Silicon Stadium

Project Team: Hello World or Bust

Table of Contents

**1. Project Definition**

* Why

We are proposing a machine learning model that will improve the way people enjoy fantasy football. Our model will enhance the experience of football and fantasy football with friends and family, by providing users with more accurate predictions of player performance. Our goal is to provide an entry-level tool that will allow anyone to get into fantasy football, without needing extensive knowledge of the game. At the same time, we understand that experts need more granularity in their predictions and we will provide them with a deeper level of analysis and insight. Fantasy football is the most popular of all the fantasy sports with 20% of US adults playing. We believe that our model will be a valuable addition to this market, providing users with a competitive edge and a more enjoyable experience.

* What

Our goal of building a neural network for fantasy football is to streamline the statistical analysis behind the game and make it more accessible to users. By using advanced machine learning techniques, we aim to provide users with more informed decisions when it comes to drafting and managing their fantasy teams. Additionally, our neural network will take into account team injury analysis to provide users with a more complete picture of a team or player's potential.

* How

This project utilizes machine learning model to predict the outcomes of fantasy football games, by using a combination of data sources and techniques. One key aspect of this project is to use a weather API to give up to date forecasts that will be used to predict the effects of weather on player or team's performance. Additionally, past player injury statistics will be used to predict future injuries. Our application will import and maintain data by storing it in a SQL database, perform data operations, and represent the analysis graphically using various visual representation methods like numbers and meters.

**2. Project Requirements**

* Functional
  + Import and maintain data:

The application will import data on current and past NFL players and use this information to perform statistical analysis and extrapolate trends, predictions, as well as simply representing current statistics in various scopes framed under individual players and entire teams. The data will be updated on a set interval, which will allow for analysis to remain up to date

* + Data storage:

The data will be stored within a mySQL database. The data will be separated into a master database, upon which branch databases will be created to satisfy analysis of varying scopes as well as to maintain the integrity of our data pool.

* + Perform data operations:

The data operations being conducted will query our databases and then utilize appropriate machine learning and/or data wrangling techniques to display trends, statistics, and make predictions from the appropriate data.

* + Retrieve analytical insights:

The individual player statistics we will be displaying include projected points(calculated by standard ESPN scoring guidlines), injury history/severity, scoring history, comparative ranking via scoring, performance at home/away, performance in regards to different weather extremes. The team wide statistics include team performance in regard to player position, team injury frequency, home/away performance, and inclement weather performance.

* + Represent analysis graphically:

The visual representation of this data will be created by the data returned from our statistical analysis. Statistics will be represented by numbers as well as meters to offer the user a more tangible result.

* Usability
  + User Input:

The user will be able to navigate several different options for what kind of analysis they are interested in. The choices will start by deciding if they want to look at an individual player or an entire team, from here they will be able to select a player or team depending on their selection and the desired player or team’s statistics will be returned and visually represented to the user.

* Website design:
  + Languages:

The front end website will be designed using a combination of html, scss/css, and javascript. The framework of the site will be VueJS.

* + Layout:

The layout of the website will be a start page asking the user to type in the player or team they are interested in viewing, which will then link to another page that will display relevant statistics in the scope of the user’s search.

* Front end/back end interface
  + API:

We will be using RapidAPI to pull data into our database from our listed data source(s).

* + Data Sources:

We will be pulling information from the following data sources:

* www.advancedsportsanalytics.com

* + Performance Metrics:

We are currently using an accuracy score of 70%

as our minimum performance requirement for predictions.

* System
  + Developer hardware:

The hardware being used to create this web application currently includes personal computers. These will be utilized to code and test the application and its features.

* + User hardware:

To access the web application the user will be required to have access to a computer. This is the only method by which the user can gain access to the site.

* + Development Software

The software being utilized by the developers include:

* Visual Studio Code
* Jupyter Notebooks
* Xampp
* phpMyAdmin
  + Database:

The database structure will be a main database consisting of all the information gathered from the RapidAPI. This data will then be separated into different views depending on the scope of the query and the required fields for analysis.

* Security
  + Concerns
    - SQL Injection attacks:

Utilizing databases comes with the inherent risk of malicious input. The user could feasibly try to enter malicious code via the initial search bar at the start page. The user could use an injection attack to alter our data and in doing so invalidate the accuracy and integrity of our analysis. Currently there will be no other places for this to occur.

* + - Account management:

When dealing with user accounts, security is imperative. There is a risk of someone getting access to an account that does not belong to them, and potentially changing critical account information.

* + Solutions
    - Prepared statements:

We will be using prepared statements in our queries to ensure that malicious syntax does not cause harm upon our data.

* + - Google login:

Google login has sufficient security measures included to safeguard account information.

