The Challenge

Sportsball wagering provides a great source of entertainment for the people of Bettorvania, a country passionate about sports.

The Major Sportsball League consists of 16 teams, spread across 12 provinces. Currently, our wagering services cover 4 provinces: Regensland, Alterburg, Vistatown, and Boroughsville.

Our latest addition to the team is Dr. Research, a new data scientist with a strong academic background. Dr. Research has developed a model capable of predicting the amount of money wagered, also known as "handle," on sportsball matches. The model's code is available in the notebook `notebooks/handle_forecast_basic_solution.ipynb`. This notebook includes both SQL queries to fetch features from the provided database and the modeling code.

It's important to note that the enterprise database only updates on a daily basis. Additionally, the enterprise database has known quality issues when it comes to accurately calculating total user numbers before an event has occurred. Luckily, the data services team has created an API containing real-time data on sportsball matches directly from the production databases for the upcoming week. (For the purposes of this project, the "upcoming" week refers to week 11 of the 2020 season.) The API's base endpoint is `https://data-dev-api.penngineering.io/sportsball-events`, and its documentation is available at `https://data-dev-api.penngineering.io/sportsball-events/docs`

The finance team has requested the ability to retrieve the most up-to-date forecast on demand.

Your mission, should you choose to accept it, is to create a deployment strategy for this predictive model. The finance team has only provided loose requirements, giving you creative freedom. While this problem is simplified for time's sake, it mirrors the kind of work we undertake at Penn Interactive/TheScore Bet.

Please allocate no more than 5 hours to this challenge. We understand that this task could potentially take longer, but we're more interested in understanding your thought process than a fully polished deployment. If you find yourself approaching the time limit, focus on conveying your thoughts and insights in the write-up section.

Requirements

To complete this challenge, ensure that you have Docker and docker-compose installed on your machine.

If you don't have these tools, we recommend using Docker Desktop:

- Mac: Docker Desktop for Mac
- Windows: Docker Desktop for Windows
- Linux: Docker Desktop for Linux

Setup

We've provided a `docker-compose.yml` file that sets up everything you need to tackle this challenge. To get your environment up and running, navigate to the directory where you've unpacked these files using your Command Prompt or Terminal, and execute `docker-compose up -d`. This might take a little while, especially if it's your first time running it.

The Database

Once `docker-compose up -d` completes, you'll gain access to a Postgres Database and PGAdmin, a tool for interacting with the data. To access PGAdmin, open your web browser and go to `localhost:5050`. From there, you'll need to configure a connection to the server. To do this, select "Add New Server" from the dashboard.

Connection Details:

- Host: `local_pg`
- Port: `5432`
- User: `postgres`
- Password: `postgres`

The database comprises three tables:

`events` (`SELECT * FROM events`) - Sportsball Event Data

Column Name	Data Type	Description
event_id (primary_key)	VARCHAR	Unique ID for the event
home_team	VARCHAR	Home team for the event
away_team	VARCHAR	Away team for the event
home_team_location	VARCHAR	Base city of the home team
away_team_location	VARCHAR	Base city of the away team
day_of_week	VARCHAR	Day of the week ('Sunday', etc.)

Column Name	Data Type	Description
time_slot	VARCHAR	Categorical time/day grouping
game_day	TIMESTAMP	Game date (EST)
game_start_time	TIMESTAMP	Event start time (EST)
week_start	TIMESTAMP	Start time of the week (EST)
week_of_season	FLOAT	Week number of the season

`users` (`SELECT * FROM users`) - User Demographic Data

Column Name	Data Type	Description
user_number (primary_key)	VARCHAR	Unique user identifier
age	FLOAT	User's age in years
registration_timestamp	TIMESTAMP	Registration timestamp for the sportsbook (EST)
location	VARCHAR	User's place of residence

`wagers` (`SELECT * FROM wagers`) - Individual Wager Data

Column Name	Data Type	Description	Additional Resources
bet_id (primary_key)	INTEGER	Unique bet identifier	
user_number	VARCHAR	Unique user identifier	
event_id	VARCHAR	Event associated with the bet	
bet_offer_type_name	VARCHAR	Description of the bet type	Spread, Over/Under, Outright
wager_amount	FLOAT	Wager amount in USD	
outcome_decimal_odds	FLOAT	Bet odds in decimal format	Decimal Odds
outcome_american_odds	FLOAT	Bet odds in American format	American Odds
bet_placed_time	TIMESTAMP	Bet placement time (EST)	
bet_status	VARCHAR	Bet settlement status	
bet_result	VARCHAR	Bet outcome	
payout	FLOAT	Payout amount in USD if the bet wins	

Jupyter Server

The compose file also sets up a Jupyter Server instance for you to use. You can access it by going to `localhost:8888?token=sports` in your web browser.

Challenge Structure

Code

For this challenge, feel free to modify any existing files or add new ones as you see fit. This includes the provided Jupyter notebook. The only files that must remain untouched are the CSV files in the `data` directory.

Unlike the Data Scientist version of this challenge, the MLE version is more open-ended. Your interpretation of the problem and how you address the loose requirements will be a significant part of the evaluation.

Technical Write-up

We genuinely mean it when we say not to spend too much time on this challenge. Beyond technical skills, we're interested in other soft skills, including time management.

Provide us with a write-up in whichever format you prefer, outlining your solution concisely. Additionally, take this chance to suggest extra features or improvements you would implement with more time.

(Note: While mentioning "monitoring model outputs" is valuable, please elaborate on how you would do this.)

Please save your write-up as `challenge_submission.<file_extension>` and place it in the project root.

Submission

Once you've completed the challenge, repackage the project as `<your_name>_interview_challenge.zip` and send it back to the recruiter.