Monash University

Information Technology

FIT3036

Computer Science Project

Semester 1, 2018

Project 4

David - Friday 12pm

Predictive Sports - Final Report

Jake McNee

25198572

May 29, 2018

Word Count: 3667

Contents

1 Abstract

Probabilistic functions can be used to add a element of skill to usually simple tasks. In modern times most football tipping competitions have a simplistic scoring set up, designed to be based more on luck then educated estimations. (Dowe et al., 2011) has developed a system which turns a normal tipping competition into a version which uses logarithms to devise a new scoring system. This rewards users for predicting the chances of a team winning, rather then a binary choice. Instead of simply picking which team will be victorious, users are able to enter a score between 0 and 1, which determines what probability they believe team A will win over team B. The logarithm function used to be able to give these desired outputs is as follows.

$$Score \ = \ 1 \ + \ log_2(x) \qquad where\ x\ is\ the\ guess \quad x = (0,1)$$

This shows the function that is used to determine the score a player receives based on their tips. This guess is picking one team to win over another team. Below are examples of different key scores a player can receive.

First being the minimum score with the lowest guess a player can enter.

$$1 + log_2(0.001) = -8.966$$

Similarly the highest score a player can receive occurs when the highest probability possible is given.

$$1 + log_2(0.999) = 0.999$$

Finally if a player does not think either team have a better chance to win, they have the option to guess directly in the centre, giving a score of 0.

$$1 + log_2(0.5) = 0$$

It can be observed from these values that a player is penalised much more for guessing wrong, compared to if they guess right.

1.1 Key Words

Probabilistic, Sports, Tipping, Predicting, Firebase

2 Introduction

This project is an extension on an existing competition created by Prof. David Dowe. The competition takes a different approach to normal binary tipping competitions that most are based on in recent times. It provides an interesting insight into how different sports can have different degrees of randomness. This report will go into detail about the main ideas behind the project, including objectives, limitations, requirements and future work.

2.1 Project Objectives

The main objectives of this project is to extend an existing probabilistic tipping competition, to one that could be generalised for all sports. The idea is the existing competition is built for only AFL, however to be more inclusive of all sports, creating a generalised version allows a user to start a competition for whatever sport they choose.

2.2 Requirements

- Spreadsheet Of Users

For the system to generate the user data, a spreadsheet of all users must be present in the project code folder. It must have an extension of .xlsx for the system to be able to convert the data into Google Firebase. The file simply needs to be present in the 'input documents' folder and the system will implement it.

- Spreadsheet Of Fixtures

Similarly in the same location as the spreadsheet of users, the fixtures must also be placed here. Following the same .xlsx format, the system will import all data in the Firebase and create the table..

2.3 Constrains

Due to time constraints of this project not all ideal functionality could be implemented. The main goal was to get a generalised version that could work with many projects, some functionality was left out.

- Local Host Server

  The system has been implemented in such a way that it requires the admin to run the server file on their local machine. Once the server is running, clients can connect to the application and upload their tips. However if the local host turns off and restarts, the competition will also reset.

- Pre-Chosen Data

  For this system it is required that the admin inserts the fixture file and user file in a spreadsheet in the root directory. This allows the system to pull this information and transfer it to Google Firebase.

- Email Log In

  The functionality in this project is set in such a way that clients can log in by simply entering in their email address. If this matches the database, they are able to enter the competition.

- Results Automated

  In this competition, it is set up in a way such that the home team is always the victorious team. This means each users will be successful if they predict the home team to be the winner.

3 Background

3.1 Preparation Literature

During the research of this project there was a shift from neural networks in predicting sports, to journals based on mathematical models. Initially many of the journals that were researched had strong links between probabilistic guessing of sporting outcomes and how the use of neural networks achieves this. (Loeffelholz, Bednar, & Bauer, 2009) found interesting results using neural networks to predict NBA matches based on past history. Using 620 NBA games as input data, correct results could be predicted 74.33 percent of the time (on average). Similarly it was shown in the sport of javelin throwing, results could show a significantly greater accuracy in using neural networks over regression models (Maszczyk et al., 2014). This kind of information could be a useful precursor to further research, however for the scope of this project it was decided to not implement neural networks.

