Systems and Toolchains Course Project – Option 1

Deadline: November 14th, 11:59pm ET

Accept this assignment by accessing GitHub classroom via the following URL:
https://classroom.github.com/a/VuODydzp

In this project, you will use FIFA Dataset available on Kaggle
https://www.kaggle.com/stefanoleone992/fifa-22-complete-player-dataset/

This dataset contains Soccer player statistics for 2015-2022. Male player data start in 2015 while Female player statistics start in 2016. You will need to collect player stats for both Male and Female players across all years to conduct the following tasks.

General Expectations

- Follow coding best practices with well-documented code.
- The course project will include a lot of self-learning that is needed to complete the project. Students shouldn't expect any project support during office hours or on Piazza.
- Add your dataset under a new **data folder** on GitHub repository
- You may choose 1 peer to work with on this project.
- If you choose to work with peer, **write the name of your peer in the Canvas submission**. If you fail to do so, your peer will not get the grade.
- We will not handle cases where students forget to submit the name of their peers.
- In this dataset, you will see incomplete data for Career Mode and Female Players. For this reason, don't include them in your analyses. The files of interest are named "players_15.csv", "players_16.csv", etc.

Task-I: Build and populate necessary tables (30% of course project grade)

- Ingest the data from all years (Male: 2015-2022 and Female: 2016-2022) into one Postgres Database table.
  - Conduct any column name changes to ensure data from various years are properly aligned in the correct columns in your DB table.
- Add a new column for the year. Also, ensure every record can be uniquely identified in the database table.
- Your tables should be created in schema with the name "fifa".
- In your ReadMe.md, add a description of the features in the dataset.
- In your ReadMe.md file, comment on the benefit of using PostgreSQL DB table compared to a NoSQL Database in this case.

Task-II: Conduct analytics on your dataset (20% of course project grade)

Develop Python functions that run Spark to answer the following questions (given that x, y and z) are user-entered parameters. Core analysis should be conducted via Spark and data should be ingested from Postgres database.

- In Year **X**, what were the **Y** clubs that had the highest number of players with contracts ending in year **Z** (or after)?
    - X is a year between (2015 and 2022, inclusively).
    - Y is a positive integer.
    - Z is a year that can hold the value of 2023 or a year after it.
- In sports, maturity and energy of teams depend on the average age of team players (among other factors). Therefore, it's important to have a function that can find clubs with such features.
    - List the **X** clubs with the highest (or lowest) average player age for a given year **Y**.
        - X represents a positive integer, but you should handle a scenario if X is not positive value.
        - Y represents a year between 2015 and 2022 inclusively.
        - Provide the user with the ability to choose if they want the highest average age or the lowest average age.
        - Make sure to handle this scenario as well: if the user requests 5 clubs with highest averages but there are 3 clubs that share the same count at rank number 5, please include all of them in your output
- What is the most popular nationality in the dataset for each year? (i.e. display the most frequent nation for 2015, 2016, etc.).

Task- III Machine Learning Modeling (30% of course project grade)

- Build a machine learning model that can predict the overall value for each player based on their skillsets.
    - Use proper feature engineering principles (including data cleaning and data engineering)
    - Build two versions: one in Spark and the other one in PyTorch or Tensorflow.
    - For each version, choose two different classifiers/regressors. You can use the same two choices for Spark and PyTorch/Tensorflow, and neural networks of substantial different structures (deep vs shallow, MLP vs CNN) count as two different classifiers/regressors. For each classifier/regressor, identify a few tunable parameters for your model and tune the parameters (using proper metric(s)). Then, run the best model (after tuning) on the test data set and record the test accuracy.

  - In your ReadMe file, explain why you chose the classifiers/regressors and provide comments on the impact of the tunable parameters on the accuracy. Also, compare the selected models.

Task- IV Deploy your code to the Cloud: (10% of the course grade)

- Run a version of your code for the three tasks above on the cloud.
- In this version, you may skip the creation of the Database on the cloud (i.e. on the cloud version, you don't need to write data to table for simplicity). You may ingest the data from CSVs directly.
- If you run the PostgreSQL on the cloud: you will receive **10% extra-credit**.

Course Project Checkpoint is planned for **October 10th** and will constitute **10% of the course project grade**. In your course project checkpoint, submit **Task I** and **Task II**.

Submission Guidelines:

- You MUST use the GitHub classroom URL to create your repository. Post your GitHub repository's URL created via GitHub classroom to Canvas. Use the starter code that is provided above as the starter for your code.

- Your GitHub repository should have a ReadMe.md file that lists the "exact" steps on how to run your code and the input configurations associated with it. I will follow the steps in your ReadMe file and if I can't get it running on my machine, I will deduct considerable number of points from your project grade.

- **You should record a video demonstrating two elements:**

  1. Code Walkthrough while you are explaining your code changes.

  2. Demoing the running application while you are navigating through EVERY functionality that is working in your application. I will use this video to help assessing your grade. You may lose points for the functionalities that are not demonstrated in the demo.

- Your video size may be large to be uploaded to GitHub. You may use Box to upload the video and add the URL to your ReadMe.md file in your GitHub repository.

  1. Make sure that your video is publicly shared. Private videos won't be visible to the instructor and TAs and therefore, your project grade will be impacted

Grading Notes:

- Unlike HW-assignments that has 20% late penalty, late project submissions on Canvas or GitHub **will receive 0 points** (won't be graded)
- **Not submitting the GitHub video (for both code walkthrough and functionality demo): you will get up to 80% of the maximum grade.**
- Not providing clear details in the ReadMe file on how to run the application (or any variables that need to be updated/replaced): **you will get up to 90% of the maximum grade.**

Refer to Course Syllabus for planned course project checkpoints.