OWL Predict

Jude Dillon – B00737902  
BEng Software Engineering

Contents

[1 Introduction 3](#_Toc57907820)

[1.1 Problem Elucidation and Statement 3](#_Toc57907821)

[1.2 Project Aim 3](#_Toc57907822)

[1.3 Project Objectives 4](#_Toc57907823)

[2 Literature Review 5](#_Toc57907824)

[2.1 Initial Investigation on the Project Context Area 5](#_Toc57907825)

[2.1.1 Machine Learning 5](#_Toc57907826)

[2.1.2 K Nearest Neighbour 5](#_Toc57907827)

[2.1.3 Artificial Neural Networks 6](#_Toc57907828)

[2.1.4 The Overwatch League Statistics Map Stats Dataset [17] 7](#_Toc57907829)

[2.1.5 The ELO System 7](#_Toc57907830)

[2.2 Similar Solution Investigation and Analysis 8](#_Toc57907831)

[2.2.1 Real-time eSports Match Result Prediction [21] 8](#_Toc57907832)

[2.2.2 Mel the Rugby Bot [23] 9](#_Toc57907833)

[2.2.3 Findings on using Machine Learning for sports betting [31] 9](#_Toc57907834)

[3 Project Plan & Requirement Specification 10](#_Toc57907835)

[3.1 Stakeholder Identification 10](#_Toc57907836)

[3.2 Justification of Requirement Gathering Methodology 10](#_Toc57907837)

[3.3 Justification of Requirement Prioritisation Strategy 10](#_Toc57907838)

[3.4 System Requirement Specification 11](#_Toc57907839)

[3.5 Justification for Selected Software Lifecycle Methodology 12](#_Toc57907840)

[3.6 Implementation Plan 12](#_Toc57907841)

[3.6.1 Work Breakdown Structure 12](#_Toc57907842)

[3.6.2 Gantt Chart 13](#_Toc57907843)

[3.6.3 Resources Identification 13](#_Toc57907844)

[3.7 Verification Plan 14](#_Toc57907845)

[3.8 Validation Plan 14](#_Toc57907846)

[4 Risk Assessment 15](#_Toc57907847)

[5 Initial Functional Prototype 16](#_Toc57907848)

[5.1 Rationale for Selection of the Risk to be Addressed by the Initial Prototype 16](#_Toc57907849)

[5.2 Design Artefacts for the Initial Prototype 17](#_Toc57907850)

[5.2.1 Visualisation of K Nearest Neighbour Algorithm for the Initial Implementation 17](#_Toc57907851)

[5.2.2 System Flow Diagram 18](#_Toc57907852)

[6 Bibliography 19](#_Toc57907853)

[7 Appendix 1: Code Document 23](#_Toc57907854)

# 1 Introduction

## Problem Elucidation and Statement

Overwatch league is an online esports league which hosts and streams between some of the top teams and players in the world and many people worldwide watch it, in 2020 “Overwatch League (OWL) Grand Finals drew 120,000 concurrent viewers on the esports competition’s live stream on YouTube.” [1] These viewers understand the general hierarchy between teams and as such like to predict the outcomes of games and even whole tournaments before they happen.

Sometimes though there are games with seemingly unpredictable outcomes where a team that before the game looked like they were almost certain to win may perform worse than expected or even lose. This may be due to differences in playstyle or strategy that allows a seemingly worse team to outperform certain teams with generally higher records because their playstyles counteract them.

Often when these outcomes occur viewers are shocked and surprised and wonder if their predictions could have been more accurate and how they could have expected it at all.

OWL Predict will take a different perspective on predicting game outcomes using the data of previous matches and predicting a winner purely based on the data without any human bias involved in the calculation. This could lead to more accurate predictions than even people well versed in the teams and the league could make.

People would be able to choose two teams and OWL Predict would tell them what it predicts to be the most likely outcome so they could compare that to their own predictions to enhance their decision.

The target audience for this product would be any Overwatch League viewers, especially those who like to discuss predictions and strategize how some teams may be able to improve their results with changes.

## Project Aim

The aim of OWL Predict is to create a web-based application which will predict the outcome of an Overwatch League match between 2 teams using a Machine Learning algorithm based on data from previous games played in Overwatch League.

## Project Objectives

|  |  |
| --- | --- |
| Objective number | Objective |
| 1 | Research literature surrounding the project |
| 2 | Investigate similar solutions |
| 3 | Identify stakeholders |
| 4 | Decide on requirements gathering method |
| 5 | Decide on requirements prioritisation method |
| 6 | Specify and explain requirements |
| 7 | Decide on and justify chosen software lifecycle |
| 8 | Make an implementation plan |
| 9 | Risk assessment |
| 10 | Design initial implementation addressing a risk |
| 11 | Build the initial implementation of the prediction system with a smaller data sample |
| 12 | Verify the initial implementation works and demonstrate it |
| 13 | Expand the implementation to the full dataset |
| 14 | Create website and GUI to interact with prediction implementation |
| 15 | Verify it fits the requirements |
| 16 | Validate the product with end users |
| 17 | Finalise and fix any bugs or issues brought up in verification and validation |
| 18 | Hand in and demonstrate final project |

# Literature Review

## 2.1 Initial Investigation on the Project Context Area

### Machine Learning

“Machine Learning is the science of getting computers to learn and act like humans do, and improve their learning over time in autonomous fashion, by feeding them data and information in the form of observations and real-world interactions.” [2] doing this allows them to complete a task without being explicitly programmed to do so.   
In the recent years Machine Learning is becoming far more relevant than it ever has been for three main reasons: We have more data available than ever before being collected at a very fast rate; Computers are getting a lot more powerful than before; and Machine Learning algorithms are also getting much better [3].

There are three main approaches to Machine Learning:   
Supervised Learning  
The computer is given a number of examples of data inputs and what should be output to effectively teach it. It then tries to replicate these inputs and outputs with a rule of some kind. [4] A downside to this is that all the data has to be labelled which can take a large amount of time.

Unsupervised Learning  
Data isn’t labelled so it has to find patterns that were previously undetected with minimum human supervision [5] but this can lead to confusion in how the system gets its results, may be inaccurate and requires a very large amount of data to be accurate. [6]

Reinforcement Learning  
The computer interacts with a dynamic environment and is attempting to complete a goal by maximising its rewards which are effectively a measurement on how well it is completing the task. [7] the main issue with this is that it is only effective in solving certain problems.

### K Nearest Neighbour

The developer has decided on using K Nearest Neighbour for this project because with the time constraints and difficulty of learning Machine Learning as a whole this fits well as it will be more straight forward and flexible than most Machine Learning methods, but should still be accurate.

The K Nearest Neighbour algorithm is a supervised, nonparametric Machine Learning algorithm [8] and is one of the more simple yet robust methods when used for classification tasks. A large advantage of K Nearest Neighbour is that it is a lazy learner [9] [10].  
The way the K Nearest Neighbour works is by finding the shortest distance between the query data and K amount of Nearest Neighbours (the closest data points) and then using these Nearest Neighbours to predicts how the object should be classified based on the most popular response from them [11] [12]. An example of this can be seen in Figure 1 where K = 3 with the solid line leading to a prediction that the colour is red but when K = 5 it instead predicts that the colour is green. One way to get further accuracy with this system may be to use a weighted Nearest Neighbour classifier where Neighbour’s votes are instead weighted higher based on how close they are to the query data [13].

