Contents Page

1. System Overview

The MMA Fight Predictor is an android based application that can be used to predict the winner of Mixed Martial Arts bouts. The app consists of two modes which are extremely user-friendly and which can be used to predict an outcome.

The first mode is known as the “Calculate” mode. This uses our specially formulated algorithm to choose the winner. This mode also has two possible paths the user can take. The first simply involves the click of a button to use our algorithm, along with fighter data stored in our database, to predict the outcome of the fight. The second path

allows the user to interact more with our algorithm, as it provides the option to add an opinion regarding which fighter is better in certain areas of Mixed Martial Arts.

The second mode available in our app is known as the “Create Algorithm” mode. This segment of the app allows the user to take advantage of our simple user interface, consisting of interactive sliders, to weight the importance of certain aspects of MMA in a particular fight, thus creating their own algorithm. They are also quizzed on which fighter they think is more dominant in each skillset involved in MMA. In simple terms, this mode provides the user with an opportunity to create an algorithm to predict MMA bouts, without requiring knowledge in areas such as genetic algorithms, or even computer programming.

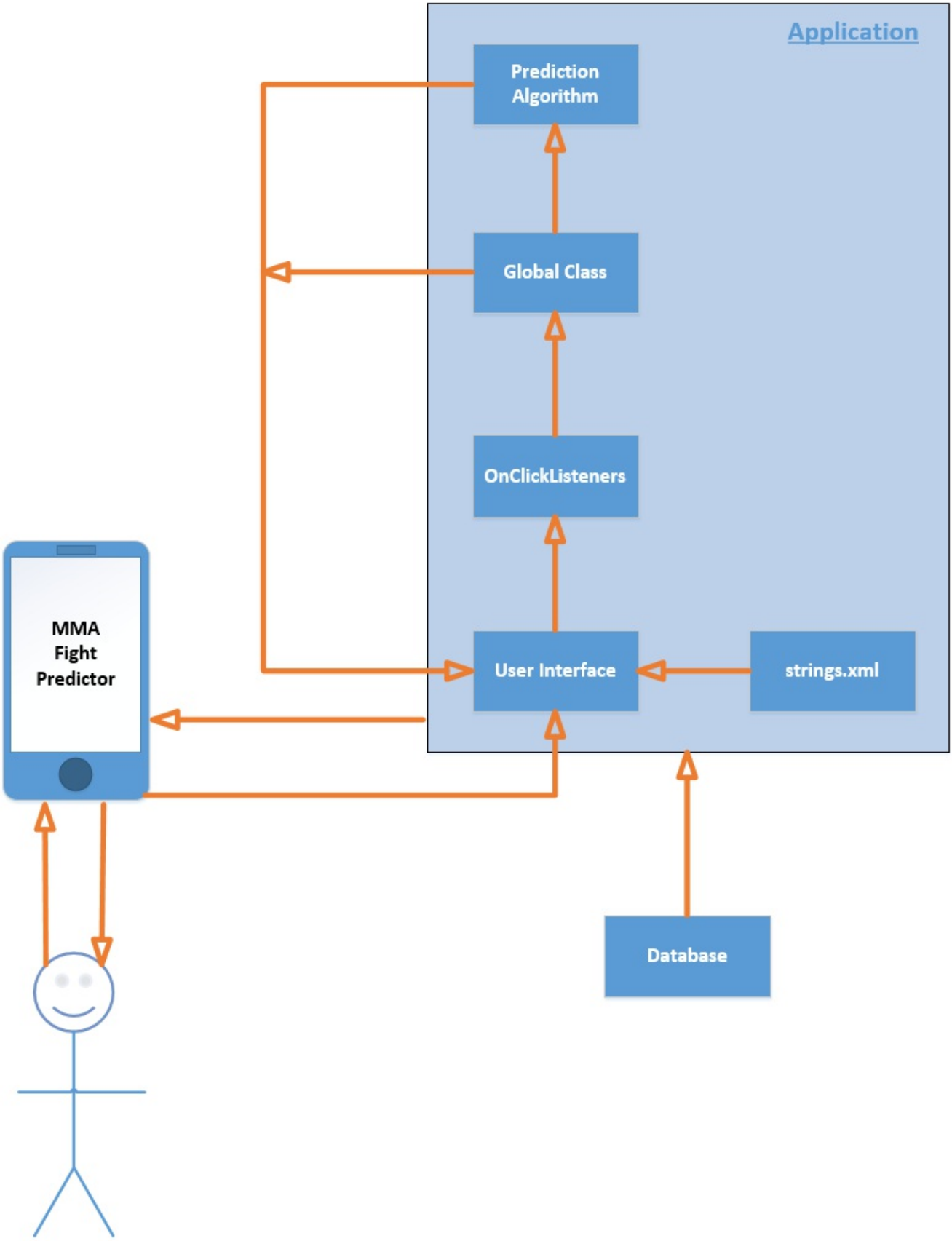
2. Glossary

- Mixed Martial Arts (MMA): A form of combat sports, involving all aspects of fighting, inclusing striking and grappling.
- Octagon: An eight sided cage in which the fight takes place.
- Brazillian Jiu Jitsu (BJJ): A from of mixed martial arts involvng chokeholds and submissions.
- Wrestling: A form of mixed martial arts involving takedowns and submission holds.
- Algorithm: A series of instructions carried out by the computer to complete a task.
- Genetic Algortihm: Used to find optimal solutions to problems

by using a process that is very similar to biological evolution.

- Gene: An integer that is used to make up part of the identity of a Genotype.
- Genotype: A series of genes.
- XML: A markup language used in android studio to design the appearance of an app.

