# Premier League predictive learning algorithm (PLePA)

# Abstract

Section 1 of this report concerns the actual project covered in this report. The project is to create predictive algorithms to predict Premier League results. It also gives a full account of the literature used and reviews each piece of literature and how they affect the project. The final part of this section is a summary of all the work involved on this project and gives an account of what was involved for each stage of development. It then goes onto compare all the algorithms and discusses what is entailed to complete this project.  
Section 2 of this report is the review, reflection and evaluation of the author and how they feel the project went and what could have been done better and what went well. The plan, resources, skills and risks are all reviewed.

Contents

[Premier League predictive learning algorithm (PLePA) 1](#_Toc50209154)

[Abstract 1](#_Toc50209155)

[Figures 2](#_Toc50209156)

[Tables 2](#_Toc50209157)

[Glossary 2](#_Toc50209158)

[Section 1 – The problem 3](#_Toc50209159)

[1.1 Project Description 3](#_Toc50209160)

[1.1.1 What is the problem 3](#_Toc50209161)

[1.1.2 Why is it a problem 3](#_Toc50209162)

[1.1.3 Benefits and disadvantages of solve the problem 3](#_Toc50209163)

[1.1.4 Scope of the problem 3](#_Toc50209164)

[1.1.5 Format of solution 6](#_Toc50209165)

[1.1.6 Delivery aims 6](#_Toc50209166)

[1.2 Literature review 6](#_Toc50209167)

[1.2.1 Problem related 6](#_Toc50209168)

[1.2.2 Data related 6](#_Toc50209169)

[1.2.3 SDLC 7](#_Toc50209170)

[1.2.4 Algorithm and software research 7](#_Toc50209171)

[1.2.5 LEPSI 8](#_Toc50209172)

[1.2.6 OU courses 9](#_Toc50209173)

[1.3 Work done 9](#_Toc50209174)

[1.3.1 Preparation 9](#_Toc50209175)

[1.3.2 CF1 Setting up the database 10](#_Toc50209176)

[1.3.3 CF2 Develop the random forest algorithm 12](#_Toc50209177)

[1.3.4 CF3 Develop the k-NN algorithm 14](#_Toc50209178)

[1.3.5 CF4 Develop the SVM algorithm 16](#_Toc50209179)

[1.3.6 CF5 Develop the Naïve Bayes algorithm 18](#_Toc50209180)

[1.3.7 Comparing algorithms 20](#_Toc50209181)

[1.3.8 What is completed and what is outstanding 20](#_Toc50209182)

[1.3.9 How to complete project 21](#_Toc50209183)

[Section 2 – Review and reflection 22](#_Toc50209184)

[2.1 LEPSI Review 22](#_Toc50209185)

[2.2 Planning 23](#_Toc50209186)

[2.3 SDLC and project management review 25](#_Toc50209187)

[2.4 Resources 27](#_Toc50209188)

[2.5 Risks 27](#_Toc50209189)

[2.6 Personal review 29](#_Toc50209190)

[2.7 Effective working review 30](#_Toc50209191)

[Bibliography 30](#_Toc50209192)

[Appendices 33](#_Toc50209193)

[Appendix 1 – First draft of schedule 33](#_Toc50209194)

[Appendix 2 – Minor functions in Naïve Bayes 34](#_Toc50209195)

[Appendix 3 – Snippet of tutor communication 35](#_Toc50209196)

[Appendix 4 – Initial resource list 37](#_Toc50209197)

[Appendix 5 – Initial risk list 37](#_Toc50209198)

# Figures

|  |  |  |  |
| --- | --- | --- | --- |
| Figure | Page | Name | Description |
| 1 | 11 | Structured-case lifecycle | An image to show the process of the structured-case lifecycle |
| 2 | 12 | Database conceptual model | An image to show the PLePA conceptual model |
| 3 | 13 | Core tables creation script | Python script to create the database tables |
| 4 | 13 | Main function code for database insert | The main() code for database insert for the Premier League data |
| 5 | 15 | Code snippet of Random Forest algorithm | A small portion of the code to show how each tree is created in the Random Forest algorithm. |
| 6 | 15 | Thresholds of Random Forest algorithm | A graph to show the thresholds of the Random Forest algorithm. |
| 7 | 16 | Correct prediction rate of training data for the Random Forest algorithm | A graph to show the fluctuation of the prediction rate when training the Random Forest algorithm. |
| 8 | 17 | Code snippet of k-NN | A small portion of code to show the code for the main three functions for the k-NN algorithm |
| 9 | 18 | Correct prediction rate for the k-NN algorithm | A graph to show the prediction rate for each neighbour of the k-NN algorithm. |
| 10 | 19 | Plot of training data for the SVM algorithm | A graph to show the plot of SVM training |
| 11 | 19 | Code snippet of obtaining training data and beginning to plot for the SVM algorithm | The code snippet for SVM and a good account of how Figure 10 was produced |
| 12 | 20 | Plot of test data for the SVM algorithm | A graph to show the plot of SVM testing data |
| 13 | 20 | Naïve Bayes function to calculate the probabilities | The main function used for working out the probabilities for Naïve Bayes |
| 14 | 21 | Snippet of main function in Naïve Bayes | A snippet of the main() function for Naïve Bayes |
| 15 | 21 | Insert into CSV file for Naïve Bayes | The code used to insert the algorithm results into a CSV file |
| 16 | 26 | Schedule of PLePA with markings of differences | The full schedule of PLePA with differences from beginning clearly marked |

# Tables

|  |  |  |  |
| --- | --- | --- | --- |
| Table | Page | Name | Description |
| 1 | 6/7 | Project scope | A table to describe each part of the project scope and a current evaluation. |
| 2 | 22 | Table to show the results of the different algorithms used in PLePA split between two and three class classifications | The results of the algorithms shown in a table format. |
| 3 | 22 | Activity evaluation | A table to show the evaluation of the activities, giving detail on where each activity got up to. |
| 4 | 24 | LEPSI review | Review of LEPSI. |
| 5 | 29 | Resource list | A list of resources required for PLePA |
| 6 | 30 | Risks list | A list of risks to be managed for PLePA |
| 7 | 31 | Skills list | A list of skill required for PLePA |

# Glossary

PLePA – Premier League predictive learning algorithm, the name of the project.  
Premier League – The highest league in English football.  
Structured-case lifecycle – The lifecycle chosen for PLePA, it consists of multiple Conceptual Frameworks, described below.  
Conceptual Framework, also known as CF – A conceptual framework is a chunk of the lifecycle, it has a plan, process, analysis and reflection piece.   
LEPSI – Legal, Ethical, Professional and Social issues.  
Random Forest – Random Forest is one of the algorithms developed.  
k-NN – k-Nearest Neighbours, commonly known as k-NN, is one of the algorithms developed.  
SVM – Support Vector Machines, commonly known as SVM, is one of the algorithms developed.  
Naïve Bayes - Naïve Bayes is one of the algorithms developed.  
SDLC – Software Development Life Cycle, it is the process from beginning to end of a software project, there are many different life cycle choices.

# Section 1 – The problem

## 1.1 Project Description

The project is to create a predictive algorithm for Premier League results. It will use historic data; it will use data since the 2014-15 season. It is now a trivial task to add more data if it is deemed needed because the work has been done on creating the Python scripts to insert the data and notes have been made on cleansing the data, which is a couple of simple steps. The difference between this project and similar projects out there is that all the algorithms developed will use the distance travelled for the away team as a key factor.

The aim is to produce four separate algorithms and determine which has the most accurate results. The four algorithms used are Support Vector Machines (SVM), Random Forest, Naïve Bayes and k-NN. The goal is to have around a 75-80% pass rate, this will be a benefit to all football fans who have an interest in statistics, additionally, it could support fans who also place bets on football matches.

If the project is unable to give accurate results after many iterations for each algorithm and tweaking the numbers, then it will serve as research for people who take on a similar project. There is no real issue if it not developed because it is more of a personal project and the author is the only stakeholder.

There are different ways to achieve the results, the aim is to develop four separate predictive algorithms, this will give the opportunity to move on if one is not going well.

The stats data will be stored in a MySQL database and all the algorithm code will be written in Python. All code and documents will be stored in a GIT repository. A stretch goal would be to have an interface for user interaction or to pull the latest set of fixtures from a football website.

### 1.1.1 What is the problem

The problem is the inability to predict football results with a high degree of accuracy. It is very difficult to know which statistics and which algorithms will best be able to predict outcome of the matches, hence, four algorithms have been chosen.

### 1.1.2 Why is it a problem

The struggle to predict the outcome of matches makes things such as betting on football results difficult. It is also interesting for football fans and statistics fanatics to see what the expected result is, and which statistics have determined the likelihood of that result

### 1.1.3 Benefits and disadvantages of solve the problem

The main benefit of solving the problem is that it can be used as a tool for loosely predicting premier league results. It is mainly a personal project so if it remains unsolved there is no negative impact on the wider community.