Studies with a basis on mathematical models and not on neural networks provided some common themes that lay a foundation for this project. One theme identified has been studying past data can lead to higher sucess rates of picking a team to win (Wunderlich & Memmert, 2016). However there was more research to suggest randomness is a bigger factor. In one example, it was shown in some cases non-experts were just as likely to guess a correct team to win compared to experts. (Andersson, Edman, & Ekman, 2005) discovered some insights into the somewhat randomised nature of sporting outcomes. Experts such as soccer fans, sports journalists and soccer coaches were tasked to predict the outcome of the 2002 soccer world cup. The predictions of these experts were compared with those who come from a background of little to no knowledge of the sport of soccer. The results showed both parties were equally accurate at predicting the results. This has an interesting link to this project as the tipping in the world cup competition was answered on a win/lose/draw basis, compared to this project which measures the probability of a team winning. In further studies this competition could be used by experts and non-experts in a given sport to have their results compared to see if it makes a noticeable difference.

3.2 Project Risks

The risks for this project were broken down into four states: Very unlikely, Unlikely, Likely, Very likely. In addition, the risk projected from the plan are addressed to evaluate how much of an impact they had during implementation.

- Fixture Variances

  Description: Although most sports share common elements they can all have different rules and scoring systems. Since this competition is to generalise all sports into one user friendly system, different fixtures were predicted to potentially cause problems.

  Predicted Probability: Unlikely

  Risk Outcome: Throughout the implementation steps were made to try and make this as generalised as possible. However the main solution to this risk was the use of spreadsheets to be imported as the fixture. The spreadsheet used follows a generalised format which encapsulates most sports and their respective rules. Common themes such as home, away, date, location, time, round are the main categories the spreadsheet follow. The majority of sports can be slotted into this format and once they are, the system can create the competition.

- Data Security

  Description: As a user it is expected that the data you provide to websites and applications is going to be kept private and not shared without your knowledge. Data attacks and hacks however can fault this assumption and it is necessary that measures are put in place to prevent this.

  Predicted Probability: Very Likely

  Risk Outcome: Originally it was planned that a simulated database would be used with the possibility of a SQL database implementation to house all data. It was decided that Google Firebase was a better option. Firebase has the options to make your database private and data attacks such as SQL injections can be prevented. In the scope of this assignment, however password have not been implemented however Firebase provides relative ease for this addition in the future, the main idea of this project was to get a working system and set up the security such that it can be added in later without difficulties.

- Database Functionality

Description: Testing may be difficult due to database being in a simulated format. With a feature that allows users to create accounts, testing can be provided for that functionality.

Predicted Probability: Likely

Risk Outcome: During this project the original idea of a simulated database was kept. A spreadsheet which is located in the project code section, holds all information about the users. The system was designed such that it transfer all this information to Google Firebase and can update it in the future from there. Using further implementation, an admin could add users to the original spreadsheet or use Firebase functionality to add users to the user child node of the Firebase database.

3.3 Resource Requirements

- Hardware Resources

  Admin: The admin or server operator of this system requires a computer. At time of implementation this system cannot be run on mobile device or tablet. The computer is required to run the node.js server to allow clients to connect to the system.

  Client: Unlike the admin the client has a lot more freedom in the ways they wish to connect to the competition. Any device that can run a webpage can be used, such as a computer/mobile/tablet. If the competition is particularly big a computer may be preferred, due to the bigger screen to view the content.

- Software Resources

  Web Development The Website was created using PhpStorm which provides functionality support for Node.JS/HTML/CSS/PHP. The main web page was created using HTML with some CSS. The language version for the Javascript used is ECMA Script 6.

  Spreadsheets: The spreadsheets that were used for the fixture and user information were provided in .xlsx format, this was required since the module used to convert the spreadsheet to JSON and then to Firebase requires this format.

  Cloud Database Google Firebase was used for all cloud database implementation. Scores/Fixtures/Users/Game Data were all stored in this database.

  Main Server Node.JS has been used to code this file. Although most of the logic has been implemented on the client side, this allowed a simpler way to create the table and extract data.

4 Method

4.1 Methodology

- Needed Modules

  Since a Node.JS system was to be used modules such as Socketio, Express, XLSX (for converting .xlsx files to json), firebaseAdmin were installed. A npm init file was created in the directory of project code and npm install used for all modules. Furthermore, the Firebase was set up on firebase.google.com and the serviceAccountKey created to allow access for the server. An express server was created and port 8000 was the local host used to run the program.

