Remote Enviroment Setting

- Jupyter Notebook remote environment setting.
- Python IDE remote environment and debugging
- Kaggle auto submission

We provide two working environment settings. The first one is jupyter notebook based. And the second one is python source code based. Both should be easy to debug and interactively understand your code.

1. **Jupyter notebook + AWS EC2.**
    - Download the key pair of your EC2, `deeplearning.pem`. (The following steps is assuming you know how to connect to you EC2 machine, if you do not, you can check the AWS recitation)
    - Log on EC2, Run the following code to set the password so that your local machine could access it using the password.

      jupyter notebook password

    - On EC2, run the following code to open a jupyter notebook server listening at port 8889. If you close the terminal, then this server would be killed. (trick: You can add nohup to prefix this command line code to let it run in the background. So the server would not be terminated if you close the terminal.)

      (nohup) jupyter notebook --no-browser --port=8889

    - On local machine, run the following command to connect 8888 port number on your machine to 8889 port on remote machine.

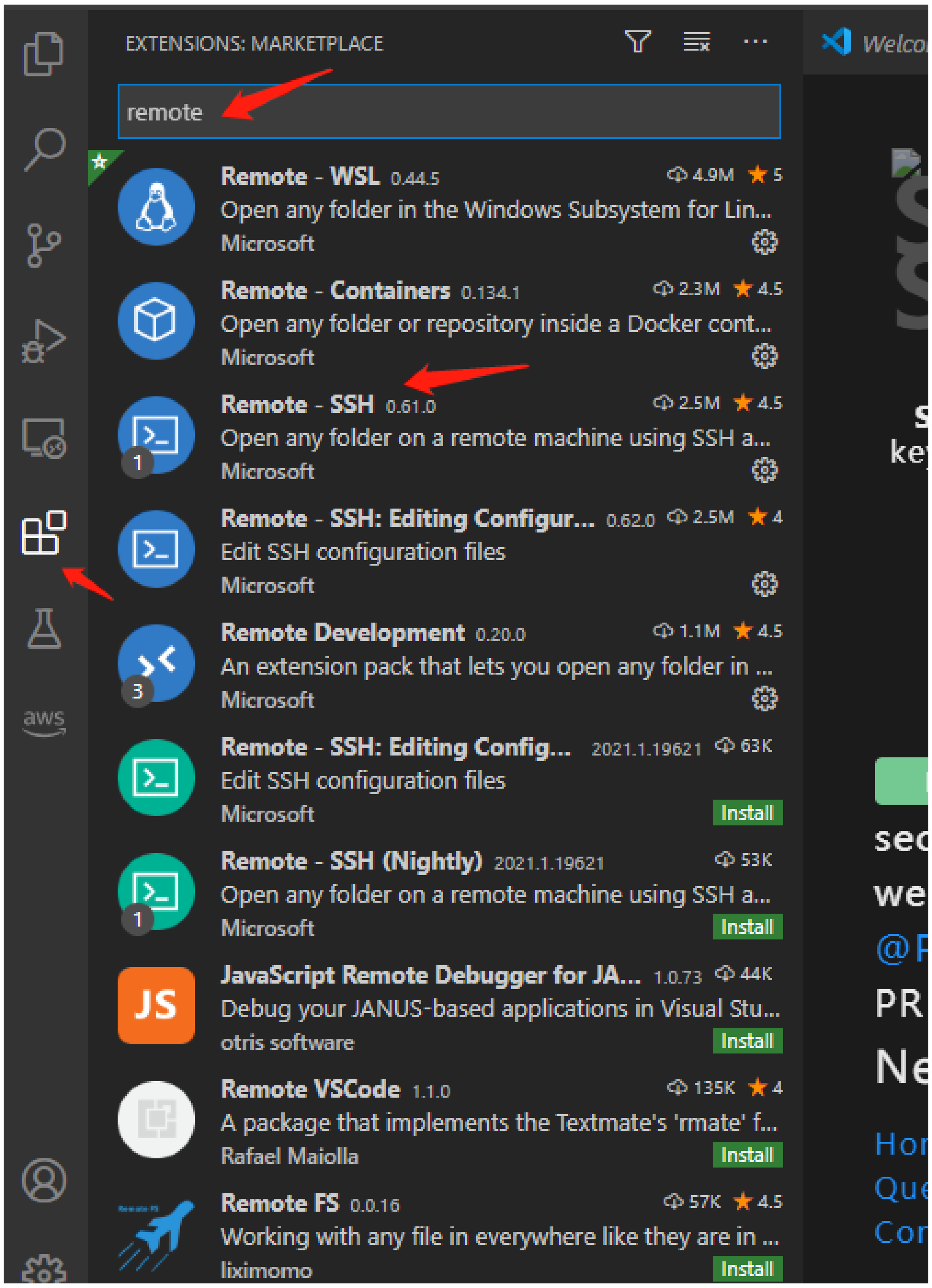
  ssh -i deeplearning.pem -N -f -L localhost:8888:localhost:8889 ubuntu@ec2instance