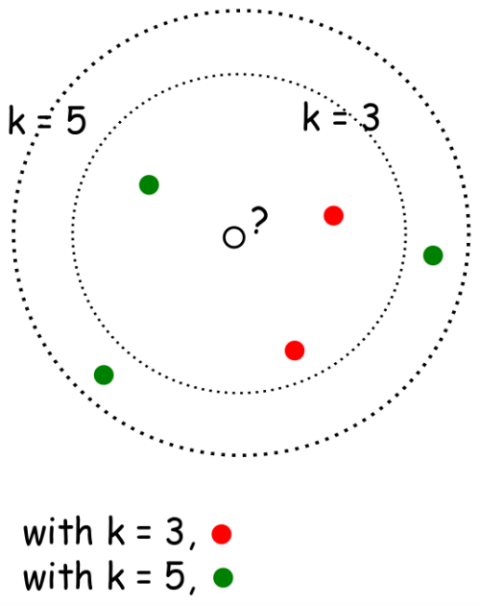


Figure 1 K Nearest N for classification [42]

### Artificial Neural Networks

The developer also considered using Artificial Neural Networks for the prediction of the system but ultimately decided against it due to time constraints of the project which may hinder achieving accurate predictions and complexity of the algorithm which may halt the developers progress in creating the system.

An Artificial Neural Network can be supervised or unsupervised and they are inspired by how actual brains function. [14]. Neural Networks work by taking the input, where each neuron is a component in the input data and introducing new patterns at each hidden layer they are passed down to.   
The processing is done in these hidden layers by using weightings on each of the connections to determine the strength of that connection and the neuron must then do a weighted sum. If the result of this weighted sum is high enough based on the pre-set activation function [15] then the neuron will fire, which will then pass that information to the next hidden layer. The last layer then has one neuron for each output possible and the chosen one is the one with the highest weight. [16]. This kind of Machine Learning requires training, which is done with predetermined data, comparing how close the result of the Machine Learning is to the actual result and then adjusting the weightings for one of the rules. With each adjustment the Neural Network will become more accurate.

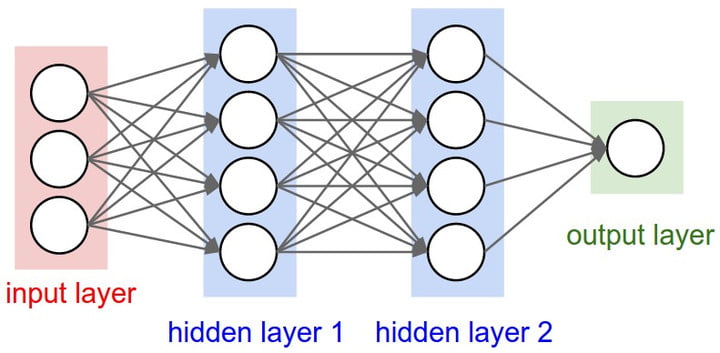


Figure 2 Artificial Neural Network [45]

The main issues with this method are that the training can take a long time to become accurate and due to how the Neural Network decides the output, it can be hard to see how it arrives at its decision. For these reasons, the developer has decided against using it especially how long it may take to train it because of the time constraints of the project.

### The Overwatch League Statistics Map Stats Dataset [17]

The developer of OWL Predict plans on using this dataset because it is the official data from Overwatch League games and specifically the Map Stats dataset as it includes data for the whole team rather than player specific data. Although the player specific data could benefit prediction accuracy by a large amount it also would require a lot more data to be processed for each prediction to be made and there is no way to know which players in a team will be playing before the game occurs.   
Some of the restrictions of this dataset are that due to the fact it is in beta it can take a few days after each match to be updated so real time prediction is not feasible which is part of the reason for only predicting from pre-match data. The data is accurate because it is harvested directly from the game by the developers and there are 2 years of data collected to make predictions from. This provides a large amount of accurate data to base predictions on.

The predictors I plan to use in my K Nearest Neighbour algorithm will be harvested from or calculated from this dataset including average win percentage, recent performance, average point differential and an ELO or similar system.

### 2.1.5 The ELO System

The ELO system is a system used to rate players or teams in competition with each other. It was originally made by and named after Arpad Elo as a rating system for chess but is now used for many esports and sports games [18]. It is a mathematical system where all players have an ELO rating which is a number that effectively evaluates their skill. When a player wins against another player, they gain rating, and the other player loses rating. The amount gained and lost is based on which player was rated higher before the game, where if the higher rated player loses they lose more than if the lower rated player lost because they were expected to lose [19].

The developer of OWL Predict plans to use an ELO system or similar one to evaluate each of the team’s skill ratings to allow them to be used as predictors for which team will win in a K Nearest Neighbour Algorithm. It could also be used in combination with the K Nearest N Prediction as there is a formula using ELO ratings that allows for predicting which player is expected to win and a second way to evaluate the prediction could increase accuracy levels.

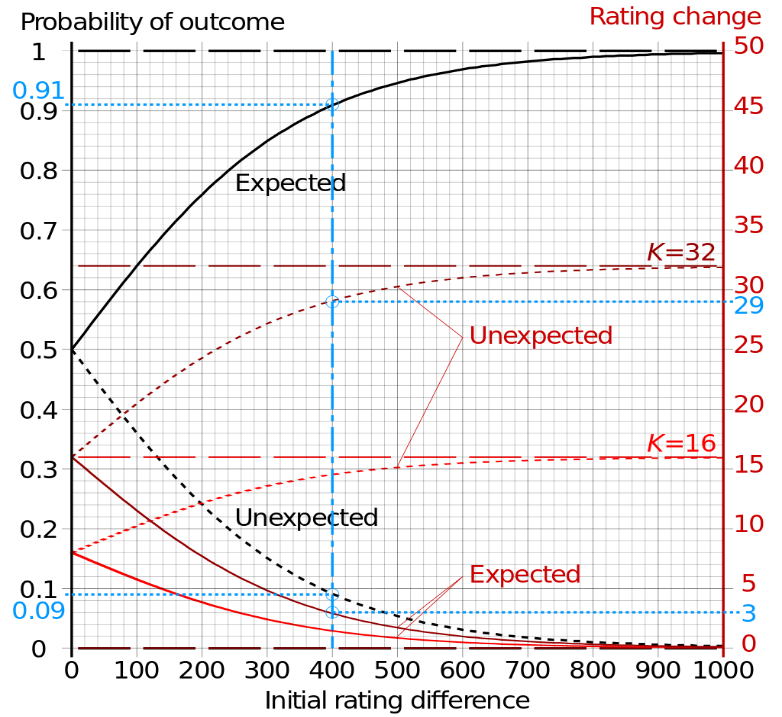


Figure 3 Graphs of probabilities and Elo rating changes (for K=16 and 32) of expected outcome (solid curve) and unexpected outcome [46]

## Similar Solution Investigation and Analysis

After extensive searching into applications using Machine Learning the author has been unable to find any that exist for the purpose of esports [20] results predictions and very few for sports results predictions, so this section’s focus will mostly be on a spectrum of the literature that is most relevant to this project.

### Real-time eSports Match Result Prediction [21]

In this paper the authors attempted to predict the winning team in a match of Dota 2 [22] with Machine Learning algorithms using both pre-match and real-time data. It aligns better with this project than most other reports or papers that the author could find due to the fact that it is also based on esports which may make it more relevant.   
In their paper they stated that with just pre-match data they achieved an accuracy of 71.49% when using Logistic Regression and 70.46% accuracy using a Neural Network, their pre-match data heavily featured data on individual players in the teams and when combined with real time data from games as they occurred could lead to an accuracy of up to 93.73% at the 40th minute of the game.