3. System Architecture

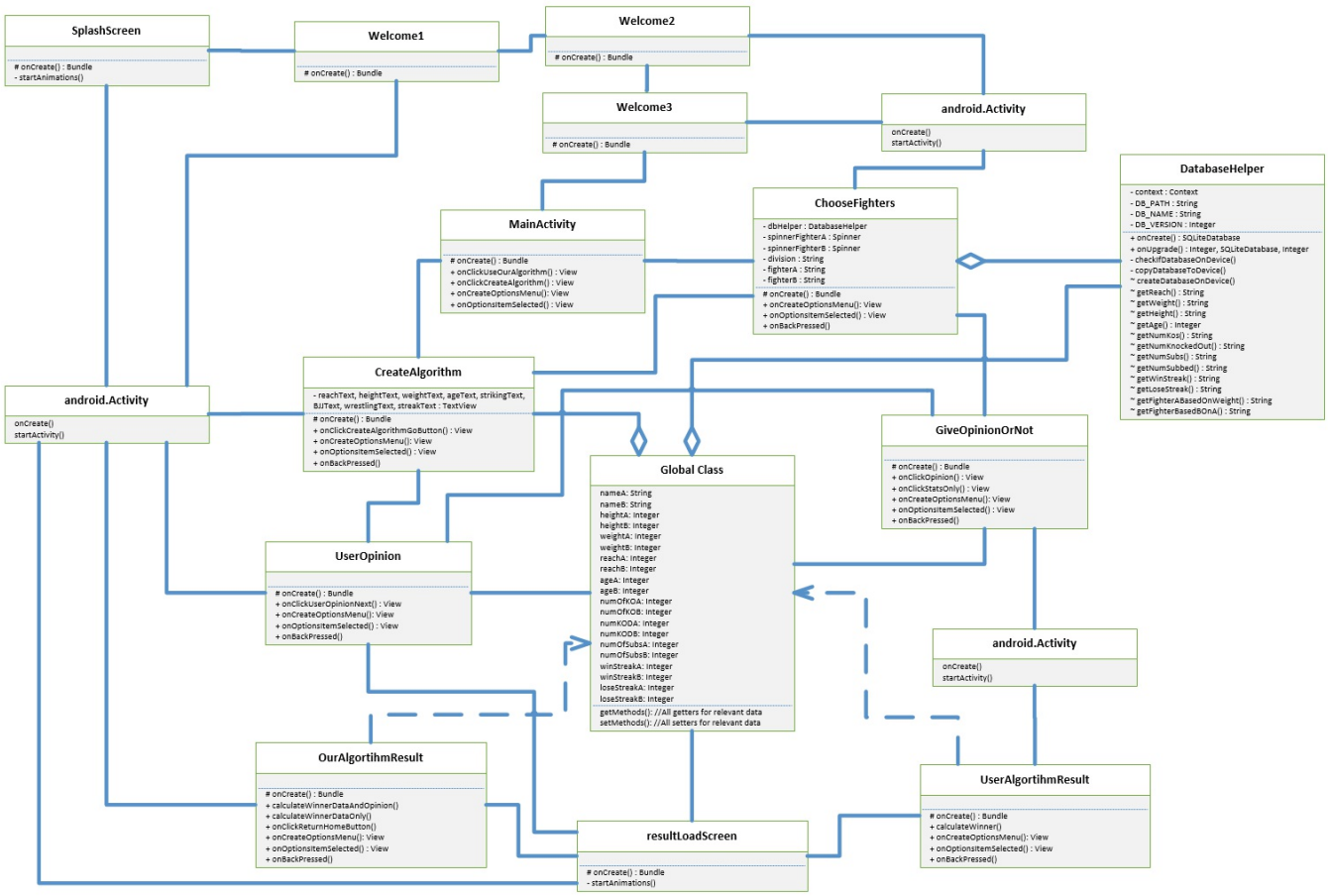


The system architecture of our project is almost identical to the one we portrayed in the functional specification in December. There are some small changes however, as the design and implementation process

meant it could not be done exactly how we had first intended. The main change is that we removed the fighter class, as we instead decided to store the information on both fighters within the global class, once it is retrieved from the database. This also resulted in the global class linking back to the User Interface to display all relevant information, instead of the fighter class.

With these changes in place, our application still has the same system architecture that we had originally intended. This begins with the user interacting with the device to input information, all while all relevant information is being displayed to the user through the user interface. This information will come from both the database, which will contain fighter information, and the strings.xml file which will contain all the sentences used on the display, such as questions or instructions for the user. The user then takes advantage of the onClickListeners, (otherwise known as buttons, sliders, etc) to progress through the application. All information provided by the user, and all information displayed to the user, are stored and retrieved from the global class. The information from this class is then fed into the prediction algorithm, which makes a prediction regarding the winner of the bout. The name of the fighter is then displayed to the user through the User Interface, and the cycle can be repeated.

4. High-Level Design



This Class diagram shows all the classes contained within our Android Application, as well as the relationships between them. All of these classes are activities, except for Global Class and the DatabaseHelper Class.

