modelling. For instance, [16] introduced the Virtual Strategy Engineer (VSE), an AI-driven Formula 1 strategy optimisation system. This system automated pit stop timing and tyre compound decisions using neural networks trained on six years of Formula 1 data. While it performed well under dynamic conditions, such as full-course yellow flags, its narrow focus on specific strategies limited its ability to generalise to broader race dynamics, such as driver interactions or weather changes.

Similarly, [9] proposed RankNet, a hybrid architecture that decomposed motorsport rank forecasting into separate models for pit stop decisions and rank changes. This method improved prediction accuracy by 19% in short-term forecasts, making it highly effective for dynamic motorsport scenarios. However, like [16], RankNet's reliance on curated datasets restricts its generalizability to other racing series or unpredictable conditions. RankNet demonstrated a novel

approach to handling extreme events in motorsport by separating pit-stop effects from continuous dynamics.

Departing from these approaches, [7] demonstrated the scalability of ensemble learning and RNNs in handling dynamic data streams. This study provided adaptable techniques for telemetry-driven racing and esports applications, including traffic optimisation and predictive maintenance. While not directly focused on these fields, the study emphasised real-time adaptability, aligning with the demands of competitive and simulation contexts. However, unlike [9] and [16], [7] faced challenges that limited its practical application despite its scalability advantages.

2.5.2 Identified Gaps and Research Direction

The reviewed studies demonstrate significant advancements in predictive modelling for esports and racing simulations. While some are not directly focused on virtual motorsports, their methodologies and insights remain transferable to this field. Reported accuracies range from 85% to 98%, illustrating the feasibility of real-time prediction and highlighting the complementary role of real-time telemetry and historical data. Techniques such as Gaussian Processes [28] and Curriculum Learning [29] have expanded the scope of performance modelling. However, challenges persist, including limited data representation, high computational costs, and narrow model scopes, which restrict scalability and generalisability.

This research addresses several critical gaps. A key issue is the lack of integration between historical and real-time data, as most studies rely exclusively on one, such as [25], [30], [16], and [27]. These works often focus on specific

tasks, such as driver behaviour or performance prediction. This research bridges the divide by combining past performance trends with live race dynamics to enable more robust and context-aware predictions.

Another gap is the limited emphasis on ranking-specific forecasts. Studies like [26], [30], and [9] highlight the difficulty in predicting rank positions or top-three finishes, often relying on single-algorithm approaches. This research compares multiple algorithms to evaluate their effectiveness in ranking prediction.

Studies such as [28] and [7] also point to the limitations of static datasets, which fail to adapt to changing race conditions, such as weather or strategy. This study leverages live telemetry to update predictions during races, addressing this limitation dynamically.

Finally, many advanced machine learning models suffer from a lack of transparency. Studies such as [9] and [27] raise concerns over "black box" models, which obscure the decision-making process. This research prioritises interpretability to ensure predictions remain clear and actionable.

In summary, this research aims to advance predictive modelling in virtual motorsports by addressing key gaps in data integration, ranking-specific prediction, real-time adaptability, and interpretability, with applications extending to esports, simulation, and real-world racing contexts.

2.6 Applications of Predictive Modelling in Real Racing

Advancements in predictive modelling in virtual motorsports empower teams and drivers to analyse performance, refine strategies, and explore competitive oppor-

tunities. This section examines real-world applications of predictive modelling, compares virtual and real-world motorsports, and explores its potential impact on virtual racing, particularly in enhancing player experiences and strategic decision-making.

2.6.1 Case Studies in Real-World Motorsports

Predictive modelling in real-world motorsports demonstrates its effectiveness in optimising strategies. In Formula 1, telemetry data enhances fuel efficiency, tyre degradation, aerodynamic behaviour, and more [31]. Tools like the Virtual Strategy Engineer (VSE) automate pit stop and tyre decisions based on race conditions, minimising pit times to maintain race positions [16]. For example, at the Belgium GP 2023, Red Bull Racing combined telemetry and ride height compliance to adjust cornering strategies, while the Dutch GP 2023 highlighted tyre compound analysis to assess degradation and lap-time impacts [31].

Autonomous racing utilises Gaussian Processes and Model Predictive Control (MPC) to dynamically adapt strategies, improving lap times by updating models in real-time [28]. Probabilistic driver behaviour models further optimise lap times by simulating human strategy, such as vehicle setup and cornering lines [32].

2.6.2 Comparison to Virtual Racing Predictions

Virtual racing offers a safe and cost-effective platform for skill development and strategy refinement. Virtual and real-life racing use telemetry analysis, but virtual racing is easier to implement and scalable for testing strategies [30]. However, the absence of physical variables, such as weather and mechanical failures, limits

its realism. Additionally, virtual racing data often requires significant preprocessing due to incomplete datasets [30].

Technologically, real-world applications like VSE rely on intensive frameworks for real-time updates [16]. Meanwhile, virtual racing prioritizes cost-effective, feature-rich simulations using tools like MoTec telemetry to analyse car setups and driver behaviour [32].

2.6.3 Potential Impact on Player Experience and Strategy

Predictive modelling can enhance player experiences by improving decision-making through insights into optimal setups and lap strategies [30]. Machine learning models provide personalised feedback to refine skills and strategy. It also replicates the strategic depth of real motorsports, encouraging players to experiment with tactics used by professional teams, deepening engagement [32].

Additionally, predictive modelling offers scalable environments for esports players and amateurs to analyse performance without resource constraints [30]. These systems identify strengths and weaknesses, enabling targeted training and better outcomes [30].

2.7 Conclusion

Reviewing existing research highlights critical gaps in predictive modelling for virtual motorsports. One major limitation is the lack of integration between historical and real-time data; most studies focus exclusively on past trends or live telemetry. Additionally, research on ranking-specific predictions, like forecasting the top three finishers, is limited, despite its importance in motorsports analyt-

ics. High computational demands and scalability challenges restrict the applicability of advanced models, such as Gaussian Processes and Curriculum Learning, particularly in real-time. Many models have narrow scopes, focusing on specific scenarios, which limits their generalizability. Transparency and interpretability in machine learning are essential, especially where "black box" systems hinder trust and actionable insights.

This research aims to address these gaps by incorporating both historical and real-time data for more robust predictions. A comparative analysis of multiple algorithms will identify the most effective approach for ranking-specific forecasts, emphasising scalability and resource efficiency. Furthermore, the research prioritises model transparency to ensure predictions are interpretable and actionable, bridging advanced machine learning techniques with stakeholder trust.

Ultimately, this study will advance predictive modelling for virtual motorsports, with applications in real-world racing and esports. It aims to support strategic decision-making, enhance user engagement, and ensure ethical data practices by improving predictive accuracy, promoting fair competition, and introducing scalable, interpretable models.

Chapter 3: Research Methodology

3.1 Introduction to Methodology

Machine learning models are transforming virtual motorsports by allowing analysis of key racing patterns, especially in iRacing [16] [25]. These models enhance racing teams' strategic decision-making by identifying variables influencing a driver's position [16] [25]. This research predicts the top three drivers using ranking-based models, such as XGBoost, a boosted tree model, Random Forest, and RankNet, rather than binary classification techniques. The methodology outlines the research design, justifies model selection, and specifies evaluation metrics to identify the optimal model. It also details data collection, integrating historical data via API and real-time telemetry using the iRacing SDK to gain insights into driver performance. Feature engineering, preprocessing, data splitting, and hyperparameter tuning methods are discussed to optimise model performance. Ethical considerations are addressed to ensure responsible data usage, compliance with the iRacing terms, and adherence to the college ethics committee requirements.

3.2 Research Design

Designing a predictive model for virtual Motorsports demands an approach that handles the complexity of real-time telemetry while incorporating contextual depth offered by historical data [16] [17]. Existing studies often prioritise specific data types or isolated outcomes, such as real-time telemetry data, historical race per-

formance, or pit stop rankings, while overlooking the potential of integrated, rank-based forecasting. This research bridges the gap by employing a multi-model approach that combines ensemble learning and deep learning models, such as XGBoost, Random Forest, and RankNet. While predicting the top three drivers is often associated with a classification problem, this research focuses on a ranked-based approach to better capture positional nuances inherited in motorsports. Classification models usually fail to reflect the relative positioning of drivers, leading to oversimplified and inaccurate predictions [9] [27]. This framework proposes a model comparative analysis between gradient-boosted trees (GBT), decision trees (DT) and neural networks (NN) to evaluate their effectiveness in real-time prediction as demonstrated in prior work [9] [27]. It integrates live telemetry data and historical driver data while addressing model transparency. This approach ensures that forecasts are both accurate and explainable, a crucial element missing from previous research, which supports more informed strategic decisions in virtual motorsports.

3.2.1 Study type, methods and choices

The nature of this study adopts a quantitative, experimental, and comparative research architecture, aligning to systematically evaluate machine learning models for rank-based prediction in virtual motorsports in the Iracing platform [5] [4]. A quantitative approach is essential for measurable comparisons between models, focusing on model metrics to determine the most suitable model for this application [5]. These metrics include Top-3 Accuracy (Podium-Accuracy), Spearman's Rank Correlation Coefficient, learning curves, bootstrapping, GroupKFold cross-

validation, Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and feature importance analysis.

The primary metrics are the Podium-Accuracy, which determines the degree of correctly predicted driver rankings with actual rankings, the Spearman metric, which is between worst and one best in ranking order and the overall ranking metric [9] . Additionally, with the potential of bias detection, for example, misplacing low-performing drivers with high-ranking drivers [16]. These primary metrics are critical in determining model prediction accuracy, as Podium-Accuracy and Spearman determine whether the model's accuracy aligns with the predicted and actual Podium drivers. Furthermore, the Spearman metric significantly impacts Podium-Accuracy, meaning that as the Spearman value increases, Podium-Accuracy improves to a certain extent [33].

The secondary metrics are MAE, the average of absolute differences between predicted and actual values, and RMSE, equivalent to MAE, which is the square root of the average squared differences, making it more sensitive to significant errors [34]. These directly indicate how far off the model's predictions are from actual values, as RMSE represents the average distance between predictions and actual values [35]. Moreover, RMSE is similar to MAE but penalises larger errors more. While these metrics may be less informative for ranking tasks, they offer valuable insights into the overall model's raw value accuracy [34] [30].

As tools for analysing overfitting, underfitting, and assessing whether adding more data enhances model performance, the learning curve is utilised, visualising performance on training and testing data [34]. As the training set size increases,

it plots training and test errors against the number of training samples, revealing how the model learns and where it struggles. This can prove helpful when there is a significant gap between training and testing curves, which may indicate that the model is overfitting; thus, either more data or regularisation is required [30]. Conversely, if both curves exhibit high error rates, this suggests underfitting, necessitating the use of better features or more advanced models. Lastly, flat curves suggest that adding more data may not improve model performance, indicating a need to enhance the model architecture or feature engineering. Moreover, bootstrapping can test for stability and generalisation by running the model across various resampled datasets and analysing the variance in its predictions to assess model sensitivity to specific dataset samples and overall consistency [35].

Furthermore, Group K-fold cross-validation is a technique that prevents data from the same race from appearing in both the training and testing sets [34]. It ensures the model is tested on unseen data, thereby preventing data leakage and providing a more realistic evaluation than relying solely on a metric assessment from the test split [30]. Finally, for model Interpretability evaluation, Shapley Additive exPlanations (SHAP) will be utilised to analyse feature importance and potentially evaluate the extent of feature bias by measuring the model's reliance on specific inputs [36]. For instance, if SHAP reveals that high-class drivers heavily influence model predictions, this could indicate a bias that needs to be addressed, as high and low-class drivers in terms of skill cannot guarantee being a podium driver, as races are highly situational. Additionally, this allows black box models, such as RankNet, to be interpretable [30]. This is crucial, as in motorsports, any

variable, regardless of dimension, can affect a driver in the top three, including weather, driver history, and more. Ultimately, these tools will aid in model selection using primary and secondary metrics. These demonstrate whether the model performs effectively and facilitate a more straightforward interpretation of model suitability.

The experimental component allows controlled manipulation of variables, such as tuning model parameters, variables on the dataset, or more feature engineering variables to improve model performance [36]. As discussed earlier, the tools mentioned can be utilised to evaluate models and their behaviour concerning generalisation, bias, and other factors. Additionally, testing models under the same race conditions by selecting identical weather settings and driver performance classes ensures that model comparisons reflect algorithmic differences rather than external variables.

Furthermore, the comparative aspect of the study is crucial for evaluating how different machine learning architectures manage the complexities of integrating real-time telemetry and historical driver data [30]. This multi-model analysis benchmarks individual algorithm performance but explores how varied architectures influence real-time prediction accuracy and behaviour. Lastly, a critical aspect of model behaviour observations is to analyse models of their strengths and weaknesses in these field tests. Incorporating these methodologies will provide full-scale actionable insights into the strengths and limitations of each model type, providing a foundation for future development in virtual motorsport analytics and potential applications in real-life motorsports. Additionally, incorporating baseline

results from existing studies can help position this research within a broader academic context, enabling comparative analysis and providing a foundation for evaluating the effectiveness and generalizability of the proposed approach [33].

3.2.2 Limitations of Research Design

The study adopts a comprehensive and systematic methodological approach to analysing and experimenting with machine learning models for rank-based virtual motorsport prediction. However, several limitations within this research design could affect the validity and generalizability of the research outcomes. These limitations stem from data accessibility, model complexity, evaluation strategies, data noise, and the dynamic nature of virtual motorsports, all of which pose challenges to predictive Podium Accuracy and consistency.

One of the most significant limitations arises from data collection constraints imposed by the iRacing platform's API and telemetry SDK [21]. These restrictions impact the granularity, frequency, and volume of historical and real-time data available for this research, potentially leading to gaps in data coverage and influencing the model's capacity to generalise across diverse race scenarios [21]. Also, the telemetry data may present inconsistencies, such as missing values, noise, or erroneous entries, due to factors like connectivity issues, logging errors, or irregular sensor outputs [21]. Despite rigorous preprocessing strategies—including noise filtering, outlier detection, and imputation methods—such inconsistencies could misrepresent race dynamics and driver performance, leading to skewed model predictions [21]. This is caused by the iRacing API limits, such as the lack of API batch functionality and iRacing members who choose not to

make their data accessible via their account privacy settings, resulting in missing driver data that can create noise in the dataset. The iRacing SDK presents several limitations in terms of available telemetry, including the lack of real-time positional tracking, restriction to lap-to-lap results, and a lack of built-in detection for overtakes, crashes, and speed metrics, among other gaps. This can pose a significant challenge to the model in accurately predicting Podium-Accuracy, unless feature engineering is incorporated to help mitigate these issues to some extent.

Additionally, accessing iRacing tools necessitates a recurring subscription plan and secure user authentication to obtain data access. Furthermore, while real-time spectating through the SDK requires the user to own the relevant cars and tracks, the API allows access to historical race data even for content the user does not own. This distinction limits real-time data collection, but still enables broader access to historical data, which affects the potential scope and granularity of the dataset.

A significant limitation is the time-intensive data collection process, as race durations can vary between 15 minutes and 1 hour, further constraining the rate of dataset growth.

Another notable limitation is the potential for sampling bias within the dataset. Due to the nature of iRacing data collection, races featuring high-performing drivers are more frequently uploaded, logged, or made publicly visible, primarily official and ranked events. As a result, the dataset may overrepresent elite drivers, popular tracks, or specific racing conditions [21]. This could result in an unbalanced representation of different driver classes and track types, causing

the models to overfit or bias towards dominant patterns while underperforming in less-represented or more complex racing scenarios [22].

The complexity of the machine learning models used in this research also presents inherent limitations. While models like XGBoost and Random Forest offer strong predictive capabilities, and RankNet enables deep learning of ranking-specific tasks, their interpretability remains challenging, particularly for more complex architectures [30]. Despite employing SHAP to enhance interpretability and identify feature importance, neural network-based models still function primarily as black boxes, making it difficult to fully understand how specific predictions are made, especially under dynamic race conditions [36]. This limitation is critical in motorsports analytics, where transparent and explainable predictions are essential for strategic decision-making and model transparency.

Additionally, computational complexity poses a challenge, particularly when handling real-time data integration [28]. Algorithms like XGBoost, RankNet and Random Forest are robust models, but they're computationally intensive in real-time prediction and may struggle to meet real-time prediction requirements [30]. This could impact the responsiveness of live predictions and limit their applicability in real-time race strategy scenarios.

