Dashboard App creation via Python Dash

Visualize and create a model

Dashboard App

Our use case lets users visualise datasets without having any dependencies on the technical prerequisites. The aim here is to provide users with a dashboard driven machine learning tool. Some parallels of this tool are Tableau and weka. The powerful edge that the tool has is that no download of any shorts is needed. Our Tool is completely cloud based. The ease this gives is that even the non technical members of the group can use the aspects of machine learning easily. As we know the demand for the cloud based solution is rising every quarter. Due to this wide utility of cloud based solution I have implemented the solution on a cloud platform Heroku and AWS. Moreover the project is available on Github too for further research and development. In today's times the size of data is increasing day by day. Thus making it very critical to have a GUI driven cloud hosted ML tool that runs just like a blogging or any other site.
We notice that the dashboard approach is also something that many tools tend to integrate these days. I have followed the same trend and implemented a dashboard approach rather than logging the event system for any particular data point. This dashboard is used to do following functionality :
i.) visualize the datasets and download its report(consists of accuracy, val_accuracy, loss and val_loss), shows as accuracy and loss graphs and .h5 extension model using artificial neural networks with different types of activation function and optimizers.
ii.) visualize the datasets and download its report(consists of accuracy, val_accuracy, loss and val_loss), shows as accuracy and loss graphs and .h5 extension model using inverse artificial neural networks with different types of activation function and optimizers. **Note: Work is still in progress.**
iii.) creates model and report/results for image dataset (in available online platform data should not exceeds 100Mb capacity), shows as accuracy and loss graphs. We can also test the model by providing input image to the online application and classify its output.

Installation Manual with deployment in Heroku.

Every Dash script so far has used app.run_server() to launch the app. By default the app runs on **localhost**, and you can only see it on your own machine.

The good news is that Dash uses Flask as its web framework, so anywhere you can deploy Flask, you can deploy Dash. While there are many options out there including Digital Ocean, PythonAnywhere, Google Cloud, Amazon Web Services, Azure, etc., we'll walk through an app deployment on Heroku.

We can run the application in two ways:
Steps to install in your own computer:
Step 1 - Clone project form Git
Use below command on command prompt to clone the project form git or directly download the project from github.

git clone LINK

STEP 2 - Directed to Development Folder

Extract the downloaded or cloned development folder, then move to the development folder:

C:\>**cd demo_easy_dash**

STEP 3 (WINDOWS) - Create virtualenv and download all the dependencies:
see below for macOS/Linux instructions!

1. Create a virtual environment. We're calling ours "venv" but you can use any name you want:
   C:\my_dash_app>**python -m virtualenv venv**
2. Activate the virtual environment:
   C:\my_dash_app>**.\venv\Scripts\activate**
3. Install dash and any desired dependencies into your virtual environment
   (venv) C:\my_dash_app>**pip install -r requirements.txt**

STEP 3 (macOS/Linux) - Create virtualenv and download all the dependencies:

1. Create a virtual environment. We're calling ours "venv" but you can use any name you want:
   $ **python3 -m python3 -m virtualenv venv**
2. Activate the virtual environment:
   $ **source venv/bin/activate**
3. Install dash and any desired dependencies into your virtual environment
   $ **pip install -r requirements.txt**

STEP 4 - Run the program:

You can run the program by below code:

python app3.py

Steps of deployment:

STEP 1 - Install Heroku and Git

Heroku is a cloud platform that lets users deploy apps on the web.

Git is a version control system that will let you keep a local copy of your app for development, and enable you to push changes from your development copy to the deployed version stored at Heroku.

1. Open a **Heroku** account. Free accounts are available at https://signup.heroku.com/dc

Follow the instructions to obtain a username and password. Write them down!

2. Log into your Heroku account. It should take you to https://dashboard.heroku.com/apps
3. Click on **Python**. On the next screen select **Set Up**. An option should appear to download the **Heroku Command Line Interface (CLI)**. Choose your operating system from the dropdown list and follow the instructions to install the utility. You should have the option to install **Git** as well.
4. If **git** was *not* installed with Heroku CLI, you can download it directly from https://git-scm.com/downloads and follow the instructions for your operating system.

STEP 2 - Install virtualenv

1. Install **virtualenv** if you don't already have it by typing **pip install virtualenv** at your terminal. Virtualenv allows you to create virtual environments for your app that house Python and all the dependencies your app requires. This includes specific version of plotly, dash, and other libraries that you know will work.