The application begins by launching the Splashscreen, which runs as a separate thread to display an animation while the rest of the application is being prepared. This avoids displaying a blank screen to the user. The Splashscreen then moves to either the Main Activity, or the first of three welcome screens. By default, the welcome screens are the first choice, unless the user checks a box labelled “Do not show this again”, which will disable the welcome screens. This is done by setting a value equal to true in the Shared Preferences, which are similar to a global class, with the main difference being shared

preferences are permanent and do no reset after the app shuts down. The user then navigates through these welcome screens, which explain the different modes in the application as well as how to use them. Upon finishing the welcome screens, the user will be redirected to the Main Activity, or Home Screen of the app.

In the Main Activity, the user is presented with two options. These are “Create Algorithm”, and “Calculate”, as shown in the class diagram. If the user chooses the create option, a Boolean is set in the global class so the path of activities the user is taking throughout the application can be determined, as both modes re-use some classes. This is seen when both modes use the ChooseFighter class, when the user is being asked to choose which fighters they wish to predict. Both of these modes proceed to the Choose Fighter activity, with the only difference being the Boolean. In both modes, the fighters chosen are saved to the global class for later use. Information is also retrieved from the database by creating an SQL Query using the fighters name. This data is also stored in the global class, and contains information regarding reach, height, number of knockouts, number of submissions, etc.

