Submitted in fulfillment of the requirements for the degree of
Master of Science in Computer Science:
Data Science and Artificial Intelligence

Data-driven Approach for Road Cycling Performance Prediction

Pravesh Gurung

Adviser: Leonid Kholkine

Supervisor: Steven Latré

Faculty of Science
2022

Contents

Acknowledgements

First of all, I would like to express my gratitude to my supervisor, Prof. Dr. Steven Latré allowing me to work on this thesis. I am also very grateful to my adviser, MSc. Leonid Kholkine, for providing me with valuable feedback. Thanks to his guidance, I was able to bring this thesis to completion.
I am thankful to my parents who have supported me every step of the way. Thank you for providing me with everything I needed to complete my education. Without you, I would not be here today. Finally, I would like to thank my beloved cat for the comfort and companionship she provided throughout this journey.

Samenvatting

Naarmate de hoeveelheid verzamelde data van wielerwedstrijden toeneemt, kunnen modellen die nuttige informatie uit de gegevens halen, coaches en teams helpen om optimale beslissingen te nemen. Een predictive performance model gericht op het classificeren van renners in de top 20 is zo'n manier om de teams te helpen nuttige informatie te verkrijgen. Om dit te bereiken werd een analyse uitgevoerd van de rankingsystemen dat laat zien hoe goed PCS het UCI rankingsysteem kon weergeven.

Alle data over renners en wedstrijden werden verzameld van de PCS-website en bestaat uit data van het jaar 1995 tot 2022. Om de modellen te bouwen werden classificatiealgoritmen en gradient boosting-algoritmen gebruikt. Er werd een rolling cross-validation toegepast om rekening te houden met de temporele aard van de data en om te voorkomen dat tijdens de training informatie weglekt die tot biased voorspellingen zou kunnen leiden. Er werden speciale imputatietechnieken toegepast om ontbrekende data te verwerken en de modellen werden geëvalueerd met behulp van Precision, Recall en F1-score metrieken.

De modellen die gebruikt maakten van de speciale imputatietechniek gebaseerd op K-Nearest Neighbour (KNN) imputatie, presteerden beter wanneer de algoritmen de ontbrekende waarden met hun ingebouwde methode behandelden. De boosting-algoritmen haalden vergelijkbare scores en presteerden het best over alle metriek, waarbij eXtreme Gradient Boosting (XGBoost)

iets beter presteerde als men kijkt naar de gemiddelde scores over de train-, validatie- en testsets.

Door middle van een Shapley Additive Explanation (SHAP) studie van het model, toonde het thesis aan dat de huidige rang en punten van een renner de grootste invloed hebben op de voorspelling van het model, maar ook het consequent winnen van etappekoersen grote impact had bij aan het behalen van een top 20 klassering. De lengte van de carrière van een renner was een andere factor die een significante invloed had op de uitkomst van het model.

De resultaten van het thesis geven aan dat het model kan worden gebruikt om de prestaties van een renner op een redelijk niveau te voorspellen.

Summary

As the amount of data collected from road cycling races increases, models that can extract useful information from the data can help coaches and teams in making optimized decisions. A predictive performance model focused on classifying riders into whether they would achieve a top 20 ranking is one such way to help the teams gain insightful information. To accomplish this, an analysis was conducted to assess the effectiveness of the PCS ranking system in capturing the UCI ranking system.

All the data on the riders and races were collected from the PCS website and consist of data from the years 1995 to 2022. Classification algorithms and gradient boosting algorithms were used to build the models. A rolling cross-validation approach was implemented to account for the temporal nature of the data and prevent future information leaking during training that could lead to biased predictions. Special data imputation techniques were applied to handle missing data and the models' performance was evaluated using Precision, Recall, and F1-score metrics.

The models that utilized the special imputation technique based on KNN imputation performed better than those that relied on the algorithms' built-in methods for handling missing values. The boosting algorithms demonstrated comparable scores and performed the best across all metrics, with XGBoost performing slightly better when considering the average scores across the train, validation, and test sets.

Through a SHAP study of the model, the thesis showed that a rider's current rank and points have the largest impact on the model's prediction while consistently winning stage races was also a large contributor in a rider securing a top 20 ranking. Additionally, the length of a rider's career was another factor that had a significant impact on the model's outcome.

The results of the thesis indicate that the model can be used to predict rider performance to a reasonable level.

List of Abbreviations

BMI Body Mass Index

GRU Gated Recurrent Unit

KNN K-Nearest Neighbour

LightGBM Light Gradient-Boosting Machine

LSTM Long short-term memory

LTR Learn-to-Rank

MLE Maximum Likelihood Estimation

NDCG Normalized Discounted Cumulative Gain

PCS Pro Cycling Stats

SHAP Shapley Additive Explanation

SMOTE Synthetic Minority Over-Sampling Technique

UCI Union Cycliste Internationale

XGBoost eXtreme Gradient Boosting

List of Tables

List of Figures

Data-driven Approach for Road Cycling Performance Prediction

Pravesh Gurung
University of Antwerp
IDLab, Belgium

Leonid Kholkine
University of Antwerp
IDLab, Belgium

Steven Latré
University of Antwerp
IDLab, Belgium

***Abstract*—This paper explores the potential of predictive performance models in road cycling. The study focuses on classifying a rider's potential for a top 20 ranking. To aid our research, a comparative rank analysis is conducted between the official ranking system, UCI and an alternative ranking system, PCS. The results show the PCS ranking system to be effective in capturing the top rankings of the UCI ranking system. The data is collected from the PCS website and spans from 1995 to 2022. Classification and gradient boosting algorithms are employed, which are further enhanced through techniques like data imputation and rolling cross-validation. The results indicate that boosting algorithms, particularly XGBoost, demonstrate superior performance. A SHAP analysis highlights the key features which include current rank, points, stage race wins, and career length. The research highlights the viability of the model, providing insights into critical factors that influence rider performance.**

I. Introduction

In recent years, data analytics has seen increased usage in many industries and the sports industry is no exception. With the ever-growing quantity of data being collected from modern sporting events, models capable of interpreting these results in a meaningful way can provide the coaches and athletes with valuable insights to help improve their performance. We believe that predicting an athlete's performance is an interesting topic in professional sports, specifically in road cycling.

The study aims to develop a talent identification method by building a model that assists coaches and scouts in predicting the performance of road cyclists effectively. Our model achieves this by classifying riders as either top 20 or not in the general rankings. In addition, we study the effectiveness of an alternate ranking system,PCS, to represent the performance in the official UCI ranking system. Our motivation for exploring another ranking system lies in the fact that the official UCI ranking system has undergone multiple changes over the years [1], leading to inconsistent data that may impact the accuracy of our model. While previous studies [2] [3] [4] have used either the PCS or UCI ranking system, there has been no analysis conducted on how effectively PCS can capture the UCI ranking system.

Lastly, our study aims to identify the key success factors for riders who achieve a top 20 ranking. We believe that discovering these factors will lead to a deeper understanding of the dynamics that separate elite riders from the rest.

II. Related Work

This section introduces knowledge that helped bring this project to completion. It discusses the relevant literature within the field of predicting performance in professional sports using machine learning, with an emphasis on similar research conducted in road cycling.

Sports where contestants are entered by a team and the contestants compete against members of other teams but also against each other for points towards championship standings can be classified as multi-contender sports. Studies on predicting performance have been researched in different sports such as horse racing [5], golf [6], marathon runners [7][Z] and swimming [8]. These studies used a number of different techniques such linear regression models [6],bagging regression trees [7],feature selection [6] and non-linear regression models [8].

Road cycling falls under the multi-contender category, but it also has other peculiarities. While the riders compete in teams, the winner of a race is an individual rider. It also offers a diverse selection of stage races ranging from flat surfaces to very steep mountain stages. Achieving peak performance in each of these stages requires different types of physiological capabilities. Atkison et al. study the physiological factors in road cycling [9] while Lucia et al. demonstrate the physiological difference through the BMI [10].

Van Bulck et al. [2] focus on talent identification through result-based analysis. Kholkine et al. [11] built a framework that predicted the result of the Tour of Flanders, a famous cobbled classic race. Following this, Kholkine et al. [4] built another framework for predicting the top 10 contenders for 1-day road cycling races, using the machine learning technique, LTR. Another study done by Karetnikov [3] focuses on predicting a rider's performance in mountain stage races. Schumacher et al. [12] studied whether a rider's success during an amateur career lead to a more likely top 10 placing in the elite professional races and concluded that for road cycling there is not a significant trend between success in junior and elite races. Mostaert et al. [13] studied the importance of performance in youth competitions as an indicator of future success in cycling.

These studies have shown us that accurate results can be achieved through machine learning techniques and publicly available data [5]. Selecting the right features can have a significant impact on prediction accuracy [6]. While PCS points were used for prediction in some of these studies [14] [11], the validity of this approach was not studied. Furthermore, we found an interesting gap in literature, that is the framing of performance prediction in road cycling as a classification challenge at the upper ranks. To address these gaps, we conduct some in-depth analysis of the ranking systems as well as build a specialized classification model.

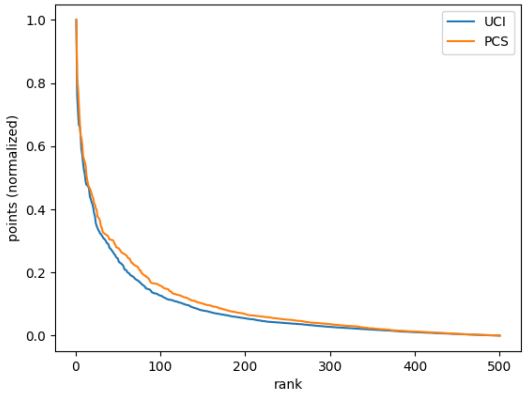
III. UCI AND PCS RANK ANALYSIS

In the pursuit of improving the ranking system, the UCI rules and ranking system have undergone a lot of changes [1]. These changes have resulted in data that is not consistent with our use case. In this section, we compare the two ranking systems on how well the PCS captures the UCI ranking system.

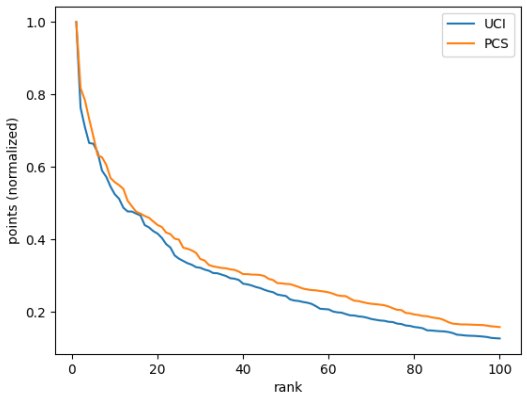
A. Comparative Graph Analysis

First, we plot the graph that compares the points obtained by the top-ranked riders for both PCS and UCI from 2016 till 2019. The points are normalized to ensure that the rankings systems are being compared on an equal footing. After normalization, the point distribution of each year is aggregated to a single point distribution curve by taking their median. The points are normalized using min-max feature scaling. This method is used to bring all values into the range [0,1]. We use the following formula for the PCS point normalization, where X' is the normalized point of the rider, and X_{min} is the lowest point among the 500 riders, X_{max} is the highest point among the 500 riders:

$$X_{sc} = \frac{X - X_{min}}{X_{max} - X_{min}}$$



(a)



(b)

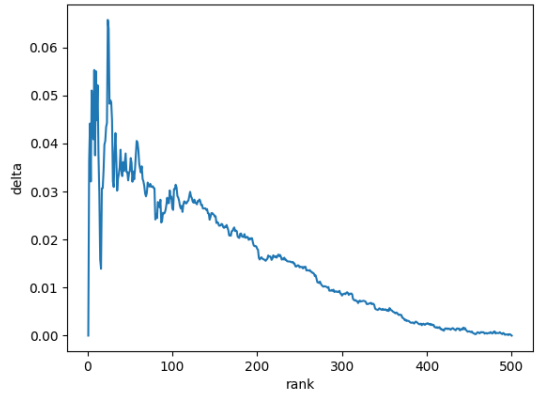
Fig. 1: Normalized point distribution curve of top 500 and top 100 riders for PCS and UCI aggregated by taking their median from 2016 till 2019.

The resulting graph in Figure 1a reveals a decaying curve where the top end between both ranking systems are quite similar but diverge at the lower ranks (top 20-300) before it converges again. Figure 1b gives a closer look at the point distribution curve at the top ranks. We observe that the difference between the two curves is minimal up until the top 20 ranks. We can hypothesize that predicting lower PCS ranks might have bigger inconsistencies with the UCI rank. However, the curve shows that the point distribution is nearly identical for the top 20 riders, which is our primary focus.

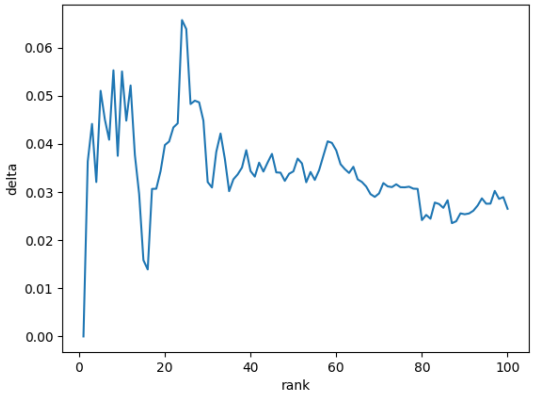
Next, we visualize the delta's(Δ's) between the top 500 riders of UCI and PCS in a graph. We portray delta (Δ) as the mathematical difference in points for each of the top 500 ranked riders between the PCS and UCI rankings. The delta is calculated for each year from 2016 till 2019. To capture the disparity between the two ranking systems during this time period, we aggregate the individual delta values by taking their median. We calculate the delta(Δ) for each rider at rank i as follows:

$$\Delta_i = |UCI_{pts,i} - PCS_{pts,i}|$$

In the resulting delta graph shown in Figure 2a, we observe that the delta fluctuates more in the upper rankings and gradually decreases as the rank decreases. This is reasonable because the scoring system awards more points to the upper ranks. As the competition at the top is much fiercer, this means that the difference in points between the very top-ranked riders in PCS and UCI is likely to be larger than the difference in points between the lower-ranked riders. A closer look through Figure 2b shows us that riders around rank 30-35 experience the greatest change. While there are fluctuations in the graph, the observed delta values indicate a relatively small difference.



(a)



(b)

Fig. 2: Normalized delta graph between UCI and PCS points accumulated by the top 500 and top 100 riders aggregated by taking their median from 2016 till 2019.

B. Correlation Analysis

We aim to understand how closely PCS and UCI rankings align among the elite riders. By exploring the correlation, we can discover whether the two ranking systems consistently recognize and reward the same set of top-ranked riders. We use Spearman's rank correlation [15] to measure the correlation between the ranking systems. It measures the strength and direction of association between two ranked variables. The following is the formula for Spearman's rank coefficient, where d_i is the difference between the two ranks of each observation and n is the number of observations:

$$p = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}$$

The results of the correlation graph can be found in Figure 3. We calculated the Spearman correlation between PCS and UCI for different thresholds of rankings: top 10, 20, 30 and 40 riders from 2016-2021.

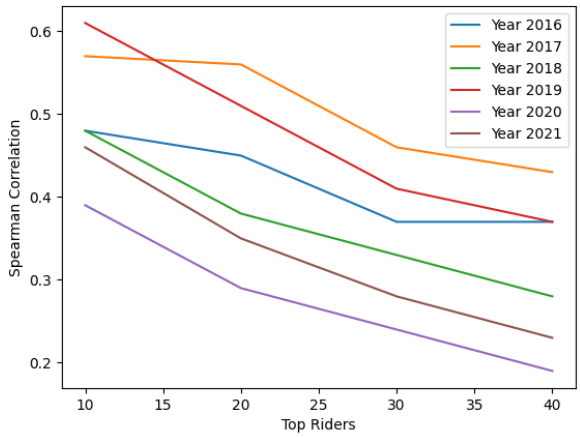


Fig. 3: Spearman Correlation curve between PCS and UCI for top 10/20/30/40 riders from 2016-2021. The correlation decreases with each increasing thresholds for the ranks and as such predicting lower PCS ranks will have larger inconsistencies with the UCI rank.

We observe that as the rank threshold increases, the correlation decreases, meaning that predicting lower PCS ranks will have larger inconsistencies with the UCI rank. However, for the top 10 and top 20 ranks, there is a moderately significant positive correlation. This means that predicting these top ranks is more consistent with the UCI rank. The outbreak of the COVID-19 pandemic in 2020 is the likely cause for the consistently lower correlation for that year.