### 1.1.4 Scope of the problem

PLePA will have four algorithms developed and each algorithm will have a pass percentage rate, possibly with some more statistics. This will depend on what is gatherable. The algorithms will use Premier League data stored in a MySQL database. There are two stretch goals, which have not yet been developed; develop a feed from a sports website to pull in latest fixtures and develop a user interface so PLePA can be distributed beyond personal use. The four algorithms will also be compared to see the difference in prediction results. The full project scope can be seen below in Table 1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Scope | Aspect of problem | Associated tasks | In or out  If out, why out? | Must, should, could, won’t have |
| Preparation stage, plan and make decisions on how the project will be carried out. | The goals give a clear indication of what the project is aiming for. This is useful for keeping the project on schedule. | Define the goals and contents of my project.  Plan the database structure to use.  Look into Python modules that will be useful.  Set up base Python project with GIT version control.  Install database software.  Find the best source for the Premier League statistics required.  Review legal, ethical, social and professional issues and decide if there are additional actions required. | In and completed. | Must have |
| Research additional material that will be useful. | Carry out research so PLePA can make informed decisions. | Research SDLC choices and decide on one for the project.  Research difference between Oracle and MySQL and decide on one of them to use.  Research similar studies for ideas and document key findings.  Research predictive algorithms. | In and completed. | Should have |
| Conceptual Framework (CF) 1 - Set up a database to store the Premier League and algorithm data. | All algorithms will need the data to work, so it is essential work to solve the problem. | Gather required Premier League data.  Cleanse the data gathered.  Create the database.  Create a database reset script. | In and completed. | Must have |
| CF2 - Develop the Random Forest algorithm. | The first of the four algorithms to be developed for PLePA. | Research algorithm.  Plan how algorithm will be coded.  Code algorithm.  Test algorithm.  Evaluate the algorithm.  Produce statistics on algorithm. | In and completed. | Must have |
| CF3 - Develop the k-NN algorithm. | The second of the four algorithms to be developed for PLePA. | As above. | In and completed. | Must have |
| CF4 - Develop the Support Vector Machines, SVM, algorithm. | The third of the four algorithms to be developed for PLePA. | As above. | In and completed. | Should have |
| CF5 - Develop Naïve Bayes algorithm. | The fourth of the four algorithms to be developed for PLePA. | As above. | In and completed. | Could have |
| CF6 - Develop a feed from a sports website to pull in the latest fixtures. | This could allow automation of predicting upcoming results. | Research what websites could be used to pull in the Premier League data required.  Development for the data to come in from the chosen feed. | Out for the scope so far but further work on this project could mean it is developed. | Won’t have |
| CF7 - Develop a user interface so PLePA can be distributed. | This would allow other users to use, this is not essential since it is a personal project and there are no additional stakeholders. | Sketch a user interface for screen/s to use the PLePA algorithms developed.  Create the screen/s which have been sketched. | Out for the scope so far but further work on this project could mean it is developed. | Won’t have |

Table 1 (Blagg, 2020) PLePA full project scope.

### 1.1.5 Format of solution

The code for the algorithms will be developed in Python, this is because it is an established scripting language with modules built in for doing algorithm calculations. Python is also the most familiar programming language to the author. When the algorithm is run, the results are saved into a comma-separated, CSV, file which can be used in excel for creating graphs. If the stretch goals are met, then there will be a graphical user interface for users to input teams and which algorithm they wish to choose. The results outcome would then be shown on the screen.

### 1.1.6 Delivery aims

PLePA will have four main deliverables: Premier League stats stored in a database, a working predictive algorithm to predict the results, an interface for user interaction and a feed from a football website. Each deliverable is discussed below.

* Premier League stats stored into the database must have been complete with a years’ worth of data, ideally it will have three or more years’ worth have data but a best-case scenario, time permitting, there will be five years of Premier League stats stored.
* There will be at least two predictive algorithms developed, this is a must have for PLePA to allow for comparisons between the algorithms. It should be within scope to achieve three, but, also, without any development obstacles, four is easily achievable.
* There are two nice to haves if time goes well, firstly, an interface for user interaction will be developed, there are two parts to this, the first is for the user to load up an executable, select two clubs and they will receive a prediction from the algorithm. The second would allow the user to select the algorithm to predict with.
* The second nice to have is a feed from a football website, the upcoming fixtures would be pulled from a website, the user would select the algorithm and the predicting results would be displayed.

## 1.2 Literature review

All the external material used for PLePA is reviewed below and split into categories.

### 1.2.1 Problem related

Machine Learning Algorithms in Python (Dataflair Team, September 2018) was a useful read, it was a recent article and gave an informative account of developing algorithms in Python. The Dataflair team is a professional and reliable source of information so the training article can be trusted. The Beautiful Game: Predicting the Premier League with a random model (Nguyen, 2018) and Betting on the English Premier League (Campanelli, 2019) were research readings to see other views on solving the same problem. It was helpful to see the different approaches, what went well, and problems that were encountered so it was relevant to PLePA. Both articles were well presented and written relatively recent, so the information presented was timely.

### 1.2.2 Data related

The next two articles provided the data statistics used for PLePA, the first was Data-files: England (Football-Data,2020), this was where the data used for match history stats was collected from, cross checking was performed against the data files provided on some examples to ensure the validity of the statistics. The website also has a link to contact them with any errors which gives you confidence that if errors had been found by other users then the data would have been fixed. The data was fully up-to-date to the latest English football results. The second source of statistics was 2019-2020 Premier League Stats (FBREF, 2020), this was where the stats used for league positions was collected from, they also store historical data, so the last five seasons’ worth of data was also from that website. Again, some cross checking was performed on the statistics provided and everything was in check. The most up-to-date league positions were updated and correct.

### 1.2.3 SDLC

Choosing a Lifecycle Model (The Open University, 2014) gave insight into multiple lifecycles that were available and discussed how to go about choosing one for the project. The document was well presented and, being an OU document, was relevant for PLePA. Further reading into SDLCs was SDLC Models Explained: Agile, Waterfall, V-Shaped, Iterative, Spiral (Osetskyi, 2017), this explained in detailed some of the popular SDLC choices and what they are best for. It was an article by Osetskyi who has multiple useful articles which are highly rated. The article was from 2017 which is not too old considering there are not many newly created SDLC styles. The final piece of literature used for researching SDLC choices was Structured-case: A methodological framework for building theory in information systems (J.M. Carroll et al., 2000), it provided a descriptive insight into a SDLC which was unknown to the author of PLePA, this went on to be the SDLC of choice used for PLePA. The document was not the easiest to follow because it was presented as a wall of text, however, it had a lot of detail and it was well written. It also had an image which helped portray the structured-case clearly. The document was a bit old but the life-cycle is still used today.

### 1.2.4 Algorithm and software research

An Implementation and Explanation of the Random Forest in Python (Koehrsen, 2018), How Random Forest Algorithm Works in Machine Learning (Synced, 2017) and How the Random Forest Works in Machine Learning (Polamuri, 2017) was all research done in preparation for development of the first algorithm, Random Forest. The articles all explained how they had developed the Random Forest algorithm in Python, which is the same development language used for PLePA, so this was suitable research for the project. They are mostly up-to-date, but caution was used because more up-to-date modules may have been developed in Python which could have been used. They were all well-presented, particularly Polamuri’s article; it was engaging and used images to break up the text for the reader. The final research into Random Forest was How to Implement Random Forest From Scratch in Python (Brownlee, August 2019), this piece of literature gave a very detailed account of Random Forest with different ways to implement the algorithm. It was well presented and easy to follow. The style of Brownlee’s writing triggered the author to use more of Brownlee’s work when investigating later algorithms. This reading also triggered the reading of How to Implement The Decision Tree from Scratch in Python (Brownlee, December 2019) because it was useful to learn a little more about Decision Tree development. Again, being written by Brownlee, the layout was very similar and easy to follow with good examples of how to implement. Both of his pieces of literature were written recently and were using up-to-date Python functions.

Next was the research into the k-NN algorithm, the first bit of research was Develop k-Nearest Neighbors in Python from Scratch (Brownlee, 2019), this was well presented with lots of detail on the concepts of k-NN. Brownlee’s articles are useful for explaining the algorithm and ways they could be implemented in Python. It was relevant because it was the algorithm and the scripting language being used for PLePA. K Nearest Neighbor Algorithm In Python (Maklin, July 2019) was another bit of research into k-NN, it was quite basic but had a clear layout which made it easy to follow. It was a good way to really understand the basics of k-NN but did not go into too much depth. It was posted on towards data science less than a year ago with good reviews so assumed to be mostly reliable and up-to-date. 1.6 Nearest Neighbors (Buitinck et al., n.d) provided more information on k-NN, it had a lot of text, so it was a difficult read to focus on, it did have images to break up the text in places though. It was mostly useful, however, there was parts of it that were beyond the scope of PLePA. PLePA did not end up using Scikit but it was good to get an alternative implementation idea. It is widely used by data analysts so can be assumed to be trusted. The final bit of research for k-NN was KNN Classification using Scikit-learn (Navlani, August 2018) which was also on Scikit, it was easier to read and follow than Buitinck’s article on Scikit because it had code examples and pictures to assist the explanations. It is under two years old and post on DataCamp’s website so deemed to be reliable and mostly up-to-date.

Research into the third algorithm, SVM, began with An introduction to Support Vector Machines (SVM) (Stecanella, 2017), it was well presented with a compliment of images with the text. The piece did not go into too much depth so it was a good starting block, there were not loads of reviews on this text so it is hard to say how reliable it is, however, it was posted on MonkeyLearn and the few reviews available were positive. It is timely for the algorithm because there won’t be much change on the core workings of SVM. 1.4 Support Vector Machines (Buitinck, n.d) was the next part of reading, however, similar reviews to 1.6 Nearest Neighbors (Buitinck et al., n.d). It had lots of text and was hard to follow, Scikit was used for this algorithm so the article had useful parts on the module in it. More reading into Scikit, Support Vector Machines with Scikit-learn (Navlani, 2019), this article was valuable for learning more on SVM, particularly since it was more reading in Scikit which was used for this module. It was well presented with a good break up for text, images and code examples. Published in 2019 on datacamp so it was timely and had great reviews. The final bit of research into SVM was Support Vector Machine Python Example (Maklin, 2019) which was mostly well presented, it had a part with a complex calculation which was not explained clearly. Maklin had images to help explain most of his write up. It was posted on towards data science recently with positive community response so it can be deemed reliable and timely.

