

PROJECT REPORT

INSIGHTS INTO 2022 NRL EVENT DATA



By **SPORTALYTICS** (Team 4)

@ UNSW (Data3001):

- Dylan Upton (z5308844)
- Nikhil Khatri (z5311164)
- Kyle Brown (z5308312)
- Diya Patel (z5311113)
- Saumil Talati (z5309456)

Table of Contents	
MISSION STATEMENT	3
EXECUTIVE SUMMARY	3
BACKGROUND	3
PREVIOUS WORK	4
OBJECTIVES	5
SCOPE	5
PLAN	7
APPROACH	7
• RESEARCH	7
• DATA CLEANING AND EXPLORATORY ANALYSIS	7
• MODELLING	7
• VISUALISATION	7
• COMMUNICATION	8
CHANGES TO INITIAL PLAN	8
DATA ENGINEERING	9
DATA DESCRIPTION	9
OUTLIERS	10
SETS	10
SEQNUMBER	10
VENUEID	11
HALF	11
EVENTCODE	12
XMPHYSICAL	12
OFFICIALS	13
WEATHERCONDITIONID	13
DATA SKEWNESS	14
DATA CLEANING	15
LABEL ENCODING	15
TARGET VARIABLE	16
DATA IMPUTATION	16
FEATURE ENGINEERING	17

CORRELATION	17
STANDARDISATION	17
DATA EXPLORATION	18
DISTRIBUTION OF FEATURES.....	19
TRENDS IN RUCK INFRINGEMENTS	20
MODELLING	27
SVC	27
LOGISTIC.....	27
ADABOOST	27
RANDOMFOREST	27
DECISIONTREE.....	27
MLP	28
XGBOOST	28
MODEL SELECTION.....	28
HYPER PARAMETER TUNING	28
MEASURING MODEL SUCCESS	29
FINDINGS	30
CONCLUSION	31
REFERENCES	31
APPENDIX	32
PEER REVIEW RESPONSE	32
CODE.....	33

MISSION STATEMENT

SPORTALYTICS mission is to provide insights into sporting event data by using leading computational and statistical data science techniques. We strive to create value for our clients by presenting these new insights in a way so they can make informed decisions about future improvements to their sports.

EXECUTIVE SUMMARY

NRL games are won and lost through key moments that alter the teams' momentum as the game goes on. These moments typically are errors, more specifically ruck infringements that are either made as unforced errors or tactical decisions by the defending team.

One of these moments is the tactical ruck infringements which play a massive part of the game as it allows both teams an opportunity to reset their lines and provide a temporary rest break. In doing so, teams can recollect and proceed with their attack in the most meaningful way to either gain distance or to go for further points through a try or a field goal.

The NRL hopes to be more informed on the effect of ruck infringements, specifically, their effect on match outcomes and the distribution of teams committing ruck infringements. It is highly important to determine the significance of how ruck infringements affect the game as this will allow the NRL a greater insight into optimizing the on-field actions of rugby league, allowing greater accessibility to new fans and providing greater entertainment to existing fans. Additionally, the NRL will be able to strengthen the game by considering the needs and abilities of existing teams and players through aspects of rest and performance.

Through classification modelling, the team at **SPORTALYTICS** has showcased the NRL's changes to the ruck infringements rules and how these changes have impacted ruck infringements and the game. We have shown this through predicting, both during half time the final game result, outlining which features (or parts of the game) impact results the most, if the current rule changes have significantly impacted ruck infringements, and if these changes are correlated to winning. Furthermore, considering our findings, we have provided recommendations to the NRL to strengthen Rugby League as a whole for the players and fans alike.

BACKGROUND

The flow of an NRL game is critical for its entertainment level for spectators. Great Rugby League games are those with the most amount of flow which is defined as the least number of stoppages. The ruck area is the most liable to decrease the flow of the game which is what happens around the area surrounding a tackled player. This is then used to recover from the tackle and restructure the team's formation. For this reason, ruck area rules have been changed multiple times in recent years such as any ruck infringement or offside conceded by the defending team within the attacking team's 0-40m zone would be penalized instead of a set restart. This is done to deter ruck infringements by the defending team as a penalty would result in a larger loss of field position or an increase in scoring opportunities. This loss in field position may disrupt the flow of the game and dissatisfy consumers creating a negative impact on the NRL.

Thus, it is important that the NRL are informed of the repercussions of these rule changes, and what this means Rugby League. For this reason, we at **SPORTALYTICS** have established and created this project

and report that aims to explore the 2022 rule change by analyzing and creating conclusions to the provided data to understand how the rule changes to the ruck area in the past couple of years has made an impact on the game. Another aim we at **SPORTALYTICS** have examined is determining how the value of field position over possession has changed in previous years, and what this means for the outcome of games.

PREVIOUS WORK

Due to the rule change being for the current 2022 season, **SPORTALYTICS** understand that no other work has been done on this problem specifically. We do however acknowledge other NRL analysis:

- Factors affecting performance in profession Rugby League - UTS Thesis by Thomas Kempton. Bachelor of Human Movement (<https://opus.lib.uts.edu.au/bitstream/10453/43430/7/02whole.pdf>)
- Conceptualizing Rugby League Performance – Sports Medicine Article by Tannath J. Scott. (<https://sportsmedicine-open.springeropen.com/articles/10.1186/s40798-021-00375-x#citeas>)
- The Rule change affecting time in play that's rarely talked about by League Eye Test (<https://www.rugbyleagueeyetest.com/2020/08/18/nrl-round-14-notes-and-trends/>)

The first resource was used due to its description of the biological and physical factors that affect performance in individual rugby players. This work caught our eye for its mention of well-needed rest which acts as a direct implication of the performance of players within games. For this reason, we were inspired to create a greater focus on the importance of the ruck infringement rules such as the frequency within the scope and objectives of this report. The second resource speaks to the importance of rucks in order to create flow and “affordances” within the game. As a result, we were driven to further understand the changes in ruck occurrences as the rules change, we can then use this to determine the importance of flow as a factor in successfully won games. The final resource acts as an independent study of an analysis of active game time within a match. The final conclusions from this study are that there are fewer stoppages within the game, which allows for more rugby to be played. We would like to investigate this to determine whether the changes in ruck infringements are the reason that these conclusions have been derived.

As the NRL is a private company their data regarding Rugby League is not public. As a result, very little gameplay and performance analysis have been conducted outside of the NRL. We have, however, considered public news articles to gain a better understanding of the Rugby League and the ruck area rules to accommodate the lack of professional work done on this subject matter. These are as follows:

- What is the Ruck in Rugby League? - News article by Alexander Wolf (<https://fluentrugby.com/what-is-the-ruck-in-rugby-league/>)
- Experts view: Who'll benefit from six-again rule – News article by NRL.com (<https://www.nrl.com/news/2020/05/28/experts-view-wholl-benefit-from-six-again-rule/>)
- The top five rule changes that changed the NRL – News article by ZeroTackle (<https://www.zerotackle.com/the-top-five-rule-changes-that-changed-the-nrl-122366/#:~:text=So%2C%20instead%20of%20a%20penalty,to%20do%20with%20the%20ruc>)

All three articles stress either the definition of a ruck or how this has changed over time. These have been very influential in our understanding of the academic previous work and the proposed situation by the NRL. The most notable takeaway from this would be the tactical applications of the ruck which is intentionally used to slow down the play to realign the teams structurally and for their composure.

Overall, the analysis of the previously conducted work was very influential in determining the scope of our objectives as we were able to refine our objectives to become more focused on the importance, occurrence and use of the Ruck Infringement rules that were enforced by the NRL over the years.

OBJECTIVES

Our objectives are:

- Outline and understand detailed patterns, trends, and the effect of ruck infringement rule changes through visualization and the detailed extraction of data findings. Examining how this affects the game through distribution with respect to field position and how this might influence the notion of field position versus possession.
- Produce a model, using the most relevant aspects of Rugby League in predicting if team A will win at half time in the game. What this will aim to achieve is to highlight the most important features of the game that affects a team's chances of winning. “Blow Out” wins were also investigated under this objective.
- Outline a recommended course of action to the NRL regarding ruck infringements and making games more competitive and engaging, thus improving the overall NRL product.

SCOPE

The main deliverables of this project are a report and associated presentation that will outline the team's analysis of the proposed NRL request. It has been our goal to present our findings concisely so they can be understood by both technical and non-technical individuals. The report and presentation have been developed based on the following data provided by the NRL:

- **CSV Event Data Files for 2019, 2020, 2021 and 2022 NRL Seasons**
 - Each NRL game event is captured by a third party and described in a single row in CSV format.
- **Event Data Readme File**
 - Documentation on variables in the captured CSV files in addition to an explanation on the field position calculation process and zones. Examples are also provided.
- **AAReadMe file**
 - Explanation of every column name and variable used including examples of how some columns work
- **Match CSV Data**
 - Data Captured specifically on matches

From the NRL provided data, our analysis has provided a strong focus on the ruck area events which detail the type of event, the position of these on the field and the subsequent events such as if a try was made, that can help us draw conclusions from the data. These events included:

- **EventCodes**, specifically
 - A ruck infringement occurring (Code = RINS/RIND)

- Penalty (Code = PABD)
- A try being scored (Code = TRY)
- The end of a team's set (Code = EC1S, EC2S, EC3S, EC4S, EC5S)
- The end of a team's possession (Code = ENPO)
- A tackle being completed (Code = REFT)
- **Qualifier/ QualiferName**
 - Each of the eight qualifiers give a linear description of the event. Most importantly qualifer1 outlines "Ruck Infringement", the area we are focusing on.
- Xm\YmPlayer, TeamAScores/ TeamBScores, SeqNumber, SeasonId, Set/Tackle and MatchId.

As part of our analysis, we needed to make certain assumptions. As seen below:

- How we chose to analyse the data files, where we will draw on our knowledge of the domain and then escalate any uncertainties to our NRL contact. Since we are comparing the rule changes and their effect over the years, we will assume that certain statistics and trends changed over the seasons are only related to ruck rule changes, as we identify this as our independent variable.
- COVID19 and its effect on the data. Post pandemic, NRL games were only affected off-field with players testing positive and unable to play games, but when teams took to the field, games were no different to previous years. The success of **SPORTALYTICS** project will be assessed by a series of factors:
- Another assumption has been consideration of sufficient evidence addressing the scope of our objectives and if so, were the findings significant.

We expected each of the model's performance tests to return high results, splitting the data into a training and testing set to properly explore the success of our model. Given the scope and importance of the project at hand, we at **SPORTALYTICS** have considered issues and established solutions that could interfere or hinder the progress of satisfying the elected aims. Issues that may affect the project were:

- Unfinished project due to a large scope managed by use of a Gantt chart to plan and follow strict deadlines.
- Uneven and unfair workload within the team causing the loss of expertise in certain areas managed by use of a Gantt chart with assigned roles and deliverables which are required to progress to later stages of the project.
- Lack of understanding of rugby specific knowledge leading to errors in data analysis and modelling countered by individuals with NRL knowledge to do tasks which require domain knowledge.
- Difficulty of communication due to Covid-19 restrictions whereby we have planned frequent meetings to allow for proper team communication and easier means of teamwork.
- Realism of the initially set goals and whether it can be feasibly achieved in the time span that was given to us. This can be countered by deeply monitoring the progress of the investigation and adjusting this where necessary.

PLAN

APPROACH

Our approach was structured with each member completing assigned tasks in our given timeframe. All team members came together weekly to make changes and progress the report. Teamwork was critical to support each member with specific tasks, allocated to an individual's experience and skills. Our approach was broken down into the following 5 phases:

- **RESEARCH** Collectively the team decided on the problem. This allowed us to identify if we would be predicting through statistical modelling or to only extrapolate findings from the data. Research was then required into the domain (NRL Rugby League), specifically ruck infringement rule changes. Dylan and Nikhil would lead the work on this section. Dylan is an avid NRL fan with extensive domain knowledge. Nikhil is a long-term Maroons fan who has experienced years of NRL culture.

- **DATA CLEANING AND EXPLORATORY ANALYSIS** This stage involved structuring the data in an accessible manner allowing for visualization and modelling of the project. Using exploratory data analysis, we created histograms or similar techniques to determine the relevancy of certain variables for future use in modelling and our visualization process focused on different objectives. We would then export event data from 2019 to the present to our local systems. By choosing Python, we enabled the use of SQL and any relevant libraries to help to filter out irrelevant data. Early plans were to consider using the likes of GitHub or alternatives to ensure version control and collaboration. Dylan and Kyle were specifically chosen for this task through their recent expertise and proficiency in Python and associated manipulation of databases having completed a recent subject successfully. Furthermore, they both have external working experience in this area where they could draw inspiration from.

- **MODELLING** Being one of our primary objectives, we planned to create a mathematical model using the determined relevant variables from the data cleaning stage, predicting win percentages based on differences in field position and possession. From the planning stage, we saw that this may involve a single model that is applicable to every year and game regardless of the rule changes to ruck infringements to create a wider assessment of the importance of these changes. Using our exploratory data analysis, we would then select a model to represent our data which would then be trained and built up through further analysis. This would be tested at the half-way point of the existing games to determine correctness with the actual result of these historic games. Diya, having studied a mathematical major in her time at UNSW is highly skilled in mathematical modelling. Similarly, Saumil has professional experience in modelling which would assist in further building up the model. Finally, Nikhil could inspire a more complex model through his personal experience of machine learning.

- **VISUALISATION** Visualization of the data to determine the impact of ruck infringement changes over the period of 2019-2022 was critical. This was achieved by using tools such as Tableau and the Python library MatLib to create many different models including: heatmaps and breakdowns of ruck infringements with respect to field position. Furthermore, using the algorithm created in the modelling stage, we could look to create graphs and tables to determine the rankings of individual players and teams. This will also be completed by the duo of Diya and Saumil who have both worked on the creation of dashboards and visual data representations.

• **COMMUNICATION** Led by Dylan, all group members would contribute to the creation of the report by adding their insights from the understanding each has gained from their respective roles in the project. The proposal would be created using Microsoft Word with our presentation to be created using Microsoft PowerPoint.