Evaluation metrics used in the study, though comprehensive, also introduce certain limitations [34]. While primary metrics like Podium-Accuracy and Spearman's Rank Correlation Coefficient effectively assess ranking accuracy and consistency, they may overlook subtle yet critical racing dynamics, such as overtaking events or close driver gaps that influence race outcomes [34]. Similarly, sec-

ondary metrics MAE and RMSE, though valuable in assessing overall prediction error, may be less informative in ranking-specific tasks. Furthermore, GroupKFold cross-validation, although designed to prevent data leakage, may still be susceptible to biases if the race conditions within grouped folds vary significantly, potentially skewing the evaluation outcomes. Detecting model bias can be challenging without comprehensive tools, potentially resulting in longer training times and less efficient model optimisations.

External factors beyond the model's control further harm predictive Podium-Accuracy. Safety cars, or severe crashes, are either simplified or not fully captured in the data, despite their significant impact on race outcomes [34]. Although feature engineering efforts aim to integrate such factors, the absence of real-time environmental data can limit the model's ability to adapt to rapidly changing race conditions that may affect Podium-Accuracy or even the overall Spearman metric. Moreover, the replicability of results remains a concern, as real-time telemetry data streams are inherently volatile, and slight variations in race conditions or data collection frequency can lead to divergent prediction outcomes across different race events. This includes reproducing the same racing conditions and environments for model comparisons, which is inherently problematic as there is no direct control besides the weather.

Ethical considerations also present limitations within the research design [23]. While anonymisation techniques are applied to safeguard driver identities and adherence to iRacing's terms of service and privacy policy, the potential for re-identification remains, particularly through unique driving styles or statistical per-

formance patterns [21]. Ensuring compliance with data privacy regulations, such as GDPR, adds complexity to data handling practices. Additionally, there is a risk that machine learning models may unintentionally reinforce existing biases within the data, such as consistently favouring high-performing drivers or specific race strategies. This occurs because races involving skilled drivers or standard strategies are more frequently spectated, uploaded, or included in official sessions, resulting in their overrepresentation in the dataset. While SHAP analysis is employed to detect such biases, mitigating them fully remains a complex challenge, particularly when biases are deeply embedded within the data itself.

Several mitigation strategies are incorporated into the research design to address these limitations. Data augmentation techniques enhance dataset diversity and reduce sampling bias, while robust cross-validation methods, such as Group-KFold, minimise the risk of data leakage and ensure model evaluations reflect real-world scenarios. Interpretability tools, such as SHAP, are leveraged to enhance transparency and identify feature dependencies. Ethical safeguards, including anonymisation protocols and fairness evaluations, are strictly enforced to ensure responsible data use. Despite these measures, the limitations outlined highlight the complexities and challenges inherent in predictive modelling within virtual motorsports, underscoring the need for continuous refinement and adaptation in future research.

3.3 Data Collection, Pre-processing, feature engineering and retrieval

Data collection, preprocessing, and feature engineering form the foundation of predictive modelling in virtual motorsports. However, data quality is more critical than volume [21]. It directly impacts the ability to learn effectively and achieve reliable performance. Poor-quality data can introduce biases, noise, and inaccuracies, harming model metrics [21]. The iRacing API and, more importantly, the SDK's raw telemetry data must be extensively pre-processed to remove noise, missing values, and irrelevant data [21]. Furthermore, this includes feature engineering to transform raw inputs into meaningful variables. The following sections provide a detailed breakdown of these processes, outlining the methodologies used to refine the dataset and optimise model metrics as much as possible.

Data on races, drivers, past races, laps, events, and race metrics is stored in a Database (DB) instead of a flat file, primarily due to its capability for structured data storage, faster retrieval, enhanced security, data integrity, scalability, and support for complex queries [37]. Moreover, it facilitates advanced feature engineering and dynamic data management through triggers, as illustrated in Appendix B.

Data collection is achieved by integrating historical data with live data, as depicted in the Appendix A flowchart [16] [37] and performed by spectating live online races. This is executed by dividing the data acquisition process into two Python scripts. The historical data acquisition begins by connecting to the iRacing live collection script to fetch a list of drivers, followed by looping through each driver to gather their statistics and past race history, while eliminating un-

necessary data and cleaning it using the iRacing API with the 'iracingdataapi' package. However, only the last 10 race results can be retrieved, rather than the entire race history from the date the driver's account was created. Once this is complete, feature engineering is conducted, resulting in the creation of two flat files: one containing driver statistical history and another containing past race history dataframes. These dataframes are then inserted into the respective database tables, as shown in Appendix B [5]. To prevent bias toward drivers who appear in multiple races, the system is designed to exclude repeated drivers already stored in the dataset. This precaution helps maintain fairness, reduce overfitting, and ensure high-quality predictions. While this halts data collection for races involving previously recorded drivers, it is a deliberate trade-off to preserve the integrity of the dataset. In addition to drivers who do not consent in their respective privacy settings, data collection is also stopped.

After completing this procedure, it returns a race ID, which is crucial for logging the current lap and event data for the respective driver and race, as highlighted in Appendix B. This limitation arises from the iRacing package's lack of support for batch API requests and restrictions on API call requests [21]. Data retrieval from the iRacing API is subject to timing constraints, particularly during race qualification sessions, as discussed in the research Design section. The API's lack of batch processing necessitates sequential data requests, which impacts real-time scalability unless the collection is done earlier in the qualifying process, depending on the number of drivers on the grid. Although this limitation does not directly affect telemetry-based live predictions, it constrains the dataset's

expansion rate over multiple races if that portion of the collection is conducted later than optimal, highlighting the critical importance of precise timing [37] Appendix D. These API calls utilise Async/await to speed up the historical data collection process and prevent potential errors.

For the collection of live telemetry data, the 'pyirsdk' package is employed to connect and extract data from the iRacing application [36]. The data comprises YAML-formatted information, which includes details about other drivers and extensive telemetry data about the player's vehicle. The focus of interest lies in data related to other drivers, rather than the player's telemetry data. However, this data has limitations, including inconsistent update frequency and restricted availability. To address these limitations, it is essential to identify what data can be retrieved and how frequently it is updated.

The Real-Time Events table is designed to capture key race events dynamically. It updates at a frequency of 60 times per second (approximately every 16.67 milliseconds), recording telemetry snapshots when certain conditions are met, for example pits, incidents, offtrack, distance covered % and positional changes [28]. The real-time live data pipeline is visualised in Appendix A Flowchart.

Regarding the lap data table, the data inserts a specific driver who completes a lap of the track. DB triggers capture these updates in real-time for feature engineering [16]. Unlike Python post-processing, which requires explicit function calls after data collection, triggers automatically update key performance metrics as race events unfold [5]. For instance, when an incident is recorded in the RealTimeEvents table, triggers immediately increment the bucketed event counters;

these feed directly into the decay-weighted driver feature set. This approach ensures that real-time analytics remain up-to-date without requiring additional computation overhead. This will signal the model to rank the driver lower, particularly if further incidents occur, thereby justifying triggers as a more feasible approach than coding in Python, shown in Appendix E.

The iRacing SDK does not directly provide several critical telemetry features, particularly real-time positional tracking and overtaking detection. To compensate for these missing features, for example, speed can be calculated by dividing the distance travelled by the time taken, and incident detection can be implemented by identifying low-speed levels and driver RPMs that indicate a potential incident. The driver's live position can be calculated using the lap distance covered and the estimated lap time. Overtakes and positions lost can be calculated by summing the position from the previous lap and the current lap. For more information, look into the Appendix E.

Furthermore, while other features, such as tire wear, fuel level, brake level, and more, remain absent or only present as **static snapshots** within lap data, such as the number of overtakes, positions lost, and final position, this limitation adversely impacts the depth of real-time analysis for the model. These limitations arise from restrictions by the iRacing platform to protect privacy and maintain fair play, preventing potential cheating or unfair advantages. Null values in the telemetry stream are converted to zeros before insertion into the database, following the approach used by [37]. To support real-time performance, asynchronous methods (`async/await`) are used during API data collection and database operations

in the live pipeline. This ensures non-blocking execution and improves scalability during high-frequency updates of telemetry data.

The data stored in the database is not directly utilised as a dataset for the model because it is incompatible with the format. Instead, it is first retrieved and transformed into a structured format, specifically a flat file, to ensure seamless integration with the model training pipeline Appendix C. While using a relational database may seem impractical at this stage due to its structure, it provides several advantages. For instance, it allows for additional feature engineering and preprocessing, as outlined in Appendix E.

However, this transformation process requires careful handling to ensure data consistency and prevent leakage of both group and driver information. These key considerations include avoiding duplicated driver entries, guaranteeing each driver within the same race is not duplicated, calculating features within the same driver, and normalisation performed within the same race ID group, structured by selecting five decay-weighted buckets centred on the 1st (0%), 25%, 50%, 75% a system inspired by [9] research. While not incorporating 100% buckets is to prevent temporal leakage issues. Each bucket summarises a 3-lap rolling window, so the last bucket effectively covers the penultimate and final laps, keeping column counts consistent. Because the bucket routine introspects the schema at run-time, any new numeric telemetry or historical column is automatically pulled through, eliminating the need for manual column lists [9]. This approach ensures that each race's data has consistent columns, avoiding excessive zeros that would otherwise appear due to varying total lap and event counts across different

ances [9]. Additionally, to mitigate dataset noise that can impact model performance and training. Overall, as listed in Appendix E, the flowchart and feature engineering list showcase how this process works.

Overall, the feature engineering process, including the derivation of metrics such as `LiveScoreAdjusted` and `PositionVolatility`, follows established best practices for handling high-frequency telemetry data [5] [21]. This ensures robust and reliable inputs for the predictive models. More details are in Appendix E.

3.4 Model choice justification, Training and Validation

3.4.1 Model Choice Justification

As described in the previous sections and chapters, the primary machine learning algorithms selected for this comparison experiment are XGBoost, Random Forest, and RankNet. These models represent three key machine learning categories: GBT, DT, and NNN. These models are selected based on existing evidence from previous research that supports their effectiveness in motorsports, e-sports, and simulation-based ranking tasks.

Multiple studies, such as [9] and [25], have successfully applied DT and GBT models in e-sports contexts to process large volumes of time-series and telemetry data. While these applications are not directly related to virtual motorsports, their methodologies are transferable due to their reliance on historical and real-time data [9] and [25].

XGBoost and Random Forest excel in processing structured tabular data, offering robust handling of missing values and noisy inputs, a crucial requirement

in iRacing telemetry data, where dropped frames and inconsistent sensor logging can introduce gaps. Their ability to adapt to dynamic race conditions makes them well-suited for real-time motorsport prediction.

Although RankNet has proven effective for short-term rank forecasting in motorsport due to its Pairwise Ranking nature [26], it suffers from black-box interpretability issues [9,27]. To mitigate this, SHAP values can be employed to analyse feature contributions, providing insights into how RankNet ranks drivers in different race conditions.

Although XGBoost and Random Forest have been rarely applied to motorsport ranking problems, their demonstrated robustness in noisy, high-frequency esports analytics, which parallels motorsport telemetry despite not being explicitly ranking-based, combined with the strengths of RankNet, makes them strong candidates for comparative analysis in this study.

3.4.2 Training and validation

The dataset used for training is generated via the database-to-flat file conversion process (see Appendix E) and is applied uniformly across all models to ensure fair comparisons. Before training, the data are separated into feature matrices and the target variable (RankScoreAdjusted), with each row representing a driver's performance snapshot that integrates historical data and live telemetry. Missing values are addressed during preprocessing, and any remaining gaps are imputed using a SimpleImputer; all features are subsequently scaled via min-max normalisation. The dataset is then partitioned into 70 % training and 30 % testing sets at the race level: GroupShuffleSplit is employed for RankNet and XGBoost,

while a manual group-based shuffle is applied for Random Forest to prevent test or train metrics from being mixed.

Due to this race-based splitting, Group K-Fold cross-validation is employed (instead of traditional cross-validation) to ensure that entire races are confined exclusively to either the training or validation sets, thereby providing a realistic evaluation of unseen races. Primary evaluation metrics include Podium Accuracy and Spearman's Rank Correlation, while MAE and RMSE serve as secondary metrics to assess numerical prediction accuracy; learning curves are examined to detect underfitting or overfitting. To further evaluate model stability, which is highly sensitive to particular race subsets, bootstrapping is conducted over 30 race-level samples. In addition, SHAP feature-importance analysis is performed to identify potential biases by quantifying each feature's contribution to the predicted rankings. A more detailed exposition of these validation strategies is provided in the study type, methods and choices section.

Finally, the target variable is RankScoreAdjusted; features that indicate temporal leakage (including LiveScoreAdjusted, Position, LivePosition, F2Time, DriverGapInFront, and PodiumProximity) as well as the normalised target itself are excluded before training (see Appendix E and the feature list).

Model Training and Validation Configurations

XGBoost (Ranker)

- **Implementation:** `XGBRanker` with GPU (`tree_method="hist", device="gpu:0"`)

- **Core Parameters:** `n_estimators=2000, learning_rate=0.01,`
`max_depth=1, min_child_weight=10, subsample=0.7,`
`colsample_bytree=0.6, colsample_bylevel=0.6, gamma=4,`
`reg_alpha=4, reg_lambda=4, objective="rank:pairwise",`
`eval_metric="rmse", random_state=42`
- **Evaluation Metrics:** RMSE, MAE, Spearman correlation, per-race Spearman via GroupKFold, and Top-3 accuracy.
- **Data Processing:** Drop identifier/leakage columns; target is `RankScoreAdjusted_norm`; scale with `MinMaxScaler`; impute with `SimpleImputer(mean)`; split via `GroupShuffleSplit (70/30 by RaceID)`
- **Training:**
 - Fit model on $(X_{\text{train}}, y_{\text{train}})$ with `group=group_sizes`.
 - Save model and preprocessing tools using `joblib`.
 - Predict on X_{test} ; compute RMSE, MAE, Spearman.
- **Cross-Validation:**
 - GroupKFold (10 folds): re-fit scaler/imputer, train model, predict on X_{test} .
 - Compute per-race Spearman ρ and Top-3 accuracy, average across races.
 - Final metrics: $\text{final_spearman} = \frac{1}{10} \sum_i \rho_i$, $\text{final_top3} = \frac{1}{10} \sum_i \text{Top3}_i$.

- **Bootstrapping (30x):**
 - Sample races with replacement from training set.
 - Train and evaluate on fixed test set.
 - Aggregate mean/std of Spearman: mean_rmse, std_rmse.
- **Learning Curve:**
 - For fractions $f \in \{0.1, 0.2, \dots, 1.0\}$:
 - Train on f -fraction of races; compute train/test error as $1 - \rho$.
- **Feature Importance:** Analyze SHAP values.
- **Persistence:** Save model, scaler, imputer.

Random Forest

- **Implementation:** RandomForestRegressor with n_jobs=-1
- **Core Parameters:** n_estimators=2000, max_depth=6,
min_samples_split=22, min_samples_leaf=5, max_features=0.5,
random_state=42
- **Preprocessing:** Drop identifier/leakage columns; scale via
MinMaxScaler; impute via SimpleImputer(mean)
- **Data Split:** 70/30 by unique RaceID; retain RaceID for group evaluation.
- **Evaluation Metrics:** RMSE, MAE, Spearman, group-based Spearman, Top-3 accuracy.

- **Cross-Validation:**

- GroupKFold (10 folds): reprocess each fold, retrain, evaluate Spearman and Top-3.

- **Bootstrapping:**

- Sample races with replacement; retrain and evaluate on fixed test set.
- Compute mean/std of Spearman.

- **Learning Curve:**

- For each fraction f : train on subset of f training races; compute $1 - \rho$ for train/test.

- **Feature Importance:** SHAP via TreeExplainer.

- **Persistence:** Save model, scaler, imputer.

RankNet

- **Architecture:** PyTorch feedforward net: FC layers $D \rightarrow 512 \rightarrow 256 \rightarrow 128 \rightarrow 64 \rightarrow 1$, with BatchNorm1d, ReLU, and dropout (0.1) after each layer except final.