C. Conclusion

We conclude that the analysis shows that the PCS ranking system can be used to predict the performance of the top 20 riders with a moderately significant positive correlation with the UCI rank. However, predicting lower PCS ranks might have larger inconsistencies with the UCI rank. Both PCS and UCI scoring systems award a lot more points to the upper rankings, which leads to larger differences in delta values for the upper rankings. We should also consider the impact of the COVID-19 pandemic on the cycling season when interpreting the results.

IV. Predicting Performance

This section is focused on the prediction performance of the riders. We create a model that predicts whether the rider will be in the top 20 of the PCS ranking.

A. Data

All the data was web scrapped from the PCS website. We extracted information about the rankings and PCS points of the top 500 riders from 1995 to 2022. For these top riders, we also extracted their rider profiles containing information on all the races they participated in. Additionally, information

on a race was extracted if at least one of the top 500 riders participated in that race.

B. Feature extraction

We grouped our features into different categories to help make the feature engineering process more organized. It also helps better understand which types of features are most important in predicting the target variable. Table I showcases all the features we extracted and the operations taken to obtain those features.

Performance features are features related to a rider's performance in races. These features measure the rider's success in finishing in the top positions in races they participate in. Each of these features is further extended into multiple features by separating them according to different operations such as the career average, the average and standard deviation in the last 3 years, the ratio of the current year and the race type (one day/stage races). The results of the last 3 years give us more information to more accurately judge the performance of the rider.

Race result features contain features that measure the performance of riders for specific races. These races all have different types of profiles and can be categorized into four types: flat races, hilly races, cobble classics and multi-day races. Race result features can help differentiate the performance of riders in different types of races.

The additional features include the rider's age and how long they have been active professionally. We believe these features can help indicate cases such as a young rider with a short career length who might have potential but not the time to fully develop their skills yet.

Features	Operations								
	Career avg	Ratio of last 3 years	Avg of last 3 years	Stdev of last 3 years	Career sum	Race format (one day/ stage races)	Previous year	Sum of last 3 years	Current Year
Performance features									
Number of participated races					X	X		X	X
Win ratio	X		X	X	X	X		X	X
Top 3 ratio	X		X	X	X	X		X	X
Top 5 ratio	X		X	X	X	X		X	X
Top 10 ratio	X		X	X	X	X		X	X
Win ratio slope		X				X			
Top 3 ratio slope		X				X			
Top 5 ratio slope		X				X			
Top 10 ratio slope		X				X			
PCS points			X	X			X		X
PCS rank			X	X			X		X
Race result features									
Flat one day races						X			X
Hilly one day races						X			X
Cobble classics						X			X
Multi day races						X			X
Additional features									
Rider age					X				X
Career year					X				X
Rider BMI					X				X

TABLE I: Feature table showcasing all the types of operation taken to get the different types of features.

C. Data Imputation: Altered KNN

We apply data imputation techniques to prevent our model from making unreliable predictions due to potential loss of insightful information. A widely used technique is the KNN imputation, but its limitation is that it needs complete cases to estimate the missing values [16]. However, our dataset does not have any complete cases since it is physically impossible for a rider to compete in all races.

Through a previous study [14], we have knowledge of an alternate way to perform KNN imputation. First of all, the features with missing values are split into groups based on domain knowledge. Then, we impute each group of these groups separately. Note that the remaining features that are not part of any of these groups form the complete observed set. During the actual modeling process, we find the best k using rolling cross-validation.

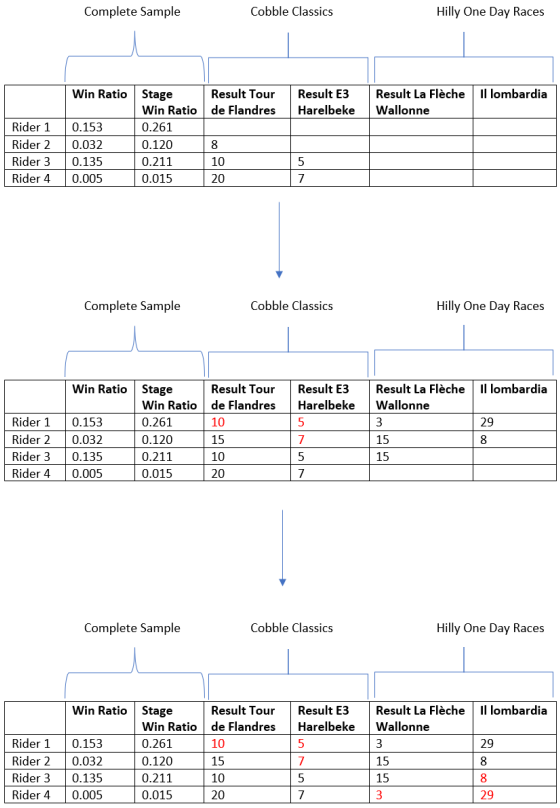


Fig. 4: Example of an alternate KNN imputation using imputation groups to split data to get complete samples.

A simplified version of our dataset with 4 samples and 6 features where we set the number of neighbors, K=1 can be seen in Figure 4. The features are divided into 3 parts: Complete samples, Cobble Classics and Hilly one-day races. The latter two are our imputation groups. The first table in Figure 4 is the dataset before imputation. To get complete samples we split the data by the imputation groups. This results in two groups: Complete samples + Cobble Classics and Complete samples + Hilly one day races. In the second

table in Figure 4, we see the KNN imputation implemented on the Complete samples + Cobble Classics group where riders 3 and 4 are considered complete cases. Rider 1's missing values get filled with the same value as that of rider 3 because of his similarity to rider 3 rather than to rider 4. Following the same process, rider 2 is more similar to rider 4 than rider 3 so he gets the same value as rider 4. The process is repeated in the last table in Figure 4 for the Cobble Classics + Hilly one-day races group. This time riders 1 and 2 are considered complete cases. Note that the results from the previous table are not considered when imputing. Once all groups have been imputed, the dataset is complete as seen in Figure 4.

D. Feature Selection

We have a total of 84 features and for a dataset of our size that only includes the top 500 riders from the years 1995 to 2022, this amount of features is quite large. To avoid falling prey to the curse of dimensionality [17] a feature selection method is proposed. We used SelectKBest which is a filter type method of feature selection. This method removes all but the k highest scoring features. The scoring function used is mutual information. It measures the dependency between the two variables and captures any kind of statistical dependency where higher values mean higher dependency.

E. Algorithms

There are 5 algorithms considered for our models, and each of them is briefly discussed below.

1) Logistic Regression: Logistic Regression is a classification technique that uses a logistic function to predict the dependent variable [18].

2) Random Forest: Random Forest is a supervised learning algorithm that uses an ensemble method [19].

3) XGBoost: XGBoost is a gradient boosting algorithm first introduced by Chen et al. [20]. In gradient boosting, multiple weak models, usually decision trees, are created and then combined to get better performance as a whole. Each iteration of the model learns from the previous iteration and reduces the error. XGBoost utilizes this boosting technique and made further improvements by introducing regularization to reduce overfitting.

4) Catboost: Catboost is a gradient boosting algorithm that uses ordered boosting, a variant of gradient boosting, to better handle categorical features [21].

5) LightGBM: LightGBM is a gradient boosting framework that uses decision trees for classification and regression tasks [22]. Unlike XGBoost or CatBoost, where the decision trees are grown with a depth-first approach, LightGBM builds its trees leaf-wise by building upon the leaf with the highest gain.

F. Hyperparameter Optimization

In this section, we explain the approach we took to optimize the hyperparameters of the algorithms.

1) Rolling cross-validation: In road cycling, riders compete with other riders for a whole season and it is immediately followed by the next season. Thus, we believe the data that we have to not be independent. To prevent information from the future leaking into our model, we use rolling cross-validation. The model is trained on past data and tested on future data.
The rolling cross-validation process is visualized in Figure 5 where green boxes signify the training period, yellow is the validation period and red is the testing period. For each hyperparameter combination for an algorithm, the model trains on a total of 11 different periods and for each period it validates for the next year, and tests on the year after that. The final evaluated metric is the average of all the metrics calculated in the different periods. The hyperparameters with the best metric result during the validation set are used for the testing set.

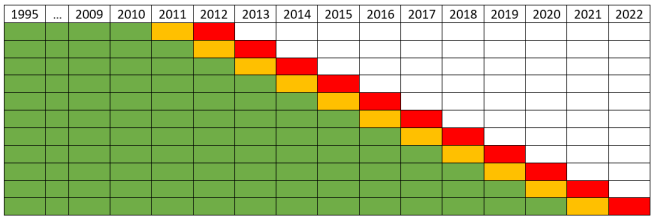


Fig. 5: Rolling cross-validation: each row is a different train/val/test period, green boxes represent the training period, yellow boxes represent validation period and red boxes represent testing period

2) Hyperparameter Tuning: The parameters that were tuned for each of the algorithms and their candidate values can be seen in Table II. Each of these parameters were carefully selected to prevent overfitting, underfitting and making sure the model does not take too long converge.

G. Evaluation Metrics

Our model predicts whether a rider will make it in the top 20 PCS ranking for next year and to evaluate this performance, we chose several evaluation metrics used for classification models. We also took into account the class imbalance that is present in our dataset and chose the following metrics for evaluation: Precision, Recall and F1-score.
Since we are interested in who is going to be in the top 20 riders, *Precision* is an important metric as it measures the proportion of positive predictions that are correct. It answers our question by telling us how many riders predicted to be in the top 20 PCS ranking are actually in the top 20. *Recall* is another evaluation metric that is interesting for our model. It measures the proportion of true positives that are correctly identified by the model and thus, tells us how many of the riders that made it into the top 20 are correctly predicted by our model. *F1-score* is used to compare the performance between models. It combines the precision and recall metrics by taking their harmonic mean. Since F1-score considers both false positives and false negatives, it is especially useful when the classes are imbalanced.

Algorithm	Hyperparameter	Candidate values
Logistic Regression	/	/
Random Forest	Maximum depth	[6,8,10,12]
	Number of trees	[100,200,300,400,500]
	Minimum sample split	[2,3,5]
XGBoost	Alpha	[0,4,8,12]
	Number of trees	[100,200,300,400]
	Max depth	[6,8,10,12]
	Learning rate	[0.1,0.3,0.5,0.7,0.9]
	Lambda	[1,3,5,7]
	Subsampling rate	[0.5,0.7,1.0]
	Feature Subsampling rate	[0.1,0.3,0.5,0.7,1.0]
	Gamma	[0,4,8,12]
CatBoost	Number of trees	[100,200,300,400]
	Max depth	[6,8,10,12]
	Learning rate	[0.1,0.3,0.5,0.7,0.9]
	Lambda	[1,3,5,7]
	Subsampling rate	[0.5,0.7,1.0]
	Feature Subsampling rate	[0.1,0.3,0.5,0.7,1.0]
	Minimum samples in leaf	[0,4,8,12]
LightGBM	Alpha	[0,4,8,12]
	Number of trees	[100,200,300,400]
	Max depth	[6,8,10,12]
	Learning rate	[0.1,0.3,0.5,0.7,0.9]
	Lambda	[1,3,5,7]
	Subsampling rate	[0.5,0.7,1.0]
	Feature Subsampling rate	[0.1,0.3,0.5,0.7,1.0]
	Gamma	[0,4,8,12]
	Num leaves	[10,20,30,40,50]
	Min data in leaf	[5,10,15,20]
NGBoost	Number of trees	[100,200,300,400,500]
	Learning rate	[0.1,0.3,0.5,0.7,0.9]
	Feature Subsampling rate	[0.1,0.3,0.5,0.7,1.0]

TABLE II: This table lists the considered hyperparameters and their candidate values for each algorithm.

H. Results

This section shows the results for all our models that were evaluated using the metrics that we discussed in Section IV-G. We compare the models on each of the metrics and discuss which model performed the best.

1) Precision: There are 11 models for each algorithm and the average across all the periods is used for evaluation. In addition, the boosting algorithms have built-in methods to handle missing values without just ignoring them. These variants of the boosting algorithms were also trained and tested alongside our method of data imputation through an alternate KNN method discussed in Section IV-C. Including all the variants of the algorithms, we end up with 8 algorithms with 11 models each, so we have 88 models in total. Table III shows the average precision values for our models where for each column we set the value for the overall best performing model in bold.

We observe that the boosting algorithms outperformed the other algorithms and the performance between the boosting algorithms is also comparable. Our base model, the Logistic Regression performed the worst. Logistic Regression works well when the relationship between the independent variables and dependent variable is linear but if it is non-linear then the performance suffers. Another thing to note is the performance of the models through an altered KNN (Section IV-C) performed better than the default way the algorithms handle missing values. Overall, we saw about a 2-4% increase in precision which we consider significant because even small differences can have a large impact on the outcomes. Overall, the performances of the boosting models were similar across all the sets. XGBoost with Altered KNN performed the best for the training set and validation set, while CatBoost with Altered KNN performed the best for the test set.

2) Recall: The average recall values for our models can be seen in Table IV where for each column we set the value for the overall best performing model in bold. We observe that the boosting algorithms outperformed Logistic Regression and Random Forest for the recall metric as well. This might be because boosting algorithms generally handle feature interactions and class imbalance problems better than Logistic Regression and Random Forest. Altered KNN data imputation also performs better than the standard way these algorithms deal with missing data. We observe that so far this increased performance is consistent across both precision and recall metrics. The best performing models were also the same as the ones for precision. XGBoost with Altered KNN performed the best for the training set and validation set, while CatBoost with Altered KNN performed the best for the test set.

3) F1-score: The average F1-score values for our models can be see in Table V where for each column we set the value for the overall best performing model in bold. The observations for the results of F1-scores are similar to the observations we made for the results of precision and recall metrics. First, the boosting algorithms gave the best results. Second, the models that used KNN imputation performed better than others. We have now observed that this method has consistently improved the performance of the models across all evaluated metrics. Third, the performance difference between the boosting algorithms is relatively small and Logistic Regression is consistently the worst performing model. For the training set and validation set, XGBoost with Altered KNN performed the best, while CatBoost with Altered KNN performed the best for the test set.

Model	Precision Train	Precision Validation	Precision Test
Logistic Regression	0.88117	0.50482	0.48110
Random Forest	0.94135	0.71956	0.64152
Random Forest with Altered KNN	0.99575	0.76887	0.67966
XGBoost	0.95782	0.86013	0.76993
XGBoost with Altered KNN	**0.99945**	**0.88469**	0.78403
CatBoost	0.97996	0.80177	0.75274
CatBoost with Altered KNN	0.99549	0.82894	**0.78468**
LightGBM	0.96367	0.81473	0.70961
LightGBM with Altered KNN	0.99249	0.84810	0.74976

TABLE III: Aggregated precision results of the models for training, validation and testing. The best scoring models are highlighted in bold for the train, validation and test set.

Model	Recall Train	Recall Validation	Recall Test
Logistic Regression	0.88091	0.50169	0.47392
Random Forest	0.93991	0.71666	0.63950
Random Forest with Altered KNN	0.99198	0.75879	0.67511
XGBoost	0.95606	0.85812	0.76975
XGBoost with Altered KNN	**0.99892**	**0.88278**	0.78017
CatBoost	0.97985	0.80067	0.75144
CatBoost with Altered KNN	0.99526	0.81995	**0.78290**
LightGBM	0.95749	0.81357	0.70823
LightGBM with Altered KNN	0.99119	0.84759	0.74775

TABLE IV: Aggregated recall results of the models for training, validation and testing. The best scoring models are highlighted in bold for the train, validation and test set.

Model	F1-score Train	F1-score Validation	F1-score Test
Logistic Regression	0.88104	0.50325	0.47748
Random Forest	0.94063	0.71811	0.64051
Random Forest with Altered KNN	0.99386	0.76380	0.67737
XGBoost	0.95694	0.85912	0.76984
XGBoost with Altered KNN	**0.99918**	**0.88373**	0.78209
CatBoost	0.979905	0.801218	0.752084
CatBoost with Altered KNN	0.995375	0.824235	**0.78378**
LightGBM	0.96874	0.81415	0.70892
LightGBM with Altered KNN	0.99184	0.84784	0.74875

TABLE V: Aggregated F1-score results of the models for training, validation and testing. The best scoring models are highlighted in bold for the train, validation and test set.

I. Model Discussion

We made the observation that the Altered KNN imputation method combined with the boosting algorithms gave us the best results across all evaluated metrics. When comparing the precision and recall metric, we discovered that the models score better albeit very slightly on precision. While it would be ideal to have a higher recall, as false negatives are more concerning for our use case. A false negative means that a rider who was supposed to be in the top 20 was not predicted as such by our model. In a real-world scenario, it would mean our model missed the mark and might have negatively affected the rider's career. While we know the importance of this, we believe that the precision and recall scores are close enough (0.001% difference) that the impact of this difference on the model is negligible.

