



# Machine Learning Predictions for the Fantasy Premier League

by

Aaron Talbot

This thesis has been submitted in partial fulfillment for the  
degree of Bachelor of Science in Software Development

in the  
Faculty of Engineering and Science  
Department of Computer Science

May 2021

# Declaration of Authorship

I, Aaron Talbot, declare that this thesis titled, ‘Machine Learning Predictions for the Fantasy Premier League’ and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for an undergraduate degree at Cork Institute of Technology.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at Cork Institute of Technology or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this project report is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: Aaron Talbot

Date: 16/05/2021

CORK INSTITUTE OF TECHNOLOGY

## *Abstract*

Faculty of Engineering and Science  
Department of Computer Science

Bachelor of Science

by Aaron Talbot

Over 6 million users worldwide play fantasy premier league each year to come out as the best manager whether that be against their friends, within their favourite club or if you are a seasoned player to be best overall worldwide. To help with all the seasoned veterans or the newbies who have no idea what they are doing, this project will be used to help Fantasy Premier League players improve their predictions by recommending them possible transfers game week by game week. It will use a machine learning algorithm and take the players team and several other pieces of data through a mobile application interface as input and use this data to predict what transfers (if any) the player should make as output. Using this output then the user can then decide whether to trust the algorithm and make the recommended changes or to go with user instinct.

# *Acknowledgements*

I would like to thank my semester one and semester two Co-ordinator Brian Murphy for assisting me throughout the research phase, he has been excellent with ensuring that my project, he gave excellent advice and assurance throughout the process.

I would like to thank my close friend Lucy Harkin for being there for me throughout my time in college who was there with support and helped me throughout the time I had here in college. This would not have been possible without her.

I would like to thank my three closest class mates and friends, Bahaluddin Shammery, Cian O'Sullivan and Michael O'Keeffe. They have been instrumental in helping me get to this stage in the degree and have been extremely helpful throughout the last four years.

I would like to thank my Father Owen Talbot, who has been extremely supportive throughout my software development career. He has been my best friend and has helped me alot.

And, finally I would like to thank my three best friends Aaron Walsh, Cormac O'Hanlon and Eugene O'Brien for being great support throughout the years. They have been incredible in helping me achieve what I have and I would have not gotten here without them.

Would like to thank the reddit FPL community r/FPL for helping with the testing and evaluation phase. I could not populate my test league without the users who helped out.

# Contents

<b>Declaration of Authorship</b>	<b>i</b>
<b>Abstract</b>	<b>ii</b>
<b>Acknowledgements</b>	<b>iii</b>
<b>List of Figures</b>	<b>vii</b>
<b>List of Tables</b>	<b>viii</b>
<b>Abbreviations</b>	<b>ix</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Contribution . . . . .	2
1.3 Structure of This Document . . . . .	3
<b>2 Background</b>	<b>4</b>
2.1 Thematic Area within Computer Science . . . . .	4
2.1.1 Android Application Development . . . . .	4
2.1.2 Data Analytics . . . . .	5
2.1.3 Machine Learning . . . . .	5
2.1.4 CI/CD . . . . .	5
2.2 A Review of Machine Learning in Sports and Fantasy Prediction . . . . .	6
<b>3 Problem - Machine learning in Fantasy Premier League</b>	<b>11</b>
3.1 Problem Definition . . . . .	11
3.2 Objectives . . . . .	12
3.3 Functional Requirements . . . . .	12
3.4 Non-Functional Requirements . . . . .	13
<b>4 Implementation Approach</b>	<b>14</b>
4.1 Architecture . . . . .	14
4.1.1 Infrastructure . . . . .	15
4.1.2 Technologies . . . . .	15

4.1.3	Platform . . . . .	15
4.2	Use Case Description . . . . .	17
4.3	Risk Assessment . . . . .	20
4.3.1	Risk One - Gathering FPL Data . . . . .	20
4.3.2	Risk Two - Training Model(s) . . . . .	20
4.3.3	Risk Three - Converting Model(s) to Tflite models . . . . .	21
4.3.4	Risk Four - Continuous Data integration . . . . .	21
4.3.5	Risk Five - Software deploying with Bugs . . . . .	21
4.3.6	Risk Six - Mobile application with no internet connection . . . . .	22
4.4	Methodology . . . . .	22
4.4.1	Dedicated Information Gathering . . . . .	22
4.4.2	Waterfall . . . . .	23
4.4.2.1	Requirements . . . . .	23
4.4.2.2	Design . . . . .	23
4.4.2.3	Implementation . . . . .	23
4.4.2.4	Verification . . . . .	23
4.4.2.5	Maintenance . . . . .	24
4.4.3	Gantt Charts . . . . .	24
4.4.4	Feature Driven Development . . . . .	24
4.4.5	Jenkins and CI Learning . . . . .	24
4.5	Implementation Plan Schedule . . . . .	24
4.5.1	Android Application . . . . .	25
4.5.2	Machine Learning Algorithm . . . . .	25
4.5.3	Machine Learning algorithm and Application Communication . . . . .	26
4.6	Evaluation . . . . .	26
4.7	Prototype . . . . .	28
<b>5</b>	<b>Implementation</b>	<b>33</b>
5.1	Difficulties Encountered . . . . .	33
5.2	Actual Solution Approach . . . . .	34
5.2.1	Data Gathering . . . . .	34
5.2.1.1	Fixtures and Results . . . . .	34
5.2.1.2	Teams . . . . .	35
5.2.1.3	Player Data . . . . .	35
5.2.2	Machine Learning Development . . . . .	36
5.2.3	Android Application Development . . . . .	37
5.3	Implementation retrospective . . . . .	38
5.3.1	Schedule Retrospective . . . . .	38
5.3.2	Implementation Retrospective . . . . .	39
5.3.3	Risk Assessment . . . . .	39
5.3.3.1	Risk One - Gathering FPL Data . . . . .	40
5.3.3.2	Risk Two – Training Model(s) . . . . .	40
5.3.3.3	Risk Three – Converting Models to tflite models . . . . .	40
5.3.3.4	Risk Four – Continuous Data Integration . . . . .	40
5.3.3.5	Risk Five – Software Deploying with Bugs . . . . .	40
5.3.3.6	Risk Six – Mobile Application With no Internet Connection . . . . .	41

<b>6</b>	<b>Testing and Evaluation</b>	<b>42</b>
6.1	Metrics . . . . .	42
6.1.1	Regression Analysis . . . . .	42
6.1.2	Function Analysis . . . . .	42
6.2	System Testing . . . . .	42
6.2.1	Regression Analysis . . . . .	42
6.2.2	Functionality Analysis . . . . .	43
6.2.3	Application Analysis . . . . .	43
6.3	Results . . . . .	44
6.3.1	Regression Analysis . . . . .	44
6.3.2	Functional Analysis . . . . .	47
6.3.3	Application Analysis . . . . .	48
6.3.3.1	The application must accommodate several screen sizes .	48
6.3.3.2	Information to users must be detailed and give reasoning to why these recommendations were made. . . . .	48
6.3.4	Use Case Functionality . . . . .	49
6.3.4.1	Registration . . . . .	49
6.3.4.2	Login . . . . .	49
6.3.4.3	Prediction Made . . . . .	49
6.4	Functionality Demonstration . . . . .	50
<b>7</b>	<b>Discussion and Conclusions</b>	<b>55</b>
7.1	Solution Review . . . . .	55
7.2	Project Review . . . . .	56
7.2.1	Machine Learning Development . . . . .	56
7.2.2	Android Application Development . . . . .	56
7.3	Conclusion . . . . .	57
7.3.1	Primary Conclusions . . . . .	57
7.3.2	Secondary Conclusions . . . . .	58
7.4	Future Work . . . . .	58
	<b>Bibliography</b>	<b>60</b>
	<b>A Code Snippets</b>	<b>62</b>

# List of Figures

2.1	Clean sheet Probability . . . . .	7
4.1	Architecture . . . . .	14
4.2	Login Prototype . . . . .	28
4.3	Player Inputs Prototype . . . . .	29
4.4	Register Prototype . . . . .	30
4.5	Update Details Prototype . . . . .	31
4.6	Delete Account Prototype . . . . .	32
6.1	Defenders Mean Error . . . . .	44
6.2	Midfielders Mean Error . . . . .	45
6.3	Attackers Mean Error . . . . .	45
6.4	Tyrone Mings Comparison Graph . . . . .	46
6.5	Bruno Fernandes Comparison Graph . . . . .	46
6.6	Patrick Bamford Comparison Graph . . . . .	47
6.7	Home Screen . . . . .	51
6.8	Login Page . . . . .	52
6.9	Input Players . . . . .	53
6.10	Recommendation . . . . .	54



# List of Tables

4.1	Use Case One . . . . .	17
4.2	Use Case Two . . . . .	17
4.3	Use Case Three . . . . .	18
4.4	Use Case Four . . . . .	18
4.5	Use Case Five . . . . .	19
4.6	Risk Assessment Matrix . . . . .	20

# Abbreviations

<b>FPL</b>	<b>F</b> antasy <b>P</b> remier <b>L</b> eague
<b>CI</b>	<b>C</b> ontinuous <b>I</b> ntegration
<b>ML</b>	<b>M</b> achine <b>L</b> earning
<b>NaN</b>	<b>N</b> ot <b>a</b> <b>N</b> umber

# Chapter 1

## Introduction

### 1.1 Motivation

Fantasy Premier League is one of the most addictive and challenging games I have ever played. For anyone who has an interest in the premier league they most likely play fantasy premier league. With over six million users and nearly one million new users this year alone fantasy premier league is growing fast. With that the competitiveness is high whether it is in overall ranks or it is within private leagues among friends. It slowly became a passion of mine two years ago when I first started playing. The game is basic with a few set rules and mechanics. Each fantasy player has a squad consisting of fifteen players.

- Two Goalkeepers
- Five Defenders
- Five Attackers
- Three forwards

Each player in your squad has a value and that value can change throughout the season. It can go up or down based on form and ownership. Each fantasy player has a budget starting off at one hundred million and cannot exceed this. No more than three players in your squad can be from the same team in the premier league. You will then pick ten outfield players and one goalkeeper to play each week, if an outfield player does not play in the weeks fixtures your next outfield player on the bench will come on to replace them, this rule also applies for your goalkeeper. Players during game weeks earn points for the following:

- Playing Time
- Goals Scored
- Assists Scored
- Saves
- Clean Sheets
- Bonus Points

Players can also lose points for the following:

- Yellow/Red Card
- Every Two Goals Conceded
- Own Goals
- Penalty Miss

With the rapid increase in the popularity of the game, the skill gap between new and experienced players can be grow exponentially throughout the year. So, I decided to give up as I had nothing else to gain. This is a microcosm of what happens across the player base, which can happen early in the season or late on the season as people are not as skilled as others and fall behind.

With this being the case for many managers across the player base my project is to help with the decision making for managers who are unsure on what to do by using a machine learning algorithm to decide what transfer they should be making each week if they need to make a transfer at all. The main goal is to be maximise points for the user in the next upcoming game week.

## 1.2 Contribution

The main contribution of this project can be enumerated by several core factors of the software development bachelor's degree. The structure and curriculum of the degree enabled a strong core software development skill to set out and undertake this project. Throughout my time in in the degree the core modules delivered were extremely helpful in creating a strong understanding of software development and its life cycle. This was

done through how passionate the lecturers were in delivering their module(s) throughout the years.

Over the course of second and third year there were several modules that helped in relation to this project. Throughout second year the two main modules for problem solving was linear and non-linear data structures and algorithms, also for statistical analysis Probability and Statistics was very relevant. Throughout third year application development framework showed me how to develop a mobile application and programming for data analytic's put probability and statistics into a software development setting.

With my massive interest into sports and sports analysis fantasy premier league was big incentive to me to play. Fantasy Premier League is one of the fastest growing fantasy games over the past few years. Within the states they have a big interest in fantasy football which is extremely similar but is based on the NFL. As I have played Fantasy Premier League over the last three years and competed against my friends it was a great contribution to use my software development skills within my interest for the fantasy premier league.

### 1.3 Structure of This Document

Chapter One details what this project is all about. What this project is undertaking is in this section. Chapter Two reviews the area in which this project falls and reviews the area which the topic falls under. These areas are based around ML, Mobile Application Development and CI.

