基于Linux系统的远程科学计算环境配置

0 云服务器配置

系统镜像：centos_8_0_x64_20G_alibase_20200218.vhd（阿里云）

1. Anaconda3

1.1 安装Anaconda3

首先要用ssh远程连接服务器（SSH连接软件—Windows：Xshell；Mac：Termius）

Linux系统可以直接打开终端输入

```
ssh user@ip
```

用wget命令将安装包下载至远程服务器Anaconda3-2019.10-Linux-x86_64.sh

```
wget https://mirrors.tuna.tsinghua.edu.cn/anaconda/archive/Anaconda3-2019.
10-Linux-x86_64.sh
```

在终端输入 `ls` 可以看到安装包后，使用bash命令来进行安装

```
bash Anaconda3-2019.10-Linux-x86_64.sh
```

一直按Enter继续，接受许可条款。 选择 `yes or no?` 时键入 `yes` 继续。

1.2 查看是否安装成功

安装完成后断开ssh并重新连接至远程服务器，输入python可以激活anaconda内的Python环境。

使用以下命令可以将auto_activate_base参数设置为false，取消每次启动时自动激活conda的base环境。

```
conda config --set auto_activate_base false
```

1.3 Conda创建新环境

```
conda create --name AlphaEnv # 不指定Python版本则使用默认版本。
conda create --name AlphaEnv python=3.7.4  # 指定Python版本。
conda create --name AlphaEnv numpy scipy # 创建包含指定包的环境。
```

下载 `NEW packages` 时如果询问 `Proceed ([y]/n)?`，键入 `y` 确认即可。

创建完成后终端上会出现如下提示：

```
# To activate this environment, use
#
#     $ conda activate AlphaEnv
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

激活刚才新创建的Conda环境

```
conda activate AlphaEnv
```

1.4 Conda的一些常用命令

```
conda -V # 查看当前版本
conda -h # 获取帮助
conda update -h # 获取指定命令的帮助
conda list  # 查看这个环境下安装的包和版本
conda install numpy scikit-learn  # 安装numpy sklearn包
conda env list  # 列举当前所有环境
conda activate AlphaEnv  # 激活环境
conda deactivate AlphaEnv  # 退出环境
conda env remove -n AlphaEnv  # 删除环境
conda remove -n AlphaEnv package_name  # 删除环境中的某个包
```

2. Jupyter

2.1 安装Jupyter

使用 `pip` 命令来安装Jupyter：

```
pip install jupyter
```

如果出现以下警告提示，说明使用的是root用户进行全局安装，安装后全局可用。如果是信任的安装包可使用该命令进行安装，未信任的安装包处于安全考虑一般建议使用 `pip3 install --user packagename` 为某一特定普通用户安装，此处略过即可。

```
WARNING: Running pip install with root privileges is generally not a good
 idea. Try `pip3 install --user` instead.
```

2.2 设置远程访问Jupyter的密码

在这一步我们要得到的是所设置的明文口令被加密过的哈希密码。

输入Python进入Python环境，输入以下内容通过自己设置的明文口令得到加密后的哈希密码。

```
>>> from notebook.auth import passwd
>>> passwd()
Enter password:...
Verify password:...
'argon2:...`
```

得到密码后复制保存下来，输入 `exit()` 退出Python环境。

2.3 生成配置文件

输入以下内容会在根目录（~/）自动生成配置文件（~/.jupyter/jupyter_notebook_config.py），配置文件在默认隐藏的文件夹内，输入 `l.` 可查看。

```
jupyter notebook --generate-config
```

2.4 写入配置信息

使用Vim修改配置文件：

```
vim ~/.jupyter/jupyter_notebook_config.py
```

输入 `i` 进入编辑模式，可以看见配置文件内的所有配置信息默认是被注释掉的，直接在文件最后写入以下配置信息。

```
# Custom Configuration file for jupyter-notebook

c.NotebookApp.ip = '*'
# 如果启动服务报错 则修改为c.NotebookApp.ip='0.0.0.0'
c.NotebookApp.password = u'argon2...' # 之前得到的哈希密码
c.NotebookApp.open_browser = False # 设置是否自动打开浏览器
c.NotebookApp.port = 6080  # 设置访问端口
```

```
c.NotebookApp.allow_remote_access = True # 设置允许远程访问
```

然后按Esc键退出编辑模式，输入 `:wq` 保存并退出。

2.5 启动远程服务

启动远程服务前先在服务器安全组上将配置信息中的设置的访问端口开启。

```
jupyter notebook --allow-root
```

使用 `Ctrl + C` 结束服务。

> Jupyter默认不允许root用户登录，可以设置强制允许。此行启动命令会一直占用一个终端窗口。

为了使窗口关闭后Jupyter服务可以继续运行，可以使用以下命令：

```
nohup jupyter notebook --allow-root &
```

终端出现以下提示信息则为正常

```
# nohup: ignoring input and appending output to 'nohup.out'
```

> `nohup` 命令作用： `no hang up` 的缩写，即不挂断，忽略所有挂断命令，并将程序输出重定向到当前目录下 `nohup.out` 文件内。
>
> `&` 命令作用；在后台运行程序。

结束服务则要使用以下命令

```
lsof -i :{端口号} # 查看当前占用该端口的所有程序。
kill -9 {PID} # 结束占用该端口的程序。
```

要使程序输出重定向至指定文件，如 `jupyter.log` ，则启动命令应为：

```
nohup jupyter notebook --allow-root > jupyter.log 2>&1 &
```

启动服务时如果出现报错，可能有以下几种情况；

1. 没有在服务器安全组中开启与配置文件相对应的访问端口。
2. 端口正在被其他程序占用。

```
lsof -i :{端口号} # 查看当前占用该端口的所有程序。
kill -9 {PID} # 结束占用该端口的程序。
```

3. 检查配置文件中IP设置。

```
c.NotebookApp.ip = '*'
```

如果使用 `*` 报错 则尝试修改为 `c.NotebookApp.ip='0.0.0.0'` 。

2.6 安装附加扩展功能

依次执行以下命令即可成功安装，安装后看不到 `nbextensions` 选项则需要重启Jupyter服务查看。

```
pip install jupyter_contrib_nbextensions
jupyter contrib nbextension install --user
pip install jupyter_nbextensions_configurator
jupyter nbextensions_configurator enable --user
```