Project Schedule

By: Sportalytics

Project Start Date
12th Sep 2022

Week	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11																											
Date	12 Sep 2022	19 Sep 2022	26 Sep 2022	3 Oct 2022	10 Oct 2022	17 Oct 2022	24 Oct 2022	31 Oct 2022	7 Nov 2022	14 Nov 2022	21 Nov 2022																											
TASK	<table border="1"> <thead> <tr> <th>START</th> <th>END</th> <th>DAYS</th> </tr> </thead> <tbody> <tr> <td>Project Plan</td> <td></td> <td></td> <td></td> </tr> <tr> <td>1 Project Formulation</td> <td>Wed 9/14/22</td> <td>Wed 10/12/22</td> <td>21</td> </tr> <tr> <td>2 Exploratory Data Analysis</td> <td>Wed 9/21/22</td> <td>Tue 11/01/22</td> <td>30</td> </tr> <tr> <td>3 Modelling</td> <td>Wed 10/12/22</td> <td>Tue 11/01/22</td> <td>15</td> </tr> <tr> <td>4 Visualisation</td> <td>Wed 10/19/22</td> <td>Tue 11/15/22</td> <td>20</td> </tr> <tr> <td>5 Communication</td> <td>Wed 11/02/22</td> <td>Wed 11/23/22</td> <td>16</td> </tr> </tbody> </table>											START	END	DAYS	Project Plan				1 Project Formulation	Wed 9/14/22	Wed 10/12/22	21	2 Exploratory Data Analysis	Wed 9/21/22	Tue 11/01/22	30	3 Modelling	Wed 10/12/22	Tue 11/01/22	15	4 Visualisation	Wed 10/19/22	Tue 11/15/22	20	5 Communication	Wed 11/02/22	Wed 11/23/22	16
START	END	DAYS																																				
Project Plan																																						
1 Project Formulation	Wed 9/14/22	Wed 10/12/22	21																																			
2 Exploratory Data Analysis	Wed 9/21/22	Tue 11/01/22	30																																			
3 Modelling	Wed 10/12/22	Tue 11/01/22	15																																			
4 Visualisation	Wed 10/19/22	Tue 11/15/22	20																																			
5 Communication	Wed 11/02/22	Wed 11/23/22	16																																			

Project Plan

1 Project Formulation Wed 9/14/22 Wed 10/12/22 21

2 Exploratory Data Analysis Wed 9/21/22 Tue 11/01/22 30

3 Modelling Wed 10/12/22 Tue 11/01/22 15

4 Visualisation Wed 10/19/22 Tue 11/15/22 20

5 Communication Wed 11/02/22 Wed 11/23/22 16

Project Plan Due 5th Oct

Draft Group Report 2nd Nov

Peer Review Drafts 9th Nov

Group Presentation 16th Nov

Group Report 23rd Nov

As time went on, we discovered that our original plan was not feasible and could be optimized to better fit time and technology constraints. As a result, the team had mutually agreed to change our objectives making the change from predicting match score for blowouts wins to a classification task of team A winning or not. This change was significant as we decided to change the model used to a classification model which ended up changing some of our prior processes such as exploratory data analysis and visualizations to determine new variables that may be relevant to our new objectives.

8

DATA ENGINEERING

DATA DESCRIPTION

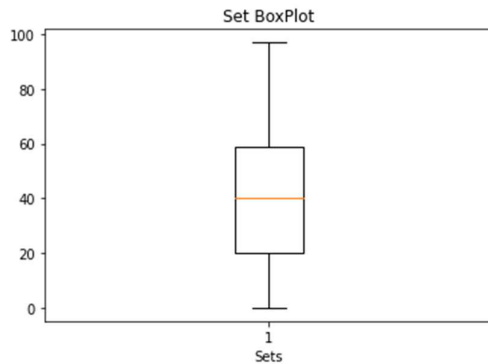
- The NRL provided **SPORTALYTICS** with two csv files named 2020-2022 Event Data Set, and 2019 Event Data. Each file had the same covariates, only the 2019 file has columns with slightly different names which need to be made the same as the 2020- 2022 file in the data cleaning process.
- The files contain identification covariates:
 - MatchId- Identification for each match
 - SeasonId- Year of each match
 - CludId/ InPossessionClubId
 - OppositionId- Identification for the opposition team
 - PlayerId/ InPossessionPlayerId
 - OfficialId- Identification for the officials
- Contain Event Description covariates:
 - Set- Counter for set number
 - Tackle- Counter for number of tackles in set
 - EventCode/ EventName- describes the type of event
 - QualifersName- outlines what happened in event
 - Qualifiers- outlines what happened in event
 - SeqNumber- linear continuation time covaraite
- In game statistics covariates:
 - Score/ OppScore- current game score
 - XmPlayer/ YmPlayer- field position relative to player involved
 - XmPhysical/ YmPhysical- field position as coordinate
 - XmPossession/ Possession- field position relative to team
 - Zones- split field into 84 zones of 10x10 size
 - Include the graphic we have been provided
 - Channels- Horizontal rows down field that indicate distance to ends of field
 - Include the graphic we have been provided
 - Sections- vertical rows across the field, with codes
 - (S)ideline
 - (L)eft
 - (N)umbers (L)eft
 - (L)eft, (C)entre
 - (R)ight
 - (N)umbers (R)ight
 - (S)ideline (R)ight
 - PossessionSecs- how long possession has lasted
 - TeamAScore/ TeamBScore- end of game scoreline
- The most significant data is located within:
 - The distance measures such as XmPlayer/YmPlayer
 - The different events for each row of data described in EventCode and EventName
 - The identification for each team and what team has possession of the ball
 - The score total and points covariate
- To make sense of the data, sorting needs to occur.
 - We must sort the data on MatchId, SeqNumber and Half. This enables us to separate the matches, then order each match by SeqNumber which is the data's time variable and by half, since for each half of the match the SeqNumber resets.

OUTLIERS

The dataset is densely populated with identification covariates and hence outliers are nearly improbable. Some specific covariates we needed to check over to ensure future correctness were as follows:

SETS

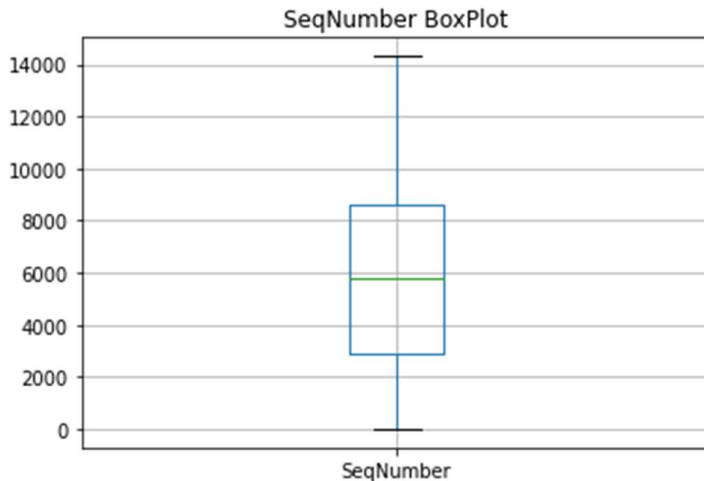
Figure 2: Boxplot of set counts in the dataset



Some of the games, the set count goes upward towards 100 which seemed unreasonable at first but these figures as highly likely to occur due to the continuous nature of the data. This is affirmed by boxplot below.

SEQNUMBER

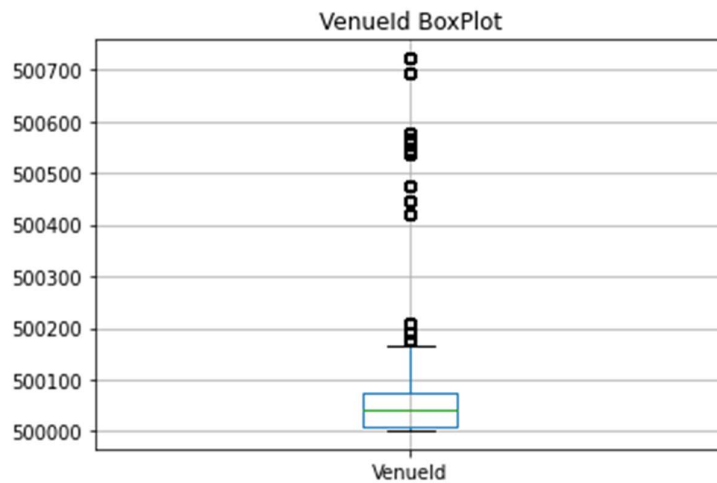
Figure 3: Boxplot of "SeqNumber" column in the dataset



This covariate is the primary continuous numerical time covariate that enables the sorting of events during the game into chronological order. From the boxplot, we can see that this number reaches as high as 14000. This seems illogical as an 80-minute game could not possibly have 14000 possession changes. But in fact, applying a simple query (`csv[csv['SeqNumber'] > 13000]`), we find there is a series of entries outputs, indicating there is not an outlier but rather a continuous stream of events until this number.

VENUEID

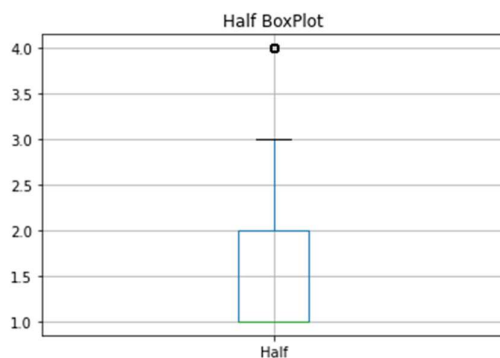
Figure 4: Boxplot of "VenueId" column in the dataset



Though the boxplot indicates outliers, this covariate and all other similar covariates are identification values. As a result, covariates similar as these have negligible influence on the dataset so we cannot classify them as outliers.

HALF

Figure 5: Boxplot of "Half" column in the dataset



Initially, **SPORTALYTICS** were confused on halves going to 3 and 4, but quickly realized overtime games don't reset the half count, but in fact continue to add to it hence allowing the half count to be > 2 .

EVENTCODE

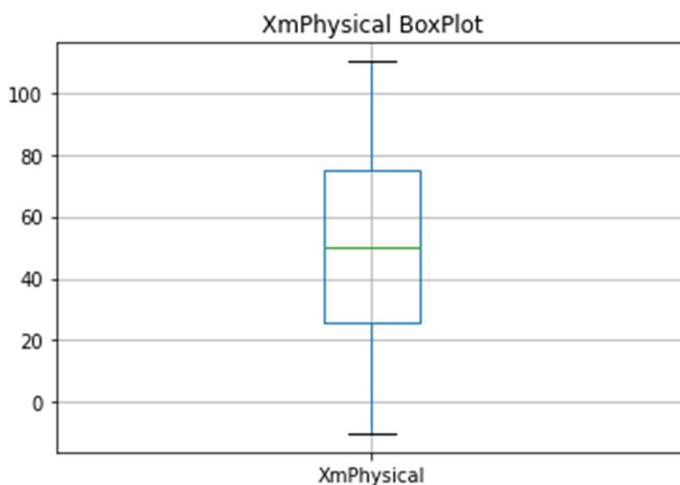
Figure 6: Counts of most frequently occurring event codes

```
Output exceeds the size limit. Open the full output data in a text editor
MACD      154548
MBCD      144753
RCVS      138385
RCVD      138077
TKES      123335
MKRS      93649
MKRD      93649
MACS      57176
MBCS      54632
TKED      54525
REFT      53549
OADS      49275
OADO      49275
PBND      43846
PBNS      43846
PDLD      43357
PDLS      43344
HTPS      26407
HTPD      26407
RSPD      22243
RSPS      22243
RUND      21275
RUNS      21275
ENPO      20258
STPO      20258
...
TRY8       1
P1SS       1
SCHW       1
STPD       1
Name: EventCode, dtype: int64
```

Firstly, **SPORTALYTICS** removed all event codes for the initiation of a match since this data was not potent and useful for our problem at hand. Since there were 192 unique event codes it is inefficient to graph the distribution and outliers of all 192 codes. Hence a sample has been taken. We can see at the tail there could possibly be some outliers but since we are not worried about anything other than ruck infringements (codes: RINS and PABD), all other codes need not be considered. Noted there are 1404 PABD and 930 RINS across all the data.

XMPHYSICAL

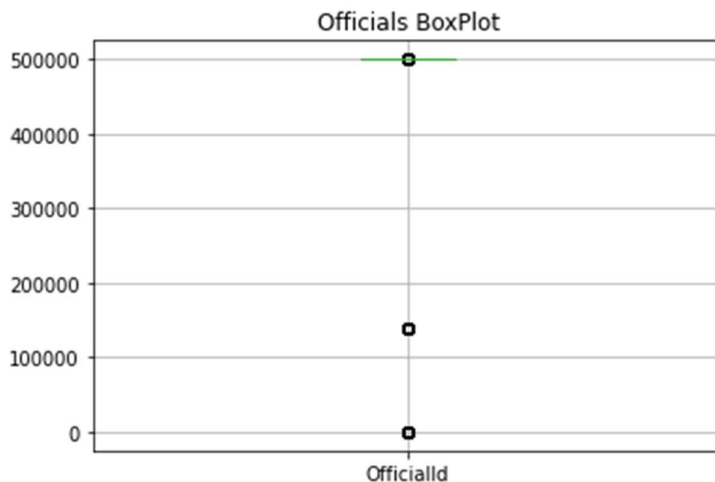
Figure 7: Boxplot of “XmPhysical” column in the dataset



Firstly, this covariate was misinforming, as how could there be negative position? **SPORTALYTICS** realized though through using a query that we start at the 0 m position at the goal line as the baseline for XmPhysical, so from the goal line to the dead ball line is up to 10 metres (allowing -10 metres to be accepted) and on the other side of the field, going from 100 to 110 metres.

OFFICIALS

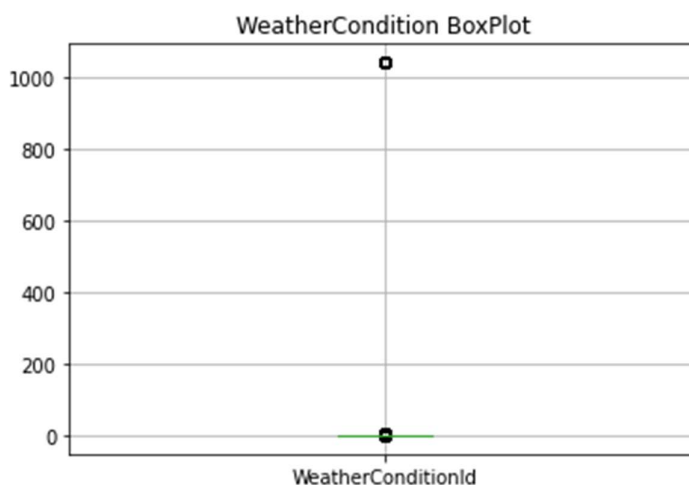
Figure 8: Boxplot of “Officials” column in the dataset



Officials are an important aspect of the game, as it is widely discussed if officials bias one team over another. Thus, we look at the distribution of officials for all matches. But it is seen that only three officials are recorded and hence there would be insufficient evidence to suggest bias to certain teams may or may not occur.

WEATHERCONDITIONID

Figure 9: Boxplot of “WeatherCondition” column in the dataset



Weather conditions seem to have an outlier at 1040 based on the boxplot result below. After specifically querying, only 5 rows were found at the start of one specific match. This could be a specific weather condition not known, but nonetheless these rows can be removed as since they were at the start of the match, they hold no significance to ruck infringements or modelling match outcome.