As new updates become available, they won't break your app until you've had a chance to test them first!

STEP 3 - Create a Development Folder

1. Create a new folder for your project. This will house the "development" copy of your app:
   C:\>**mkdir demo_easy_dash**

C:\>**cd demo_easy_dash**

STEP 4 - Initialize Git
1. Initialize an empty git repository:
C:\demo_easy_dash>**git init**
Initialized empty Git repository in C:/demo_easy_dash/.git/

STEP 5 (WINDOWS) - Create, Activate and Populate a virtualenv
see below for macOS/Linux instructions!
1. Create a virtual environment. We're calling ours "venv" but you can use any name you want:
C:\my_dash_app>**python -m virtualenv venv**
2. Activate the virtual environment:
C:\my_dash_app>**.\venv\Scripts\activate**
3. Install dash and any desired dependencies into your virtual environment
(venv) C:\demo_easy_dash>**pip install dash**
(venv) C:\demo_easy_dash>**pip install -r requirements.txt**
4. Install a new dependency **gunicorn** for deploying the app:
(venv) C:\my_dash_app>**pip install gunicorn**

STEP 5 (macOS/Linux) - Create, Activate and Populate a virtualenv
1. Create a virtual environment. We're calling ours "venv" but you can use any name you want:
$ **python3 -m python3 -m virtualenv venv**
2. Activate the virtual environment:
$ **source venv/bin/activate**
3. Install dash and any desired dependencies into your virtual environment
$ **pip install -r requirements.txt**
4. Install a new dependency gunicorn for deploying the app:
$ **pip install gunicorn**

STEP 6 - Add Files to the Development Folder
The following files need to be added:

app1.py	a Dash application
.gitignore	used by git, identifies files that *won't* be pushed to production
Procfile	used for deployment
requirements.txt	describes your Python dependencies, can be created automatically

.gitignore

```
venv
*.pyc
.DS_Store
.env
```

Procfile

```
web: gunicorn app1:server
```

app1 refers to the filename of our application (app1.py) and server refers to the variable **server** inside that file.
requirements.txt
This can be automatically generated by running pip freeze > requirements.txt at the terminal.
Make sure to do it from inside the development folder with the virtual environment activated.

(venv) C:\demo_easy_dash>**pip freeze > requirements.txt**

STEP 6 - Log onto your Heroku Account
At the terminal, login using the credentials you established in **STEP1**:
(venv) C:\demo_easy_dash>**heroku login**
Enter your Heroku credentials:
Email: my.name@somewhere.com
Password: ********
Logged in as my.name@somewhere.com

STEP 7 - Initialize Heroku, add files to Git, and Deploy

(venv) C:\demo_easy_dash>**heroku create *my-dash-app***
You have to change ***my-dash-app*** to a unique name. The name must start with a letter
and can only contain lowercase letters, numbers, and dashes.
(venv) C:\demo_easy_dash>**git add .**
Note the period at the end. This adds all files to git (except those listed in .gitignore)
(venv) C:\demo_easy_dash>**git commit -m "Initial launch"**
Every git commit should include a brief descriptive comment. Depending on your operating system, this comment may require double-quotes (not single-quotes).
(venv) C:\demo_easy_dash>**git push heroku master**
This deploys your current code to Heroku. The first time you push may take awhile as it has to set up Python and all your dependencies on the remote server.
(venv) C:\demo_easy_dash>**heroku ps:scale web=1**
Scaling dynos... done, now running web at 1:Free
This runs the app with a 1 heroku "dyno"

STEP 8 - Visit Your App on the Web!
You should be able to view your app at https://**my-dash-app**.herokuapp.com
(changing **my-dash-app** to the name of your app)

STEP 9 - Update Your App

Any time you make changes to your app, add new apps to your repo, or install new libraries and/or upgrade existing dependencies in your virtual environment, you want to push the latest updates to Heroku. These are the basic steps:

If installing a new package:
$ **pip install *newdependency***
$ **pip freeze > requirements.txt**

If updating an existing package:
$ **pip install *dependency* --upgrade**
$ **pip freeze > requirements.txt**

In all cases:
$ **git status** *# view the changes (optional)*
$ **git add .** *# add all the changes*
$ **git commit -m "a description of the changes"**
$ **git push heroku master**