If the chosen mode is Create Algorithm, the user is then moved from choosing fighters to a screen displaying a series of sliders, known as ‘seekbars’ in Android Studio. We considered these a more efficient way for the user to rate the importance of each attribute in a specific fight, by simply moving the slider to choose a number between one and ten. By choosing the significance of each of these skillsets, the user is being granted the ability to create a prediction algorithm without requiring any programming knowledge. This information is then stored in the

global class, allowing the user to progress to the next class, which is UserOpinion.

In this activity, the User has the ability to rate which fighter is more skilled in the areas of Striking, Brazilian Jiu-Jitsu, and Wrestling. These ratings are also selected using seekbars for ease of use. The advantages of each skillset range from significant, to slight, to neutral. This allows the user to add their opinion on which fighter is more skilled in each aspect of MMA. After completion of this activity, the user is directed to the result load screen, which is a thread containing an animation which runs while the prediction is calculating in the background. From here the user proceeds to the result screen, which displays its prediction of the winner of the chosen MMA bout, together with a percentage of how certain it is of the result. The user can choose to navigate backwards to previous activities to edit information by pressing the back arrows at the top or the bottom of the screens. Once finished, they can press the finish button to be redirected back to the Main Activity. From here, they can choose the other mode, labelled Calculate, to use our specially formulated algorithm, which was created by our genetic algorithm, which we will describe now.

The way in which our genetic algorithm works is as follows. We start out with a population of individuals who have genotypes that contain instructions on how to build an individual that has the ability to predict the outcome of fights. At the beginning of a run, all individuals have genotypes with completely random values. The algorithm then starts to run, and each individual attempts to predict the outcomes of 30 fights which we provide to it. The amount of fights that the individual

predicts correctly is known as its fitness. Then all of the individuals with the highest fitness among the population will be allowed to pass on their genotype to the next generation of individuals by means of crossover, which we will explain shortly. The less fit individuals will not pass on their genotype to the next generation and will die out. As such, the new generation will be a mixture of all the fittest individuals. After that, the older generation is destroyed and the process begins again with the new generation. This keeps repeating until the population of individuals stops evolving. In order to achieve this we used a number of classes.