Research into the final algorithm, Naïve Bayes, was Introduction to Naïve Bayes Classification (Soni, 2018). This really was an introduction, it was very short, however, well laid out and easy to follow with a very positive response from readers. It gave a very good account of the basics of Naïve Bayes. A Gentle Introduction to Bayes Theorem for Machine Learning (Brownlee, 2019) was additional research into Naïve Bayes, it was also an introduction, however it gave a lot more detail and went more in-depth. Everything explained used examples, this text would have benefited by using some pictures to help explain parts of the algorithm. Overall, it was a good account of the algorithm and gave a lot of detail. The final piece of research into Naïve Bayes and algorithms was Naive Bayes Classifier From Scratch in Python (Brownlee, 2019), this article was great for examples, it had an easy to follow layout broken up with pictures to help explain. It was also using Python, so it was relevant to PLePA. It was posted on Machine Learning Mastery with great reviews in 2019 so it is in touch and dependable.

### 1.2.5 LEPSI

Legal, Social, Ethical and Professional issues (The Open University, 2012), The Data Protection Act (UK Government, 2018), Equality Act 2010 (UK Government, 2015), Computer Misuse Act 1990 (UK Government, 1990), How copyright protects your work (UK Government, n.d), What is the freedom of Information Act? (ICO, n.d) and Code of Conduct for BCS Members (BCS, 2019) were all important to read to ensure PLePA was not in breach of any Legal, Social, Ethical or Professional issues. If PLePA was in breach, these articles would have helped to resolve and steer the project in the correct way. They also helped to keep the author up-to-date on the government’s latest laws and guidance. These articles are all strict guidelines presented by the bodies managing the laws and rules, therefore, the literature can be trusted to be correct. They are all also kept up-to-date with any changes to the laws and rules.

### 1.2.6 OU courses

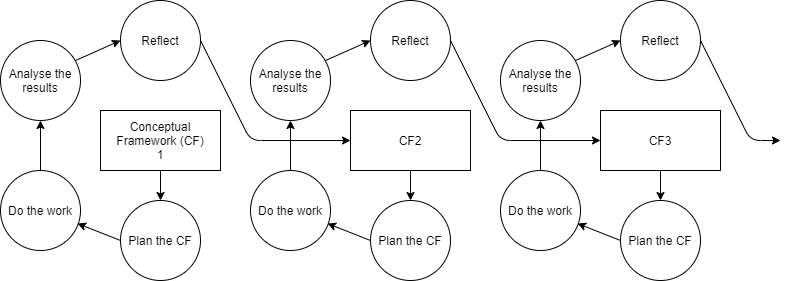
TM351 part 1 -26 (The Open University, 2014) is the Data management and analysis module provided by the OU, this was pivotal and relevant to PLePA because it was the inspiration for the project, and it provided the skills for developing and managing a database and algorithms. It had an easy to follow layout and, because it was provided by the OU, it is expected to be correct, timely and reliable.

## 1.3 Work done

All work done followed the schedule created, the latest version can be seen in Figure 1 below. There were a few adjustments as the project went on, these will also be explained in section 2.2.

### 1.3.1 Preparation

Before any development could begin, the preparation stage had to be complete. The first task to be completed was to set up a GitHub repository for storing all the project work, this ranged from the Python files to the research documents. It was important to keep everything version controlled to ensure no work was lost when overwritten, this allows for things such as the schedule to be monitored throughout the project.   
Once this was setup, the next, important, step was to choose a software development life cycle (SDLC), after doing research on multiple lifecycles (The Open University, 2014)(Osetskyi, 2017)(J.M. Carroll et al., 2000), the chosen SDLC was the structured-case lifecycle. This involves multiple conceptual frameworks (CF) which each have four parts, plan, do the work, analyse and reflect. This can be seen more clearly in Figure 1 below.

  
Figure 1 (Blagg, 2020) Structured-case lifecycle

Once the lifecycle was chosen, the first draft of the schedule could be created, which can be seen in Appendix 1, the latest schedule can be seen in Figure 1.   
After this, the research begun, the first part of research was to look at similar projects (Nguyen, 2018) (Campanelli, 2019), from these projects, it was useful to see things that worked well and did not work so well. One thing that was noticed is that both projects researched only used one algorithm for result prediction so this is one way that PLePA would differ from these. The research into predictive algorithms also began at this point (Dataflair Team, September 2018), it was useful to see ways of implementing the algorithms.   
Next came the search for Premier League data that could be used for PLePA (FBREF, 2020) (Football-Data,2020). The data chosen was from the seasons between 2014 and current day. This data was downloaded and cleansed, it was not a difficult task to find the data to use, the first websites checked had the data required downloadable in csv format. The cleansing of data was mainly to match up the results data with the historic league positions. There were some small differences such as Leicester in the results data and Leicester City in the league position data.   
At this point, the legal, ethical, professional and social issues (LEPSI) surrounding PLePA could be researched, investigated and determined if any work needed to be altered. This research can be seen in section 2.1, the result of this research was that there was no work to be changed for PLePA, but the awareness of the acts and guidelines was important.

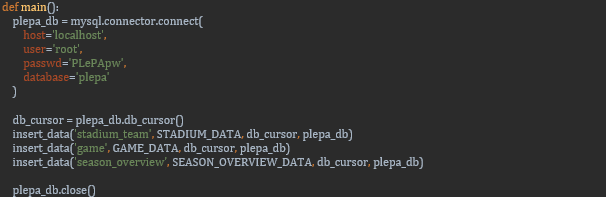
### 1.3.2 CF1 Setting up the database

The planning stage of the CF was to think about the database structure that could be used and what data would be required for the database.  
The first piece of work in CF1 was to create a conceptual model, the final structure decided upon for the main database can be seen below in Figure 2. This went through multiple iterations and there were a few points discussed with my tutor such as dropping pks on stadium\_name since it was not required with stadium\_id\_pk already there and using home\_team\_id\_pk\_fk and away\_team\_id\_pk\_fk instead of team\_one\_id\_pk\_fk and team\_two\_id\_pk\_fk.  
  
A screenshot of a cell phone

Description automatically generated  
Figure 2 (Blagg, 2020) Database conceptual model

Once the format had been chosen, the tables needed to be created in the database, the table\_creation script can be seen below in Figure 3.   
Figure 3 (Blagg, 2020) Core tables creation script

After that, the stadium data was manually looked up in Google, coordinates were identified, and this data was added into a csv file. Next, the data found in the preparation stage and the stadium data created could be inserted. This was done by using a Python script which iterated over the CSV files which has been downloaded and cleansed. The script iterates over each row, inserting the correct fields into the database from the data given. The main function for this script can be seen below in Figure 4.

  
Figure 4 (Blagg, 2020) Main function code for database insert

Once all the data was inserted, a database reset script was created. This was useful when mistakes were made, such as data becoming invalidated because it allowed for a safe restart point where the state of the database was known.  
There was not a lot of analysis done on CF1 other than a further review of the database and data used to ensure it would all work correctly.  
CF1 went successfully without any major issues, it was hard to establish any problems because the database was not yet in use by any of the algorithms.

### 1.3.3 CF2 Develop the random forest algorithm

CF2 began with researching more into the Random Forest algorithm and how it could be implemented in Python (Koehrsen, 2018) (Synced, 2017) (Polamuri, 2017) (Brownlee, Aug 2019) (Huneycutt, 2018) (VanderPlas n.d). A Random Forest algorithm is made up of multiple decision trees so further research was carried out on understanding decision trees more clearly as well (Brownlee, Dec 2019). Once the research had been complete, it was important to plan how the algorithm would work. It was decided that a new table would be created in the database to store moving thresholds for determining whether a result was a win, draw or loss for the home team. Each tree would have a score ranked between zero and three, the score for the tree would be obtained through three branches giving a score between zero and one hence the total score for the tree being up to zero. The thresholds would be trained using the data from 2014 to the end of the 2018/2019 season. This would allow the testing of the algorithm to be on the current season’s data.  
Once the development for CF2 had started, the need for a new table was identified, random\_forest\_results table, amongst other columns, this would be used to store which three functions were used, what the predicted result was and the actual result. This was needed to keep store the results from the algorithm.  
Figure 5 below shows part of the code for the random\_forest\_algorithm, “for i in range(100):” means 100 tree would be used for the forest, this was the number decided upon after testing with various numbers. “for i in range(3):” is the number of branches used for each tree, what this means is that it will go into that loop three teams and randomly select one of the five functions. The criteria for each of those five functions would determine how successful the algorithm would be.

A screenshot of a cell phone

Description automatically generatedFigure 5 (Blagg, 2020) Code snippet of Random Forest algorithm

Before the algorithm could start training, the thresholds had to be inserted into the database, the algorithm was run a few times to get a feel for what results would be produced. The algorithms then got set at 1.8 for a win and 1.2 for a draw, anything below 1.2 would be a loss. The movement of the thresholds can be seen below in Figure 6.

Figure 6 (Blagg, 2020) Thresholds of Random Forest algorithm

Each number on the x-axis represents a whole year’s worth of data used to train the thresholds, there were five years of training data and each year was used a thousand times. From Figure 6, it is clear the thresholds did not move much, the most noticeable changes were between 1000 and 2000 where the win threshold dropped, this was the 2015/16 season and between 3000 and 4000 where the draw threshold increased, this was the 2017/18 season. The correct prediction rates were also recorded during this training and can be seen below in Figure 7.

Figure 7 (Blagg, 2020) Correct prediction rate of training data for the Random Forest algorithm

After all the training, the algorithm could run over the current season’s data, this gave a correct prediction rate of 46.40%. This did not go as well as expected but without other algorithms to compare it to, it was difficult to know whether this was the work of the algorithm or that the data used, mainly, the distance travelled did not have a huge impact on the result of the game. It is important to note that there are so many factors influencing a game of football which is why it is so difficult to predict.

### 1.3.4 CF3 Develop the k-NN algorithm

Again, CF3 started in a similar manner to CF2 with research into the algorithm (Maklin, 2019) (Buitinck, n.d) (Brownlee, Oct 2019) (Navlani, 2018). Prior knowledge to the k-NN algorithm was also obtained through OU studies in TM351 (The Open University, 2014), this meant less preparation was required for working on this algorithm. The algorithm involved using the main data and test data all in the same program, unlike the Random Forest algorithm where training the data could be in a separate module to the main algorithm. The main three functions written for k-NN can be see below in Figure 8.