DATA SKEWNESS

Figure 10: Skewness of the features in the dataset

1	MatchId	0.150511
2	SeqNumber	0.042251
3	SeasonId	0.150507
4	SeriesId	0.000000
5	RoundId	-0.073158
6	VenueId	1.812683
7	WeatherConditionId	-0.735069
8	PlayerId	6.805726
9	Jumper	0.130095
10	PositionId	-0.317360
11	Half	0.088721
12	ElapsedMins	0.082142
13	ElapsedSecs	0.017501
14	ElapsedMillisecs	0.081081
15	GameMins	0.021171
16	GameSecs	0.005238
17	Set	0.047283
18	Tackle	0.158037
19	DurationSecs	0.250924
20	DistanceMs	3.508201
21	Points	-0.106494
22	Score	1.124359
23	OppScore	1.124125
24	XmPhysical	-0.011494
25	YmPhysical	-0.003446
26	ZonePhysical	-0.018700
27	XmPlayer	-0.034806
28	YmPlayer	-0.040031
29	ZonePlayer	-0.045572
30	XmPossession	0.131010
31	YmPossession	0.059594
32	ZonePossession	0.124297
33	PossessionSecs	0.120413
34	OppPossessionSecs	0.120289
35	TotalPossessionSecs	0.059480
36	OfficialId	-3.177888
37	0_20_ElapsedSecs	0.221678
38	20_Half_ElapsedSecs	0.190975
39	20_Try_ElapsedSecs	0.458984
40	TeamAScore	0.614332
41	TeamBScore	0.954000

Due to the nature of the data as a time series with majority of the columns as identification covariates, skewness cannot exist and hence does not need to be understood. Further as seen above the rest of the data doesn't have any significant skewness to warrant any feature engineering. We can assume this is due to the time series nature of the data that naturally, skewness doesn't arise.

DATA CLEANING

SPORTALYTICS worked on a solution to change the names of columns in the 2019 dataset that had closely similar column names to those in the 2020-2022 dataset. We then dropped columns not in one of the csv's compared to other so we can concatenate the two csv's successfully.

SPORTALYTICS then looked quickly into duplicate rows as it is possible that this time series dataset has duplicate entries. Though none were found.

Before diverging into the distributions of covariates, **SPORTALYTICS** decided to first drop any columns that are non-valuable, to reduce computational stress and further clean the dataset.

The following columns as stated have been removed:

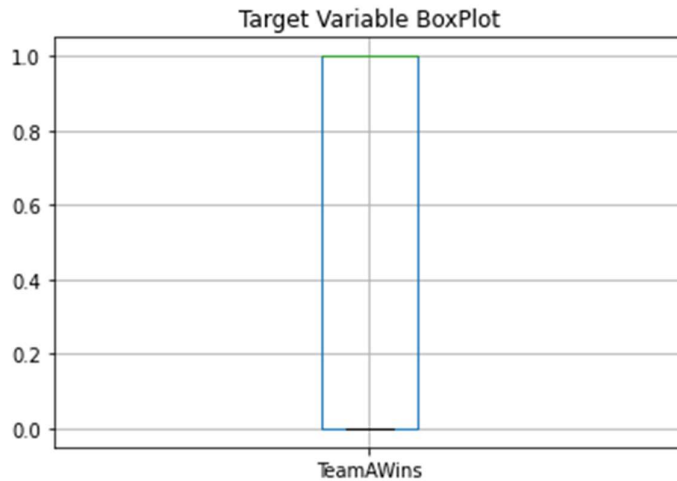
- ElapsedSecs/mins/millisecs
 - All physical field position related covariates. We are only using player so we can clearly separate the two teams and their field position, regarding distance from their own try line. Enables easier extraction of data around distance.
- MatchYear/day/month/hour/minute
 - Redundant information as we have other columns depicting the same information i.e.. SeasonID and MatchID
- SeasonName as this data is already captured in SeasonId
- Seriesname
- Roundname
- matchNumber
- OfficialId
 - OfficialId will not be considered due to only three officials being recorded as stated. Thus, it is not reasonable to attempt to correlate officials to match outcome.
- These columns could have been useful but due to their significantly high null values, the data is made useless and hence removed
 - Jumper
 - Run on
 - Captain
 - PositionId

LABEL ENCODING

SPORTALYTICS used label encoding on TeamIds, to make TeamAId and TeamBId features in our model by converting each individual distinct id into its own integer. Hence allowing a correlation between teams and match outcome to be made.

TARGET VARIABLE

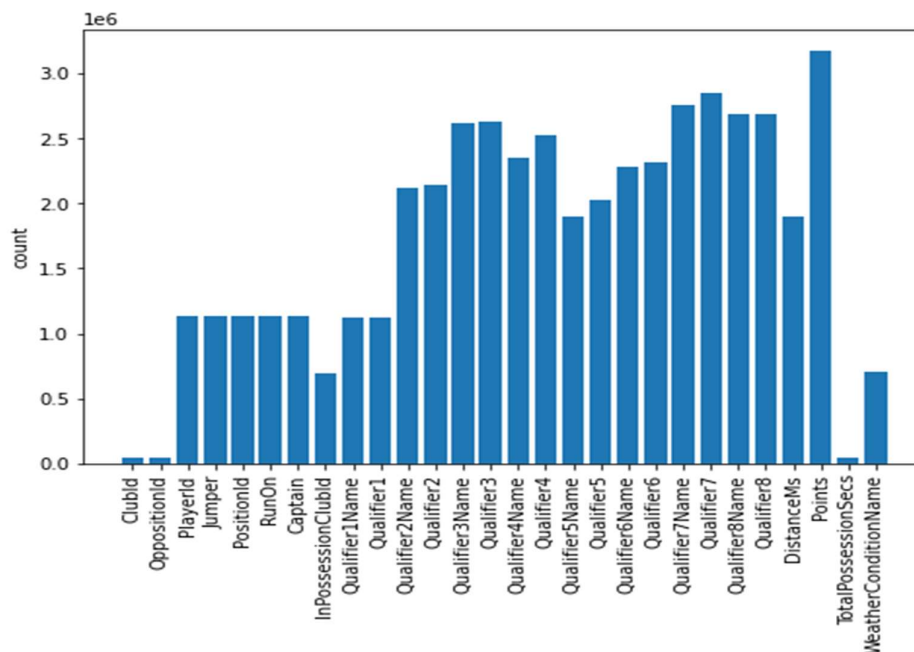
Figure 11: Boxplot of our engineered target variable “TeamAWins”



NRL games can have 3 distinct outcomes: Team A wins, Team B wins and a draw. Since we can only work with a binary 1,0 outcome. We have decided that our target variable will be if Team A wins, denoted by 1. And for Team B wins and a draw, we denote this as a 0. This is clearly defined and hence no skewness or outliers will occur.

DATA IMPUTATION

Figure 12: Bar chart null values count for covariates



The nature of the data at hand records values that are not able to be estimated if said value is null. Hence all data imputation will be done for applicable covariates, setting their null values to zero. These covariates include:

- Duration
- Points
- Score and oppScore
- PossesstionSecs/ OppPossesstionSecs/ TotalPossessionSecs

For the remaining nulls present in covariates, we simply don't need to worry about these covariates as we will not be considering them as features in our model. These features will be discussed later.

FEATURE ENGINEERING

To improve model accuracy, we performed feature engineering to create new columns based on the half time score, distance and possession seconds covariates. We chose to do this as finding the difference between the team's results works as a better overall measure to predict final game outcome. Each new column was determined using the following manipulations:

- $\text{ScoreMargin} = \text{ScoreAHalfTime} - \text{ScoreBHalfTime}$
- $\text{DurationSec} = \text{How long average ruck lasted for Team A}$
- $\text{DistMargin} = \text{TeamADistHalf} - \text{TeamBDistHalf}$
- $\text{SecsMargin} = \text{TeamASecsHalf} - \text{TeamBSecsHalf}$

We intentionally decided to use these features in so that predicting full-time match outcomes with half-time data simply due to the overly complex data at hand. Overall, the data provided is very given with each match having roughly 10,000 rows. Thus, it was imperative to use group by function to extract meaning. As a team, we decided to use half-time data to predict full-time for a game as instead, using full-time data to predict other games induces too many errors. This is critically obvious as each game simply has a significantly different amount of data.

CORRELATION

Figure 13: Correlation heatmap of target variable and possible predictors

	TeamAId	TeamBId	ScoreMargin	SecsMargin	DistMargin	DurationSecs	Venueld	WeatherConditionId	TeamAWins
TeamAId	1.000000	-0.067116	0.029767	0.133024	0.107630	0.140866	0.117272	-0.025935	0.033999
TeamBId	-0.067116	1.000000	-0.062441	-0.083217	0.103759	-0.043738	0.028858	0.033105	-0.037207
ScoreMargin	0.029767	-0.062441	1.000000	0.635858	-0.407596	0.035382	-0.074096	-0.049489	0.658899
SecsMargin	0.133024	-0.083217	0.635858	1.000000	-0.313357	0.049045	-0.081458	-0.005991	0.355617
DistMargin	0.107630	0.103759	-0.407596	-0.313357	1.000000	0.058315	-0.009621	0.111521	-0.330497
DurationSecs	0.140866	-0.043738	0.035382	0.049045	0.058315	1.000000	0.083305	-0.103138	0.124315
Venueld	0.117272	0.028858	-0.074096	-0.081458	-0.009621	0.083305	1.000000	-0.112487	-0.009600
WeatherConditionId	-0.025935	0.033105	-0.049489	-0.005991	0.111521	-0.103138	-0.112487	1.000000	0.016761
TeamAWins	0.033999	-0.037207	0.658899	0.355617	-0.330497	0.124315	-0.009600	0.016761	1.000000

Due to the nature of the data at hand there is no significant correlation between predictors as seen in the correlation matrix above. Hence, no feature engineering in combining columns was required, and the covariates seen above are our final predictors and response variable for modelling.

STANDARDISATION

Using SKlearn's standard scalar library, **SPORTALYTICS** standardized the data to bring all features to a common scale, hence improving predictability for our models.

DATA EXPLORATION

Figure 14: Pairwise plot for our target and possible predictor variables

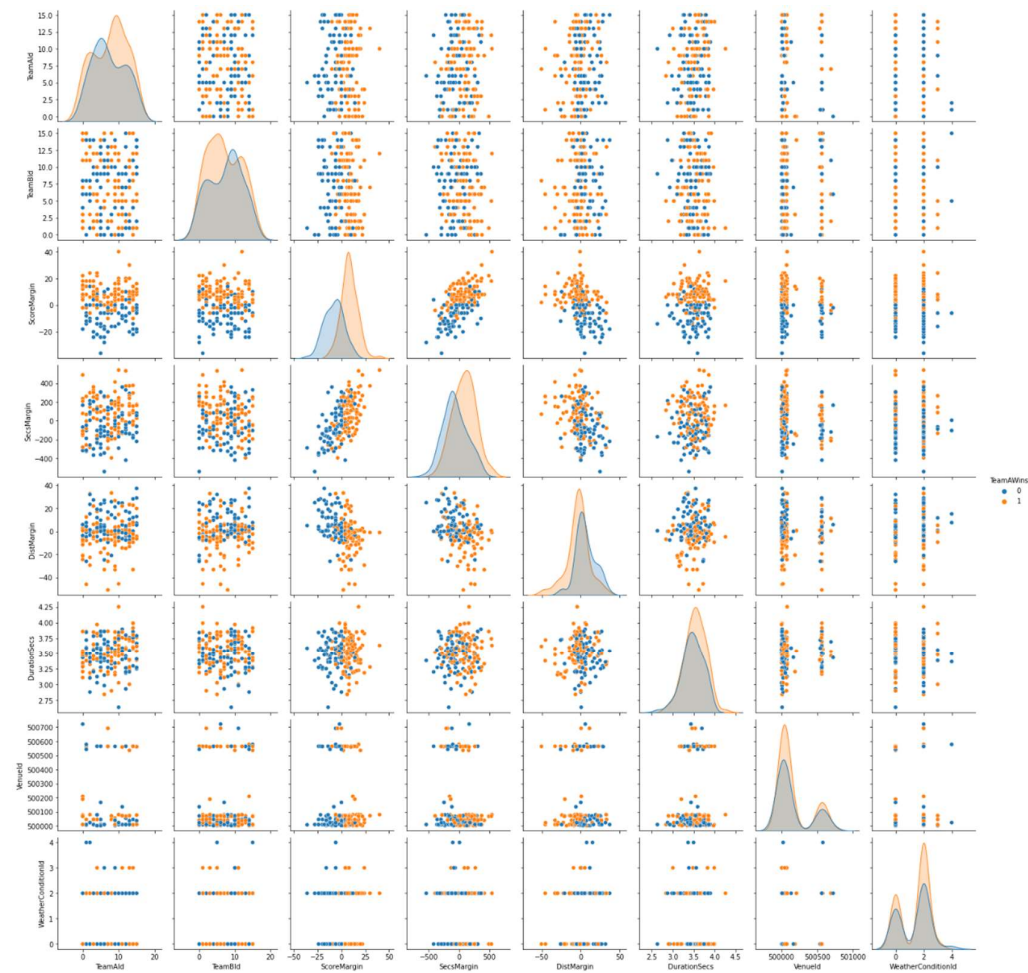


Figure 15: The distributions of target and possible predictor variables

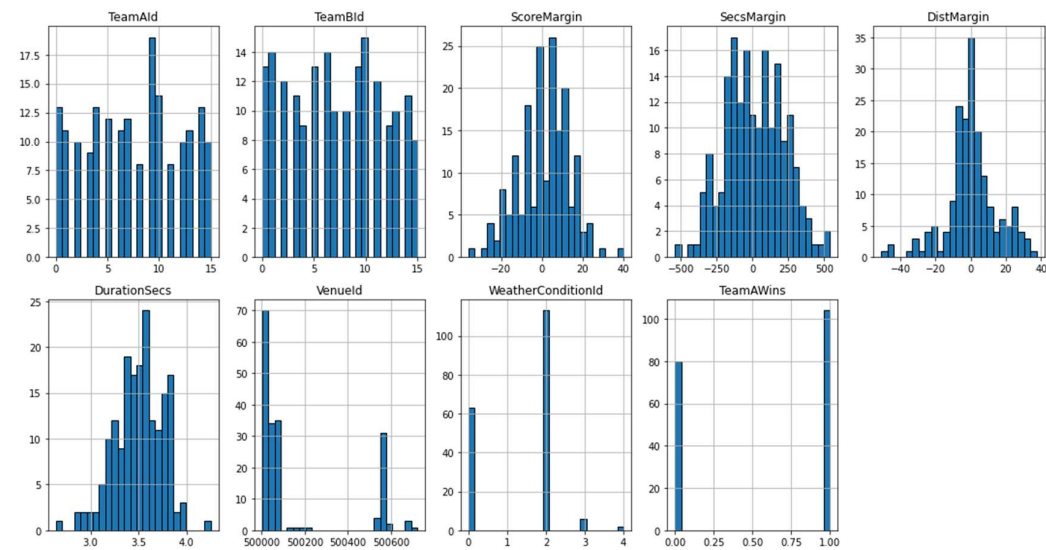
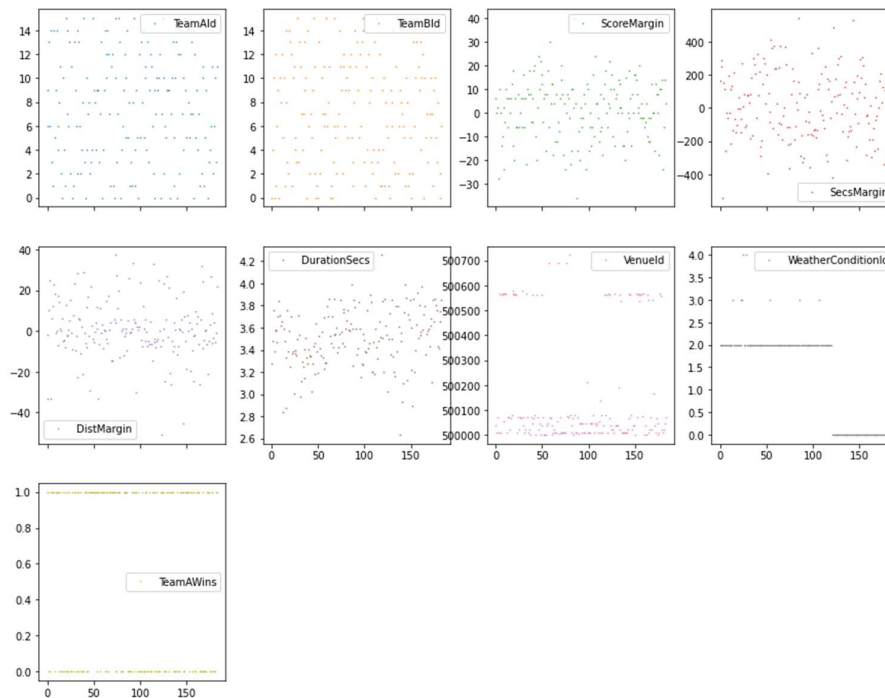


Figure 16: Distribution of features including target variable



DISTRIBUTION OF FEATURES

TeamAId and TeamBId are both evenly distributed.

