-
-
- Search hackathons
- - Log in
  - Sign up
-

-
- Hackathons
-
- Projects
-
- Host a hackathon
-

-
- Hackathons

Olympic Oracle - Predictor for Athletes

A powerful API that predicts whether a given olympic athlete will win their next game in their chosen sport, powered under the hood by advanced statistical methods and machine learning techniques.

Inspiration

There are many products that predict the outcomes of sporting events based on historical factors or betting data. However, fewer put the central focus on the individual athletes playing the sport. We wanted to make an API that predicts the probability of an athlete winning a medal or not in an Olympic event, based primarily on the physical characteristics of the athlete and how well suited it made them for their particular sport. We also wanted to give importance to public opinion and what people currently think of the athlete's chances - so we built a twitter bot to gather data from and provide results to curious sports enthusiasts, to unify the sport with the spectator.

What it does

The API receives some input in the form of an athlete's data - their weight, height, age, nationality and sex, and the user chooses their appropriate sport (automatically selected to the right sex), and the API will return the probability of the athlete winning a medal - the most likely medal they will win (if any) and the corresponding probability of them winning this medal. It also runs polls on twitter, asking users to give their opinion of if they think the athlete is likely to win this medal or not. Then, this data will be fed back into the original prediction, and a carefully weighted average of the two predictions (one based on machine learning based historical analysis, and the other on current opinions) will be made. The weighting is such that the more people answer the poll with a higher skew towards one answer, the more weight it will take relative to the historical analysis. We wanted to have this feature as we realise that our few physiological traits will not amount to a complete predictor of the athlete's chances and we also recognise that people's opinions on sporting success in olympics events are usually based on reasonable grounds. Furthermore, we saw this as an excellent opportunity to unite the statistics with the community, to bring the past data and trends and present beliefs together.

How we built it

We went over an abstract blueprint for our API - what it should do behind the scenes (the machine learning and data analysis), and the front-end - how it should present its results and interact with the users, and how it would communicate between the two. We cleaned the data set, and calculated standard results about different sports, medals awarded, and calculated correlations with height, weight and age using Alteryx. Then we fed this data into a feed-forward neural network that would then predict the probability of a given athlete (with physical data) winning certain medals based on this large amount of data. After this, we used a Bayesian model to deal with the nationality parameter. The prior probability was simply the probability of winning a medal of some athlete with some parameters competing in a sport, but the posterior was the probability of this same athlete with the same parameters in the same sport given that we knew what national team they were representing. We then developed a twitter bot with the ability to run polls about certain athletes, the results of which we would collect and then judiciously weigh against the rigorous analysis we've performed, as a measure of how different real people would feel compared to what the data and numbers say.

On the front end side, the API was written in Node.js. The server.js file retrieves functions from other files to form the get and post methods that retrieve and writes data to and from the server. Loading the site loads links to the other files of the site, each of which has a corresponding file that is reading from when the link is clicked. Another file contains the method that is used to read from graphs imported via python and return a list of bytes. The front end code required the modules express, multer and body-parser to be installed as dependencies and nodemon to be installed as a development dependency. A 'start' script was added to the package.json file so that the server could be initiated with npm start. As an opinion collection method and mock demonstration, we use a twitter bot. Given more time, we would implement something more robust and targeting different audiences.

Challenges we ran into

We originally tried to use a LSTM neural network to model and solve our problem, but this turned out to be an unsuitable choice. After that, we pivoted to a feed-forward neural network. We faced many challenges in getting the twitter bot to actually communicate with our API. The data cleaning and training process also took a long time, and while we originally planned to include every sport in the olympics, we ultimately had to limit ourselves to just 5 sports. Bayes' theorem in its default form as well as naive linear interpolation did not work well as some of our data was greatly biased (i.e. over 60% of the overall gold medals in swimming were achieved by American athletes), so we had to design some sort of weighting system to account for this imbalance inherent in past olympic games. It was also hard for all members to communicate with each other perfectly when some were particularly focused on just the front end or the back end, and we had varying skill sets so we had to have certain members that acted as intermediaries between the two.

Accomplishments that we're proud of

The weighting system we designed to account for the bias in different nations winning disproportionately many medals in various games (i.e. America in Swimming, France in Cycling etc) works very well. It gives very reasonable increases to an athlete's chances if they're from a nation that is historically dominant in that sport while not ignoring the possibility for upsets and other nations to win medals. We're also happy with how well connected the back and front ends of our system is, as well as how we designed novel forms of interpolation appropriate for the data analysis we were performing. Running this API on some samples we generated including unrealistic and edge cases, we were very happy to see how output was kept reasonable by our AI for the most part, although there is still plenty of room for further refinement.

What we learned

We learnt about the appropriateness of various statistical tests for different types of data, the best way to clean and process large data sets, and how to connect numerical results to a more meaningful presentation. We learnt about the workflow of how a real-life product should be developed - from conception to different stages, in which various working parts were constructed, each with interrelated dependencies. We had to learn to organise our time and schedule in an efficient manner such that there wasn't any moment in which multiple subparts of the project were waiting on just one thing to be concluded. We tried to work in parallel as much as possible with the front end and back end and then finally unify the two towards the end of the hackathon. We also learnt that it is very important to keep tabs on what stage of processing certain subsets of data were in.

What's next for Olympic Oracle - Predictor for Athletes

The natural next step is to extend our API for all games of the olympics, not just 5. We also want to consider other factors like athletes' individual track records, if they will be playing in their home country or somewhere far away and rigorously analyse how this affects the probability of them winning medals in that particular sport.

More ambitiously, we hope to extend this to other non-olympic sports games and tournaments, and perhaps even to non-sports events with a lot of uncertainty and historical data to draw from like elections.

Built With

- alteryx
- c#
- keras
- node.js
- python
- tensorflow
- tweepy

Try it out

- GitHub Repo
- twitter.com

Submitted to

- DurHack2022
    - Winner Alteryx Challenge

Created by

- I worked on the back-end. I had to look at designing and implementing a lot of unfamiliar statistical and machine learning algorithms, which was challenging and enjoyable.

Ayu Kharel

- I worked on implementing the machine learning and the backend.

Christopher Teo

- I worked on the front-end, creating a twitter bot using its API to run through an account and interact with others. It was frustrating at times but ultimately quite interesting and satisfying.

[Anivarth Gopikrishnan](https://devpost.com/software/olympic-oracle)

-

[Joseph Bannon]

-

[johnny-stevie]

[Like 4]
4 people like this:

-

-
-
-

Share this project:

Updates

Ayu Kharel started this project — 50 days ago

Leave feedback in the comments!

- Christopher Teo · 50 days ago

  Truly innovational

- Alexandre Delaitre · 50 days ago

  solves a big problem

- Alexandre Delaitre · 50 days ago

  should be proud!

-

  Ryan Arrowsmith · 50 days ago

  BIG STONKS

Log in or **sign up for Devpost** to join the conversation.

Devpost

- About
- Careers
- Contact
- Help

Hackathons

- Browse hackathons
- Explore projects
- Host a hackathon
- Hackathon guides

Portfolio

- Your projects
- Your hackathons
- Settings

Connect

- Twitter
- Facebook
- YouTube

- Community guidelines
- Security
- CA notice
- Privacy policy
- Terms of service