Next, we aim to understand how our model is making predictions. We try to accomplish this by identifying the features which that had the largest impact on the outcomes of our models. We chose the best performing model, XGBoost with altered KNN, to showcase the feature importance. We used the SHAP method invented by Lundberg et al. [23] to quantify the importance of each feature. The SHAP values represent the contribution of each feature to the outcome of a prediction.

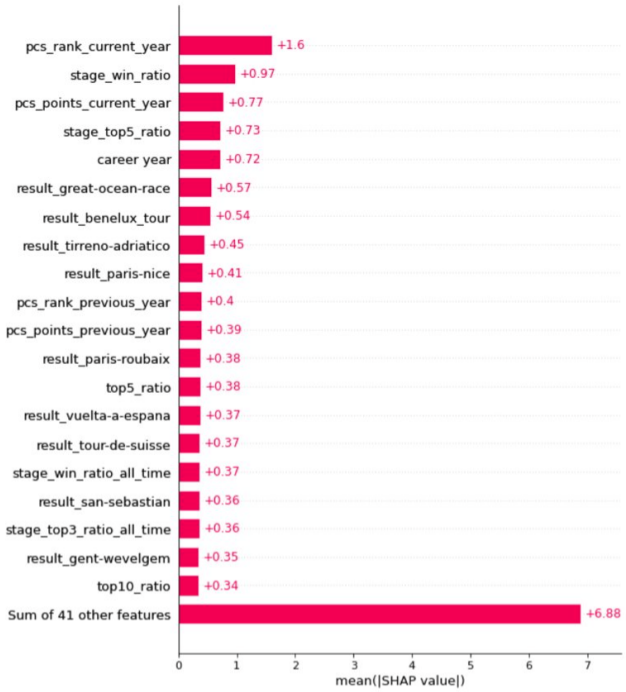


Fig. 6: SHAP values for the XGBoost model.

The feature importance for our XGBoost model is shown in Figure 6. We see that the *pcs_rank_current_year* is the largest contributor. This is reasonable as we are trying to predict whether a rider will make the top 20 for the general ranking which includes all the races, and our model sees features such as the most recent rank and points of a rider as a good indicator of a rider's current strength. We also observe that *stage_win_ratio* is the second largest contributor and *stage_top5_ratio* is among the top 5 contributors. This indicates that riders who consistently finish high on stage races are more likely to make it into the top 20 rankings. We also observe that consistency is a key factor as showcased by features such as *stage_win_ratio* and *stage_top5_ratio* having high mean SHAP values.

We also see some races that have a high feature importance. Some of these races such as Paris-Roubaix and Vuelta a España are very prestigious races in road cycling. Benelux Tour also showed a fairly high contribution. As it is a relatively new race added to the UCI World Tour calendar, we decided to analyze the credibility of this feature's contribution. Analysing our dataset before imputation, we found that Benelux Tour had the most amount of missing values. It's possible that the imputation method accurately captured the relationships in our dataset which led to Benelux Tour contributing useful information. However, it's also possible that the imputation process overlapped different features that already provide similar information. By conducting a sensitivity analysis, where we removed the Benelux Tour feature, we found our test results to be similar to before the removal of this feature. This leads us to believe that even though Benelux Tour had a high SHAP

value after imputation, its impact on the model's performance might be limited. We can also deduce that our model is not heavily reliant on one specific feature and the predictions are based on a combination of multiple factors.

Coaches can refer to these races, but not exclusively, as a guideline to help them figure out who is more likely to make it into the top 20 in the future. Also, the high contribution of *career_year* means that our model emphasizes the fact that the length of the rider's career has a significant impact on the chances of them making it to the top 20. Thus, we believe that the length of a rider's career can be indicative of his experience and ability to endure and compete at a high level which can be important in predicting future success.

V. Conclusion

One of the goals of this study was to determine how well the PCS ranking system represented the UCI ranking system. Through our analysis, we concluded that the PCS ranking system could be used to predict the performance of the top 20 riders with a moderately significant positive correlation with the UCI rank but predicting lower PCS ranks might have larger inconsistencies with the UCI rank. The effect of COVID-19 was also visible in the 2020 season.

Another goal of this study was to build a model that can predict the performance of professional road cyclists by classifying them into whether they will be top 20 riders or not. This ranking was not limited to certain types of races but rather the overall general rankings. A key finding was the improvement provided by the data imputation technique. The precision was slightly higher than the recall for all of our models. We believe a higher recall to be more desirable because misclassifying a top 20 rider is a bigger mistake than misclassifying a not-top 20 rider. However, we believe that on both metrics our model provided similar scores (0.001% difference) and that the impact of this difference on the model is negligible. A feature importance performed on our XGBoost model through the SHAP method showed that a rider's current rank and points have the largest impact on the model's prediction but also consistently winning stage races was a large contributor in a rider making it to the top 20 rankings. The length of a rider's career was another factor that had a significant impact on the model's outcome. We believe it to be indicative of his experience and ability to endure and compete at a high level which can be important in predicting future success.

Overall, our study has shown the model we built to be able to predict rider performance to a reasonable level while also highlighting the factors that are important to a rider's future success. We believe that our model can be used as a helpful guideline by coaches and riders alike. It can be used alongside their expert knowledge to help make better choices.

VI. Future Work

There are still many aspects of road cycling that can be researched which may also lead to the improvement of our models. We believe the effect of the COVID-19 pandemic especially in the 2020 and 2021 seasons to be significant. It could be interesting to investigate the impact it had on the performance of riders in more detail. An approach might be to do a separate analysis of COVID-19's effect on the rider's performance and use the results to adjust the predictions made by the model.

Another study could be done on role players in road cycling like the *domestiques* who have an important role in the team but their performance cannot be captured in the rankings. Future works could try to implement a model that can capture how good a domestique is.

A shortcoming in our model that could be improved in future studies is the addition of physiological features. These features could be useful in improving the performance of our model granted that such data be publicly available on a large scale.

Acknowledgment

The author would like to thank Prof. Dr. Steven Latré and MSc. Leonid Kholkine for their assistance in this research.

References

[1] D. V. Reeth and D. J. Larson, "The economics of professional road cycling," *The Economics of Professional Road Cycling*, 2022.

[2] D. V. Bulck, A. V. Weghe, and D. R. Goossens, "Result-based talent identification in road cycling: discovering the next eddy merckx," *Annals of Operations Research*, pp. 1 – 18, 2021.

[3] A. D. Karetnikov, "Application of data-driven analytics on sport data from a professional bicycle racing team," 2019.

[4] L. Kholkine, T. Servotte, A.-W. de Leeuw, T. D. Schepper, P. Hellinckx, T. Verdonck, and S. Latré, "A learn-to-rank approach for predicting road cycling race outcomes," *Frontiers in Sports and Active Living*, vol. 3, 2021.

[5] E. Davoodi and A. R. Khanteymoori, "Horse racing prediction using artificial neural networks," 2010.

[6] O. Wiseman, "Using machine learning to predict the winning score of professional golf events on the pga tour," 2016.

[7] D. Ruiz-Mayo, E. Pulido, and G. Martınoz, "Marathon performance prediction of amateur runners based on training session data," *Proc. Mach. Learn. and Data Min. for Sports Anal*, 2016.

[8] A. Maszczyk, R. Roczniok, M. Czuba, A. Zajαc, Z. Waśkiewicz, K. Mikołajec, and A. Stanula, "Application of regression and neural models to predict competitive swimming performance," *Perceptual and Motor Skills*, vol. 114, pp. 610 – 626, 2012.

[9] G. Atkinson, R. C. R. Davison, A. E. Jeukendrup, and L. Passfield, "Science and cycling: current knowledge and future directions for research," *Journal of Sports Sciences*, vol. 21, pp. 767 – 787, 2003.

[10] A. Lucia, C. P. Earnest, and C. Arribas, "The tour de france: a physiological review," *Scandinavian Journal of Medicine & Science in Sports*, vol. 13, 2003.

[11] L. Kholkine, T. D. Schepper, T. Verdonck, and S. Latré, "A machine learning approach for road cycling race performance prediction," in *MLSA@PKDD/ECML*, 2020.

[12] Y. O. Schumacher, R. Mroz, P. Mueller, A. Schmid, and G. Ruecker, "Success in elite cycling: A prospective and retrospective analysis of race results," *Journal of Sports Sciences*, vol. 24, pp. 1149 – 1156, 2006.

[13] M. Mostaert, P. Vansteenkiste, J. Pion, F. J. A. Deconinck, and M. Lenoir, "The importance of performance in youth competitions as an indicator of future success in cycling," *European Journal of Sport Science*, vol. 22, pp. 481 – 490, 2021.

[14] B. Janssens, M. Bogaert, and M. Maton, "Predicting the next pogaar: a data analytical approach to detect young professional cycling talents," *Annals of Operations Research*, pp. 1 – 32, 2022.

[15] J. Hauke and T. M. Kossowski, "Comparison of values of pearson's and spearman's correlation coefficients on the same sets of data," 2011. [Online]. Available: https://api.semanticscholar.org/CorpusID:5311856

[16] A. S. Jadhav, D. Pramod, and K. Ramanathan, "Comparison of performance of data imputation methods for numeric dataset," *Applied Artificial Intelligence*, vol. 33, pp. 913 – 933, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:198351111
[17] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, 2003.
[18] D. W. Hosmer and S. Lemeshow, "Applied logistic regression," 1989.
[19] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2004.
[20] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.
[21] A. V. Dorogush, V. Ershov, and A. Gulin, "Catboost: gradient boosting with categorical features support," *ArXiv*, vol. abs/1810.11363, 2018.
[22] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *NIPS*, 2017.
[23] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *ArXiv*, vol. abs/1705.07874, 2017.

Chapter 1

Introduction

1.1 Problem statement

In recent years, data analytics has seen increased usage in many industries and the sports industry is no exception. With the ever-growing quantity of data being collected from modern sporting events, models capable of interpreting these results in a meaningful way can provide the coaches and athletes with valuable insights to help improve their performance. We believe that predicting an athlete's performance is an interesting topic in professional sports, specifically in road cycling.

Road cycling is a sport with many peculiarities. Firstly, it is a multi-contender sport, unlike most sports that are two-contender based where only two teams compete against each other at any given time. In road cycling, multiple teams compete against each other, and the winner of a race is an individual rider. Furthermore, unlike numerous other sports, road cycling races take place across diverse routes that vary significantly in curves, climbs, and distances. These variations may even occur between different editions of the same race. Additionally, there are different types of races with different

goals. One-day races, as the name implies, are settled within a day, with one finish line and every rider for themselves. Stage races, on the other hand, take place over several days to weeks even, featuring several sections with differing roads like flat or hilly terrain.

Given these unique attributes of road cycling, we believe it would be interesting to see if a model could be built that can interpret and analyse the data, and give valuable insights. This is especially significant since the abundant amount of data makes manually reviewing them an overwhelming job for coaches and scouts.

1.2 Goal

The goal of this master's thesis is to develop a talent identification method in road cycling. We aim to develop a model that assists coaches and scouts in predicting the performance of road cyclists effectively. Our model achieves this by classifying riders as either top 20 or not in the general rankings.

In addition to our classification model, one of our goals is to determine whether an alternate ranking system can represent the performance in the official UCI ranking system effectively. Our motivation for exploring another ranking system lies in the fact that the official UCI ranking system has undergone multiple changes over the years[1], leading to inconsistent data that may impact the accuracy of our model. We conduct an insightful analysis of an alternate ranking system known as PCS, which is employed by the cycling database website *www.procyclingstats.com*. By evaluating how effectively PCS captures the UCI ranking system, we aim to determine whether the PCS ranking system can serve as a viable alternative during our model development process. Furthermore, while previous studies[2][3][4] have used either the PCS or UCI ranking system, there has been no analysis conducted on how effectively PCS can capture the UCI ranking system.

To construct our classification model, we rely on publicly available data obtained by web scraping from the PCS database website. We utilize several

machine learning algorithms and techniques to build our model. We chose a classification approach because we believe that predicting the explicit ranking of riders relies on too many variables for which we lack sufficient data. Furthermore, the difference between a rider's performance in the top rankings can be quite subtle. Therefore, categorizing riders based on their inclusion in the top 20 rankings provides a more practical approach that can still offer valuable insights.

Lastly, our study aims to identify the key success factors for riders who achieve a top 20 ranking. We believe that discovering these factors will lead to a deeper understanding of the dynamics that separate elite riders from the rest.

1.3 Outline

The thesis is structured as follows:

- Chapter 2 contains an overview of related works. We discuss various topics such as the uniqueness of road cycling, performance prediction done in other multi-team sports, two team sports, and road cycling.
- Chapter 3 compares two ranking systems used in professional road cycling. We discuss the official ranking system used in professional road cycling along with its inconsistencies. It presents an alternative ranking system, compares the two systems, and shows the alternative can be used over the other.
- Chapter 4 focuses on performance prediction. We showcase how we built our model and the results of our experiments. We discuss the interpretability of our model by showing the features that contributed most to our model's decision making.
- Chapter 5 summarizes the results of our project and draws conclusions and in Chapter 6, we present the shortcomings of our study and provide some possible future works.

Chapter 2

Related works

This section will focus on the relevant literature within the field of predicting performance in professional sports using machine learning. The aim is to introduce knowledge that helped bring this project to completion. The chapter discusses the current advances in literature, with an emphasis on similar research conducted in road cycling and other sports.

The first section of this chapter centers on works that feature performance prediction in sports other than road cycling. Subsequently, we explain road cycling's uniqueness when compared to most other sports and focus on the relevant works related to road cycling.

2.1 Performance prediction in sports

This section highlights some of the contributions made regarding the prediction of results or player performance in sports other than road cycling.

2.1.1 Two-contender sports

Sports in which exactly two teams or individuals compete against each other can be classified as two-contender sports. There have been plenty of studies

conducted on predicting results for two-contender sports such as football, rugby, and tennis. While road cycling does not fall under this category, the research in this area is nonetheless insightful and helped better understand the various ways of modeling in performance prediction tasks.

Lehman et al.[5] predicted the NFL Potential of players based on their college career by drawing a performance curve. The model is a player rating system based on the ELO rating system. It evaluates player performance at the game level and tracks that performance throughout a player's college career. Berrar et al. [6] constructed a model that won them the 2017 Soccer Prediction Challenge. Their model used the k-nearest neighbour algorithm, recency feature extraction, and rating feature learning for constructing predictive features from soccer match data.

Zimmermann et al.[7] applied machine learning techniques to predict college basketball matches. They concluded that machine learning techniques have an upper limit to their predictive accuracy of about 74% and selecting the right features can be the difference between failure and success. Some other works that helped to get a deeper understanding of the subject include a player ranking machine learning model for soccer by Pappalardo et al.[8], another study [9] focused on the application of Artificial Neural Networks and k-Nearest Neighbour classification models in the scouting of high-performance archers.

Reviewing the literature on two-contender sports offered us valuable insights into the prediction models. It also made apparent that the direct applicability of these models intended for two-contender sports might be limited for the complex nature of road cycling sport. Nonetheless, techniques such as k-Nearest Neighbour algorithm and feature selection are important benchmarks that guided us in the development of our road cycling performance prediction model.

2.1.2 Performance prediction in Multi-contender sports

Another type of team sport that involves more than two teams can be classified as a multi-contender sport. This category includes sports such as golf, horse racing, swimming, and motorsports. Road cycling also falls under this category. These types of sports have different objectives compared to traditional team sports. In multi-contender sports, where contestants are entered by a team, the contestants compete against members of other teams but also against each other for points towards championship standings.

In 2010, Davoodi et al.[10] predicted horse racing outcomes using Artificial Neural Networks (ANN). They used publicly available datasets that contained a limited number of features. However, they still achieved an accuracy of 77% and showed us that accurate results can be achieved through limited amount of publicly available data. Wiseman et al.[11] used machine learning techniques to predict the score of professional golf events. Their best-performing algorithms were the "Bayesian Linear Regression" and "Linear Regression". The features were selected based on domain knowledge and statistical analysis. They concluded that feature selection is the key to ensuring accurate predictions. In 2016, Ruiz-Mayo et al.[12] predicted the performance of amateur marathon runners. They used bagging regression trees which they argue is best suited to deal with the unpredictability of a race such as a marathon. The training data was obtained from a public dataset from which they extracted features such as the distance covered and pace. They compared their regression tree model with the linear regression model and found that their bagging method provided the best results. Maszczyk et al. [13] used non-linear regression and neural networks to predict performance of athletes in competitive swimming. They used Absolute Error as a measure of evaluation for their models. Upon comparison, they found that the neural networks outperformed the non-linear regression. A difference from previously mentioned works is their use of personal data. They used features such as lung capacity and strength which are not publicly available. A downside of using this private data is the impact it has on the general applicability of the model since it is hard to model the performance of athletes

who do not want to share personal data.