- **Loss Function:**

$$\text{diff_true} = y_{\text{true}} - y_{\text{true}}^T,$$

$$\text{sign} = \text{sign}(\text{diff_true}),$$

$$\text{mask} = (\text{sign} \neq 0),$$

$$\text{diff_pred} = y_{\text{pred}} - y_{\text{pred}}^T,$$

$$\text{bottom} = (y_{\text{true}} < 0.2) \vee (y_{\text{true}}^T < 0.2),$$

$$w_{ij} = \begin{cases} 2, & \text{if } \text{bottom}_{ij} \\ 1, & \text{otherwise} \end{cases},$$

$$L = \frac{1}{|\{(i, j) : \text{mask}_{ij}\}|} \sum w_{ij} \log(1 + e^{-\text{sign}_{ij} \cdot \text{diff_pred}_{ij}})$$

- **Optimization:** AdamW(lr=1e-4, weight_decay=1e-5); scheduler:

ReduceLROnPlateau, patience=500

- **Training:** 8000 epochs, batch size 512; PyTorch DataLoader;

backprop on GPU (if available)

- **Evaluation Metrics:** RMSE, MAE, Spearman, Top-3 via GroupKFold.

- **Preprocessing:** Drop identifier/leakage cols; MinMaxScaler,

then SimpleImputer; split via GroupShuffleSplit

- **Cross-Validation:**

– GroupKFold (10 folds): train RankNet; evaluate Spearman, Top-3.

– Final metrics: average over folds.

- **Bootstrapping (30x):**

– Sample RaceIDs with replacement; train and evaluate on global test set.

– Compute mean/std of Spearman.

- **Learning Curve:**

- For fractions f : train on subset of $f \cdot$ races.
- Compute train/test error as $1 - \rho$.
- **Implementation:** PyTorch; GPU-enabled; SHAP via `GradientExplainer`; explainability on test set
- **Persistence:** Save model, scaler.

3.5 Ethical approaches

This study adheres to strict ethical standards and guarantees complete adherence with iRacing’s Terms of Use and Privacy Policy [38] [39] [40]. Data collection is conducted exclusively through the iRacing API and SDK, respecting the platform’s policies on data access and user privacy [38] [39] [40] [41]. Since all iRacing participants agree to these terms upon registration, no additional consent is required for data retrieval [38]. Users also have control over their data visibility through privacy settings, allowing them to opt out of API-based data access at any moment, [40] [41]. This ensures that only publicly accessible data is utilised, aligning the study with ethical research practices.

All collected data is anonymised to safeguard participant privacy, ensuring driver IDs cannot be directly linked to individual players [24] [40] [41]. Identifiable details are either pseudonymized or removed entirely, mitigating any risk of personal identification. Data storage and handling are conducted with strict security protocols: only stripped, feature-level data with driver IDs is stored in a secure MS SQL database (no raw telemetry is ever persisted), and when data collection is completed, those IDs shall be anonymised [24] [40] [41]. The re-

search will not allow external access to raw, unanonymized data, ensuring that only the researcher and the supervisor have direct access to sensitive information [24] [40] [41].

As iRacing is a proprietary platform, this study does not commit unauthorised data extraction beyond what is explicitly permitted [39] [40] [41]. The research is conducted solely for academic purposes and does not interfere with iRacing’s proprietary algorithms, anti-cheat systems, or commercial interests [40] [41]. There is no intent to develop commercial software or tools that could compromise the platform’s integrity. Also, the study does not engage in player profiling or ranking manipulation that could impact individual performance metrics, mastery levels, or in-game rewards [39] [40] [41]. Participants will not be subjected to any analysis that alters their experience within the platform [41].

Data scraping and retrieval methods adhere to the ethical boundaries outlined in iRacing’s Terms of Use [38] [40] throughout the study. The methodology ensures responsible data usage while maintaining transparency and fairness in research practices. Integrating these ethical safeguards upholds the integrity of the study while ensuring compliance with all necessary regulations [24] [40] [41]. Additionally, all data collected and pre-trained models are owned by the iRacing platform and cannot be released publicly without written permission [40] [41].

3.6 Conclusions

In summary, previous research on virtual motorsport analytics underscores both the promise and the challenges of integrating historical and real-time data for pre-

dictive modeling [42]. While multiple studies have successfully utilised machine learning for outcome prediction or performance analytics, most approaches rely primarily on historical data or classification-based methods [42]. Additionally, the absence of cohesive ranking-based solutions in fast-evolving race conditions highlights a gap that demands methods capable of real-time adaptability and deeper positional insights [33].

These findings from the literature have significantly shaped the methodological approach adopted in this study. Leveraging insights into the utility of gradient boosting, ensemble trees, and neural networks for complex rankings, this research design integrates iRacing's live telemetry with driver history, employing rank-oriented metrics such as Spearman's correlation and Podium Accuracy [33]. This chapter details how these insights translate into a multi-phased data collection pipeline, tailored feature engineering, and the comparative training and validation of XGBoost, Random Forest, and RankNet models [42].

Chapter 4: Analysis of Results and Discussion

4.1 Introduction to Analysis

This chapter presents and discusses the results of the comparative analysis of machine learning algorithms—RankNet, XGBoost, and Random Forest—utilised for predicting podium drivers in virtual motorsports on the iRacing platform. The primary objective is to determine the most suitable model for predicting podium drivers under live race conditions. The suitability of the models is evaluated based on prediction accuracy, consistency across races, the effectiveness of real-time predictions, performance metrics analysis, and the influence of individual features on prediction outcomes.

The analysis utilises real-time racing data on driver performance, as well as historical data obtained through the iRacing API and SDK. Performance is assessed via a ranking-based evaluation, focusing on each model's ability to accurately predict the relative finishing positions of drivers, with an emphasis on podium placements rather than a traditional classification-based approach.

The results are compared with research studies outlined in the literature review to situate these findings within a broader academic context. This comparative perspective illuminates areas of alignment, contradiction, or advancement within existing research on predictive modelling in motorsport analytics.

The remainder of this chapter comprehensively evaluates the performance of each model, incorporating model metrics and visual representations of feature im-

portance. This is followed by findings and an interpretation that discusses these results in relation to previous research, examining how the findings align with the existing literature. Furthermore, the findings are analysed to reflect on the broader implications of this study. Finally, the study's limitations and recommendations for future research are articulated. The chapter concludes with a summary of the study's key findings and conclusions.

4.2 Findings and Interpretation

This section critically discusses the main findings and illustrates how they relate to the dissertation's original objectives. Each significant finding is presented, interpreted, and analysed, highlighting whether the goals were entirely, partially, or not achieved, with supporting reasoning from the literature. These findings are substantiated by empirical evidence; for visual corroboration, the reader is referred to the live-prediction and model training demonstrations in Appendix Appendix F

4.2.1 Results

Comparison of Model Metrics

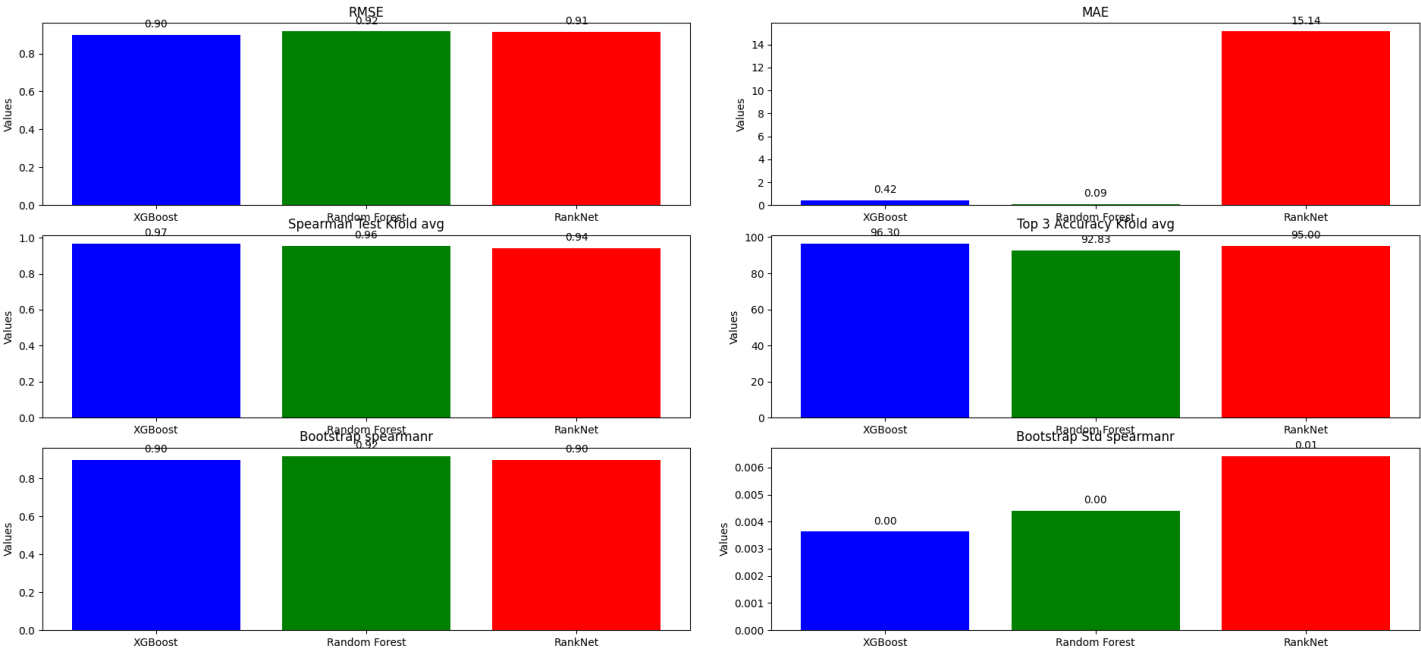


Figure 4.1: Comparison of model metrics across XGBoost, Random Forest, and RankNet.