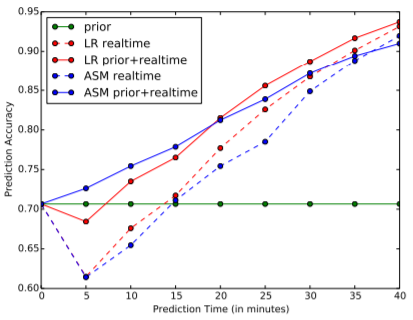


Figure 4 Prediction accuracy plot using different models and features [21]

Issues with this though are that the author of OWL Predict instead plans to predict the outcome of a match before it occurs and gathering data during a match would not be possible due to limitations with the OWL API. Although gathering data on individual players seems like a good strategy for DOTA 2 it seems less feasible with Overwatch League because teams have substitute players and rosters aren’t announced until the game is beginning not to mention that it would also require much more data to be collected.

In conclusion this project achieves an impressively high accuracy but due to restrictions in the data source to be used in OWL Predict it may not be able to achieve similar levels of accuracy but it still reinforces the development theory that accurate results can be achieved with similar Machine Learning techniques.

### Mel the Rugby Bot [23]

Mel the Rugby Bot is a twitter bot called [@mel\_rugby](https://twitter.com/mel_rugby) [24] created by Chris Brownlie which predicts the outcomes and score of rugby game before they happen and then posts them to twitter. To train the prediction he uses a combination of a Principal Components Neural Network [25] and a Linear Support Vector Machine [26]. He then takes an average of the two predictions for increased accuracy. To predict games with a large score disparity more accurately he uses a Lasso Regression Model [27] to get the score assuming there is a large score difference and a Random Forest Classifier [28] to predict if there will be a large score difference. If the system predicts there to be a large score disparity from this algorithm then it will use the score calculated with the assumption of a large score disparity.  
It focuses on using team variables rather than player specific data such as the total number of team caps [29], recent performance, home/away status, and team rankings.   
During the 2019 world cup Mel predicted and posted each of its predictions on twitter before the match occurred and after every game it achieved an accuracy of 86.7% for outcome prediction and an average difference of 12.53 points for score prediction [30].

In conclusion, this achieves an impressive accuracy with the outcome prediction, especially given that it only uses data from before the match occurs which the developer of OWL Predict also plans on doing. It is reassuring to see a similar product working in real world sports predictions, although it uses a number of different Machine Learning techniques that are quite complex to achieve this and this many may be too much for the developer of OWL Predict to implement within the time constraints of the project as a beginner to Machine Learning techniques. It also provided the developer with good knowledge on possible predictors that could be harvested from the data such as average win rate or recent performance.

### Findings on using Machine Learning for sports betting [31]

The researcher mainly used this article to decide upon what levels of accuracy they should aim at achieving for OWL Predict by looking at accuracy levels Machine Learning was achieving in varying sports with differing algorithms.  
The writer of the article Manuel Silverio conducted a literature review on a number of studies which he references in the article and created a chart (Figure 5) showing the accuracy of various algorithms for different sports.

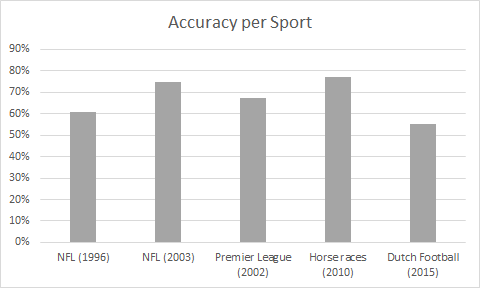


Figure 5 Accuracy of ML predicting models per Sport [31]

In conclusion from the results of Manuel’s literature review the author of OWL Predict would like to achieve an accuracy of at least 60% as most of these studies achieved around this amount so it seems feasible and still more accurate than just a guess may be.

# Project Plan & Requirement Specification

## Stakeholder Identification

|  |  |
| --- | --- |
| Stakeholder | Role |
| Jude Dillon | Project manager/ Developer/ Research/ Tester/ Designer/ User |
| Pat Corr | Mentor/ Advisory/ User |
| Peer Support Group | Peer support/ Advisory |
| Focus Group of Target Users | Advisory |

## Justification of Requirement Gathering Methodology

Initially there was a large amount of brainstorming from the developer, which although it can only see one perspective it provides a good baseline to get initial requirements without spending too much time on it. The developer used a whiteboard to write ideas on and visualise how the Machine Learning algorithm chosen would work.

Outside of this the developer set up a Hothouse [14] with a focus group of 8 target users online via discord [15]. This was decided as a requirement gathering methodology because there are many online communities on discord which are readily available for the developer to get in contact with that fit the target userbase as they are fans of Overwatch League and it was very beneficial to get feedback directly from future users. One downside there may have been was the possibility that there might not have been enough different people in the group so some perspectives might have been missed.

## Justification of Requirement Prioritisation Strategy

The developer chose the MoSCoW prioritisation strategy [32] for the requirements of the project as it is important to prioritise the requirements so the focus of development can be on the more important requirements. Some of the issues within the MoSCoW strategy would be that there is no way to prioritise requirements within a category and deciding which category a requirement fits within can be ambiguous as it isn’t completely clear how to decide a requirement is a must have rather than a should have.  
The developer also considered using the Karl Wieger’s relative weighting [33] model to prioritise the requirements but felt that it unnecessarily overcomplicated things and preferred the more simplistic and intuitive approach that MoSCoW provided.

## System Requirement Specification

#### 3.4.1 Functional Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement | Priority | Reasoning |
| F1 | User can select two different Overwatch League teams | Must Have | The system requires these inputs from the user to do the prediction |
| F2 | System will predict team that will win using Machine Learning | Must Have | The basis of this project is the use of Machine Learning in predicting the result |
| F3 | System will output team that it predicts to win | Must Have | The users need to see the result of the prediction from the system |
| F4 | System needs to be able to extract data used for predictions from the dataset | Must Have | Having to manually extract this data would take too long and be inefficient |
| F5 | System will provide accurate predictions (above 60% accuracy) | Should Have | The chosen accuracy of 60% was from the literature review on similar systems where most of the systems achieved an accuracy of this level. |
| F6 | System will output a percentage stating how sure it is of its prediction | Should Have | It is important to display to the user that there is a level of uncertainty in the prediction as there are many unpredictable factors |
| F7 | Users can tune the range of closest data points that the system uses to make decisions | Should Have | Will allow users to increase or decrease the sample size the data is based off to a degree they think is most reliable for the game |
| F8 | Users can choose which data will be used to make predictions in the system | Should Have | Will allow users to tune the prediction based on the deciders they think are the most relevant |
| F9 | User can access the system through an API | Could Have | Will allow for external sites or tools to get a response from my project |
| F10 | System will predict upcoming matches in advance | Won’t Have | Most users will want to predict the next match that will occur so it would be good to have the result in advance of them asking but at the time of this project there is not an ongoing season so it wouldn’t be possible to test this |

#### 3.4.2 Non-Functional Requirements

|  |  |  |
| --- | --- | --- |
| ID | Requirement | Reasoning |
| NF1 | System will make prediction within 2 seconds | Studies have shown that “the tolerable waiting time for information retrieval is approximately 2 seconds” [34] |
| NF2 | System will be robust | It is important that the system can withstand reasonable usage |
| NF3 | System will be intuitive | The user experience needs to be good and should not require explicit instructions |
| NF4 | System will look visually appealing | This is important to create a good first impression of the project |
| NF5 | System must work on majority of browsers | I do not want to exclude any possible users just because they are using a different browser |

## Justification for Selected Software Lifecycle Methodology

The developer decided on using Kanban [35] as the chosen software lifecycle methodology as the Kanban board [36] will allow tasks to be quickly identified and prioritised and will show progress on each task as work goes on. It means instead of working within time constraints the developer will be able to finish the most important features first and be able to flexibly work around the other features if there are unforeseen difficulties. A possible downside of this may be spending too much time on a single feature so it will be important to split the development time in an effective way. A large amount of the work on this project will be quite new to the developer and choosing the correct values for N in K Nearest N could require a large amount of experimentation and as such that step could require more time than others which suits Kanban well as it allows for a large amount of flexibility in development.  
The developer had also considered using an Agile [37] method but due to the amount of experimentation and research needed in certain areas of the project thought that the flexibility provided by Kanban comparatively would work better for this project.

