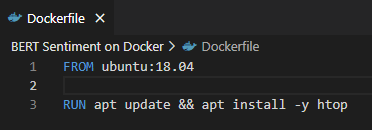
1. Below is the starter Dockerfile:



1. For windows, launch Docker Desktop first.

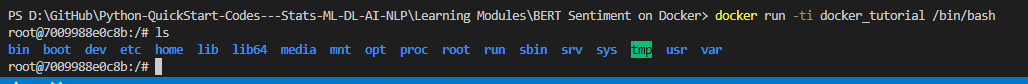
Execute below command in cmd/powershell :



-f Dockerfile : indicates the filename from step 1.

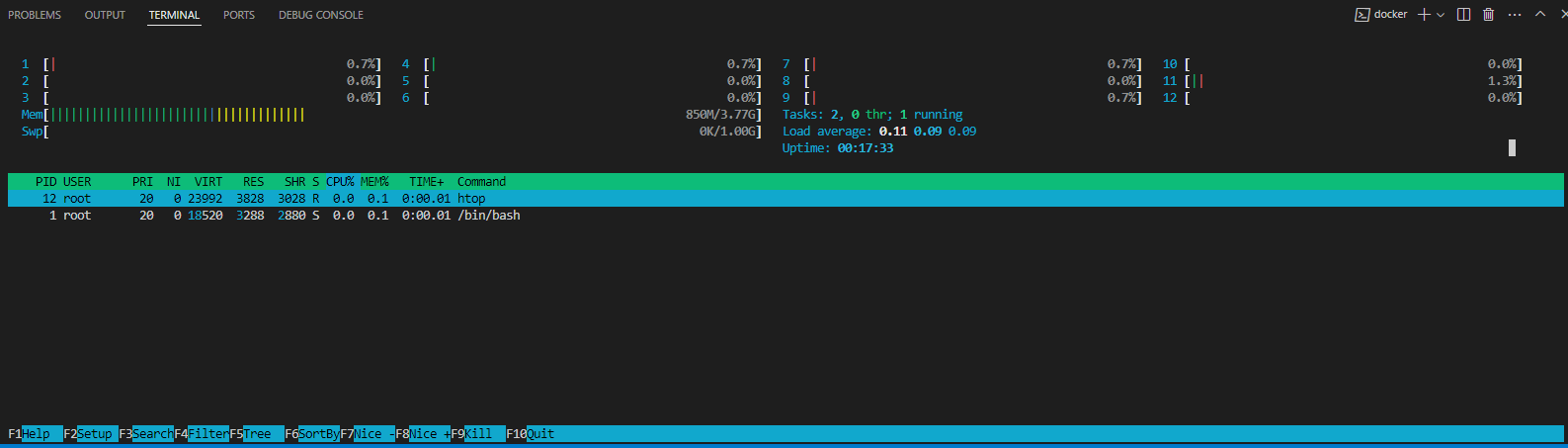
-t docker\_tutorial . : indicates the target image name we are creating and ‘.’ Is location of this image.

1. Run the docker now. Once you’re inside this container you’ll see something like root@\*\*\*\* as indicated below. We can do ls to confirm that we are in filesystem from ubuntu now.



-ti docker\_tutorial : indicates the target docker image we created in step 2.

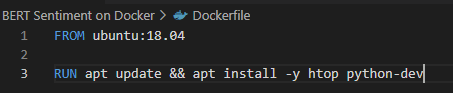
Since we installed htop in the docker image we created, we can use command ‘htop’ to view below:



We don’t have python in this container yet:



1. Let’s install python in this container now. Change the Dockerfile as below:



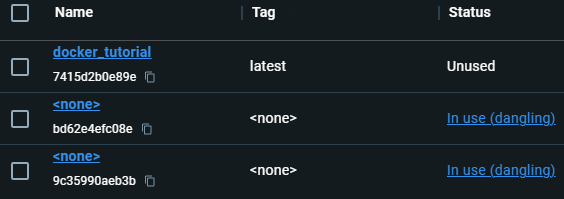
Now, exit from the docker image running in terminal using ‘exit’ command:



Now use command from step 2 to build the docker and use command from step 3 to run it. Check python version, if its version 2 as shown below then modify Dockerfile again as indicated here and build it again and recheck python version (or straight away go with this new Dockerfile in this step):