ScoreMargin

Normally distributed which indicates most games have a lower score margin at half time. Though there is a respectable frequency of 20 margin difference games, presenting the idea that there is still many games that are blow outs, even at half time.

SecsMargin

Still normally distributed similar to scoreMargin, but a lot wider in distribution. Meaning there is more times where the secsMargin (difference in average time of possession) is a value further away from zero. This indicates that games tend to be generally more one sided in possession time, but this doesn't translate necessarily to a larger difference in score margin.

DistMargin

Intuitively, distMargin is much narrower in normal distribution. Indicating the average distance per possession margin is a lot closer to zero. This is confirmed later in figure 1 by the average field position being centre field. What does this mean? This indicates that NRL games no matter the score will tend to be played evenly on both sides, but this doesn't take into account if one team is in fact looking to score on one side of the field and the other is always kicking to try force the attacking team away from the defending teams try line.

DurationSecs

This feature captures the average ruck time for team A for each match. With this feature we are hoping to identify if the winning teams are taking longer on defense to let the attacking team play the ball as they wish to have longer to set their defensive line.

VenueId

Although we were unable to extract home field as this would have been an excellent feature to determine home field advantage, we still will use venueId as a feature to identify if certain fields may pose an advantage to select teams, hence indicating this as a significant feature in our model.

WeatherConditionId

The majority of weather conditions seem to be condition id 2, which one can assume is fine weather. This feature should be significantly correlated as weather always plays an important role in NRL matches.

TeamAWins

Finally, our target variable teamAWins, as expected is a binary 0,1 response variable. One interesting note is there is a greater number of wins for team A than losses or draws.

TRENDS IN RUCK INFRINGEMENTS

In this section we set out to identify and visualize trends in the position of ruck infringements in the years 2019-2022 and analyze the reasons behind them. The end goal of this analysis is to be able to make recommendations on which ruck infringement rules should be the best for the future of the game.

We first needed to define what ruck infringements are in our dataset. For all years (2019,2020,2021,2022) we used the same search conditions for ruck infringements:

CODE	QUALIFIERS
RINS	Ruck Infringement 2nd effort Crowding Flop Hand in face Hand on ball Holding down Leg pull Lying in ruck Markers not square Other Slow peel Working on ground
PABD	Markers not square Markers split 2nd effort Flop Hand on ball Holding down Lying in ruck Slow peel Working on ground

Ruck Infringement rules over the years 2019-2022:

- **2019:** All ruck infringements resulted in penalties
- **2020** - Replacing a penalty with a set restart for a ruck infringement anywhere on the field
- **2021-** Replacing a penalty with a set restart for any player deemed to be offside from play of the ball (PTB)
- **2022** - Any ruck infringement or offside conceded by the defensive team within the attacking team's 0-40m zone was penalised instead of a set restart. Any ruck infringement or offside conceded by the defensive team outside of the attacking team's 0-40m zone was given a set restart

The Event Code RINS is the ruck infringement code. However, as described above ruck infringements always resulted in penalties in 2019, and sometimes resulted in penalties in 2021 and 2022. Therefore, the Event Code PABD is also used in our exploration.

*Note that below when we refer to ‘the defending team’s goal line’ we mean the goal line they defend, not the goal line they score at.

Figure 17: Scatter plot of all ruck infringements coloured by year, with a line for the 0-40m zone (2022 rules) superimposed. Also displayed are the mean coordinates for each year. (For this plot and all that follow: the defending team’s 0-40m zone is left of screen, defending team’s goal line is right of screen)

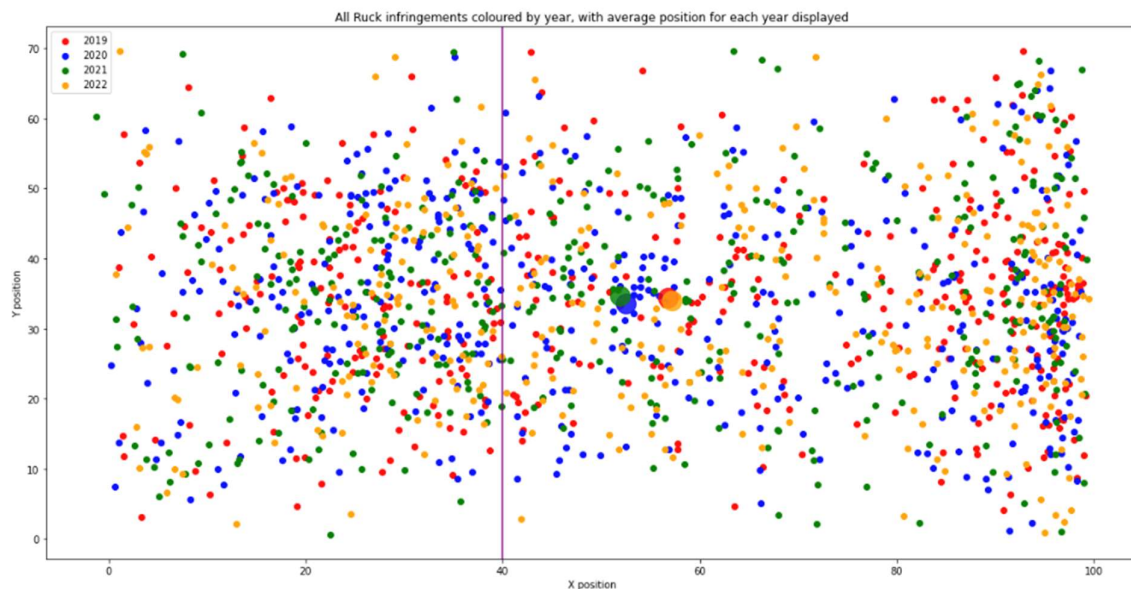
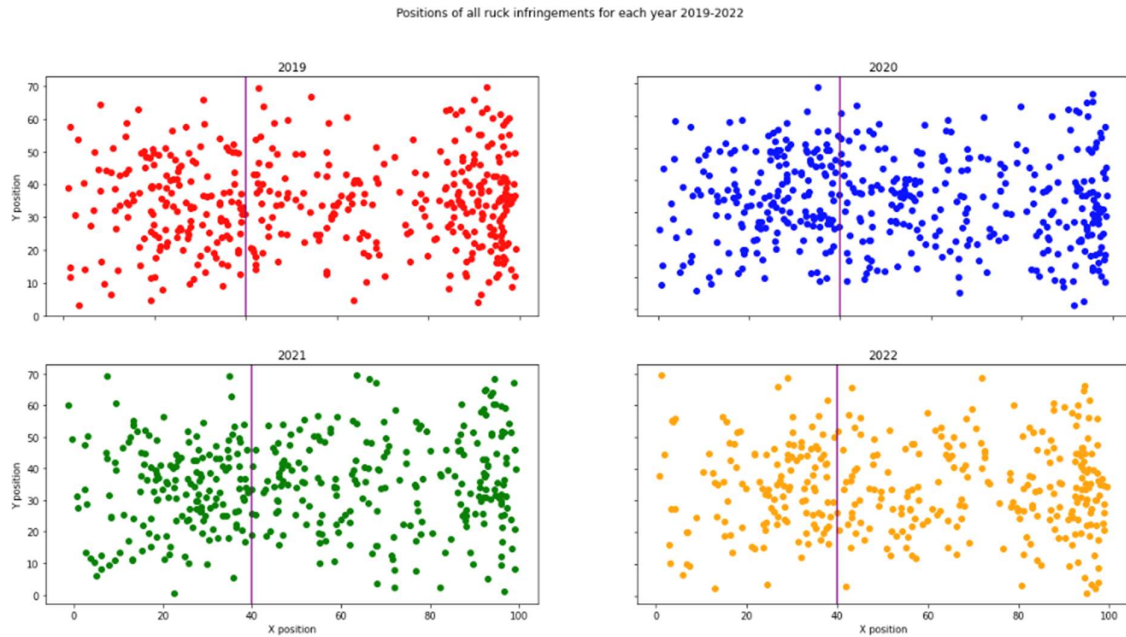
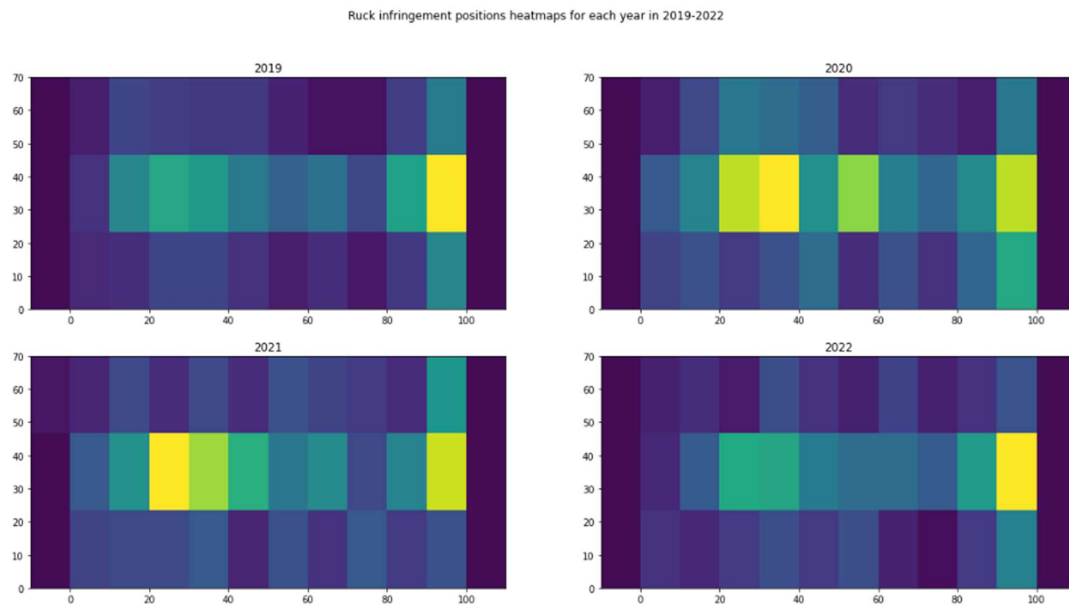


Figure 18: Individual scatterplots for each year's infringements, with colours matching the joint plot, and the 0-40m zone line again superimposed.



From figure 17, while there is a clear distinction between the average ruck infringement position from 2019 and 2022, compared to 2020 and 2021, it isn't very significant. Furthermore, neither figure 17 or figure 18 indicate clear differences in the distribution of infringements. Nor do they show any visible trend around the 0-40m zone to the left of the superimposed line (penalised differently over the 4 years). However these scatter plots may simply be inadequate to identify trends that do exist, so we next create 2d histograms or heatmaps to hopefully see some clearer trends.

Figure 19: Heatmaps illustrating the frequencies of ruck infringements across the field for each year.



The heatmaps in figure 19 suggest some clear parallels in the relative frequencies of ruck infringements, with the 2019 and 2022 heatmaps looking almost identical, and the 2020 and 2021 heatmaps also quite similar. In particular, the effect of the 2020 and 2021 0-40m ruck infringements rules (a set restart, rather than a penalty) are evident, with more of the infringements occurring in the 0-40m section of the field in these years, particularly between 20 and 40m. There also appears to be a higher relative frequency of ruck infringements close to the goal line in 2019 and 2022, compared to the other years.

While the above heatmaps seem to suggest the effects of rule changes, particularly ruck infringements in the 0-40m zone, the relative amounts of tackles occurring in each section of the field for each year could also be the main cause of the apparent trends. It is also important to consider not only the relative frequencies as shown so far, but also absolute numbers. We will make sure to analyse these possibilities in the next section.

Figure 20: Heatmaps illustrating the frequencies of tackles across the field for each year.

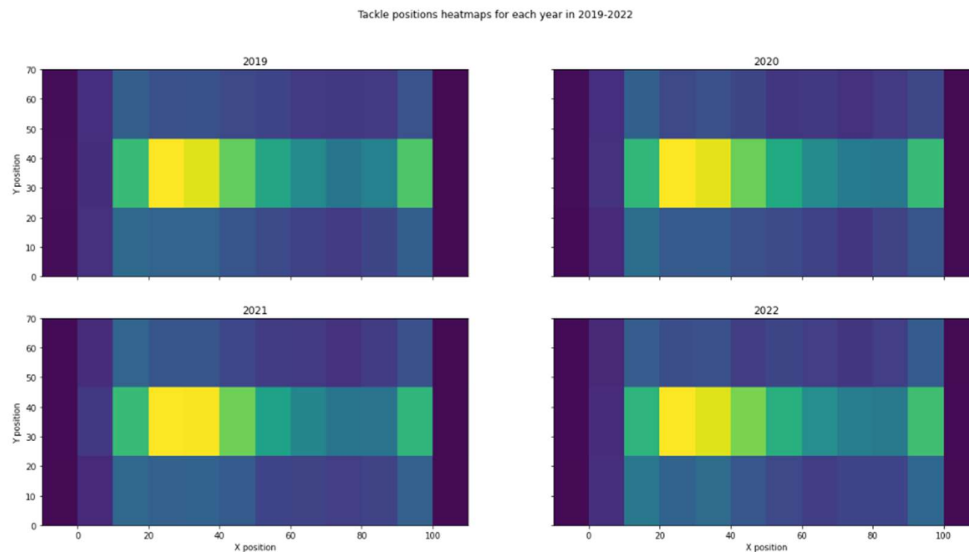


Figure 21: Tabulated data regarding yearly ruck infringements in specific sections of the field of play. Included below are the number of tackles in the same sections to see if the relative frequencies of tackles are responsible for any trends in ruck infringements.