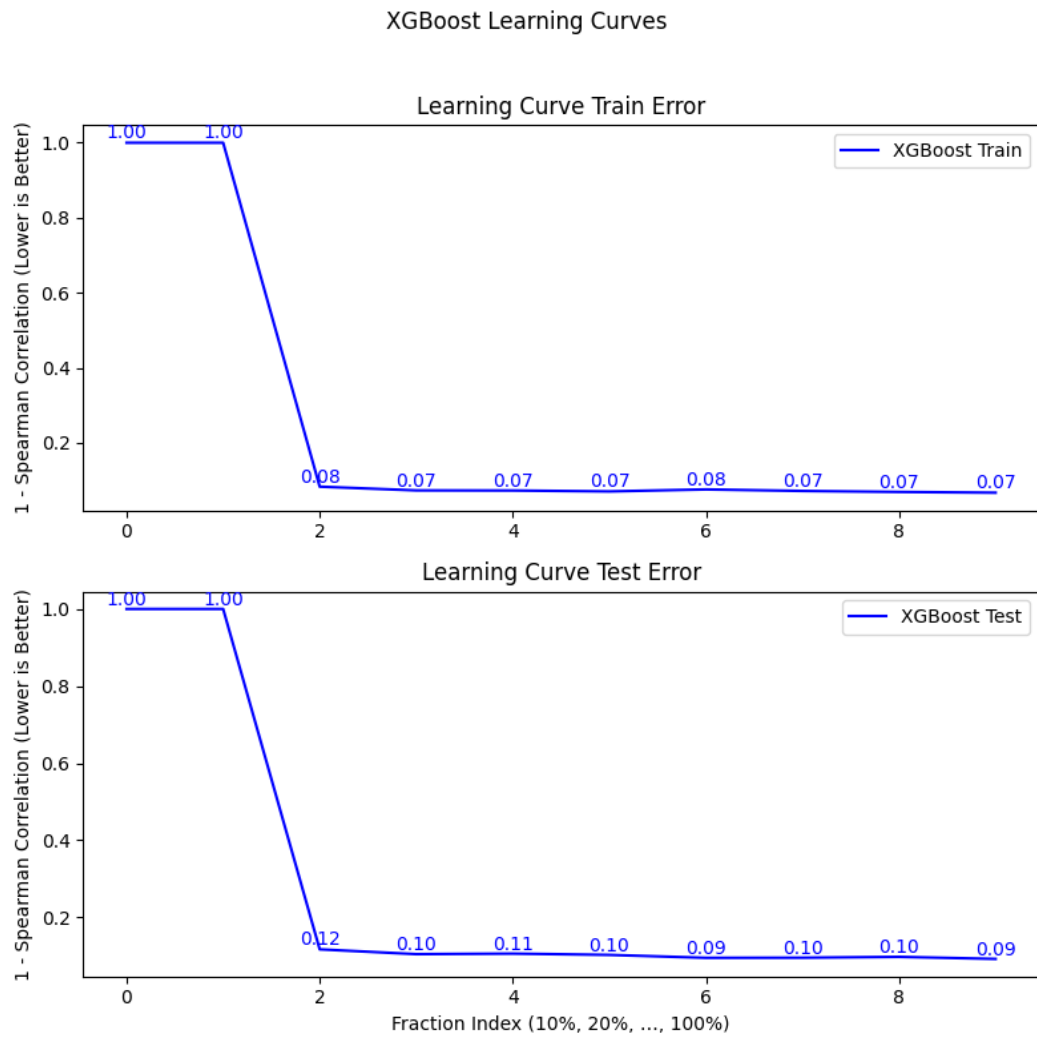


Figure 4.2: XGBoost Model learning rate

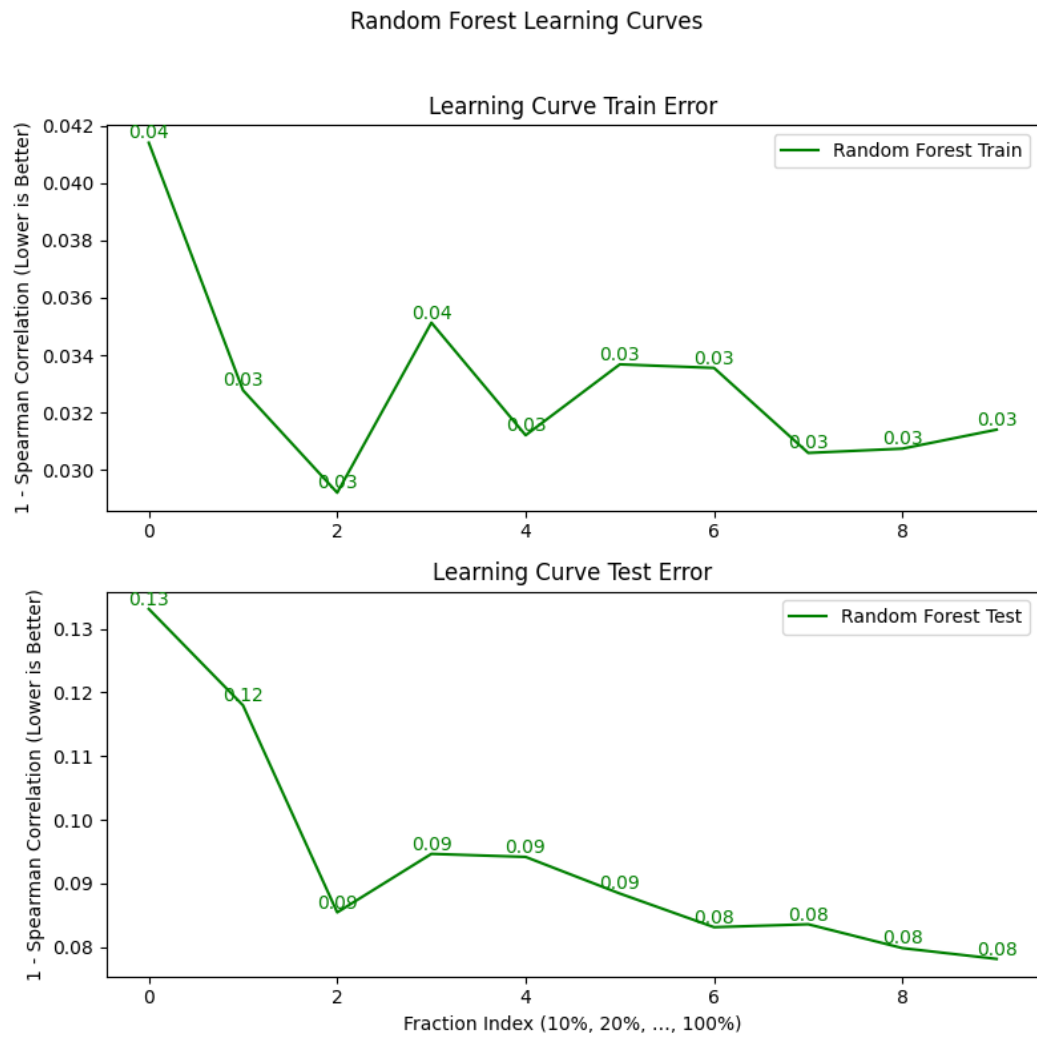


Figure 4.3: Random forest Model learning rate

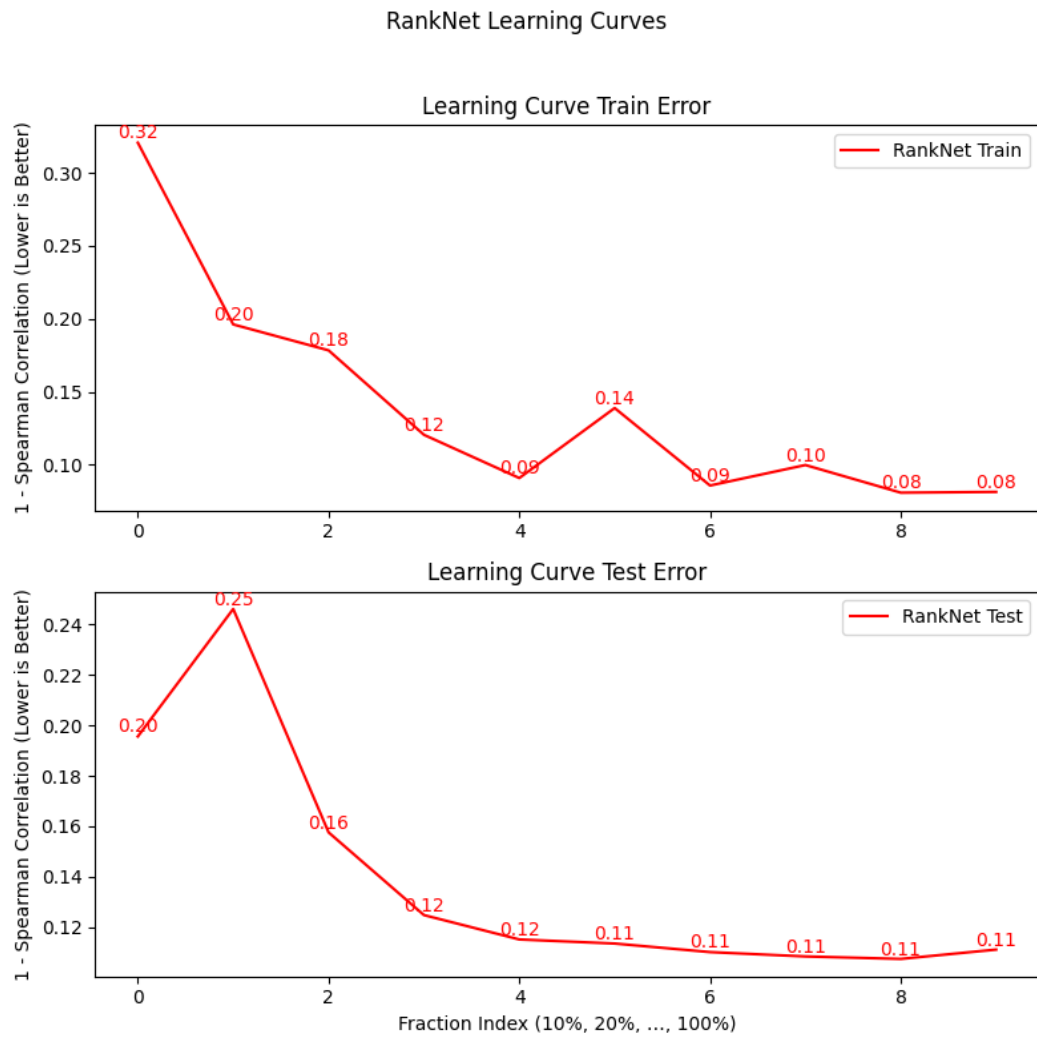


Figure 4.4: RankNet Model learning rate

Differences Between Models

Metric	XGBoost - Random Forest	XGBoost - RankNet	Random Forest - RankNet
RMSE	-0.01650000000000007	-0.014900000000000024	0.0016000000000000458
MAE	0.32667624923681104	-14.72326292405735	-15.04993917329416
Spearman	0.010292638558883183	0.023912019296841613	0.01361938073795843
Top 3 Accuracy	3.462962962962962	1.2962962962963047	-2.166666666666657

Figure 4.5: Model differences

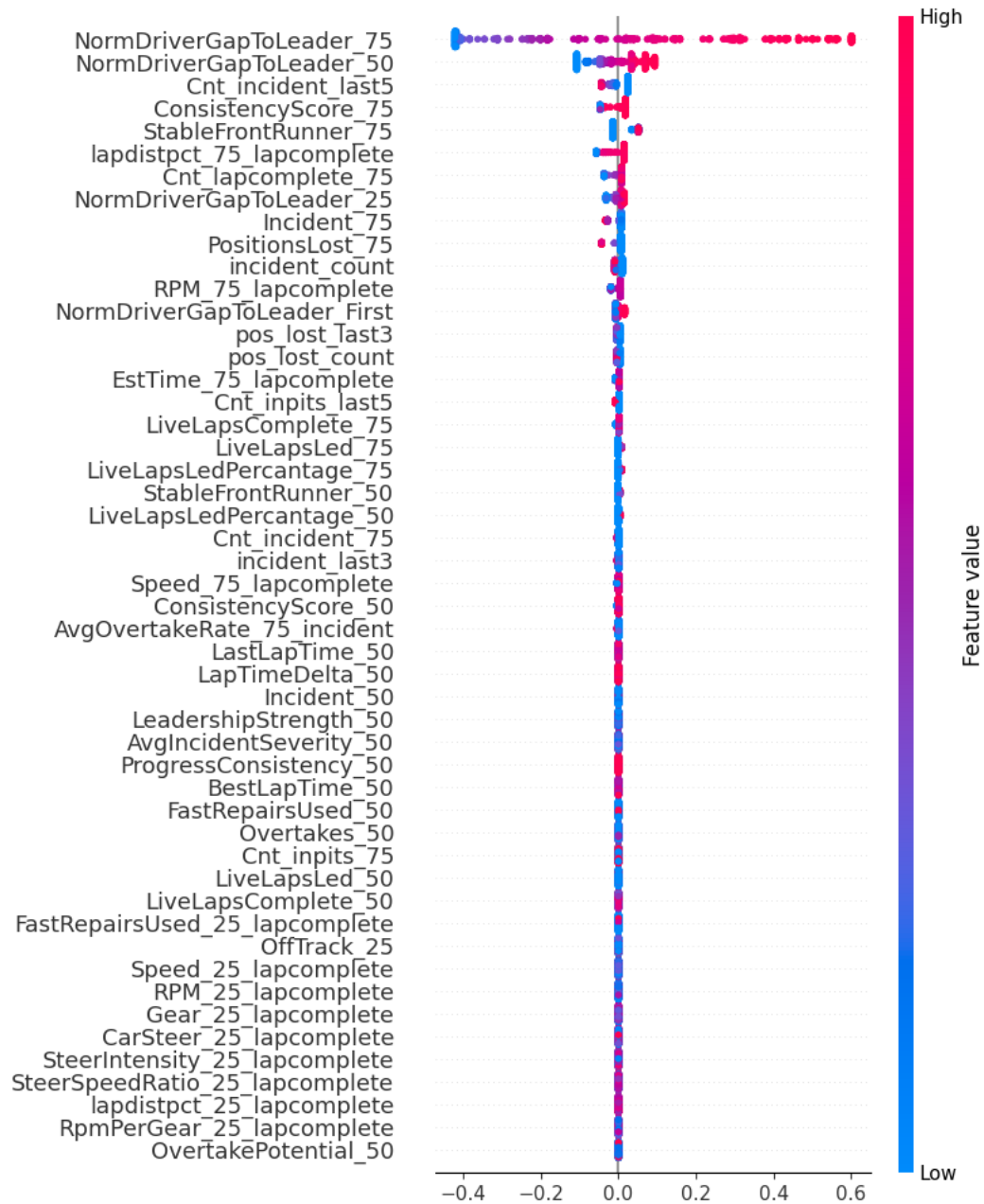


Figure 4.6: XGBoost feature importance



Figure 4.7: Random Forest feature importance

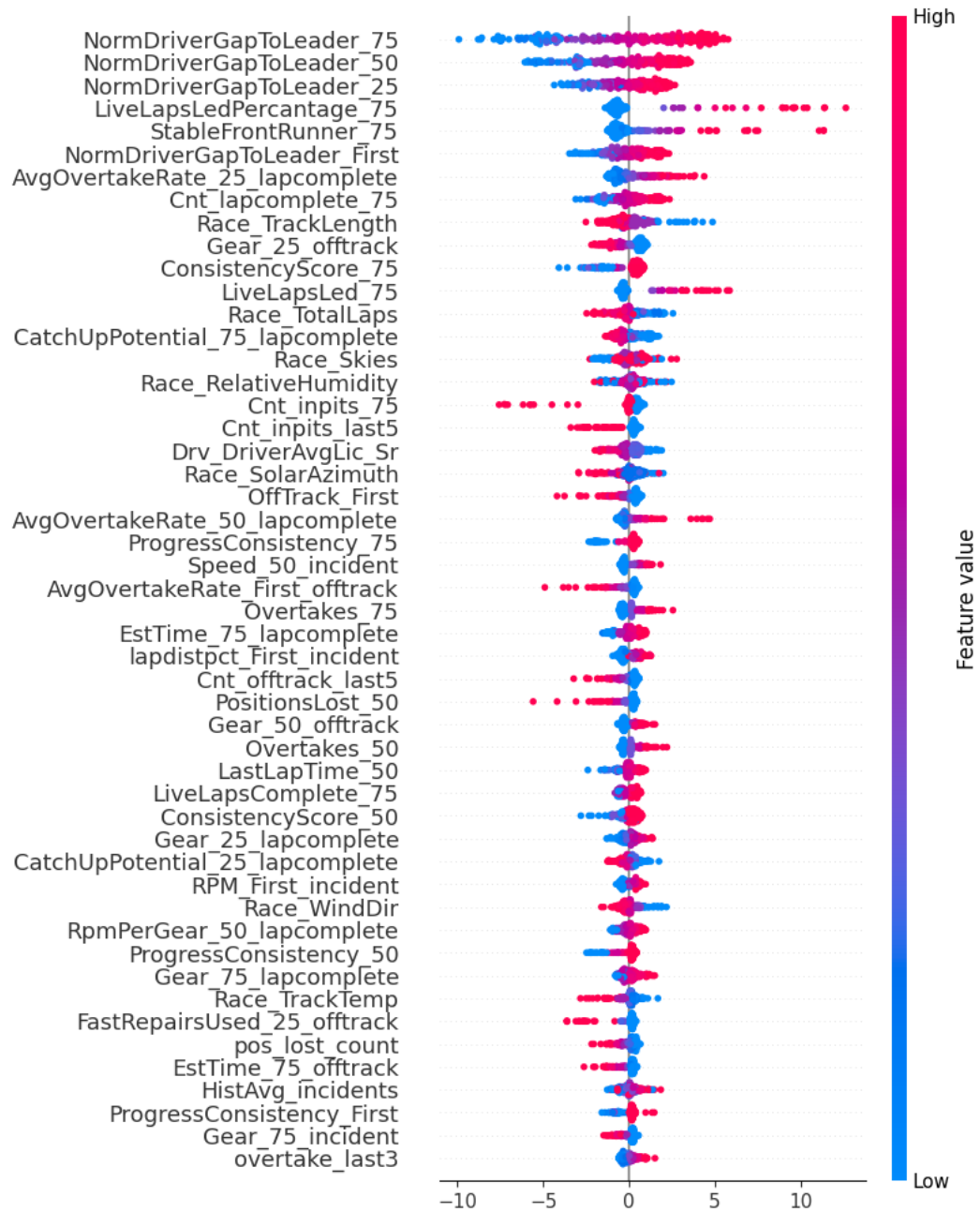


Figure 4.8: RankNet feature importance



Figure 4.9: XGBoost feature importance (bar graph)



Figure 4.10: Random Forest feature importance (bar graph)

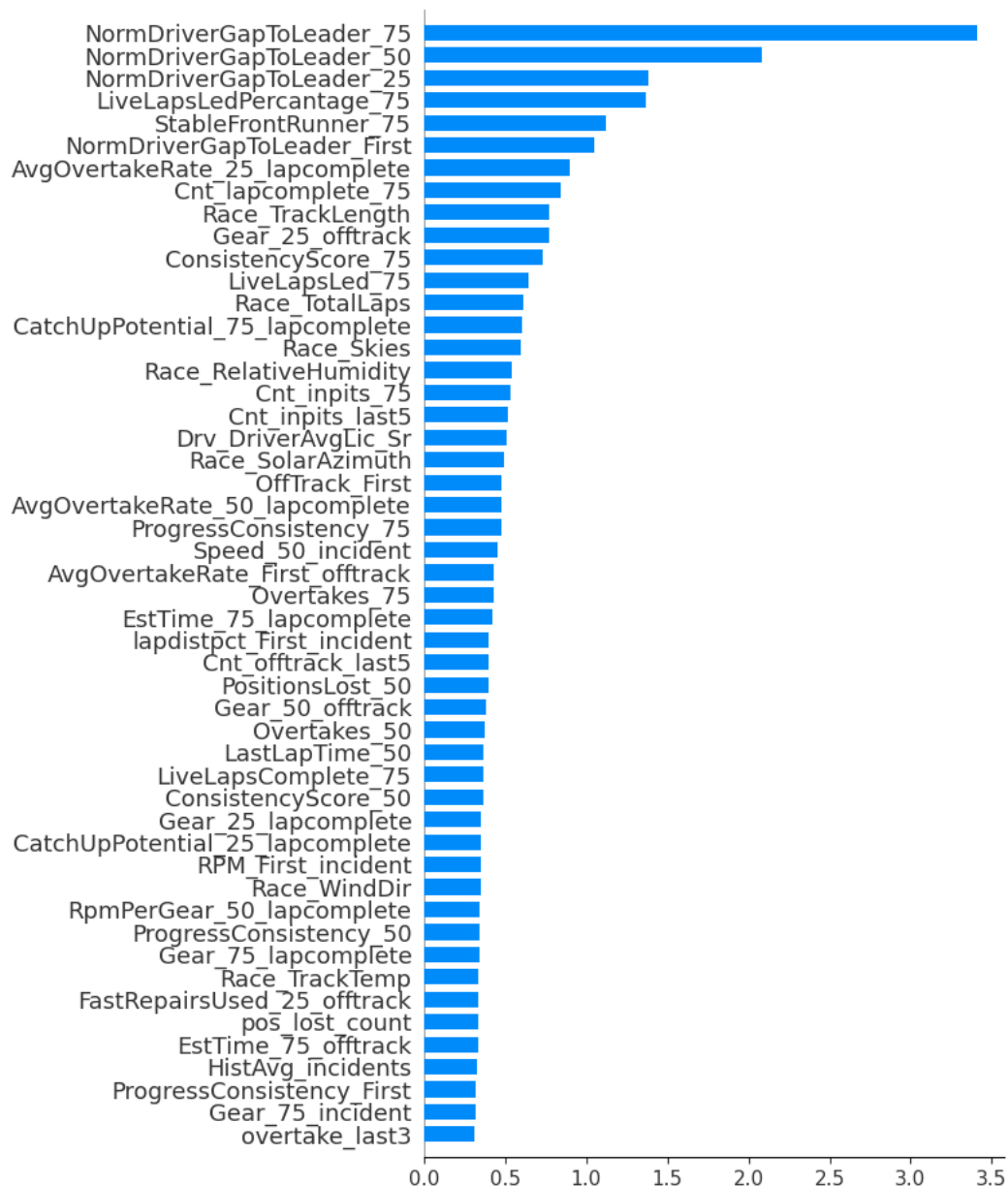


Figure 4.11: RankNet feature importance (bar graph)

4.2.2 Overall best practical predictor due to its mid-lap capability and stronger generalisation, despite XGBoost's slightly higher top3 accuracy and Spearman

The initial finding emphasises that RankNet offers the most practically valuable predictive capability compared to traditional tree-based algorithms — specifically XGBoost and Random Forest — when forecasting podium drivers in virtual motorsport racing. In comparison, RankNet's Top-3 accuracy (95%) is slightly lower than XGBoost's (96.3%) and higher than Random Forest's (92.83%). This is due to its ability to deliver reliable mid-lap updates of mean Top-3 accuracy of approximately 75% in live prediction testing at 25%, 50%, and 75% decayed buckets of laps and events snapshots during live testing. Thus, making it the ideal choice for real-world applications that require a dynamic predictive model. Additionally, RankNet's MAE and RMSE are not noteworthy, because RankNet is optimised on absolute ordering rather than predicting exact values [43].

