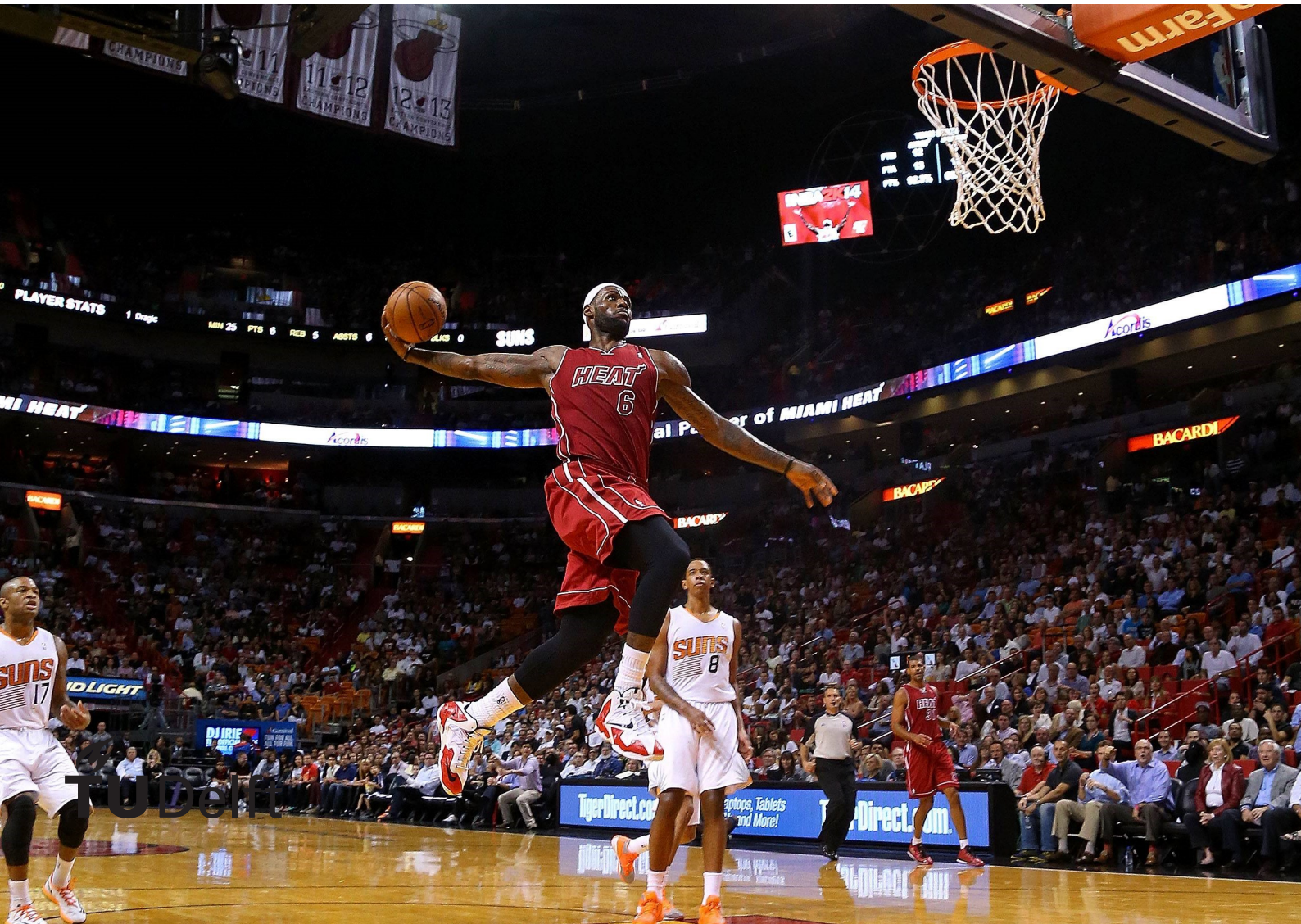
Project Development Plan

NBA game result predictor

Group DS4



by

Maciek Jankowski (4651294),
Ray Sambath (4665295),
Luke Chin A Foeng (4609972),
Paul Simon Schön (),
Matteo Rebosolan (4664973)

Contents

1

Project Objectives, Context, Requirements and Constraints

The goal of our project is to create a tool that will predict the results of upcoming NBA games based on results from past seasons. It will do so through use of a neural network that will independently assess the skill of all the players in the league according to their performances.