- Supporting Documents

  Next the spreadsheets that were needed for the implementation were created. One being the fixture spreadsheet. One was used that followed the 2018 AFL structure as this gave a ideal generalised form that could be used for all sports. A spreadsheet for users was also created.

- Firebase Implementation

  After the Firebase modules were completed, functions were then created to convert our spreadsheets into fixtures and user nodes, in our Firebase database. In addition, a game data node was created to hold important game features such as number of rounds/users/games.

- Create User Inputs

  For the main project table, a generalised table is first generated based on the number of teams and users. Once this is created inputs are given for whichever player has logged onto the application. Once their tips have been entered, they are updated onto the Firebase. Finally data is then applied to the visible table so that it reflects the corrosponding Firebase results.

- Generalised Structure

  With all the parts in place, a game format was ready to be created. A variable which reflected the current round was used to dictate when scores should show for each round. Additionally, it gave the game a start and finish condition which allowed other variable initialisations to be reset.

4.2 Internal Design

The following digram (Figure 1) depicts a generalised logic structure that the program follows. This is the cycle a user follows from logging on to the application to entering the tips with the scores being calculated.

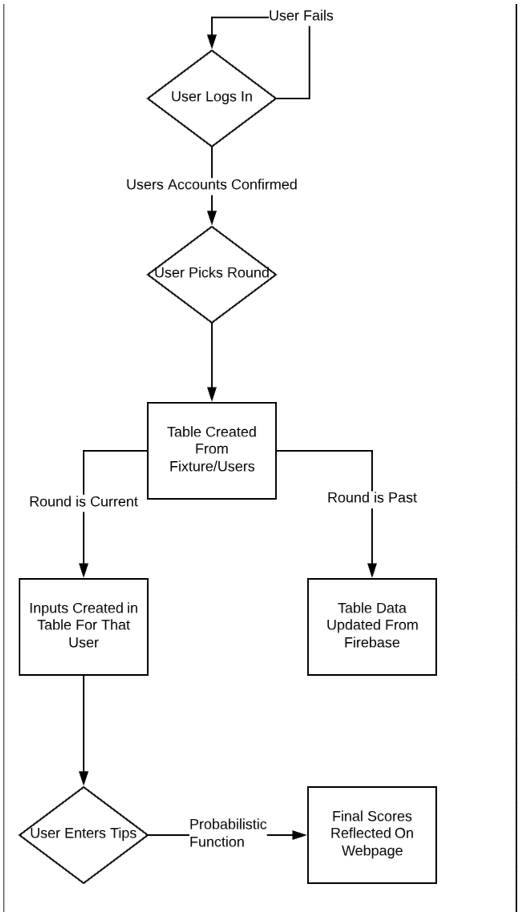

User Fails
User Logs In
Users Accounts Confirmed
User Picks Round
Table Created From Fixture/Users
Round is Current
Round is Past
Inputs Created in Table For That User
Table Data Updated From Firebase
User Enters Tips
Probabilistic Function
Final Scores Reflected On Webpage

Figure 1: General Logic Diagram (Credit: lucidchart.com)

The following digram (Figure 2) shows a basic UML diagram of the experience for the user.