Furthermore, RankNet demonstrates enhanced generalisation, evidenced by the minimal train-test gap in $1 - \rho$ (approximately 0.03), confirming that the model learns robustly without overfitting, an essential trait since live races never exactly match the training data. RankNet also showcases the most diverse feature-importance profile due to its capacity to capture non-linear relationships among telemetry variables. These varied features facilitate improved adaptability to new or unfamiliar racing conditions, in contrast to XGBoost and Random Forest, which rely on a more limited subset of predictors.

Additionally, RankNet exhibits a very low standard deviation in bootstrap Spearman scores (approximately 0.01), ranking third only to Random Forest and XG-

Boost by a negligible margin. Additionally, the bootstrap Spearman values are almost identical. This stability across resamples indicates that the model is unlikely to produce erratic predictions if the data distribution shifts. Similarly, XGBoost is similar, but Random Forest displays a slightly lower error rate and standard deviation on unseen snapshots, but on a narrow margin ($\tilde{0.05\%}$), which reinforces RankNet's overall robustness.

These findings are supported by prior research, such as [34], which demonstrated that RankNet models rank problems better than tree-based models. In motorsports, virtual and non-virtual ranking is a dynamic, order-sensitive task, making neural rankers like RankNet a more natural fit. Moreover, [26], [27] [44] argued that while traditional tree-based models perform respectably well, they support the use of deep learning models like RankNet due to their superior ability to capture non-linear and multi-variable relationships.

The results strongly indicate that integrating real-time telematic data and historical driver statistics into rank-aware models yields more accurate predictions for podium rankings than traditional regression and classification methods. Thus, the first objective of evaluating and comparing predictive model accuracy is fully achieved by demonstrating RankNet's superior overall performance.

4.2.3 Enhanced Real-Time Adaptability of RankNet

The second significant finding underscores RankNet's enhanced real-time adaptability in predicting podium drivers under live race conditions. Compared to XGBoost and Random Forest, RankNet exhibited a superior capability to adjust its predictions based on real-time incoming data throughout the laps. As events are

logged into the database whenever a racing incident occurs or a driver gains or loses a position, RankNet conducts analyses and ranks the drivers accordingly, which becomes evident by the conclusion of the lap. While XGBoost and Random Forest concentrated on lap-to-lap results, RankNet focused on live results. Although RankNet attained highly accurate predictions at lap-to-lap intervals, its predictive accuracy under live lap conditions was approximately 75% during live testing, providing a realistic and optimistic estimation in light of the volatile nature of real-time racing events.

This observation is further corroborated by the model's performance across live data streams. Unlike tree-based models, RankNet consistently generated more dynamic updates for podium predictions as laps progressed, which rely on lap-to-lap results. Although very accurate, it is noteworthy that RankNet dynamically adjusts during the lap's initial, mid-phase, and concluding phases.

This enhanced adaptability can be attributed to RankNet's neural network architecture, which allows it to generalise from complex and evolving race dynamics. For instance, in motorsport racing, every variable, regardless of its magnitude, can significantly influence podium positions; therefore, neural networks can assign appropriate weights to determine whether to rank a driver within the podium. In contrast, tree-based models depend on a fixed number of features, as evidenced by their feature importance metrics. This flexibility enables RankNet to accommodate unforeseen developments in the race, such as unexpected overtakes or strategic pit stops.

These findings align with previous research by [26], [27], which emphasised

the superior real-time performance of deep learning models in dynamic esports environments. Given the similarly dynamic nature of motorsport racing, these studies support the conclusion that RankNet is better suited for real-time podium prediction tasks [26], [27].

Therefore, the enhanced real-time adaptability during laps observed further supports RankNet’s suitability as the optimal predictive model for live podium forecasting in virtual motorsports, fulfilling the objective of identifying models capable of dynamic race prediction under real-world conditions.

4.2.4 XGBoost stability and reliability across races

The third significant finding highlights the stability and reliability of XGBoost across different race scenarios. While XGBoost achieved the highest Top-3 prediction accuracy compared to RankNet and Random forest, it exhibited consistent performance across multiple races due to its ability to accurately predict lap-to-lap outcomes, suggesting strong generalisation and low sensitivity to race-specific variations. Notably, XGBoost’s behaviour focused on lap-ending results rather than continuous in-lap events, with the model ranking drivers based on completed laps rather than dynamically updating during ongoing lap activity.

This observation was supported by the relatively low variance observed in XGBoost’s prediction metrics across races. The model maintained a stable Top-3 accuracy and exhibited minimal fluctuations in Spearman correlation scores, even as race contexts varied in terms of track type, driver count, weather, and duration. This supports the claim that XGBoost achieved reliability by emphasising lap-to-lap outcome stability.

The stability of XGBoost can be attributed to its gradient boosting architecture and regularisation mechanisms, such as shrinkage and subsampling. These mechanisms prevent overfitting and enhance robustness across different datasets. This structure enables XGBoost to maintain predictive reliability even when facing unseen race conditions.

These findings are consistent with prior research, such as [26] [30], which observed that XGBoost provides stable performance in structured esports and sim racing environments. However, it sometimes lacks the adaptability of neural network models in highly dynamic scenarios.

Therefore, although XGBoost did not achieve dynamic prediction, its demonstrated reliability and consistent performance across races suggest its value in applications where prediction stability and lap-to-lap result accuracy are critical.

4.2.5 Random Forest: Stable Point-Estimate Specialist

The final key finding indicates that while **Random Forest performs almost as well as XGBoost on lap-to-lap predictions**, it is still less suited for the dynamic, mid-lap ranking required in live podium forecasting. Across the conducted live experiments, Random Forest exhibited strong consistency and the lowest bootstrap variance, finishing only 3.47% behind XGBoost in Top-3 accuracy, 92.83% vs 96.3% , and closely matching it on RMSE (0.91 vs 0.90) and slightly lower MAE (0.10 vs 0.42). Additionally, 2.17 % behind RankNet in top-3 accuracy, 92.83% vs 95% , while having much lower MAE and slightly higher RMSE

Specifically, Random Forest’s Spearman correlation (0.96) sits midway between XGBoost (0.97) and higher than RankNet’s (0.94), confirming that its ordering

skill is solid, just shy of best-in-class. *Conversely, it recorded the lowest MAE (0.10), underlining its strength in producing stable and accurate single-output predictions* such as final lap times or other specific race metrics.

This behaviour can be attributed to Random Forest’s ensemble structure, where numerous independent decision trees are aggregated to predict target outputs. While this architecture excels for static—or lap-end—prediction tasks, it is less capable of capturing the complex, evolving interactions required for accurate rank ordering *within* a lap. (The same lap-end limitation, though with higher accuracy, is shared by XGBoost.)

Previous research by [29, 30] , similarly observed that Random Forest excels in structured prediction tasks but struggles with dynamic ranking or adaptive modelling where relationships between features evolve.

Thus, while Random Forest serves well in applications where predicting exact performance metrics is critical, it remains competitively close to XGBoost in lap-to-lap accuracy. It’s observed that limited ability to model dynamic rank relationships reinforces the preference for models like Random Forest, where small datasets, computational complexity and interpretability are required.

4.3 Comparison with Previous Research

This section critically compares this study’s findings with those of previous simulation and esports literature research. The comparison focuses on three key areas: predictive accuracy, adaptability to real-time data, and model suitability for ranking-based tasks in dynamic environments. The objective is to position this

study’s contributions within the existing literature and identify areas of alignment, divergence, or advancement.

4.3.1 *Alignment with Prior Evidence on Neural Rankers*

These research results suggest that **RankNet offers the best overall when both static accuracy and real-time adaptability are considered**, reproducing the improvements reported by [9]. It has been identified that RankNet cuts MAE by $\approx 19\%$ relative to the strongest non-neural baseline (LightGBM/ARIMA) when forecasting two-lap and stint positions [9]. Although slightly lower in Spearman and top3 accuracy than Random Forest and XGBoost, this is due to its capability for dynamic predictions, rather than focusing on lap-to-lap results.

Likewise, [44] showed that a Siamese ranking network outperformed Random Forest for pairwise ordering of Parkinson-gait signals, reinforcing the versatility of neural ranking in noisy, high-variance time series.

In esports, [27] demonstrated a Two-Stage Spatial-Temporal Network (TSSTN) that surpassed tree ensembles for live *Honour of Kings* win prediction by integrating fine-grained temporal context, an architectural principle also exploited by RankNet’s pairwise loss. This research’s consistently superior generalisation aligns with a growing body of work favouring deep pairwise or sequence models for order-sensitive tasks in dynamic environments.

4.3.2 *Real-Time Adaptability versus Lap-Aggregated Stability*

Another result showed that **RankNet adapts mid-lap, whereas XGBoost and Random Forest react only at lap-to-lap results**. This resonates with claims in deep-learning

esports literature that neural models ingest dense event streams more effectively than boosted tree-based algorithms. [26] claimed XGBoost retained high accuracy in *League of Legends* when using aggregated post-event features, but lagged behind LSTM-based models once second-by-second telemetry was supplied. Similarly, [25] demonstrated that Random Forest was competitive in static, pre-match forecasts, yet offered diminishing gains as the prediction horizon shortened to real-time within the data 2 platform. Therefore, the observations in live-lap testing corroborate a cross-domain trend: tree ensembles excel on temporally summarised inputs. At the same time, neural networks yield stronger minute-by-minute responsiveness when predicting live data streams.

4.3.3 XGBoost’s Cross-Race Robustness in Structured Contexts

The observation that **XGBoost delivers very low inter-race variance** parallels earlier sim-racing studies. [30] showed XGBoost retains $F1 = 0.88$ when classifying driver-skill tiers across multiple Assetto Corsa circuits, attributing the robustness to gradient-boost regularisation. [29] revealed that, once hyperparameter-tuned, XGBoost’s lap-time RMSE remained stable after track layout changes in a Unity-based racing prototype. This research dataset featured wider variation in grid size, race length, and weather conditions. Yet the modest dispersion in Spearman coefficients (across-race $SD \approx 0.03$; within-race bootstrap $SD \approx 0.004$) is consistent with these reports and underscores the method’s strength for structured, lap-resolved outputs.

4.3.4 The Suitability of Random Forests for Point Estimation over Ranking Tasks

Earlier work helps explain why **Random Forest delivered the most precise single-value predictions yet trailed slightly in rank ordering**. [32] applied Random Forest to lap-time prediction in the Formula Student simulation. Although the lap-time error was low, the study did not evaluate podium-ranking accuracy, which was the target metric here. In esports, [26] noted that Random Forest yielded tight kill-count estimates but mis-ranked teams under volatile late-game swings.

Furthermore, Random Forest achieved the *lowest* mean-absolute-error (MAE = 0.10) and a near-identical root-mean-square-error (RMSE $\approx$ 0.91) to XGBoost (0.90), confirming its strength for scalar point estimates. However, its Top-3 accuracy (93.7%) and Spearman correlation ($\rho \approx 0.95$) were modestly lower than those of XGBoost (96.1%, $\rho \approx 0.97$) and just behind RankNet (93.9%, $\rho \approx 0.94$).

The ensemble (a collection of individual models trained on different bootstrap samples) bootstrapped voting mechanism effectively reduces variance in scalar outputs; however, it does not capture the pairwise differences essential for fine-grained ordinal tasks, which is why a small, yet consistent, ranking deficit is observed in both prior studies and this study experimentation.

4.3.5 Why Some Results Diverge from Earlier Studies

These studies revealed two apparent discrepancies when compared with the results of this research. The studies in the literature neither addressed these discrepancies nor provided supporting evidence or discussion that could help explain these discrepancies observed in our results. These discrepancies include research

studies of [9] [26] [30] [27], as they did not report XGBoost beating RankNet on raw Top-3 accuracy in static after lap snapshots. The reasoning behind such claims stems from how these works evaluate global rank correlation, regression error or win probability AUC rather than a podium hit metric. They do not analyse scenarios where a gradient-boosted tree can outperform a neural ranker on raw Top-3 accuracy. Lastly, these studies did not report the use of random forest, showing slightly higher bootstrap stability than RankNet. Because their work evaluated either seed-level variance for neural nets or fold-level variance for tree models, and rarely a full race-bootstrap [9] [26] [30] [27]. None reported a case where Random Forest edges a neural ranker in terms of stability. This research’s whole-race bootstrap methodology reveals that Random Forest’s bagging still delivers the lowest prediction spread, even if its mean accuracy lags behind RankNet.

4.3.6 Contribution to Current Understanding

The cross-study review collectively confirms three progressive insights in motorsport and broader analytics. Firstly, pair-wise (comparing each pair of items to see which one should rank higher) neural architectures are recognised as the state-of-the-art for rank-ordering tasks that process dense, live telemetry streams [27, 34]. Secondly, gradient-boosted trees continue to serve as a pragmatic baseline in situations where updates are required solely at lap-to-lap granularity, or when model transparency and stringent control overfitting are of utmost importance [26, 30]. Thirdly, traditional bagged ensembles, such as Random Forest, remain effective for precise scalar regression targets. However, they require sup-

plementation when the objective is strictly ordinal rather than scalar [4,29].

By evaluating all three model families head-to-head on the same telemetry-rich iRacing dataset rather than through disparate studies, this research more effectively isolates the causes of performance divergence than previous meta-analyses were able to achieve.

4.4 Implications of the Findings

The comparative evaluation of RankNet, XGBoost, and Random Forest yields insights into motorsport analytics, presenting valuable practical and theoretical implications for future research or advisory implementations [16].

RankNet's distinctive capability to update podium predictions in real time during a lap by juxtaposing the retrieved data against other drivers notably enhances strategic decision-making [9]. Concurrently, this suggests that virtual strategy engineers can theoretically anticipate and respond promptly to tactical scenarios such as undercuts, safety-car interventions, and tire wear, thereby affording teams a competitive advantage through informed and timely decisions [27]. However, it is more appropriately regarded as an advisory tool to assist race strategy engineers, rather than serving as a direct replacement for them. Furthermore, an increased volume of racing data will be requisite [30].

Moreover, applying SHAP values provides transparent insights into the key performance factors influencing podium placements across all models [36]. This transparency can potentially inform teams in tailoring driver coaching programs to improve late-race pace and effective incident management. It also furnishes an

objective basis for talent scouting by clearly delineating the skills and behaviours that strongly correlate with successful racing outcomes. Nevertheless, its applicability is limited to certain track types; the dataset encompasses single-car classes and sports car series. Before adopting similar rules in a series of multi-class car racing, a comprehensive feature audit, model tuning, and modification of the API and SDK data collection streams will be necessary to accommodate other types of racing series [26].

Lastly, real-time predictive analytics substantially enrich fan engagement and broadcasting experiences, which have improved by 8%. By providing dynamic and continuously updated visuals of podium probabilities, commentators and streamers can enhance viewer interest and narrative engagement, akin to the real-time analytics extensively employed in traditional sports broadcasting [25]. Although the model output graph necessitates a short rolling average or confidence band, additional racing data will be vital before its implementation in these events.

Theoretically, this study substantiates existing assertions that pairwise neural ranking models, such as RankNet, outperform traditional boosted-tree models in managing high-frequency, sequential data streams [44]. This supports the prior literature and establishes neural rankers as an effective tool in complex, data-intensive predictive scenarios. As previously discussed, traditional boosted tree models remain a viable alternative depending on computational budgets or transparency, which are primary considerations [26].

Finally, this research's comprehensive feature engineering approach exemplifies how raw telemetry can be systematically structured into actionable data inputs.

Techniques such as decay-weighted multi-lap snapshots and database-triggered event flags maintain the granularity of telemetry data [28]. This framework can be repurposed or inspire researchers working with similarly sparse data in esports or amateur motorsports series, albeit it will necessitate significant modifications to cater to their requirements.

4.5 Limitations of the Study

This research aimed to replicate realistic motorsport racing conditions within the iRacing environment; however, several limitations affect the generalizability and broader applicability of the findings [21].