In order to create a fully functioning AI capable of predicting NBA matches, some objectives have to be set. These objectives include being able to reach 52% match prediction accuracy. This would prove that our AI is able to successfully predict the outcome of a match, with better odds than a simple guess.

This program should be able to take in a list of players from two teams playing in an NBA match, then predict which set of players will win the match. Requirements for our code include it being able to scrape the web for match data for every match for the last ten seasons. The scraping will be conducted using several match data sites, including NBA.com as well as ESPN.com, with ESPN.com being the main site as it contains match data from the past 20+ seasons. This will be more than enough data for our program, as we have decided that the average career length of an NBA player is around 10 years. This match data will include the score differential between the two teams playing, the players that played during that game as well as which team is home or away. From this data, the AI will train itself to see which players lead to an increase in a team's likelihood of winning and which players will lead to a decrease in likelihood of winning.

Some constraints for this code will include us avoiding the use of individual player statistics, and instead getting the code to weigh the players according to the games the players have already played. Another constraint will involve us not inputting the team name to prevent the code from associating the wins to the team names. We would rather want the code to associate the wins with the players within the teams rather than the teams themselves. We will also be avoiding the use of pre-season and playoff match data.

2

Project approach, resources, priorities, results optimizations

2.1. Approach

In order to have a working software, we must first break down our project's approach into smaller steps and gradually follow an incremental process until the software gets to work with low fidelity.
The approach is broken down into the follow steps:

- **Data gathering**: In this step, we scrape data from an appropriate website for collecting data.
  For scraping data, we have decided to use "Beautiful Soup" which is part of the python library. The website we have decided to scrape from is ESPN.com(R E F E R E N C E H E R E), due to the large amounts of data it contains.
  The data scraped contains details of the Home and Away games of all teams as well as the players in their respective teams. This is done due to the fact that Home teams have a higher winning percentage as compared to the team's performance when Away.
- **Data preparation**: In this step, the data collected in converted into a machine readable format. For this, we have decided to use CSV files as they are easy to read from and write to. We have also split the training data set from the ones for testing. The training data set contains information regarding 1 Home and 1 Away game of each team against each and every team in the league. The testing data set contains information of other Home and Away game played by each team.
- **Network choice**: In this step, we decide on a Neural Network for the project. We have decided to use the Multi layer Perceptron with Back Propagation. We have decided to use this as our preferred network as it enable the software to learn from it's mistakes. This is useful to optimize the program.
- **Training set**: This is essentially all the data used for training the software. As mentioned before, our training data set contains information regarding the 1st half of each season.
- **Result evaluation**: In this step, we evaluate our results and check if it satisfies our desired accuracy.
- **Tuning**: In this step, we adjust the weights and biases of the nodes to get fine tuned results.

2.2. Resources

The resources used for our project can be split into 2 categories:

- **Data gathered**: The data gathered contains information regarding the Home and Away games, as well as a list of players playing in each and every match and their respective ages.
- **Software used**: For most of our programming, Python will be used along with some additional features/libraries such as beautifulsoup, sci

3

Design objectives, design strategy, critical features, risk factors, 'to-avoids'

To accomplish the objectives stated in the first section of this development plan, the project has to be split into two separate parts. First the team will design the data mining script. The objective of this sub-design is to automatically gather data about past NBA season from specified sites and save it into a .csv file. As the web pages that will provide the data are already known, emphasis will be put on understanding the HTML layout of the page and creating a script targeted for this layout. 'Beautiful soup' module will be used for data mining. Another script will have to be created to split the data into training and test sets. The biggest risk is anticipated in inconsistency of the games played, for example some matches could be cancelled due to weather in certain years, so there would be a difference in the data sets between seasons. As the amount of games per season is around 2460, the team has to avoid any need for manual data preparation, as it would be too time consuming.

