Replicating Data Science Models in the Cloud

Yan Jiang Dong (dongyj@bu.edu)
Christine Duong (duongch@bu.edu)
Wei Jiang (jiangwei@bu.edu)
Caroline Jones (ctjones@bu.edu)
Mentor: Sri Krishnamurthy (s.krishnamurthy@northeastern.edu)

1. Vision and Goals Of The Project:

When you work on data science projects, typically the data is given and modelers are expected to build the best models to address the problem. While final predictions can be evaluated and ranked through an evaluation criterion, the actual methods, tools, packages, and libraries can be varied. This makes it difficult for both instructors and data science researchers to obtain all the needed dependencies to reproduce experiments. The problem we seek to solve is the difficulty of evaluating methodology and standardizing the environments used when developing data science models and replicating experiments.

Our project will be a system that allows instructors to generate environments that are replicable in order to both easily be given to the students and to then evaluate the students' work. Our high-level goals include:

- Structuring an environment generator that an administrator can prime with certain seeds (such as software packages like Python, R, etc, and datasets) in order to create a standardized image that all students can use as a starting point.
  - The system should have an administrator portal to add tools for a baseline.
  - Once the administrator selects the tools and data sets to be used in the environment, the system will automatically generate a Docker image.
- Allowing the environments to be replicable for future use primarily using Docker.
  - Administrators can distribute the primed Docker environments through whatever service they choose (such as Docker Hub or GitHub), and students can download the primed image.
  - Once the environment is downloaded, students are able to change the environments in order to adjust to their own models.

- An administrator has the ability to track the progress and changes made in the environment.
    - A log file will be created as the users work in the image. This can be viewed but not edited by the user. It can be exported in order to be submitted to the administrator.
- Enabling the use of a multitude of cloud vendors.
    - These cloud vendors include Amazon Web Services, Microsoft Azure, and the Massachusetts Open Cloud.
    - Users will be able to specify the needs of their project (storage, computing, etc.), enter their vendor-specific cloud credentials, and an instance will be created for them.

Users/Personas Of The Project

This project will be used by instructors teaching data science classes, data science students, and data science researchers. It can also be used for other applications, such as for users who wish to demo a live project to a potential employer, as opposed to showing the employer something static such as charts or a code repository. End-users do not require any prior knowledge of cloud technology.

It does not target:

- Cloud engineers
- General cloud users who need services beyond the scope of data science models or wish to configure their own cloud settings
- High performance computing users

2. Scope and Features Of The Project:

Administrator Portal

- Presents a simple interface for the administrator to generate an environment.
    - Allows for administrator to include tools for baseline environment.
        - Checkboxes to include languages and packages that are required.
    - Allows for administrator to upload or assign datasets used for the task.
    - After all configurations are set and confirmed, a Docker based environment will be generated and can be distributed to students.

Environment/Container

- Simple interface to allow users to run their data science models.
    - Allow users to choose preferred Cloud vendor and insert credentials.
        - Supports protocols for different clouds such as MOC, Azure, and AWS.
    - Allows users to run models against a given data set and returns results to user.
    - Allows choosing from a set of environments
- Includes all of the packages and datasets that the administrator chooses.
    - Students have freedom to install additional packages; if the environment is replicated it will contain these additionally installed packages and libraries.
- Tracks activity of users.
    - This tracked activity should be accessible for view.
        - Information should include the change and the date of the change.
- Ability to replicate and be distributed to others.

3. **Solution Concept**

Global Architectural Structure Of the Project and a Walkthrough

Below is a description of the system components that are building blocks of the architectural design:

- Administrator Portal: A web application for administrator to construct environments. This should contain checkboxes for packages and a method to upload data sets to the environment.
- Environment Scripts: Generated based on administrator preferences. These are run automatically to create the Docker image.
- Docker: The container for the data science environment is deployed to Docker.
- Downloadable Image: A replicable Docker image that contains all of the parameters, packages, libraries, and datasets assigned by the instructor (admin). Users should be able to update and interact with the image to run their models.