Chapter Three defines what problem we pose, the objectives seen to be undertaken by the project and the requirements whether functional or Non-Functional will be implemented into the project. Chapter Four details underlines how the project plans to be implemented, what risks are associated with the project being undertaken and the prototype for the project.

Chapter Five discuss what can be done after the project and what conclusions the project can take away from the opening phase.

## Chapter 2

# Background

### 2.1 Thematic Area within Computer Science

The main area of this project is machine learning and application development. These are the core topics that this project falls under. As this project is used to predict transfers for fantasy premier league managers the main part is the prediction which will be covered by a machine learning algorithm. To get access to this machine learning algorithm the interface used will be an android application developed using java and Firebase. To provide the data to analyse we will need to implement a CI/CD to retrieve data constantly as the Fantasy premier league's data changes day to day. The data given by fantasy premier league needs to be analysed correctly before being used to train the machine learning algorithm, for example using certain statistics on players can be useless, like the save rate for an attacking player should not be used as the statistic is mainly used for goal keepers and would have nothing to do with attacking players.

#### 2.1.1 Android Application Development

With the variety of methods to present this to the user we found an application to be the most useable. With everyone's familiarity with applications and with the parent service mainly being an application using an application seemed to be the safest choice. With my knowledge of android development from the third-year module "Mobile application Development" and our interaction with the native android development tool "Android studio" we believed that an android application was the best choice for me. Android application development is one of the largest growing markets currently within software development. Many industries have been moving to an application-built software as nearly half the world's population own a smart mobile device. This means that pushing

services to an application on people's phones has been higher than ever. Now there is applications for nearly everything whether it's a social media platform to games or an application for mobile banking. Applications have enabled the world to be "on the go".

### 2.1.2 Data Analytics

Data analytics is used across the globe for a plethora of services. With the surge of big data and how the price of data has surpassed the price of oil in recent years data now needs to be analyzed and used in a way to provide information to consumers. Whether it is for stock traders how analyze the stock market for certain patterns to know whether a certain stock is about to gain in price or drop in price or from analysing blood samples to see whether someone has a high cancer or diabetes.

### 2.1.3 Machine Learning

Machine Learning is the staple into how to usefully and affectively user data in the field. Machine Learning can be broken down into the sub streams whether it be supervised learning where there is a target set for the algorithm to meet and using dependent variables and independent variables to get desired outputs or unsupervised learning where the algorithm does not have a set of desired outputs it is used more to cluster data together such as K-means clustering algorithm. Then reinforcement learning where the algorithm is shown to make specific decisions where the machine trains itself within an environment with trial and error and learns from past training to make the best possible decision it can in the future. A question that needs to be answered by the paper is what type of machine learning is affective for a prediction model for individual fantasy premier league teams, which will work well, and which will not work? This will be done by using the different types of machine learning mentioned above.

### 2.1.4 CI/CD

Continuous integration is a coding philosophy used in agile methodologies. A CI is mainly used to drive development teams to make small changes and to check into version control repositories frequently. In relation to this topic and how the fantasy premier league statistics change daily. If this data was not kept up to date, then the predictions made by the machine learning algorithm could lead to incorrect predictions and may lead to predications being possible according to older data but not according to newer data[1].

## 2.2 A Review of Machine Learning in Sports and Fantasy Prediction

Throughout this review, the author hopes to gain an understanding of what has been done across the Fantasy type games regarding machine learning and the prediction of sports fixtures and statistics. Prediction of fantasy games and prediction of just sports fixtures have been done by many in the past.

Fantasy Premier League themselves do not offer any sort of recommendation team by team but do offer a scout selection and a scout team. What this comprises of is a selection of players they believe will bank a decent amount of points for the week ahead, and they also recommend the team that is predicted to be the best performing team that week under the rules. This comprises of players who are consistent each week and worth their value and selections the scout believes can do well for this week as their opponent may not be up to standard or have an underlying statistic that goes against them that can be exploited by the choice of the scouts selection[2].

With the high complexity of the problem at hand there have been many takes on predicting sports results in general. Most have taken a purely historical approach and applying the result of what happened in the past as what will happen now. With predicting premier league games using a machine learning algorithm it is found that decision tree algorithms such as a random forest and gradient boosting algorithms work well. Non-Naïve algorithms worked well as lots of the feature selection within the premier league is dependant on each other.

According to papers trying to predict fantasy predictions for the NFL whether that be for a single position selection or to predict a team for the week ahead they have found that regression type algorithms to perform a lot better then others [3]. In predicting sole positions like for example a goal keeper a linear regression model is the most affective in making that selection. Weighting statistics that are important for the set position and then analyzing each players score can help to define selections of players that do not change.

Using a predicative based algorithm based form for the player over the last n games. Within the FPL players have a statistics based on form and this is based on how the player has been performing based on their previous games and rates them on a scale out of ten and so when deciding on whether to pick a player maybe their statistics may



Home	Clean sheet %		Away
Everton	16	29	Liverpool
Chelsea	38	12	Southampton
Man City	36	9	Arsenal
Newcastle	18	36	Man Utd
Sheffield Utd	38	25	Fulham
Crystal Palace	30	31	Brighton
Tottenham	37	14	West Ham
Leicester	31	16	Aston Villa
West Brom	33	28	Burnley
Leeds	31	29	Wolves
Gameweek 5			

FIGURE 2.1: Clean sheet Probability

look fantastic but if the form has a low score then maybe it would be best to steer clear. This would show further down in the season as for example in the previous season of the premier league Teemu Pukki for Norwich had a fantastic start of the season scoring eleven goals over the course of the season with nine of the eleven coming before the Christmas break. While this is fantastic on paper and you would want him in your team going forward as his statistics are incredible his form is not. This can be said for many players and shows that the form statistics is very important. Also, this year it would show that a player is maybe injured. A player may be performing well if you look at the statistics, they could be injured the game previous and if so their form statistics would go to zero.

Using injuries can also be pivotal as it does not just mean you should not select the player that is injured but you can also select the player that may benefit due to this injury. It can also mean that a player that may have not gotten a chance or has been a popular choice could be picked as they would slot into the injured players position and value may not be as high as the other player. It could also mean that the opposing player on the other team could have a better game. If a very good attacking player was out injured the teams defence would have a higher chance of having a better game as they could hopefully not concede or score and increase points for the manager.

A popular Reddit forum on fantasy premier league like to make predictions on clean sheet probability each week using betting odds. This would be used for the defence assets in your team and especially the goal keeper as a lot of points come from clean sheets which can be seen in fig 2.1 [4].

A naïve Bayes algorithm seems to be a common occurrence, while it has its pros for classifying data it struggles when data is dependent of each other. Within statistical data for the fantasy premier league nearly all the data is dependent on each other such as if form in a player starts to increase then also the players value will increase as a result of that due to managers thinking it is a good idea to bring this player in, while also how many shots on target an attacking player has will directly affect his goals per game, so using a naïve Bayes algorithm cannot be feasible when trying to predict fantasy football teams. Even though the model can do surprisingly well due to its inherent mathematical assumptions it does not perform to the standard of other classification models. For running models and a solidified position such as a goalkeeper where you only have one of them on your active team at any one time can mean that the algorithm used to select that position may differ to other positions within the active team. In G.Sugar and T.Swenson's paper "Predicting optimal game day fantasy football teams" They found in the positions where they only have one active player they found that a Bayesian regression models works well as only one of those positions exists, so when picking the four positions with only player the Bayesian regression model worked well[5].

A basic Linear regression model can be used to predict scores in sports games. This being a inherently statistical technique which used in data mining. This is used between relationships of a dependant variable being continuous and a dependable variable(s) can be either continuous or discrete variables. This gives an output of either one unknown variable or several unknowns variables. A simple regression model can be written as " $Y = b_0 + b_1X_1 + b_2X_2 + \dots + b_pX_p$ " which is used to find the Y value [6].

Model Selection is difficult, the variance in sports prediction can lead to either high bias or inaccurate predictions. Ridge Regression can be used to help counter these problems such as ridge regression and Bayesian Ridge Regression [5]. This only works when the number of training examples is high as the training and test error normalizes the higher the training and test data is high.

Support Vector machines are a set of supervised algorithms that are effective with a high dimensional feature space. Finding an optimal gamma for SVM is important, a high gamma leads to models with high bias and low variance but a low gamma leads to the opposite [7].

Predicting sporting outcomes and individual player performances are two separate predictions, and can be predicting the former can be quite difficult. When looking at it purely at a statistical approach players may have a good game but the outcome could be a negative one for the team they play for [8].

Home advantage can lead to a boost in player and overall team performance. This has been shown to be a big factor when predicting outcomes of sporting events [8]. The base of home advantage being a massive factor when predicting outcomes is due to familiarity to the players and team as a whole, this leads to a boost in numbers shown in the Australian Football League [8] and when a team travels away from home they can be fatigued especially when travelling a long distance.

As seen in Daniel Cervone's paper about a multiresolution stochastic process model helps to predict outcomes for basketball possessions. He uses a possession criteria along side an estimator criteria to see calculate if in a possession the play will end in either a two pointer made, three pointer made or the possession is lost [9].

We can see in Wei Gu that a correlation coefficient is trained on previous tennis matches and how points were scored within these tennis matches gives us an outcomes of winning the match for when predicting future tennis matches [10]. This algorithm can be applied to single Premier League players and how they perform over the course of the game and tally how many FPL points they could score over the course of the game.

A big factor in forward and attacking selection can be set piece takers and in particular penalties [11]. With how frequent penalties are across the course of the year it would be very important to have a penalty taker within your team as they can score up to six points with a goal from a penalty. This should weight a selection for a player on whether they are a penalty taker or not. With other set pieces such as corners or free kicks whether they are direct or not would also weigh into a selection of a player as with indirect set-pieces could lead to a higher volume of assists and with direct set-pieces your selection has a higher chance of scoring, not as high as if the player was taking a penalty but higher than non set-piece takers.

Physical statistics also play a role in how well a player may do, if for example a defending player can be seen to have higher pace while still being taller it would directly impact on how many goals conceded the team would have, it also links into set-pieces as many of the taller players would score from indirect set-pieces throughout the game as their aerial ability would be stronger than the players around them. An attacker's pace and shooting ability would directly impact his goal and assist rate for a club, this would increase the popularity of that player as they will end up getting more points than others but will also go up in value due to their popularity [12].

Players can also be unknowns, many players could come up from the academy during the year due to injury to a main player or could come to the premier league down to a transfer at the start of the season or in the mid-season transfer window. These players would have to be factored into an algorithm separately as they do not have any historical

data in the season and may not have a form statistic given to them by the fantasy premier league. This could leave the manager confused as they would not know whether this brand-new player could perform in the league. Some players may do exceptionally well such as Mohammed Salah in his first season getting the golden boot (most goals scored by a single player) with thirty-two goals. Others go against this trend like Fernando Torres whom signed for Chelsea and in his first year only managed one goal in fourteen appearances.

## Chapter 3

# Problem - Machine learning in Fantasy Premier League

### 3.1 Problem Definition

The problem we are trying to solve is deciding the optimal position change for a manager's player transfer each week, dependent on the manager's team and current budget. This varies from manager to manager as each team differs in players, and you are only given one free transfer in a single game week which can be carried over to one additional week if not used. The problem is broken down by several factors. The first of those being is a player sick or injured and will not play in the upcoming fixture(s). If this is then the player will not score points for the upcoming game week(s) dependent on whether the injury or illness is severe or not. If a player of high value (Points to price ratio) it may not be worth it to use a transfer on this player, if you do you might lose potential points in future weeks when the player is fit again. The next is if a player has a set of hard fixtures coming up, such as a player who could have consistent points but plays for a side that does not perform well versus bigger teams. When this happens players could dip in points production for example if a goal keeper came across a team who has a strong attacking threat, then the goal keeper would not score many points in this run of games and could be transferred out for a player with a better run of games. This applies across the pitch, from the defensive set of players to the attacking sense of players. The examples given above are more of the simple examples, there are a lot of more intricate examples when it comes to efficiency in point production, cost and form. These are a lot harder to understand for the average user and this usually leads to the split between the upper echelon of players and the casual players. We can take a set of defensive players for example, the five defenders on your team must be good enough to

play three each game week, this means they must get some sort of return, hopefully an attacking return as they score more for defenders. So you could have a set of defenders that could be worth a lot of money, but then this will impact your team later as you could overly spend on premium defenders but then lose out when you are picking an attacking player. So, finding the balance between points production and value. This for the newcomers can be hard to figure out as they may not have enough knowledge when it comes to picking players.