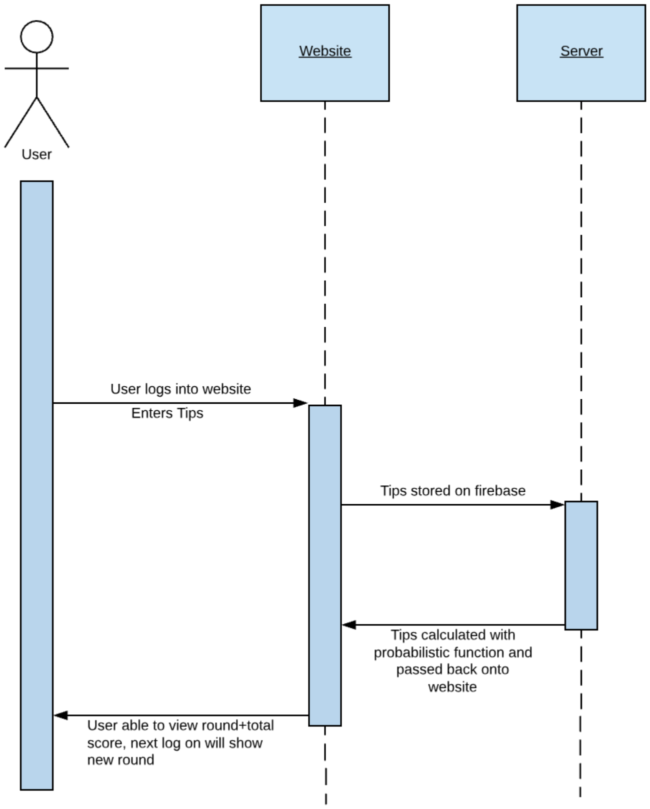

User
Website
Server
User logs into website
Enters Tips
Tips stored on firebase
Tips calculated with probabilistic function and passed back onto website
User able to view round+total score, next log on will show new round

Figure 2: UML diagram (Credit: lucidchart.com)

4.3 Software Architecture

The following diagram (figure 3) depicts the data structure that Firebase implements. This is how data is saved and when items are updated they stay in these data structures.

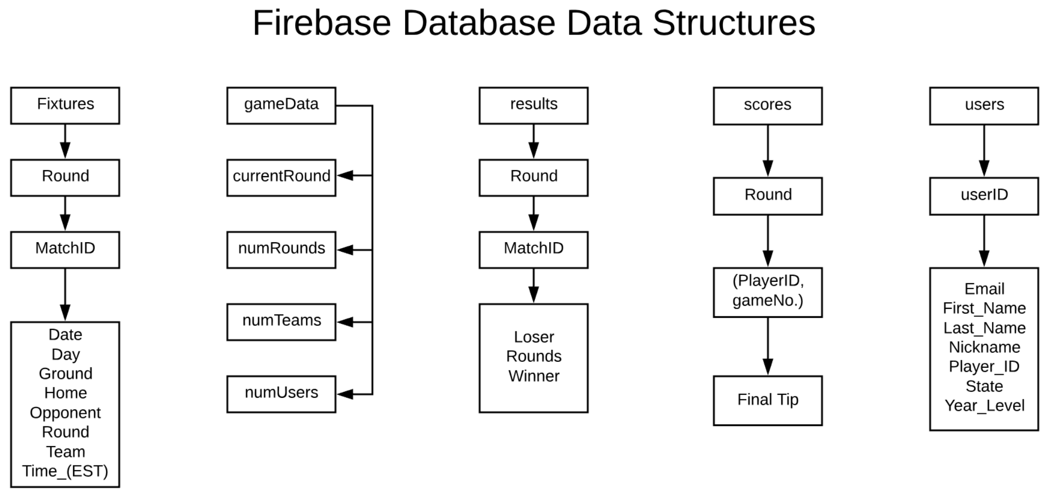


Figure 3: Firebase Data Structure (Credit: lucidchart.com)

5 Results

5.1 Type Of Results

The results that this program provides can be broken down into three main categories.

- Firebase Functionality

  Throughout the implementation of this program the Firebase Database that is attached to this project is constantly being updated. It happens initially when the spreadsheets for users and fixtures are converted into a JSON dictionary, which is then able to be stored in Firebase. Additionally, game data, scores and results are also stored here. The results this provides for this project is, you are able to see the relevant scores and game data that is required for the program to run.

- User Feedback

  There are many instances throughout the competition where a user receives some feedback, such as, entering an incorrect values or not using the program as expected. For example, if an incorrect email is entered on the first page, the user will be alerted that this doesn't match the database. Additionally, on entering values for the input that are either not numerical or outside the range (0,1) the user is alerted. This kind of feedback can be tangible results for the user as well as the admin to get an idea of where the system is progressing.

- Player Scores

  The final type of results that can be identified in the system are the scores a player gets throughout the tipping process. It is observed that each input shows the corresponding score through the probability function. This is added up to show the round total, which is also used to work out the match total. These numerical results can be tested to show they display correct results.

5.2 Externally Observable Features

- Sign In Page

  The first page the user comes to is the sign in page. It gives an input for the user to enter their email to log onto the application. If the user enter an email that doesn't match the database, an alert will be given informing them of a incorrect value entered.