A notable limitation of this study is that the dataset was exclusively sourced from the iRacing platform. The selection of this particular platform constrains the direct applicability of the findings to other recognised simulation platforms, such as Assetto Corsa and rFactor 2, as well as real-world motorsport racing series [30]. Each distinct platform and real-world scenario introduces variations in physics models, damage simulations, and driver demographics, potentially restricting the transferability of the model's insights and performance [16].

Another critical limitation arises from constraints associated with telemetry data. Specifically, the iRacing SDK does not provide continuous XYZ coordinates of the cars, explicit indicators of overtaking events, or comprehensive telemetry data concerning other drivers [40]. Consequently, much of this telemetry data necessitated feature engineering to compensate for the absence of these critical features, which may introduce inaccuracies or noise into the dataset [2].

Furthermore, API rate limits and individual driver privacy settings impeded data collection processes. The retrieval of historical data encountered significant throttling, and specific drivers concealed their performance statistics and race history [41]. This resulted in missing data points, as once detected, the data collection ceases, thereby hampering the overall data-gathering process.

Additionally, while the dataset included approximately sixty races involving around 800 unique drivers, it was adequate for XGBoost and Random Forest analyses [30], yet relatively modest for deep learning approaches such as RankNet [9]. Due to limitations in data collection, 40 races are insufficient to capture the diversity of racing scenarios adequately [16]. Although the metrics from these models suggest they performed exceptionally well, the results would benefit from a larger dataset to enhance both concreteness and accuracy [21]. This would also improve RankNet’s mid-lap ranking accuracy [27].

In terms of computational considerations, the GPU inference latency and the conversion of the database to a flat file, averaging around 2 seconds per prediction, is suitable for use in broadcasting but remains insufficient for real-time, closed-loop driver-assist applications that require immediate feedback within 10 milliseconds or less [28].

Finally, inherent trade-offs exist concerning interpretability. While SHAP values provide valuable post-hoc explanations of model predictions, they do not fully elucidate causal relationships [36]. Additional methodologies, such as feature-masking ablation studies or causal graph analyses, could afford more profound insights into the specific contributions of individual features [22]. Moreover, RankNet

remains considerably less interpretable with SHAP than the XGBoost and Random Forest models.

These considerations should be meticulously evaluated when adapting this analytical framework to other racing series, real-life motorsports, or deploying it in scenarios where precision and safety are paramount.

Chapter 5: Conclusion

5.1 Summary of Key Findings

This study determined which machine-learning algorithms—XGBoost, Random Forest, and RankNet—best predict the top three finishers on the iRacing platform. After reviewing each model’s metrics, the dataset includes streamlined telemetry and historical data from sixty races (800 unique drivers). Results indicate that the optimal approach varies based on specific objectives. Although RankNet was superior due to its real-time accuracy and responsiveness, XGBoost and Random Forest also performed well. All three models achieved podium accuracy and beyond, accurately ranking positions and predicting finishers. RankNet has downsides, including computational demands and interpretability issues, which also apply to XGBoost and Random Forest, though they are less demanding.

Model stability analyses showed that tree-based ensembles (XGBoost and Random Forest) better handle class imbalances and noisy data, while RankNet’s pairwise formulation occasionally overreacted to noise. SHAP interrogation confirmed that decay-weighted livescore-adjusted was the most influential predictor among the three models.

5.2 Answers to the Research Questions

1. Research Question 1 — Highest Accuracy.

XGBoost consistently achieved the highest podium prediction accuracy across

the dataset, reaching approximately 96% accuracy. It outperformed both Ranknet and Random Forest. However, regarding live adaptability and ranking precision, RankNet came out on top, especially in dynamic race conditions where real-time telemetry fluctuations had a more substantial impact.

2. Research Question 2 — Prediction Stability.

While all three algorithms performed strongly, tree-based models (XGBoost and Random Forest) exhibited greater robustness in handling race variability, particularly in cases involving class imbalance or unpredictable incidents. RankNet demonstrated occasional sensitivity to mid-stint telemetry noise but maintained stable rank prediction in most scenarios, particularly on high-sample tracks with consistently high data quality.

3. Research Question 3 — Model Transparency.

SHAP analysis revealed that the most influential predictors across all models were the decay-weighted live NormDriverGapToLeader, average overtake rate, and stable front-runner. While XGBoost and Random Forest provided more transparent feature importances, RankNet required post-hoc interpretability tools but successfully highlighted strategic decision-making features relevant to podium outcomes.

5.3 Hypothesis

The hypothesis proposed that integrating real-time telemetry data and historical driver statistics into rank-aware models would significantly increase the predictive

accuracy for podium outcomes, in contrast to regression or classification-based models.

This study's findings supported the hypothesis by demonstrating that XGBoost, Random Forest, and Ranknet could predict the podium driver accuracy over 90% and managed to go beyond by accurately predicting the actual finishing orders, rather than just the podium drivers. The success is credited to extensive feature engineering, as indicated by the feature importance, data collection, and storage pipeline design. As all models achieve above 90% Spearman rank correlation alongside strong validation metrics such as group k-fold, bootstrap and learning curves. RankNet outperformed both XGBoost and Random Forest in terms of responsiveness during live race scenarios, while XGBoost and Random Forest offered greater interpretability and robustness under challenging race conditions. RankNet's rank-optimised architecture consistently exhibited exceptional performance in aligning live telemetry fluctuations and close one-to-one driver comparisons with accurate positional forecasting, confirming the core assumption of this research.

5.4 Recommendations for Future Work

For future work recommendations, this research proposes extending the data collection dataset by at least 250-500 races to include all possible racing conditions, such as extreme incidents, close driver-to-driver battles, variable weather conditions, and other scenarios, to make these models more accurate and generalizable. Coupled with this is a request to the iRacing platform to gain permission to ac-

cess more in-depth driver telemetry data on other drivers, such as tire wear, fuel levels, brakes % , and more. This will unlock a more in-depth analysis of driver performance for models. Additionally, potentially combine [42] research on performance prediction to further benchmark these models or SHAP interpretations and indicate which features contribute to being on the podium. Lastly, this research should be expanded to other virtual racing platforms, such as *rFactor 2* and *Assetto Corsa Competizione* , to further cross-test the model's validity rather than relying on a single platform.

5.5 Final Reflection

Virtual motorsports is not solely for entertainment; it can also serve as a platform for serious drivers to refine their skills in a safe and controlled environment. It contains extensive motorsport analytics for various predictive applications. By demonstrating the use of Ranknet, XGBoost, and Random Forest to predict podium drivers using live telemetry and historical data of drivers, it is possible to forecast the podium drivers and the actual order of finishers. Additionally, SHAP highlights the variables that can be attributed to being on the podium. This research lays the foundation for future research to expand virtual motorsport analytics in rank-based prediction and model comparison. It also demonstrates potential applications, such as immersive broadcasting, virtual strategy aids, and more. The methodology of combining live and historical data would aid future similar research and enhance the reproducibility of this research, thereby expanding the frontier of esports analytics.

List of References

- [1] K. R. Dalal, “Analysing the role of supervised and unsupervised machine learning in IoT,” in *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, 2020, pp. 75–79. [Online]. Available: <https://ieeexplore.ieee.org/document/9155761/?arnumber=9155761>
- [2] P. Hart, L. He, T. Wang, V. S. Kumar, K. Aggour, A. Subramanian, and W. Yan, “Application of big data analytics and machine learning to large-scale synchrophasor datasets: Evaluation of dataset ‘machine learning-readiness’,” *IEEE Open Access Journal of Power and Energy*, vol. 9, pp. 386–397, 2022. [Online]. Available: <https://ieeexplore.ieee.org/document/9852477/?arnumber=9852477>
- [3] A. N. Wilson, K. A. Gupta, B. H. Koduru, A. Kumar, A. Jha, and L. R. Cenkeramaddi, “Recent advances in thermal imaging and its applications using machine learning: A review,” *IEEE Sensors Journal*, vol. 23, no. 4, pp. 3395–3407, 2023. [Online]. Available: <https://ieeexplore.ieee.org/document/10014654/?arnumber=10014654>
- [4] A. Akilandeswari, B. V. Kumar, S. Padmakala, T. Tirumalaikumari, R. Kesavan, and M. J. Carmel Mary Belinda, “Comparative analysis of XGBoost and random forest for predicting water potability,” in *2023 2nd International Conference on Automation, Computing and*

- Renewable Systems (ICACRS)*, 2023, pp. 1656–1659. [Online]. Available: <https://ieeexplore.ieee.org/document/10405315/?arnumber=10405315>
- [5] T. Chauhan, S. Rawat, S. Malik, and P. Singh, “Supervised and unsupervised machine learning based review on diabetes care,” in *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*, vol. 1, 2021, pp. 581–585. [Online]. Available: <https://ieeexplore.ieee.org/document/9442021/?arnumber=9442021>
- [6] Encord, “Mastering supervised learning: A comprehensive guide,” 2023. [Online]. Available: <https://encord.com/blog/mastering-supervised-learning-a-comprehensive-guide/>
- [7] S. R. Gayam, R. R. Yellu, and P. Thuniki, “Artificial intelligence for real-time predictive analytics: Advanced algorithms and applications in dynamic data environments,” *Distributed Learning and Broad Applications in Scientific Research*, vol. 7, pp. 18–37, 2021.
- [8] A. R. Naini, S. Subramaniam, S.-G. Leem, and C. Busso, “Combining relative and absolute learning formulations to predict emotional attributes from speech,” in *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2023, pp. 1–8. [Online]. Available: <https://ieeexplore.ieee.org/document/10389753/>
- [9] B. Peng, J. Li, S. Akkas, T. Araki, O. Yoshiyuki, and J. Qiu, “Rank position forecasting in car racing,” in *2021 IEEE International Parallel and*

- Distributed Processing Symposium (IPDPS)*, 2021, pp. 724–733. [Online]. Available: <https://ieeexplore.ieee.org/document/9460538/?arnumber=9460538>
- [10] UBI AI, “Data labeling with unsupervised learning,” 2023. [Online]. Available: <https://ubiai.tools/data-labeling-with-unsupervised-learning/>
- [11] R. Genuer and J.-M. Poggi, *Random Forests with R*, ser. Use R! Springer International Publishing, 2020.
- [12] P. Li, M. Abdel-Aty, and J. Yuan, “Real-time crash risk prediction on arterials based on LSTM-CNN,” *Accident Analysis and Prevention*, vol. 135, p. 105371, 2020. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0001457519311108>
- [13] Miro, “Supervised learning diagram 1,” 2023. [Online]. Available: https://miro.medium.com/v2/resize:fit:660/1*REUu0LJytEt6kIOFQeB0Tw.png
- [14] —, “Supervised learning diagram 2,” 2023. [Online]. Available: https://miro.medium.com/v2/resize:fit:592/1*i0o8mjFfCn-uD79-F1Cqkw.png
- [15] GeeksforGeeks, “Neural networks architecture,” 2023. [Online]. Available: <https://media.geeksforgeeks.org/wp-content/cdn-uploads/20230602113310/Neural-Networks-Architecture.png>
- [16] G. Heilmeier, J. R. Smith, and A. P. Kumar, “Virtual strategy engineer: Using artificial neural networks for making race strategy decisions in circuit motorsport,” Master’s thesis, 2020. [Online]. Available: <https://www.mdpi.com/2076-3417/10/21/7805>

- [17] S. Koh, B. Zhou, H. Fang, P. Yang, Z. Yang, Q. Yang, L. Guan, and Z. Ji, “Real-time deep reinforcement learning based vehicle navigation,” *Applied Soft Computing*, vol. 96, p. 106694, Nov 2020.
- [18] H. A. Javaid, “Ai-driven predictive analytics in finance: Transforming risk assessment and decision-making,” *Advances in Computer Sciences*, vol. 7, no. 1, 2024.
- [19] Y. Chen, J. Mei, and S. Huang, “Machine learning and data mining on the innovation of e-sports industry,” *International Journal of Education and Information Technologies*, vol. 14, pp. 121–132, 2020.
[Online]. Available: [https://www.naun.org/main/NAUN/educationinformation/2020/a302008-015\(2020\).pdf](https://www.naun.org/main/NAUN/educationinformation/2020/a302008-015(2020).pdf)
- [20] H. A. Javaid, “AI-driven predictive analytics in finance: Transforming risk assessment and decision-making,” *Advances in Computer Sciences*, vol. 7, 2024.
- [21] S. E. Whang, Y. Roh, H. Song, and J.-G. Lee, “Data collection and quality challenges in deep learning: A data-centric ai perspective,” Dec 2022, accessed: November 6, 2024. [Online]. Available: <http://arxiv.org/abs/2112.06409>
- [22] R. González-Sendino, E. Serrano, and J. Bajo, “Mitigating bias in artificial intelligence: Fair data generation via causal models for transparent and explainable decision-making,” *Future Generation Computer*

- Systems*, vol. 155, pp. 384–401, 2024. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0167739X24000694>
- [23] Kabir Singh Chadha, “Bias and fairness in artificial intelligence: Methods and mitigation strategies,” *International Journal for Research Publication and Seminar*, vol. 15, no. 3, pp. 36–49, 2024. [Online]. Available: <https://jrps.shodhsagar.com/index.php/j/article/view/1425>
- [24] R. Mühlhoff, “Predictive privacy: towards an applied ethics of data analytics,” *Ethics Inf Technol*, vol. 23, no. 4, p. Art. no. 4, Dec 2021.
- [25] V. J. Hodge, S. Devlin, N. Sephton, F. Block, P. I. Cowling, and A. Drachen, “Win prediction in multiplayer esports: Live professional match prediction,” *IEEE Transactions on Games*, vol. 13, no. 4, pp. 368–379, 2021. [Online]. Available: <https://ieeexplore.ieee.org/document/8906016/>
- [26] F. Bahrololloomi, S. Sauer, F. Klonowski, R. Horst, and R. Dörner, “A machine learning based analysis of e-sports player performances in league of legends for winning prediction based on player roles and performances:,” in *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. SCITEPRESS - Science and Technology Publications, 2022, pp. 68–76. [Online]. Available: <https://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0010895900003124>
- [27] Z. Yang, Z. Pan, Y. Wang, D. Cai, X. Liu, S. Shi, and S.-L. Huang, “Interpretable real-time win prediction for honor of kings, a popular mobile

- MOBA esport,” 2021. [Online]. Available: <http://arxiv.org/abs/2008.06313>
- [28] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger, “Learning-based model predictive control for autonomous racing,” *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3363–3370, 2019, conference Name: IEEE Robotics and Automation Letters. [Online]. Available: <https://ieeexplore.ieee.org/document/8754713>
- [29] A. Picardi, “A comparison of different machine learning techniques to develop the AI of a virtual racing game,” 2021. [Online]. Available: <https://webthesis.biblio.polito.it/18168/>
- [30] F. Hojaji, A. J. Toth, and M. J. Campbell, “A machine learning approach for modeling and analyzing of driver performance in simulated racing,” in *Artificial Intelligence and Cognitive Science*, L. Longo and R. O’Reilly, Eds. Springer Nature Switzerland, 2023, pp. 95–105.
- [31] B. Msakamali, “F1 data analysis and tactical insights: Exploring formula 1 race performance strategies,” Master’s thesis, Tampereen ammattikorkeakoulu, Tampere, Finland, 2024, uRN: urn:nbn:fi:amk-2024051712928. [Online]. Available: https://www.theseus.fi/bitstream/handle/10024/856650/Msakamali_Baraka.pdf?sequence=6
- [32] S. A. Löckel, “Machine learning for modeling and analyzing of race car drivers,” Master’s thesis, Technische Universität Darmstadt, Darmstadt, 2022, accessed: 2024-11-30. [Online]. Available: <https://tuprints.ulb.tu-darmstadt.de/20218/>