## 3.2 Objectives

This project's main objective is to help the casual player of Fantasy Premier League, to make the transfer dilemma easier week by week which can be quite difficult when there is no obvious transfer to make. we wish to create a machine learning algorithm that encapsulates the process of transfers for a manager and makes the optimal prediction. This algorithm will be using the managers current team and if the manager has a free transfer available. This algorithm will not work unless we can gather raw data of players that can be used and their respective data. This will be tackled by collecting data consistently throughout the week with the changing market and the price of players. Once the players data has been gathered the algorithm can work accordingly, to encapsulate the algorithm and to make the inputs and outputs clearer we plan to develop a mobile application on top of this which can communicate to the algorithm what team the manager has and so the manager can see what transfers are suggested by said algorithm.

## 3.3 Functional Requirements

The following are the defined functional requirements for this project.

- Create an Algorithm that inputs a team and current budget and outputs the recommended transfers for that team.
- Create a mobile application that can be used by users to communicate with that algorithm.
- Create a CI that can constantly update the players data for changes.
- Create an authentication service responsible for user management.

### **3.4 Non-Functional Requirements**

The following are the defined non-functional requirements for this project.

- The application must accommodate several different screen sizes.
- Information to users must be detailed and give reasoning to why predictions are made.
- Application should be user friendly and straight to the point.

## Chapter 4

# Implementation Approach

### 4.1 Architecture

Throughout this section, we will outline the three main pieces that comprise the architecture. These three subcategories are infrastructure, technologies and platform. This chapter will elaborate through each of these headings and explain how I plan to implement and configure these headings.

- Infrastructure
- Technologies
- Platform

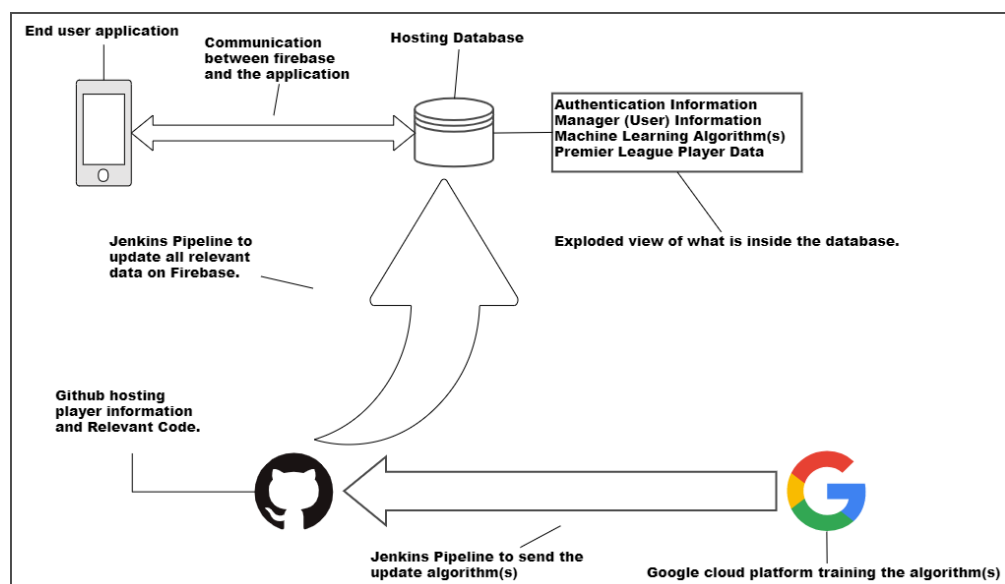


FIGURE 4.1: Architecture



### 4.1.1 Infrastructure

The central infrastructure that we will use is firebase. Firebase is a hosting tool that can host many applications from websites to mobile applications, whether they be Android or iOS applications. It can also host a database of users of these applications or what data may see fit to be used or these end applications. Firebase can also host machine learning models for use within the applications or the websites. Once we have our firebase setup, we can set up continuous integration systems to update our data within the database and keep the Machine Learning modules updated. We can also integrate the GitHub repository to keep the mobile application up to date.

### 4.1.2 Technologies

Now that the base infrastructure, we will be looking to deploy our technologies onto this base infrastructure to be used. As a part of the pipeline the platform will use many technologies such as;

- Android Studio will be used in development with firebase to create the mobile application, so it is ready to deploy.
- SciKit Learn and python to create machine learning algorithms.
- SciKit Learn and python will be used in tandem with Google collab to train these algorithms.
- We will use Jenkins as a CI tool to update players' database and update the mobile application.
- Tensor Flow and Tensor Flow lite so that after the creation of the models, they can be hosted correctly on firebase for communication between the models and the application.
- Firebase as a hosting platform.
- FPL framework within python will be used to gather the relevant data during the data gathering stage.

### 4.1.3 Platform

The central platform will be what the end-user will use to access the algorithms and to make predictions. The mobile application will do this created using android studio.

This platform would have to communicate between the algorithms hosted on firebase and update the algorithm if it were to update on firebase. It will also need to communicate to gather information from the database such as user credentials and a user's team if they have input their team. For a user to use this platform, we will need to build a robust authentication platform. A user will be able to register or login through their email or using a Google account. They could also link their Fantasy Premier League team number with their account so that the mobile application could have access to their team without the user inputting it. A user should also delete their account after they have created one, this will need to delete all entries the user has made in the database.

## 4.2 Use Case Description

The following are use cases, which include the description and how they are fulfilled.

Use Case Name	Registration
Description	The user would like to register their account when opening the application.
Actors	Any user of FPL.
Pre-Condition(s)	The application is installed and the user is connected to the internet.
Post-Condition(s)	There are no post-conditions.
Flow Of Events	<b>Actor</b> 1. The actor opens the Application. 2. Actor clicks the registration button. 3. Actor puts in details to register within the application. <b>System</b> 1. The application is opened. 2. Application moves to the registration page. 3. System then saves the users information in the database.
Exceptions	There are no exceptions.

TABLE 4.1: Use Case One

Use Case Name	Login
Description	The user would like to open the application and login
Actors	Any user of FPL.
Pre-Condition(s)	The user has already opened the application and registered. The user is connected to the internet.
Post-Condition(s)	There are no post-conditions.
Flow Of Events	<b>Actor</b> 1. The actor opens the Application. 2. The actor enters their email and password to log in. 3. The actor clicks the login button. <b>System</b> 1. The application is opened. 2. Application opens the login page. 3. System then accepts the user if their credentials are correct/ 4. The application then logs them in.
Exceptions	There are no exceptions.

TABLE 4.2: Use Case Two

Use Case Name	Enter Team
Description	The user inputs their FPL team.
Actors	Any registered user of FPL.
Pre-Condition(s)	The user has already opened the application and logged in. The user is connected to the internet.
Post-Condition(s)	There are no post-conditions.
Flow Of Events	<b>Actor</b> <ol style="list-style-type: none"> <li>1. The actor opens the Application.</li> <li>2. The actor enters their email and password to log in.</li> <li>3. The actor clicks the log in button.</li> <li>4. Actor inputs all team members.</li> <li>5. Actor also inputs price for each player.</li> <li>6. Actor inputs how much money the player has ITB.</li> </ol> <b>System</b> <ol style="list-style-type: none"> <li>1. The application is opened.</li> <li>2. Application opens the login page.</li> <li>3. System then accepts the user if their credentials are correct/</li> <li>4. The application then logs them in.</li> <li>5. The team page is opened.</li> <li>6. All data is saved to the database.</li> </ol>
Exceptions	There are no exceptions.

TABLE 4.3: Use Case Three

Use Case Name	Prediction Made
Description	The user would like to predict which transfer to make.
Actors	Any registered user of FPL.
Pre-Condition(s)	The user has already opened the application and logged in. The user is connected to the internet.
Post-Condition(s)	The user has been given a predicted transfer.
Flow Of Events	<b>Actor</b> <ol style="list-style-type: none"> <li>1. The actor opens the Application.</li> <li>2. The actor enters their email and password to log in.</li> <li>3. The actor clicks the log in button.</li> <li>4. Actor clicks make the prediction page.</li> <li>5. Actor clicks on the make the prediction button.</li> </ol> <b>System</b> <ol style="list-style-type: none"> <li>1. The application is opened.</li> <li>2. Application opens the login page.</li> <li>3. System then accepts the user if their credentials are correct/</li> <li>4. The application then logs them in.</li> <li>5. The prediction page is opened.</li> <li>6. The system then computes a prediction.</li> <li>7. The prediction is displayed on the screen.</li> </ol>
Exceptions	<ol style="list-style-type: none"> <li>1. If the user is not connected to the internet, then the user will be prompted to reconnect.</li> <li>2. If the user's team is not stored then a prediction cannot be made.</li> <li>3. During the prediction if the user disconnects from the internet they will be prompted to reconnect.</li> </ol>

TABLE 4.4: Use Case Four

Use Case Name	User updates their team.
Description	A user would like to change their team if they have a made a transfer.
Actors	Any registered user of FPL.
Pre-Condition(s)	The user has already opened the application and logged in. The user is connected to the internet.
Post-Condition(s)	The users team has been changed.
Flow Of Events	<b>Actor</b> 1. The actor opens the Application. 2. The actor enters their email and password to log in. 3. The actor clicks the log in button. 4. The Actor moves to the team page. 5. The Actor makes changes where applicable <b>System</b> 1. The application is opened. 2. Application opens the login page. 3. System then accepts the user if their credentials are correct/ 4. The application then logs them in. 5. The team page is opened. 6. The system then saves the new team
Exceptions	If the user is not connected to the internet, then the user will be prompted to reconnect.

TABLE 4.5: Use Case Five

### 4.3 Risk Assessment

TABLE 4.6: Risk Assessment Matrix

Frequency/ Consequence	1-Rare	2-Remote	3-Occasional	4-Probable	5-Frequent
4-Fatal	1				
3-Critical	3		2		
2-Major	6				
1-Minor	5		4		

#### 4.3.1 Risk One - Gathering FPL Data

- Consequence - Fatal
- Frequency - Rare
- Explanation - With the gathering of data to train my models the API could fail, or the API may not return the exact type of data we are looking for. If so, we have a backup repository which updates the data nearly every day. So, if we cannot use the API, we can use the repository to gather the data.
- Mitigation - is to use a backup repository which allows externally use of their data. This will then mean we do not gather the data myself and will be getting it from someone who has already gathered said data.

#### 4.3.2 Risk Two - Training Model(s)

- Consequence - Critical
- Frequency - Occasional
- Explanation - Google Collab only allows user to use the virtual machines in a 12-hour time slot. The training of the models may fail in that time or may come across a bug, or the virtual machine may fail. If so, then the window to train the models would have passed. As the models must be trained with every update of the data then this could happen on occasion.
- Mitigation - To mitigate this we will just need to train the models locally on my own machine, this will take a little longer than the virtual machines provided but does mean there is less chance for failures and bugs.

#### 4.3.3 Risk Three - Converting Model(s) to Tflite models

- Consequence - Critical
- Frequency - Rare
- Explanation - As we are not too educated on Tensor flow and tensor flow lite models, we are unsure to whether the models we may use will be able to be converted to Tensor flow lite models in order for them to be used on the mobile application. This would lead to a severe outcome.
- Mitigation - we would then have to change how my application and machine learning models are hosted so that they can support other model types.

#### 4.3.4 Risk Four - Continuous Data integration

- Consequence - Minor
- Frequency - Occasional
- Explanation - We will rely on Jenkins to continuously update the data within the data base on firebase. Such as the machine learning modes and the player data. As the machine learning models change on a daily basis like the player data this will have to be updated accordingly.
- Mitigation - We will have to manually update both the models and the player info instead of relying on Jenkins to update it.

#### 4.3.5 Risk Five - Software deploying with Bugs

- Consequence - Minor
- Frequency - Rare
- Explanation - If the software is not tested fully to ensure that bugs are not within the application then there may be bugs and failures within the application. This will cause crashes and the user experience will be poor.
- Mitigation - To ensure that this does not occur then we will enable plenty of testing across both the mobile application and for the machine learning. Ensuring that all possible reasons for a crash are tested and all exceptions are handled.

