**Installation Steps**

Contents

[**1.** **First Time Setup** 1](#_Toc435312307)

[1.1. Extract the Zip files into a folder on local machine 1](#_Toc435312308)

[1.2. Open a new AWS EC2 Instance (do not open an existing one) 1](#_Toc435312309)

[1.3. Run Putty for EC2 Instance 2](#_Toc435312310)

[1.4. EC2 Command Prompt 2](#_Toc435312311)

[1.5. WinSCP the data files into EC2 Instance 2](#_Toc435312312)

[1.6. Mounting the data files into docker image and launching tmpnb server 3](#_Toc435312313)

[1.7. Access the final tmpnb server with the following url: 4](#_Toc435312314)

[**2.** **Rerun once you shut down your amazon instance** 4](#_Toc435312315)

[2.1. Run Putty for EC2 Instance 4](#_Toc435312316)

[2.2. EC2 Command Prompt 5](#_Toc435312317)

[2.3. Access the final tmpnb server with the following url: 5](#_Toc435312318)

# **First Time Setup**

# Extract the Zip files into a folder on local machine

The zip files open into the following structure

* + - **<train>** folder which has the data files and notebook
      1. validationIdx1-transformed.csv
      2. **Python notebook**

# Open a new AWS EC2 Instance (do not open an existing one)

Follow the steps below

* Select Amazon Linux Instance
* Choose m4.2xlarge (8vCPU, 32 GB memory) Instance Type
* Add 30 GiB Storage (SSD)
* Configure Security Group for ports: 8000, 8888 and http, https
* Launch Instance

# Run Putty for EC2 Instance

Follow the steps below

* Enter Public IP and SSH keypair in Putty Configuration window
* Click Open

# EC2 Command Prompt

Follow the steps below

* Login as **ec2-user**
* Follow the following steps to setup tmpnb Server on AWS
  + Type the following command without quotes to install docker, and if prompted, allow the installation:

**sudo yum install docker**

* + Type the following command to start the docker daemon:

**sudo service docker start**

* + Type the following command in the AWS EC2 instance terminal to save a unique random token to your linux environment. This will be used for authentication by your docker containers.

**export TOKEN=$( head -c 30 /dev/urandom | xxd -p )**

* + Type the following command in the AWS EC2 instance terminal to start the proxy server. This proxy server will be used to route the request from the web to your tmpnb orchestrator running locally.

**sudo docker run --net=host -d -e CONFIGPROXY\_AUTH\_TOKEN=$TOKEN --name=proxy jupyter/configurable-http-proxy --default-target** [**http://127.0.0.1:9999**](http://127.0.0.1:9999)

* + Type the following command to download the docker image containing the custom ipython notebook that you want to run on the tmpnb docker instance

**sudo docker pull waituck/custom\_nb**

This will take some time to download and extract images.

# WinSCP the data files into EC2 Instance

Follow the steps below

* Download and Install WinSCP 5.7.5 for windows from the following URL

<https://winscp.net/eng/download.php>

* Open installed WinSCP
* In the window follow the following steps
  + Choose SCP in File Protocol
  + Copy AWS EC2 instance Public IP in Host name. Keep port at 22
  + Do not enter User Name and Password
  + Click Advanced button
  + Choose SSH>authentication in left panel
  + In the field Private key file…Select the keypair used for logging into putty
  + Click OK
  + Click Login
  + You will get a **popup “Continue connecting to a unknown server and add its host key to a cache?”.** Click Yes to that window
  + You will be asked for user name. Enter **ec2-user**
  + You will see a window with an LHS and and an RHS window. LHS is local drive and RHS is your ec2 instance drive
  + In RHS, create a new folder **“train”** in the default **home/ ec2-user/**
    - To create folder righ click and choose New > Directory
    - **Name the folder “train”**
  + From LHS, drag the data files extracted in step 1 above and drop it in RHS within “trial” folder
  + The upload process will begin. Please wait as this will take few minutes.
  + Keep the WinSCP window running

# Mounting the data files into docker image and launching tmpnb server

Follow the steps below

* Type the following command to SSH into the notebook docker image from your EC2 instance:

**sudo docker run -v /home/ec2-user/train:/mnt -it waituck/custom\_nb /bin/bash**

* You will be in the terminal of the image. Type the following command to copy the files into the image:

**cp -r /mnt/\* .**

Copy the entire command including the dot that comes after asterix and space. Please do not remove space between “\*” and “.”