A screenshot of a cell phone

Description automatically generatedFigure 8 (Blagg, 2020) Code snippet of k-NN

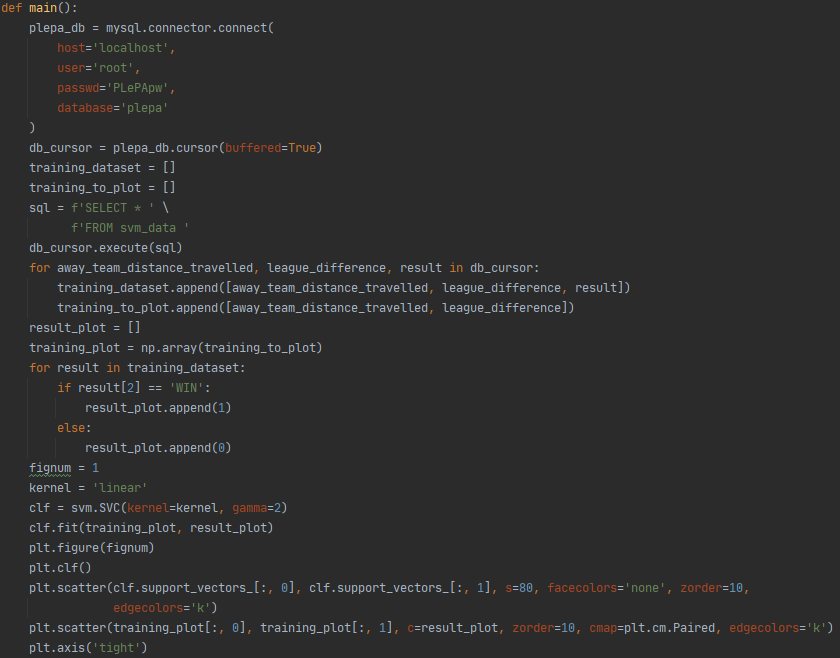
The first function, euclidean\_distance, takes in the coordinates of two points, neighbors, and returns the distance calculated.   
The second function, get\_k\_nearest\_neighbors, takes in the training\_data, the row being classified, new\_row, and the amount of neighbours used to determine the classification, neighbor\_num, this function will calculate the distances to all the training\_data and return the closest amount of neighbours which is determined by neighbor\_num.  
The third function, get\_classification, takes in the same parameters, finds the classification for each neightbour and determines the classification for the new point by using the maximum classifications. For example, if there are ten nearest neighbours and six of them are wins, the classification for the new point will be a win.  
The results for the k-NN algorithm can be seen in Figure 9 below, the graph shows that the peak for correct prediction rate was when k was at 3. The percentage was 49.64% which is still low but is slightly better than the Random Forest algorithm at 46.40%.

Figure 9 (Blagg, 2020) Correct prediction rate for the k-NN algorithm

### 1.3.5 CF4 Develop the SVM algorithm

Similarly, to CF2 and CF3, CF4 started with research into the algorithm, SVM (Stecanella, 2017) (Buitinck et al., n.d) (Navlani, 2019) (Maklin, 2019). The research provided a good insight into developing SVM using the Scitkit module, however, as you will see below, it did not go quite as planned in PLePA. SVM provided difficulties because a lot of the research done showed SVM classifying into two separate classes, we initially had three classifications: win, draw and loss. A lot of time was spent trying to get the algorithm to work with three different classes. Unfortunately, this was not able to be done in PLePA and it did take a chunk of time from the schedule. Therefore, for SVM, the classification was split between either a win or not a win, i.e. draw and loss were combined. Once the classification was set into two groups, there were further complications for this algorithm trying to plot different kernels using Python’s matplotlib module. In the end, only a linear kernel was used for SVM. For this algorithm, we only used two main data points for predicting the result, the distance travelled for the away team and the position difference between the two clubs when they played. This was worked out by the home team position minus the away team position, so it ranged from 19 to -19. The plot produced for this algorithm can be seen below in Figure 10 and the code for obtaining the training data and plot can be see in the code below in Figure 11. In Figure 10, you can see the split of where a win and loss were determined. The blue points in the brown section and vice versa represent outliers. This split will be used on the test data to check the validity of the algorithm produced.

  
Figure 10 (Blagg, 2020) Plot of training data for the SVM algorithm

  
Figure 11 (Blagg, 2020) Code snippet of obtaining training data and beginning to plot for the SVM algorithm

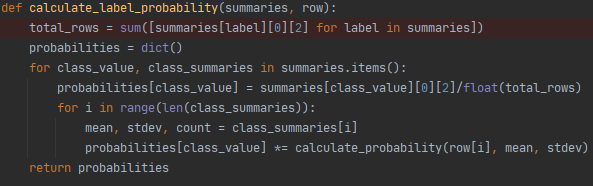
Once the algorithm had been trained, we could then look to test the successfulness of the algorithm by applying the test data on to the plot seen in Figure 10. The code for applying the test data onto the same place is comparable to the code for the training data seen in Figure 11. The plot produced by the test data can be seen below in Figure 12. This time the brown points in the blue section and vice versa show where the algorithm has incorrectly suggested a classifications. The brown section is the home team winning and the blue section is the home team drawing or losing.

  
Figure 12 (Blagg, 2020) Plot of test data for the SVM algorithm

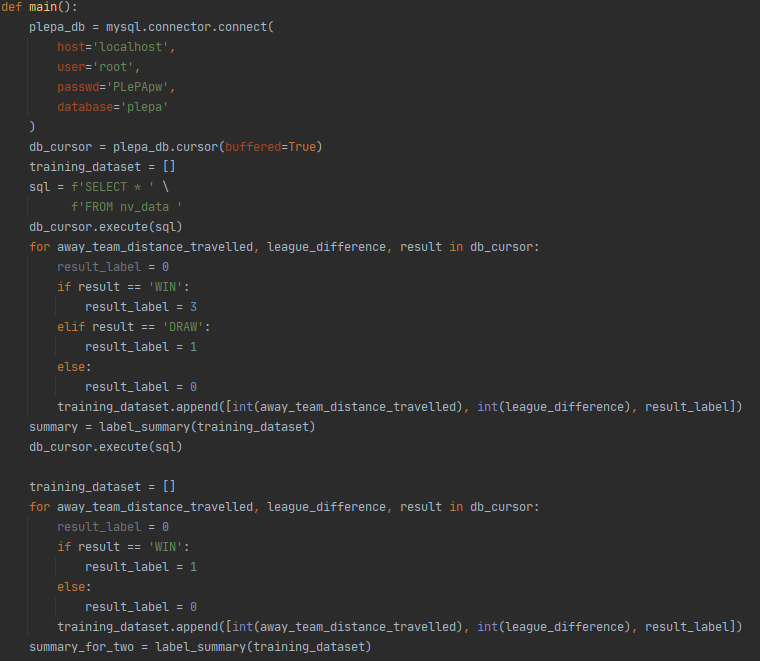
As seen from Figure 12, the algorithms predictions were mostly right. The actual correct classification figure was 72.3%, however, this was only classifying whether the home team won or did not win.

### 1.3.6 CF5 Develop the Naïve Bayes algorithm

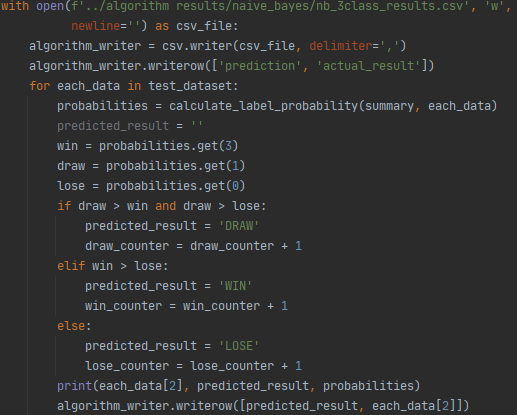
Again, in the same fashion as all other algorithm development, CF5 started with research into Naïve Bayes (Soni, 2018) (Brownlee, 2019) (Brownlee, 2019). The research for this algorithm was very useful for determining what route to take. With this algorithm, the development went into the variations seen in between the other algorithms, two and three separate classifications. This allows the algorithm to be comparable to all variations seen in other CFs. Unlike SVM, Naïve Bayes development went a lot more smoothly with a lot less complications. The function doing most of the work for this algorithm can be seen below in Figure 13. This function calls a lot of smaller functions which can be seen in Appendix 2.

  
Figure 13 (Blagg, 2020) Naïve Bayes function to calculate the probabilities

A snippet of the main function showing the algorithm being trained can be seen below in Figure 14. You can see the db\_cursor must be looped twice to sort the data between the two variations of classifications. The two-classification variation followed the same principle as SVM, using win or not a win for the home team. This algorithm was also able to utilise only two pieces of data, again, following SVM, distance travelled for the away team and league difference was used.

  
Figure 14 (Blagg, 2020) Snippet of main function in Naïve Bayes

The results were recorded into a csv file, the code to do this can be seen below in Figure 15. This made it easy to pull the statistics for the algorithm.

  
Figure 15 (Blagg, 2020) Insert into CSV file for Naïve Bayes

The results for this algorithm varied quite drastically between two and three classifications. The two-class classification had a 66.55% correct prediction rate whereas the three-class classification was only at 46.4% correct prediction rate. 46.4% is still better than just a guess which would sit around 33% because it would be assumed that you would guess one in every three games correctly. It is expected that a two-class classification would perform better since there is less variation for sorting, however, the author did not expect a 20% difference in the two tests.