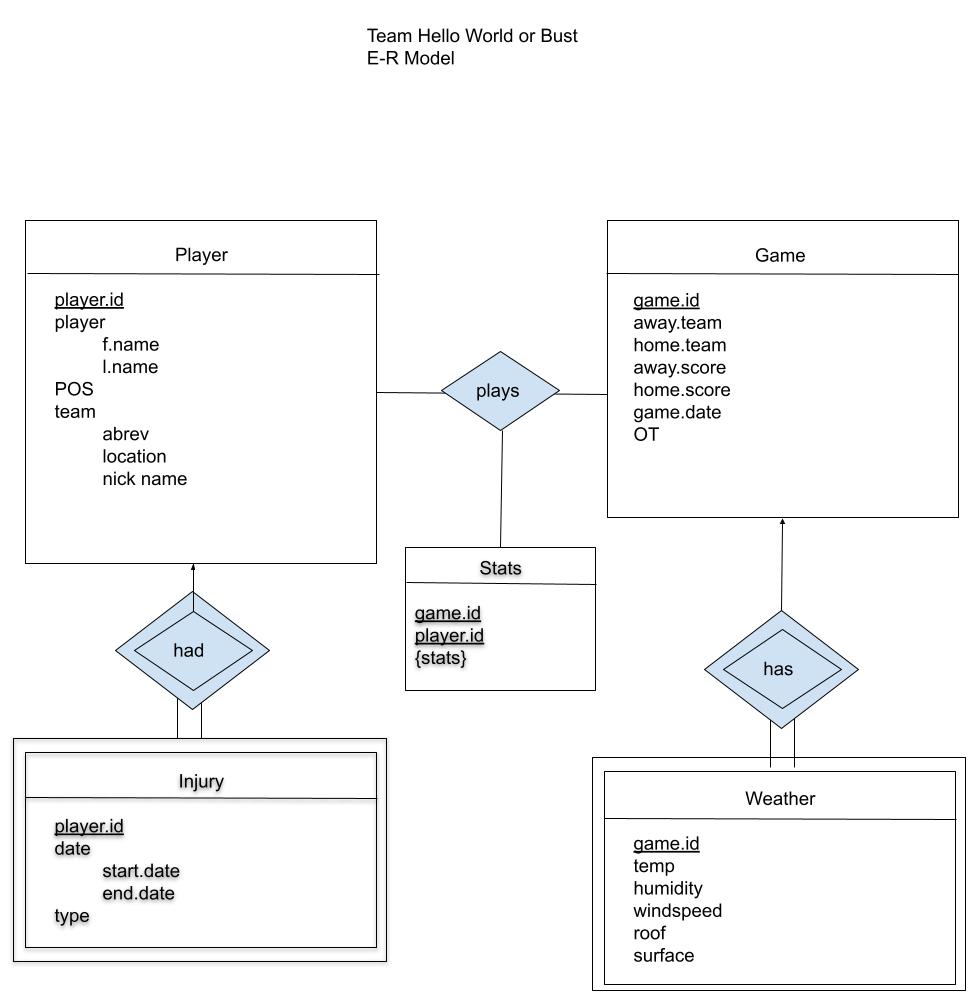
We will establish a secure and stable connection between the tool and the database using a web based authentication method for data transfer protocols to extract relevant data from the database, and store the results in a new database for later use.

**3. Project Specification**

* Focus / Domain / Area
  + The focus of our project is sports analysis. In particular, we are focusing on American football and fantasy football.
* Libraries / Frameworks / Development Environment
  + The libraries we will be using include sklearn, matplotlib, pandas, tensorflow keras and numpy. Front end development will use the VueJS framework. We will be using Visual Studio Code, Jupyter Notebooks, Xampp, and phpMyAdmin for our development environments.
* Platform (Mobile, Desktop, Gaming, Etc)
  + Our platform will be a website. We plan to host our prediction algorithm on a server that the website can access.
* Genre (Game, Application, etc)
  + Our project is a web application that focuses on sports analysis. In particular, we are focusing on NFL and fantasy football. The libraries we will be using to develop our algorithm include sklearn, matplotlib, pandas, tensorflow and numpy. We will be using Visual Studio Code, Jupyter Notebooks, Xampp, and phpMyAdmin for our development environments. We will host the prediction algorithm on a server that the website will have access to.

**4. System – Design Perspective** – *Group responsibility*

* Identify subsystems – design point of view
  + Illustrate with class, use-case, UML, sequence ..... diagrams
  + Design choices (Optional)
* Sub-System Communication (Diagram and Description)
  + Controls
  + I/O
  + DataFlow
* Entity Relationship Model:



* Overall operation - System Model
  + Simplified Sub-system to System interaction

