

# AWS, FLASK, and Deploying Models

W07D2

Instructor: Mohammad Raza

# Agenda for today

- Overview of Cloud Computing
- AWS
- Flask + Flask-restful
- Deployment with Tmux









**Cloud Computing** 

# Advantages:

- -Easy Implementation
- -Accessible
- -No Hardware Required
- -Cost per head
- -Flexible for growth
- -efficient

# Disadvantages:

- -No longer in control
- -bandwidth issues
- -downtime (slack 2021)

# **AWS**

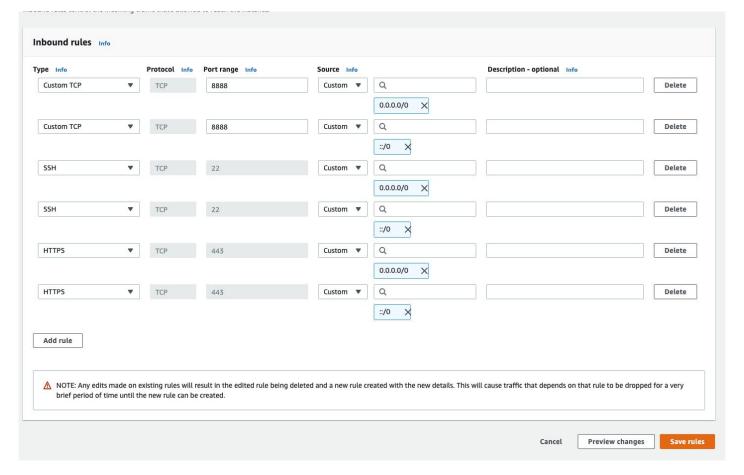
- Top Tier Cloud Computing Service 31% of market share
- Pay as you Go model
- Netflix, Reddit, Nasa, Quora, Airbnb, Foursquare, and much of the internet is run on AWS
- Tonnes of services
- Accounts for 12% of Amazon's Revenue, but 57% of Operating Income (Profits)

# AWS - Demo

- 1.) Create an Account
- 2.) Create a key pair (.pem for MacOSX or Linux, .ppk for WINDOWS)
- 3.) Create Instance
- 4.) Edit Security Group (next slide for specific protocols)
- 5.) WINDOWS SPECIFIC: Install PUTTY (tutorial)(tutorial video)
  - a.) Start PUTTY
  - b.) Category Pane CLICK SESSION
    - i.) Host Name = ec2-user@ec2-15-223-49-239.ca-central-1.compute.amazonaws.com
    - ii.) Connection Type = SSH
    - iii.) Port = 22
  - c.) Category PANE CLICK CONNECTION/SSH/AUTH
    - i.) Browse to .ppk file
    - ii.) OPEN
- 6.) Connect to Instance
- 7.) <a href="https://www.anaconda.com/products/individual">https://www.anaconda.com/products/individual</a> (Copy the linux download link to get the latest one)
- 8.) Use wget method to download or curl
- 9.) sh Anaconda3-2020.11-Linux-x86\_64.sh (run this command to install)
- 10.) Exit connection and re-enter connection.
- 11.) jupyter lab --ip 0.0.0.0 --port 8888 --no-browser

https://repo.anaconda.com/archive/Anaconda3-2021.11-Linux-x86\_64.sh

# AWS - Security Inbound Rules



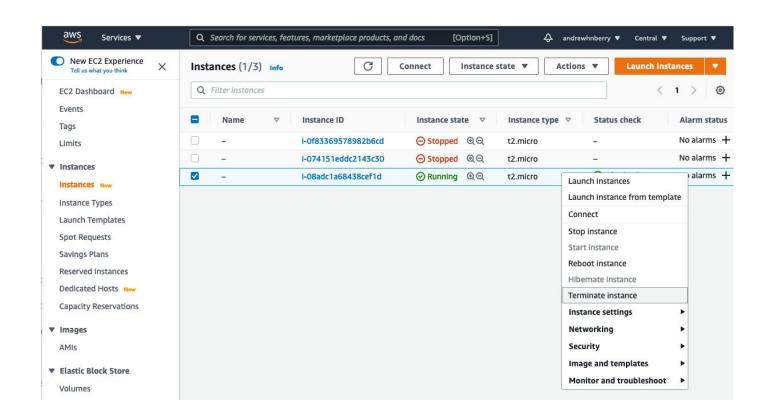


# Don't forget to shut down your instance!

1.) Click Instance

2.) Right click on instance

3.) Select Stop



# **APIs review**

What is an API?