#### 4.3.6 Risk Six - Mobile application with no internet connection

- Consequence - Major
- Frequency - Rare
- Explanation - If the phone has not connected to the internet then it cannot check to see if the machine learning algorithm updated and if the database updated. If this is not the case then a user could be viewing old data and may not be getting correct.
- Mitigation - Only give the functionality for running the algorithm if it is connected to the internet and has the most up to date data on start up. If not, then give the user a prompt to say they are off-line so access will be limited to the application and would need to reboot before they could regain full access.

### 4.4 Methodology

Methodology will be broken down into three key development ideologies that will drive the development of the application. These techniques are –

- Dedicated Information Gathering
- Waterfall, a workflow system
- Gantt Charts
- Feature Driven Development
- Jenkins and CI Learning

#### 4.4.1 Dedicated Information Gathering

Firstly, we take time to gather as much information as possible on how Fantasy premier league points are accumulated. We would look at how points can be gained in all positions. Seeing what statistics are essential for each player and their respective positions. As already have an extensive knowledge base in this, we can look deeper to see which statistics lead to better points instead of just looking at for example this defender has more clean sheets meaning he gets more points. Instead, we would look at why this defender gets more clean sheets, is it due to having an excellent defensive line? Or is it down to the way the team press and hold possession?



### 4.4.2 Waterfall

The waterfall methodology is a linear project management approach. It is used when there is a set of requirements established at the start of the life cycle. It follows 5 steps which are;

- Requirements
- Design
- Implementation
- Verification
- Maintenance

#### 4.4.2.1 Requirements

The main aspect of waterfall is that requirements are met before any work has been done while developing. All requirements for the project have been made clear at this stage so that development can start without being change mid development.

#### 4.4.2.2 Design

Design is broken down into two different sub sections. Logical design and physical design. The logical phase is when possible ideas are brainstormed and theorized. Physical design is when you decide the schemas and theoretical ideas into actual specifications for the project.

#### 4.4.2.3 Implementation

This is where the programmers take all the requirements and designs from the last two phases to implement. All requirements should be met here in the implementation phase.

#### 4.4.2.4 Verification

This is where the receiver of the project is to review the final project to see that all requirements are met. If all requirements are met, then the project can move forward to the next phase and if not then they move back to the requirements phase.

#### 4.4.2.5 Maintenance

If verification is completed, then maintenance is the next step in the waterfall methodology. This is after the project has been released to the target consumer and all bugs/updates are completed while not adding additional requirements.

#### 4.4.3 Gantt Charts

During development, we will need to track what work has been done and what will be done. We will use Wrike to track the development over the course of next semester. Wrike is a platform used during waterfall development to track all features and how far through development we are at any time.

#### 4.4.4 Feature Driven Development

To work in waterfall the requirements must be broken down into features. These features will be broken down accordingly. Each of these features must be broken down so that they can be worked on incrementally. Each functional requirement will be broken down into key features to break down features into smaller manageable segments to drive development.

#### 4.4.5 Jenkins and CI Learning

To use Jenkins and Continuous Integration we will need to work through several courses before the semester starts to understand how to update and change both the player data and the ML Algorithms when the player data is trained. These pipelines will then run at a set time, to train the models and update them in Github and firebase.

### 4.5 Implementation Plan Schedule

As working with feature-driven development, the project will be broken up into segments and work on several segments at a time. The semester weeks will be used as guideline plus the two weeks before the semester starts. We will break the project down into three core features or epics.

- Android Application Development.

- Machine Learning Algorithm Development.
- Communication Platform and Algorithm.

Each of these steps will take between three to five weeks to complete. These steps' core features will be divided into smaller and more actionable features to allow for a fast development life cycle.

#### 4.5.1 Android Application

Due to the size of the android application, it will be developed in two stages. One being before the semester starts to up to and including the second week of the semester. The features that will be developed in the first stint will be;

- Authentication with Google account and email.
- Page(s) created to input a Manager's team.
- Input players data into the firebase database for a read of the application.
- Create a Jenkins pipeline to update player data when it is updated.

The second stint for android application will be done after the machine learning model is made and this development time will go from week five to week eight of the semester and the features developed will be;

- Allowing a user to input their team number and accessing their team through the Fantasy Premier League API.
- Improving the overall user experience, improving how the UI looks.
- Ensuring the user can delete their account, along with all record of them within our authentication service.

#### 4.5.2 Machine Learning Algorithm

Due to the complexity of the algorithm we will leave four weeks of the semester from week three up to and including week five of the semester. The development goals are;

- Create an algorithm to select the optimal transfer for a manager's team.
- Convert this algorithm into a tflite file so that it can be stored on a mobile device.

- Create a Jenkins pipeline to train the algorithm when the player data updates and then place that algorithm to the firebase storage area.

#### 4.5.3 Machine Learning algorithm and Application Communication

To close the semester we will develop the algorithm into the mobile application so users can use the functionality of the algorithm. The development here will last from week eight up to and including week twelve of the semester. The development features here are;

- Place the tflite file onto firebase.
- Get the application to make predictions using the model hosted on firebase.
- Create predictions for each manager that uses the application.

### 4.6 Evaluation

Headings that we will use to evaluate the success of the project are – Was the minimum criteria met by this project?

- Were all the core features achieved by this project? If not how many of the core features were et by this project?
- Did we solve the question that this project set out to solve?
- Was the project expanded on? Were we able to meet more than just the core requirements?

First, we need to look at whether we met the minimum criteria set out at the start of the project. This was to create a machine learning algorithm to predict the correct transfer to maximize points potential for the manager. At the minimum we want the user to be able to input their team and be able to get an out put of what transfer they should make this would meet the minimum requirements for this project. Once this is achieved then we can focus on creating external features such as the automated process of updating and training the algorithm when the player data changes. As we will be using feature driven development, we will be working on features for a set time. Features can then be broken down into core features and extra features. The core features being the ones that need to be completed such as the machine learning algorithm to make predictions and the mobile application for that machine learning algorithm to run on. These two

are the key features in the development cycle. We can then look at a broader picture into features such as;

- Creating an authentication page so that now users of the applications can log in and view their saved data, they can also then make predictions every week without having to resubmit their entire team.
- Creating Jenkins pipelines so that when the player data is updated that the models can be retrained and updated on the hosting service and also a pipeline so that the player data is accurately reflected within my database.
- Creating a new machine learning model to predict the optimal team for each week, to then show this week on the application for users to see.

With the wide array of features we must not lose sight of the main problem we are trying to solve. Which is to create a platform for users to get help in choosing the optimal transfer for their team. If this goal cannot be delivered, then we have failed in our task within this project.

## 4.7 Prototype

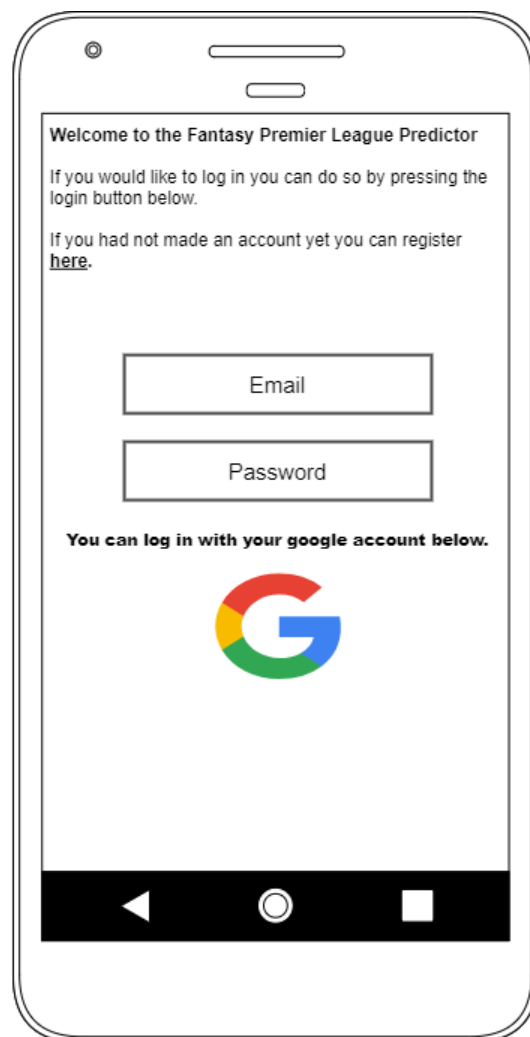
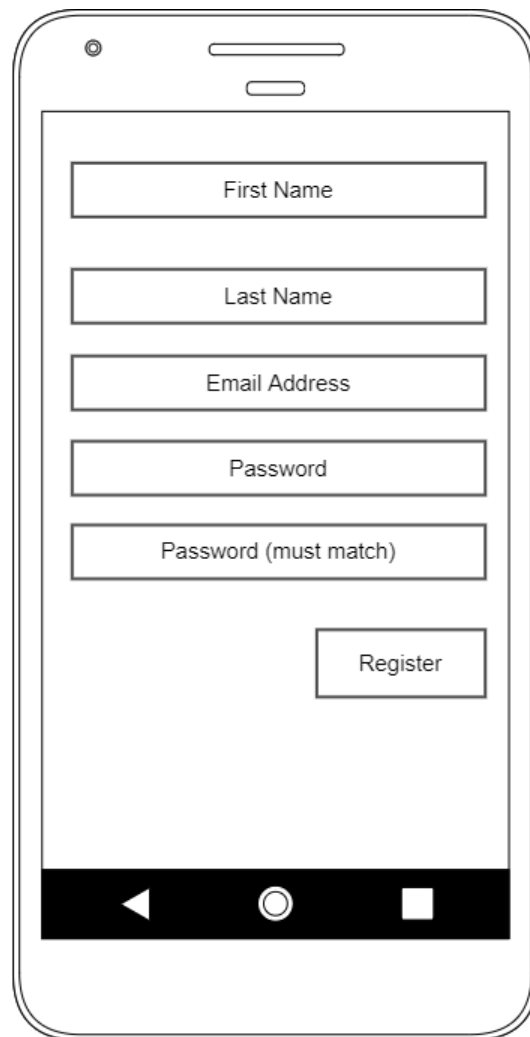


FIGURE 4.2: Login Prototype

The image shows a mobile application prototype for entering player data. At the top, there is a purple navigation bar with four tabs: "Goalkeepers", "Defenders", "Midfielders", and "Forwards". The "Goalkeepers" tab is currently selected. Below the navigation bar, the form contains two identical input sections. Each section starts with a text input field labeled "Goalkeeper 1" and "Goalkeeper 2" respectively, each followed by a downward-pointing arrow icon. Below each name input is a "Sale Price" input field. At the bottom of the form is a "Save Players" button. The entire interface is displayed within a smartphone frame with a black Android-style navigation bar at the very bottom.

FIGURE 4.3: Player Inputs Prototype



A wireframe of a mobile application's registration screen. The screen is enclosed in a rounded rectangle representing a smartphone. At the top, there is a status bar with a camera icon, a signal strength indicator, and a battery icon. Below the status bar, the registration form is contained within a white box with a thin black border. The form consists of five text input fields stacked vertically, each with a label: "First Name", "Last Name", "Email Address", "Password", and "Password (must match)". To the right of the "Password" and "Password (must match)" fields, there are two small vertical rectangles representing volume buttons. Below the input fields, there is a "Register" button. At the bottom of the screen, there is a black navigation bar with three white icons: a back arrow, a home circle, and a recent apps square.

FIGURE 4.4: Register Prototype



UPDATE DETAILS

First Name

Last Name

Email Address

Password

Update Details

FIGURE 4.5: Update Details Prototype



FIGURE 4.6: Delete Account Prototype

## Chapter 5

# Implementation

### 5.1 Difficulties Encountered

Throughout the implementation phase of this project we came across several difficulties and tackled them differently and accordingly. These difficulties are broken up into three sub sections, **Easy**, **Medium** and **Hard**. The easy difficulties were easy to solve and did not cause and set backs, the medium difficulties were not easy to solve but were solved with some time spent and the hard difficulties were to complicated to solve.