- Main Table Page

  Once the user has logged on they will be greeted and informed to click on the desired round tab. The tabs are designed such that only the current and previous tabs will appear. Once first on this page, only the round 1 tab will be present. When a round tab is clicked, a table is generated based on number of users and number of teams. If this represents a past round, corresponding data from Firebase will appear. Similarly, if another user has entered their tips for that week, they will be present. The users are listed on the 1st column and teams on the 1st row. Additionally, the last 2 columns will show the score for the round and the total score for that user. Input boxes for tips will only appear for the user that is currently logged in.

5.3 Performance

- Space Complexity

  In this implementation the only data that is required to be stored on the server is the inputs that are required. First being the list of users (U) and second the list of rounds (R). Therefore the space complexity on the system is O(U+R). The use of Firebase allows all other information to be stored on the cloud server.

- Time Complexity

  Time complexity is based on the biggest factor to affect the program, since when input values get very large, whichever function has the highest degree of complexity will make a significant difference. The factor that has been identified to have the biggest impact occurs during the creating of the table. To make the table there is a for loop for number of games (G) nested inside another for loop for number of users (U). This means the time complexity with the most significant factor is O(U*G). This quadratic time can slow down the system with large users or games. To prevent this, different competitions would be implemented depending on the number of users.

6 Analysis and Discussion

The main outputs that could be analysed from this project were the tips that get entered via the user. From the test cases, it was observed the tips entered give the correct results per cell and the total round sum correctly adds each item in the row. The final column, which represents the total score for that user, is based off the previous round and is added to that value. The other interesting thing to note about this project is it can be generalised for all competition. The only requirement is the fixtures are implemented in such a way, that they follows the spread sheet format used for this competition.

7 Future Work

- Local Host Server

  Future implementations could use technologies such as Firebase Functions to host the server. This allows the server to be constantly running and no local machine needed to enable use.

- Auto Generated Results

  Due to limitations in time, the results for these matches are coded such that the team in bold presented in the fixture is always the winner. Therefore the score the user enters in the input is always given a winning calculation. Further implementations could import a spreadsheet of results directly to the server which decide the winners and losers of each round.

- Gaussian Tipping

  Variations of the competition could be implemented for different results. The Gaussian competition (Dowe et al., 2011) is one example of a different scoring system. The implementation would follow a similar structure to this competition, all that would be required is a function to compute the Gaussian score based on a tip, such function would be inserted into the main HTML page.

- Client Side Adding Of Data

  As the current implementation takes all the data from the pre made spreadsheets, future work could extend this functionality to allow client side additions. For example, if a new user logs onto the page they could enter their personal details to register themselves in the competition. Similarly, if the admin wanted to create multiple competitions, they could directly upload their fixture onto the client page, which would create the additional game for people to join.

- Additional Security

  In addition to the email log in system that is currently being used, security factors such as a password, two-factor authentication and security questions could be implemented. This will allow the data that users enter on the page to be more secure and be safe against outside threats.

8 Conclusion

The competition is in a format which allows most sports to be able to insert their fixtures and the system will generate the website. The use of Google Firebase allows all data to be saved externally and not on a local machine. This project provides a foundation to further development to identify new results. Software features such as Firebase Functions for a constant on-line server or further data security are some of the possible future works for this application. In terms of different research, an experiment using this application partnered with experts and non-experts in virtually any sport could provide interest results. Since the probabilistic nature is different to most binary tipping competitions, there could be tests to identify if experts do have a greater understanding of the sport and therefore more likely to achieve greater results through this competition.

9 References

References

Andersson, P., Edman, J., & Ekman, M. (2005). Predicting the world cup 2002 in soccer: Performance and confidence of experts and non-experts. *International Journal of Forecasting*, *21*(3), 565–576.

Dowe, D., et al. (2011). Probabilistic and gaussian football tipping. *Vinculum*, *48*(2), 10.

Loeffelholz, B., Bednar, E., & Bauer, K. W. (2009). Predicting nba games using neural networks. *Journal of Quantitative Analysis in Sports*, *5*(1).

Maszczyk, A., Gołaś, A., Pietraszewski, P., Roczniok, R., Zając, A., & Stanula, A. (2014). Application of neural and regression models in sports results prediction. *Procedia-Social and Behavioral Sciences*, *117*, 482–487.

Wunderlich, F., & Memmert, D. (2016). Analysis of the predictive qualities of betting odds and fifa world ranking: evidence from the 2006, 2010 and 2014 football world cups. *Journal of sports sciences*, *34*(24), 2176–2184.