    - Open your browser and type in "localhost:8888", you can then input the password and access the remote jupyter server like you running a jupyter notebook on a local machine.
    - Trick: Sometimes you could close the terminal and run your model overnight. The second day you wake up and connect to the server. You will find that the output disappears. This happens because each terminal is a process and when you close it, it would be killed and the next time you connect to it, it's a new process. The best way to keep track of your output is to using logging package in python.(https://docs.python.org/3/howto/logging-cookbook.html). You can add two handlers so that your output would be printed and also be written to the log file.

2. **VScode Remote-SSH and remote debugging**

One problem of coding on a remote machine is that you cannot use user-friendly IDE or text edit like you are coding on your macbook. VScode + Remote-SSH could help you. VScode is and IDE, while Remote-SSH is a powerful plugin in VScode.

- First, install Remote-SSH plugin first.



EXTENSIONS: MARKETPLACE

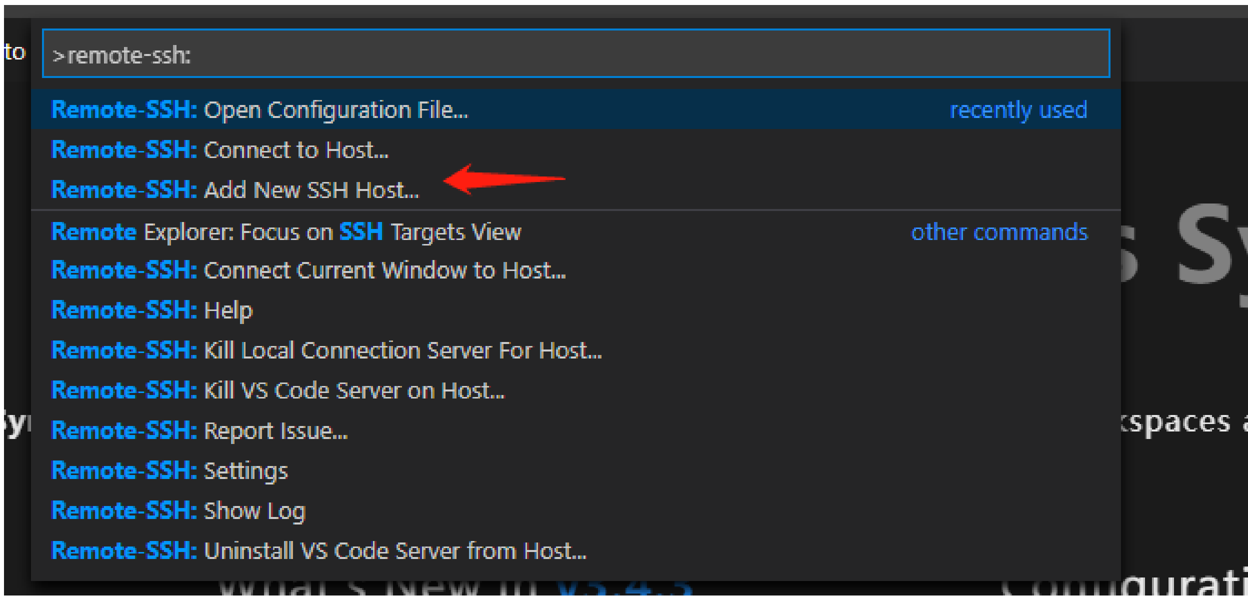
remote

Remote - WSL 0.44.5	Open any folder in the Windows Subsystem for Lin...	Microsoft	4.9M ★ 5
Remote - Containers 0.134.1	Open any folder or repository inside a Docker cont...	Microsoft	2.3M ★ 4.5
Remote - SSH 0.61.0	Open any folder on a remote machine using SSH a...	Microsoft	2.5M ★ 4.5
Remote - SSH: Editing Configur... 0.62.0	Edit SSH configuration files	Microsoft	2.5M ★ 4
Remote Development 0.20.0	An extension pack that lets you open any folder in ...	Microsoft	1.1M ★ 4.5
Remote - SSH: Editing Config... 2021.1.19621	Edit SSH configuration files	Microsoft	63K Install
Remote - SSH (Nightly) 2021.1.19621	Open any folder on a remote machine using SSH a...	Microsoft	53K Install
JavaScript Remote Debugger for JA... 1.0.73	Debug your JANUS-based applications in Visual Stu...	otris software	44K Install
Remote VSCode 1.1.0	A package that implements the Textmate's 'rmate' f...	Rafael Maiolla	135K ★ 4 Install
Remote FS 0.0.16	Working with any file in everywhere like they are in ...	liximomo	57K ★ 4.5 Install