### 1.3.7 Comparing algorithms

Initially, PLePA was looking to compare all the algorithms developed and it would be easy to see which algorithm performed best, however, due to the complications with two and three class classifications the results are not as comparable. From Table 2 below, it can be seen that the best classification prediction rate was 72.3% from the SVM algorithm, however, this was a 2-class classification. The best 3-class classification prediction was 49.64 from the k-NN algorithm. Random Forest and Naïve Bayes had the same 3-class classification rate, this is quite surprising because Random Forest used five different data points and Naïve Bayes only used two, but they still came out with the same prediction rate. There was not a massive difference in correct prediction rate between all 3-class classifications, only 3.24%, and all the prediction rates were relatively low, only around 15% above a blind guess which would equate to around 33%, this could mean the data used, distance travelled for away team, is not substantial when predicting the result of the match.

|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm | 2-class classification (%) | 3-class classification (%) | Best classification (%) |
| Random Forest | N/A | 46.4 | 46.4 |
| k-NN | N/A | 49.64 | 49.64 |
| SVM | 72.3 | N/A | 72.3 |
| Naïve Bayes | 66.55 | 46.4 | 66.5 |

Table 2 (Blagg, 2020) Table to show the results of the different algorithms used in PLePA split between two and three class classifications

### 1.3.8 What is completed and what is outstanding

All the must have and should have activities were completed, the nice to have criteria for these activities were also completed. The only activities left incomplete was an interface for user interaction and a feed from a football website. This can all be seen below in Table 3.

|  |  |  |  |
| --- | --- | --- | --- |
| Activity | Criteria | Evaluation | Progress to date |
| Premier league stats in the database. | Must have. | There will be a year’s worth of data in the database. | As can be seen in section 1.3.2, this activity has been completed with five years of data. |
| Should have. | Three years. |
| Nice to have. | Five years. |
| A working predictive algorithm to predict the football results. | Must have. | Two algorithms will have been developed and can be used. | As outlined in section 1.3.3 and 1.3.7, this activity is at the nice to have. All four algorithms were developed for PLePA |
| Should have. | Three algorithms. |
| Nice to have. | Four algorithms. |
| An interface for user interaction. | Nice to have. | A user can load up an executable and select two clubs, they will then receive a prediction. | No progress. |
| Nice to have. | Further development could mean the user could select the algorithm to predict with and the results using the teams and the algorithm are displayed. | No progress. |
| A feed from a football website. | Nice to have. | The upcoming fixtures are pulled from a website and predicted using the chosen algorithm. | No progress. |

Table 3 (Blagg, 2020) Activity evaluation

### 1.3.9 How to complete project

The first of the two activities left to complete, an interface for user interaction, could be completed in Python using a module called tkinter. The author of PLePA has experience in developing interfaces in Python using tkinter so this task would not be too difficult, however, it would be time consuming to develop nicely. The original idea was to have three drop down selections, home team, away team and algorithm to use. The user would select their three choices and generate a prediction rate which would be displayed on screen. However, a fourth drop down or radio button could also be available for two or three class classifications. It is expected this would take about 20 hours of work to complete.  
The second activity, feed from a football website, would need a lot more investigation because the author does not have experience in this area. However, it would most likely be an API call to a sports website to get the upcoming fixtures which then could be matched to a club in the teams table and a prediction could be made by each algorithm and shown on screen or sent by an email if the call to the API was made weekly, as an example. It is hard to say how long this task might take without further investigation but around 20 hours is expected.

# Section 2 – Review and reflection

## 2.1 LEPSI Review

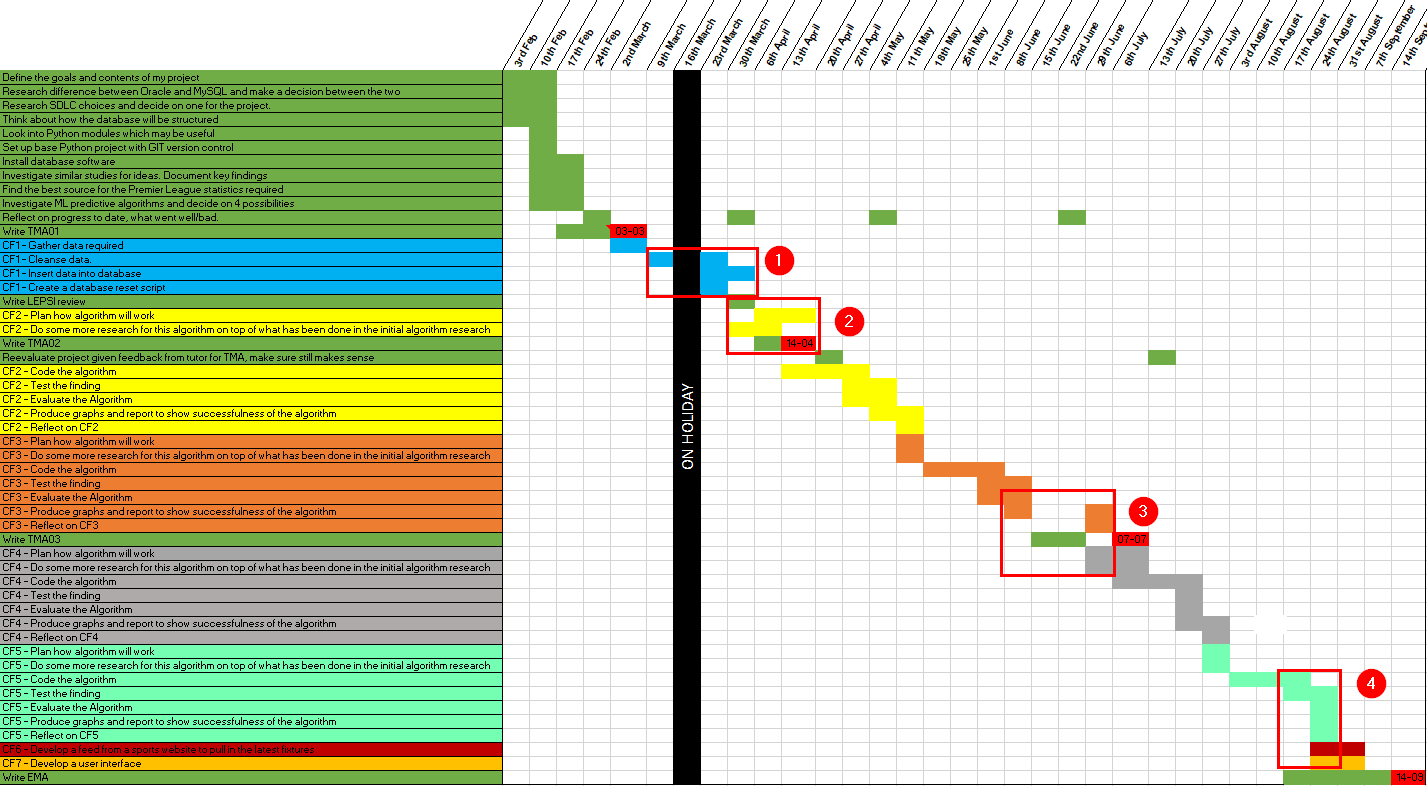
The full LEPSI review can be seen below in Table 4.

|  |  |  |  |
| --- | --- | --- | --- |
| Act / Law / Guidance | Purpose | Relevance to PLePA or n/a and why not applicable | How PLePA was affected |
| Data Protection Act 2018 (DPA) | Control how personal information is used by organisations, businesses or the government (UK Government, 2018) | PLePA did not use any personal information, so this was not applicable. | PLePA was unaffected. |
| Equality Act 2010 | Protection from discrimination in the workplace and society (UK Government, 2012) | PLePA will not be looking at people specifically, so this was not applicable. | PLePA was unaffected. |
| Computer Misuse Act 1990 | The act was brought into place to prevent people gaining unauthorised access to computer material, commonly known as hacking. (UK Government, 1990) | PLePA did not involve connection to the internet. It may do in the future, if it reaches development for a connection a sports API then this may need to be reviewed again | PLePA was unaffected so far, if the further development is reached to complete this project then the project may be affected. |
| Copyright | To protect your work by preventing it being copied, redistributed, adapted and put on the internet are some of the example (UK Government, n.d) | PLePA needed to make sure that the data used for past football results was not protected by copyright and could be used. | I will need to ensure I am able to use the data and not breaking copyright rules. |
| Freedom of Information Act | It gives the general public access to certain information on request from the public authorities. The public authorities are also obliged to publish certain information. (ICO, n.d) | This did not affect PLePA because I am not working for a public authority and the project did not require data from a public authority. | PLePA was unaffected. |
| Protection from harm | Protect participants of studies from any harm, physical or psychological, particular care should be paid to children. (TM470 course team, 2012) | This did not affect PLePA because it is not using participants. | PLePA was unaffected. |
| Professional codes of practice and ethics | This allows participants to understand the purpose of the study, the researchers must state their intentions. (TM470 course team, 2012) | This did not affect PLePA because it is not using participants. | PLePA was unaffected. |
| BCS Code of Conduct | To set out standards across the board for all members of the BCS. An example is having respect for public health, privacy, security and wellbeing of others and the environment. (BCS, June 2019) | This will not affect PLePA directly because I’m not a member of the BCS, however, I tried to comply with the standards set out by the BCS such as honesty with my skillset and acknowledgment to any borrowed source code used in my project. | No code was borrowed in the end so PLePA was mostly unaffected other than consciously thinking about the standard when developing PLePA. |

Table 4 (Blagg, 2020) LEPSI review in relation to PLePA

## 2.2 Planning

The first, see Appendix 1, and current schedule, see Figure 16 below, differ slightly. All changes are marked below with a red circle and a number in, they’re numbered 1, 2, 3 and 4. I will refer to each one as change one, change two, etc. Change one was to add a section for creating a database reset script. This was realised to be an important part of work when generating results for algorithms. It was useful to be able to clean the database down and just have the core football data inserted into the tables without any of the adjustments which got made by the Python algorithm scripts. Change two was to adjust when some of the work would start for CF2, this was to get TMA02 completed in time for the deadline. It was useful to be able to start coding the algorithm directly after completing the TMA rather than having to go back into a bit of research again. Change three affected a large part of the schedule because it involved changing the lengths of CF3, 4 and 5 were adjusted. This was because, initially, CF5 had a very small-time frame to be completed, it started with overlapping on CF4 and only lasted two full weeks after that. This was adjusted to have three full weeks after realising how long the previous algorithms took with research, development and gathering results. This also had a knock-on effect on CF6 and CF7, which can be seen in change four, meaning they were less likely to be completed but that is the benefit of adjusting CFs in a structured-case life cycle. I also had to move around the time to complete TMA03 in change three so that I could be sure it would be completed by the deadline.  
There were no real issues with the schedule, it remained fairly similar to the first draft with only minor changes highlighted above. This is probably due to no massive negative impacts on the project, I was mostly able to continue the project through the seven months without anything greatly reducing time I could be into the project.