* You may check if the files are in with the following command:

**ls –a**

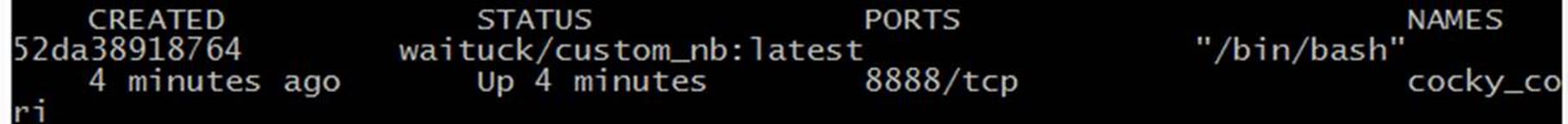
Please do not remove space between “ls” and “-a”

You should see your files.

* Next, very important, press Ctrl + P, then Ctrl + Q, without releasing Ctrl. You will be back in the ec2-user terminal.
* Type the following:

**sudo docker ps**

* You should see something like this



With reference to the name, in this case cocky\_cori (replace it with what you see in the ec2 command screen of yours), type the following command (This will take some time! Be patient!)

**sudo docker commit cocky\_cori mydocker**

**Remember:** Replace “cocky\_cori” in above script with what you see in the ec2 command screen of yours before executing

**This may take sometime**

This will save your docker image.

* Now when you launch the tmpnb server start command from this:

sudo docker run --net=host --name=tmpnb -d -e CONFIGPROXY\_AUTH\_TOKEN=$TOKEN \

           -v /var/run/docker.sock:/docker.sock \

           jupyter/tmpnb python orchestrate.py --image='mydocker' --pool\_size=10 --command="ipython notebook --NotebookApp.base\_url={base\_path} --ip=0.0.0.0 --port {port}”

# Access the final tmpnb server with the following url:

**http://<Public IP>:8000**  
**Replace <Public IP> with your public IP that can be found below**

# **Rerun once you shut down your amazon instance**

# Run Putty for EC2 Instance

Follow the steps below

* Enter Public IP and SSH key pair in Putty Configuration window
* Click Open

# EC2 Command Prompt

Follow the steps below

* Login as **ec2-user**
* Follow the following steps to setup tmpnb Server on AWS
  + Restart Docker

**sudo service docker restart**

* + Remove the previously started docker instances

**sudo docker rm proxy  
sudo docker rm tmpnb**

* + Set the Environment Variable

**export TOKEN=$( head -c 30 /dev/urandom | xxd -p )**

* + Start the Docker Instances

sudo docker run --net=host -d -e CONFIGPROXY\_AUTH\_TOKEN=$TOKEN --name=proxy jupyter/configurable-http-proxy --default-target <http://127.0.0.1:9999>

sudo docker run --net=host --name=tmpnb -d -e CONFIGPROXY\_AUTH\_TOKEN=$TOKEN \

           -v /var/run/docker.sock:/docker.sock \

           jupyter/tmpnb python orchestrate.py --image='mydocker' --pool\_size=10 --command="ipython notebook --NotebookApp.base\_url={base\_path} --ip=0.0.0.0 --port {port}”

# Access the final tmpnb server with the following url:

**http://<Public IP>:8000**  
**Replace <Public IP> with your public IP that can be found below**