- **Easy:**

1. Fixture Gathering. While gathering information to apply to the machine learning we needed to know which fixture is next for each player. The API used would fail if the fixture had not been confirmed. To solve this I stored the ID's of each unconfirmed fixture and ensure the fixture would be omitted while running.
2. Data Sanitation. Due to players being added in real time to the API, these players would have very little stats to their name. So while gathering data concurrently it would crash if any player feature was given a null or **not a number** value. We the had to ensure all null or NaN values were converted to real values while training the algorithm.
3. When creating the test team there was deadlines for each week, teams would need to be confirmed before the game week starts. When optimizing the algorithm and creating a prediction for the team I was under time constraints and if the algorithm was not optimized initially then the team would not be changed until the following week.

- **Medium:**

1. Double and Blank Game weeks. Due to the nature of the premier league there would be clashes with other competitions, meaning games would be cancelled and rescheduled. This would lead to teams having more than one game in a set game week meaning players would have more potential for points as they have a potential of playing 180 minutes instead of 90. Then if game weeks were to be blank certain teams would not be playing this week. There was no way for the algorithm to pick this up as it would chose the next fixture to predict points. Due to how inconsistent these weeks would be this did not affect the functionality but did cause issues for one particular week.

## 5.2 Actual Solution Approach

This chapter will be looking back at the implementation phase and looking back at the research phase, where we created plans for implementing the project. After completing the implementation phase, we will review these aspects and see the reality of the implementation phase over the expectations of the implementation phase.

### 5.2.1 Data Gathering

Data gathering had to be broken into three parts, Fixtures/Results, Teams and Player data. Each data set has features shared so that when we built the algorithm, we could use features from all three. We used the built-in FPL library within python to gather the data. This library directly communicated with the FPL API and allows the user to gather data efficiently.

#### 5.2.1.1 Fixtures and Results

Fixtures and Results data was easier to gather because sanitization was not needed. Fixtures data set had features of:

- ID – This would be the numerical identifier of the match. These would increment in chronological order of when the game is in relation to the rest.
- Away Team – The numerical Identifier of who the away team is in the match.
- Home Team – The numerical Identifier of who the home team is in the match.

Results also share the same features as Fixtures but also includes more match data such as:

- Home Team Score – This would be how many goals scored by the home team.
- Away Team Score – This would be how many goals scored by the away team.

This data is stored and changed every week to reflect time passing. Then moved certain games from fixtures to results, appending the goals scored for each team so the machine learning algorithm can access it down the line.

#### **5.2.1.2 Teams**

This data set changed over time due to changes needed in the machine learning phase. Initially, the features in this set include:

- Team ID – Unique Identifier for each team
- Name – The name of the team
- Short Name – The abbreviation of the team name.
- Strength Overall – This was an identifier to decided how strong/weak the team was in an overall standard from attacking to defending.

The strength overall changes weekly depending on the team's form, but all other data will stay the same week by week. This was added during the algorithm phase as it was needed as a feature.

#### **5.2.1.3 Player Data**

We split player data up into position-based features. As the primary function only offers recommended transfers for outfield positions, we only needed to gather a minimal amount of data for goalkeepers. The features for goalkeepers are:

- Name – The name of the player.
- Code – The unique identifier of the player.
- Team – The unique identifier of the team the player plays for.

- Chance of Playing Next round – A percentage value of whether the player is eligible to play in the next round.

Defenders, Midfielders, and Attackers have a plethora of features, specifically related to the position and core features that relate to the position. While also having the same features of the Goalkeepers as mentioned above. The core features are:

- Cost – The current live cost of the player.
- Points – The points the player received in the last game week.
- PPG – The points per game average for the player.
- Total Points – The overall points the player has received.
- Yellow Cards – The total amount of Yellow cards the player has received.
- ICT – Influence, Creativity and Threat index.
- Goals – Total Goals scored by the player.
- Assists – Total assists scored by the player.
- Penalties Missed – The total amount of penalties the player has missed.

Defenders and Midfielders shared one feature, such as clean sheets, as they both affected the points total. Defenders also have an extra feature that is goals conceded. It affects the points scored if they concede a multiple of two in a single game.

### 5.2.2 Machine Learning Development

When creating the algorithm, we first tried to create an algorithm to predict average points per game per player. This algorithm predicted average points for all outfield players. We created a regression-based algorithm that had a mean error rate of 1.4 points. The predicted points were highly accurate but did not help the player. The average points per game were already a given feature by the API. We made a new algorithm to predict the following potential points in the next fixture to combat this.

We then created three separate regression systems and added features such as overall strength of opposition in the previous fixtures, and mapped it to each player's points in that fixture. Using a K-fold split of 6, we found the best to train the algorithm with a mean absolute error rate of 1.5 points for attackers, 1.1 for defenders and 1.2 for midfielders.

Then predicted each player potential points for the upcoming game week using this algorithm using the fixture difficulty and the fixture team. Predicted points got appended to the data set for later use.

Now that we have the data, we need to get the data to use within the application. We created batch files and ran them manually to update the player data every night. This ensured that data was correct for the application use.

This batch file would run all three algorithms and ensure that the data in the correct format. It would also then write the data to the firebase database so the endpoint application can view this data. All player data is stored. However, the defenders, midfielders and attackers would have a predicted point's values stored with them.

### 5.2.3 Android Application Development

Now that we have cleaned the data, we need to create the functionality to recommend players. The functionality split into three separate parts, like the algorithms. We started the functionality to recommend the optimal transfer for each position. We needed to make sure that the transfer would need to be possible. To ensure this, we had to check that the number of players from a single team does not go over three and the manager has enough money to make the transfer. The functions would return a string value showing the recommendation and the potential points differential for the transfer.

Creation of authentication activities was next. We created a login and registration activity. We only made one way of registration using email and password. The user will need to confirm their email, password, and full name to log in to the app. Firebase would take care of the user's email and password. The database would store all other non-sensitive data.

We then created a menu activity with fragments of Home, Team News, Input Team and settings. Home has the basic introduction information of what the application does and some FPL information. The team news has a lot more information about what is going on currently within the Premier League and some of our own opinions on FPL. Settings have some fundamental setting values like log out, remove the account and give feedback.

Input Team is the main functionality of the application. This page shows the transfers screen which the user will need. It also asks the user to input the total transfer budget they have available.

Once the user proceeds from this activity, they are brought to the goalkeeper's activity, asking the user to input the current goalkeepers with their sell now price. Even though

the functionality does not offer a goalkeeper transfer, we still need to know who is a part of the team not to exceed the max player count per team.

After inputting goalkeepers, the user is brought to each player position activity until all the player's team is filled. Before confirming the players, the user is asked to ensure that the info inputted is correct using a context menu. Once the user has confirmed, it is brought to an output activity which shows a recommendation for each player position and shows the predicted points differential.

## 5.3 Implementation retrospective

### 5.3.1 Schedule Retrospective

We decided that it would take three to five weeks to complete each project section throughout the research phase. We had to adhere to each sub-feature and how long the semester is. Initially, the machine learning and data gathering phase was going to occur in the middle of the semester, however, with heavy reliance on this section. We decided to move it forward to the start of the semester. We were leaving a foundation for the rest of the sections. The mobile application development section had to move back to after the data was gathered and cleaned.

Overall the implementation put in place was adhered to by the authors. The biggest issue was getting an accurate prediction by the machine learning algorithm. We had some unexpected hiccups, but these were easily adhered to and fixed throughout this phase.

- Android Application – Did not fall into its designated time due to Machine Learning Algorithm development taking priority. Its completion time was still within its designated time of five weeks. However, they were done consecutively with each other and not separate as the initial plan outlined.
- Machine Learning Algorithm – This took priority at the start of the semester, so it was not completed in the time slot given but was completed in four weeks like specified. This included data gathering, algorithm refining and automating scripts to put the data in the correct database for use by the application.
- Communication between ML and Android Application. – This was done within the time frame given. It did have a tflite file like initially outlined, as we did not use that type of machine learning. By the end of the semester, we completed this, including the functions to select an optimal transfer for all outfield players.



### 5.3.2 Implementation Retrospective

To begin, getting up and running with a project of this size was more straightforward than expected. We had a strong interest in machine learning and sports predictions, especially in Fantasy Premier League. To our knowledge, a project of this type had not been implemented before. Many similar subjects within the field were studied and implemented. However, this specific recommendation system had not been implemented for FPL users. Once we began implementation, the confidence grew with data gathering within the machine learning algorithm phase.

The first question to pop up was how the machine learning algorithm was going to be implemented. We had first come up with our plan was to create a tensor flow algorithm to load onto the application. This would then take an input of a user's team and then output the optimal transfer. We realized that this would be much work that could be done a lot easier with a simple linear regression system. We then changed the linear regression algorithm from predicting average points per game to predicting potential points for the next game week for all players. This issue caused an extra few weeks to get added to this section of the implementation phase. However, there were a few weeks to take from other sections of the implementation phase.

Tackling the application design process was not as challenging as the machine learning process due to having more of a grounded base in this technology. We spent some time ensuring that the application design was correct and it is functionality correct. Due to the functionality changing in certain technologies in android development such as Gradle and Android studio code, we did come across a few issues but were not fatal to development. They created some extra work to be done, causing android development to take two to three days longer than expected. Nevertheless, due to knowing Android development, it was completed faster than outlined in the plan.

The biggest issue was the covid-19 pandemic and how that affected us during this phase of the project. This led to mental health issues during the phase and caused the authors to get delayed in certain areas. This caused a delay in both personal and work time. With this in mind, we are still very pleased with how this project came out and how helpful the system is.

### 5.3.3 Risk Assessment

In this section, we will be reviewing our risk assessment before the start of the implementation phase. We will see what risks we did experience, how these risks were averted and if we did encounter a risk, how did we resolve it.

#### **5.3.3.1 Risk One - Gathering FPL Data**

Risk Status – Did not encounter.

Explanation – We did not encounter this risk thanks to the API. The API did not fail, and the data was up to date. There was no new update to it and allowed the gathering of data. As a result, the risk was averted.

#### **5.3.3.2 Risk Two – Training Model(s)**

Risk Status – Did not encounter.

Explanation – This risk was not encountered due to the models were able to be trained locally. We did not need to use google collabratory to train these models as they were simple linear regression. As a result, the risk was averted.

#### **5.3.3.3 Risk Three – Converting Models to tflite models**

Risk Status – Did not encounter.

Explanation – This was not encountered as our initial plan was to have these specific models to be loaded onto the application. This was not necessary because of the type of model. We were able to append the output data to the database and read this information from the application. As a result, the risk was averted.

#### **5.3.3.4 Risk Four – Continuous Data Integration**

Risk Status – Did encounter but mitigated successfully.

Explanation – We planned to have a constant update of data to ensure that price changes and player availability was reflected when predicting potential points. This was not a big issue as this data updated twice a day, once in the morning and once in the evening. The batch files was always manually ran at these times to ensure data was correct. As a result, this risk was mitigated successfully.

#### **5.3.3.5 Risk Five – Software Deploying with Bugs**

Risk Status – Encountered but mitigated.

Explanation – We ran into this issue several times, from functional activities crashing, to the main function of the recommendation system offering transfers for a player the user may already have. To mitigate this we ensure that the application was tested thoroughly to ensure that this did not happen for the final product. As a result, this risk was encountered but was successfully mitigated.

#### **5.3.3.6 Risk Six – Mobile Application With no Internet Connection**

Risk status – Encountered but mitigated.

Explanation – With the use of firebase, it stores a cache of the database in the application from the last time it used the database. If a user were to be disconnected from the internet for a long period then they would not get an accurate recommendation, but whats more likely to happen is a user may disconnect from the internet for a short period of time, this would lead to the user having a close to up to date data base allowing for the recommendation to offer an accurate recommendation. As a result, this was mitigated, but can still happen to certain users.

## Chapter 6

# Testing and Evaluation

Some suggested sections (the nature of this chapter should be discussed in detail with your term 2 supervisor):

### 6.1 Metrics

#### 6.1.1 Regression Analysis

- How accurate were the predicted points algorithm?
- What was the mean absolute error for all players?

#### 6.1.2 Function Analysis

- Does the recommendation system work?
- Does it help the below-average user?