## Implementation Plan

### Work Breakdown Structure

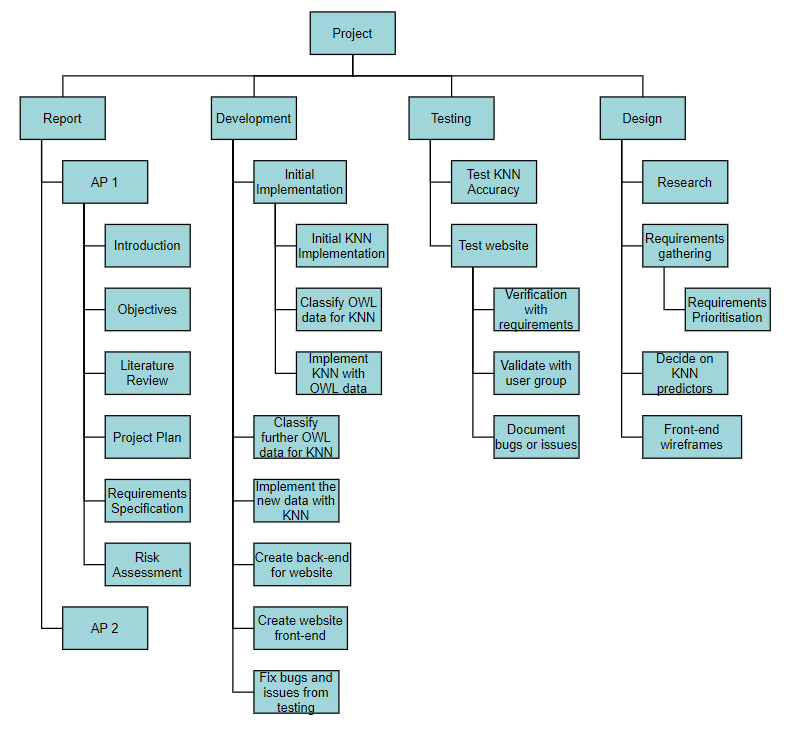


Figure 6 Work Breakdown Structure Diagram

### Gantt Chart

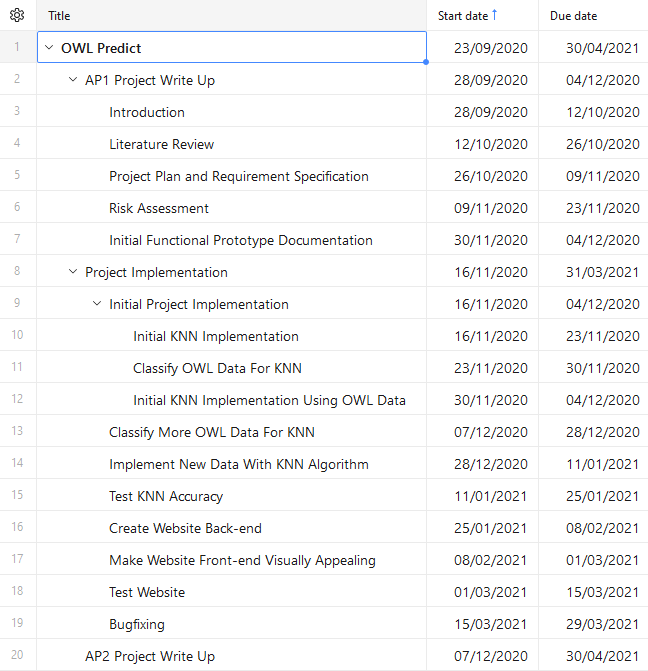
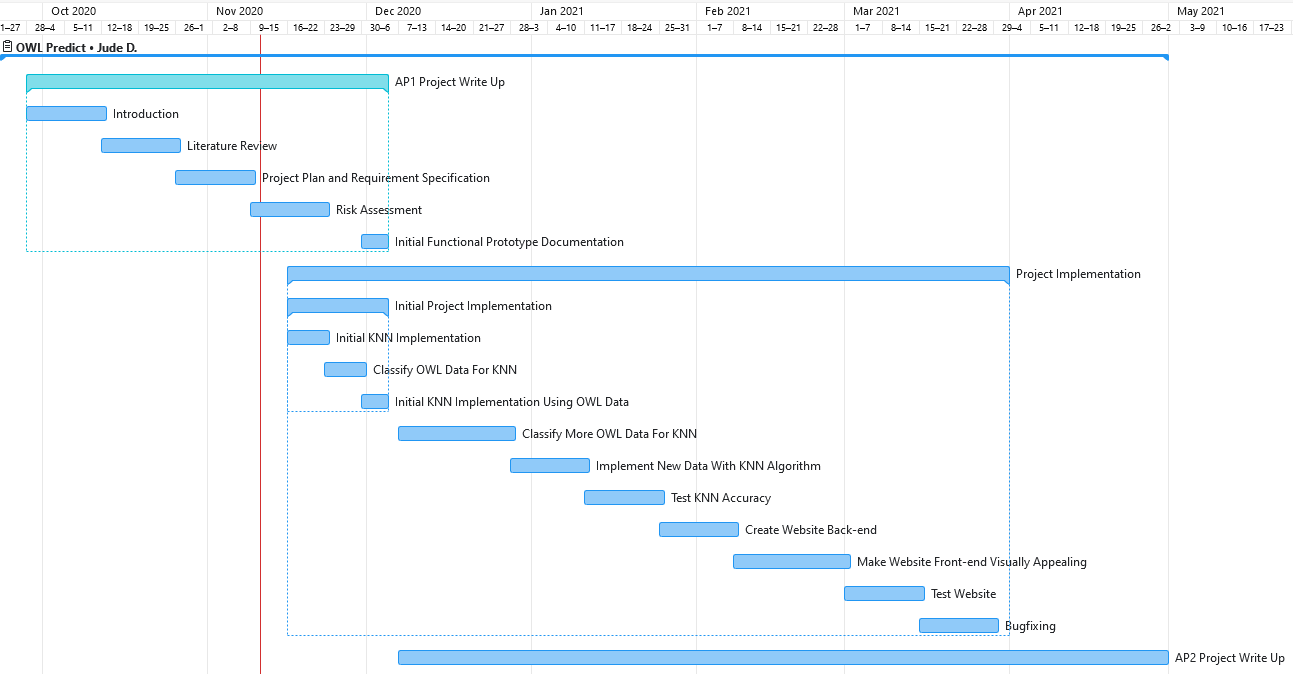