Second part of the design will be the machine learning algorithm itself. After a literature study in the field of sports result prediction, a first guess on a type of neural network was made. Multi-layer feed forward network was chosen, and as the team planned supervised learning, the backpropagation algorithm will be added to optimize the learning time. In order to determine the hyperparameters of the neural network, genetic algorithms will be utilized to optimize the configuration of network parameters, for example the number of hidden layers. Genetic algorithms strive to imitate biological optimization processes using methods like mutation, crossover and selection. We hope we can use these algorithms to find the best combination of parameters to configure the neural network we want to use. First a population of varying parameters will be generated, training them on the same data, comparing their performance, selecting the best performing ones, crossing their parameters, introducing some random variation (mutation) and then selecting out of the individuals in this new set of networks. There is a chance that this might be very resource or time consuming though, therefore it needs to be checked whether this approach is viable for the scope of this project, since it is like to increase performance quite a bit. As everyone knows, team sports can be very unpredictable, random injuries can happen during the game or even before it. Thus the biggest risk for this design, is that the neural network that we chose will struggle to find a pattern in the data. If this happens, we will try to refine the data with more parameters, but maybe it will turn out that NBA is completely random and we cannot predict the winner. As the machine learning algorithms are very dependant on the data you feed them with, a special attention has to be brought to splitting the data into training and testing sets. Mistakes done during this splitting will negatively impact the objectivity of the testing.

4

Design Documentation Set Layout

The required design documents will be provided as part of our design plan. These design document will serve as a guideline on which we will base the implementation of our software.

- **Web-scraping plan**: describes how we will scrape the data to be fed to the algorithm from the web and how it will be processed.
- **Neural network description**: describes which kind of neural network/AI we will use and why we chose it.
- **Software architecture description**: describes the interactions between different parts of the software, such as the data pipeline and the neural network.
- **Code implementation plan**: describes the coding strategies/structures that will be implemented.

Although we'll try to avoid diverting from the original design plan as much as possible during the implementation phase, we will use the Scrum methodology with weekly sprints to manage configurations and changing requirements if need be.

5

Design Validation

The purpose of design validation is to check whether the design plan leads to a product that satisfies the original requirements. The main objective to be achieved by our software is to predict the outcome of future NBA games. Thus, the goal of the design validation process is to ensure that every step in the design process leads to this output. To do so, we must check for different types of validity:

- **Internal Validity**: we must make sure that all variables not directly related to the individual skill of a player are considered, such as his age (every player gradually gets worse past a certain age) and whether the team is playing at home or not (the home team has an advantage due to not having to travel for the game). These variables are very likely to somehow affect the result of a game independently of the skill of the players involved.
- **Construct Validity**: the software should assess the individual skill of players independently of which team they may play on. This decision has been made to better assess players moving to different teams. Also, basketball is a fast-paced sport where, according to a general consensus, individual skills are more important than coaching due to the players themselves making most in-game decisions.

6

Testing approach, test planning, validation reporting

As every software, machine learning algorithms have to be tested, to assess whether they serve their purpose. Testing for this project will be split into 3 separate test phases, on different aspects of the software.
First phase of testing will be conducted during web-scraping for data. As a proper data set is vital for machine learning, web-scraping script will be tested as a unit, to assure that:

- Correct data is gathered
- Correct output format of gathered data
- All possible data from certain source is gathered

We chose manual testing over automated testing for this unit, as the combination of inputs is limited, so we can extensively test if we get desired output, without spending too much time on creating test cases. As this unit is a basis for further software development, whole team will be focused on testing, so that the unit is working correctly as soon as possible.
Second phase of testing covers testing of the machine learning algorithm. This algorithm is the core for this project, so it has to be verified, whether it works as requested.This tests will be conducted when the neural network is established. During this phase it will be checked whether:

- Proper input layer
- Correct amount of hidden layers
- Correct output layer
- Correct type of neural network