### 6.2 System Testing

#### 6.2.1 Regression Analysis

In this section, we will evaluate how accurate the algorithms were. We will evaluate this based on two factors: how close the algorithm was to predicting popular player results using the most selected players in each position. Then getting the mean absolute error for each position over each week.

Mean absolute error is a function to test the accuracy of linear regression algorithms. It is denoted by:  $mae = (\frac{1}{n}) \sum_{i=1}^n |y_i - x_i|$  where  $y_i$  = predicted points and  $x_i$  is actual points for each player.

### 6.2.2 Functionality Analysis

In this section, we will see how well the application does functionality. As we set out at the start of this paper, we are here to improve the below-average user to become a better user by helping them select transfers to optimize points. To do this, we need to look at how it would do overtime against the mean average.

We will do this by creating a test team, making transfers on this team for the weeks available, and seeing how well this team does versus the average for these weeks. As it is a sport, results are not defined by past statistics but by how players perform every week. We will see if it can compete not over one week but for several weeks.

### 6.2.3 Application Analysis

As we want to provide a service to users, we want this to be a pleasant service and help the user make the recommended decisions. We will evaluate this in two ways. Does the application meet the non-functional requirements? Does the application meet the uses cases set out in chapter four?

## 6.3 Results

### 6.3.1 Regression Analysis

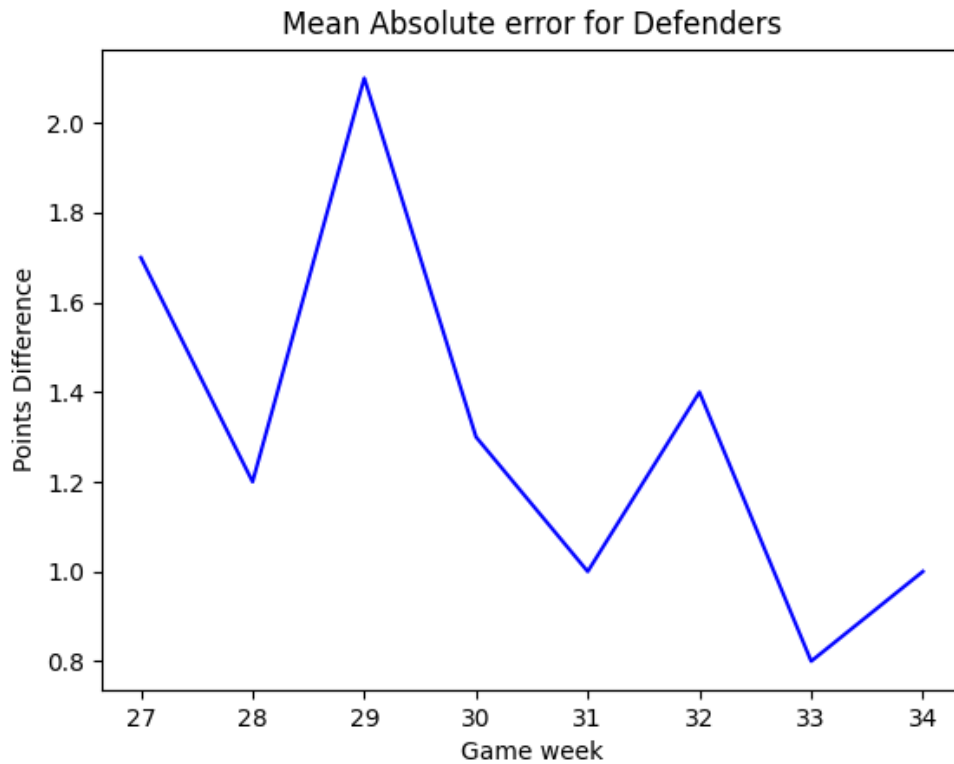


FIGURE 6.1: Defenders Mean Error

First we look at how the algorithms have done over the course of the seven week test period allotted. We can have a look first at the Defensive mean error in figure 6.1. We can see that the mean error may seem high initially and slowly start reducing over time as the algorithm continues to learn. We have some notable outliers in game-week twenty-nine and thirty-two. During these game weeks there was a double game week, where a select number of teams have two games within the one week. This is a common occurrence throughout the testing.

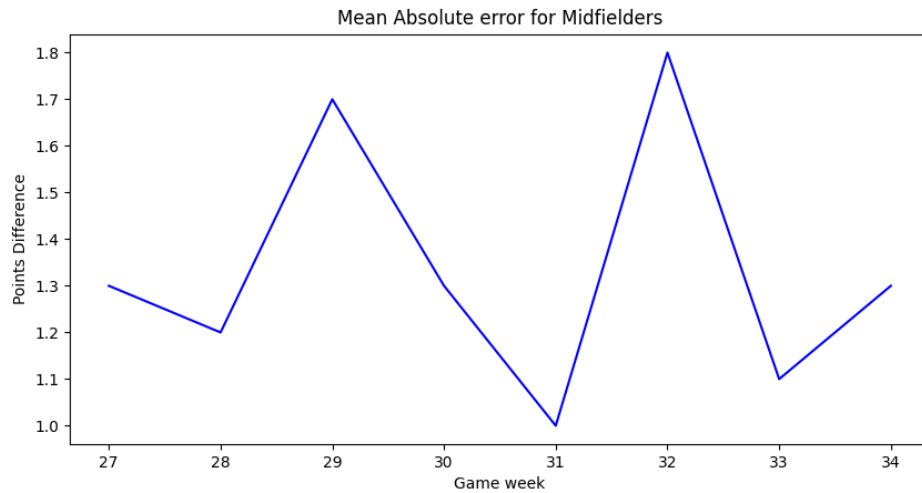


FIGURE 6.2: Midfielders Mean Error

The Midfielder mean error over time also has a similar graph in figure 6.2. We see that the mean reduces from each week that are not affected as a double game-week.

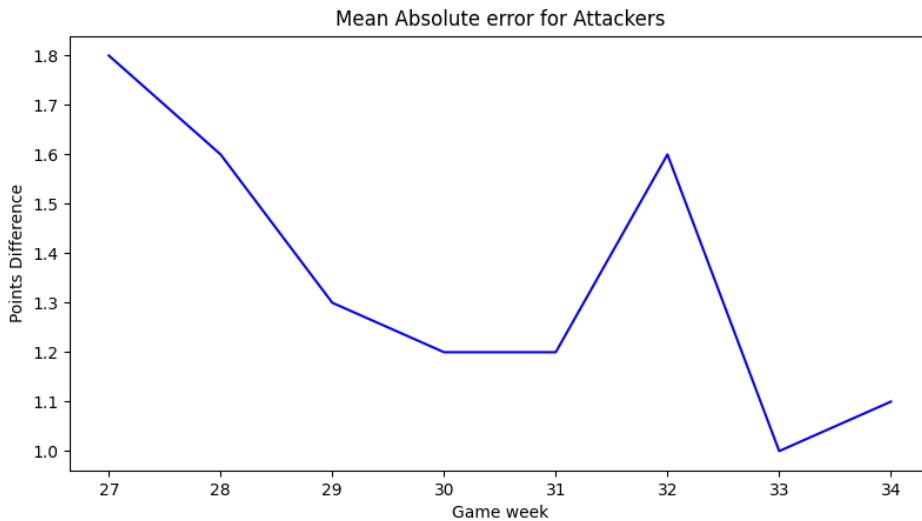


FIGURE 6.3: Attackers Mean Error

The attacking mean error graph does not show this drastic difference as shown in the 6.3. With there being a substantially smaller number of attackers than the previous positions. But as the algorithm learns we can see it consistently improving. We only see a similarity in game-week thirty-two where it spikes. We then also want to highlight specific players who may show more spiratic graphs and how the algorithm learns this. We decided to take one specific player from each position and plot their predicted versus actual points in a graph.

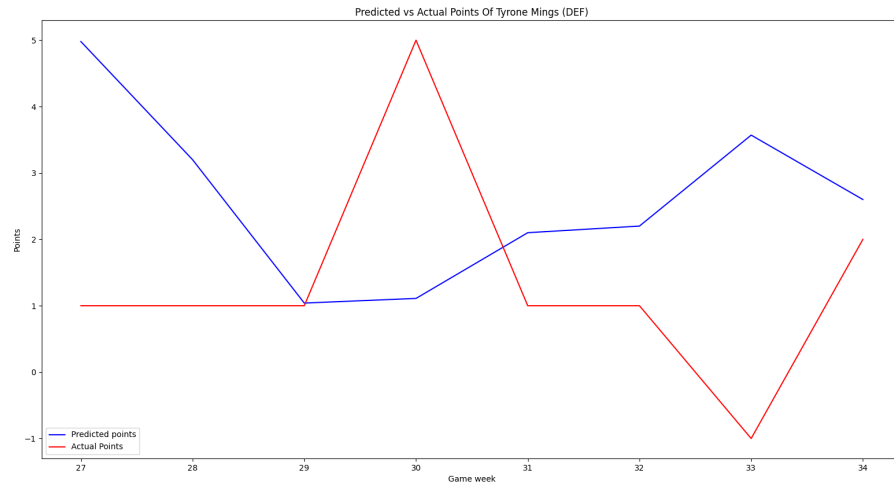


FIGURE 6.4: Tyrone Mings Comparison Graph

As you can see in the graph 6.4. We see that in the opening game week he the prediction is very off. This is due to this specific defender conceding and getting a yellow card resulting in one point. Two points being from over sixty minutes played and minus one for having a yellow card. As we see then the algorithm learns from this and the predicted points falls over the following two weeks to eventually match in game-week twenty-nine. This happens then only for the player to perform well in week thirty as we see then the trend matches that and slowly starts increasing the predicted points. Only then for the player to perform poorly in week thirty-three for the algorithm to adjust to this and near matching in the final week.

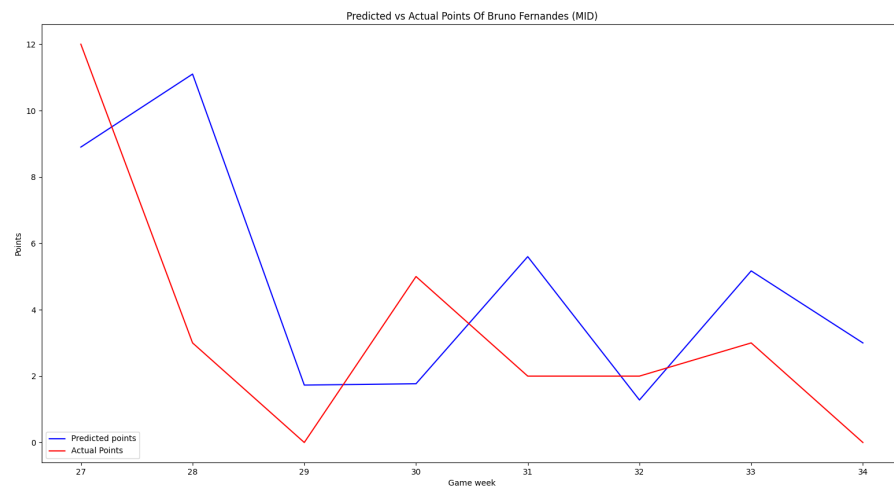


FIGURE 6.5: Bruno Fernandes Comparison Graph



With the graph 6.5 we see that it is a much more varied graph as this player has high potential. Having the most points for any player in this current season. We see the prediction is lower than the actual, which is immediately reflected in the following game week, whereas the player performed quite poor which is then reflected well in the prediction for the next week. We then see a quite consistent differential over the coming weeks, as the error did not go over two points.