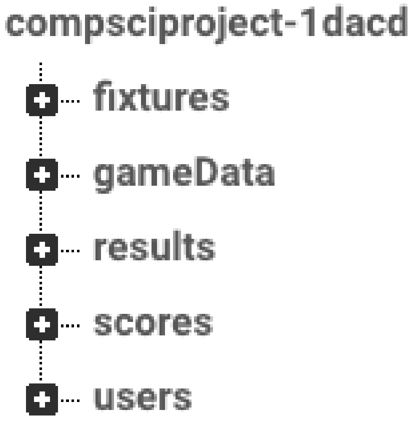
10 Appendices

10.1 Production and Deployment

- Steps to run program (commands in bold are required to be run)

  1. Unzip folder and save 'JakeMcNee-A2-ProjectCode' in a directory of your choosing.
  2. Use terminal/command prompt to reach directory and run the following npm commands INDIVIDUALLY to install node modules.
  3. **npm install express –save; npm install socketio –save; npm install xlsx –save; npm install firebase-admin –save**
  4. Once modules installed, use this command to begin server. **node app.js**
  5. This will begin running the server, go onto a web browser and type in the following **localhost:8000**
  6. Now you are on the client page, enter in a user email, an example one is there for you to click on, otherwise in (*rootDirectory* > inputDocuments > simulatedDatabase) there is a list of user information and any email can be used.
  7. Once logged in, you can enter your tips.

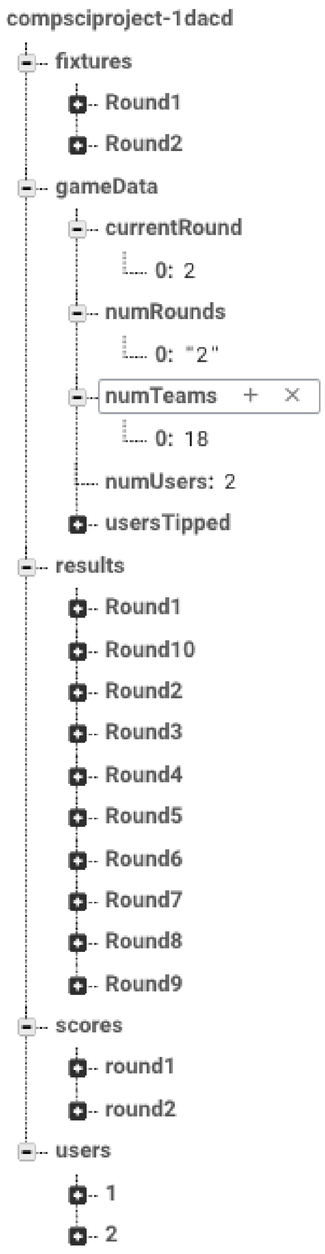
10.2 Firebase Screen Shots



compsciproject-1dacd

- fixtures
- gameData
- results
- scores
- users

Figure 4: Firebase database not expanded (Credit: firebase.google.com)

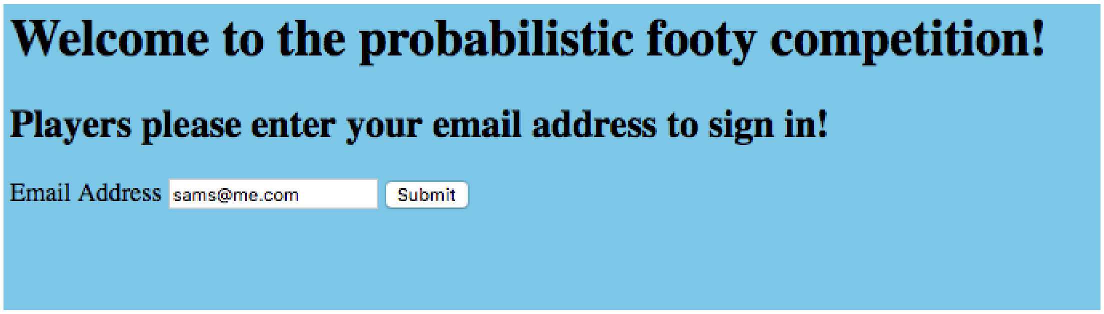


```
compsciproject-1dacd
  fixtures
    Round1
    Round2
  gameData
    currentRound
      0: 2
    numRounds
      0: "2"
    numTeams  + ×
      0: 18
    numUsers: 2
    usersTipped
  results
    Round1
    Round10
    Round2
    Round3
    Round4
    Round5
    Round6
    Round7
    Round8
    Round9
  scores
    round1
    round2
  users
    1
    2
```

Figure 5: Firebase database expanded (Credit: firebase.google.com)

10.3 User Interface Screen Shots

User starts on this page, enter email address to log in.