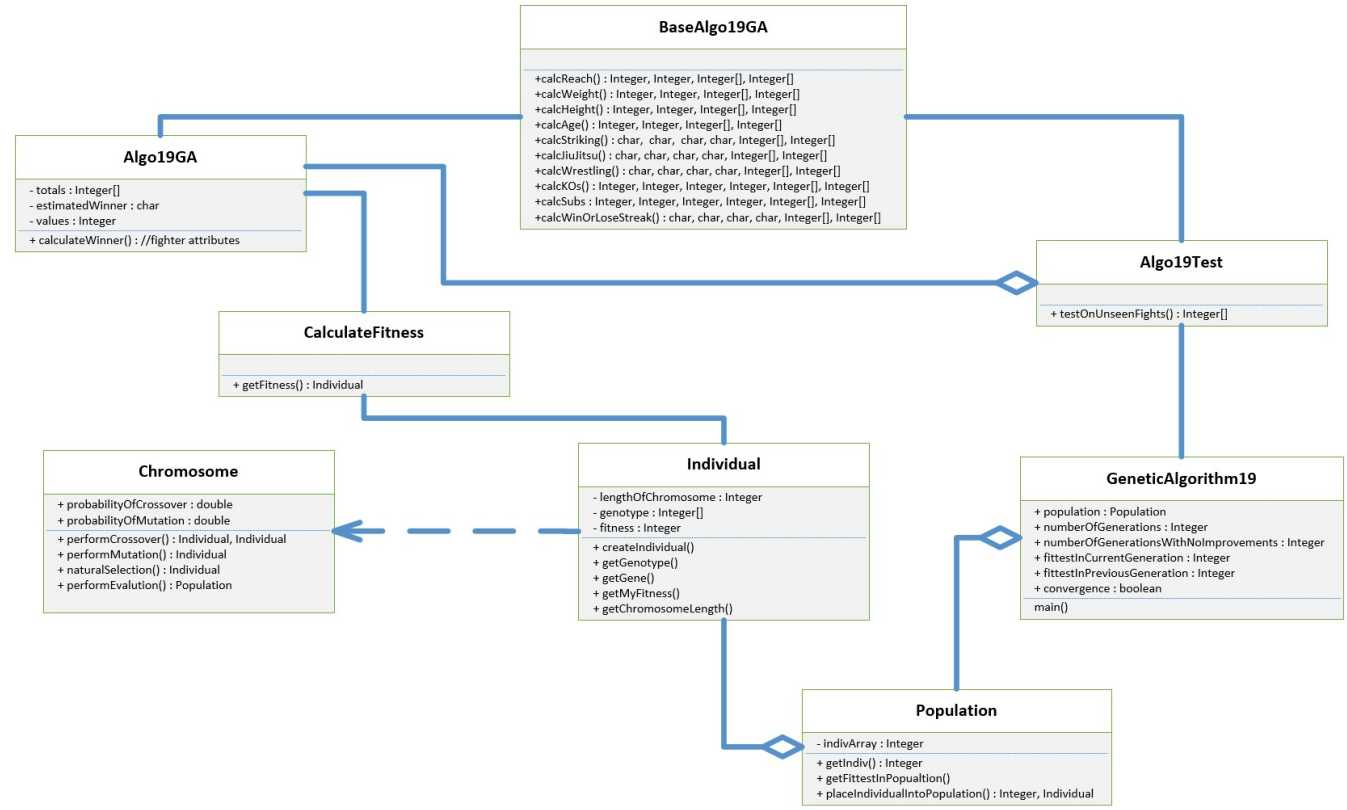
Firstly, we created an ‘Individual’ class which contained an individual’s genotype and methods to get and set specific genes. This class also had a method to create an Individual with a completely random genotype.

Next, we built a ‘Population’ class whose job it was to build a population of individuals. This class also had a method to get the fittest individual among a population of individuals, as well as some getters and setters.

The ‘Chromosome’ class used both the ‘Population’ class and the ‘Individual’ class. This class contained a lot of important methods. It had a method to perform crossover, which took the genotypes of two fit individuals, and used these genotypes to make offspring with similar, but slightly different genotypes. It also had a method for performing mutation, whereby an individual’s genotype would be slightly altered occasionally, in an attempt to make a fitter individual. Next, there was

a method for natural selection, where we would find the fittest individual among five random individuals in a population, and return that individual. All methods described above were used in the method for performing evolution, which is the crux of this genetic algorithm. Here, we would take the old population and use natural selection, crossover and mutation to build a new, and hopefully fitter, population of individuals.

