Premier League predictive learning algorithm (PLePA)

|  |  |
| --- | --- |
| **Contents** | |
| 1 | Project title |
| 1 | Glossary |
| 1 | Section 1 Preparation and planning |
| TBD | Project work to date |
| TBD | Review and reflection |
| TBD | Bibliography |
| TBD | Appendices |

TODO - TABLE OF APPENDICES

TODO - TABLE OF APPENDICES

TODO - TABLE OF APPENDICES

**Glossary**

**Section 1 Preparation and planning**

**1.1 Project description**

The project is to create a predictive algorithm for Premier League results. It will use historic data; it will use the last three years’ worth of data. This may be extended or reduced but that will depend on the ease of gathering and cleansing the data.

The result is aimed at Premier League fans and people interested in the prediction of football results. It’s also something I have an interest in solving, I do a bit of sports betting so this could prove yourself for this.

The aim is to produce four separate algorithms and determine which has the most accurate results. The goal is to have around a 75-80% pass rate, this will be a benefit to all football fans, fans who place bets and even possibly teams, that may be a bit far though.

If the project is unable to give good 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 I am 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.

**1.2 Activities, tasks and subtasks**

* Define the goals and contents of my project.
* Research SDLC choices and decide on one for the project.
* Research difference between Oracle and MySQL and decide between the two.
* Think about how the database will be structured
* Look into Python modules which may be useful.
  + Distance calculation
* Set up base Python project with GIT version control
* Install database software
* Investigate similar studies for ideas. Document key findings.
* Find the best source for the Premier League statistics required.
* Investigate Machine Learning, ML, predictive algorithms and decide on 4 possibilities.
* Write TMA01
* Write TMA02
* Write TMA03
* Write EMA
* Reflect on progress to date, what went well/bad (Needs to be done multiple times in the project.
* Revaluate project after each TMA given feedback from tutor for TMA. Make sure project still makes sense.
* Conceptual Framework (CF) 1 – setting up the database and data
  + Gather data required.
  + Cleanse data.
  + Insert data into database.
* CF2-5 – work on the four algorithms, each one is a separate CF.
  + Plan how algorithm will work.
  + Do some more research on top of what has already been done.
  + Code the algorithm
  + Test the findings
  + Evaluate
  + Produce graphs and report to show successfulness of the algorithms
* CF6 – develop a feed from a sports website to pull in the latest fixtures
  + This will feed the predictions for the upcoming fixtures and display on screen or email to a user.
* CF7 – develop a user interface
  + This will allow users to select two clubs and will display the predicted results
  + This could be developed further to allow the user to select which algorithm to predict with.

**1.3 Life cycle choice**

The life cycle chosen to use is a structured-case life cycle; the reasoning behind the decision is below. Initially the iterative waterfall was chosen but after some discussion with my tutor I chose to move towards structured-case life cycle. I was not aware of the structured-case life cycle but after looking into it, I found it seemed perfect for my project.

The four main parts of a life cycle are analysis, design, implementation and evaluation.

***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

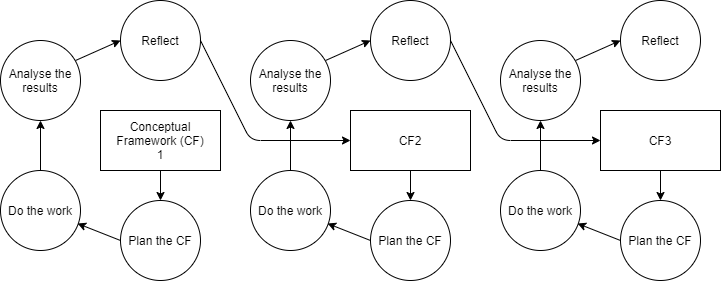
**Structured-case** (J.M.Carroll et al., 2000)

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 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.

