ENGTECH 4FD3 – Senior Engineering Project: Project Specification

Instructor: Dr. Seshasai Srinivasan

Group Members: Branislav Kukulj
Wenbo Liu
Emily Ramanna
Mingming Zhang

Date: Jan 30, 2022

Title: NBA Predictions Web Application

Team Members: Branislav Kukulj, Wenbo Liu, Emily Ramanna, Mingming Zhang

Introduction

This project will produce a user-friendly web application that will allow people to predict wins and losses for NBA (National Basketball Association) games. Points will be credited for correct predictions and users can compete in pools (groups) of their choosing to see who is the best at predicting. To encourage more casual fans to participate, and add an interesting dynamic of artificial intelligence versus super-fans, the system will provide an option to let a ML (machine learning) model predict on a user's behalf. The system will be extensible for other sports in the future through its design.

Objectives

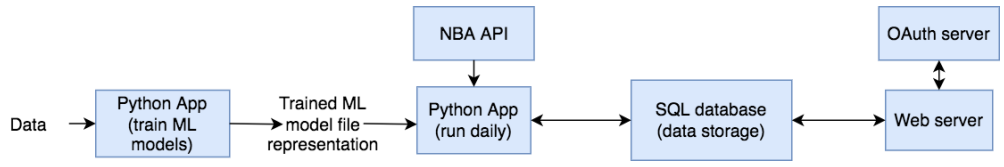
- Using different ML models such as Logistic regression, Random tree etc. to achieve the best accuracy and understanding the underlying differences.
- Demonstrate abilities of dataset cleaning, normalization and features engineering.
- Web programming a GUI application to achieve user-friendly experiences.
- Application of multiple languages and frameworks (see *Solution Methodology* section).
- Implementation of multiple architecture patterns to achieve functional and non-functional requirements (e.g. singleton for configuration info).

Solution Methodology

Software:

- Python applications to train machine learning models, make predictions, and interact with the database and NBA API (accessed through Python wrapper). Daily runs of the Python application that makes predictions will be handled by Microsoft Azure Logic Apps.
- SQL database engine for storage, MySQL hosted on Clever-Cloud for early development with later deployment to Microsoft Azure.
- Web server written in C# with the ASP.NET Core framework using Google OAuth for authorization and authentication, also deployed to Microsoft Azure. The web application will make use of JavaScript and Bootstrap to be responsive on desktop and mobile web browsers.

Figure 1: Software pieces (the NBA API and OAuth server exist and are not in scope for development)



Hardware: This project will make use of the public cloud for hardware infrastructure associated with hosting the web server, database, and Python application.

Steps:

- Gather data and train machine learning model(s) via a Python application, save models

- Set up a database that will store user account info, pool info, game info, and predictions
- Implement a Python app that will use the trained models to pull game info daily from the API, predict NBA games daily, save those predictions to the database, and update the results from the previous day's games. This will need to be set up to run once per day.
- Implement a web server and web GUI to allow users to perform the functional requirements detailed in the *Validation Strategy* section of this document.

Validation Strategy

Functional requirements:

- Users can create accounts and log in to application.
- Users can create a pool and invite others to join their pool.
- Users can predict the day's games, all user's prediction results will be stored in the database.
- Users can review past activity once logged in (i.e. see past predictions and their point total).
- Users can review the leaderboard in the pool.
- Users can choose to turn on auto-select to make predictions on their behalf.

Nonfunctional requirements:

Performance: Daily prediction (maximum 4 matchups or 8 teams) should be generated within 15 seconds (worst scenario performance metric). Validated via testing during runtime by recording and logging result writing to console or file.

Traceability: At least 90% of the application errors including database errors will create a log entry at runtime. Validated via testing and verifying the log files.

Unit testing of the web server and the Python classes can also be part of the overall validation strategy. UAT (user acceptance testing) scripts can be used to ensure that the user experience is functioning as expected. Cross-validation can be performed when training the machine learning models to try to ensure that the predictions are better than a coin flip.

Ethics and Sustainability Considerations

Inclusivity and accessibility: The website will be designed to be compatible with screen readers. This means using semantically-rich tags, including alt text for images, and laying out the pages in a way that would make sense to a reader.

Ethical concern: This app could be seen as a gateway to a gambling addiction. To mitigate this concern, this application will clearly disclaim that it is only for individuals 18+ and is not a gambling site.

Sustainability: All pieces of this application are intended to be hosted on the public cloud rather than on privately owned machines. This is better for the environment because service providers' equipment/infrastructure are generally more energy efficient. This also reduces the relatively large carbon footprint associated with commissioning physical devices.

Commercialization of the project

- How important is it to the end user? Rank: 1/10. It's a fun, recreational app.
- Other things that perform the same function? Rank: 4/10
    - Other similar web apps exist (e.g. runyourpool.com)
- Why choose ours?
    - None of the other apps encountered during research incorporate machine learning models directly into choices on behalf of users.
    - The prices for running a private pool can be quite high (e.g. 26-50 members costing $49.95 USD on runyourpool.com). Assuming we can get a similar licensing deal, costs to the users could be reduced by running ads on the website.
- The potential to grant access to more accurate ML models to VIP users who pay a premium.

Proposed Timeline Table

Table 1: Project activity schedule, Week 1 (starts: January 24, 2022) - week 10 (starts: March 28, 2022).

No.	Activity	Week #
1	Planning/Documentation of design for database (normalization/scripts to create tables), web server, and Python applications	1-2
2	Set up database in cloud	2
3	Retrieve the NBA dataset and prepare data needed for training (Python application)	3
4	Implement account creation and authentication (web server)	3
5	Train machine learning model(s) (Python application)	4
6	Implement joining pools, reading matchups from db, and saving predictions (web server)	4
7	Implement using trained model(s) to make predictions with stats pulled each day from API (Python application)	5
8	Implement auto-select predictions, dashboard and leaderboard (web server)	5
9	Prepare to present prototype (cleanup, integrate, etc.)	6-7
10	Implement saving daily game info, predictions, and actual results from previous day to database (Python application)	7-8
11	Clean up web GUI (web server)	7
12	Integrate interactions between website, Python application, database	8-9
13	Host finalized pieces of application in cloud, set up timed jobs	9-10
14	Update documentation and prepare for project submission	10