Why do APIs exists?

How does a data scientist use an API?

# **APIs review**

What is an API?

Why do APIs exists?

How does a data scientist use an API?

How do we create our own APIs?



# **RESTFUL API Review**

### ### The Anatomy Of A Request

### It's important to know that a request is made up of four things:

- 1. The endpoint (the URL)
- 2. The method (verb)
- 3. The headers (parameters)
- 4. The data (or body)

### 1. The endpoint (or route) is the url you request for

root-endpoint/?

https://api.github.com

- 2. The Method is the type of request you send to the server. You can choose from these types below:
- a. GET Used to get resource from server
- b. POST Used to create new resource on server
- c. PUT/PATCH update resource on server
- d. DELETE delete a resource on the server

# with FLASK

- Flask is a web development framework for Python, similar to Django
- However, there we can make APIs with it
- Using Flask restful

**Alternatives to Flask-restful for creating APIs**: FastAPI, Django REST framework

## Setting up a flask development environment:

- Create a new conda environment
- Conda install flask
- Pip install flask-restful

# Deploying your model to the cloud - DEMO

Copy your app.py file and your pickle file to the cloud

For MacOSX and Linux: use scp or CyberDuck

Example: scp -i myAmazonKey.pem /path/to/file username@address:/path/to/destination

https://stackoverflow.com/questions/11388014/using-scp-to-copy-a-file-to-amazon-ec2-instance

For Windows: use winSCP

or....use nano or vim to write your code in ec2 terminal

# Cyberduck tutorial

Secure File Transfer Protocol (SFTP) with Cyberduck (compatible for MacOS and Windows)

- Install https://cyberduck.io/download/
- 2. On MacOS brew install -- cask cyberduck -
- 3. Cyberduck is free but will promt for donations. To disable the prompt, a registration key can be purchased from the link above or the app can be purchased from the Mac App Store
- 4. Connect to AWS EC2 Instance with Cyberduck 8.2.1
- 5. Open Cyberduck
- 6. Open Connection
- 7. SFTP (SSH File Transfer Protocol)

Server: < Public IPv4 DNS>,

Port: 22 # Default,

Username: 'ec2-user' or 'ubuntu', # depends on Amazon Machine Image

Password: <blank>,

SSH Private Key: /path/to/key.pem

- Connect Once connected, you can disconnect using the Disconnect button at the top of the window. Connections are saved in the Bookmarks list

Process adapted from the [tutorial by Jhonny Fransis](https://www.youtube.com/watch?v=hd4oL3WIPVM)

# Don't forget!

In your app.py file in your ec2. Don't forget to add the port and host parameter

```
If __name__ == '__main__':
app.run(host = "0.0.0.0", port = 8888)
```

Port can be anything that is not in use.

# tmux - Command (Essentials) - CHEAT SHEET

\$sudo yum install tmux or

\$sudo apt install tmux

### CRTL + B:

- % (New Pane)
- $\leftarrow$  or  $\rightarrow$
- " (Pane Bottom)
- exit (to close, pane & windows)
- c (new window),
  - (switch back CRTL B + 0)
  - , (rename)

- CRTL+B & d (Detach Session)
- tmux ls (LIST ALL SESSIONS)
- tmux attach -t NAME
- tmux rename-session -t old\_name new name
- Tmux new -s new name
- Tmux kill-session -t session\_name

# How to save your custom transformers in your pipeline to load later?

- One caveat of the pipelining in scikit-learn is that you can't save your custom transformers in your pipeline to a pickle file. You will have to copy your custom transformers code to the app.py file.
- Unless you use cloudpickle
  - <a href="https://github.com/cloudpipe/cloudpickle">https://github.com/cloudpipe/cloudpickle</a>

\_