Ruck Infringements						
	Total	Within 20m of goal	Within 20m %	0-40m zone	0-40m zone %	
SeasonId						
2019	425	151	35.53	228	53.65	
2020	440	111	25.23	271	61.59	
2021	399	98	24.56	241	60.40	
2022	358	118	32.96	193	53.91	

Tackles						
	Total	Within 20m of goal	Within 20m %	0-40m zone	0-40m zone %	
SeasonId						
2019	28178	5508	19.55	12702	45.08	
2020	16802	3123	18.59	7620	45.35	
2021	18886	3532	18.70	8702	46.08	
2022	17861	3485	19.51	7921	44.35	

Figure 22: Data from figure 21, using averages per game rather than season totals

Ruck Infringements per game						
	Total	Within 20m of goal	Within 20m %	0-40m zone	0-40m zone %	
SeasonId						
2019	4.25	1.51	35.53	2.28	53.65	
2020	7.86	1.98	25.23	4.84	61.59	
2021	6.05	1.48	24.56	3.65	60.40	
2022	5.68	1.87	32.96	3.06	53.91	

Tackles per game						
	Total	Within 20m of goal	Within 20m %	0-40m zone	0-40m zone %	
SeasonId						
2019	281.78	55.08	19.55	127.02	45.08	
2020	300.04	55.77	18.59	136.07	45.35	
2021	286.15	53.52	18.70	131.85	46.08	
2022	283.51	55.32	19.51	125.73	44.35	

Firstly, regarding the question of if changing tackle occurrences could be confounding ruck infringement trends, all the data in figures 20, 21 and 22 clearly show stable tackles data for all four years. The heatmaps in figure 20 are almost identical, meaning that the distribution of tackles across the field has not changed in the four years in question.

As can be seen, the first of the major two heatmap findings as also supported by the above tables. Namely, that the 2021 and 2022 rule changes led to significantly more infringements occurring far away from a team's defensive goal line with 4.84 and 3.65 ruck infringements in the 0-40m zone per game in 2020 and 2021 respectively, compared to 2.28 and 3.06 in 2019 and 2022 respectively. This supports the idea that players and/or teams do respond to the way ruck infringements are penalised, with players more willing to give away set restarts far away from their defensive goal line compared to penalties.

However, the second major heatmap trend discussed previously can be seen to have been primarily a result of the higher overall infringement counts in 2020 and 2021. As seen in figure 22, infringements within 20m of goal are roughly the same across the 4 years, contrary to what the heatmap may have suggested about more occurrences in 2019 and 2022.

Figure 23: Average number of Ruck infringements per game.

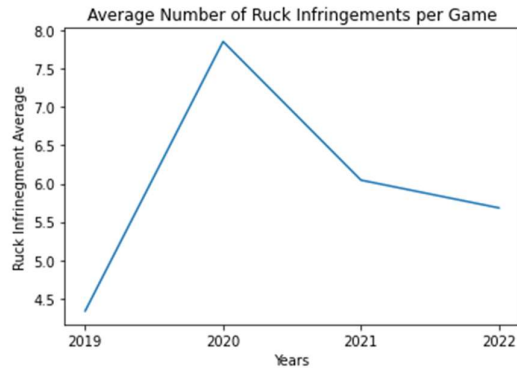
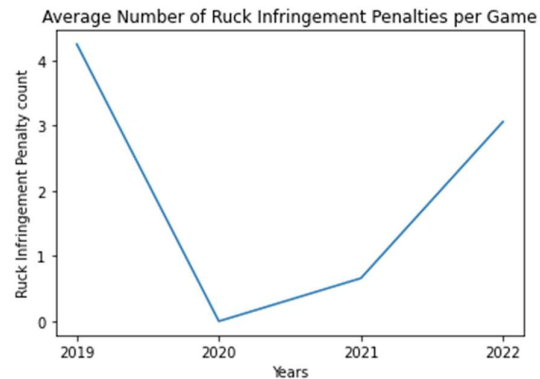


Figure 24: Average ruck infringement penalties per game



In terms of which year's rules are best, the key aspect of a good set of rules is a balance between keeping infringements low, ie. deterrence, while simultaneously reducing the number of game slowing penalties, ie. game speed. As clearly shown in figure 22 and figure 24, 2019 has the least ruck infringements per game of 4.25. However, they are all penalties. 2020 has no penalties but almost double the number of total infringements (7.86). 2021 has fewer infringements than 2020, but as shown in figure 25 below, a small proportion (11%) of these are penalties. Finally, 2022 has more total infringements than 2019 with less penalties, and less total infringements than 2020 and 2021 but more penalties. The pros and cons of each year's rules can be best visualized by the simple pair of graphs, figure 23 and figure 24.

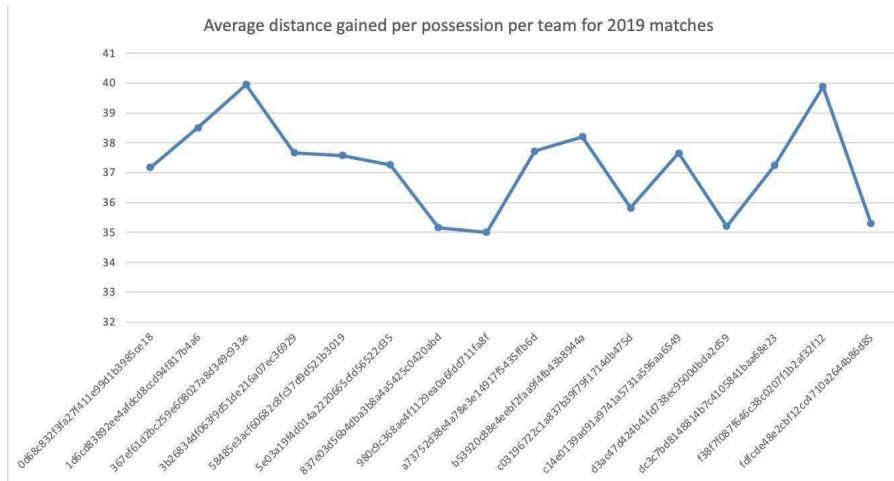
Figure 25: Data for the numbers of each penalty type given in 2021

2021 Ruck infringements penalty type counts

EventCode	Count	Percentage
RINS	355	88.97
PABD	44	11.03

As demonstrated above, the different rules for the 4 years essentially make a sliding scale of balancing infringement deterrence and game speed, with 2019 at one end with the fewest total infringements but most penalties, followed by 2022, 2021 and finally 2020 with the most total infringements but no penalties. As far as which rules are the best for the game, it really depends on how much the NRL values the two factors of total infringements and penalties. However, we will still make a recommendation of our own later in this report.

Figure 26: Graph of average distance gained per possession for all teams in 2019



For the graph above, we were able to identify and separate each possession during all the 2019 matches. This allowed us to calculate the distance covered during each possession. We then grouped the data by the Team Id allowing us to calculate the average distance covered by each team in 2019 matches.

MODELLING

SVC

SVC is a classification model type that maps data points to a higher dimension then finds the hyperplane that divides the data into its respective classes. SVC has advantages of, it works well with a well-defined disassociation between classes and is more effective in higher dimensions. But doesn't produce accurate results with overlapping classes, particularly with large datasets.

LOGISTIC

Logistic regression models the probabilities for classification problems with two possible outcomes (binary). Making the model easy to train, implement and interpret with good accuracy for simple data, whilst being fast at classifying any unknown records. On the contrary, logistic regression overfits when the number of observations is less than the number of features. Further, logistic regression is restricted in only being able to construct linear boundaries.

ADABOOST

AdaBoost uses an iterative approach to learn from mistakes of weak classifiers to turn them into strong ones, in essence learning from its mistakes. This means AdaBoost is less prone to overfitting as the input parameters are not jointly optimized and the accuracy of weak classifiers can be improved significantly. On the other hand, Adaboost needs a quality dataset. Meaning, noisy data and outliers must be avoided as they can have a severely negative impact on model accuracy and results.

RANDOMFOREST

RandomForest builds regression trees on different samples and takes their majority vote for classification. Simply, a more accurate model than decision tree at the cost of more computational and time usage. RandomForest produces good predictions that are easily understood, as well as handles large datasets well. But in doing so, requires large amount of computational power which takes large amounts of time to compute.

DECISIONTREE

A sequence of queries done adaptively so the outcome of previous tests can influence the test performed next. Meaning it has easily explainable results, and requires less preparation in pre-processing:

- Doesn't need data to be normalised, scaled
- Missing data doesn't effect model significantly

While if there is a small change in the data, this causes massive change in the trees structure and can significantly increase time to train the model.

MLP

Multilayer Perceptrons (MLPs) are feedforward artificial neural network models that are defined as a directed graph from input to output nodes. MLP has advantages of being easily used with non-linear data, efficient with large datasets and very quick predictions. Contrastingly, MLP has disadvantages of being very time consuming in training and the quality of model depends on training. Hence MLP can be severely limited by its training.

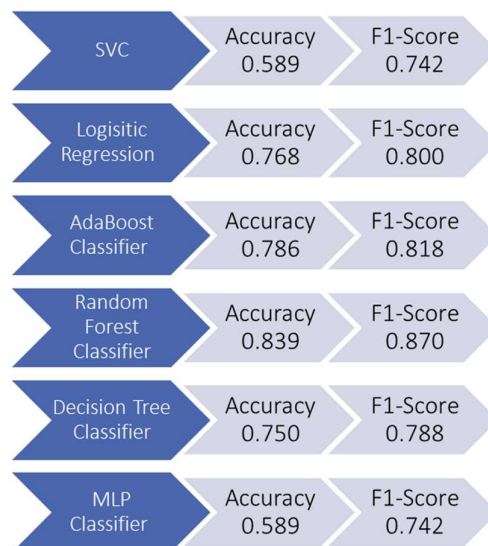
XGBOOST

XGBoost extends AdaBoost, but uses regularized boosting (L1 and L2) to prevent overfitting. This gives advantages that it can handle missing values, incorporates various features such as parallelization and distributed computing, and makes use of sophisticated hyper-parameter tuning. But in doing so becomes extremely sensitive to outliers.

MODEL SELECTION

Each model will be compared against each other based off of accuracy and f1-score. Given the best model chosen, only this model will be trained with hyper parameter tuning to extract the best results.

Figure 29: Diagram of models and associated accuracy and f1-scores



SVC	Accuracy 0.589	F1-Score 0.742
Logistic Regression	Accuracy 0.768	F1-Score 0.800
AdaBoost Classifier	Accuracy 0.786	F1-Score 0.818
Random Forest Classifier	Accuracy 0.839	F1-Score 0.870
Decision Tree Classifier	Accuracy 0.750	F1-Score 0.788
MLP Classifier	Accuracy 0.589	F1-Score 0.742

HYPER PARAMETER TUNING

With random forest being selected as our best model. We can hyperparameter tune the model. Given a range of 0 to 150 estimators, we firstly based off of f1 score linearly loop through each of the estimator numbers and append the f1 score to each amount. Proceeding to then apply a grid search using the f1 score, we are able to find the number of estimators that give the best accuracy.

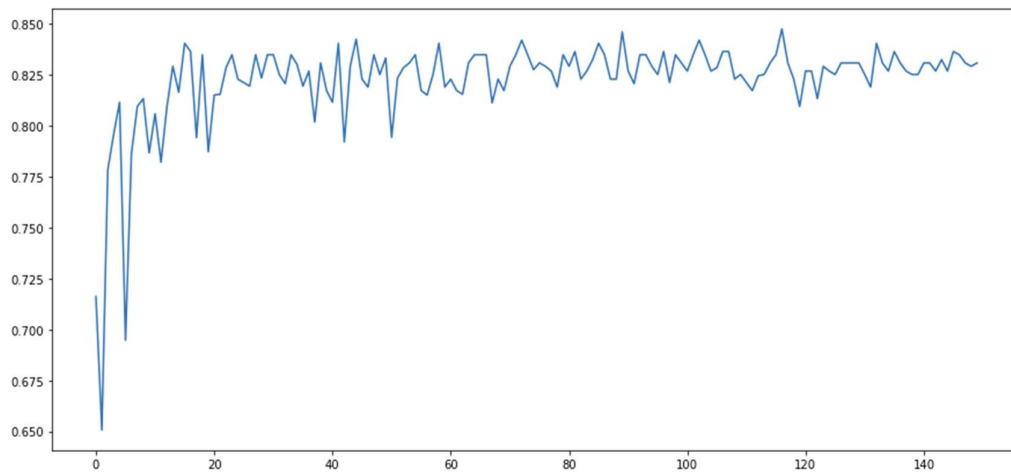
- Number of estimators: 160
- Accuracy: 0.842201

Again using the same grid search technique, we instead find the best number of features and the associated accuracy score.

- Maximum number of features: 6
- Best accuracy: 0.833483

Resulting in a final graph of number of estimators against accuracy score.

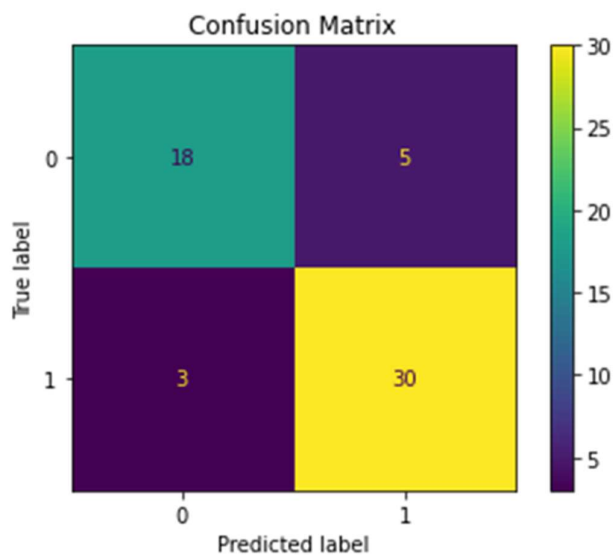
Figure 28: Graph of the accuracy score vs the number of estimators in the model



MEASURING MODEL SUCCESS

Finally, to measure the success of our model, we chose to use mean squared error to value our model's prediction accuracy. We were able to extract a score of 0.1429, which is a very low score.

Figure 29: Correlation matrix indicating predictions



FINDINGS

Firstly, in relation to ruck infringement exploration. As explained previously there is no objectively superior set of rules for ruck infringements in the years 2019-2022, with the best rules essentially dependent on how much deterrence of committing infringements is valued relative to maintaining a fast game speed. However, we believe that the 2021 rules are the best balance of reducing both total infringements and having as few penalties as possible. This is due to our belief that the 2019 and 2020 rules are both too extreme, with far too many penalties in 2019 and far too many total infringements in 2020. Between 2021 and 2022, there are slightly fewer total infringements in 2022 (5.68 per game vs 6.05 in 2021). However, the penalties per game is considerably higher in 2022 (approximately three per game in 2022 vs less than one in 2021). Hence why we believe that the 2021 rules provide the best balance of deterring infringements whilst maintaining game speed.