By reviewing predictive modeling techniques in multi-contender sports, we gained insights on potential strategies for our road cycling performance prediction model. Davoodi et al.[10] demonstrated that accurate results can be achieved with a limited amount of publicly available data. From Wiseman et al.[11], we learned the significant impact that feature selection can have on prediction accuracy and Ruiz-Mayo et al.[12] showed us the effectiveness of bagging regression trees to tackle the unpredictability of races. We use the insights gained from these studies and tackle the unique nature of road cycling to create a specialized classification model for predicting the top 20 rankings.

2.2 Performance prediction in road cycling

Performance in road cycling is a relatively less researched subject compared to other sports because of its uniqueness. Like swimming and horse racing, road cycling falls under the category of multi-contender sports. However, road cycling is different because while the riders compete in teams, the winner of a race is an individual rider. The team will work towards assisting a specialist on their team, most suited for the race section, to the highest possible finishing position[14]. This makes road cycling a much more strategic sport compared to other multi-contender sports such as swimming or horse racing. The riders that focus on assisting are called *domestiques*. While the *domestiques* have an important role of protecting the team leader, they end up spending a large amount of their energy in doing so and their individual standings suffer as a result. This makes it hard to assess the strength of the *domestiques* with just the end results.

Another particularity of road cycling is the diverse selection of stage races. Stages can range from flat surfaces to very steep mountain stages. Achieving peak performance in each of these stages requires different types of physiological capabilities. Atkison et al. study the physiological factors in road cycling[15] while Lucia et al. demonstrate the physiological difference

through the Body Mass Index (BMI)[16].

These unique attributes of road cycling make performance prediction a challenging and nuanced task. In the next section, we explore the existing literature on machine-learning approaches for road cycling performance prediction. By understanding the uniqueness of this sport, we aim to fill in the gaps in the existing predictive techniques.

2.2.1 Prediction of road cycling performance

Several works investigate the use of machine learning algorithms to predict road cycling performance. For example, Van Bulck et al. [2] focus on talent identification through result-based analysis. Their models focus on discovering talented Under-23 (U23) riders by employing linear regression and random forest techniques and designating UCI points as the dependent variable.

Kataoka et al.[17] built a model that predicted the power output of a cyclist. For training, they were provided with the labeled power output by the professional cycling teams. They tested two types of regression methods: tree-based (Random Forest and XGBoost) and time-series neural networks (Long short-term memory (LSTM) and Gated Recurrent Unit (GRU)). The tree-based models outperformed the neural networks and XGBoost achieved the best results.

Kholkine et al.[18] built a framework that predicted the result of the Tour of Flanders, a famous cobbled classic race. A variety of features are taken into consideration such as a rider's past performance in similar races, their overall performance in previous years, and their career points. They use data available from the PCS website. They used the XGBoost algorithm and evaluated their model by comparing their results to the results of an online poll. This poll was voted on by cycling fans on who they thought would be the top 10 riders in the race. The model outperformed the human predictions for the 2018 edition of the race, while it had a similar performance for the 2019 edition. Following this, Kholkine et al. [4] built another framework for

predicting the top 10 contenders for 1-day road cycling races, this time using the machine learning technique, Learn-to-Rank (LTR). Again their model performed better than the fan predictions for both of the evaluated metrics, namely Normalized Discounted Cumulative Gain (NDCG) and number of correct top 10 guesses. Another interesting part of this is that they analyzed which variables had the most influence on the prediction of the race. Their model favoured consistent riders and while it could predict consistent riders, the success rate went down when predicting fast-rising stars.

Another study done by Karetnikov [3] focuses on predicting a rider's performance in mountain stage races. Like the aforementioned work, the author also used the data available on the PCS website as well as some private training data. Schumacher et al.[19] studied whether a rider's success during an amateur career lead to a more likely top 10 placing in the elite professional races and concluded that for road cycling there is not a significant trend between success in junior and elite races.

Several other interesting works have been published on the topic of road cycling itself. Mostaert et al. [20] studied the importance of performance in youth competitions as an indicator of future success in cycling. The study showed that the competitive success rate of U15 cyclists could not predict success at adult age. Atkinson et al.[15] provide knowledge on the factors that impact cycling power output. Mignot J.[1] provides an in-depth introduction to the history of professional road cycling while Rebeggiani L.[1] explains the organizational structure of professional road cycling. Cintia et al.[21] analyse a dataset of 29,284 cyclists' training sessions, to understand what factors distinguish successful athletes. They first observed the relation between the *training effort*, how much a cyclist struggled during the workout, and *training performance*, the results achieved during the training. They found a weak correlation between the two, meaning that greater effort didn't always produce better results. Upon further analysis where they clustered cyclists according to their effort and performance, they discovered that riders that followed a training routine with the appropriate amount of rest performed much better than those that didn't stick to a routine.

While these studies have explored and made many insightful discoveries, there remains a notable gap in the field of predicting road cycling performance. One such gap is the lack of critical analysis on the validity of using the PCS ranking system as an alternative to the official UCI ranking system. While some prior works[22][18] have used PCS as an alternative to UCI, a critical analysis on its validity is still absent. Another gap lies in the untapped potential of framing the prediction performance in road cycling as a classification problem in identifying elite riders.

2.3 Conclusion

The field of road cycling has many unique attributes such as team dynamics, individual efforts, and varied race profiles that make predicting performance a complex and difficult task. Previous works have shown that accurate results can be achieved through machine learning techniques and publicly available data. While PCS points were used for prediction in some of these studies, the validity of this approach was not studied. Furthermore, we found an interesting gap in literature, that is the framing of performance prediction in road cycling as a classification challenge at the upper ranks. To address these gaps, we conduct some in-depth analysis of the ranking systems as well as build a specialized classification model. In the next chapters, we dive deeper into our methodologies as we aim to bridge these gaps in the literature and provide valuable insights into predicting road cycling performance.

Chapter 3

UCI and PCS rank analysis

Previous works[22][18] on road cycling have used PCS data instead of UCI because of the constant changes in the way the UCI ranking system works, making UCI data inconsistent. While there have also been studies[2] [23] where they preferred to work with UCI despite its flaws because it is the official ranking. However, there have not been any concrete studies done on how well PCS points can represent the performance in the official UCI ranking. One of our goals is to predict the performance of our riders and to do that we aim to predict the top riders. If we want to use the PCS ranking system we need to know how well it represents the official UCI ranking system. In the chapter, we compare the two ranking systems on how well the PCS captures the UCI ranking system.

3.1 UCI and PCS ranking

Professional road cycling has a long history that dates to the late nineteenth century. To put this into perspective in terms of other sports, football and basketball became professional sport around the same period while other sports like rugby and sports event like the Olympic games remained amateur until the late 1980s.

In the beginning, there was the issue of ranking riders and teams. Tour-

nament organizers needed to invite the world's best teams as such there was a need for a singular system of ranking for riders and teams [1]. Thus, the UCI organization was formed to be the governing body to regulate, administer, and promote professional road cycling. With it came the formation of the official ranking system, called the UCI ranking. This ranking is determined based on the season-long results of the riders and teams.

In the pursuit of improving the ranking system, the UCI rules and ranking system have undergone a lot of changes [1] as listed in table 3.1. These changes have resulted in data that is not consistent with our use case. Data consistency is necessary for machine learning because prediction accuracy relies upon the quality and quantity of the data used for training the model. Inconsistent data leads the model to learn to make incorrect predictions which affects its performance. Furthermore, the data should be consistent over a long period, so that the model can identify long-term patterns and trends accurately.

Year	Rule changes
1948	Introduction of the Challenge Desgrange Colombo (until 1958)
1958	Introduction of the Super Prestige Pernod Trophy (until 1988)
1984	Introduction of the UCI Road World Rankings (until 2004)
1989	Introduction of the UCI Road World Cup (until 2004)
2005	The UCI ProTour replaces both World Rankings and World Cup
2008	Persistent issues between race organizers and UCI lead to the exit of races organized by ASO, RCS, and UniPublic from the ProTour
2009-2010	UCI ProTour and Historical Calendar exist in parallel. The UCI World Ranking provides an overall classification considering both
2011	Merger of the UCI ProTour and the Historical Calendar to the UCI WorldTour

Table 3.1: *The evolution of the series of cycling events. It highlights the changes in UCI ranking system throughout the years.*

A popular alternative to the UCI ranking is called the PCS ranking. PCS is a large database website created in 2015, that keeps track of cycling statistics, race results, PCS and UCI rankings, starting lists, and rider profiles. It has its own ranking system that has remained consistent through the years.

In both UCI and PCS rankings, the way riders earn points is similar. Riders get points according to the position they finish the race at. The key differences between the two ranking systems lie in the distribution of these points. The UCI ranking gives points only to the top 60 riders that finish the race while the PCS ranking distributes points to all riders that finish the race. In both systems, riders can also gain points by winning stage races but the amount gained differs in both systems as shown in Table 3.2.

Rank	**UCI points**	**PCS points**
1	180	80
2	130	50
3	95	35
4	80	25
5	60	18
6	45	15
7	40	12
8	35	10
9	30	8
10	25	6
11	20	5
12	15	4
13	10	3
14	5	2
15	2	1

Table 3.2: *Point distribution for a stage section of the 2023 edition of Giro d'Italia. UCI awards different number of points to each position compared to PCS ranking system.*

The evolution of the UCI ranking system and the consistency of the PCS ranking system lays the groundwork for us to transition to the methodology section. We will discuss the methods used to understand how well PCS can represent the UCI ranking system, which will bring us a step closer to a robust performance prediction model.

3.2 Methodology

While some works[22][18] assume that PCS ranking can be used to determine UCI's success, no actual analysis has been done yet to prove this point. In this section, we discuss the methodologies used to compare the two ranking systems on how well the PCS captures the UCI ranking system. Our analysis consists of two main methods, which are as follows:

- Comparative Graph Analysis: We visualize the point distribution curves for top-ranked riders as well as race point distributions in both PCS and UCI rankings.
- Correlation Analysis: We use rank correlation to assess the degree of similarity and consistency between PCS and UCI rankings.

3.2.1 Comparative Graph Analysis

We analyze the distribution of points between the two ranking systems to assess their compatibility. By understanding the point distribution, we can discover differences and similarities that could indicate how well PCS captures the UCI ranking system. To give us insight into the various aspects of the ranking systems, we employ different graph analyses such as point distribution graphs for top-ranked riders, visualizing the delta (difference in points of the top riders between the two systems) and race point allocation for one-day race and multi-day races between PCS and UCI.

Point distribution graph between top riders

For the point distribution curve, we plot the graph that compares the points obtained by the top-ranked riders for both PCS and UCI from 2016 till 2019. The points are normalized to ensure that the rankings systems are being compared on an equal footing. Through normalizing, we take into account the differences in the total number of points assigned by each system and adjust it so that the graphs can be directly compared.

The points are normalized using min-max feature scaling. This method is used to bring all values into the range [0,1]. This is also called unity-based normalization. We use the following formula for the PCS point normalization, where X' is the normalized point of the rider, and X_{min} is the lowest point among the 500 riders, X_{max} is the highest point among the 500 riders:

$$X_{sc} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

After normalization, we aggregate the point distribution of each year to a single point distribution curve by taking their median. If there are any systematic differences between the two ranking systems, it can be identified through the graph. For example, if the UCI ranking system consistently assigns more points to higher-ranked riders than the PCS counterpart, it may indicate that the two systems are not comparable.

Delta graph

We the visualize the delta's(Δ's) between the top 500 riders of UCI and PCS in a graph. We portray delta (Δ) as the mathematical difference in points for each of the top 500 ranked riders between the PCS and UCI rankings. The delta is calculated for each year from 2016 till 2019. To capture the disparity between the two ranking systems during this time period, we aggregate the individual delta values by taking their median. This delta graph helps us visualize which rank has experienced the greatest change in points between the two systems. We calculate the delta(Δ) for each rider at rank i as follows:

$$\Delta_i = |UCI_{pts,i} - PCS_{pts,i}|$$

Race point allocation graph

We take a look at how points are allocated for a one-day race and how they differ between PCS and UCI. We repeat the process for a multi-day race's general classification section. The race point allocation curve enables us to

evaluate how each ranking system assigns points to different types of races. It also helps us identify if a ranking system favors a certain type of race over others.

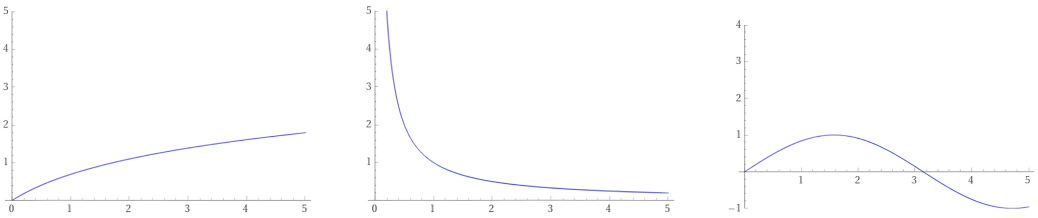
3.2.2 Correlation Analysis

In this section, we examine the compatibility between PCS and UCI ranking systems through a correlation analysis. This method helps assess the degree of similarity and consistency between the two ranking systems. As we are primarily interested in the top-ranked riders, we aim to understand how closely PCS and UCI rankings align among the elite riders. By exploring the correlation, we can discover whether the two ranking systems consistently recognize and reward the same set of top-ranked riders.

Correlation determines how closely two variables fluctuate. A positive correlation means that an increase of value in one variable will correspond with an increase of value in the other variable as well and the same goes for if the value decreases. A negative correlation means that an increase of value in one variable will correspond with a decrease of value in the other variable and vice versa. The higher the correlation value, the higher the degree to which those variables increase or decrease in parallel. A high correlation suggests that the two ranking systems are similar and that they assign points to riders or teams in a similar manner. Conversely, a low correlation suggests that the two systems are different and that they focus on different aspects of performance. For our use case, we use Spearman's rank correlation[24] to measure the correlation between the ranking systems. Spearman's rank correlation tells us whether the two rank variables covary. It measures the strength and direction of association between two ranked variables.

To understand Spearman's rank correlation, we need to understand monotonic functions. A monotonic function is one that either never increases or never decreases as its independent variable changes. Figure 3.1 illustrates the monotonic function. The first plot shows that as the variable X increases, the variable Y never decreases and the second plot shows that as the variable

X increases, the variable Y never increases. Both are monotonic functions. The last plot is not a monotonic function because as the X variable increases, the Y variable sometimes decreases and sometimes increases.



(a) *A monotonically increasing function*

(b) *A monotonically decreasing function*

(c) *A function that is not monotonic*

Figure 3.1: *Monotonic Function is a function that either never increases or never decreases as its independent variable changes.*

Spearman's rank correlation gives the measure of the monotonicity of the relation between two variables. It shows how well the relationship between two variables could be represented using a monotonic function. The following is the formula for Spearman's rank coefficient, where d_i is the difference between the two ranks of each observation and n is the number of observations:

$$p = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}$$

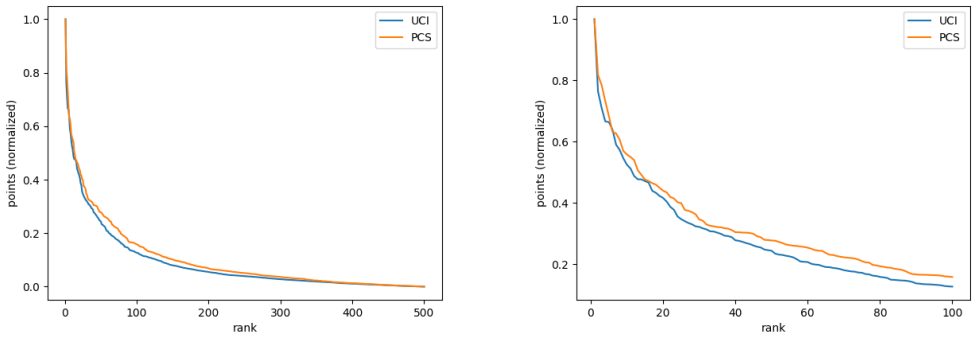
3.3 Results

In this section, the results of the analysis will be shown and discussed. The first part shows the results of the Comparative Graph Analysis while the second part focuses on the Correlation Analysis results of the two ranking systems.

3.3.1 Comparative Graph Analysis

Point distribution graph between top riders

The comparison of the aggregated points accumulated by the top 500 riders for both PCS and UCI from 2016 till 2019 reveals an interesting pattern. As can be seen in Figure 3.2a, this decaying curve is quite similar for the top end between both ranking systems but diverges at the lower ranks (top 20-300) before it converges again. Figure 3.2b gives a closer look at the point distribution curve at the top ranks. We observe that the difference between the two curves is minimal up until the top 20 ranks. We can hypothesize that predicting lower PCS ranks might have bigger inconsistencies with the UCI rank. However, the curve shows that the point distribution is nearly identical for the top 20 riders, which is our primary focus.