  
Figure 16 (Blagg, 2020) Schedule of PLePA with markings of differences

## 2.3 SDLC and project management review

The SDLC choice, as seen throughout the project, was a structured-case lifecycle, this can be seen in Figure 1. The lifecycle choice was considered after discussion with my tutor, Judith Tope, a snippet of the communication can be seen in Appendix 3. I had not heard of this particular life cycle choice before, but it worked very well in this project. It allows the project to be compartmentalised, this is useful when the trying to scope stretch goals because you can put them into a separate CF and it does not affect the main project if that goal is not reached. If the project was to be done again, I would use the same lifecycle because I don’t think any shortcomings that were experienced in the project were due to the SDLC choice and I think a lot of benefits were gained from the choice. The work done to compare the different lifecycles for PLePA can be seen below:

***Classic* Waterfall** (The Open University, 2020)

A classic waterfall is a life cycle which follows an order and does not revisit previous parts of the life cycle. It follows analysis -> design -> implementation -> design

Benefits  
The main benefit of the waterfall method is that it is harder to deviate off track because it is a flow through one cycle. You do each task in a linear fashion, i.e. you do the analysis, then the design, then implement your design and evaluate at the end.

Disadvantages  
The big disadvantage is also because of the linear fashion of the life cycle, you do not revisit any from before. If there is an oversight which is discovered at the end, using this life cycle, you would not revisit and improve.

Potential usefulness in project  
This would be useful for keeping the project on track since it has a set deadline which is the length of the module, however, by not allowing iteration this would not work for my project.

Accept/reject  
Reject

***Iterative* Waterfall** (The Open University, 2020)

An iterative waterfall life cycle is like the classic waterfall but iterates over and over each part.

Benefits  
The biggest benefit of an iterative waterfall is that each of the four main parts are revisited over and over and can be revisited from any part, e.g. when you are at the evaluation stage of the project, you can return to the analysis, beginning, stage.

Disadvantages  
A disadvantage is that this can mean you might keep revisiting earlier stages of the life cycle and not finish in the timeline given.

Potential usefulness in project  
This would be useful for the project if the time is managed using a schedule to ensure that the targets are met.

Accept/reject  
Accept, but later declined based on feedback from tutor. It was suggested to investigate a Structured-case life cycle which was ideal for this project.

**Agile** (Victor Osetskyi, 2017)

An agile life cycle sets out work for a set period, usually two weeks, called a sprint. The progress made from each sprint is then factored in the following sprint.

Benefits  
With agile development, you may have long term goals, but you are mainly thinking in terms of every two weeks. You set yourself goals for each two-week period. This is very good for keeping you on track short term as you’re not allowed to deviate from the goals you’ve set yourself for those two weeks. If you surpass your goals, you may have work to do from the backlog.

Disadvantages  
A disadvantage for this project as it is very for collaborative work because you will all collaborate on how things have gone and review each sprint. It may be difficult for me to think in my mind after every two weeks what went well and what didn’t.

Potential usefulness in project  
This could be useful for the project, but I don’t think it would get all the benefits of an agile life cycle given that it’s a solo project. There is also a set deadline so working in sets of two weeks may put the project behind and wouldn’t be realised until too late.

Accept/reject  
Reject

**Structured-case** (J.M.Carroll et al., 2000)

A structured splits multiple parts of work into 4 phases: Plan, collected data, analyse and reflect. These are known as conceptual frameworks and each one is referred to as CF1, CF2, CF3 etc.

Benefits  
This organises each chunk of work into its own section which can be seen below in figure 1 and will be worked through chronologically which means you will not get distracted by starting to work on CF4 if you’re still on CF2 as an example.

Disadvantages  
The tasks which are later in the life cycle may never get touched because of the chronological ordering. If you have ten CFs but only manage to do eight, the final two will never get looked at.

Potential usefulness in project  
This will be very useful for the project because there will be three to four algorithms which will be worked through but each bit of work is separate so if only three are managed then that’s ok because the other work will not be affected by having not looked at the fourth algorithm.

Accept/reject  
Accept

## 2.4 Resources

The full list of resources required for PLePA can be seen below in Table 5, the only change to this list from the original, seen in Appendix 4 was the addition of “Information from previous studies”. It was a very necessary resource because it helped guide the development of PLePA into a way that I thought was most suitable.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Resource | Why needed | When needed | Problems if not available | How to ensure availability |
| Time | Complete any task for the project. | Throughout project. | The less time available, the fewer tasks can be completed. | Cannot ensure time but can try to get ahead on tasks in case of any complications. |
| PC | To complete all tasks, requires pc availability. | Throughout project. | Whilst no PC is available, most tasks will be unable to be achieved. | I have a desktop and laptop. As a last resort, I can also use my work laptop. |
| Python | Write the code for the algorithms. | During CF2-5. | The code for the algorithms will not be able to be written. | It is installed on all PCs available. |
| MySQL | Store the data used for algorithms. | Set up in CF1 but will be required from CF1 – CF7. | The algorithms will have no database to read from. | It is installed on all PCs available. |
| Premier League data | The project is using data from the Premier League for the algorithms. | Pre CF1, it will then be stored in the MySQL database. | No data to be used for the algorithms. | Once gathered and stored in the database, store a copy of the database. |
| Information from previous studies | To understand what went well and not so well for similar projects. | Studied prior to any real task being completed, used throughout. | No knowledge from other teams’ experiences so PLePA may make similar mistakes as other studies which were not investigated. | Look on the internet and in the library early in the project. |

Table 5 (Blagg, 2020) A table to show the resources required for PLePA

My review of the resources required is below:

* Time has been managed appropriately, as previously stated, and PLePA has kept to schedule where possible, not too many changes to the schedule and no drastic changes.
* There have been no PC issues, still have laptop available and all work is stored in GitHub as well as locally.
* Python has been installed everywhere, no Python issues.
* MySQL has been installed everywhere, all database files are backed up, the database runs on the main desktop.
* All data stored in CSV files and in the database. Backed up to GitHub.
* Research has been carried out throughout the project, there was lots of research done at the beginning of PLePA and each CF stage starts with further research.

## 2.5 Risks

The full list of risks can be seen below in Table 6, there are a couple of differences from this list and the initial list, in Appendix 5. A high risk added was the inability to obtain the Premier League data required, PLePA was obviously pivotal on the data being available so this was added. The second risk added was time and being able to manage it correctly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Risk involved | Priority – low to high | Overcome/manage risk | What to do if not managed | Review of risk |
| Planning on a house move. | Medium | Get ahead on project when possible, manage time during move. | Make up the time after the house move is complete. | Accept – try to manage time accordingly around busy times for house move. |
| Work travel, 2-4 days a month. | Low | Manage workload accordingly around work travel to get ahead. | Try to find a bit of extra time following the travel to catch up on project. | Accept – try to get ahead before work travel and catch up if needed afterwards. |
| Desktop and laptop failure | Low | Data stored at GitHub and locally on both machines. | Library work or purchase new laptop/desktop to replace. | Avoid – Unlikely that both desktop and laptop will fail. |
| Knowledge gap on new Python modules | High | Research the modules required, also have a Python expert at my work who I can consult with. | Lower the detail of the algorithms used. | Accept – start researching required modules early and learn how to use. |
| Unable to gain knowledge on report writing | High | Lots of preparation and research ahead of the EMA, the TMAs are good practice too. | A poor representation of work done, won’t be reflective of the work put in in the project. | Avoid – make sure to research and ask for advice from contacts who are good at report writing. |
| Unable to obtain Premier League data | High | Ensure the data is available and download onto PC before lost. | No data available for the project, very difficult to continue | Avoid – it would be disastrous to not obtain the data required. |
| Time | Medium | Manage time correctly, try to productive when working and keep up with the schedule. | Project may fall behind, and milestones may not be met. The project could be unfinished. | Accept – manage time appropriately, try not to fall too far behind on schedule. If behind on schedule, try to catch up. |

Table 6 (Blagg, 2020) A table to show the risks to be manage for PLePA

The review of the risks to PLePA is below.

* Planning on a house move would have taken up some time, however, due to COVID-19, the house move was cancelled so this is no longer a risk.
* Work travel has also been cancelled due to COVID-19, so this is no longer a risk.
* Desktop and laptop failure, as stated above, there have been no computer related issues.
* Knowledge gap for new Python modules is being handled by plenty of research being carried at the start of each CF so there is a good understanding of the modules required.
* Unable to gain knowledge on report writing has been managed by speaking to my partner who writes plenty of reports in her work. I have also had to write more lengthy documents at work which has improved my report writing.
* Premier league data has been downloaded and used, it is currently stored in the database, no longer a risk.
* Time, kept to schedule mostly with no major differences.