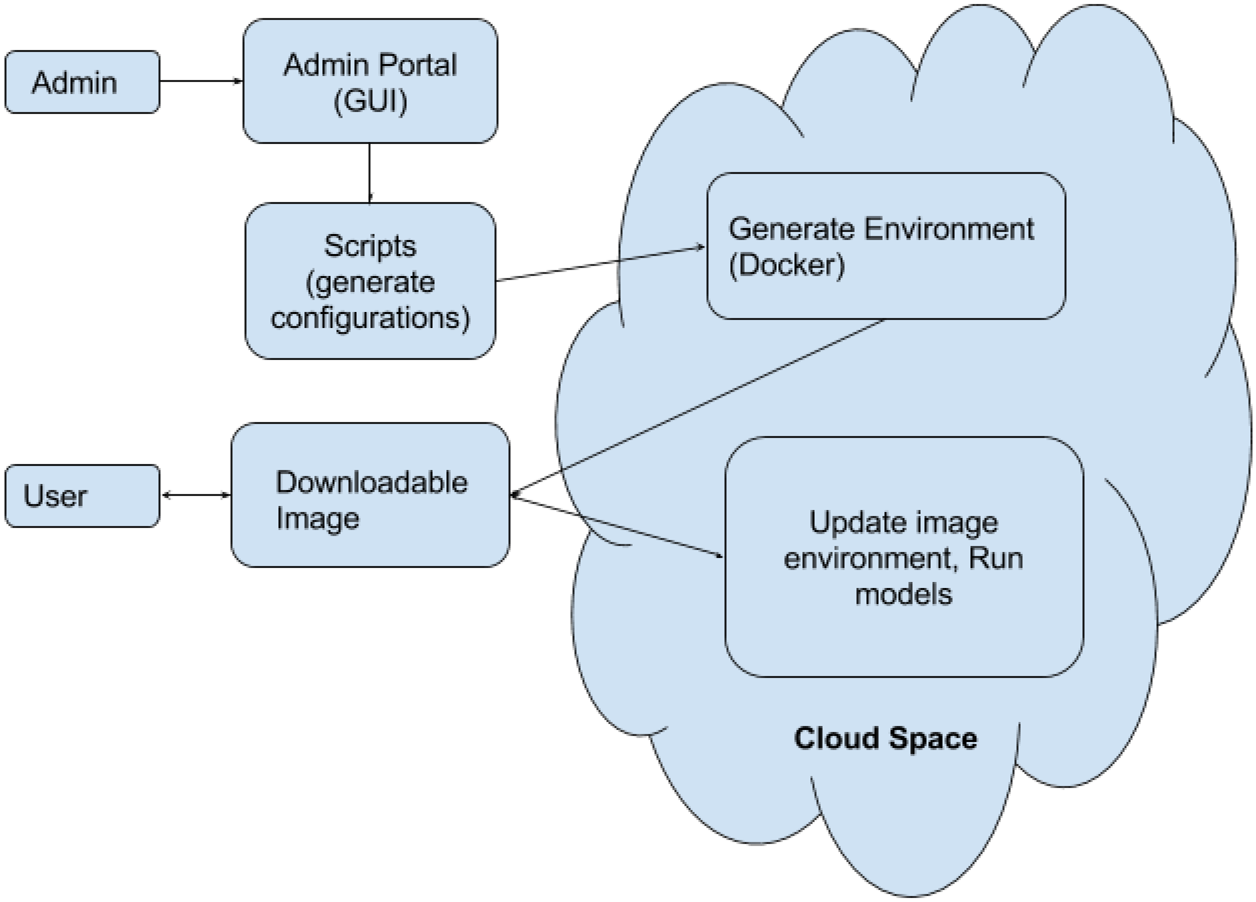
Admin
Admin Portal (GUI)
Scripts (generate configurations)
Generate Environment (Docker)
User
Downloadable Image
Update image environment, Run models
Cloud Space

Figure 1: System Block Diagram

Figure 1 presents a system block diagram of our system. The diagram is separated into two spaces: the desktop space and the cloud space. The desktop space contains processes run locally or on a web application on a user or administrator’s machine. The cloud space contains processes that can be run on the cloud; the cloud provider is specified by the administrator or user.

Design Implications and Discussion

Key design decisions and the motivation behind them.