This phase can be seen as a preparatory phase for machine learning. As the learning process is quite resource intensive, it is smart to check beforehand if the 'education system' (neural network) is bug free etc. This prevents costly mistakes and time delays. The type of testing is not yet determined, but as this neural network has to fit our needs extremely well, the probable choice will be manual testing.
The last phase of testing actually consists of 2 sub-phases. The first phase will be automatic testing of the neural network. The algorithm will be switched from learning mode to running mode. The same data set as during training will be fed into system to check whether with increasing number of training cycles the "training error"is going down. Training error is the difference of predicted output compared to provided output in the training set. If this error is not decreasing with increasing number of trainings, the neural network might have difficulty to find a pattern in the data - this means that the data set needs to be refined with more input parameters or that the phenomena that we want to predict is purely random. The second phase of the testing will be based on test data set, which is not known to the algorithm. This is the actual benchmark of the algorithm performance. NBA seems to be perfect fit for this purpose, because each team plays 4 games (2 home and 2 away) with each team in their conference. This allows for training on 2 games(1

home and 1 away) and then testing on the other 2 games. Otherwise the system would be biased (Home games are usually easier than away games). With the increasing number of NBA season data fed to the system, it is expected that the testing error will decrease.
This testing phase will be wrapped up with manual tests regarding input outside of data set, like 'draw' (which cannot happen in NBA) or teams outside of NBA etc.

7

Implementation Planning

In order to carry out the plan discussed in the previous chapters, each step can be split into smaller steps. The goals we expect to achieve include:

- Gather data from the correct URLs specified earlier, and compile these into a CSV file to be read by the neural network.
- Implement a Back Processed Multi-Layered Perceptron Neural Network
- Have a correct number of input nodes for the neural network
- Have a correct number of hidden layers that would provide the greatest amount of accuracy.
- Have an accuracy over 52%.

Since we have a total of 5 weeks to complete this project, the following weekly schedule will be our plan for creating a working NBA match predictor:

- Week 2.3: Be able to scrape the internet for the correct data via the URLs provided earlier in the report.
- Week 2.4: Have our neural network be able to take the correct number of inputs and provide the correct number of outputs specified.
- Week 2.5: Train the network on the data scraped during week 2.3 and adjust the hidden layers to increase the accuracy.
- Week 2.6: Continue to increase the accuracy by potentially adding more match statistics such as the amount of points scored per player and their total play time.
- Week 2.7: Continue to increase the accuracy of the network by adjusting the necessary nodes.

In order to complete this project on time, we must allocate the proper amount of design time as well as the proper amount of neural network training time. In order to create the entire code, we will make use of the time allotted for the project. However, in order to properly train the neural network, we will make use of a desktop computer owned by a group member, which will be running the code in order to give it a proper graphics card to run on and also the proper amount of time to run.
In order to properly assess the success of this program, the accuracy of the neural network when training on the given data should be used. This would be the most explicit way of measuring the success of our program once the neural network is working.
We will be dividing the group in two during the beginning of the construction of the code. These two groups will be working on a code to scrape the web and a preliminary neural network code.

8

Evaluation

Once the software is fully implemented and completed, its quality must be evaluated from a qualitative point of view. Namely, things such as the quality of the code, the quality of the architecture and, of course, the performance. This is not a high-priority part of the project, however, it would become important if the software were to be used for commercial purposes. As the team is small the nature of the interaction will be self organizing with lots of communication between the members. This will ensure that design decisions are uniformly agreed upon and code standard will be kept. Although the primary aim is mainly to produce functioning software, refactoring code and keeping the development process organized is important for debugging and changing the design later. Since the length of this project is only five weeks, time is really a resource that needs to be managed.

With respect to performance, the network will asked to predict the outcomes of current games, to see how well it does at determining the winner. Reliability is also another performance parameter. This needs to be measured similarly over an (ideally) extensive period of time, which might not be possible at the completion of the project. Reliability needs to be tested over the course of a new season, therefore the team decided to preliminary assess reliability qualitatively, subjectively rather than generating a valuable requirement, that would be hard to achieve in that time frame. However like many others, we will compare the predictions of the network to the predictions of experts in the sport, to gauge how experienced the network is, and whether its conclusions line up with other trained observers of the sport.

Bibliography

Gerritsen, B., *WI3615TU – CS Minor Projects BigData and AI*, Delft: Delft University of Technology, Nov 2019.