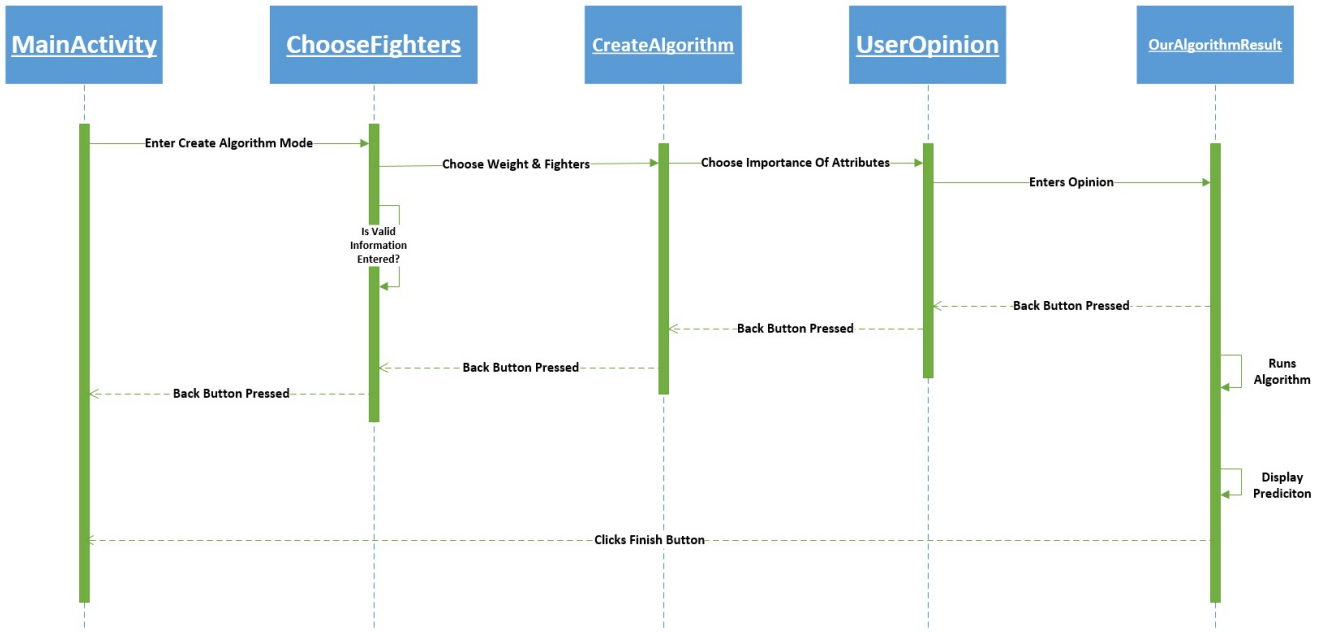
The remaining classes in the genetic algorithm were used to test the fitness of individuals. We had a base class which contained methods for forming a prediction using data taken in from text files that we built. Essentially, we tested Individual's prediction ability on 30 fights that had already taken place. We also tested an Individual's ability to predict fights that were previously unknown to it. The only other part of the genetic algorithm was the main method, which used all classes described above to actually run the genetic algorithm and print out relevant information along the way.



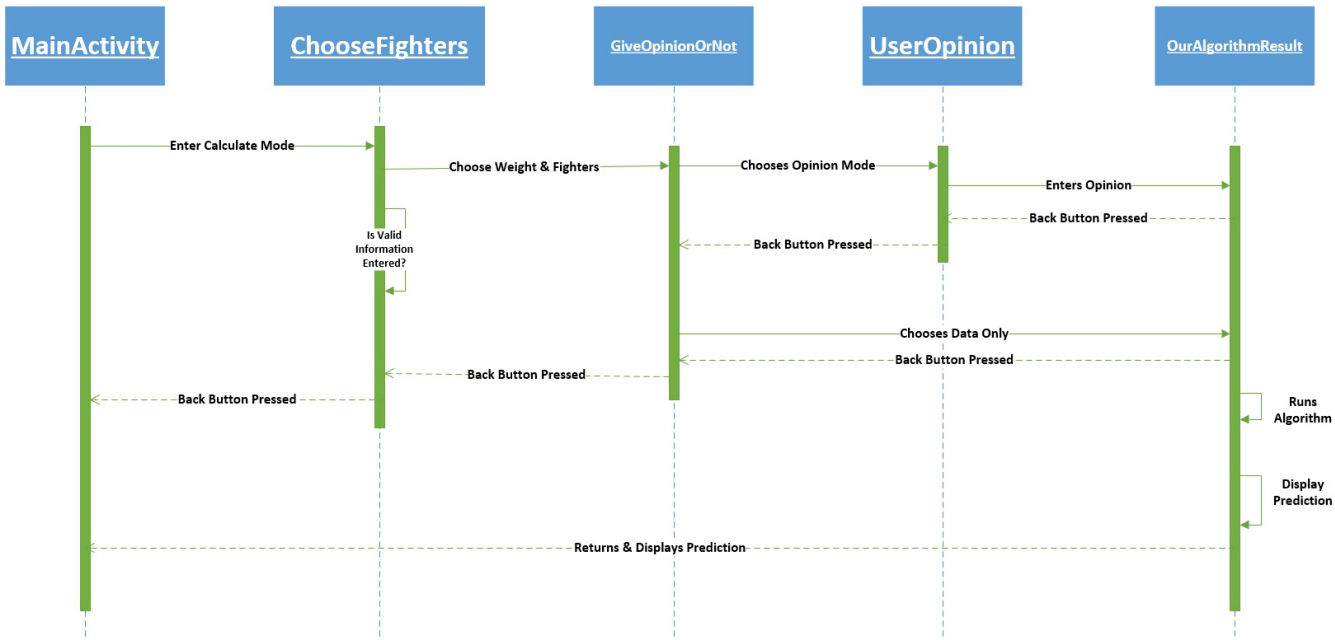
As previously stated, the user will be first be directed to the Choose Fighters screen. From here, they will progress to a “GiveOpinionOrNot” screen. This activity gives the user the option to add their own personal opinion on the fighting abilities of each competitor, or to purely use the statistics stores in our database to calculate the winner. If the opinion option is chosen, the user will be redirected to the Opinion screen mentioned above, also setting a Boolean value so a specific path can be followed.