Figure 7 Gantt Chart

### Resources Identification

|  |  |  |
| --- | --- | --- |
| Resource | Type | Use |
| Personal Computer | Hardware | This will be used to complete development of the project |
| Gitlab | Software | This will be used for version control and to back up the project |
| Git | Software | This will be used to access git from the developer’s PC |
| Overwatch League Stats | Software | This is the dataset that the Machine Learning will be based on |
| VSCode | Software | This is the IDE that will be used to develop the project |

## Verification Plan

|  |  |  |
| --- | --- | --- |
| Requirement Tested | Testing Steps | Pass Criteria |
| F1 | 1. Try to select less than two teams 2. Try to select more than two teams 3. Try to select the same team twice | It is not possible to select more than two teams; less than two teams; or the same team twice |
| F2 | This is ensured during development | N/A |
| F3 | 1. Input two teams and get the system to make a prediction | The system outputs a prediction of which team will win |
| F4 | This is ensured during development | N/A |
| F5 | To be evaluated in validation | N/A |
| F6 | 1. Input two teams and get the system to make a prediction | The system outputs a percentage stating how sure it is of its prediction |
| F7 | 1. Change the value of K and make predictions 2. Set K = 1 and make a prediction | You should see the results of the how sure it is percentage and possibly the winner change with different values  For values of K = 1 the percentage on how sure it is of the prediction should be 100% |
| F8 | 1. Change the predictors to be used and make predictions | The results should change based on the changes |
| F9 | 1. Try to make a prediction with the API | The API should return prediction results |
| F10 | 1. Open the site | Predictions for upcoming games should be visible |
| NF1 | 1. Try to make a prediction and time how long it takes for prediction to be output | Prediction should be less than 2 seconds |
| NF2 | To be evaluated in validation | N/A |
| NF3 | To be evaluated in validation | N/A |
| NF4 | To be evaluated in validation | N/A |
| NF5 | 1. Try the system on several browsers | There should be no errors |

## Validation Plan

After each new feature is added, validation will take place in the form of black box testing [38] with my focus group of target users. They will then be interviewed individually to see what they think about the product; feedback both on positive and negative aspects from their perspective; or suggestions for things they would like to be added.

There will also be validation taking place to ensure the accuracy of the Machine Learning. This will be done by using hold-out validation [39]. The developer will hold out the most recent 20% of the data from the system because using the most recent data will most closely emulate the system acting in its real environment predicting a game as if it was to occur at the time of the user asking for a prediction, with only the data it has on previous games. The reason for using 20% of the data is that it would be bad to remove too much data from the dataset as it could negatively affect the results, yet it should still be enough of a sample size to not be skewed [40].

# Risk Assessment

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Risk | Risk Description | Likelihood | Impact | Risk Level | Mitigation Strategy |
| **1** | Inexperience with K Nearest Neighbour may it difficult to create the system within the timeframe allocated for the project | Medium | High | High | The developer will create an initial implementation of K Nearest neighbour using a smaller amount of data and less predictors than the final project will have to gain more understanding of K Nearest Neighbour. |
| **2** | The predictions from the system may be too inaccurate | Medium | Medium | Medium | The developer will conduct testing of the prediction algorithm using an amount of known values to evaluate the accuracy and then make any changes if necessary. |
| **3** | Processing the large amount of data to retrieve the predictors for KNN may take more time than expected which may cause project delays | Medium | High | High | The developer will choose predictors that can be evaluated from the data programmatically so they will not need to manually evaluate each of the matches which would be very time consuming. |
| **4** | May have allocated time to specific parts of the project incorrectly which could lead to some parts being rushed | Medium | Medium | Medium | The developer’s chosen software methodology Kanban allows for flexibility with time management which will allow for more or less time to be allocated when necessary. |
| **5** | Predictions from the system may take too long to be created which could lead to user’s being frustrated and ultimately not using the system | Low | Low | Low | The developer will monitor this risk and if it seems to become an issue will add a message warning the user that the system may take some time to predict. |
| **6** | With the current pandemic of Covid-19, if the developer were to contract this it may cause some time loss which would make it difficult to finish the project within the time allocated | Low | High | Medium | The developer will maintain the appropriate social distancing, wearing of masks, and good hand hygiene as appropriate. |
| **7** | Hardware failure or data loss could lead to the developer’s project files or code being lost which would be a huge setback | Low | High | Medium | The developer will be using gitlab to back up the project and maintain version control digitally as well as storing the files on multiple local hardware devices. |
| **8** | It may be more difficult than expected to create the website and deploy it than initially expected which may cause issues with the project at a planned point near completion | Low | Medium | Low | The developer has in parallel been completing a module which includes creating and designing a website and will use this knowledge to assist in the project. |

# Initial Functional Prototype

## Rationale for Selection of the Risk to be Addressed by the Initial Prototype

In the initial implementation of the project the risks that the developer will be addressing are Risks 1, 2, and 3, the focus with this initial implementation will be concentrated on Risk 1 but both Risks 2 and 3 are closely related to this. The reasoning for addressing these risks are that they are the highest risk levels of my project and addressing them is crucial for the project to succeed because without a good understanding of K Nearest Neighbour the developer will not be able to use it to create an effective Machine Learning system.

The developer plans to use a smaller sample of the data with perhaps only one or two predictors that will be calculated from the dataset automatically by the system such as average win rate, average score, or an ELO rating [41] and then to use these with the dataset to create a smaller scale version of what the Machine Learning will be in the final project.  
This should increase the developers understanding of the K Nearest Neighbour algorithm and how to implement it appropriately. It will also give a good starting point for testing accuracy to see what values can be tweaked to increase the level of accuracy and should provide good guidance on how the data will need to be processed to calculate the predictors.