- Administrator Portal: A web application was chosen to make this more accessible compared to a desktop application. This will be developed using Python 2.7 and Django. The web application will be hosted on the Massachusetts Open Cloud (MOC). This will provide scalability and the ability to upload and store datasets during the image creation process.
- Environment Scripts: The environment scripts are generated through the options selected in the administrator portal. This process is

automated so that the administrator does not need any knowledge of docker in order to create a standardized baseline image for students. The scripts will be written in Python 2.7.

- Docker: Docker allows an image to be created with packages. Users can then grab these images and alter them in containers. A user should be able to upload new data sets, download new packages and libraries, and run their data science models by entering in their cloud credentials. When the user wishes to use the cloud to perform large scale models, we will generate the necessary instances based on their cloud provider of choice. Additionally provides the ability to be updated and retain new packages and datasets that have been added. If replicated, the image should keep all of these parameters. It should also keep a trail of changes and cloud interaction which are logged, and the image can be downloaded as a new image for reusability.
- Downloadable Image: In addition to the images created by administrators, the ability for the users to save and share their environments (packages, tools, datasets, and libraries) is a key pillar of the project. When the user is finished with the environment, a new image is created to be shared with others. Users should only need to supply their cloud credentials to run large scale data science models.

4. Acceptance criteria

The minimum acceptance criteria is a usable container that can run data science models on the cloud, and other users can replicate and alter it. Stretch goals are:

- Allow administrators to save customization options from previously generated environments via user accounts on the web application.
- Version control: Give users and administrators the ability to rollback changes on the timeline.
- A ranking systems for students to compare their models against each other.

5. Release Planning:

Detailed user stories and plans are on the Trello board:
https://trello.com/b/stUVaiSr/replicating-data-science-models-in-the-cloud

Demo 1 Feb 23:
User stories: Admin Portal to create docker image based on package selection and dataset.

Admin Portal:

(Web Application) The admin portal should be developed as an intuitive web application.

(Package Selection) On the admin portal, the admin selects the libraries, packages, and data sets they want to include in their Docker image.

(Docker Base Image Creation) After submitting preferences, a script creates a base Ubuntu Docker image with selected preferences and data sets. This image is then presented to the admin for download.

Demo 2 Mar 16:
User stories: Image Update
Image Update:

(Package selection): User selects packages they want to install and scripts run to update the Docker image to include selected packages.

(File Upload): Users can place their data science models and datasets into the image.

(Image Replication): When the image is replicated it retains updated information of installed packages and files.

Demo 3 Mar 30:
User stories: Automated running of data science models
Automated running of data science models:

(Cloud Credentials) User enters their cloud credentials for the MOC.

(Model and Dataset selection) User selects the main program that encapsulates their data science model and the specific dataset they want to run it against.

(Running Model) Scripts will create the needed instances and run specified main program against dataset using given user credentials. Results are returned to the user.

Demo 4 Apr 13:
User stories: Selection of preferred cloud provider, Activity Log

Selection of preferred cloud provider:

(Cloud selection) User selects their preferred cloud vendor (MOC, Azure, AWS, etc…) and enters their credentials.

(Automated running of data science models [multi-cloud support]; refer to user story listed in previous release) Scripts will create the needed instances on the selected cloud vendor using user credentials. Results are returned to user.

Activity Log:

(Tracking) As a user performs actions to the image (ie, updating packages or running their models in the cloud), the image creates a log of activity.

(Viewing) Activity log can be viewed by any user, but cannot be edited by user.

Final Demo Apr 27:
User stories: Documentation, Saved administrator preferences
Documentation:

(Open Source Repository) The project should be distributed as open source on Github. The project should come bundled with appropriate documentation.

(Usage guide) The project should also contain a usage guide, detailing step-by-step instructions for administrators and regular users for how to get started using the project.

Saved administrator preferences(Stretch goal):

(User creation) The user should select “Register” on the web application in order to create a new account. This information will be registered and securely stored in an authorization database. The user will then be brought to their homepage where they can view previously saved image preferences.

(Saved preferences) The administrator should be able to specify if they wish to save their selections for future editing on the web application. A list of these will be saved in a database and visible on the homepage.