If the statistics only option is chosen, the user will be directed immediately to the same result load screen discussed above, followed by the algorithms’ prediction for the fight.

Create Algorithm Mode



Calculate Mode



5. Problems

5.1 Use Of Neural Networks

When we first began this project, and as previously stated in our Functional Specification, our original intention was to implement a neural network to create an algorithm to predict MMA bouts. This algorithm would learn from past fights and attempt to “teach” itself which fighter should win in a specific fight. Unfortunately, after days of research, and following advice from our project supervisor, we decided against a neural network and instead chose to implement a genetic algorithm. This was mainly due to the time constraints of this project, as a neural network is a difficult system and takes a lot of time to implement. As we only had approximately two months to do the majority of our project, which including the design and testing as well as the implementation of the system, we thought a genetic algorithm was more achievable. Also, genetic algorithms are still widely considered a form of machine learning, which is the form of Artificial Intelligence we were trying to implement from the beginning.

5.2 Implementing a Database

As part of our project, another goal of ours was to implement a database of fighters, containing all of their relevant statistics, which included the fighter’s reach, height, number of knockouts, number of submissions, etc. Initially we intended to locate this database on a server. However, we soon realised that the process of making the database server-based was a lot more complex than we had first anticipated. Given the limited time-constraint, we decided against this course of action. Instead, we decided to implement a database which would be stored locally on the device of the user. This database would

be downloaded at the same time as the application, and requires minimal space on the device. We felt that this was just as convenient as a server-based database on the basis that the information updates would only be required approximately once every month. These updates would be sent through application updates, as updating apps approximately every fortnight tends to be the norm nowadays.

5.3 Whether User Opinion Should Be Considered

As we built our sample space of Genetic Algorithms, many of them used our opinion on which fighter was more skilled in all aspects of MMA. This presented a problem, as asking which fighter is better is specific to a fight, and not to a lone fighter. This meant that we couldn’t store that information in our database. Therefore, as a result, we decided to build an algorithm that only used factual data and statistics, and did not consider the user’s opinion. Even though the predictions made by this algorithm were not as accurate as the opinion based predictions, we felt it would be the better algorithm to use, taking all matters into consideration.

We didn’t want to completely scrap the ability of the user to add their own opinion regarding fighters’ skillsets. Therefore, we decided to add this option, in addition to the option to not use opinion and just predict the winner based off statistics alone. We feel this functionality is more attractive to the user, and can help to increase their involvement in the app.