## Design Artefacts for the Initial Prototype

### 5.2.1 Visualisation of K Nearest Neighbour Algorithm for the Initial Implementation

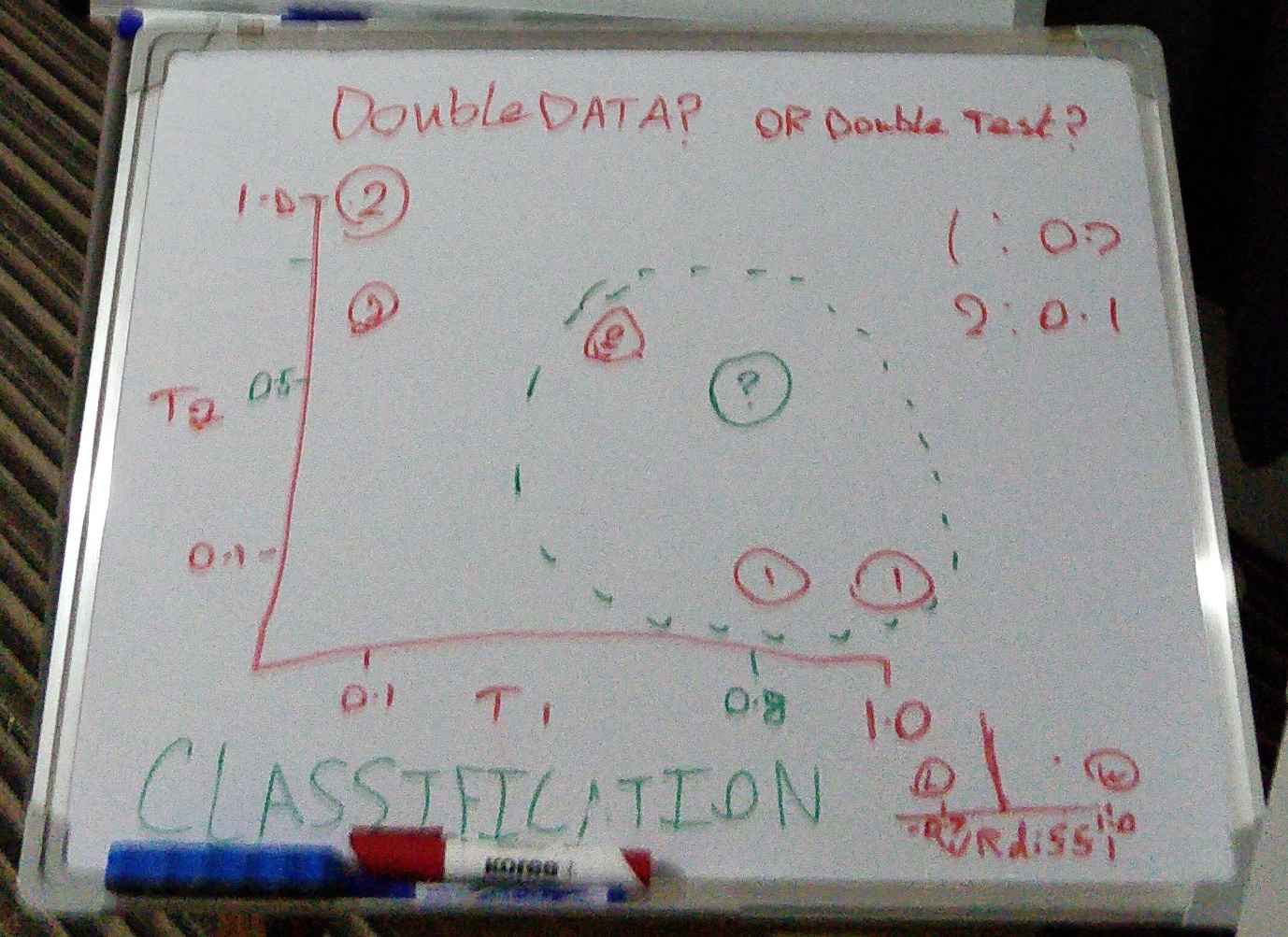


Figure 8 Visualisation of My KNN Algorithm

To help understand K Nearest Neighbour in regards to the initial implementation the developer used a whiteboard to draw out an example of how the data may look plotted into a graph when there is only one or two dimensions as it was planned in the initial implementation.   
In Figure 8 the large central diagram was plotting team one’s win rate (T1) against team two’s win rate (T2) for each match and the small diagram in the bottom right is plotting the difference in win rate between team one and team two for each match game. The developer also took into consideration that for both of these methods there is a possibility that there could be a skew in the data, where team one for example wins more often which could lead to predictions being swayed towards team one. To counteract this, it may be necessary to plot each match that occurs twice, but swapping which team is team one and which team is team two, then getting the win rate difference with this new order as well.  
This will be useful in the development as it allows the developer to better understand what is happening to the data and how it is being used in a visual way.

### 5.2.2 System Flow Diagram

The developer created a flowchart (Figure 9) showing the planned steps that would happen to help in developing the initial implementation. This will be useful as a way to keep track of what needs to be done and in what order to ensure the developer doesn’t get confused during development or do them in the wrong order which may cause issues with the prediction.

