**Awesome-Hyperopt**

Web application for scheduling, monitoring and analyzing cloud-based hyperparameter optimization runs

**Motivation**

At DeepCure, we train and evaluate hundreds of machine learning models for different projects every day. We have developed algorithms that use cloud-based, distributed computing to perform efficient hyper-parameter optimization, i.e. selecting the best parameters for each model. Currently the algorithms are run by command line scripts, and the results are analyzed by manual inspection of a database.

**Goal**

Have a web-based platform that allows the machine learning scientists at DeepCure to schedule, [monitor and analyze the results of a hyper-parameter optimization run]. We expect the platform to minimize the overhead in problem solving and increase the speed of our hypothesis testing cycle.

**Milestones**

**Environment Setup (Git, Tutorial, and IDE)**

Follow the single page flask/Vue.js tutorial given by Thras while trying to integrate the newer versions of vue, axios, etc... Possibly using other CSS frameworks we find that will fit our needs(bulma.) Goal is to create a skeleton application, Get to know the Backend and HTTP communication, and finally commit to a branch on GitLab using the provided skeleton.

**Design study of UX**

Before committing significant development effort to the frontend, research, prototype, and test possible designs and user experiences. The first step is to understand the current workflow followed by DeepCure machine learning scientists when starting an experiment. This requires watching a current employee go through the steps of their workflow and asking follow up questions to understand the different decisions they are making. Once the workflow options and objectives are clear, ideate possible user experiences. Rapidly draw several options and consider using team design exercises like Crazy-8s. Take the option that best accomplishes the objectives and create a high fidelity prototype. To validate our assumptions and ensure we are creating an intuitive user experience, test the prototype on 3 to 5 people. Incorporate their feedback and create a more polished design vision that we can start coding.

**Have a High Level Plan for Application**

To ensure we understand how all parts of the application communicate and work, create a high level plan for the application. For this, we will assume the DeepCure APIs are working as intended. At the end, we should have an overall architecture design for the application structure and understand how all parts communicate. This includes defining how the front end needs to request information from the backend for displaying on the UI, defining the format in which the backend will supply information for displaying, defining how the frontend will format and send experiments to the backend once the user has selected option and decided to start a run, and understanding how the backend needs to format requests to the scheduler.

**Have Unit Testing**

We will begin by obtaining a good understanding of what the possible and correct inputs into our program should be. We will also determine what the appropriate output should be, up to (but not including) how the output is displayed to the user. We will then do some research on effective methods of implementing unit testing, including best practices and standard conventions. Once we have a good understanding of where to begin, we will design the different tests that we would like to implement. This is likely a good opportunity to create UML diagrams (or something similar). After we are satisfied with our design, we will begin writing the code for our testing classes. Finally, if we can figure out how, we will
run our tests to make sure they work correctly. Because we probably won’t have any code to test our tests, it may suffice to prove the correctness of our testing algorithms.

**Minimum Viable Frontend Alpha Version**
The goal is to create a minimum working version of the frontend with minimal backend integration. Start by researching design tools and libraries. Come up with a rough implementation plan. Next, create a simple version of the user interface with dummy data. At that point, there is no integration with the backend. Next, start to communicate with the backend. Sending requests to the backend and get a JSON object back. Use that to display real data on the user interface. Finally, create a fully-working, minimally-interactive display. The user will be able to select 1-2 options from each stage, and the application will dynamically update. When the user submits an experience, a properly formatted request will be sent to the backend.

**Minimal Viable Backend Alpha Version**
The order of tasks in this milestone are still a bit nebulous, but the tasks to completion are as follows. We will prepare a barebones user interface for manual testing of backend features. This UI will allow minimal options that allow the backend to communicate with DeepCure’s job scheduler. We will also create an object-relational mapping to provide a high level understanding of how the backend will communicate with the different aspects of this project. Finally, we will link this backend alpha version with the frontend alpha version. This milestone will be complete once functioning linkage between the frontend and backend has been established.

**Integration with DeepCure Hyperopt Scheduler**
Once we have an interface and a way to talk to our backend we can begin to talk to the Deepcure Scheduler. Begin to understand the DeepCure Scheduler and how we Query and receive information from it. Create methods for sending Jobs to the Scheduler. Depending on project progression, consider creating methods to read job results and display results to the frontend. However, the primary focus is to schedule jobs, not to monitor or analyze.

**Create Final Product**
This will be the final coding-heavy part of our project. We will review the frontend and backend linkage to ensure both speed and security. We will also put finishing touches on the user interface and any intermediate (technical/developer) interfaces. Time permitting, we would also like to conduct live user testing to obtain feedback on usability. We say time permitting because at this stage it may be too late to make significant changes, but minor improvements could be implemented. This milestone will be complete once a fully functional version of our application is thoroughly tested and designated final.

**Delivery of Final Product**
Delivery of our final product will begin with the refinement of the documentation that we will have been building as we worked on the project. We will create a system schematic that provides a technical overview of our product. We will turn usage notes and code comments into proper documentation and do the same for technical documentation that will be geared towards developers instead of users. Finally, we will package our application for delivery and presentation.

**Deliverables**
- Web app with an intuitive user interface that allows user to configure and schedule an experiment
- Documentation describing project implementation and architecture
- Unit tests to validate frontend and backend functionality

**Out of Scope**
- Analyzing experiment results
- Monitoring experiment progress in detail
- Educational material that guides user through best experiment configuration options
- Making product generalizable to systems besides DeepCure. Configuration System
- User authentication and identity management
- Saving results from multiple past experiments for analysis within the application