Figures 18, 19 and 20 confirm that even with the new rule changes, the value of positions vs possession has not been significantly affected. Which is what was intended by the NRL. Though if the value was wanted to be changed, placing more consequences on ruck infringements could be effective. The better teams generally are in an advantageous position, to score a try or go for a field kick more often. This is connected to ruck infringements as not seen in data, the better defending teams will use ruck infringements to allow more time for their team to set their defense. As they feel they can stop any scoring attempt and regain possession. The worse defending teams will use kicks to force the ball away from the try line, when they regain possession after a successful defense, which may skew average possession distance battles. Meaning on the surface these teams seem to be better at defending, but they are not. The better teams will be attacking more often than defending, meaning that shorter kicks are used more, and less field position would be gained as teams tend to kick to the goal line to score, instead of using larger kicks to force the attacking team further away from their try line.

From the model and the correlation graph, we can determine that not all features are relevant and effective in predicting final match result. From our modelling hyper parameter tuning, only 6 features were used. Referencing the correlation matrix, we can identify that these features were: TeamAId, TeamBId, ScoreMargin, SecsMargin, DistMargin, DurationSecs. Most notably, we can identify scoreMargin as the most prominent feature, due to the high correlation between itself and final match outcome. The model shows as though ruck infringement may not be significant, however, it is useful for stopping scoring momentum of the opposition to set the defense, which again is not captured in the data.

Our Random Forest model sufficiently predicted final match outcomes with a MSE of 0.1429 and this success is further proven by the correlation matrix in figure 28, with minimal incorrect labels. Hence, we can say our model effectively completes our objective to predict final match outcomes.

In the future if **SPORTALYTICS** were to “tackle” this project again, we would look at incorporating more covariates in our final model. Whether that may be looking into what happens after a ruck infringement or the events leading up to a ruck infringement or extracting more data from events that occur before and after ruck infringement events occur.

CONCLUSION

The 2021 rules create the best balance of deterrence and maintaining game speed compared to previous years. However, the best ruck infringement rules are criteria dependent. Ruck infringements are still prevalent but highly lessened over the years especially before half-time, so ruck infringement rules have been influential, but also slow the overall flow of the game as there are many more penalties called.

Could NRL implement a penalty count system? Could a certain player only be able to commit a certain number of penalties before needing to be removed from the game, like basketball fouls. Could there be a team penalty count and after a certain number is reached a free kick for goal is given to the other team.

Ruck infringements are still one area that slow down the flow of the game, especially if they are penalized. In creating a greater punishment for committing them could we see a further reduction in their occurrences, or will the added penalties further slow the game down?

REFERENCES

- Kempton, T. (2016) *Factors affecting performance in professional Rugby League*. UTS. Available at: <https://opus.lib.uts.edu.au/bitstream/10453/43430/7/02whole.pdf>
- Scott, T.J. *et al.* (2021) *Conceptualising rugby league performance within an Ecological Dynamics Framework: Providing direction for player preparation and development - sports medicine - open*, SpringerOpen. Springer International Publishing. Available at: <https://sportsmedicine-open.springeropen.com/articles/10.1186/s40798-021-00375-x#citeas>
- Test, L.E. (2021) *The rule change affecting time in play that's rarely talked about - NRL round 14 2020 stats and Trends*, *The Rugby League Eye Test*. Available at: <https://www.rugbyleagueeyetest.com/2020/08/18/nrl-round-14-notes-and-trends/>
- NRL.com. (2020) *Experts view: Who'll benefit from six-again rule*, *National Rugby League*. NRL. Available at: <https://www.nrl.com/news/2020/05/28/experts-view-wholl-benefit-from-six-again-rule/>
- Zero Tackle (2022) *The top five rule changes that changed the NRL*, *Zero Tackle*. Available at: <https://www.zerotackle.com/the-top-five-rule-changes-that-changed-the-nrl-122366/#:~:text=So%2C%20instead%20of%20a%20penalty,to%20do%20with%20the%20ruck>
- Wolf, A. (2021) *What is the ruck in rugby league?* *FluentRugby*. FluentRugby. Available at: <https://fluentrugby.com/what-is-the-ruck-in-rugby-league/>

APPENDIX

PEER REVIEW RESPONSE

In response to our peer's review:

“We did notice there is limited evidence on hyperparameter choices for each algorithm”

- In response to this we had not yet begun model selection and hyper parameter tuning when the draft report was handed in, this was later added.

“How you will calculate the accuracy or performance of each model”

- Similar to hyperparameter tuning, we had not included any indication of model selection as we had not completed it at the current version of the report. This was later added.

“It would be great to also see more references and conversion of dot points to paragraphs as the final report fleshes out”

- This is a main concern of our team and something we worked religiously towards changing.

“In addition, it would be better that captions are stated clearly under each data visualization chart, table and graph, with clear descriptions for each figure in composite sets”

- Thank you for this suggestion, the relevant changes will be made accordingly.

“A point to improve is to expand the ‘Previous Work’ section and give a brief summary of what the resources you have listed talk about and how they contribute to the work you are doing”

- This has been noted and looked at. Particularly how each previous work point has helped contribute to our work.

“Secondly, your results are difficult to follow in relation to scope and objectives, and your objective of ‘outlining trends in the data such as the number of blow out wins’ does not coincide with your data cleaning where you converted your target variable into a binary value”

- This was our initial objective but due to the problem at hand we changed our final objectives and had overlooked making those changes to the objectives, this has been now changed.

“Lastly, we suggest your team expand on the theory behind each model type and make sure you label your figures correctly for ease of the reader”

- Figures have been labelled from now, but we have chosen to not pursue expanding on the theory of each model as our report is about the findings and conclusions, we can draw not what model is best or engaging in detailed data science modelling communications.

CODE

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from scipy import stats
import seaborn as sns
from scipy.stats import norm, skew
from sklearn.preprocessing import LabelEncoder
from matplotlib.pyplot import figure
import scipy.ndimage as sp

import math
from sklearn.model_selection import GridSearchCV
from sklearn.svm import OneClassSVM
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score
from sklearn.model_selection import cross_val_score

from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPClassifier

from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error

csv = pd.read_csv('/Users/dillu/Downloads/2020-2022 Event Data Set (2).csv')
csv1 = pd.read_csv('/Users/dillu/Downloads/2019 Event Data.csv')

csv2 = csv1.rename({'Team.A.id': 'TeamAId', 'Team.A.Name': 'TeamAName', 'Team.B.ID': 'TeamBId',
'Team.B.Name': 'TeamBName', 'Club.Id': 'ClubId', 'Club.Name': 'ClubName', 'Opposition.Id':
'OppositionId', 'Player.Id': 'PlayerId'}, axis=1)
csv2 = csv2.rename({'InPossessionClub.Id': 'InPossessionClubId', 'InPossession.Player.Id':
'InPossessionPlayerId', 'Unnamed: 0': 'Half_20_ElapsedSecs', 'X0_20_ElapsedSecs': '0_20_ElapsedSecs'},
axis=1)
csv2 = csv2.rename({'X20_Half_ElapsedSecs': '20_Half_ElapsedSecs', 'X20_Try_ElapsedSecs':
'20_Try_ElapsedSecs'}, axis=1)

csv2 = csv2.drop(columns = 'Half_20_ElapsedSecs')

#drop columns in csv2 that arent in csv
csv2 = csv2.drop([col for col in csv2.columns if col not in csv.columns and col in csv2.columns], axis=1)

#drop columns in csv that arent in csv2
csv = csv.drop([col for col in csv.columns if col not in csv2.columns and col in csv.columns], axis=1)

#Ordering csv2 columns same as csv
csv2 = csv2[csv.columns]

csv_merged = pd.concat([csv, csv2])

csv_merged = csv_merged.sort_values(by=['MatchId', 'SeqNumber'])
```

```

#outliers

plt.boxplot(csv_merged['Set'])
plt.title("Set BoxPlot")
plt.xlabel('Sets')

csv_merged.boxplot(column=['SeqNumber'])
plt.title("SeqNumber BoxPlot")

csv_merged.boxplot(column=['VenueId'])
plt.title("VenueId BoxPlot")
#looks like there is outliers but this is an id covariate and thus outliers are not a thing with this.....

csv_merged.boxplot(column=['PlayerId'])
#same as venueId

csv_merged.boxplot(column=['Half'])
plt.title("Half BoxPlot")
#after looking into halves, we find that when there is extra time the halves count goes beyond 2 -> add
code to how we know this

value = csv['EventCode'].value_counts()
value
rucks = csv_merged[(csv_merged['EventCode'] == 'RINS') | (csv_merged['EventCode'] == 'PABD')]
value = rucks['EventCode'].value_counts()
value

csv_merged.boxplot(column=['XmPhysical'])
plt.title("XmPhysical BoxPlot")
#first though how could there be negative position, but realised through using above query that we start
the 0 m position at the goalline
#, so from the goalline to the deadball line is upto -10 metres and on the other side of the field it goes
from 100 to 110

#officals
csv_merged.boxplot(column=['OfficialId'])
plt.title("Officials BoxPlot")

csv_merged.boxplot(column=['PossessionSecs'])

csv_merged.boxplot(column=['WeatherConditionId'])
plt.title("WeatherCondition BoxPlot")
csv_merged['WeatherConditionId'].unique()
#wet = csv_merged[csv_merged['WeatherConditionId'] == 1040]
#1040 is a outlier and thus is removed.....
csv_merged = csv_merged[csv_merged['WeatherConditionId'] != 1040]

#skewness
csv_merged.skew(axis = 0, skipna = True)

#target variable
#team a wins
csv_dropping = csv_merged.copy()

```

```

csv_dropping['TeamAWins'] = 0
csv_dropping['TeamAWins'].loc[csv_dropping['TeamAScore'] > csv_dropping['TeamBScore']] = 1
target = csv_dropping[['MatchId', 'TeamAWins']]
target = target.groupby(['MatchId']).max()
#what about draws

#target variable.....
target.boxplot(column=['TeamAWins'])
plt.title("Target Variable BoxPlot")

temp = csv_dropping.isnull().sum()
temp2 = pd.DataFrame(temp, columns = ['Count'])
temp2 = temp2.reset_index()

data = temp2[temp2['Count'] > 0]

# create figure
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
# set Y axis label
ax.set_ylabel('count')
# set orientation for X axis labels
plt.xticks(rotation='vertical')

ax.bar(data['index'], data['Count'])
plt.show()

#missing data
pd.set_option('display.max_rows', None)
csv_dropping.isnull().sum()* 100 / len(csv_dropping)

#can get rid of set 0 since no one has possession of ball then
csv_dropping = csv_dropping[csv_dropping['Set'] != 0]

csv_dropping['DurationSecs']=csv_dropping['DurationSecs'].fillna(0)
csv_dropping['Score']=csv_dropping['Score'].fillna(0)
csv_dropping['OppScore']=csv_dropping['OppScore'].fillna(0)
csv_dropping['PossessionSecs']=csv_dropping['PossessionSecs'].fillna(0)
csv_dropping['OppPossessionSecs']=csv_dropping['OppPossessionSecs'].fillna(0)
csv_dropping['TotalPossessionSecs']=csv_dropping['TotalPossessionSecs'].fillna(0)

#remove small number of rows where we dont know what player has ball
csv_dropping = csv_dropping[csv_dropping['InPossessionPlayerId'].isnull() == False]

#search for dups, remove if any found
csv_dropping[csv_dropping.duplicated(subset=['MatchId', 'SeqNumber', 'SeasonId']) == True]

csv_dropping.isnull().sum()* 100 / len(csv_dropping)

#play the ball duration average
#PBFS PTB - fast
#PBNS PTB - neutral
#PBSS PTB - slow
ruckDur = csv_dropping[(csv_dropping['EventCode'] == "PBFS") | (csv_dropping['EventCode'] ==
"PBNS") | (csv_dropping['EventCode'] == "PBSS")]

```

```

ruckDur = ruckDur[['MatchId', 'DurationSecs']]
duration = ruckDur.groupby(['MatchId']).mean()
duration.head()

#HALFTIME POSSESSION SECS
test_HT = csv_dropping[['MatchId', 'Half', 'SeqNumber', 'TeamAId', 'TeamBId', 'InPossessionClubId',
'PossessionSecs', 'OppPossessionSecs']]
print(test_HT.isnull().sum())
test_HT = test_HT[test_HT['Half'] == 1]
test_HT = test_HT[test_HT['PossessionSecs'] != 0]
test_HT = test_HT[~test_HT['PossessionSecs'].isnull()]
#match = [19111015, 21111016]
#test_HT = test_HT[test_HT['MatchId'].isin(match)]
#test_HT = test_HT[test_HT['MatchId'] == 20111016]
test_HT['teamA_secs_half'] = 0
test_HT['teamB_secs_half'] = 0
test_HT.reset_index(drop=True, inplace=True)
idx = test_HT.groupby(['MatchId'])['SeqNumber'].transform(max) == test_HT['SeqNumber']
test_HT = test_HT.loc[test_HT.groupby(['MatchId'])['SeqNumber'].idxmax()]
for i, row in test_HT.iterrows():
    if row['InPossessionClubId'] == row['TeamAId']:
        test_HT.at[i, "teamA_secs_half"] = row['PossessionSecs']
        test_HT.at[i, "teamB_secs_half"] = row['OppPossessionSecs']
    elif row['InPossessionClubId'] == row['TeamBId']:
        test_HT.at[i, "teamB_secs_half"] = row['PossessionSecs']
        test_HT.at[i, "teamA_secs_half"] = row['OppPossessionSecs']
test_HT.reset_index(drop=True, inplace=True)
#print(test_HT[['MatchId', 'SeqNumber', 'TeamAId', 'teamA', 'TeamBId', 'teamB']])
test_HT = test_HT[['MatchId', 'TeamAId', 'TeamBId', 'teamA_secs_half', 'teamB_secs_half']]
pd.set_option('display.max_rows', 284)
print(test_HT)