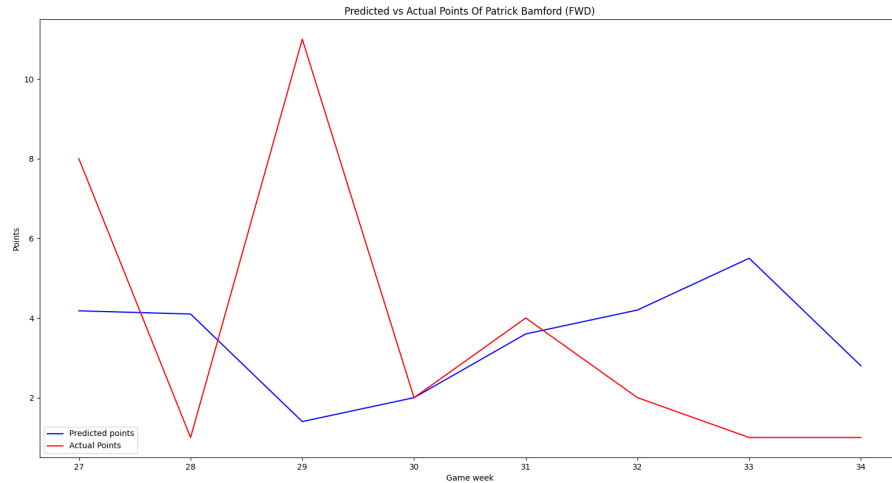


FIGURE 6.6: Patrick Bamford Comparison Graph

With the Bamford graph as seen 6.6, we see a similar issue in the defensive Mings graph 6.4. Game-week twenty-nine we see a drastic increase in points production as it is a double game week. This change is reflected within the following game-weeks leaving the difference to be very accurate over the final 5 game weeks.

### 6.3.2 Functional Analysis

With improving the average player, we created a test team. Making transfers that were purely recommended by the function based on the algorithms. We see that this test team beats the average on a consistent basis. We see the average is beaten in five of the eight weeks in which this was tested, and the test team has an overall score over these weeks of three hundred and sixty one in comparison to the average of three hundred and thirty seven, beating the average by twenty four points. Out of all the teams who had started in game-week twenty-seven, the test team had come in the top twenty-two percent. But a more realistic position is top forty percent with the number of teams created for that week to get max points and players who had given up. This shows that functionally the recommendation does improve the rookie player to become an average player. While this is, we created a test league which the test team came in the top

seventy-five percent. We populated this test league by asking reddit users to join, and as we see in the graph shown, this league outscores the average consistently except for the final week in thirty-four. For each transfer made we recorded results for each player removed and their points scored over the weeks they were removed and compared that to the players brought in and it showed that they players brought in lead to a net gain of 15 points over these weeks over the players removed.

### 6.3.3 Application Analysis

We defined the non-functional requirements in chapter three. They were outlined as such:

- The application must accommodate several screen sizes.
- Information to users must be detailed and give reasoning to why these recommendations were made.

#### 6.3.3.1 The application must accommodate several screen sizes

To do this we ensure that the application was created using dynamic layouts such as constraint layouts. Constraint layouts when implemented correctly ensure that all elements are constructed using ratios to other elements and to borders. When the application is running on a large size screen such as a tablet then many of the activity's elements are not overlapping each other and are more spread out then in comparison to a smaller device such as a mobile phone. On a mobile phone the attributes are shown to be closer to each other and closer to the screens edge.

#### 6.3.3.2 Information to users must be detailed and give reasoning to why these recommendations were made.

After the user gets a recommendation, they are shown why this is by the predicted difference. Each position is shown to output how much potential points difference we could get you for that week coming and each prediction ensures that the user can afford this transfer.

### 6.3.4 Use Case Functionality

#### 6.3.4.1 Registration

As a user opens the application, they are brought to the application log in screen. On this screen they are given an option to login using an email and password or if they do not have an account, they can register by clicking the register button. This opens a screen where the user is prompted to enter in all the relevant details such as email, name, and password. If the password match, then the user has successfully registered their account with the application. The system then brings the user to the home page. If the passwords do not match, then they are prompted this and are asked to re-enter this detail.

#### 6.3.4.2 Login

As a user opens the application, they are brought to the application log in screen. This use case requires the user to already have a registered account to log in with. The user then enters their email and password to the application and if the details are correct, they are brought to the home screen, if not they are prompted that details are incorrect and asked to re-enter.

#### 6.3.4.3 Prediction Made

For step one we ensure that the user has a team gathered. As the user passes through the input team activities, they are prompted to add players into their team, which is stored in a variable to hold these players.

---

```
MT.addPlayer(DEF_Team_one.get(def_one_name.getSelectedItemPosition()), (float) def_one_price.get());
Log.d(TAG, "INDEX 2");
Log.d(TAG, "Price = " + (float) def_two_price.getSelectedItem());
MT.addPlayer(DEF_Team_two.get(def_two_name.getSelectedItemPosition()), (float) def_two_price.get());
Log.d(TAG, "INDEX 3");
MT.addPlayer(DEF_Team_three.get(def_three_name.getSelectedItemPosition()), (float) def_three_price.get());
Log.d(TAG, "INDEX 4");
MT.addPlayer(DEF_Team_four.get(def_four_name.getSelectedItemPosition()), (float) def_four_price.get());
MT.addPlayer(DEF_Team_five.get(def_five_name.getSelectedItemPosition()), (float) def_five_price.get());
```

---

Once all players are stored then we get to the prediction made section. This then loops through each position from defenders to attackers selecting one transfer from each position. To ensure that we keep to FPL rules, we still ensure that the number of players from a specific team is not more than three and the user can afford the transfer in question. This can be shown here in the Attackers function:

---

```

public String Attackers(){
    ArrayList<Player> Attackers = new ArrayList<Player>();
    for(int i = 0; i < Players.size();i++){
        if(Players.get(i).getPosition() == 4){
            Attackers.add(Players.get(i));
        }
    }
    float Differential = 0.0f;
    ArrayList<Player> different = new ArrayList<Player>();
    for(int i = 0; i < Manager_Team.size(); i++){
        float budget = Manager_Team.get(i).getCost() + itb;
        if(Manager_Team.get(i).getPosition() == 4){
            for(int j = 0; j < Attackers.size();j++){
                if(budget >= Attackers.get(j).getCost()
                    & !Manager_Team.contains(Attackers.get(j))){
                    if(Attackers.get(j).getPredicted_points()
                        - Manager_Team.get(i).getPredicted_points() > Differential){
                        different.clear();
                        Differential = Attackers.get(j).getPredicted_points()
                            - Manager_Team.get(i).getPredicted_points();
                        different.add(Attackers.get(j));
                        different.add(Manager_Team.get(i));
                        Log.d(TAG, "We are have found a attacker");

                        Log.d(TAG, different.get(0).getName());
                        Log.d(TAG, different.get(1).getName());
                    }
                }
            }
        }
    }
    Double down = Math.floor(Differential);
    String S = different.get(0).getName() + " has a greater potential than "
        + different.get(1).getName() + " by " + String.valueOf(Differential) + " points";
    Log.d(TAG, S);
    return S;
}

```

---

The user is then returned with three options of a transfer to make and the potential points difference for each of these transfers.

## 6.4 Functionality Demonstration

This section outlines the final application and it's functionality. It shows the work done from the prototype to the final product.

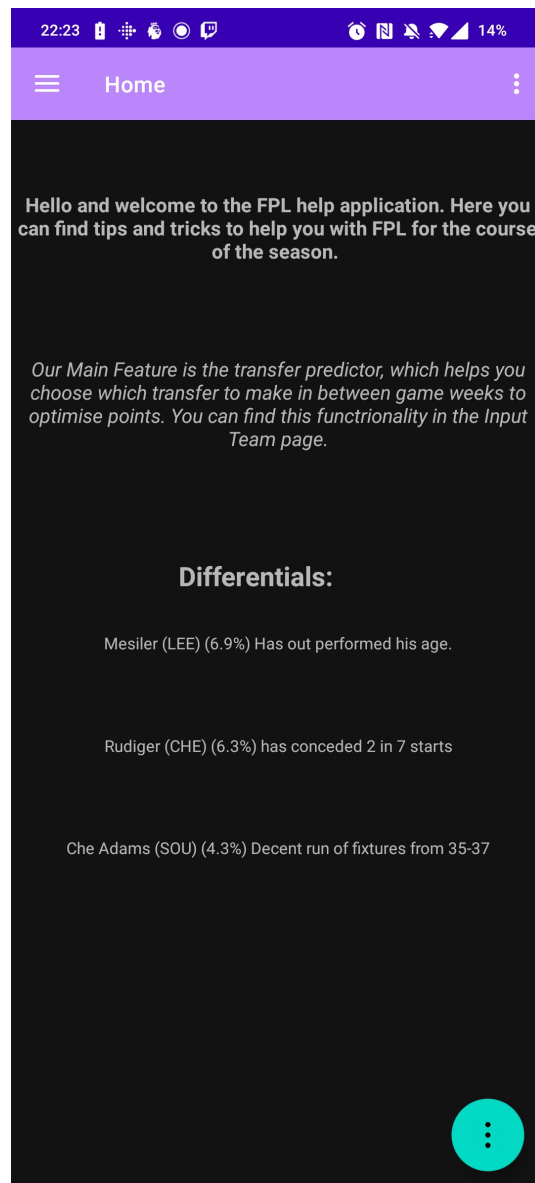


FIGURE 6.7: Home Screen

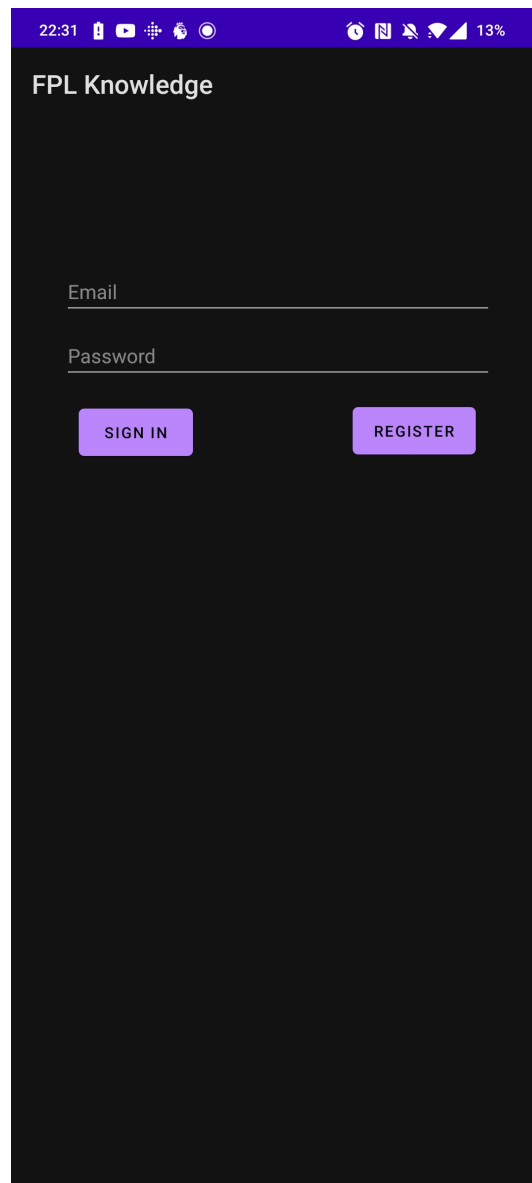


FIGURE 6.8: Login Page

The screenshot shows a mobile application interface with a title bar labeled "Opening\_input". Below the title bar, there is a text prompt: "Please enter how much you have in the bank currently:". A vertical list of numerical values is displayed, ranging from 0.2 to 1.6 in increments of 0.1. To the right of the value 0.8, there is a red rectangular button with the text "ENTER" in white. At the bottom right of the screen, there is a green circular button with a white envelope icon. The background of the app is black.