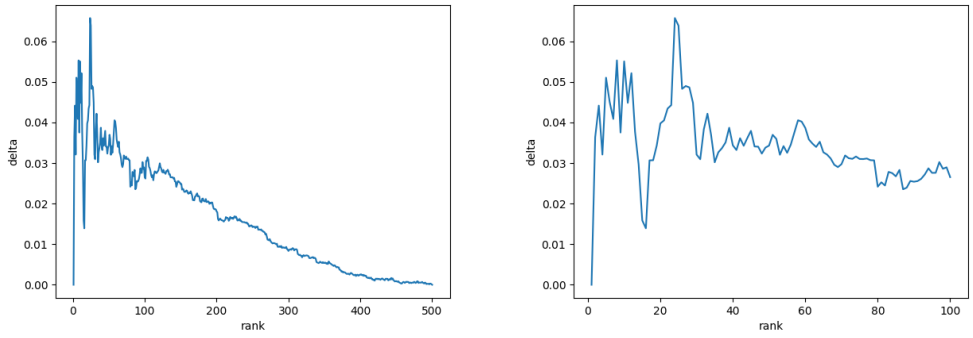
(a) **(b)**

Figure 3.2: *Normalized point distribution curve of top 500 and top 100 riders for PCS and UCI aggregated by taking their median from 2016 till 2019.*

Delta graph

Next, we turn our attention to the normalized delta graph as shown in Figure 3.3a. It depicts the difference in points between the top 500 riders' points for PCS and UCI ranking system from 2016 to 2019. The delta fluctuates more in the upper rankings and gradually decreases as the rank decreases. This

is reasonable because the scoring system awards more points to the upper ranks. Both PCS and UCI rankings assign more points to higher-ranked finishes. As the competition at the top is much fiercer, this means that the difference in points between the very top-ranked riders in PCS and UCI is likely to be larger than the difference in points between the lower-ranked riders. Figure 3.3b is a closer look at the delta graph and shows only the top 100 riders. We observe that riders around rank 30-35 experience the greatest changes. While there are fluctuations in the graph, the observed delta values indicate a relatively small difference.

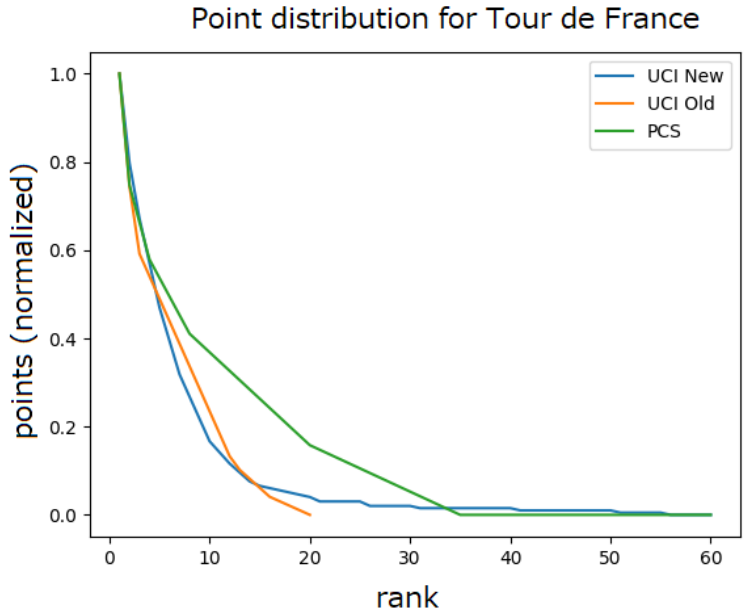


(a) **(b)**

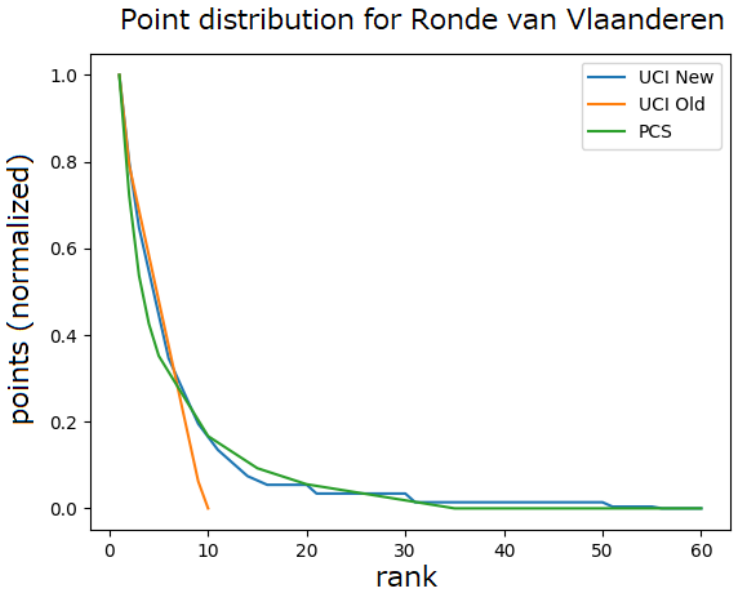
Figure 3.3: *Normalized delta graph between UCI and PCS points accumulated by the top 500 and top 100 riders aggregated by taking their median from 2016 till 2019.*

Race point allocation graph

Analyzing the race point allocation graphs of Tour de France's general classification section and Tour de Flandres also shows interesting results. As shown in Figure 3.4, while the point allocation for one-day races like Tour de Flandres (Figure 3.4a) is largely similar across PCS, old UCI (pre-2016) and new UCI (2017-current) ranking, there are visible differences for multi-day races such as Tour de France (Figure 3.4b). It indicates that while the point system for one-day races might be similar, the PCS point system might privilege more multi-day races.



(a)



(b)

Figure 3.4: *Normalized point distribution graph of Tour de France's general classification section and Tour de Flandres for PCS and UCI. UCI is split into two curves: Old UCI (pre-2016 when the last change occurred in the UCI ranking system), and UCI (current UCI from 2016 onwards)*

3.3.2 Correlation

The results of the correlation graph can be found in Figure 3.5. We calculated the Spearman correlation between PCS and UCI for different thresholds of rankings: top 10, 20, 30 and 40 riders from 2016-2021. The correlation was calculated separately for each year because we want to know whether the correlation is consistent throughout the year.

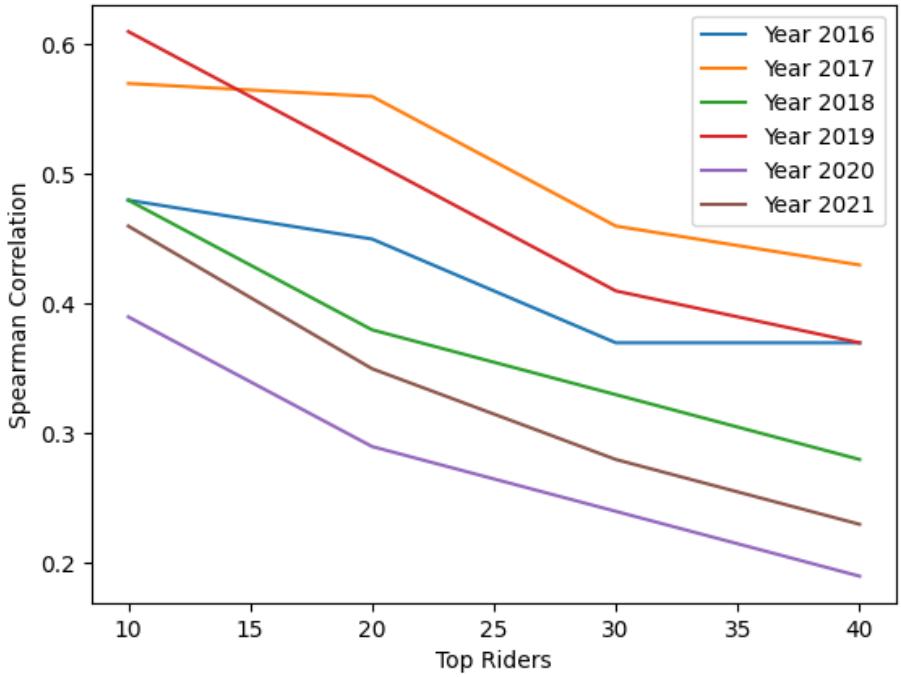


Figure 3.5: *Spearman Correlation curve between PCS and UCI for top 10/20/30/40 riders from 2016-2021. The correlation decreases with each increasing thresholds for the ranks and as such predicting lower PCS ranks will have larger inconsistencies with the UCI rank.*

Spearman's correlation ranges from -1 to 1, with values closer to -1 indicating a strong negative correlation, values closer to 1 indicating a strong positive correlation, and values closer to 0 indicating little or no correlation between the variables. As the rank threshold increases, the correlation decreases, meaning that predicting lower PCS ranks will have larger inconsis-

tencies with the UCI rank. This finding is in line with our earlier hypothesis. However, for the top 10 and top 20 ranks, there is a moderately significant positive correlation. This means that predicting these top ranks is more consistent with the UCI rank. The correlation curve for every year has a sharp drop after the top 20 rank, as such the PCS rank is not reliable enough to be consistent with the UCI ranking system after the top 20 rank. Another notable finding is the consistently lower correlation for the 2020 season. The outbreak of the COVID-19 pandemic in 2020 disrupted the cycling season which led to the cancellation or postponement of many races. This could have led to a lower correlation between the PCS and UCI rankings. Due to the pandemic, many riders chose not to participate in races due to health concerns. Travelling was also restricted in many parts of the world during the pandemic, which may have led to riders unable to travel to the race location and thus unable to participate in the race. We believe the difference in participation could have affected the results and rankings and thus led to a lower correlation between the PCS and UCI rankings in the 2020 season.

We also mapped the top 40 UCI riders from years 2016-2021 to their corresponding PCS ranking as shown in Figure 3.6. We can see that as the range of UCI ranks increases, PCS needs a larger range to cover the UCI rank. In Figure 3.6, we can see that a top 40 ranked rider in UCI, can be ranked as low as rank 70 in PCS. However, since we are mostly interested in the top ranks, we look at the top 10 ranked riders in UCI which in the graph is covered by the top 20 riders in the PCS ranking system. Another notable finding is the placements for the 2020 season. These placements are extreme when compared to other years. For example, a top 19 ranked UCI rider is ranked in the top 5 in PCS. This is a big discrepancy and it happened in the year when the COVID-19 pandemic was at its peak and disrupted the cycling season.

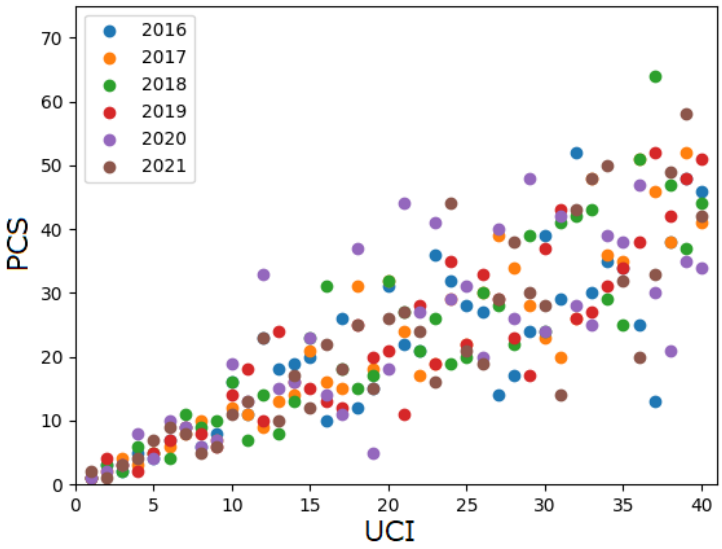


Figure 3.6: *UCI top 40 riders mapped to their PCS rankings from 2016-2021. While PCS needs a larger range of ranks to cover the UCI rank, the top 10 UCI ranked riders can be covered by the top 20 ranked riders of PCS.*

3.4 Conclusion

We compared the point distribution curve of the top 500 riders' points for PCS and UCI after normalization. It showed that the point distribution is quite similar for the top end between both ranking systems but diverges at the lower ranks before converging again. The difference between the two curves is minimal up until the top 20 ranks, which indicates that predicting the performance of the top 20 riders using the PCS ranking system is consistent with the UCI rank. However, predicting lower PCS ranks might have bigger inconsistencies with the UCI rank.

We plotted the delta graph of the difference between the top 500 riders' points for PCS and UCI ranking system which showed that the delta values fluctuate more in the upper rankings and slowly decrease as the rank decreases. This is logical because the scoring system awards more points to the upper ranks.

Calculating the Spearman correlation showed us that the correlation between PCS and UCI decreases as the rank threshold increases, which again indicates that predicting lower PCS ranks will have larger inconsistencies with the UCI rank. However, for the top 10 and top 20 ranks, there is a moderately significant positive correlation. There is also a consistently lower correlation for the 2020 season, which could be due to the outbreak of the COVID-19 pandemic that disrupted the cycling season.

Lastly, we mapped the top 40 UCI riders from years 2016-2021 to their corresponding PCS ranking which showed that a top 40 ranked rider in UCI can be ranked as low as rank 70 in PCS. However, we are mostly interested in the top ranks, and the top 10 ranked riders in UCI are covered by the top 20 riders in the PCS ranking system. The placements for the 2020 season are extreme when compared to other years, again this is likely due to the disruptions caused by the COVID-19 pandemic.

We can conclude that the analysis shows that the PCS ranking system can be used to predict the performance of the top 20 riders with a moderately significant positive correlation with the UCI rank. However, predicting lower PCS ranks might have larger inconsistencies with the UCI rank. Both PCS and UCI scoring systems award a lot more points to the upper rankings, which leads to larger differences in delta values for the upper rankings. We should also consider the impact of the COVID-19 pandemic on the cycling season when interpreting the results.

Chapter 4

Predicting Performance

This chapter is focused on the prediction performance of the riders. Based on the last chapter, the top 10 riders of UCI can be captured within the top 20 of PCS. In this chapter, we create a model that predicts whether the rider will be in the top 20 of the PCS ranking. We explain the steps we took to build our predictive model. This includes data collection, preprocessing, and model training. At the end of the chapter, we present and discuss the results and draw our conclusions.

4.1 Data

This section contains an overview of the dataset used for our project. All the data was gathered from the PCS website. Unfortunately, the PCS website does not provide any dataset. The website also does not have an API that could be used to directly extract the required data. Therefore, we developed a Python-based web scraper using the BeautifulSoup library to extract the required data from the website. As previously mentioned, the data should be consistent over a long period, so that the model can identify long-term patterns and trends accurately. We know from a previous study [4] that the average career length of a rider is 11 years. Taking into consideration the average career length and long-term data consistency, our dataset will include information that dates back up to 27 years ago from 1995 to 2022.

Moving on, we describe the types of data we extracted. We are interested in the top-ranked riders, so we extracted information about the rankings and PCS points of the top 500 riders from 1995 to 2022. Table 4.1 shows the format of the data that we extracted. Next, we extracted the rider profiles for each of the top 500 riders from years 1995 to 2022 from the PCS website. The rider profile contains information on all the races that the rider participated in and information on the race such as the type of race, distance travelled in that race, their PCS and UCI points gained for that race and the absolute and relative finish time of the rider as shown in Table 4.2. Additionally, information on a race was extracted if at least one of the top 500 riders participated in that race. The race profile has the statistics of the race on all the race participants such as the final ranking, points gained, relative finish time and whether or not a rider finished the race. The data format is illustrated in 4.3.

As we progress further into the next step of development for our model, it will be evident that we use the gathered data to create predictive features.

rider_name	Name of the rider
rider_slug	String to identify the rider
rider_url	Link to the rider's profile
team	Name of the rider's team
pcs_points	Total pcs points of the rider in that year
team_slug	String to identify the team
team_url	Link to the rider's team profile
rank	Rank of the rider at the end of that year

Table 4.1: *Top riders year: dataset format of the top 500 riders from year 1991 to 2022*

race_id	String to identify the race for a specific year
race_slug	String to identify the race
stage_slug	String to identify the stage
year	Year the race took place
race_type	Type of the race (ex.one day race)
class	Tier of the race (ex. 2.1)
race_name	Name of the race
race_url	Link to the race profile
rank	Rank obtained by the rider for the race
distance	Distance travelled in kilometers
pcs_points	PCS points obtained by the rider from the race
uci_points	UCI points obtained by the rider from the race
time_abs	Absolute finish time of the rider
time_rel	Relative finish time of the rider

Table 4.2: *Rider: dataset format of a rider*

rider_name	Name of the rider participating in the race
rider_slug	String to identify the rider
rider_url	Link to the rider's profile
team	Name of the rider's team
pcs_points	PCS points obtained by the rider from the race
uci_points	UCI points obtained by the rider from the race
team_slug	String to identify the team
team_url	Link to the rider's team profile
rank	Rank obtained by the rider in the race
time_abs	Absolute finish time of the rider
time_rel	Relative finish time of the rider

Table 4.3: *Race: dataset format of a race*

4.2 Feature Engineering

In this section, we explain how the raw data obtained from the PCS website was transformed into features used to train our model. We give a detailed explanation for each of the following processes: Feature Extraction in Section 4.2.1, Data Imputation in 4.2.2 and Feature Selection in 4.2.3.