#HALFTIME DIST AVG 201....
poss1_HT = csv_dropping[['MatchId', 'Half', 'TeamAId', 'TeamBId', 'SeqNumber', 'EventCode',
'EventName', 'InPossessionClubId', 'XmPossession']]
matches1 =
[20111016,20111023,20111026,20111041,20111044,20111047,20111051,20111053,20111058,20111063,20
111076,20111093,20111095,20111096,20111097,20111098,20111101,20111102,20111105,20111106,20111
107,20111108,20111111,20111112,20111113,20111115,20111117,20111121,20111124,20111132,20111144
,20111152,20111153,20111158,20111161,20111165,20111171,20111172,20111173,20111176,20111178,20
111185,20111186,20111188,20111191,20111192,20111197,20111198,20111202,20111203,20111207,20111
211,20111212,20111214,20111232,20111241,21111016,21111023,21111026,21111044,21111047,21111051
,21111053,21111058,21111063,21111076,21111093,21111095,21111096,21111097,21111098,21111101,21
111102,21111105,21111106,21111107,21111108,21111111,21111112,21111113,21111115,21111117,21111
121,21111124,21111132,21111148,21111156,21111157,21111164,21111165,21111171,21111181,21111182
,21111183,21111186,21111188,21111195,21111196,21111198,21111201,21111202,21111207,21111208,21
111212,21111213,21111217,21111221,21111222,21111224,21111228,21111231,21111233,21111234,21111
237,21111238,21111241,21111243,21111248,21111255,21111261,21111263,21111282,22111016,22111023
,22111026,22111044,22111047,22111051,22111053,22111058,22111063,22111076,22111093,22111095,22
111096,22111097,22111098,22111101,22111102,22111105,22111106,22111107,22111108,22111111,22111
112,22111113,22111115,22111117,22111121,22111124,22111132,22111148,22111156,22111157,22111164
,22111165,22111171,22111181,22111182,22111183,22111186,22111188,22111195,22111196,22111198,22
111201,22111202,22111207,22111208,22111212,22111213,22111217,22111221,22111222,22111224,22111
228,22111231,22111233,22111234,22111237,22111238,22111241,22111243,22111248,22111255]
poss1_HT = poss1_HT[poss1_HT['MatchId'].isin(matches1)]
poss1_HT = poss1_HT[~poss1_HT['InPossessionClubId'].isnull()]

```

```

poss1_HT = poss1_HT[poss1_HT['Half'] == 1]

poss1_HT['P'] = None
poss1_HT['dist'] = None
f = None
p = 1
for i, row in poss1_HT.iterrows():

    if f == None:
        f = poss1_HT.at[i, 'InPossessionClubId']
        poss1_HT.at[i, 'P'] = p
    elif f == row['InPossessionClubId']:
        poss1_HT.at[i, 'P'] = p
    elif f != row['InPossessionClubId']:
        f = poss1_HT.at[i, 'InPossessionClubId']
        p += 1
        poss1_HT.at[i, 'P'] = p

distance1_half = poss1_HT.groupby(['P', 'MatchId', 'InPossessionClubId', 'TeamAId',
'TeamBId'])['XmPossession'].agg({'first', 'last'})
distance1_half['dist'] = distance1_half['last'] - distance1_half['first']
distance1_half

#HALFTIME DIST AVG 191....
poss_HT = csv_dropping[['MatchId', 'Half', 'TeamAId', 'TeamBId', 'SeqNumber', 'EventCode',
'EventName', 'InPossessionClubId', 'XmPossession']]
matches = [19111015, 19111023, 19111028, 19111032, 19111033,
19111034,
19111035,
19111038,
19111043,
19111048,
19111053,
19111054,
19111058,
19111061,
19111062,
19111063,
19111064,
19111065,
19111066,
19111071,
19111072,
19111073,
19111074,
19111075,
19111077,
19111086,
19111087,
19111088,
19111091,
19111097,
19111098,
19111101,
19111102,
19111104,

```

19111107,
19111108,
19111111,
19111112,
19111113,
19111114,
19111121,
19111122,
19111123,
19111131,
19111133,
19111135,
19111137,
19111141,
19111142,
19111143,
19111145,
19111151,
19111153,
19111155,
19111158,
19111161,
19111162,
19111163,
19111171,
19111173,
19111174,
19111175,
19111176,
19111177,
19111178,
19111183,
19111188,
19111192,
19111193,
19111194,
19111195,
19111197,
19111202,
19111205,
19111206,
19111211,
19111212,
19111213,
19111214,
19111215,
19111225,
19111227,
19111231,
19111233,
19111234,
19111236,
19111237,
19111238,
19111241,
19111243,

```

19111244,
19111248,
19111252,
19111253,
19111254,
19111257,
19111262,
19111271,
19111282,
19111291]
poss_HT = poss_HT[poss_HT['MatchId'].isin(matches)]
poss_HT = poss_HT[~poss_HT['InPossessionClubId'].isnull()]
poss_HT = poss_HT[poss_HT['Half'] == 1]

poss_HT['P'] = None
poss_HT['dist'] = None
f = None
p = 1
for i, row in poss_HT.iterrows():

    if f == None:
        f = poss_HT.at[i, 'InPossessionClubId']
        poss_HT.at[i, 'P'] = p
    elif f == row['InPossessionClubId']:
        poss_HT.at[i, 'P'] = p
    elif f != row['InPossessionClubId']:
        f = poss_HT.at[i, 'InPossessionClubId']
        p += 1
        poss_HT.at[i, 'P'] = p

distance_half = poss_HT.groupby(['P', 'MatchId', 'InPossessionClubId', 'TeamAId',
'TeamBId'])['XmPossession'].agg({'first', 'last'})
distance_half['dist'] = distance_half['last'] - distance_half['first']
distance_half.head()

poss_merged_half = pd.concat([distance_half, distance1_half], axis = 0)
poss_merged_half

dist1_HT = poss_merged_half.groupby(['MatchId', 'InPossessionClubId', 'TeamAId',
'TeamBId'])['dist'].agg(avg=('mean'))
dist1_HT = dist1_HT.reset_index(drop=False)
dist1_HT['TeamAdist_half'] = np.where((dist1_HT['TeamAId'] == dist1_HT['InPossessionClubId']),
dist1_HT['avg'], 0)
dist1_HT['TeamBdist_half'] = np.where((dist1_HT['TeamBId'] == dist1_HT['InPossessionClubId']),
dist1_HT['avg'], 0)
distance2_HT = dist1_HT.groupby(['MatchId'], as_index=True)['TeamAdist_half',
'TeamBdist_half'].max()
#print(distance1.columns)
distance2_HT = distance2_HT.drop_duplicates()
distance2_HT = distance2_HT.reset_index(drop=False)
distance2_HT.head()

half2 = csv_dropping[['MatchId', 'SeqNumber', 'Half', 'Points', 'XmPhysical', 'TeamAId', 'TeamBId',
'InPossessionClubId', 'EventCode', 'Score', 'OppScore', 'TeamAScore', 'TeamBScore']]
half2 = half2.dropna(axis=0, subset=['InPossessionClubId'])
half2 = half2[half2['Half'] == 1]

```



```

half2 = half2[(half2['Score'] > 0) | (half2['OppScore'] > 0)]
half2 = half2[half2['EventCode'] == 'ENPO']
half2 = half2[half2['TeamAId'] == half2['InPossessionClubId']]
half2 = half2.groupby(['MatchId'], as_index=False).agg(ScoreA_halftime=('Score', 'last'),
ScoreB_halftime=('OppScore', 'last'))
#half2 = half2[half2['TeamBScore'] < half2['ScoreB']]
pd.set_option('display.max_rows', 5000)
half2.head()

final = pd.merge(test_HT, half2)
final = pd.merge(final, distance2_HT)
final = pd.merge(final, duration, on='MatchId')
final = pd.merge(final, target, on='MatchId')
final.head()

#ADDING VENUE AND WEATHER COLUMNS
vewa = csv_merged[['MatchId', 'VenueId', 'WeatherConditionId']]
vewa = vewa.drop_duplicates('MatchId')
vewa = vewa[['MatchId', 'VenueId', 'WeatherConditionId']]
vewa = vewa.sort_values('MatchId')
final1 = pd.merge(final, vewa)
final1.isnull().sum()

test = csv_dropping.copy()
ruckInf = test[(test['EventCode'] == 'RINS') | ((test['EventCode'] == 'PABD') & ((test['Qualifier1'] ==
'Markers not square') | (test['Qualifier1'] == '2nd effort') | (test['Qualifier1'] == 'Flop') | (test['Qualifier1']
== 'Hand on ball') | (test['Qualifier1'] == 'Holding Down') | (test['Qualifier1'] == 'Lying in ruck') |
(test['Qualifier1'] == 'Slow peel') | (test['Qualifier1'] == 'Working on ground')))]

count = ruckInf[['SeasonId', 'MatchId']]
amount = count.groupby(['SeasonId']).count().reset_index()
amount

plt.plot(['2019', '2020', '2021', '2022'], amount['MatchId'])
plt.ylabel('Ruck Infringement Count')
plt.xlabel('Years')
plt.title('Number of Ruck Infringements per Year')

#games per year
gY = ruckInf[['SeasonId', 'MatchId']]
gY1 = gY.groupby(['SeasonId', 'MatchId']).size()
gY1 = gY1.groupby(['SeasonId']).count().reset_index(name="count")

#avg ruck per year
avg = pd.concat([amount, gY1])
#gY1['avg'] = 0
#gY1['avg'] = amount['MatchId']/gY1['count']
avg = avg.groupby(['SeasonId']).max()
avg['avg'] = avg['MatchId']/avg['count']

plt.plot(['2019', '2020', '2021', '2022'], avg['avg'])
plt.ylabel('Ruck Infringement Average')
plt.xlabel('Years')
plt.title('Average Number of Ruck Infringements per Game')

#show distribution of features

```

```

sns.distplot(csv_dropping['XmPlayer'], fit=norm);

# Get the fitted parameters used by the function
(mu, sigma) = norm.fit(csv_dropping['XmPlayer'])
print( '\n mu = {:.2f} and sigma = {:.2f} \n'.format(mu, sigma))

#Now plot the distribution
plt.legend(['Normal dist. ($\mu=${:.2f} and $\sigma=${:.2f} )'.format(mu, sigma)],
          loc='best')
plt.ylabel('Frequency')
plt.title('SalePrice distribution')

#Get also the QQ-plot
fig = plt.figure()
res = stats.probplot(csv_dropping['XmPlayer'], plot=plt)
plt.show()

#show distribution of features
sns.distplot(csv_dropping['YmPlayer'], fit=norm);

# Get the fitted parameters used by the function
(mu, sigma) = norm.fit(csv_dropping['YmPlayer'])
print( '\n mu = {:.2f} and sigma = {:.2f} \n'.format(mu, sigma))

#Now plot the distribution
plt.legend(['Normal dist. ($\mu=${:.2f} and $\sigma=${:.2f} )'.format(mu, sigma)],
          loc='best')
plt.ylabel('Frequency')
plt.title('SalePrice distribution')

#Get also the QQ-plot
fig = plt.figure()
res = stats.probplot(csv_dropping['YmPlayer'], plot=plt)
plt.show()

#Basic Rule change info for reference

#2019:
#Always a penalty

#2020:
#Replacing a penalty with a set restart anywhere on the field

#2021:
#Replacing a penalty with a set restart for any player deemed to be offside from the play of the ball

#2022:
#Any ruck infringement or offside conceded inside the attacking team's 0-40 zone was penalised instead
#of a set restart. Outside of the 0-40m zone remained a set restart

tackles = test[(test['EventCode'] == 'REFT')]

tack19 = tackles.query("SeasonId == 2019")
tack20 = tackles.query("SeasonId == 2020")
tack21 = tackles.query("SeasonId == 2021")

```

```

tack22 = tackles.query("SeasonId == 2022")
tX19=tack19['XmPossession']
tY19=tack19['YmPossession']
tX20=tack20['XmPossession']

tY20=tack20['YmPossession']

tX21=tack21['XmPossession']
tY21=tack21['YmPossession']
tX22=tack22['XmPossession']
tY22=tack22['YmPossession']

fig, axs = plt.subplots(2, 2, figsize=(20, 10))

axs[0, 0].hist2d(tX19, tY19, bins=(np.arange(-10, 110.01, 10), np.arange(0, 70, 23.33)))
axs[0, 0].set(xlim=(-10, 110), ylim=(0, 70))
axs[0, 0].set_title('2019')
axs[0, 1].hist2d(tX20, tY20, bins=(np.arange(-10, 110.01, 10), np.arange(0, 70, 23.33)))
axs[0, 1].set(xlim=(-10, 110), ylim=(0, 70))
axs[0, 1].set_title('2020')
axs[1, 0].hist2d(tX21, tY21, bins=(np.arange(-10, 110.01, 10), np.arange(0, 70, 23.33)))
axs[1, 0].set(xlim=(-10, 110), ylim=(0, 70))
axs[1, 0].set_title('2021')
axs[1, 1].hist2d(tX22, tY22, bins=(np.arange(-10, 110.01, 10), np.arange(0, 70, 23.33)))
axs[1, 1].set(xlim=(-10, 110), ylim=(0, 70))
axs[1, 1].set_title('2022')


for ax in axs.flat:
    ax.set(xlabel='X position', ylabel='Y position')


# Hide x labels and tick labels for top plots and y ticks for right plots.
for ax in axs.flat:
    ax.label_outer()

fig.suptitle("Tackle positions heatmaps for each year in 2019-2022")
plt.show()

```

```

rins=ruckInf
X=rins["XmPlayer"]
Y=rins["YmPlayer"]
plt.scatter(X, Y)

rins19 = rins.query("SeasonId == 2019")
rins20 = rins.query("SeasonId == 2020")
rins21 = rins.query("SeasonId == 2021")
rins22 = rins.query("SeasonId == 2022")

X19=rins19["XmPlayer"]
Y19=rins19["YmPlayer"]
plt.scatter(X19, Y19)

X20=rins20["XmPlayer"]
Y20=rins20["YmPlayer"]
plt.scatter(X20, Y20)

X21=rins21["XmPlayer"]
Y21=rins21["YmPlayer"]
plt.scatter(X21, Y21)

X22=rins22["XmPlayer"]
Y22=rins22["YmPlayer"]
plt.scatter(X22, Y22)

#Calculate the average x and y coordinate on the field of ruck infringements for each year
mx19=np.mean(X19)
my19=np.mean(Y19)
mx20=np.mean(X20)
my20=np.mean(Y20)
mx21=np.mean(X21)
my21=np.mean(Y21)
mx22=np.mean(X22)
my22=np.mean(Y22)

#scatterplot all ruck infringements, colour coding by year, superimposing a line seperating infringements
inside
#the attacking team's 0-40m zone (right), from those outside it (left)
#also display the mean coordinate of ruck infringement for each year

plt.figure(figsize=(20, 10))
for index, row in rins.iterrows():
    if (row["SeasonId"]==2019):
        plt.scatter(row["XmPlayer"],row["YmPlayer"],c='Red', s=30, alpha=0.8)
    if (row["SeasonId"]==2020):
        plt.scatter(row["XmPlayer"],row["YmPlayer"],c='Blue',s=30, alpha=0.8)
    if (row["SeasonId"]==2021):
        plt.scatter(row["XmPlayer"],row["YmPlayer"],c='Green',s=30, alpha=0.8)
    if (row["SeasonId"]==2022):
        plt.scatter(row["XmPlayer"],row["YmPlayer"],c='Orange',s=30, alpha=0.8)

plt.scatter(mx19, my19, c='Red', s=400, alpha=0.8)
plt.scatter(mx20, my20, c='Blue', s=400, alpha=0.8)
plt.scatter(mx21, my21, c='Green', s=400, alpha=0.8)

```

```

plt.scatter(mx22, my22, c='Orange', s=400, alpha=0.8)
plt.axvline(x=60,c='Purple')