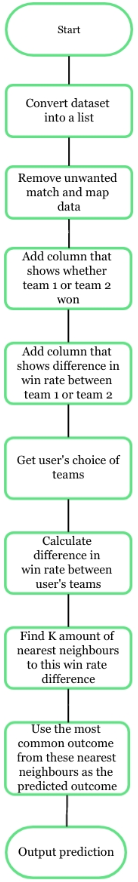


Figure Flowchart of Initial Implementation

# 6 Bibliography

|  |  |
| --- | --- |
| [1] | E. Dixon, “sportspromedia.com,” 2020. [Online]. Available: https://www.sportspromedia.com/news/overwatch-league-grand-finals-youtube-streaming-viewership-2020-esports. [Accessed 28 October 2020]. |
| [2] | D. Faggella, “emerj,” 2020. [Online]. Available: https://emerj.com/ai-glossary-terms/what-is-machine-learning/. [Accessed 28 October 2020]. |
| [3] | The Royal Society, “The Royal Society What is machine learning?,” The Royal Society, 14 June 2017. [Online]. Available: https://royalsociety.org/topics-policy/projects/machine-learning/videos-and-background-information/. [Accessed 03 11 2020]. |
| [4] | P. N. Stuart J. Russell, Artificial Intelligence: A Modern Approach, Harlow;Essex: Pearson Education, 2014. |
| [5] | T. S. G Hinton, Unsupervised Learning: Foundations of Neural Computation, MIT, 1999. |
| [6] | Guru99, “Guru99 Unsupervised Machine Learning: What is, Algorithms, Example,” Guru99, [Online]. Available: https://www.guru99.com/unsupervised-machine-learning.html#10. [Accessed 03 11 2020]. |
| [7] | C. M. Bishop, Pattern Recognition and Machine Learning, New York: Springer, 2006. |
| [8] | N. S. Altman, “An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression,” *The American Statistician,* vol. 46, no. 3, pp. 175-185, 1992. |
| [9] | M. B. &. H. B. Gianluca Bontempi, “Lazy learning for local modelling and control design,” *International Journal of Control,* vol. 72, no. 7-8, pp. 643-658, 1999. |
| [10] | D. W. Aha, Lazy Learning, Springer Netherlands, 1997. |
| [11] | L. E. Peterson, “Scholarpedia K-nearest neighbor,” Scholarpedia, 2009. [Online]. Available: http://scholarpedia.org/article/K-nearest\_neighbor. [Accessed 04 11 2020]. |
| [12] | Cognitive Class, “Youtube: Machine Learning - Supervised Learning K Nearest Neighbors,” 21 march 2017. [Online]. Available: https://www.youtube.com/watch?v=ZBHStob8lhc. [Accessed 04 11 2020]. |
| [13] | C. J. Stone, “Consistent Nonparametric Regression,” *Annals of Statistics,* vol. 5, no. 4, pp. 595-620, 1977. |
| [14] | Y.-H. L. C.-C. K. M.-H. C. I.-H. Y. Yung-Yao Chen, “Design and Implementation of Cloud Analytics-Assisted Smart Power Meters Considering Advanced Artificial Intelligence as Edge Analytics in Demand-Side Management for Smart Homes,” *Sensors,* vol. 19, no. 9, p. 2047, 2019. |
| [15] | B. Wilson, “The Machine Learning Dictionary,” 24 June 2012. [Online]. Available: http://www.cse.unsw.edu.au/~billw/dictionaries/mldict.html. [Accessed 22 11 2020]. |
| [16] | E. Inzaugarat, “Understanding Neural Networks: What, How and Why?,” towards data science, 30 october 2018. [Online]. Available: https://towardsdatascience.com/understanding-neural-networks-what-how-and-why-18ec703ebd31. [Accessed 22 11 2020]. |
| [17] | Overwatch League, “Stats Labs Beta,” 12 10 2020. [Online]. Available: https://overwatchleague.com/en-us/statslab. [Accessed 28 11 2020]. |
| [18] | R. Mittal, “What is an ELO Rating?,” Medium, 11 September 2020. [Online]. Available: https://medium.com/purple-theory/what-is-elo-rating-c4eb7a9061e0. [Accessed 28 11 2020]. |
| [19] | US Chess Federation, “Approximating Formulas for the US Chess Rating System,” US Chess Federation, Crossville, 2017. |
| [20] | M. S. Juho Hamari, “What is eSports and why do people watch it?,” *Internet Research,* vol. 27, no. 2, pp. 1066-2243, 2017. |
| [21] | T. Q. Y.-H. L. Yifan Yang, “Real-time eSports Match Result Prediction,” 10 Dec 2016. [Online]. Available: https://arxiv.org/pdf/1701.03162.pdf. [Accessed 27 11 2020]. |
| [22] | Wikipedia, “Dota 2,” Wikipedia, 15 November 2020. [Online]. Available: https://en.wikipedia.org/wiki/Dota\_2. [Accessed 27 11 2020]. |
| [23] | C. Brownlie, “An Automated Framework for Predicting Sports Results,” towards data science, 3 October 2019. [Online]. Available: https://towardsdatascience.com/creating-an-automated-framework-for-predicting-sports-results-5fe01623d74b. [Accessed 28 11 2020]. |
| [24] | C. Brownlie, “Mel the Rugby Bot,” August 2019. [Online]. Available: https://twitter.com/mel\_rugby. [Accessed 28 11 2020]. |
| [25] | P. Martin, “Understanding Principal Components Analysis (PCA),” neuraldesigner, 2020. [Online]. Available: https://www.neuraldesigner.com/blog/principal-components-analysis. [Accessed 28 11 2020]. |
| [26] | R. Gandhi, “Support Vector Machine — Introduction to Machine Learning Algorithms,” towards data science, 7 June 2018. [Online]. Available: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47. [Accessed 28 11 2020]. |
| [27] | Great Learning Team, “A Complete understanding of LASSO Regression,” Great Learning, 4 September 2020. [Online]. Available: https://www.mygreatlearning.com/blog/understanding-of-lasso-regression/. [Accessed 28 11 2020]. |
| [28] | A. C. Leo Breiman, “Random Forests,” Salford Systems, [Online]. Available: https://www.stat.berkeley.edu/~breiman/RandomForests/cc\_home.htm. [Accessed 28 11 2020]. |
| [29] | Scrum.com, “Rugby Dictionary,” ESPN, [Online]. Available: http://en.espn.co.uk/scrum/rugby/story/94069.html. [Accessed 28 11 2020]. |
| [30] | C. Brownlie, “@mel\_rugby,” Twitter, 4 November 2019. [Online]. Available: https://twitter.com/mel\_rugby/status/1191383972946882560. [Accessed 28 11 2020]. |
| [31] | M. Silverio, “My findings on using machine learning for sports betting: Do bookmakers always win?,” towards data science, 26 August 2020. [Online]. Available: https://towardsdatascience.com/my-findings-on-using-machine-learning-for-sports-betting-do-bookmakers-always-win-6bc8684baa8c. [Accessed 28 11 2020]. |
| [32] | International Institute of Business Analysis, “MoSCoW Analysis (6.1.5.2),” in *A Guide to the Business Analysis Body of Knowledge*, Toronto, International Institute of Business Analysis, 2009, p. 102. |
| [33] | K. E. Wiegers, “First Things First: Prioritizing Requirements,” 1999. [Online]. Available: https://www.processimpact.com/articles/prioritizing.pdf. [Accessed 28 11 2020]. |
| [34] | F. F.-H. Nah, A Study on Tolerable Waiting Time: How Long Are Web Users Willing to Wait?, Association for Information Systems, 2003, p. 285. |
| [35] | D. J. Anderson, Kanban: Successful Evolutionary Change for Your Technology Business, Blue Hole Press, 2010. |
| [36] | K. R. M. John M. Gross, Kanban Made Simple—Demystifying and Applying Toyota's Legendary Manufacturing Process., Amacom, 2003. |
| [37] | J. G. R. C. M. M. B. J. H. S. M. A. v. B. A. H. K. S. A. C. R. J. J. S. W. C. J. K. D. T. M. F. B. M. Kent Beck, “Manifesto for Agile Software Development,” 2001. [Online]. Available: https://agilemanifesto.org/. [Accessed 28 11 2020]. |
| [38] | Software Testing Fundamentals, “Black Box Testing,” Software Testing Fundamentals, 17 September 2020. [Online]. Available: https://softwaretestingfundamentals.com/black-box-testing/#:~:text=BLACK%20BOX%20TESTING%2C%20also%20known,%2Dfunctional%2C%20though%20usually%20functional.. [Accessed 25 11 2020]. |
| [39] | E. Allibhai, “Medium,” 03 Oct 2018. [Online]. Available: https://medium.com/@eijaz/holdout-vs-cross-validation-in-machine-learning-7637112d3f8f#:~:text=Hold%2Dout%20is%20when%20you,model%20performs%20on%20unseen%20data.. [Accessed 25 11 2020]. |
| [40] | R. Nau, “Three types of forecasts: estimation, validation, and the future,” Fuqua School of Business, 18 August 2020. [Online]. Available: http://people.duke.edu/~rnau/three.htm. [Accessed 25 11 2020]. |
| [41] | R. Mittal, “What is an ELO Rating?,” medium, 11 September 2020. [Online]. Available: https://medium.com/purple-theory/what-is-elo-rating-c4eb7a9061e0. [Accessed 27 11 2020]. |
| [42] | S. Everett, “towards data science: Building & Improving a K-Nearest Neighbors Algorithm in Python,” towards data science, 29 march 2018. [Online]. Available: https://towardsdatascience.com/building-improving-a-k-nearest-neighbors-algorithm-in-python-3b6b5320d2f8. [Accessed 04 11 2020]. |
| [43] | H. Stevens, “Street Smart Product Manager: Applying Lean Techniques in a Big Company: The “Hothouse”,” Street Smart Product Manager, 13 June 2013. [Online]. Available: http://streetsmartproductmanager.com/applying-lean-techniques-in-a-big-company-the-hothouse/. [Accessed 07 11 2020]. |
| [44] | D. Delfino, “Business insider: 'What is Discord?': Everything you need to know about the popular group-chatting platform,” Business insider, 26 March 2020. [Online]. Available: https://www.businessinsider.com/what-is-discord?r=US&IR=T. [Accessed 07 11 2020]. |
| [45] | L. Dormehl, “What is an artificial neural network? Here’s everything you need to know,” Digital Trends, 06 January 2019. [Online]. Available: https://www.digitaltrends.com/cool-tech/what-is-an-artificial-neural-network/. [Accessed 22 11 2020]. |
| [46] | Cmglee, “Elo rating system,” Wikipedia, 13 November 2020. [Online]. Available: https://en.wikipedia.org/wiki/Elo\_rating\_system. [Accessed 28 11 2020]. |
| [47] | J. D. Stroud, “THE KANO ANALYSIS: CUSTOMER NEEDS ARE EVER CHANGING,” ISIXSIGMA, [Online]. Available: https://www.isixsigma.com/tools-templates/kano-analysis/kano-analysis-customer-needs-are-ever-changing/#:~:text=The%20Kano%20model%20addresses%20the,excel%2C%20to%20be%20world%20class.. [Accessed 28 11 2020]. |

# 7 Appendix 1: Code Document

from math import sqrt

from csv import reader

def load\_csv(filename):

dataset = list()

with open(filename, 'r') as file:

csv\_reader = reader(file)

for row in csv\_reader:

if not row:

continue

dataset.append(row)

return dataset

def filter\_dataset(dataset, stage):