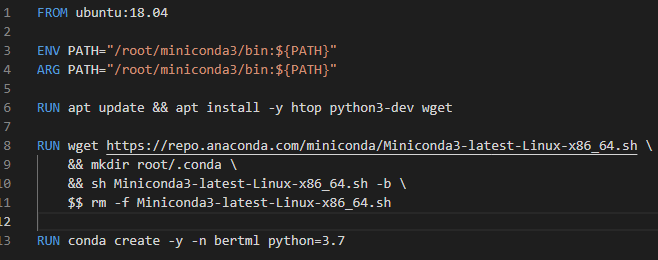
 (use ‘python3 -V’ to check py3 version)

Note: At this point docker desktop will show 3 images, two in dangling state. These spaces can be freed with ‘docker image prune’ command (but it seems those images are not gone; delete containers for these images from containers tab in docker desktop application):



Refer to this link for more details/possibilities: [How to Remove Docker Images | Step-by-Step | Cherry Servers](https://www.cherryservers.com/blog/how-to-remove-docker-images)

1. Now install conda and create a new virtual environment inside this docker container:



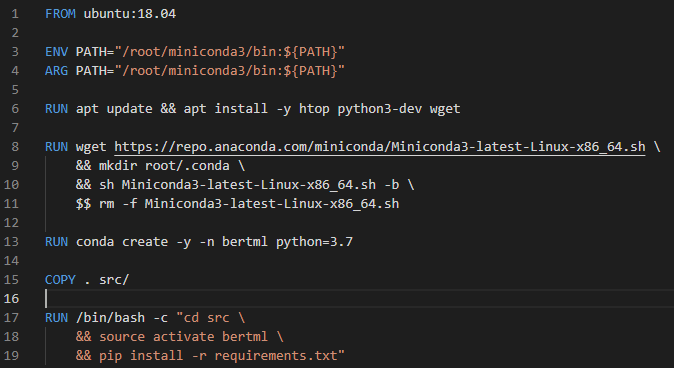
Line6: install wget to use it in line8

Line8: get url for miniconda file from its website and in the next lines, create directory to install it in ‘.conda’ folder. Install it with ‘-b’ command for silent install with no path modification in bashrc file.

Line 3,4: line13 will not work unless conda is added in PATH variable. Use ENV and ARG instructions.

Once built and run, this docker container’s bin/bash will have ‘conda’ and ‘python’ commands working. ‘conda activate bertml’ will not work so use ‘source activate bertml’ instead.

1. Now create a new folder src and copy files from local. Then, install required pip packages:



Once this is built, use ‘cd src/’ and ‘ls’ to confirm that we have the folder created and files available.

Go to the src folder which has out codes and data files and run ‘train.py’ to train the model.

1. Use following command to mount volume from local to docker container and launch it:

docker run -v <source>:/root/docker\_data/ -ti docker\_tutorial /bin/bash

This will mount local(<source>) folder into /root/docker\_data/ location so we don’t have to build it again, we can just run it as above. Any change made in this root folder inside docker will be reflected in our local as well so we are directly working with out local files but from within docker container.

The <source> folder from windows must be converted to quasi-Linux path as per below:

-- replace any drives of the format C:\ with the new format //c/

-- any back-slash is to be replaced with a forward-slash

For instance, “D:\mlruns\models” will be converted to “//d/mlruns/models”

Note: We can directly run the train command while activating docker container as below:

docker run -v <source>:/root/docker\_data/ -ti docker\_tutorial /bin/bash -c “source activate bertml && cd src && python train.py”

1. Instead of training model on docker, train it in local and upload it in docker. If training on docker, use some other image from docker hub which has nvidia cuda files installed on ubuntu. Running app.py on this container will expose a port so ass one more parameter as shown below to access flask ruinning in docker image from local browser:

docker run -p 7000:9999 -v <source>:/root/docker\_data/ -ti docker\_tutorial /bin/bash -c “source activate bertml && cd src && python train.py”

Learn more about using dockers here : [Data Science Workflows using Docker Containers (youtube.com)](https://www.youtube.com/watch?v=oO8n3y23b6M)