## 2.6 Personal review

Overall, the project went pretty well. I think I dealt with the complications of not being able the SVM algorithm working with three classifications in CF4 well and developed a good compromise. I then made it comparable to at least one other algorithm by developing the final algorithm, Naïve Bayes, with both two and three class classification results. That was the main issue experienced within the project. Other than that issue, the development of the algorithms and creation of databases went quite smoothly using my knowledge of Python learnt through work and previous modules with the OU, specifically TM351 and my knowledge of databases. The support of my tutor throughout the project was very useful, and the communication was mostly each fortnight, a snippet of the communication can be seen in Appendix 3.  
I tried my best to follow the BCS Code of conduct (BCS, June 2019) throughout the project, this was sometimes difficult to keep on top and I had to go back through work to amend. I tried my best to consciously think about it when doing any of my project work.  
The skills required for PLePA can be seen below in Table 7, the initial skills list can be seen in Appendix 6.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Skill | Why needed | When needed | Problems if not available | How to ensure availability |
| Python coding | Write the code for the algorithms. | During CF2-5. | The code for the algorithms will not be able to be written. | Research unused additional Python modules which are required. |
| SQL coding | Write the code for the database. | During CF1. | Unable to create the database structure and insert the data. | Used in everyday work, fluent in SQL. |
| Time management | Ensure milestones are met. | Throughout project. | Milestones may be missed; project will be incomplete. | Research good time management ideas and seek advice from people with good time management. |
| Report writing | To write TMA1-3 and EMA. | For all assignments but mainly the EMA. | The quality of writing for TMAs and EMAs will be lacking. | Research and check previous module’s advice on report writing and read course booklet on the subject |
| Researching | For several tasks to investigate best approach | Throughout project. | Project will suffer due to decisions being made without right information. | Look into ideas for researching, the best approaches to research and read course booklet on the subject |

Table 7 (Blagg, 2020) A table to show the skills require for PLePA

I will review each skill for PLePA below:

* There have not been issues with Python coding and lack of the skill has been resolved by doing research which has not been laborious due to the enjoyment of writing Python code.
* The SQL for this project was quite basic compared to what I am used to so this skill has not caused any problems.
* Report writing could be better, I have worked on this by conversing and asking for help from my partner.
* Researching has been an interesting skill to learn, this could still be better but the more that it is done, the more confident I feel about researching the next part required for the project.

## 2.7 Effective working review

The way of working for PLePA had a good start, I communicated with my tutor early about all my plans for the project and kept in contact almost weekly to begin with to ensure the project was going in the right direction. I attended all the tutorials, which there were a few of at the start because, presumably, it is helpful to ensure you’re on the right track to begin with. This was very helpful in the early development of PLePA. I think having a good schedule and SDLC was important in the successfulness of this project, being able to follow the schedule was very useful when planning work each week.   
I expected with the outbreak of COVID-19 which meant I was working at home, I would be able to get on top of the project and maybe push for the stretch goals, however, I was wrong. This was because I found it harder to focus on the project on the weekdays after a full day work due to it all work being done in the same place. I had no separation from full-time job and project work. I would say it was more of a hindrance than a help to my project when working from home.

(9875 words)

# Bibliography

Avinash Navlani (August 2018) *KNN Classification using Scikit-learn* [Online] Available at <https://www.datacamp.com/community/tutorials/k-nearest-neighbor-classification-scikit-learn> (Accessed 17th May)

Avinash Navlani (December 2019) *Support Vector Machines with Scikit-learn* [Online] Available at <https://www.datacamp.com/community/tutorials/svm-classification-scikit-learn-python> (Accessed 9th July)

BCS, The Charted Institute for IT (June 2019) *Code of Conduct for BCS Members* [Online] Available at <https://cdn.bcs.org/bcs-org-media/2211/bcs-code-of-conduct.pdf> (Accessed 1st April)

Bruno Stecanella (June 2017) *In introduction to Support Vector Machines (SVM)* [Online] Available at <https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/#:~:text=A%20support%20vector%20machine%20(SVM,on%20a%20text%20classification%20problem.> (Accessed 6th July)

Cory Maklin (August 2019) *Support Vector Machine Python Example* [Online] Available at <https://towardsdatascience.com/support-vector-machine-python-example-d67d9b63f1c8> (Accessed 9th July)

Cory Maklin (July 2019) *K Nearest Neighbor Algorithm In Python* [Online] Available at <https://towardsdatascience.com/k-nearest-neighbor-python-2fccc47d2a55> (Accessed 17th May)

Dataflair Team (September 2018) *8 Machine Learning Algorithms in Python* [Online]. Available at <https://data-flair.training/blogs/machine-learning-algorithms-in-python/> (Accessed 16th February 2020)

Devin Soni (May 2018) *Introduction to Naïve Bayes Classification* [Online] Available at <https://towardsdatascience.com/introduction-to-naive-bayes-classification-4cffabb1ae54> (Accessed 2nd August)

FBREF (February 2020) *2019-2020 Premier League Stats* [Online]. Available at <https://fbref.com/en/comps/9/Premier-League-Stats> (Accessed 24th February)

Football-Data (February 2020) *Data-files: England* [Online]. Available at <http://www.football-data.co.uk/englandm.php> (Accessed 24th February)

Information Commissioner’s Office (n.d) *What is the Freedom of Information Act?* [Online]. Available at <https://ico.org.uk/for-organisations/guide-to-freedom-of-information/what-is-the-foi-act/> (Accessed 1st April)

Jake Huneycutt (May 2018) *Implementing a Random Forest Classification Model in Python* [Online] Available at <https://medium.com/@hjhuney/implementing-a-random-forest-classification-model-in-python-583891c99652> (Accessed 19th April)

Jake VanderPlas (n.d) *In-Depth: Decision Trees and Random Forests* [Online] Available at <https://jakevdp.github.io/PythonDataScienceHandbook/05.08-random-forests.html> (Accessed 19th April)

Jason Brownlee (December 2019) *A Gentle Introduction to Bayes Theorem for Machine Learning* [Online] Available at <https://machinelearningmastery.com/bayes-theorem-for-machine-learning/> (Accessed 2nd August)

Jason Brownlee (October 2019) *Naive Bayes Classifier From Scratch in Python* [Online] Available at <https://machinelearningmastery.com/naive-bayes-classifier-scratch-python/> (Accessed 2nd August)

Jason Brownlee (October 2019) *Develop k-Nearest Neighbors in Python from Scratch* [Online] Available at <https://machinelearningmastery.com/tutorial-to-implement-k-nearest-neighbors-in-python-from-scratch/> (Accessed 17th May)

Jason Brownlee (August 2019) *How to Implement Random Forest From Scratch in Python* [Online] Available at <https://machinelearningmastery.com/implement-random-forest-scratch-python/> (Accessed 14th April)

Jason Brownlee (December 2019) *How to Implement The Decision Tree From Scratch in Python* [Online] Available at <https://machinelearningmastery.com/implement-decision-tree-scratch-python/> (Accessed 14th April)

J.M.Carroll, P.A.Swatman (2000) *Structured Case: A Methodological Framework for Building Theory in Information Systems Research*. Available at <https://pdfs.semanticscholar.org/caff/950b0efc8f23e17e7de171d8836c8dfa1044.pdf>(Accessed 17th February 2020)

Lars Buitinck et al. (n.d) *1.4 Support Vector Machines* [Online] Available at <https://scikit-learn.org/stable/modules/svm.html> (Accessed 9th July)

Lars Buitinck et al. (n.d) *1.6 Nearest Neighbors* [Online] Available at <https://scikit-learn.org/stable/modules/neighbors.html> (Accessed 17th May)

Nick Campanelli (May 2019) *Betting on the English Premier League* [Online]. Available at <https://towardsdatascience.com/betting-on-the-english-premier-league-making-money-with-machine-learning-fb6938760c64> (Accessed 24th February)

Saimadhu Polamuri (May 2017) *How the Random Forest Algorithm Works in Machine Learning* [Online] Available at <https://dataaspirant.com/2017/05/22/random-forest-algorithm-machine-learing/> (Accessed 5th April)

Synced (October 2017) *How Random Forest Algorithm Works in Machine Learning* [Online] Available at <https://medium.com/@Synced/how-random-forest-algorithm-works-in-machine-learning-3c0fe15b6674> (Accessed 5th April)

The Open University (2020) *Choosing a Lifecyle Model*, *TM470* [Online]. Available at <https://learn2.open.ac.uk/mod/oucontent/view.php?id=1557365>(Accessed 2nd February 2020)

The Open University (2019) ‘Data mining I: classification tasks’ *The k-nearest neighbours algorithm, TMA351* [Online]. Available at <https://learn2.open.ac.uk/mod/oucontent/view.php?id=1349974&section=3> (Accessed 24th February 2020)

The Open University (2019) ‘Data mining II: clustering tasks’ *Partitional clustering: the k-means algorithm, TM351* [Online]. Available at <https://learn2.open.ac.uk/mod/oucontent/view.php?id=1349975&section=2.2> (Accessed 24th February 2020)

TM470 course team (2012) *‘Legal, Social, Ethical and Professional issues’ in TM470 study material, The Open University, Milton Keynes* [Online]. Available at <https://learn2.open.ac.uk/mod/oucontent/view.php?id=1184585> (Accessed 1st April)

Tuan Doan Nguyen (August 2018) *The Beautiful Game: Predicting the Premier League with a random model* [Online]. Available at <https://towardsdatascience.com/o-jogo-bonito-predicting-the-premier-league-with-a-random-model-1b02fa3a7e5a> (Accessed 24th February)

UK Government (2018) *The Data Protection Act* [Online]. Available at <https://www.gov.uk/data-protection> (Accessed 1st April)

UK Government (2015) *Equality Act 2010: guidance* [Online]. Available at <https://www.gov.uk/guidance/equality-act-2010-guidance> (Accessed 1st April)

UK Government (1990) *Computer Misuse Act 1990* [Online]. Available at <http://www.legislation.gov.uk/ukpga/1990/18> (Accessed 1st April)