FIGURE 6.9: Input Players

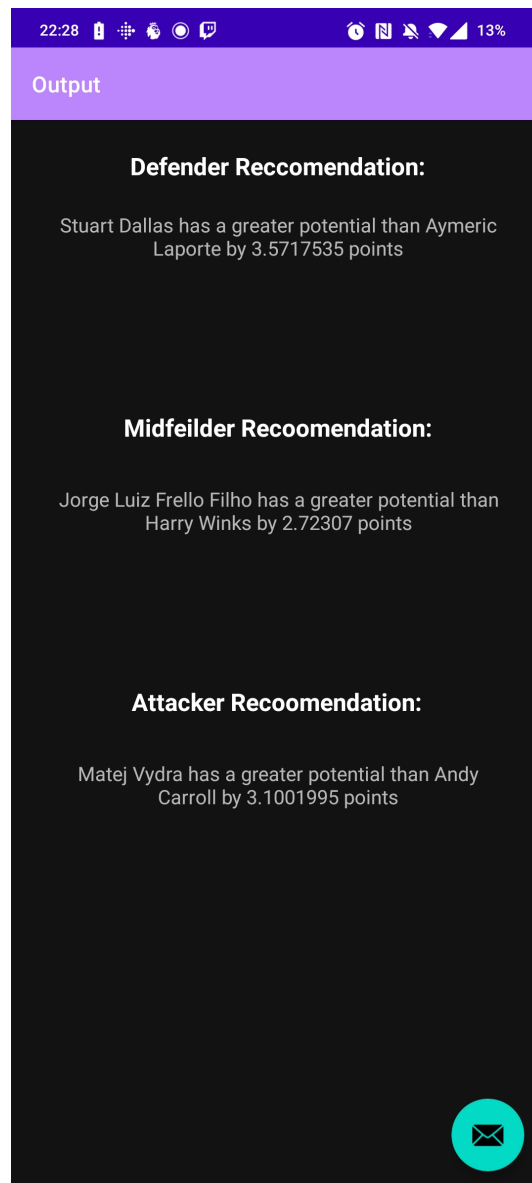


FIGURE 6.10: Recommendation



## Chapter 7

# Discussion and Conclusions

Over the course of this chapter we will look at the product as a whole and review what went well and what we could have done differently. We will also discuss the main conclusions of this project and discuss the potential future for this project.

### 7.1 Solution Review

Throughout this project we set out a goal to help the below average user to become a average or above average user, while keeping them interested in the game over long stretches of time. User's would feel a sense of burn out due to the large amount of variance in user's teams and the fluctuation of points from week to week. We showed in 4 that there is a high amount of variance in players points on a weekly basis. We see in particular defenders weigh heavily on scoring a clean sheet on a week to boost their points production. Meaning how do you plan 5 defenders to have consistent results while still allowing a good team in other places. This is the beauty of FPL.

From our regression results they showed great promise being around two points margin as predicting sports results is not an easy task as there are many variables to contend with. This margin did fluctuate but never showed a greater margin then what we would have deemed to be unacceptable.

Over the course of the test weeks we could see continue improvement as the team would beat the overall player average the majority of the time. While it was beating the average the transfers made showed a net gain on points return the players who were transferred out, meaning that the recommendation did work. It may not of shown a positive points gain week in and week out, but as an overall it was superior to the team the user had initially.

Overall we believe that the solution provided does help to solve the question to improve novice users. It shows that we have implemented a solution to help and aid users. It may not help users become the best overall as this takes lots of knowledge that cannot be represented in statistics.

## 7.2 Project Review

Overall the project was broken into two sections:

- Machine Learning Development
- Android Application Development

These two sections both had an equally pivotal role to play in achieving the solution outlined in 4. The algorithms developed needed to be accurate to ensure a high standard of recommendations. The mobile application that was developed needed to then ensure that the user was able to get the most out of the algorithm developed.

### 7.2.1 Machine Learning Development

We first ensured that we could gather the data needed to test and develop these algorithms and this was made easier by the FPL library for python [13].

If this library did not exist then we would have to use some sort of web scraper to retrieve this data and this would have taken up a lot more time than anticipated. With this library, data was ensured to be up to date at all times and allowed data gathering to be faster.

We then were able to develop an accurate linear regression algorithm for each player position. As time was saved in the data gathering phase it allowed the algorithm time to learn and make accurate predictions. We ran into a few hiccups while developing this as we were not as knowledgeable in this section as we would have liked and there was a lot of learning while implementing. However, the final product in this section was developed well and accurately.

### 7.2.2 Android Application Development

As Android application development is a much bigger section while still having an equal importance to the Machine learning phase. There was a lot more work to do and complete for this area.

Due to this we decided to remove the Continuous Integration development as it was not as important as the other two sections. This allowed more time to go to the application development.

Even with more time allocated to this development we are pleased that functionality requirements were met by the development. We are unhappy with how ascetically the development came out. This was due to how quickly the development needed to be completed.

Many applications are developed by a group of teams of specialists who would focus on certain areas of the development cycle and would be given much more time to do this development. We believe the time to develop an application which is both ascetically pleasing and meets the User Experience Definition [14].

Also with a users team being fifteen players in size there is a lot of inputs to get when asking a user to input their team, leading to many things on the screen and a lot of memory load on the user [15]. To combat this we would maybe use an endpoint of a web-site over the mobile application as there is more room to spread the inputs out and also this opens the endpoint to more than just android users.

## 7.3 Conclusion

FPL being an ever evolving game based on an ever evolving sport, with a growing population every day, there will always be novice players who are new to the game looking to improve. This being the main driving force behind this project idea there are many conclusions derived from this project.

### 7.3.1 Primary Conclusions

- **Accuracy better than expected**

With sport and sport prediction we learned in the background chapter that this can be very hard to predict. Over the course of developing the project we showed that this was possible in a fantasy setting to an accuracy of two points. This really helped the recommendation function and was massively helped by the FPL library.

- **Recommendation improved over time**

Over the course of the test period we saw the recommendation function make better decisions over time as it continued to learn. Many of the players recommended showed a greater points production over time than what it recommended initially. For example in game week twenty eight it had recommended a player who scored

eighteen points over the course of eight games. In contrast to this in game week thirty it recommended a player who brought in thirty-two points in five game weeks. Showing that over time the recommendation improved the long term benefit as well as the short term.

### 7.3.2 Secondary Conclusions

- **The spawn of a potential SaS**

With the success of the solution approach and the recommendation system showing potential to improve teams to the average, the idea has the potential to become a service. As outlined in 2 there has not been a project outlined like this in the FPL market, there have been many similar in other fantasy games such as American football but none within FPL.

- **A new way to look at FPL**

For the users who would be consistent users of FPL, this predicted points score would be a new way to look at statistics given. As we created this project over the course of this current season we have seen our personal team grow drastically over the course of this. Leading us to be winners in our private league and within the top 500,000 out of nearly 7 million overall.

## 7.4 Future Work

Throughout the course of this project, we focused on what transfer is best for each manager's team. We only applied this logic to a single mobile operating system. We would like to further branch the service to iOS and a website meaning all users of FPL can have access to the service.

We want to branch out from just a transfer recommendation and show trends throughout the year, such as nailed on players who would have a higher percentage owned value and the reasoning as to why these players are highly owned. In contrast to that, we would also like to find differential players who would have a low percentage-owned rate but would still perform to a decent standard in the premier league. This would help managers who may be lower in their league to play a different team to their opponents and hopefully get an advantage.

Week by week, the optimal team changes; we would like to use the algorithm to predict which team would be optimal to get the most points for that week. This would help the

users see what team may be weak and target or which players are performing well and to target these players.

We want to improve the algorithm and recommendation service by not just predicting the best transfer to get optimal points for the next upcoming week but for a more extended period, such as several weeks. For example, a double game week may be coming in three weeks. It may be a more significant net gain to bring in players who play twice in that week over the three weeks—essentially improving the net gain positive over multiple weeks instead of just one.

# Bibliography

- [1] “What is CI/CD?” [Online]. Available: <https://www.redhat.com/en/topics/devops/what-is-ci-cd>
- [2] “The Scout, Fantasy Football Tips — Fantasy Premier League.” [Online]. Available: <https://fantasy.premierleague.com/the-scout>
- [3] P. Dolan, H. Karaouni, and A. Powell, “Machine learning for daily fantasy football quarterback selection,” Tech. Rep. [Online]. Available: <http://www.fftoday.com/>
- [4] “(6) Fantasy Premier League.” [Online]. Available: <https://www.reddit.com/r/FantasyPL/>
- [5] G. Sugar and T. Swenson, “Predicting Optimal Game Day Fantasy Football Teams,” Tech. Rep., 2015. [Online]. Available: <https://www.nisivoccia.com/wp-content/uploads/2016/10/115{-}report.pdf>
- [6] A. Nimmagadda, N. V. Kalyan, M. Venkatesh, N. Naga, S. Teja, and C. G. Raju, “Cricket score and winning prediction using data mining,” Tech. Rep., 2018. [Online]. Available: [www.IJARND.com](http://www.IJARND.com)
- [7] H. Kaur and R. Baboota, “Premier League,” *Article in International Journal of Forecasting*, 2018. [Online]. Available: <https://doi.org/10.1016/j.ijforecast.2018.01.003>.
- [8] M. J. Bailey, “Predicting sporting outcomes: a statistical approach,” Tech. Rep., 2005.
- [9] D. Cervone, A. D’Amour, L. Bornn, and K. Goldsberry, “A Multiresolution Stochastic Process Model for Predicting Basketball Possession Outcomes,” *Journal of the American Statistical Association*, vol. 111, no. 514, pp. 585–599, 2016. [Online]. Available: <https://github.com/dcervone/EPVDemo>.
- [10] W. Gu and T. L. Saaty, “Predicting the Outcome of a Tennis Tournament: Based on Both Data and Judgments,” *Journal of Systems Science and*

- Systems Engineering*, vol. 28, no. 3, pp. 317–343, 2019. [Online]. Available: <https://doi.org/10.1007/s11518-018-5395-3>
- [11] W. W. Njororai, “Analysis of goals scored in the 2010 world cup soccer tournament held in South Africa,” *Journal of Physical Education and Sport*, vol. 13, no. 1, pp. 6–13, 2013. [Online]. Available: <http://hdl.handle.net/10950/483>
- [12] N. Danisik, P. Lacko, and M. Farkas, “Football match prediction using players attributes,” in *2018 World Symposium on Digital Intelligence for Systems and Machines (DISA)*, 2018, pp. 201–206.
- [13] A. Bastian, “fpl Documentation,” 2020.
- [14] D. Norman and Nielsen Jakob, “The Definition of User Experience (UX).” [Online]. Available: <https://www.nngroup.com/articles/definition-user-experience/>
- [15] “Cognitive Load Theory: Definition, Types, And Applications For Learning [Guest Post] — Cognition Today.” [Online]. Available: <https://cognitiontoday.com/cognitive-load-theory-definition-types-and-applications-for-learning-guest-post/>

# Appendix A

## Code Snippets

---

```
public String Attackers(){
    ArrayList<Player> Attackers = new ArrayList<Player>();
    for(int i = 0; i < Players.size();i++){
        if(Players.get(i).getPosition() == 4){
            Attackers.add(Players.get(i));
        }
    }
    float Differential = 0.0f;
    ArrayList<Player> different = new ArrayList<Player>();
    for(int i = 0; i < Manager_Team.size(); i++){
        float budget = Manager_Team.get(i).getCost() + itb;
        if(Manager_Team.get(i).getPosition() == 4){
            for(int j = 0; j < Attackers.size();j++){
                if(budget >= Attackers.get(j).getCost()
                    & !Manager_Team.contains(Attackers.get(j))){
                    if(Attackers.get(j).getPredicted_points()
                        - Manager_Team.get(i).getPredicted_points() > Differential){
                        different.clear();
                        Differential = Attackers.get(j).getPredicted_points()
                            - Manager_Team.get(i).getPredicted_points();
                        different.add(Attackers.get(j));
                        different.add(Manager_Team.get(i));
                        Log.d(TAG, "We are have found a attacker");

                        Log.d(TAG, different.get(0).getName());
                        Log.d(TAG, different.get(1).getName());
                    }
                }
            }
        }
    }
    Double down = Math.floor(Differential);
    String S = different.get(0).getName() + " has a greater potential than "
        + different.get(1).getName() + " by " + String.valueOf(Differential) + " points";
    Log.d(TAG, S);
    return S;
}
```



---

---

```
MT.addPlayer(DEF_Team_one.get(def_one_name.getSelectedItemPosition()), (float) def_one_price.get  
Log.d(TAG, "INDEX 2");  
Log.d(TAG, "Price = " + (float) def_two_price.getSelectedItem());  
MT.addPlayer(DEF_Team_two.get(def_two_name.getSelectedItemPosition()), (float) def_two_price.get  
Log.d(TAG, "INDEX 3");  
MT.addPlayer(DEF_Team_three.get(def_three_name.getSelectedItemPosition()), (float) def_three_pri  
Log.d(TAG, "INDEX 4");  
MT.addPlayer(DEF_Team_four.get(def_four_name.getSelectedItemPosition()), (float) def_four_price.  
MT.addPlayer(DEF_Team_five.get(def_five_name.getSelectedItemPosition()), (float) def_five_price.
```

---