#only include data where the stage column is equal to the chosen stage

dataset = [row for row in dataset if row[2] == stage]

return dataset

def get\_unique\_match\_ids(dataset):

unique\_matches\_dataset = []

match\_id\_set = set()

for item in dataset:

#if match id has already been seen in the dataset then don't add it to this set

if item[3] not in match\_id\_set:

match\_id\_set.add(item[3])

unique\_matches\_dataset.append(item)

else:

pass

return unique\_matches\_dataset

def add\_team\_one\_win\_status(proto\_dataset):

team\_one\_win = proto\_dataset

for row in range(0,len(team\_one\_win)):

#if the winning team is the team one then status is 1

#otherwise it is 0

if(team\_one\_win[row][5] == team\_one\_win[row][15]):

team\_one\_win[row].append(1)

else:

team\_one\_win[row].append(0)

return team\_one\_win

def get\_winrate(team\_name, dataset, match\_id):

total\_games = 0

total\_wins = 0

for match in range(0,len(dataset)):

old\_match\_id = dataset[match][3]

#if the match has already occured before the selected match or selected match has no match id

#then include it as a match to check

if((match\_id is None) or (int(match\_id) > int(old\_match\_id))):

old\_game\_team\_one = dataset[match][15]

old\_game\_team\_two = dataset[match][16]

old\_game\_winner = dataset[match][5]

#if the team was in the game increase the total games it has played counter

if((team\_name == old\_game\_team\_one) or (team\_name == old\_game\_team\_two)):

total\_games+=1

#if the team won increase the total wins it has counter

if(team\_name == old\_game\_winner):

total\_wins+=1

#calculate the winrate

try:

winrate = total\_wins / total\_games

#if total games is 0 then winrate is 0

except ZeroDivisionError:

winrate = 0

return winrate

def get\_winrate\_differences(dataset):

winrate\_list = []

winrate\_difference\_column\_dataset = dataset

#for each row in dataset calculate difference between team one and team two winrate then add it to the dataset

for row in range(0,len(dataset)):

team\_one = dataset[row][15]

team\_two = dataset[row][16]

match\_id = dataset[row][3]

#get team one and team two winrate

team\_one\_winrate = get\_winrate(team\_one, dataset, match\_id)

team\_two\_winrate = get\_winrate(team\_two, dataset, match\_id)

#calculate winrate difference and add to list of winrate differences

winrate\_diff = team\_one\_winrate - team\_two\_winrate

winrate\_list.append(winrate\_diff)

#add winrate differences list to the dataset

i = 0

for row in winrate\_difference\_column\_dataset:

row.append(winrate\_list[i])

i = i + 1

return winrate\_difference\_column\_dataset

def get\_distance(prediction\_row, data\_row):

#get distance squared and then square root of it to avoid negative numbers

distance = (prediction\_row - data\_row[26])\*\*2

return sqrt(distance)

def get\_neighbors(data, prediction\_data, num\_neighbors):

distances = list()

#get dataset with distances

for data\_row in data:

dist = get\_distance(prediction\_data, data\_row)

distances.append((data\_row, dist))

#sort distances from smallest to biggest

distances.sort(key=lambda tup: tup[1])

#create list of the shortest neighbours for selected amount of nearest neighbours

neighbors = list()

for i in range(num\_neighbors):

neighbors.append(distances[i][0])

return neighbors

def predict\_classification(data, prediction\_data, num\_neighbors):

#get list of the shortest neighbours for selected amount of nearest neighbours

neighbors = get\_neighbors(data, prediction\_data, num\_neighbors)

#create list of the team one win statuses for each of the nearest neighbours

neighbors\_votes = [row[25] for row in neighbors]

#count nearest neighbours votes

team\_one\_win\_votes = 0

team\_two\_win\_votes = 0

for vote in neighbors\_votes:

if(vote == 1):

team\_one\_win\_votes+=1

else:

team\_two\_win\_votes+=1

#choose prediction based on which team had more votes and output percentage of votes that predict this outcome

if(team\_one\_win\_votes > team\_two\_win\_votes):

prediction = "I predict that team one wins by " + str(team\_one\_win\_votes/len(neighbors\_votes)\*100) + "% of the neighbours votes"

else:

prediction = "I predict that team two wins by " + str(team\_two\_win\_votes/len(neighbors\_votes)\*100) + "% of the neighbours votes"

return prediction

def switch(id):

#created a dictionary to use as a switch case

switcher = {

1:"Atlanta Reign",

2:"Boston Uprising",

3:"Chengdu Hunters",

4:"Dallas Fuel",

5:"Florida Mayhem",

6:"Guangzhou Charge",

7:"Hangzhou Spark",

8:"Houston Outlaws",

9:"London Spitfire",

10:"Los Angeles Gladiators",

11:"Los Angeles Valiant",

12:"New York Excelsior",

13:"Paris Eternal",

14:"Philadelphia Fusion",

15:"San Francisco Shock",

16:"Seoul Dynasty",

17:"Shanghai Dragons",

18:"Toronto Defiant",

19:"Vancouver Titans",

20:"Washington Justice"

}

return switcher.get(id, "Invalid team")

#this is the filename of the OWL match map stats csv file

filename = 'match\_map\_stats.csv'

#makes a 2D list from the map stats file

dataset = load\_csv(filename)

#removes data from matches outside of the 2020 Season of OWL

dataset = filter\_dataset(dataset, "OWL 2020 Regular Season")

#removes rows with duplicate matchIDs

dataset = get\_unique\_match\_ids(dataset)

#adds a column that states if team 1 won or lost the game

dataset = add\_team\_one\_win\_status(dataset)

#adds a column that includes the difference between the winrate of team 1 and team 2

dataset\_with\_winrate\_difference = get\_winrate\_differences(dataset)

#The value of K to be used in K Nearest Neighbours

num\_neighbors = 9

print("Atlanta Reign [1]\nBoston Uprising [2]\nChengdu Hunters [3]\nDallas Fuel [4]\nFlorida Mayhem [5]\nGuangzhou Charge [6]\

\nHangzhou Spark [7]\nHouston Outlaws [8]\nLondon Spitfire [9]\nLos Angeles Gladiators [10]\nLos Angeles Valiant [11]\

\nNew York Excelsior [12]\nParis Eternal [13]\nPhiladelphia Fusion [14]\nSan Francisco Shock [15]\nSeoul Dynasty [16]\

\nShanghai Dragons [17]\nToronto Defiant [18]\nVancouver Titans [19]\nWashington Justice [20]")

#get user team choices

team\_one\_input = input("Choose team 1: ")

team\_two\_input = input("Choose team 2: ")

team\_one\_name = switch(int(team\_one\_input))

team\_two\_name = switch(int(team\_two\_input))

#get winrates for each team and calculate difference

team\_one\_win\_rate = get\_winrate(team\_one\_name, dataset\_with\_winrate\_difference, None)

team\_two\_win\_rate = get\_winrate(team\_two\_name, dataset\_with\_winrate\_difference, None)

winrate\_difference = team\_one\_win\_rate - team\_two\_win\_rate

#predict which team will win and display it to the user

prediction = predict\_classification(dataset\_with\_winrate\_difference, winrate\_difference, num\_neighbors)

print(prediction)

#uncomment these lines to create csv of the final state of the dataset

# with open('finalDataset.csv', 'w') as f:

# for item in dataset\_with\_winrate\_difference:

# f.write("%s\n" % item)