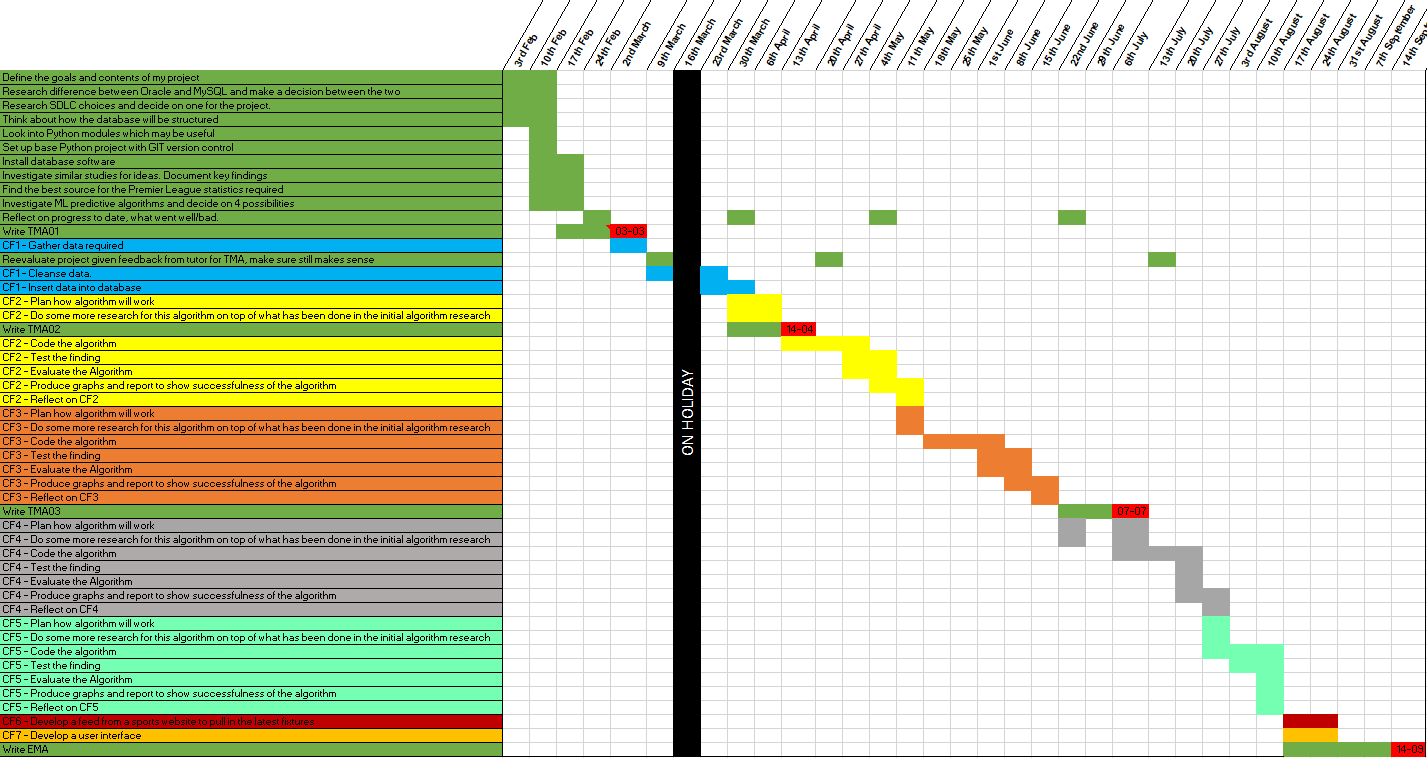
UK Government (n.d) *How copyright protects your works* [Online]. Available at <https://www.gov.uk/copyright> (Accessed 1st April)

Victor Osetskyi (2017) *SDLC Models Explained: Agile, Waterfall, V-Shaped, Iterative, Spiral* [Online]. Available at <https://medium.com/existek/sdlc-models-explained-agile-waterfall-v-shaped-iterative-spiral-e3f012f390c5>(Accessed 14th February 2020)

Will Koehrsen (August 2018) *An Implementation and Explanation of the Random Forest in Python* [Online] Available at <https://towardsdatascience.com/an-implementation-and-explanation-of-the-random-forest-in-python-77bf308a9b76> (Accessed 5th April)

# Appendices

## Appendix 1 – First draft of schedule



## Appendix 2 – Minor functions in Naïve Bayes

# ===================================================== Functions ==================================================== #  
  
def sort\_data\_by\_label(data):  
 sorted\_data = dict()  
 for i in range(len(data)):  
 vector = data[i]  
 label = vector[-1]  
 if label not in sorted\_data:  
 sorted\_data[label] = list()  
 sorted\_data[label].append(vector)  
 return sorted\_data  
  
  
# ==================================================================================================================== #  
  
  
def calculate\_mean(list\_of\_vals):  
 mean = sum(list\_of\_vals) / float(len(list\_of\_vals))  
 return mean  
  
  
# ==================================================================================================================== #  
  
  
def calculate\_stdev(list\_of\_vals):  
 average = calculate\_mean(list\_of\_vals)  
 variance = sum([(x - average) \*\* 2 for x in list\_of\_vals]) / float(len(list\_of\_vals) - 1)  
 return sqrt(variance)  
  
  
# ==================================================================================================================== #  
  
  
def calculate\_data(data):  
 summarized\_data = [(calculate\_mean(column), calculate\_stdev(column), len(column)) for column in zip(\*data)]  
 del (summarized\_data[-1])  
 return summarized\_data  
  
  
# ==================================================================================================================== #  
  
  
def label\_summary(dataset):  
 sorted\_data = sort\_data\_by\_label(dataset)  
 summary = dict()  
 for class\_value, rows in sorted\_data.items():  
 summary[class\_value] = calculate\_data(rows)  
 return summary  
  
  
# ==================================================================================================================== #  
  
  
def calculate\_probability(x, mean, stdev):  
 exponent = exp(-((x - mean) \*\* 2 / (2 \* stdev \*\* 2)))  
 return (1 / (sqrt(2 \* pi) \* stdev)) \* exponent  
  
# ==================================================================================================================== #

## Appendix 3 – Snippet of tutor communication

Hi Ju,

Thanks for the slides and useful tutorial yesterday. Thought i'd give you an update. Let me know if I'm using too much of your time. I just want to make sure, whilst in the early stages, that I'm on the right track.

Progress made - keeping my project journal up to date. Added my conceptual model, googling around it seems many people have a different opinion of what the conceptual model consists of, I've tried to base mine on your feedback mainly. Added a bit more detail to my lifecycle choices (will put this in tabular format for the TMA). Updated my schedule multiple times but still not sure if it's exactly how it should be. Wrote up my project description adn task list. Started researching algorithms, this is by no means complete.

Decisions taken - I decided on an iterative waterfall life cycle, most other bits are research at the moment, so no final decisions made.

What has gone well - In general, I feel like I'm getting on well with the tasks, I think I'm happy with my choices made so far.

What has failed to go well - Just the schedule, I'm not sure if I'm going in the right direction.

What you aim to do next - Research a bit more into the algorithms. Improve my schedule further, if necessary, based on your feedback. Research similar projects and where I can find the data required. Lots of research this week

Questions - I would like some feedback on my schedule, task list and conceptual model, if possible, please.

Kind regards,

Rob

database

you seem to have gone overboard with PKs

anyway here goes ---

stadium

only needs stadium id, the other attributes are not part of PK

team

why do you need a team id anyway, surely name is unique and stadium id not necessary in conceptual model (it is FK)

match

fine

season-overview

fine

schedule

you are still using TMAs as banner titles - do me a favour and remove those, leave colour coding as is

are you not aiming to do any reflection apart from green group? you are reviewing project in dark yellow lot I admit but ...

holidays?  or have you nothing planned?

task list

not totally consistent with schedule

I still have a minor thought that you should have a combined lifecycle of structured case and iterative where

cf1 = plan, develop and test database

cf2 = plan, research, code, test and evaluate algorithm 1

cf3 = alg 2

cf4 = alg 3

cf5 = alg 4

then you can write final summary but if time rears its ugly head you can just drop cf5 but you are tackling the project in reasonable chunks AND it will have a more coherent schedule associated with it

Good luck

Cheers

Ju

## Appendix 4 – Initial resource list

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Resource | Why needed | When needed | Problems if not available | How to ensure availability |
| Time | Complete any task for the project. | Throughout project. | The less time available, the less tasks can be completed. | Cannot ensure time but can try to get ahead on tasks in case of any complications. |
| PC | To complete all tasks, requires pc availability. | Throughout project. | Whilst no PC is available, most tasks will be unable to be achieved. | I have a desktop and laptop. As a last resort, I can also use my work laptop. |
| Python | Write the code for the algorithms. | During CF2-5. | The code for the algorithms will not be able to be written. | It is installed on all PCs available. |
| MySQL | Store the data used for algorithms. | Set up in CF1 but will be required from CF1 – CF7. | The algorithms will have no database to read from. | It is installed on all PCs available. |
| Premier League data | The project is using data from the Premier League for the algorithms. | Pre CF1, it will then be stored in the MySQL database. | No data to be used for the algorithms. | Once gathered and stored in the database, store a copy of the database. |

## Appendix 5 – Initial risk list

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Risk involved | Priority – low to high | Overcome/manage risk | What to do if not managed | Review of risk |
| Planning on a house move. | Medium | Get ahead on project when possible, manage time during move. | Make up the time after the house move is complete. | Accept – try to manage time accordingly around busy times for house move. |
| Work travel, 2-4 days a month. | Low | Manage workload accordingly around work travel to get ahead. | Try to find a bit of extra time following the travel to catch up on project. | Accept – try to get ahead before work travel and catch up if needed afterwards. |
| Desktop and laptop failure | Low | Data stored at GitHub and locally on both machines. | Library work or purchase new laptop/desktop to replace. | Avoid – Unlikely that both desktop and laptop will fail. |
| Knowledge gap on new Python modules | High | Research the modules required, also have a Python expert at my work who I can consult with. | Lower the detail of the algorithms used. | Accept – start researching required modules early and learn how to use. |