4.2.1 Feature extraction

We grouped our features into different categories to help make the feature engineering process more organized. It also helps better understand which types of features are most important in predicting the target variable. Table 4.4 showcases all the features we extracted and the operations taken to obtain those features.

Features					Operations				
	Career avg	Ratio of last 3 years	Avg of last 3 years	Stdev of last 3 years	Career sum	Race format (one day/ stage races)	Previous year	Sum of last 3 years	Current Year
Performance features									
Number of participated races					X	X		X	X
Win ratio	X		X	X	X	X		X	X
Top 3 ratio	X		X	X	X	X		X	X
Top 5 ratio	X		X	X	X	X		X	X
Top 10 ratio	X		X	X	X	X		X	X
Win ratio slope		X				X			
Top 3 ratio slope		X				X			
Top 5 ratio slope		X				X			
Top 10 ratio slope		X				X			
PCS points			X	X			X		X
PCS rank			X	X			X		X
Race result features									
Flat one day races						X			X
Hilly one day races						X			X
Cobble classics						X			X
Multi day races						X			X
Additional features									
Rider age					X				X
Career year					X				X
Rider BMI					X				X

Table 4.4: *Feature table showcasing all the types of operation taken to get the different types of features.*

Performance features

Performance features are features related to a rider's performance in races. These include features such as win ratio, top 3 ratios, top 5 ratios, and top 10 ratios. These ratios measure the rider's success in finishing in the top positions in races they participate in. Each of these features is further extended into multiple features by separating them according to different operations such as the career average, the average and standard deviation in the last 3 years, the ratio of the current year and the race type (one day/stage races). Looking only at the results of the whole career does not give us accurate information about the current form of the rider and taking only the current year can give us skewed results for a rider's performance because of anomalies such as injuries or the rider not being consistent enough and only having one lucky year. The results of the last 3 years give us more information to more accurately judge the performance of the rider. Furthermore, separating the feature by race type tells us the type of race in which a rider performs better.

Other performance features include the number of participated races, the PCS points and rank of the rider and the slope of the ratios. The number of participated races tells us how active the rider has been and separately accounting for the number of participated races in one-day races and stage races separately lets us know which type of race the rider frequents more. The PCS points and PCS rank are also one of the main determining factors that measure a rider's performance and they are accounted for as well. The slope of the ratios is an indicator of a rider's performance increasing or decreasing regarding his top-place finishes. It is represented by the slope coefficient which encapsulates the change in the ratios over time and is calculated using a linear regression method. An example of the slope ratio can be seen in Figure 4.1.

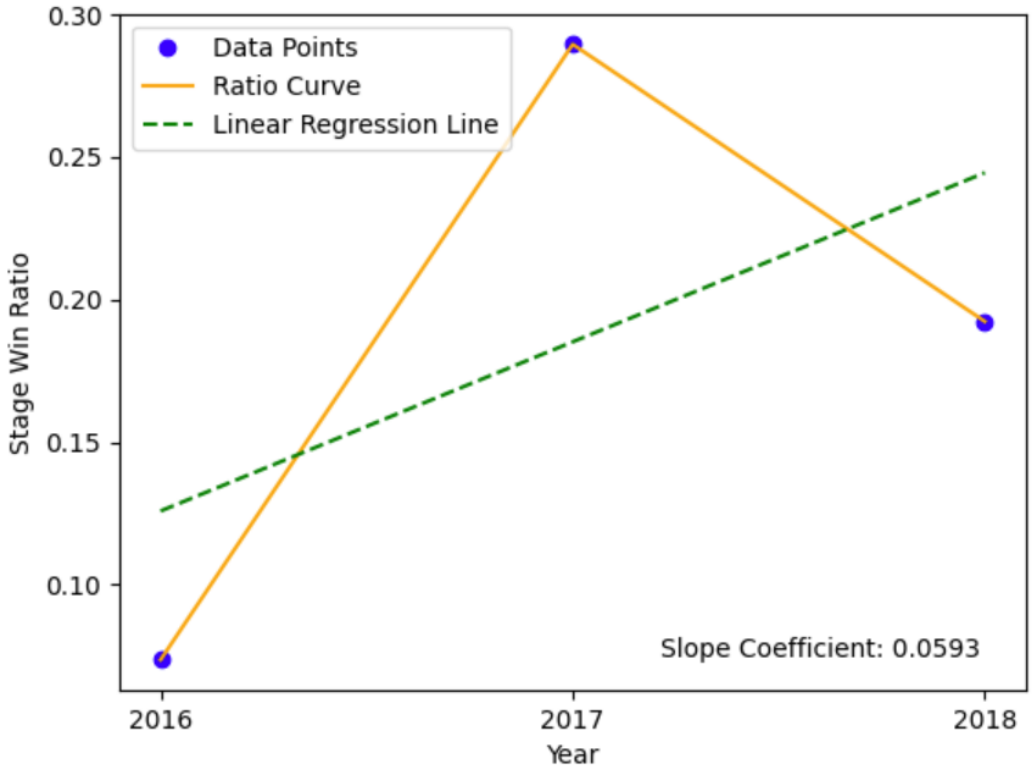


Figure 4.1: *Stage win ratio slope of a rider. Using a linear regression method on this curve gives us a slope coefficient of 0.0593.*

Race result features

Race result features contain features that measure the performance of riders for specific races. These races are categorized into four types: flat races, hilly races, cobble classics and multi-day races. An overview of the included races can be found in Table 4.5. These races all have different types of profiles. Race result features can help differentiate the performance of riders in different types of races.

Additional features

The additional features include the rider's age and how long they have been active professionally. The age when a rider reaches their peak performance might differ for each rider. As a rider passes their peak age, gradually the physical capabilities of the rider will start to decline and their performance

Flat one-day races	Hilly one-day races	Cobble classics	Multi-day races
Milano Sanremo	Liège Bastogne Liège	E3 Harelbeke	Paris - Nice
Cyclassics Hamburg	Il lombardia	Gent-Wevelgem in Flanders Fields	Tirreno Adriatico
	Strade Bianche	Ronde van Vlaanderen - Tour des Flandres ME	Itzulia Basque Country
	Cadel Evans Great Ocean Road Race	Paris-Roubaix	Tour de Romandie
	La Flèche Wallonne		Critérium du Dauphiné
	Amstel Gold Race		Tour de Suisse
	San Sebastian		Tour de Pologne
	Bretagne Classic - Ouest-France		Benelux Tour
	Grand Prix Cycliste de Québec		Santos Tour Down Under
	Grand Prix Cycliste de Montréal		Volta Ciclista a Catalunya
			Tour de France
			Giro d'Italia
			La Vuelta ciclista a España

Table 4.5: *List of races split into different groups, each with varying profiles, used for the race features.*

might degrade due to it. The length of a rider's career can indicate the experience of the rider. Riders who have been competing for a longer period have had more time to develop their skills which can be an important factor in predicting future success. Age and career together can also provide insight into a rider's performance trajectory. For example, a young rider with a short career length might have potential but not the time to fully develop their skills yet. An older rider with a long career might be past their peak age and show a decline in their performance. As such, we hypothesize that including age and career length as a feature helps us more accurately determine a rider's performance.

4.2.2 Data Imputation

After feature engineering, we have a dataset with a set of features. However, we still face the problem of missing data. These missing data come from the fact that not every rider participates in every race. Most of the time a rider will focus on races that align with his speciality. As such, riders will have some features where the data is missing because they did not participate in a race. A high number of missing values means the potential loss of a lot of insightful information which can have a big impact on the outcome of our model by creating a bias in the dataset. It also limits the algorithms that are possible to use as not every algorithm has a built-in method to deal with

missing values. This leads to our model being less reliable and trustworthy. To deal with this problem, we implement the data imputation technique to substitute the missing values in our dataset.

Mean/Median Imputation

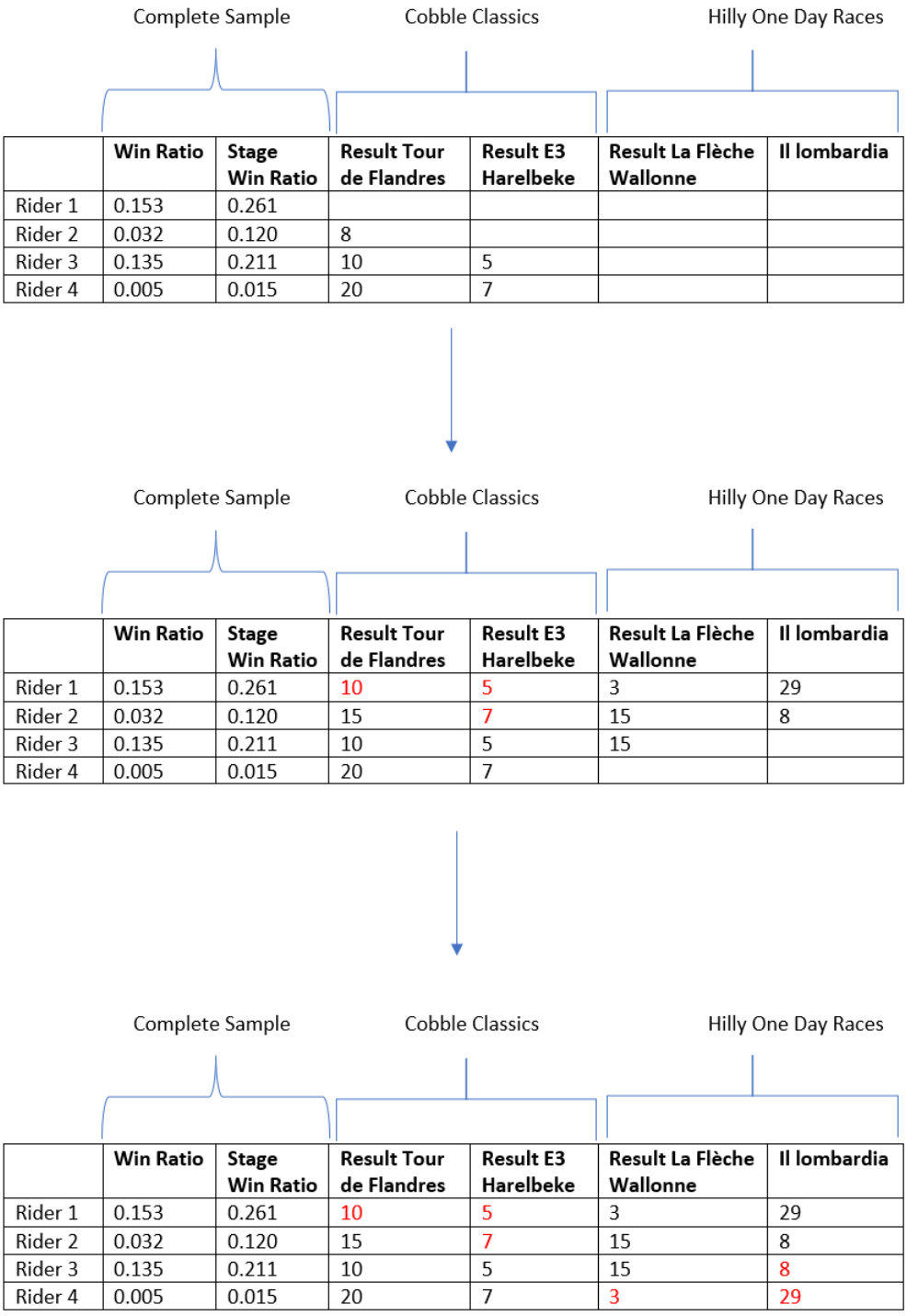
There are different methods of data imputation. One such common method is the mean or median imputation[25]. As the name implies, the mean or median is used as input for the missing data. This type of method can be implemented without the need for a complete case which makes it very easy to implement. However, the estimates resulting from this method often lead to low variance in data.

KNN

Another method is the KNN imputation where the missing values are replaced using their neighbors from the complete cases[25]. A limitation of KNN imputation is that it needs complete cases to estimate the missing values. However, our dataset does not have any complete cases since it is physically impossible for a rider to compete in all races.

Altered KNN

To combat the problem of there not being any complete cases in our dataset, an alternate way to perform KNN imputation is proposed[22]. First of all, the features with missing values are split into groups based on domain knowledge. The groups are based on the different types of races as shown in the Table 4.5. Then, we impute each group of these groups separately. Note that the remaining features that are not part of any of these groups form the complete observed set. During the actual modeling process, we find the best k using rolling cross-validation.



	Win Ratio	Stage Win Ratio	Result Tour de Flandres	Result E3 Harelbeke	Result La Flèche Wallonne	Il lombardia
Rider 1	0.153	0.261				
Rider 2	0.032	0.120	8			
Rider 3	0.135	0.211	10	5		
Rider 4	0.005	0.015	20	7		

Complete Sample
Cobble Classics
Hilly One Day Races

	Win Ratio	Stage Win Ratio	Result Tour de Flandres	Result E3 Harelbeke	Result La Flèche Wallonne	Il lombardia
Rider 1	0.153	0.261	10	5	3	29
Rider 2	0.032	0.120	15	7	15	8
Rider 3	0.135	0.211	10	5	15	
Rider 4	0.005	0.015	20	7		

Complete Sample
Cobble Classics
Hilly One Day Races

	Win Ratio	Stage Win Ratio	Result Tour de Flandres	Result E3 Harelbeke	Result La Flèche Wallonne	Il lombardia
Rider 1	0.153	0.261	10	5	3	29
Rider 2	0.032	0.120	15	7	15	8
Rider 3	0.135	0.211	10	5	15	8
Rider 4	0.005	0.015	20	7	3	29

Figure 4.2: *Example of an alternate KNN imputation using imputation groups to split data to get complete samples.*

A simplified version of our dataset with 4 samples and 6 features where we set the number of neighbors, K=1 can be seen in Figure 4.2. The features are divided into 3 parts: Complete samples, Cobble Classics and Hilly one-day races. The latter two are our imputation groups. The first table in Figure 4.2 is the dataset before imputation. If we used the standard KNN imputation then we would need a complete sample which is not available in our dataset. To get complete samples we split the data by the imputation groups. This results in two groups: Complete samples + Cobble Classics and Complete samples + Hilly one day races. In the second table in Figure 4.2, we see the KNN imputation implemented on the Complete samples + Cobble Classics group where riders 3 and 4 are considered complete cases. Rider 1's missing values get filled with the same value as that of rider 3 because of his similarity to rider 3 rather than to rider 4. Following the same process, rider 2 is more similar to rider 4 than rider 3 so he gets the same value as rider 4. The process is repeated in the last table in Figure 4.2 for the Cobble Classics + Hilly one-day races group. This time riders 1 and 2 are considered complete cases. Note that the results from the previous table are not considered when imputing. Once all groups have been imputed, the dataset is complete as seen in Figure 4.2.

Data Imbalance - Synthetic Minority Over-Sampling Technique (SMOTE)

Note that our dataset is imbalanced. Since there can only be 20 top riders each year, the distribution of samples across the classes is incredibly skewed. Most algorithms for classification assume an equal number of examples for each class so when data imbalance occurs, we get models that have poor predictive performance, especially for the minority class[26]. There exist different techniques to combat this problem such as upsampling the minority class, downsampling the majority class, generating synthetic data, or using balanced class weights. We used a technique called SMOTE which is an oversampling method that generates synthetic data using the KNN algorithm. Synthetic data is generated by applying the following formula for SMOTE to each sample in the minority class, where x_i is a sample of the

minority class, x_j is a neighbor of x_i, λ is a uniformly chosen random number between 0 and 1 and $x_i^{'}$ is the new synthetic sample:

$$x_i^{'} = x_i + \lambda(x_j - x_i)$$

4.2.3 Feature Selection

High dimensionality problem

After applying feature engineering to our dataset, we get a total of 84 features. For a dataset of our size that only includes the top 500 riders from the years 1991 to 2022, this amount of features is quite large. As the number of features grow, the possible parameter values of a data entry increase exponentially[27], which affects the computational time, space and accuracy of our models. We refer to this problem as the curse of dimensionality[27].