5.4 Fight Location Affecting Our Genetic Algorithm Results

During the design process of our Genetic Algorithm, we factored in the case of a fighter competing on home ground, and thought of it as a benefit to the fighter, thus increasing his chances of winning. In the test case fights we used to build these algorithms, we also included information regarding whether either fighter was fighting in their home country. In our algorithm, which just used factual statistics, and didn’t consider user opinion, a total of nineteen out of twenty-five fights were predicted correctly. These nineteen fights included information regarding whether a fighter was competing at home or not. However, when it came to the application development, and the information available to the algorithm was strictly based on a specific fighter instead of based on a specific fight, it was impossible to include whether either fighter was competing at home. We considered asking the user to input said information, but decided against this as many users tend to not know where the fight is taking place as it is usually considered irrelevant information to a fight fan. Thus, if we asked the user to input the information regarding the location of the bout, they would more than likely need to use the internet to find out, making our app much less convenient and centralised. This may also result in users not wanting to return to the app after leaving.

5.5 Testing the Genetic Algorithms

Upon designing the sample space of genetic algorithms, we wanted to

test them on fights to determine how accurate they were. Obviously, these had to be fights other than the ones used to build it, given that it may already have the correct genotype to predict these bouts correctly. Therefore, we decided to test it on fights where we already knew the outcome, i.e. past fights that the algorithm hadn’t seen. When we first started this process, we tried testing the fights individually on each algorithm, which proved to be a slow and painstaking process. To avoid this, we decided to place all the fight files into a folder and read the fights straight from the folder into the algorithm. This proved to be much more efficient and saved us a significant amount of time.

5.6 Screen Compatibility Issues

Throughout the design process of the Android Application, we noticed that creating one XML file for each activity resulted in the application becoming almost unusable on a vast number of devices. As usability is a big concern for us, we decided this problem needed to be solved. So, we decided to create 7 XML files for each activity, to accommodate all screen sizes. These include larger and smaller tablets, as well as larger and smaller smartphones, and phones of all pixel densities.

6. Installation Guide

To install this application, there are several requirements that the users’ devices must feature. These requirements are:

1. Device Type: The users’ device must be one of two options; a

smartphone or a tablet. While our application may download for devices such as smart watches and smart televisions, it is not designed to fit these screens and the layout may become unusable. The software is designed to fit all tablets and smartphones, regardless of screen size.

2. Android Operating System: The users' device must also be running a software version of the Android operating system, meaning it is not available for any devices running Apples' IOS operating system, or Microsofts' Windows operating system. This software cannot be downloaded and comes pre-installed in selected devices.

3. Minimum Android Version 4.1 (Jelly Bean): As well as the Android OS being a mandatory addition, the device must be running a minimum android version of 4.1 Jelly Bean. Any Android devices with an OS version less than this will not have the option to download this application from the Google Play store.

4. Sufficient Storage Space on Device: This application requires a minimum of 6MB to be downloaded to a device. If the user does not have enough space, they can make space by uninstalling other applications from their device.

5. Internet Connection: To download this application, the user must have Internet Access. This is necessary as the application is only available on Googles' Play Store, which requires an

internet connection to access. Internet connection is only required to download the application, and is not required once the app is installed.

6. Google Account: Upon entering the Google Play Store, the user will be prompted to login to their Google Account before being able to download any applications, including the MMA Fight Predictor. If the user does not have a Google Account, they must create one. This is free and easy to do and requires minimal information, such as an E-mail address, name, etc.

6.1 Using a Google Account to download MMA Fight Predictor

As previously stated, a Google Account is mandatory in order to download an application from the Google Play Store. If you already have an account, you can simply login. If you do not have an account, you must register for one. This is done by clicking the register button, and filling in all the relevant information, including E-mail address, Name, Age, etc.

Once you have a Google Account, you are now ready to download any android application. To download MMA Fight Predictor, follow these steps:

1. Click the search bar located at the top of the screen. Type MMA Fight Predictor and hit enter.

2. Search through the results for the application, and click it.

3. You will be presented with the MMA Fight Predictor Page. Here you can find all information regarding requirements, app size, etc. To install the app, just press the “Install” button.

4. The app will begin to download provided you have an internet connection. The speed of this download will vary depending on your internet package. Once the application has downloaded, it will automatically begin to install on your device.

5. Once the app is installed, it is ready to use. Locate the application in your app folder, by pressing the menu button at the bottom of the screen. Here, find the MMA Fight Predictor, and press it. The app has now launched.