- [33] E. O’Hanlon, “Using supervised machine learning to predict the final rankings of the 2021 formula one championship,” Master’s thesis, Dublin, National College of Ireland, 2022.
- [34] B. Peng, J. Li, S. Akkas, F. Wang, T. Araki, O. Yoshiyuki, and J. Qiu, “Rank position forecasting in car racing,” 2020. [Online]. Available: <http://arxiv.org/abs/2010.01707>
- [35] O.-A. Cheteles, “Feature importance versus feature selection in predictive modeling for formula 1 race standings,” *arXiv preprint arXiv:2405.00000*, 2024.
- [36] A. B. Parsa, A. Movahedi, H. Taghipour, S. Derrible, and A. K. Mohammadian, “Toward safer highways, application of XGBoost and SHAP for real-time accident detection and feature analysis,” *Accident Analysis & Prevention*, vol. 136, p. 105405, 2020. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0001457519311790>
- [37] E. Stoppels, “Predicting race results using artificial neural networks,” Master’s thesis, University of Twente, 2018. [Online]. Available: https://essay.utwente.nl/75317/1/Stoppels_MA_EEMCS.pdf
- [38] S. J. Kelly, S. Derrington, and S. Star, “Governance challenges in esports: a best practice framework for addressing integrity and wellbeing issues,” *International Journal of Sport Policy and Politics*, vol. 14, no. 1, pp. 151–168, 2022. [Online]. Available: <https://www.tandfonline.com/doi/full/10.1080/19406940.2021.1976812>

- [39] J. L. Kröger, P. Raschke, J. Percy Campbell, and S. Ullrich, “Surveilling the gamers: Privacy impacts of the video game industry,” *Entertainment Computing*, vol. 44, p. 100537, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S187595212200060X>
- [40] iRacing.com, “iracing terms of use and end user license agreement,” 2024. [Online]. Available: <https://www.iracing.com/terms-use-eula/>
- [41] —, “iracing privacy policy,” 2024. [Online]. Available: <https://www.iracing.com/privacy-policy/>
- [42] F. Hojaji, A. Toth, J. Joyce, and M. Campbell, “AI-enabled prediction of sim racing performance using telemetry data,” *Computers in Human Behavior Reports*, vol. 14, 2024.
- [43] M. Köppel, A. Segner, M. Wagener, L. Pensel, A. Karwath, and S. Kramer, “Pairwise learning to rank by neural networks revisited: Reconstruction, theoretical analysis and practical performance,” in *Machine Learning and Knowledge Discovery in Databases*, U. Brefeld, E. Fromont, A. Hotho, A. Knobbe, M. Maathuis, and C. Robardet, Eds. Springer International Publishing, 2020, vol. 11908, pp. 237–252. [Online]. Available: http://link.springer.com/10.1007/978-3-030-46133-1_15
- [44] B. B. Ogul and S. Ozdemir, “A pairwise deep ranking model for relative assessment of parkinson’s disease patients from gait signals,” *IEEE Access*, vol. 10, pp. 6676–6683, 2022. [Online]. Available: <https://ieeexplore.ieee.org/document/9656115/>

Chapter A: Data Collection

normal data gathering script workflow to gather data from specated races

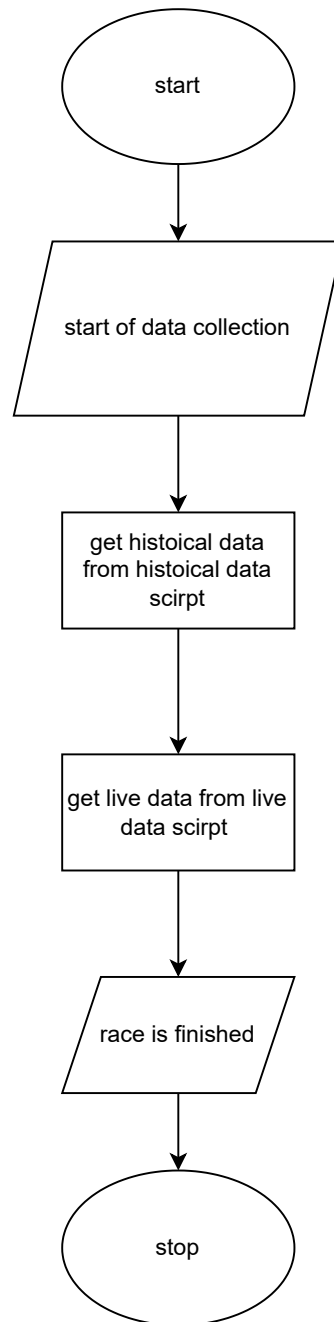


Figure A.1: This diagram illustrates the complete process of data collection from the iRacing API and SDK. Historical race data is merged with live telemetry to create a structured dataset for predictive modelling.

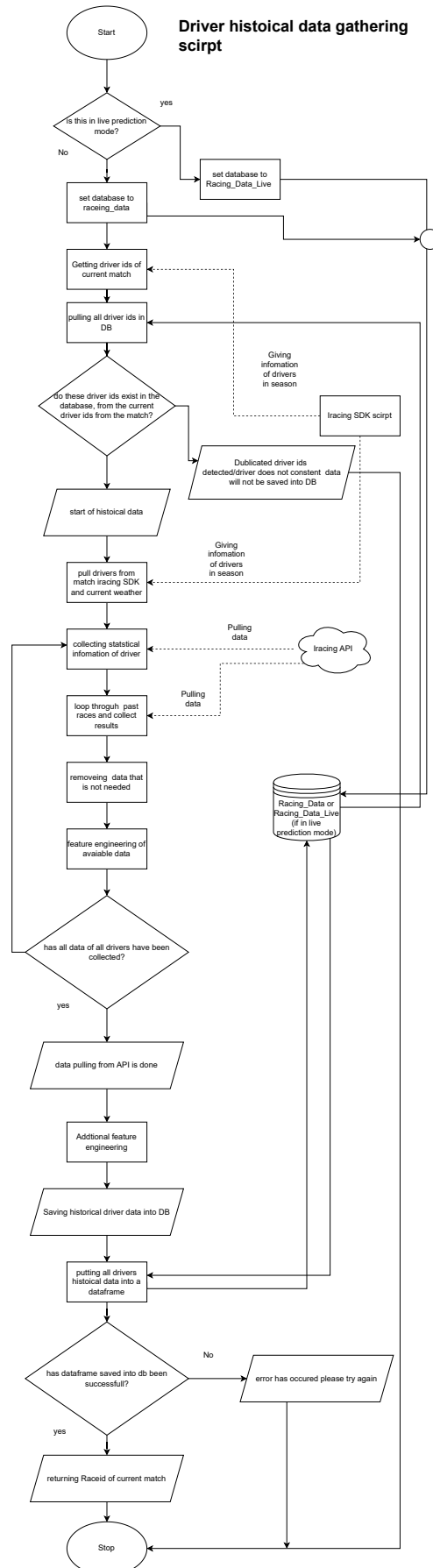


Figure A.2: Details the retrieval of past race results, driver statistics, and track conditions via the iRacing API.

Live data gathering script racing SDK Pre-processing and feature engineering

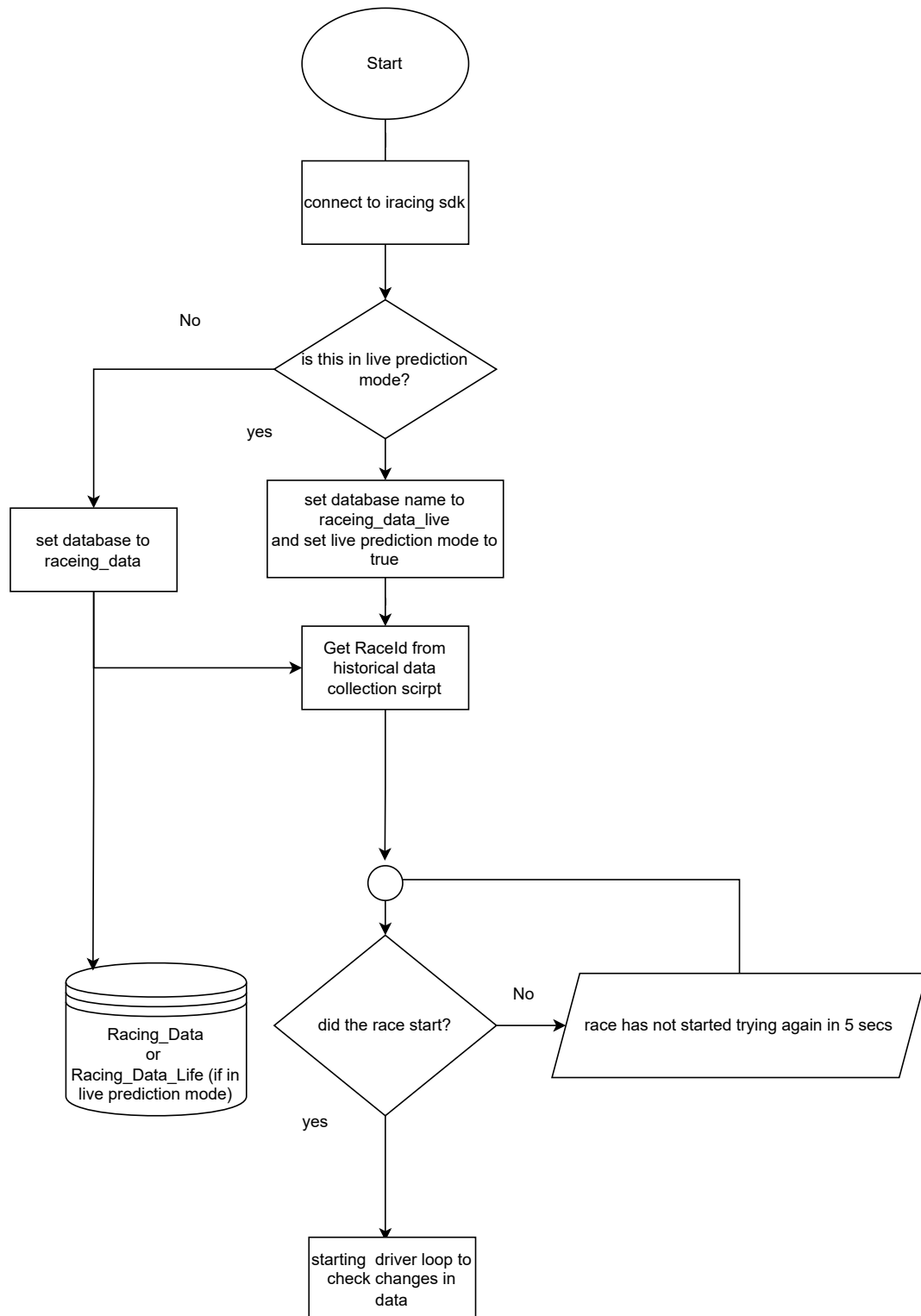


Figure A.3: Shows how real-time telemetry is captured, cleaned, and stored, with automatic event detection and feature engineering. part 1

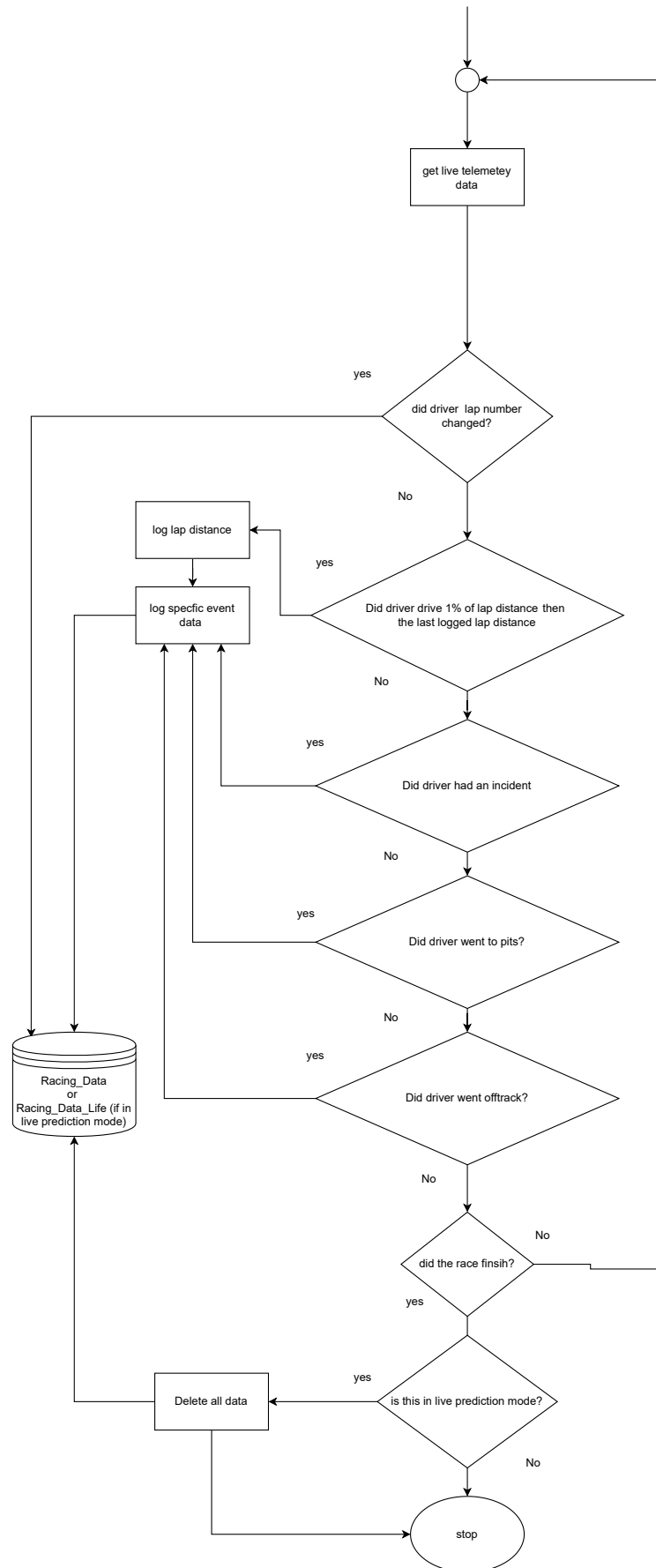


Figure A.4: part 2

Chapter B: Database Schema

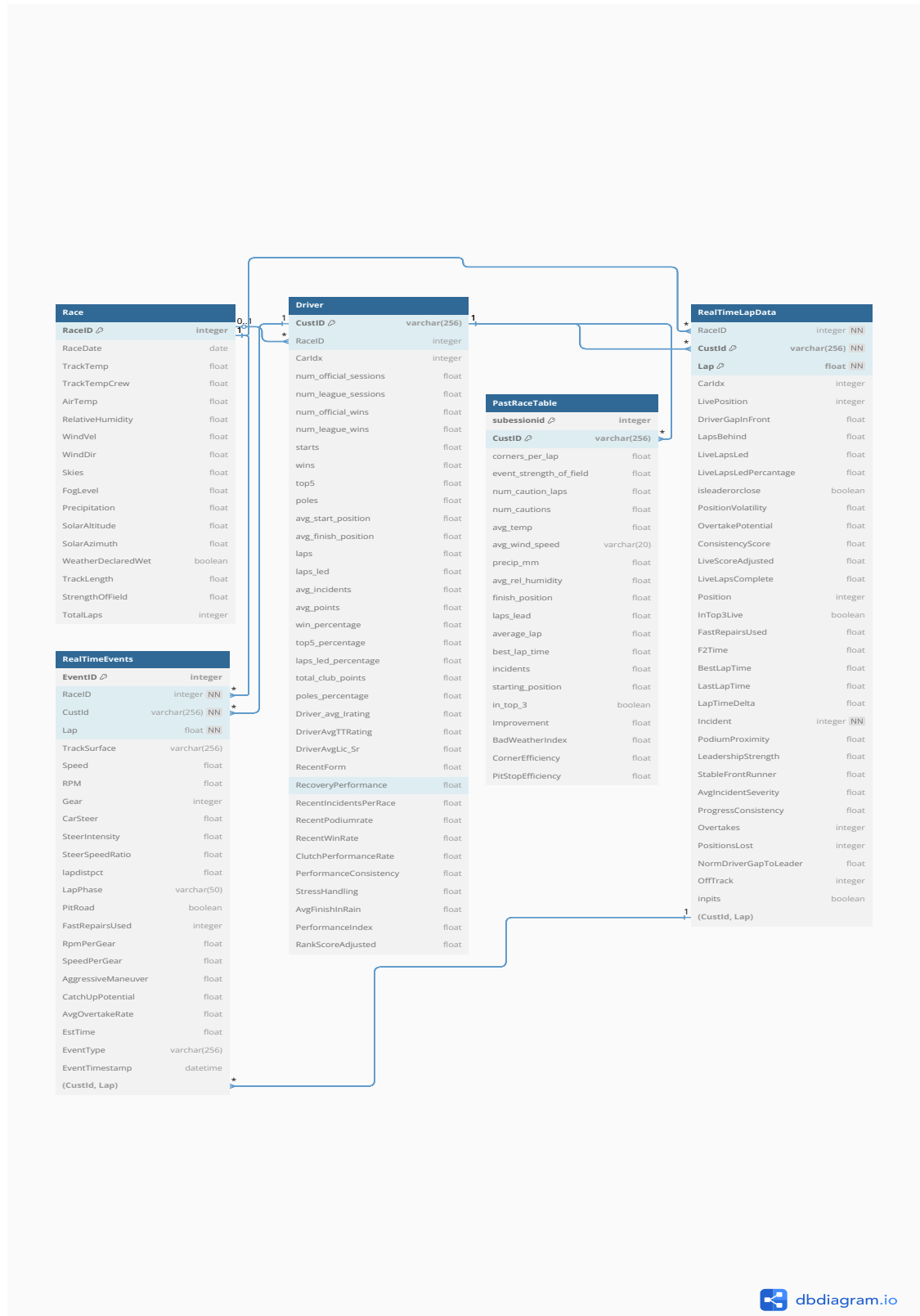


Figure B.1: Displays the relational database design used to store race results, driver stats, and real-time telemetry.