Selection Method

A solution to avoid the curse of dimensionality is to apply feature selection. By applying feature selection, a subset of features is selected to be used for our training model. There are 3 types of feature selection methods: Wrapper methods, Filter methods and Embedded methods [27]. Wrapper methods try a different subset of features until the optimal subset is reached. Drawbacks of this method are the large computation time for data with many features and the tendency to overfit when there is not a large number of data points. Filter methods select a subset of the features by ranking them by a useful descriptive measure. It has a low computation cost and will not overfit the data. Embedded methods are a mix of the two aforementioned methods. It selects and tunes the subset of features during the model creation process.

We used SelectKBest which is a filter method. This method removes all but the k highest scoring features. The scoring function used is mutual information. It measures the dependency between the two variables and captures any kind of statistical dependency where higher values mean higher dependency.

Note that the feature selection step is performed after the value imputation step, but before any algorithm is applied.

4.3 Algorithms

There are 5 algorithms considered for our models, and each of them is briefly discussed below.

4.3.1 Logistic Regression

Logistic Regression is a classification technique that uses a logistic function to predict the dependent variable. Hosmer et al.[28] have shown that training with the algorithm is very efficient and it is fast at classifying unknown records. The following is the equation of a logistic regression, where Y is the dependent variable and $X = [x_1, x_2, ..., x_n]$ is a vector of n independent variables:

$$P(Y = 1|X) = \frac{1}{1 + e^{-(\beta_0+\beta_1 x_1+...+\beta_n x_n)}}$$

The coefficients $\beta_0, \beta_1, ..., \beta_n$ were estimated using Maximum Likelihood Estimation (MLE). As Logistic regression is quick and easy to implement, we used this algorithm as a benchmark model to measure performance.

4.3.2 Random Forest

Random Forest is a supervised learning algorithm that uses an ensemble method [29]. It builds decision trees from bootstrapped data samples and a random subset of features is used to grow the tree at each node. It considers the results of all the trees that are combined to make the final result more robust. Random forest models are fast to implement due to the limited amount of parameters that need tuning and the random selection of features and data samples to build trees lowers the risk of overfitting.

4.3.3 XGBoost

XGBoost is a gradient boosting algorithm first introduced by Chen et al.[30]. In gradient boosting, multiple weak models are created and then combined to get better performance as a whole. These models are usually decision trees. Unlike Random Forest where trees are built in parallel, the boosting technique combines the models sequentially. Each iteration of the model learns from the previous iteration and reduces the error. Thus, the model learns iteratively by adjusting the error towards the optimum. XGBoost utilizes this boosting technique and made further improvements by introducing regularization to reduce overfitting. XGBoost also improved the loss function. It took into account the second order derivative of the loss function, which reduced the error.

While XGBoost has proven to be a competitive algorithm in many related works [22][18], it can take a long time to train because of the high amount hyperparameters that need to be tuned.

4.3.4 Catboost

Catboost was first introduced by Dorogush et al.[31] and it is a gradient boosting algorithm that uses ordered boosting, a variant of gradient boosting, to better handle categorical features. CatBoost has a relatively low training time compared to other boosting algorithms[31].

4.3.5 Light Gradient-Boosting Machine (LightGBM)

Like XGBoost and CatBoost, LightGBM is a gradient boosting framework that uses decision trees for classification and regression tasks. But unlike XGBoost or CatBoost, where the decision trees are grown with a depth-first approach, LightGBM builds its trees leaf-wise by building upon the leaf with the highest gain. This results in more loss reduction and gives higher accuracy while also being faster[32].

4.4 Hyperparameter Optimization

We need to tune the hyperparameters for the algorithms we are using as choosing the optimal parameter settings can have a significant impact on our model's performance. In this section, we explain the approach we took to optimize the hyperparameters of the algorithms.

4.4.1 Rolling cross-validation

We first explain the process of tuning the parameters for our models. Hyperparameter optimizations are most of the time performed through a standard cross-validation approach. However, we believe that rolling cross-validation is a better suited approach for our use case. In road cycling, riders compete with other riders for a whole season and it is immediately followed by the next season. We believe the data that we have to not be independent. When building a model with standard cross-validation, information from the future can leak into our model when our model is not supposed to know that information which leads to biased predictions. Hence, we use rolling cross-validation to avoid this leakage of information. The model is trained on past data and tested on future data.

The rolling cross-validation process is visualized in Figure 4.3 where green boxes signify the training period, yellow is the validation period and red is the testing period. For each hyperparameter combination for an algorithm, the model trains on a total of 11 different periods and for each period it validates for the next year, and tests on the year after that. The final evaluated metric is the average of all the metrics calculated in the different periods. The hyperparameters with the best metric result during the validation set are used for the testing set.

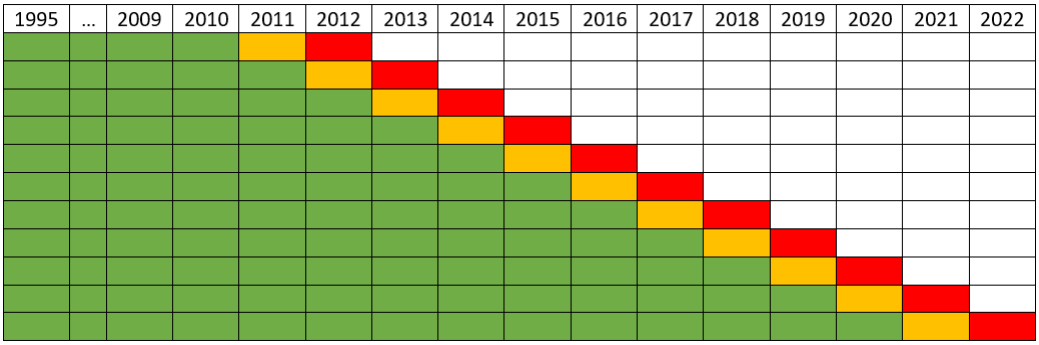


Figure 4.3: *Rolling cross-validation: each row is a different train/val/test period, green boxes represent the training period, yellow boxes represent validation period and red boxes represent testing period*

4.4.2 Hyperparameter Tuning

This section explains the parameters that were tuned for each of the algorithms. The loss function used was the log-loss. Table 4.6 shows the hyperparameters and their considered values for all of the algorithms.

Random forest

The hyperparameters considered for tuning in Random Forest as shown in Table 4.6 are the *number of trees*, the *maximum depth* of the individual trees and the *minimum samples to split* on at an internal node of the trees. A large number of decision trees creates more robust models with less variance, at the cost of increased training time[33]. So, optimizing the number of trees will reduce model variance and lead the model error to an approximate optimum value. Maximum depth value is the depth of each tree, and the higher the depth value the larger the tree becomes which leads to a greater chance of overfitting the training data. However, due to the ensemble method of Random Forest [29], overfitting rarely occurs. Larger trees mean increased computation time, so we tune this parameter and set a limit on the maximum depth value. The *min_sample_split* parameter also needs to be carefully tuned to ensure that the node splits are based on more representative sample. Setting this parameter too low can cause overfitting since underlying

patterns in the data may not be captured because of the small number of samples at a node, while setting it too high can lead to underfitting.

Algorithm	Hyperparameter	Candidate values
Logistic Regression	/	/
Random Forest	Maximum depth	[6,8,10,12]
	Number of trees	[100,200,300,400,500]
	Minimum sample split	[2,3,5]
XGBoost	Alpha	[0,4,8,12]
	Number of trees	[100,200,300,400]
	Max depth	[6,8,10,12]
	Learning rate	[0.1,0.3,0.5,0.7,0.9]
	Lambda	[1,3,5,7]
	Subsampling rate	[0.5,0.7,1.0]
	Feature Subsampling rate	[0.1,0.3,0.5,0.7,1.0]
	Gamma	[0,4,8,12]
CatBoost	Number of trees	[100,200,300,400]
	Max depth	[6,8,10,12]
	Learning rate	[0.1,0.3,0.5,0.7,0.9]
	Lambda	[1,3,5,7]
	Subsampling rate	[0.5,0.7,1.0]
	Feature Subsampling rate	[0.1,0.3,0.5,0.7,1.0]
	Minimum samples in leaf	[0,4,8,12]
LightGBM	Alpha	[0,4,8,12]
	Number of trees	[100,200,300,400]
	Max depth	[6,8,10,12]
	Learning rate	[0.1,0.3,0.5,0.7,0.9]
	Lambda	[1,3,5,7]
	Subsampling rate	[0.5,0.7,1.0]
	Feature Subsampling rate	[0.1,0.3,0.5,0.7,1.0]
	Gamma	[0,4,8,12]
	Num leaves	[10,20,30,40,50]
	Min data in leaf	[5,10,15,20]
NGBoost	Number of trees	[100,200,300,400,500]
	Learning rate	[0.1,0.3,0.5,0.7,0.9]
	Feature Subsampling rate	[0.1,0.3,0.5,0.7,1.0]

Table 4.6: *This table lists the considered hyperparameters and their candidate values for each algorithm.*

XGBoost

XGBoost has a lot of parameters that need to be carefully tuned. As shown in 4.6, we tune the following hyperparameters: *Alpha, Number of trees, Max depth, Learning rate, Lambda, Subsampling rate, Feature subsampling rate and Gamma.*
Alpha is a regularization parameter for controlling the L1 regularization and is tuned to prevent overfitting. A higher value means stronger regularization and makes the model more generalisable but if the value is too high it can lead to underfitting the model. The reasons for tuning the *number of trees* and the *maximum depth* of a tree are the same as discussed in Random Forest. The *learning rate* controls the gradient decent step size at each iteration[34]. By slowing down the learning in the gradient boosting model, it can prevent the model from overfitting. However, setting the learning rate value too low can lead to the model converging very slowly. Like Alpha, *Lambda* is also a regularization parameter and it controls the L2 regularization and is tuned to prevent overfitting. *Subsampling rate* controls the fraction of observations that are randomly sampled for each tree while *feature subsampling* controls the fraction of features that are randomly sampled for each tree. Both hyperparameters are tuned to prevent overfitting and underfitting. Finally, the *Gamma* parameter specifies the minimum loss reduction needed to make a new split in the tree. It can prevent overfitting by making sure a new split is only made if it improves the model's performance[34].

CatBoost

The hyperparameters tuned for CatBoost are for the most part the same as for XGBoost. The only difference is the absence of *alpha*. Like XGBoost, CatBoost also has a *feature subsampling* hyperparameter but unlike XGBoost, where the fraction of features is randomly sampled for each tree, feature subsampling in CatBoost is performed at each level of the tree [31]. CatBoost also does not have a *gamma* hyperparameter but instead has an equivalent hyperparameter called *min_data_in_leaf*. It controls the minimum number of samples required in a leaf node to grow the tree and prevents overfitting.

LightGBM

Most of the hyperparameters we tuned for LightGBM are the same as the ones that we tuned for XGBoost and they serve the same purpose as in XGBoost. LightGBM also has the *min_data_in_leaf* hyperparameter which serves the same purpose as in CatBoost. Aside from the common hyperparameters, we also tune the *number of leaves* which controls the maximum number of leaves in a tree. If the number of leaves is too high, it can result in the model overfitting while if the number is too low then the model may not capture the underlying patterns in the data.

4.5 Evaluation Metrics

Our model predicts whether a rider will make it in the top 20 PCS ranking for next year and to evaluate this performance, we chose several evaluation metrics used for classification models. We also took into account the class imbalance that is present in our dataset and chose the following metrics for evaluation: Precision, Recall, and F1-score.

4.5.1 Precision

Since we are interested in who is going to be in the top 20 riders, *Precision* is an important metric as it measures the proportion of positive predictions that are correct. It answers our question by telling us how many riders predicted to be in the top 20 PCS ranking are actually in the top 20. The following is the formula for *Precision*, where TP stands for *True Positive* and FP stands for *False Positive*:

$$Precision = \frac{TP}{TP + FP}$$

4.5.2 Recall

The *Recall* is another evaluation metric that is interesting for our model. It measures the proportion of true positives that are correctly identified by the model and thus, tells us how many of the riders that made it into the top 20 are correctly predicted by our model. The following is the formula for *Recall*, where TP stands for *True Positive* and FN stands for *False Negative*:

$$Recall = \frac{TP}{TP + FN}$$

4.5.3 F1-score

F1-score is used to compare the performance between models. It combines the precision and recall metrics by taking their harmonic mean. Since F1-score considers both false positives and false negatives, it is especially useful when the classes are imbalanced. The following is the formula for F1-score:

$$F1\text{-}Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

4.6 Results

This section shows the results for all our models that were evaluated using the metrics that we discussed in Section 4.5. We compare the models on each of the metrics and discuss which model performed the best.

4.6.1 Precision

As previously stated in Section 4.4.1, by using rolling cross-validation our models are trained, validated and tested on 11 different periods as shown in Figure 4.3. This means that there are 11 models for each algorithm and the average across all the periods is used for evaluation. In addition, the boosting algorithms have built-in methods to handle missing values without just

ignoring them. These variants of the boosting algorithms were also trained and tested alongside our method of data imputation through an alternate KNN method discussed in Section 4.2.2. Including all the variants of the algorithms, we end up with 8 algorithms with 11 models each, so we have 88 models in total. Table 4.7 shows the average precision values for our models where for each column we set the value for the overall best performing model in bold.

Model	Precision Train	Precision Validation	Precision Test
Logistic Regression	0.88117	0.50482	0.48110
Random Forest	0.94135	0.71956	0.64152
Random Forest with Altered KNN	0.99575	0.76887	0.67966
XGBoost	0.95782	0.86013	0.76993
XGBoost with Altered KNN	**0.99945**	**0.88469**	0.78403
CatBoost	0.97996	0.80177	0.75274
CatBoost with Altered KNN	0.99549	0.82894	**0.78468**
LightGBM	0.96367	0.81473	0.70961
LightGBM with Altered KNN	0.99249	0.84810	0.74976

Table 4.7: *Aggregated precision results of the models for training, validation and testing. The best scoring models are highlighted in bold for the train, validation and test set.*

The results show varying levels of performance between the different models. In general, we can see that the boosting algorithms outperformed the other algorithms and the performance between the boosting algorithms is also comparable. Our base model, the Logistic Regression performed the worst. Logistic Regression works well when the relationship between the independent variables and dependent variable is linear but if it is non-linear then the performance suffers. Another thing to note is the performance of the models with our proposed way of data imputation through an altered KNN (Section 4.2.2) performed better than the default way the algorithms handle missing values. Overall, we saw about a 2-4% increase in precision which we consider significant because even small differences can have a large impact on the outcomes. Overall, the performances of the boosting models were similar across all the sets. XGBoost with Altered KNN performed the best for the training set and validation set with a precision of 0.99945 and

0.88469 respectively, while CatBoost with Altered KNN performed the best for the test set with a precision of 0.74976.

4.6.2 Recall

The average recall values for our models can be seen in Table 4.8 where for each column we set the value for the overall best performing model in bold.

Model	Recall Train	Recall Validation	Recall Test
Logistic Regression	0.88091	0.50169	0.47392
Random Forest	0.93991	0.71666	0.63950
Random Forest with Altered KNN	0.99198	0.75879	0.67511
XGBoost	0.95606	0.85812	0.76975
XGBoost with Altered KNN	**0.99892**	**0.88278**	0.78017
CatBoost	0.97985	0.80067	0.75144
CatBoost with Altered KNN	0.99526	0.81995	**0.78290**
LightGBM	0.95749	0.81357	0.70823
LightGBM with Altered KNN	0.99119	0.84759	0.74775

Table 4.8: *Aggregated recall results of the models for training, validation and testing. The best scoring models are highlighted in bold for the train, validation and test set.*

We observe that the boosting algorithms outperformed Logistic Regression and Random Forest for the recall metric as well. This might be because boosting algorithms generally handle feature interactions and class imbalance problems better than Logistic Regression and Random Forest. Our method of altered KNN data imputation also performs better than the standard way these algorithms deal with missing data. We observe that so far this increased performance is consistent across both precision and recall metrics. The best performing models were also the same as the ones for precision. XGBoost with Altered KNN performed the best for the training set and validation set with a recall of 0.99892 and 0.88278 respectively, while CatBoost with Altered KNN performed the best for the test set with a recall of 0.78290.

4.6.3 F1-score

The average F1-score values for our models can be seen in Table 4.9 where for each column we set the value for the overall best performing model in bold.