**5. System – Analysis Perspective** – *Group responsibility*

* Identify subsystems – analysis point of view
* System (Tables and Description)
  + Data analysis
    - Data dictionary (Table - Name, Data Type, Description)
  + Process models
* Algorithm Analysis
  + Big - O analysis of overall System and Sub-Systems
    - Machine Learning
      * Predict using CNN: O(1)
        + Because our number of dimensions and number of nodes is constant after the model is created, the running time is O((Number of nodes in network) \* (running time of activation function) \* (Number of predictions)) = O(O(1) \* O(1) \* 1 ) = O(1)
      * Train CNN: O(n^2)
        + Note: Depending on how this is done it can still take a very long time.
        + The training time of a CNN is O((number of dimensions<Constant>) \*(Number of layers<Constant> \* (nodes per layer<Constant>) \* (training time per node<Constant>) \* (number of nodes <n>) \* (Number of epochs<n>)) = O(n^2). Note that this is still going to take a long time to train because we are fitting it with a large amount of data and there are still a large number of dimensions and layers.
    - Database
      * Querying the database: O(n)
        + Any simple query in a database will take constant time, or at worst linear.
      * Search and display teams: O(n)
        + We need to write to an output at least once per team. Assuming each write has constant time, this will make us take O(O(1) \* (number of teams)) = O(Number of teams) = O(n).
      * Insertion of data into the database: O(n).
        + Technically speaking, it is O(1) because we are using a constant as a limit for how long this can take, which we have increased to what we need. But more realistically it will be O(n).
      * Web Scraping
        + <TODO: Write this after the web-scraping code is completed.>
    - UI
      * Search and display teams: O(n)
        + We need to write to an output at least once per team. Assuming each write has constant time, this will make us take O(O(1) \* (number of teams)) = O(Number of teams) = O(n).
    - Connections
      * PHP
        + The cost of connecting through PHP is already accounted for in UI and Database.
      * Xampp through Apache
        + Connecting to Xmapp should take constant time

**6. Project Scrum Report -** *Group Responsibility*

* Product Backlog (Table / Diagram)
* Sprint Backlog (Table / Diagram)
* Burndown Chart

**7. Subsystems**

**7.1 Subsystem 1** – Name 1 - *Individual responsibility*

* Initial design and model
  + Illustrate with class, use-case, UML, sequence ..... diagrams
  + Design choices
* Data dictionary
* If refined (changed over the course of project)
  + Reason for refinement (Pro versus Con)
  + Changes from initial model
  + Refined model analysis
  + Refined design (Diagram and Description)
* Scrum Backlog (Product and Sprint - Link to Section 6)
* Coding
  + Approach (Functional, OOP)
  + Language
* User training
  + Training / User manual (needed for final report)
* Testing

**7.2 Subsystem 2** – Name 2 - *Individual responsibility*

* Initial design and model
  + Illustrate with class, use-case, UML, sequence ..... diagrams
  + Design choices
* Data dictionary
* If refined (changed over the course of project)
  + Reason for refinement (Pro versus Con)
  + Changes from initial model
  + Refined model analysis
  + Refined design (Diagram and Description)
* Scrum Backlog (Product and Sprint - Link to Section 6)
* Coding
  + Approach (Functional, OOP)
  + Language
* User training
  + Training / User manual (needed for final report)
* Testing

**7.3 Subsystem 3** – Name 3 - *Individual responsibility*

* Initial design and model
  + Illustrate with class, use-case, UML, sequence ..... diagrams
  + Design choices
* Data dictionary
* If refined (changed over the course of project)
  + Reason for refinement (Pro versus Con)
  + Changes from initial model
  + Refined model analysis
  + Refined design (Diagram and Description)
* Scrum Backlog (Product and Sprint - Link to Section 6)
* Coding
  + Approach (Functional, OOP)
  + Language
* User training
  + Training / User manual (needed for final report)
* Testing

**7.4 Subsystem 4** – Name 4 - *Individual responsibility*

* Initial design and model
  + Illustrate with class, use-case, UML, sequence ..... diagrams
  + Design choices
* Data dictionary
* If refined (changed over the course of project)
  + Reason for refinement (Pro versus Con)
  + Changes from initial model
  + Refined model analysis
  + Refined design (Diagram and Description)
* Scrum Backlog (Product and Sprint - Link to Section 6)
* Coding
  + Approach (Functional, OOP)
  + Language
* User training
  + Training / User manual (needed for final report)
* Testing

**7.5 Subsystem 5** – Name 5 - *Individual responsibility*

* Initial design and model
  + Illustrate with class, use-case, UML, sequence ..... diagrams
  + Design choices
* Data dictionary
* If refined (changed over the course of project)
  + Reason for refinement (Pro versus Con)
  + Changes from initial model
  + Refined model analysis
  + Refined design (Diagram and Description)
* Scrum Backlog (Product and Sprint - Link to Section 6)
* Coding
  + Approach (Functional, OOP)
  + Language
* User training
  + Training / User manual (needed for final report)
* Testing

**8. Complete System** – *Group responsibility*

* Final software/hardware product
* Source code and user manual – screenshots as needed - Technical report
  + Github Link
* Evaluation by client and instructor
* Team Member Descriptions

***This is just a guide, and use it to create/improve your report. Feel free to add sections. You are responsible for your own subsystem/s, not other members. You have to contribute to the team’s goals and objectives, and develop your subsystem/s, write your documents and slides.***