#Plot a 2x2 figure seperating the above data by year
plt.figure(figsize=(20, 10))
fig, axs = plt.subplots(2, 2, figsize=(20, 10))
axs[0, 0].scatter(X19, Y19, c='Red')
axs[0, 0].set_title('2019')
axs[0, 0].axvline(x=60,c='Purple')
axs[0, 1].scatter(X20, Y20, c='Blue')
axs[0, 1].set_title('2020')
axs[0, 1].axvline(x=60,c='Purple')
axs[1, 0].scatter(X21, Y21, c='Green')
axs[1, 0].set_title('2021')
axs[1, 0].axvline(x=60,c='Purple')
axs[1, 1].scatter(X22, Y22, c='Orange')
axs[1, 1].set_title('2022')
axs[1, 1].axvline(x=60,c='Purple')

for ax in axs.flat:
    ax.set(xlabel='Y position', ylabel='X position')

# Hide x labels and tick labels for top plots and y ticks for right plots.
for ax in axs.flat:
    ax.label_outer()

plt.hist2d(X, Y, bins=30)
plt.colorbar()
plt.title('Heatmap for all infringements')
plt.xlabel("x position")
plt.ylabel("y position")

plt.show()

#Create a 2x2 subplot of 2D histograms for each year with centre, left and right zones
# for each 10m lengthwise section of the field
fig, axs = plt.subplots(2, 2, figsize=(20, 10))

axs[0, 0].hist2d(X19, Y19, bins=(np.arange(-10, 110.01, 10), np.arange(0, 70, 23.33)))
axs[0, 0].set(xlim=(-10, 110), ylim=(0, 70))
axs[0, 0].set_title('2019')
axs[0, 1].hist2d(X20, Y20, bins=(np.arange(-10, 110.01, 10), np.arange(0, 70, 23.33)))
axs[0, 1].set(xlim=(-10, 110), ylim=(0, 70))
axs[0, 1].set_title('2020')
axs[1, 0].hist2d(X21, Y21, bins=(np.arange(-10, 110.01, 10), np.arange(0, 70, 23.33)))
axs[1, 0].set(xlim=(-10, 110), ylim=(0, 70))
axs[1, 0].set_title('2021')
axs[1, 1].hist2d(X22, Y22, bins=(np.arange(-10, 110.01, 10), np.arange(0, 70, 23.33)))
axs[1, 1].set(xlim=(-10, 110), ylim=(0, 70))
axs[1, 1].set_title('2022')
plt.show()

#Create a 2x2 subplots of regular histograms for each year's infringements
#Do this for both density, and frequency

```

```

plt.figure(figsize=(20, 10))
fig, axs = plt.subplots(2, 2, figsize=(20, 10))
axs[0, 0].hist(X19, rwidth=0.95, density=True)
axs[0, 0].set_title('2019')
axs[0, 0].set_ylim([0, 0.030])
axs[0, 0].plot(kind = "kde")
axs[0, 1].hist(X20, rwidth=0.95, density=True)
axs[0, 1].set_title('2020')
axs[0, 1].set_ylim([0, 0.030])
axs[1, 0].hist(X21, rwidth=0.95, density=True)
axs[1, 0].set_title('2021')
axs[1, 0].set_ylim([0, 0.030])
axs[1, 1].hist(X22, rwidth=0.95, density=True)
axs[1, 1].set_title('2022')
axs[1, 1].set_ylim([0, 0.030])

```

```

plt.figure(figsize=(20, 10))
fig, axs = plt.subplots(2, 2, figsize=(20, 10))
axs[0, 0].hist(X19, rwidth=0.95, density=False)
axs[0, 0].set_title('2019')
axs[0, 0].set_ylim([0, 110])
axs[0, 0].plot(kind = "kde")
axs[0, 1].hist(X20, rwidth=0.95, density=False)
axs[0, 1].set_title('2020')
axs[0, 1].set_ylim([0, 110])
axs[1, 0].hist(X21, rwidth=0.95, density=False)
axs[1, 0].set_title('2021')
axs[1, 0].set_ylim([0, 110])
axs[1, 1].hist(X22, rwidth=0.95, density=False)
axs[1, 1].set_title('2022')
axs[1, 1].set_ylim([0, 110])

```

```

#Numerical representation comparing total and relative number of ruck infringments inside 20m of goal,
#and outside 60m between years
# (the latter is inside the attacking team's 0-40m zone)
rinsG20=rins.query("XmPlayer <= 20")
rinsG20.groupby("SeasonId").count()

```

```

tackG20=tackles.query("XmPossession >= 80")
tackG20.groupby("SeasonId").count()

```

```

rinsI40=rins.query("XmPlayer >= 60")
rinsI40.groupby("SeasonId").count()

```

```

tackI40=tackles.query("XmPossession <= 40")
tackI40.groupby("SeasonId").count()

```

```

rinsI40column=rinsI40.groupby("SeasonId").count()["MatchId"]
rinsG20column=rinsG20.groupby("SeasonId").count()["MatchId"]
rinsTotalColumn=rins.groupby("SeasonId").count()["MatchId"]
rinsI40percentage=rinsI40column/rinsTotalColumn*100
rinsG20percentage=rinsG20column/rinsTotalColumn*100

```

```

tackI40column=tackI40.groupby("SeasonId").count()["MatchId"]

```

```

tackG20column=tackG20.groupby('SeasonId').count()['MatchId']
tackTotalColumn=tackles.groupby('SeasonId').count()['MatchId']
tackI40percentage=tackI40column/tackTotalColumn*100
tackG20percentage=tackG20column/tackTotalColumn*100

RINSdf = pd.DataFrame()
RINSdf['Total']=rinsTotalColumn
RINSdf['Within 20m of goal']=rinsG20column
RINSdf['Within 20m %']=rinsG20percentage
RINSdf['0-40m zone']=rinsI40column
RINSdf['0-40m zone %']=rinsI40percentage

TACKdf=pd.DataFrame()
TACKdf['Total']=tackTotalColumn
TACKdf['Within 20m of goal']=tackG20column
TACKdf['Within 20m %']=tackG20percentage
TACKdf['0-40m zone']=tackI40column
TACKdf['0-40m zone %']=tackI40percentage

print("Ruck Infringements")
print(RINSdf)
print('\n')
print("Tackles")
print(TACKdf)

RINSdfN = pd.DataFrame()
RINSdfN['Total']=round(rinsTotalColumn/seasonGames["MatchId"], 2)
RINSdfN['Within 20m of goal']=round(rinsG20column/seasonGames["MatchId"], 2)
RINSdfN['Within 20m %']=round(rinsG20percentage, 2)
RINSdfN['0-40m zone']=round(rinsI40column/seasonGames["MatchId"], 2)
RINSdfN['0-40m zone %']=round(rinsI40percentage, 2)

TACKdfN=pd.DataFrame()
TACKdfN['Total']=round(tackTotalColumn/seasonGames["MatchId"], 2)
TACKdfN['Within 20m of goal']=round(tackG20column/seasonGames["MatchId"], 2)
TACKdfN['Within 20m %']=round(tackG20percentage, 2)
TACKdfN['0-40m zone']=round(tackI40column/seasonGames["MatchId"], 2)
TACKdfN['0-40m zone %']=round(tackI40percentage, 2)

print("Ruck Infringements per game")
print(RINSdfN)
print('\n')

```

```
print("Tackles per game")
```

```
print(TACKdfN)
```

```
rins21column=rins21.value_counts('EventCode')
```

```
print(rins21column)
```

```
rins21tot=rins21.count()['MatchId']
```

```
print(rins21tot)
```

```
rins21percentage=rins21column/rins21tot*100
```

```
print(rins21percentage)
```

```
RINSdf21 = pd.DataFrame()
```

```
RINSdf21['Count']=rins21column
```

```
RINSdf21['Percentage']=round(rins21percentage, 2)
```

```
print("2021 Ruck infringements penalty type counts")
```

```
print("\n")
```

```
print(RINSdf21)
```

```
x = np.array([2019,2020,2021,2022])
```

```
y= np.array([4.25,0,0.66,3.06])
```

```
plt.plot(x,y)
```

```
plt.title("Average Number of Ruck Infringement Penalties per Game")
```

```
plt.xlabel("Years")
```

```
plt.ylabel("Ruck Infringement Penalty count")
```

```
plt.xticks(x)
```

```
R21=rins21.query("EventCode == 'RINS'")
```



```

P21=rins21.query("EventCode == 'PABD'")

R21X=R21['XmPossession']
R21Y=R21['YmPossession']
P21X=P21['XmPossession']
P21Y=P21['YmPossession']

plt.scatter(R21X, R21Y)
plt.scatter(P21X, P21Y)

possessions = test[(test['EventCode'] == 'CVMS') | (test['EventCode'] == 'CVOK') | (test['EventCode'] == 'ENPO') | (test['EventCode'] == 'STPO') | (test['Points']==4) | (test['Points']==2) | (test['Points']==1)]
possessions = possessions.sort_values(['MatchId', 'Half', 'SeqNumber'], ascending=[True, True, True])

position = possessions[['MatchId', 'SeqNumber', 'SeasonId', 'XmPossession', 'Points']]
position = position[position['Points'] > 0]
position = position.groupby(['MatchId', 'SeasonId', 'SeqNumber']).agg({'XmPossession':'first', 'Points':'first'})[['XmPossession', 'Points']].reset_index()

posFinal = pd.DataFrame(columns = ['MatchId', 'SeqNumber', 'Year', 'Starting Position X', 'Starting Position Y', 'Possession Score'])
for j, row2 in position.iterrows():
    pos = possessions[(possessions['MatchId'] == row2['MatchId']) & (possessions['SeqNumber'] == row2['SeqNumber'])]
    pos = pos[['YmPossession']]
    posFinal = posFinal.append({'MatchId': row2['MatchId'], 'SeqNumber': row2['SeqNumber'], 'Year': row2['SeasonId'], 'Starting Position X': row2['XmPossession'], 'Starting Position Y': pos.iloc[0]['YmPossession'], 'Possession Score': row2['Points']}, ignore_index = True)
print(posFinal)

y=posFinal['Starting Position X']
x=posFinal['Starting Position Y']
plt.scatter(y, x)

pos19 = posFinal[posFinal['Year'] == 2019]
pos20 = posFinal[posFinal['Year'] == 2020]
pos21 = posFinal[posFinal['Year'] == 2021]
pos22 = posFinal[posFinal['Year'] == 2022]

y19=pos19['Starting Position Y']
x19=pos19['Starting Position X']
y20=pos20['Starting Position Y']
x20=pos20['Starting Position X']
y21=pos21['Starting Position Y']
x21=pos21['Starting Position X']
y22=pos22['Starting Position Y']
x22=pos22['Starting Position X']

plt.scatter(x19, y19)
plt.scatter(x20, y20)
plt.scatter(x21, y21)
plt.scatter(x22, y22)

```

```

#feature engineering
final1 = final1.drop('MatchId', axis=1)
final1['Margin'] = final1['ScoreA_halftime'] - final1['ScoreB_halftime']
#attendance

#label encoding
label = final1.copy()
label['TeamAId'] = label['TeamAId'].astype(str)
label['TeamBId'] = label['TeamBId'].astype(str)

cols = ('TeamAId', 'TeamBId')
# process columns, apply LabelEncoder to categorical features
for c in cols:
    lbl = LabelEncoder()
    lbl.fit(list(label[c].values))
    label[c] = lbl.transform(list(label[c].values))
label.head()

# Look at the distribution of each individual variable including the dependent variable 'Withdraw'
label.plot(lw=0, marker=".", subplots=True, layout=(-1, 4),
          figsize=(15, 12), markersize=1)

# Look at histogram of distributions of each variable
label.hist(bins=25, figsize=(15, 8), layout=(-1, 5), edgecolor="black")
plt.tight_layout()

#correlation
corr = label.corr()
corr.style.background_gradient(cmap='coolwarm')

sns.pairplot(label, hue="TeamAWins")

X = final1.copy()
X = X.rename({'TeamAWins': 'Results'}, axis=1)
y = X['Results']
X = X.drop('Results',1)
X = X.drop('TeamAId',1)
X = X.drop('TeamBId',1)
X.head()

#standardise data
scale= StandardScaler()

# separate the independent and dependent variables
X_data = X

# standardization of dependent variables
scaled_data = scale.fit_transform(X_data)

list_operation=[SVC(),LogisticRegression(),AdaBoostClassifier(),RandomForestClassifier(),DecisionTreeC
lassifier(),MLPClassifier()]
name_operation=["SVC","LogisticRegression","AdaBoostClassifier","RandomForestClassifier","Decision
TreeClassifier","MLPClassifier"]
for i in range(6):
    Xtrain,Xtest,Ytrain,Ytest=train_test_split(X_data,y,test_size=0.7)

```

```

operation=list_operation[i]
operation.fit(Xtrain,Ytrain)
accuracy=operation.score(Xtest,Ytest)
f1=f1_score(Ytest, operation.predict(Xtest), labels=[1,0], pos_label=1)
print("%s accuracy is %.3f and f1_score is %.3f"%(name_operation[i],accuracy,f1))

Xtrain,Xtest,Ytrain,Ytest=train_test_split(X_data, y , test_size=0.7)
operation = RandomForestClassifier()
operation.fit(Xtrain,Ytrain)
scores=cross_val_score(estimator=operation,X=Xtrain,y=Ytrain,cv=10,n_jobs=1)
print("CV Accuracy Scores: %s\n" % scores)

Xtrain,Xtest,Ytrain,Ytest=train_test_split(X_data,y,test_size=0.7)
score= []
for i in range(150):
    rfc = RandomForestClassifier(n_estimators= i+1)
    rfc.fit(Xtrain,Ytrain)
    f1=f1_score(Ytest, rfc.predict(Xtest), labels=[1,0], pos_label=1)
    score.append(f1)
plt.figure(figsize=(15,7))
plt.plot(score)

Xtrain,Xtest,Ytrain,Ytest=train_test_split(X_data,y,test_size=0.7)
param_test1 = {"n_estimators":range(100,201,10)}
gsearch1 = GridSearchCV(estimator=RandomForestClassifier(),param_grid=param_test1,
scoring='f1',cv=10)
gsearch1.fit(Xtrain,Ytrain)

print(gsearch1.best_params_)
print("best accuracy:%f" % gsearch1.best_score_)

Xtrain,Xtest,Ytrain,Ytest=train_test_split(X_data,y,test_size=0.7)
param_test2 = {"max_features":range(1,15,1)}
gsearch2 = GridSearchCV(estimator=RandomForestClassifier(n_estimators=140),param_grid =
param_test2,scoring='f1',cv=10)
gsearch2.fit(Xtrain,Ytrain)

print(gsearch2.best_params_)
print("best accuracy:%f" % gsearch2.best_score_)

Xtrain,Xtest,Ytrain,Ytest=train_test_split(X_data,y,test_size=0.7)
test_model=RandomForestClassifier(n_estimators=120,max_features=6)
test_model.fit(Xtrain,Ytrain)

predict=X["ResultPredicted"]=test_model.predict(X_data)
sns.countplot(x="ResultPredicted", data=X)

test_error = mean_squared_error(np.ravel(final1["TeamAWins"]), predict)
print("test set MSE:",test_error)

```