Chapter C: Model Training, and Evaluation

model training and model evulation scirpt

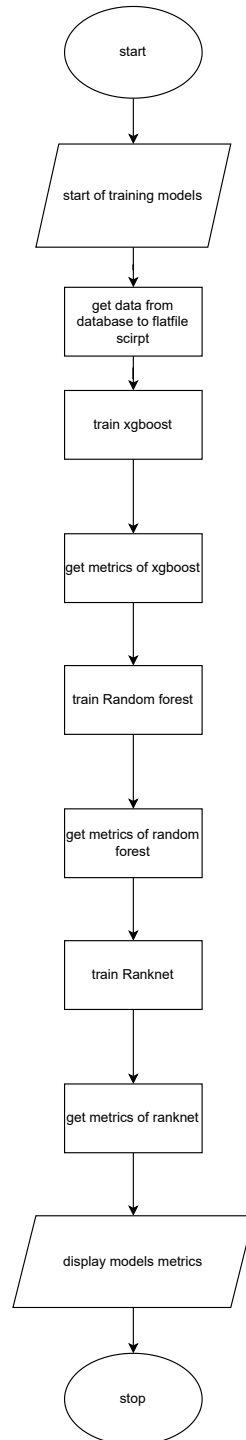


Figure C.1: Illustrates how different models (XGBoost, Random Forest, and RankNet) are trained using historical and live racing data. Also shows the process of computing performance metrics for each model.

Chapter D: Live Prediction

**Live prediction workflow to
get historical data and live
data aswell to live predict and
show predicted results**

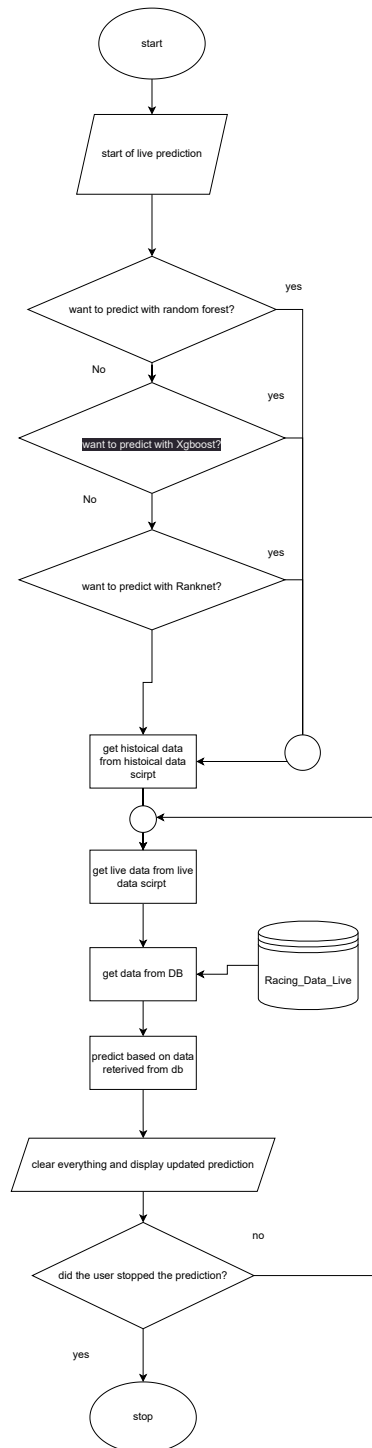


Figure D.1: Displays how live prediction works.

Chapter E: Feature engineering and database to a flat file

These feature engineering techniques are extracted from the database and output from the flowchart of Figure E.1.

Core, Historical, Live-session & SQL-Trigger Features

1. **StrengthOfField:** $\frac{(\sum \text{Driver_avg_Irating}) \times 100 \times N}{N}$
2. **PerformanceIndex:**
 $0.6 \cdot \max(\text{norm_wins}, \text{norm_winsalt}) + 0.5 \cdot \text{norm_poles} + 0.4 \cdot \text{norm_avg_finish} +$
 $0.3 \cdot \text{norm_laps_led_pct} + 0.2 \cdot \text{norm_consistency} - 0.3 \cdot \text{norm_avg_incidents}$
3. **Improvement:** $|\text{best_lap_time} - \text{average_lap}|$
4. **BadWeatherIndex:** $\text{avg_temp} + \text{avg_wind_speed} + \text{precip_mm}$
5. **CornerEfficiency:** $\frac{\text{average_lap}}{\text{corners_per_lap}}$
6. **PitStopEfficiency:** $\frac{\text{best_lap_time}}{\text{average_lap}}$
7. **RecentForm:** $\text{mean}(\text{finish_position})$
8. **RecoveryPerformance:** $\text{mean}(\text{finish_pos} - \text{start_pos})$
9. **RecentIncidentsPerRace:** $\frac{\sum \text{incidents}}{\# \text{races}}$
10. **RecentPodiumRate:** $\frac{\#(\text{finish} \leq 3)}{\# \text{races}}$
11. **RecentWinRate:** $\frac{\# \text{wins}}{\# \text{races}}$
12. **ClutchPerformanceRate:** $\frac{\#(\text{start} \geq 5 \wedge \text{finish} \leq 3)}{\# \text{races}}$
13. **PerformanceConsistency:** $\text{stdev}(\text{finish_position})$
14. **AvgFinishInRain:** $\text{mean}(\text{finish_position} \mid \text{precip_mm} \geq 1)$
15. **StressHandling:** $\text{mean}(\text{BadWeatherIndex})$
16. **LapPhase:** categorical — “Start” ($\leq 25\%$ lap), “Mid”, “Near End”, “End”.
17. **SteerIntensity:** $\frac{\text{CarSteer}}{3}$; **SteerSpeedRatio:** $\text{SteerIntensity} \times \text{Speed}$

18. **RpmPerGear:** $\frac{\text{RPM}}{\text{Gear}}$; **SpeedPerGear:** $\frac{\text{Speed}}{\text{Gear}}$
19. **AggressiveManeuver:** boolean — $(|\text{SteerIntensity}| \geq 1) \wedge (\text{Speed} > 150)$.
20. **ConsistencyScore:** $\frac{\text{best_lap_time}}{\text{last_lap_time}}$
21. **OvertakePotential:** $\frac{\text{adjusted_time}}{\text{last_lap_time}}$
22. **LeadershipStrength:**

$$\begin{cases} \text{LiveLapsLedPct} \times \text{ConsistencyScore}, & \text{if pos} = 1 \\ \max(0, \frac{\text{adjusted_time}}{\text{last_lap_time}}) \times \text{ConsistencyScore}, & \text{else} \end{cases}$$
23. **CatchUpPotential:** $\frac{\text{lapdistpct}^2 \times \text{OvertakePotential}}{1 + \text{gap_in_front}}$
24. **AvgOvertakeRate:** $\frac{\text{lapdistpct}}{1 + \text{gap_in_front}}$
25. **AvgIncidentSeverity:** $\frac{\text{incidents}}{\text{completed_laps}}$
26. **PodiumProximity:** 1 if $\text{pos} \leq 3$ else $1/(1 + \text{gap_in_front})$.
27. **StableFrontRunner:**

$$\text{ConsistencyScore} \times \left(1 - \frac{\text{incidents}}{1 + \text{incidents}}\right) \times (\text{LiveLapsLedPct} + 1)$$
28. **ProgressConsistency:** $\frac{1}{1 + \text{stdev}(\text{last } 5 \text{ lap_times})}$
29. **LivePosition:** deterministic rank after sorting (Lap ↓, LapsBehind ↑, lapdistpct ↓, EstTime ↑, CarIdx ↑).
30. **LiveScoreAdjusted:** $\frac{\text{MaxPos} - \text{Position}}{\text{MaxPos} - \text{MinPos}}$
31. **LapCompletionRate:**
$$\begin{cases} \frac{2.5 \times \text{Lap}}{\text{MaxLap}}, & \text{if final lap} \\ \frac{\text{Lap}}{\text{MaxLap}}, & \text{otherwise} \end{cases}$$
32. **InTop3Live:** 1 if $\text{LivePosition} \leq 3$ else 0
33. **WeightedLiveScoreAdjusted:** $\text{LiveScoreAdjusted} \times \text{LapCompletionRate}$
34. **RankScoreAdjusted:** $\max(\text{WeightedLiveScoreAdjusted})$ (per driver across all laps)
35. **PositionVolatility:** $\text{stdev}(\text{Position})$
36. **LapTimeDelta:** $\text{BestLapTime} - \text{LastLapTime}$

37. **Overtakes:** $\max(0, \text{prevPos} - \text{currPos})$
38. **PositionsLost:** $\max(0, \text{currPos} - \text{prevPos})$
39. **NormDriverGapToLeader:** $\frac{\text{MaxCumGap} - \text{CumGap}}{\text{MaxCumGap} - \text{MinCumGap}}$
 - **InPits:** increments by 1 whenever `TrackSurface = "In Pit Stall"` from `eventstable` in a specific lap that is ongoing.
 - **Incidents:** increments by 1 whenever $\text{rpm} \leq 3500$ and $\text{speed} \leq 150$ from `eventstable` (note: speed is inflated) in a specific lap that is ongoing.
 - **Offtrack:** increments by 1 whenever $\text{rpm} \geq 3500$ and $\text{speed} \geq 150$ from `eventstable` (note: speed is inflated) in a specific lap that is ongoing.
40. **HistAvg-*(family):** `HistAvg_col = mean(col)` Across all stored past races for that driver.

Additional Features Generated in `FetchDatabaseData.py`

1. **Race_* & Drv_* copies:** Every non-primary-key column from the `Race` and `Driver` tables is copied verbatim into the model row.
2. **Bucket-decayed telemetry means:** For each numeric column `col` and each anchor bucket $B \in \{\text{First}, 25, 50, 75\}$,

$$\text{col}_B = \text{mean}(\text{col on bucket window}) \times \text{DECAY}_B, \quad \text{DECAY} = \left\{ \frac{1}{6}, \frac{2}{6}, \frac{3}{6}, \frac{4}{6} \right\}.$$
 The window length is $\max(3, \lceil 0.06 \times \text{TotalLaps} \rceil)$.
3. **Bucket event counts:**

$$\text{Cnt}_{E,B} = \# \text{events of type } E \text{ in bucket } B \times \text{DECAY}_B, \quad E \in \{\text{lapcomplete}, \text{positionchange}, \text{inpits}, \text{incident}, \text{offtrack}\}.$$
4. **Bucket event means:**

$$\text{col}_{B,E} = \text{mean}(\text{col within events of type } E \text{ in } B) \times \text{DECAY}_B.$$
5. **Last-5-lap recency pulse:** Features such as `Cnt_incident_last5` or `Pct_lapcomplete_last5` computed solely from The most recent five laps.
6. **Checkpoint roll-ups:** For each event group G :

$$G_count = \sum_B \text{Cnt}_{G,B}, \quad G_last3 = \sum_{B \in \text{three latest}} \text{Cnt}_{G,B}.$$
7. **laps_since_pit:** `RaceLap - last_pit_lap` (with `last_pit_lap` forward-filled per driver).
8. **Zero-fill sentinel rule:** Any column matching `*_count`, `*_last3`, or containing `{incident, off-track, inpits, pit_stop}` receives 0 wherever the raw value is NA, ensuring no Missing values reach the model.

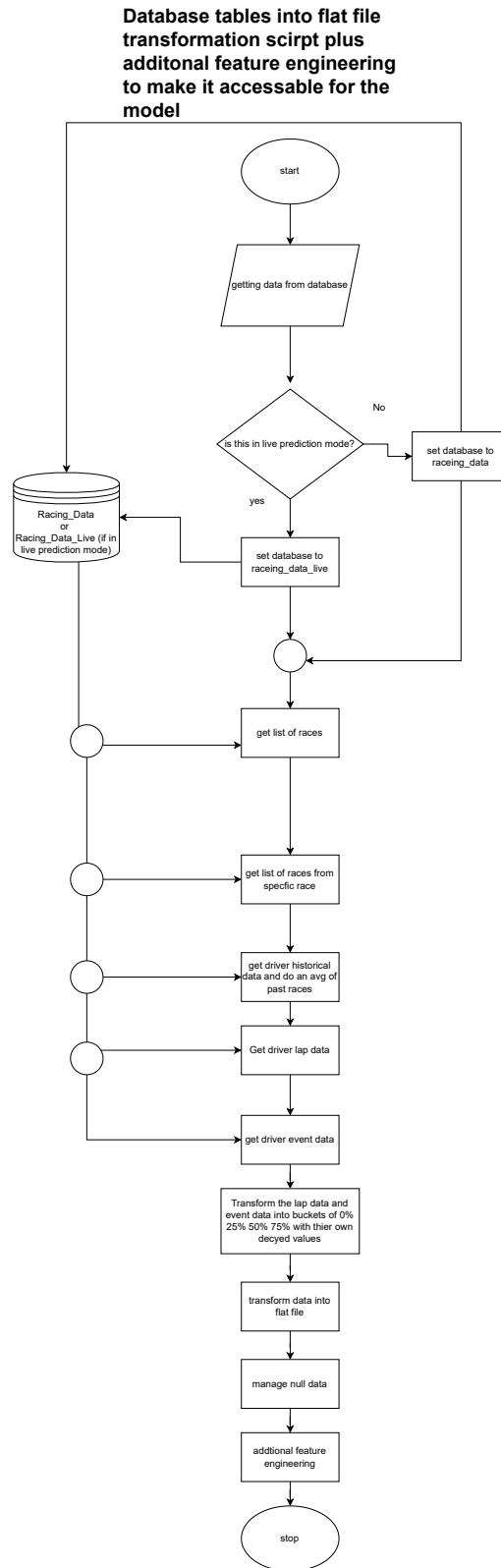


Figure E.1: This diagram shows the process of retrieving structured race and driver data from the database, transforming it into a machine-learning-ready format (flat file), and handling missing values.

Chapter F: Multimedia Supplemental Materials

Appendix Label	Video Title	Description	URL
D.1	Data Collection Demo	Shows how the Python scripts connect to iRacing's SDK/API, pull lap and driver data, and insert it into the relational database in real time.	https://www.youtube.com/watch?v=Zpz-cAR1img
D.2	Model Training Walkthrough	Demonstrates how the historical dataset is preprocessed, each candidate model (RankNet / Random Forest / XGBoost) is trained, and the resulting artifacts are saved.	https://www.youtube.com/watch?v=E50G0CWH2YA
D.3	RankNet Live Prediction Overlay	Live feed showing the RankNet model computing and overlaying top-3 podium probabilities mid-race.	https://www.youtube.com/watch?v=cBc5tH4wp8g
D.4	Random Forest Live Prediction	Demonstrates how the Random Forest model updates its top-3 picks on a live race feed, lap per lap	https://www.youtube.com/watch?v=q01jzOHJHSE
D.5	XGBoost Live Prediction	Demonstrates how the XGBoost model predicts lap to lap similarly to Random Forest.	https://www.youtube.com/watch?v=a_iTzXFFBgC

Table F.1: List of Demo Videos Referenced in This Appendix

The full source code of the project URL: <https://github.com/AntoineSpiteri2/predicting-the-top3-drivers-and-comparing-machine-learning-algorithms-in-iracing>