- (Optionally, but recommended),Second, delete following files:
  If windows system:
  C:\Users\your_user_name\.ssh\config
  C:\Users\your_user_name\.ssh\known_hosts

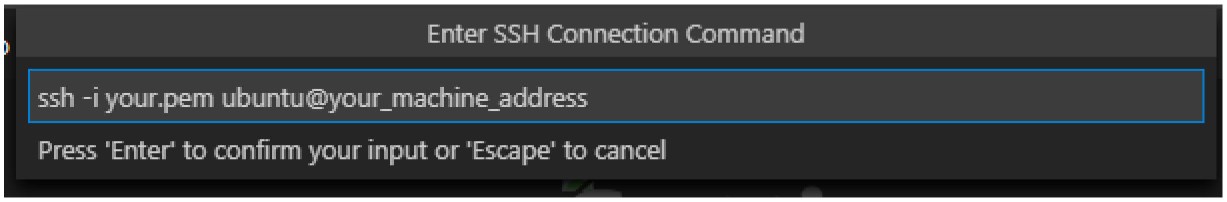
Other system:
\Users\your_user_name\.ssh\config
\Users\your_user_name\.ssh\known_hosts

- Third, press **shift+cmd+p** to all the panel out, then type in “remote-ssh:” and choose **Add New SSH Host**



```
>remote-ssh:
Remote-SSH: Open Configuration File...                recently used
Remote-SSH: Connect to Host...
Remote-SSH: Add New SSH Host...
Remote Explorer: Focus on SSH Targets View            other commands
Remote-SSH: Connect Current Window to Host...
Remote-SSH: Help
Remote-SSH: Kill Local Connection Server For Host...
Remote-SSH: Kill VS Code Server on Host...
Remote-SSH: Report Issue...
Remote-SSH: Settings
Remote-SSH: Show Log
Remote-SSH: Uninstall VS Code Server from Host...
```

- Fourth, type in the command you use to login to EC2 instance.

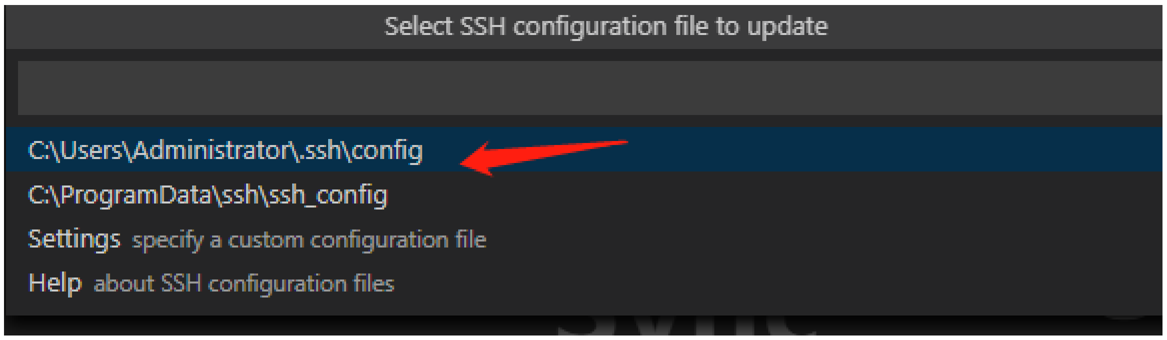


Enter SSH Connection Command

```
ssh -i your.pem ubuntu@your_machine_address
```

Press 'Enter' to confirm your input or 'Escape' to cancel

- Fifth, choose the directory you want to save your server information. Normally the first one.



Select SSH configuration file to update

```
C:\Users\Administrator\.ssh\config
C:\ProgramData\ssh\ssh_config
Settings specify a custom configuration file
Help about SSH configuration files
```

- Then connect.

Now you have successfully connected to the host, you need to configure the debugging file.

- First is to install the **Python plugin** both on your local machine and remote machine.
- Next, we need to open the remote folder and create the debug setting file. The debug setting file should be like following.
- Now, the rest is basically same as you are debugging on your local machine.

3. **Kaggle API**
   - Download your kaggle.json file at Kaggle->My Profile page.
   - Install Kaggle package on EC2 machine.

```
pip install kaggle
```

   - Copy your kaggle.json to .kaggle/ folder so that kaggle package could know who you are.

```
scp -i deep_learning.pem ./kaggle.json ubuntu@instance:~/.kaggle/
```

   - You could easily download the data using the following command. competition-name could be found on the ‘Data’ section in the competition main webpage.

```
kaggle competitions download -c competition-name
```

   - You can submit using the following code:

```
kaggle competitions submit -c competition-name -f your-submission-path -m "Message"
```