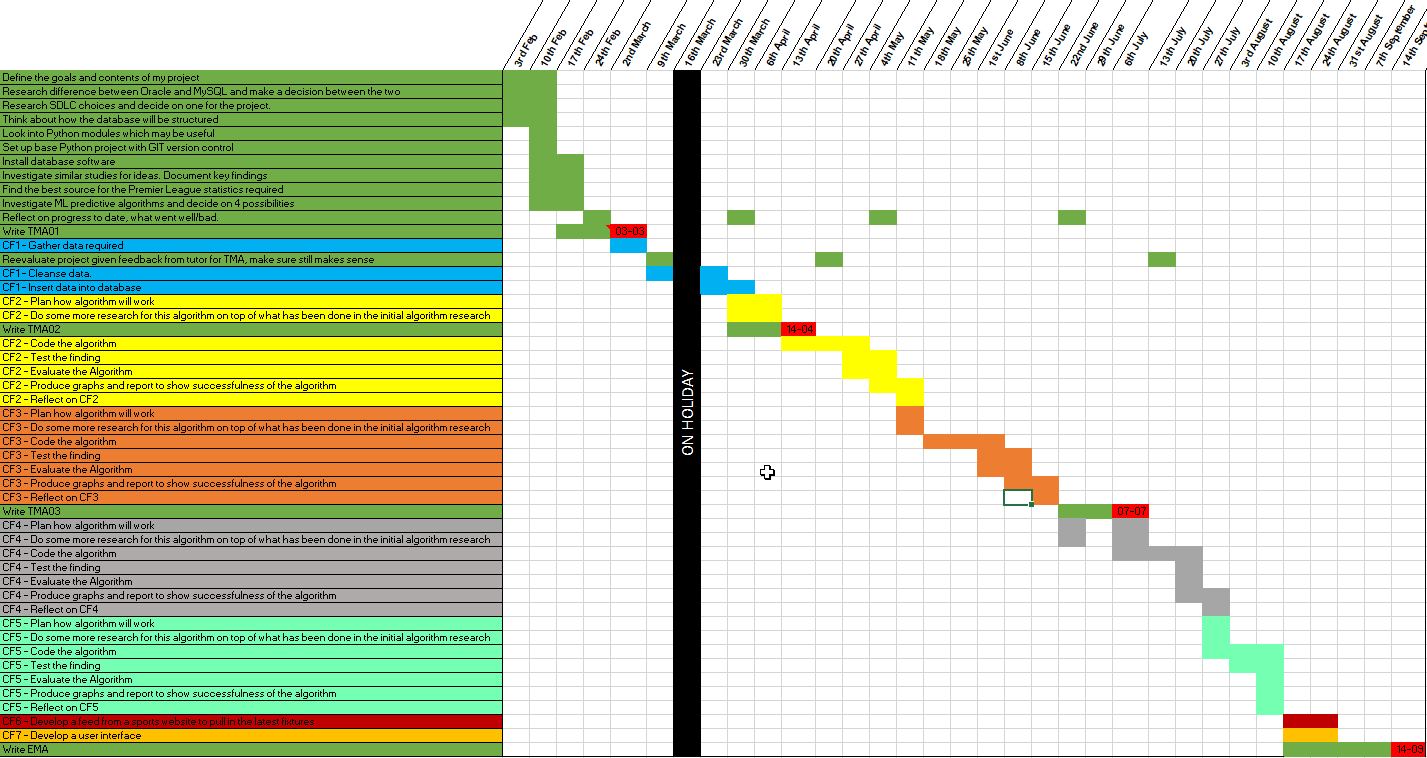
Disadvantages  
The tasks which are later in the life cycle may never get touched because of the chronological ordering. If you have ten CFs but only manage to do eight, the final two will never get looked at.

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  
 

Image(TODO)

**1.4 Schedule**

****

**1.5 How will it be evaluated**

|  |  |  |  |
| --- | --- | --- | --- |
| Activity | Criteria | Evaluation | Progress |
| Premier league stats in the database. | Must have. | There will be a year’s worth of data in the database. | Decided on database design. |
| Should have. | Three years. |
| Nice to have. | Five years. |
| A working predictive algorithm to predict the results. | Must have. | Two algorithms will have been developed and can be used. | Decided on the four algorithms to be used. |
| 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. |

**1.6 What will be needed to achieve end goals**

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. |

Skill list

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Skill | Why needed | When needed | Problems if not available | How to ensure availability |
| Python | 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 | 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. |
| 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 and the best approaches to research. |

Risk management 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. |

**Section 2 Project work to date**

**2.1 Literature review**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Literature | Presentation | Relevance | Objectivity | Method | Provenance | Timeliness |
| TMA351 part 1-26 | Very easy to follow, the material was split into 26 good chunks. | It is mostly relevant because it uses Python to design two predictive algorithms, however, it uses MongoDB rather than a MySQL database. | There are two algorithms used for the reader to compare. | It is clear where the information has come from. | The author is reliable because it is material directly from an OU module. | The material has been updated in the last three years. |
| Machine Learning Algorithms in Python (Dataflair Team, September 2018 | Very nice flow to the information presented. Easy to understand. | Very relevant as it is about ML algorithms in Python, however, not all are predictive. | The purpose of the literature is to give information on several different algorithms so there are no conclusions made by the author. | The information uses direct code snippets of Python to further explain the information. | The author is reliable, I have used them for multiple work projects in the past when learning new material. | Material is just under a year and half old. |
| Structured-case: A methodological framework for building theory in information systems (J.M.Carroll et al., 2000) | It had a clear layout, but the wall of text made it difficult to read. | It was useful for researching the structured-case life cycle. | Clearly gives the strengths and weaknesses of the life cycle. | Clear where the research has taken place. | The two authors are both university professors. | Material is twenty years old but the framework is still used today. |
| Choosing a Lifecycle Model (The Open University, 2020) | Nicely presented, easy to read. | Very useful for looking at life cycles however the one I ended up using wasn’t given. | Multiple life cycles with pros and cons given. | Clear where information has come from. | OU is a reliable source. | Material presented this year by the OU so assumed it is up to date. |
| SDLC Models Explained: Agile, Waterfall, V-Shaped, Iterative, Spiral (Osetskyi 2017) | Very good layout, broken up by pictures, text and tables. | Useful for doing additional researches on SDLCs. | All life cycles are presented with advantages and disadvantages. | The information has come from Existek, a full-cycle software development company. | The author works for Existek and his written other articles I have read which have been informative. | Written about two and half years ago. |

**2.2 Work done**