Model	F1-score Train	F1-score Validation	F1-score Test
Logistic Regression	0.88104	0.50325	0.47748
Random Forest	0.94063	0.71811	0.64051
Random Forest with Altered KNN	0.99386	0.76380	0.67737
XGBoost	0.95694	0.85912	0.76984
XGBoost with Altered KNN	**0.99918**	**0.88373**	0.78209
CatBoost	0.979905	0.801218	0.752084
CatBoost with Altered KNN	0.995375	0.824235	**0.78378**
LightGBM	0.96874	0.81415	0.70892
LightGBM with Altered KNN	0.99184	0.84784	0.74875

Table 4.9: *Aggregated F1-score results of the models for training, validation and testing. The best scoring models are highlighted in bold for the train, validation and test set.*

The observations for the results of F1-scores are similar to the observations we made for the results of precision and recall metrics. First, the boosting algorithms gave the best results. Since these algorithms are built to handle complex and high-dimensional data, it makes sense they gave the best results. Second, the models that used our method of KNN imputation performed better than those that didn't use this method. We have now observed that this method has consistently improved the performance of the models across all evaluated metrics. Third, the performance difference between the boosting algorithms is relatively small and Logistic Regression is consistently the worst performing model. For the training set and validation set, XGBoost with Altered KNN performed the best with an F1-score of 0.99918 and 0.88373 respectively, while CatBoost with Altered KNN performed the best for the test set with an F1-score of 0.78378.

4.7 Model Discussion

In this section, we first discuss some observations we made in the previous section. We made the observation that the Altered KNN imputation method combined with the boosting algorithms gave us the best results across all evaluated metrics. When comparing the precision and recall metric, we discovered that the models score better albeit very slightly on precision. While it would be ideal to have a higher recall, as false negatives are more concerning for our use case. A false negative means that a rider who was supposed to be in the top 20 was not predicted as such by our model. In a real-world scenario, it would mean our model missed the mark and might have negatively affected the rider's career. While we know the importance of this, we believe that the precision and recall scores are close enough (0.001% difference) that the impact of this difference on the model is negligible.

Next, we aim to understand how our model is making predictions. We try to accomplish this by identifying the features which that had the largest impact on the outcomes of our models. Across all metrics for the training, validation and test set, XGBoost with altered KNN performed the best on average. It had the best score for the training and validation set, while it trailed very closely behind CatBoost for the test set (0.001% difference). As such, we choose this model to showcase the feature importance. We used the SHAP method invented by Lundberg et al.[35] to quantify the importance of each feature. The SHAP values represent the contribution of each feature to the outcome of a prediction.

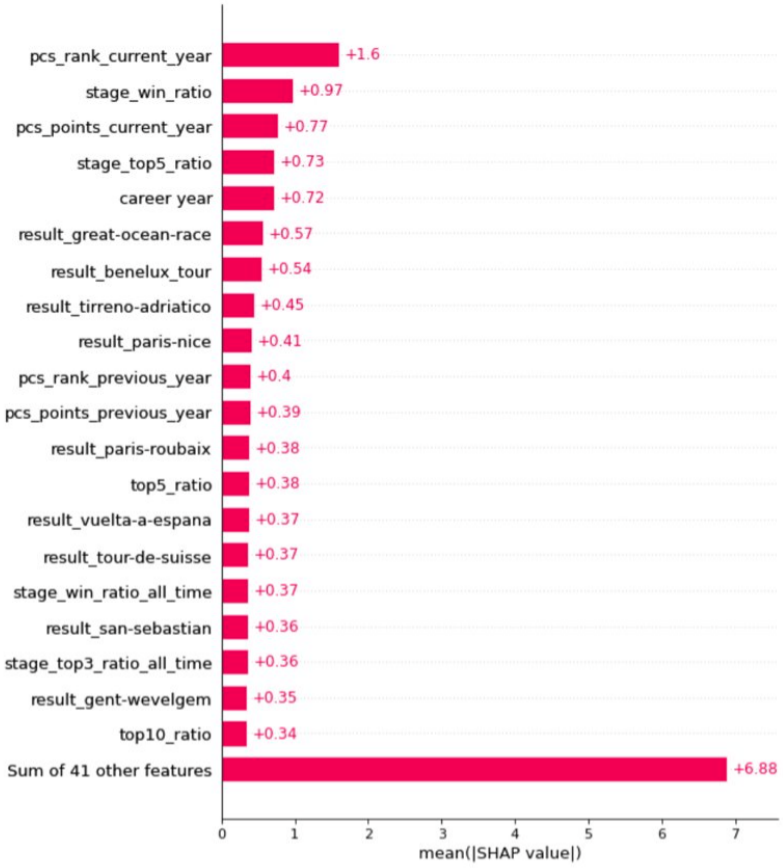


Figure 4.4: *SHAP values for the XGBoost model.*

The feature importance for our XGBoost model is shown in Figure 4.4. We see that the *pcs_rank_current_year* is the largest contributor. This is reasonable as we are trying to predict whether a rider will make the top 20 for the general ranking which includes all the races, and our model sees features such as the most recent rank and points of a rider as a good indicator of a rider's current strength. We also observe that *stage_win_ratio* is the second largest contributor and *stage_top5_ratio* is among the top 5 contributors. This indicates that riders who consistently finish high on stage races are more likely to make it into the top 20 rankings. We also observe that consistency is a key factor as showcased by features such as *stage_win_ratio* and *stage_top5_ratio* having high mean SHAP values.

We also see some races that have a high feature importance. Some of these races such as Paris-Roubaix and Vuelta a España are very prestigious races in road cycling. There is also Tirreno-Adriatico which is held in high regard as a preparatory race for Giro d'Italia, one of the big three races. Benelux Tour also showed a fairly high contribution. As it is a relatively new race added to the UCI World Tour calendar, we decided to analyze the credibility of this feature's contribution. Analysing our dataset before imputation, we found that Benelux Tour had the most amount of missing values. It's possible that the imputation method accurately captured the relationships in our dataset which led to Benelux Tour contributing useful information. However, it's also possible that the imputation process overlapped different features that already provide similar information. By conducting a sensitivity analysis, where we removed the Benelux Tour feature, we found our test results to be similar to before the removal of this feature. This leads us to believe that even though Benelux Tour had a high SHAP value after imputation, its impact on the model's performance might be limited. We can also deduce that our model is not heavily reliant on one specific feature and the predictions are based on a combination of multiple factors.

Coaches should tread carefully when referencing these races, rather than being a guideline to determine potential top 20 contenders, as their importance might be solely due to the imputation values. Another interesting feature that has a high contribution to our models decision making is the *career_year*. This means that our model emphasizes the fact that the length of the rider's career has a significant impact on the chances of them making it to the top 20. Thus, we believe that the length of a rider's career can be indicative of his experience and ability to endure and compete at a high level which can be important in predicting future success.

Chapter 5

Conclusion

One of the goals of this thesis was to determine how well the PCS ranking system represented the UCI ranking system. We analyzed and compared the two different ranking systems through various methods. We compared their point distribution curves, plotted the delta graph of the difference between the top 500 riders' points and we calculated the correlation between the two through the Spearman correlation method. A mapping of the top-ranked riders from both ranking systems showed that the top 20 riders in PCS covered the top 10 riders in UCI. Through our analysis, we concluded that the PCS ranking system could be used to predict the performance of the top 20 riders with a moderately significant positive correlation with the UCI rank but predicting lower PCS ranks might have larger inconsistencies with the UCI rank. The effect of COVID-19 was also visible in the 2020 season.

Another goal of this master thesis was to build a model that can predict the performance of professional road cyclists by classifying them into whether they will be top 20 riders or not. This ranking was not limited to certain types of races but rather the overall general rankings. For long-term data consistency, we collected data that dates back up to 20 years ago from 1995 till 2022. We extracted features from the data that we believe to be related to the performance of riders such as the race results of the rider, their win ratio, accumulated points, length of their career, age, etc. We used different techniques to further improve our dataset. We used an alternate form of KNN to

substitute the missing values in our dataset. SMOTE was applied to handle data imbalance and a filter selection method was used to combat the high dimensionality problem in our dataset. We used a rolling cross-validation approach to train our models instead of the standard cross-validation used in most works. This approach was taken to prevent future information leaking into our model during training leading to biased predictions. We used several machine learning algorithms and compared their performance through our evaluation metrics: Precision, Recall, and F1-score. A key finding was the improvement provided by the data imputation technique. When compared to the performance of the algorithms when we let them handle missing values with their built-in method, this new way of handling the missing data derived from previous work[22] improved model performance across all evaluated metrics. The precision was slightly higher than the recall for all of our models. We believe a higher recall to be more desirable because misclassifying a top 20 rider is a bigger mistake than misclassifying a not-top 20 rider. However, we believe that on both metrics our model provided similar scores (0.001% difference) and that the impact of this difference on the model is negligible. Overall, the boosting algorithms XGBoost, CatBoost, and LightGBM outperformed the other algorithms Random Forest and Logistic Regression. The performance of the boosting algorithms were comparable but on average across the testing, validation and test set, XGBoost performed slightly better.

A feature importance performed on our XGBoost model through the SHAP method showed that a rider's current rank and points have the largest impact on the model's prediction but also consistently winning stage races was a large contributor in a rider making it to the top 20 rankings. The length of a rider's career was another factor that had a significant impact on the model's outcome. We believe it to be indicative of his experience and ability to endure and compete at a high level which can be important in predicting future success.

Overall, our study has shown the model we built to be able to predict rider performance to a reasonable level while also highlighting the factors that are important to a rider's future success. We believe that our model can be used

as a helpful guideline by coaches and riders alike. It can be used alongside their expert knowledge to help make better choices.

Chapter 6

Future works

While we achieved our goal of this thesis, there are still many aspects of road cycling that can be researched which may also lead to the improvement of our models. We believe the effect of the COVID-19 pandemic especially in the 2020 and 2021 seasons to be significant. It could be interesting to investigate the impact it had on the performance of riders in more detail. An approach might be to do a separate analysis of COVID-19's effect on the rider's performance and use the results to adjust the predictions made by the model.

Another study could be done on role players in road cycling like the *domestiques* who have an important role in the team but their performance cannot be captured in the rankings. Future works could try to implement a model that can capture how good a domestique is.

A shortcoming in our model that could be improved in future studies is the addition of physiological features. These features could be useful in improving the performance of our model granted that such data be publicly available on a large scale. Most riders generally do not share their physical data as it could be analysed and their current form could be extracted and leaked to other riders, which could give those other riders a tactical advantage.

A possible future study could be on the impact of external factors like weather and road conditions on the performance of riders. A performance prediction model that could take such external conditions into account would be very useful.

Bibliography

[1] Daam Van Reeth and Daniel J Larson. The economics of professional road cycling. *The Economics of Professional Road Cycling*, 2022.

[2] David Van Bulck, Arthur Vande Weghe, and Dries R. Goossens. Result-based talent identification in road cycling: discovering the next eddy merckx. *Annals of Operations Research*, pages 1 – 18, 2021.

[3] A. D. Karetnikov. Application of data-driven analytics on sport data from a professional bicycle racing team. 2019.

[4] L. Kholkine, Thomas Servotte, Arie-Willem de Leeuw, Tom De Schepper, Peter Hellinckx, Tim Verdonck, and Steven Latré. A learn-to-rank approach for predicting road cycling race outcomes. *Frontiers in Sports and Active Living*, 3, 2021.

[5] Brian Lehman and Visionist. Projecting nfl potential from college career performance curve. 2020.

[6] Daniel P. Berrar, Philippe Lopes, and Werner Dubitzky. Incorporating domain knowledge in machine learning for soccer outcome prediction. *Machine Learning*, 108:97–126, 2018.

[7] Albrecht Zimmermann, Sruthi Moorthy, and Zifan Shi. Predicting college basketball match outcomes using machine learning techniques: some results and lessons learned. *ArXiv*, abs/1310.3607, 2013.

[8] Luca Pappalardo, Paolo Cintia, Paolo Ferragina, Emanuele Massucco, Dino Pedreschi, and Fosca Giannotti. Playerank: Data-driven performance evaluation and player ranking in soccer via a machine learning approach. *ACM Trans. Intell. Syst. Technol.*, 10:59:1–59:27, 2019.

[9] Rabiu Muazu Musa, Anwar P. P. Abdul Majeed, Z. Taha, Mohamad Razali Abdullah, Ahmad Bisyri Husin Musawi Maliki, and Norlaila Azura Kosni. The application of artificial neural network and k-nearest neighbour classification models in the scouting of high-performance archers from a selected fitness and motor skill performance parameters. *Science & Sports*, 2019.

[10] Elnaz Davoodi and Ali Reza Khanteymoori. Horse racing prediction using artificial neural networks. 2010.

[11] Oisín Wiseman. Using machine learning to predict the winning score of professional golf events on the pga tour. 2016.

[12] Daniel Ruiz-Mayo, Estrella Pulido, and G Martınoz. Marathon performance prediction of amateur runners based on training session data. *Proc. Mach. Learn. and Data Min. for Sports Anal*, 2016.

[13] Adam Maszczyk, Robert Roczniok, Miłosz Czuba, Adam Zajαc, Zbigniew Waśkiewicz, Kazimierz Mikołajec, and Arkadiusz Stanula. Application of regression and neural models to predict competitive swimming performance. *Perceptual and Motor Skills*, 114:610 – 626, 2012.

[14] Jean-François Mignot. Strategic behavior in road cycling competitions. *Research Papers in Economics*, pages 207–231, 2016.

[15] Greg Atkinson, R. C. Richard Davison, Asker E. Jeukendrup, and Louis Passfield. Science and cycling: current knowledge and future directions for research. *Journal of Sports Sciences*, 21:767 – 787, 2003.

[16] Alejandro Lucia, Conrad P. Earnest, and Carlos Arribas. The tour de france: a physiological review. *Scandinavian Journal of Medicine & Science in Sports*, 13, 2003.

[17] Yasuyuki Kataoka and Peter Gray. Real-time power performance prediction in tour de france. In *MLSA@PKDD/ECML*, 2018.

[18] L. Kholkine, Tom De Schepper, Tim Verdonck, and Steven Latré. A machine learning approach for road cycling race performance prediction. In *MLSA@PKDD/ECML*, 2020.

[19] Yorck Olaf Schumacher, Roman Mroz, Peter Mueller, Andreas Schmid, and Gerta Ruecker. Success in elite cycling: A prospective and retrospective analysis of race results. *Journal of Sports Sciences*, 24:1149 – 1156, 2006.

[20] Mireille Mostaert, Pieter Vansteenkiste, Johan Pion, Frederik J. A. Deconinck, and Matthieu Lenoir. The importance of performance in youth competitions as an indicator of future success in cycling. *European Journal of Sport Science*, 22:481 – 490, 2021.

[21] Paolo Cintia, Luca Pappalardo, and Dino Pedreschi. "engine matters": A first large scale data driven study on cyclists' performance. *2013 IEEE 13th International Conference on Data Mining Workshops*, pages 147–153, 2013.

[22] Bram Janssens, Matthias Bogaert, and Mathijs Maton. Predicting the next pogaar: a data analytical approach to detect young professional cycling talents. *Annals of Operations Research*, pages 1 – 32, 2022.

[23] Emiel De Spiegeleer. Predicting cycling results using machine learning, 2019.

[24] Jan Hauke and Tomasz M. Kossowski. Comparison of values of pearson's and spearman's correlation coefficients on the same sets of data. 2011.

[25] Anil S. Jadhav, Dhanya Pramod, and Krishnan Ramanathan. Comparison of performance of data imputation methods for numeric dataset. *Applied Artificial Intelligence*, 33:913 – 933, 2019.

[26] Jae-Hyun Seo and Yong-Hyuk Kim. Machine-learning approach to optimize smote ratio in class imbalance dataset for intrusion detection. *Computational Intelligence and Neuroscience*, 2018, 2018.

[27] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, 2003.

[28] David W. Hosmer and Stanley Lemeshow. Applied logistic regression. 1989.

[29] L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2004.

[30] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[31] Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. Catboost: gradient boosting with categorical features support. *ArXiv*, abs/1810.11363, 2018.

[32] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *NIPS*, 2017.

[33] Philipp Probst and Anne-Laure Boulesteix. To tune or not to tune the number of trees in random forest? *ArXiv*, abs/1705.05654, 2017.

[34] Candice Bentéjac, Anna Csörgo, and Gonzalo Martínez-Muñoz. A comparative analysis of xgboost. *ArXiv*, abs/1911.01914, 2019.

[35] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *ArXiv*, abs/1705.07874, 2017.

[36] Ondrej Hubácek, Gustav Sourek, and Filip Zelezný. Learning to predict soccer results from relational data with gradient boosted trees. *Machine Learning*, 108:29–47, 2018.

[37] Tony Duan, Anand Avati, Daisy Yi Ding, Sanjay Basu, A. Ng, and Alejandro Schuler. Ngboost: Natural gradient boosting for probabilistic prediction. *ArXiv*, abs/1910.03225, 2019.