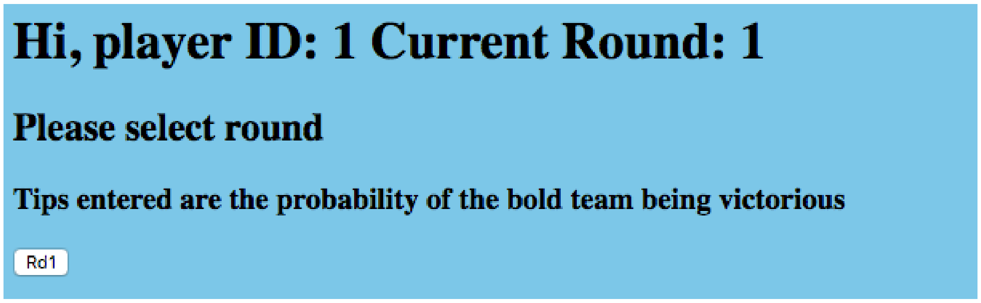
Welcome to the probabilistic footy competition!

Players please enter your email address to sign in!

Email Address sams@me.com

Figure 6: Welcome page

User clicks on desired round.



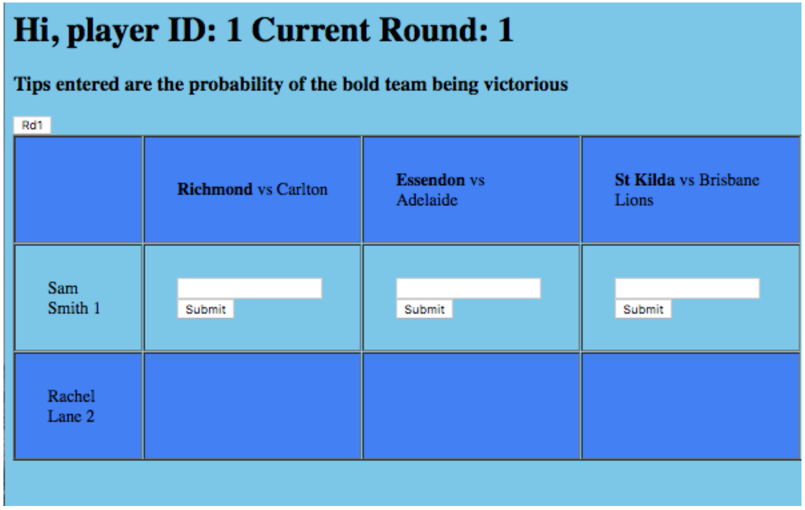
Hi, player ID: 1 Current Round: 1

Please select round

Tips entered are the probability of the bold team being victorious

Figure 7: Select round page

User enters tips and hits submit at end (last submit not shown in screen shot)



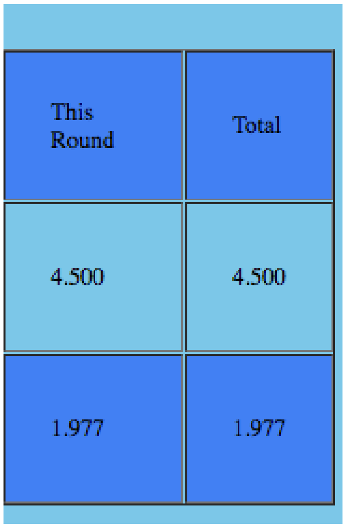
Hi, player ID: 1 Current Round: 1

Tips entered are the probability of the bold team being victorious

	Richmond vs Carlton	**Essendon** vs Adelaide	**St Kilda** vs Brisbane Lions
Sam Smith 1	Submit	Submit	Submit
Rachel Lane 2			

Figure 8: Enter Tips)

Tips displayed



This Round	Total
4.500	4.500
1.977	1.977

Figure 9: Results

NB: word count excludes title/table of contents/references/this footnote