

## 第三周金融编程与计算课程作业

1 在自己的终端(比如 GitBash、Zsh 等)配置好 Conda Init, 使得启动终端后, 在提示符(比如 \$、%)前能够看到(base)

(1) 在启动时可以运行:

```
1 # >>> conda initialize >>>
2 # !! Contents within this block are managed by 'conda init' !!
3 if [ -f '/d/86157/anaconda3/Scripts/conda.exe' ]; then
4     eval "$('/d/86157/anaconda3/Scripts/conda.exe' 'shell.bash' 'hook')"
5 fi
6 # <<< conda initialize <<<
```

(2) 重新打开会出现:

```
(base)
86157@LAPTOP-GMTRB58B MINGW64 ~
$
```

(3) 把它设置为一行:

```
(base) 86157@LAPTOP-GMTRB58B MINGW64 ~
$
```

2 使用 conda info 命令查看本机 Conda 的配置信息

```
(base) 86157@LAPTOP-GMTRB58B MINGW64 ~
$ conda info

active environment : base
active env location : D:\86157\anaconda3
shell level : 1
user config file : C:\Users\86157\.condarc
populated config files : D:\86157\anaconda3\.condarc
conda version : 24.9.2
conda-build version : 24.9.0
python version : 3.12.7.final.0
```

3 使用 conda env list 命令查看已有的 Conda 环境的名称和路径, 理解 Conda 环境的概念

```
(base) 86157@LAPTOP-GMTRB58B MINGW64 ~
$ conda env list
# conda environments:
#
base * D:\86157\anaconda3
```

理解:

(1) 定义:

Conda 环境是一个独立的空间, 在这个空间中你能够安装不同版本的软件包

及其依赖项，而不会对其他环境造成影响。可以把它想象成一个个相互隔离的房间，每个房间里都有特定的工具和软件，你可以按需进入不同的房间开展工作。

(2) 作用：

①避免依赖冲突：不同的项目可能需要不同版本的库，在同一个环境中安装这些库可能会导致冲突。借助 Conda 环境，你能够为每个项目创建独立的环境，在各个环境中安装适合该项目的库版本。

②方便项目迁移：你可以把一个环境中的所有软件包信息导出为一个文件，然后在其他机器上根据这个文件重新创建相同的环境，这样就能够确保项目在不同机器上的一致性。

(3) 常用命令：

创建环境---conda create

激活环境---conda activate

安装软件包---conda install or pip install

列出所有环境---conda env list

退出环境---conda deactivate

删除环境---conda remove

4 使用 conda create 命令创建两个 Conda 环境，一个里面安装 Python 3.12 和 requests 软件包，另一个里面安装 Python 3.9、pandas 和 statsmodels 软件包，能够在终端里切换 Conda 环境，验证 Python 和软件包的版本

(1) 创建 prj1

```
(base) 86157@LAPTOP-GMTRB58B MINGW64 ~
$ conda create -n prj1 python=3.12 requests
Channels:
- defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: D:\86157\anaconda3\envs\prj1

added / updated specs:
- python=3.12
- requests

done
#
# To activate this environment, use
#
#   $ conda activate prj1
#
# To deactivate an active environment, use
#
#   $ conda deactivate
#
(base) 86157@LAPTOP-GMTRB58B MINGW64 ~
$
```

(2) 创建 prj2

```
(base) 86157@LAPTOP-GMTRB58B MINGW64 ~
$ conda create -n prj2 python=3.9 pandas statsmodels
Channels:
- defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: D:\86157\anaconda3\envs\prj2

added / updated specs:
- pandas
- python=3.9
- statsmodels
```

```

# To activate this environment, use
#
# $ conda activate prj2
#
# To deactivate an active environment, use
#
# $ conda deactivate
#
(base) 86157@LAPTOP-GMTRB58B MINGW64 ~
$

```

(3) 查看软件包的版本:

```

>>> import pandas
>>> pandas.__file__
'D:\86157\anaconda3\envs\prj2\lib\site-packages\pandas\__init__
.py'
>>> pandas.__version__
'2.2.3'
>>> import statsmodels
>>> statsmodels.__version__
'0.14.4'
>>>

```

5 使用 conda list 命令显示 Conda 环境里的软件包列表及其版本信息:

(1) 查看 Prj2 的项目

```

86157@LAPTOP-GMTRB58B MINGW64 ~
$ conda list
# Packages in environment at D:\86157\anaconda3\envs\prj2:
# Name Version Build Channel
blas 1.0 h2bbff1b_5 mkl
bottleneck 1.0.8 h59b5b59_4032 h59b5b59
ca-certificates 2025.2.25 haad99232_0 h59b5b59
icc_rt 2025.1.0 h0b8e44_4032 h59b5b59
intel-openmp 2025.1.0 py39h8337c_50 py39h8337c
mkl-service 2.1.0 py39h8337c_50 py39h8337c
mkl_fft 1.0.1 py39h8337c_50 py39h8337c
mkl_random 1.0.1 py39h8337c_50 py39h8337c
numpy 1.26.4 py39h8337c_50 py39h8337c
numpy-base 1.26.4 py39h8337c_50 py39h8337c
openblas 0.3.21 h59b5b59_4032 h59b5b59
packaging 24.0 py39h8337c_50 py39h8337c
pandas 2.2.3 py39h8337c_50 py39h8337c
patsy 0.5.6 py39h8337c_50 py39h8337c
pip 24.0 py39h8337c_50 py39h8337c
pybind11-abi 1.0.0 h59b5b59_4032 h59b5b59
python 3.9.21 h59b5b59_4032 h59b5b59
python-dateutil 2.9.0 py39h8337c_50 py39h8337c
python-tzdata 2024.1 py39h8337c_50 py39h8337c
pytz 2024.1 py39h8337c_50 py39h8337c

```

(2) 查看 Prj1 的项目

```

86157@LAPTOP-GMTRB58B MINGW64 ~
$ conda list
# Packages in environment at D:\86157\anaconda3\envs\prj1:
# Name Version Build Channel
bottleneck 1.0.9 py312h5da7b33_9 py312h5da7b33
ca-certificates 2025.2.25 haad99232_0 haad99232
charset-normalizer 3.3.2 pyhd3eb1b0_0 pyhd3eb1b0
cymem 2.0.4 h59b5b59_4032 h59b5b59
filelock 3.14.4 py312h5da7b33_9 py312h5da7b33
freetype 2.13.1 h59b5b59_4032 h59b5b59
fonttools 4.53.0 py312h5da7b33_9 py312h5da7b33
ipython 8.12.0 py312h5da7b33_9 py312h5da7b33
jinja2 3.1.2 py312h5da7b33_9 py312h5da7b33
matplotlib 3.8.0 py312h5da7b33_9 py312h5da7b33
numpy 1.26.4 py312h5da7b33_9 py312h5da7b33
openblas 0.3.21 h59b5b59_4032 h59b5b59
packaging 24.0 py312h5da7b33_9 py312h5da7b33
pandas 2.2.3 py312h5da7b33_9 py312h5da7b33
patsy 0.5.6 py312h5da7b33_9 py312h5da7b33
pip 24.0 py312h5da7b33_9 py312h5da7b33
pybind11-abi 1.0.0 h59b5b59_4032 h59b5b59
python 3.12.0 py312h5da7b33_9 py312h5da7b33
python-dateutil 2.9.0 py312h5da7b33_9 py312h5da7b33
python-tzdata 2024.1 py312h5da7b33_9 py312h5da7b33
pytz 2024.1 py312h5da7b33_9 py312h5da7b33

```

6 使用 conda install 命令往 Conda 环境里安装更多的软件包, 并验证版本

(1) 安装 ipython

```

86157@LAPTOP-GMTRB58B MINGW64 ~
$ conda install ipython
Channels:
- defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: D:\86157\anaconda3\envs\prj1
added / updated specs:
- ipython

```

(2) 中间是版本, 版本很重要!

#	Name	Version	Build	Channel
1				
	brotli-python	1.0.9	py312h5da7b33_9	
	bzip2	1.0.8	h2bbff1b_6	
	ca-certificates	2025.2.25	haa95532_0	
	certifi	2025.1.31	py312haa95532_0	
	charset-normalizer	3.3.2	pyhd3eb1b0_0	
	curl	8.11.0	h8d14728_0	

7 根据文档，配置 Anaconda 清华镜像，加快 conda install 安装软件包的速度，将 conda-forge 设置为默认 Channel，让 conda install 能够安装更多的软件包。

(1) 生成创建名为 .condarc 的文件：

```
! .condarc
1 channels:
2   - https://repo.anaconda.com/pkgs/main
3   - https://repo.anaconda.com/pkgs/r
4   - https://repo.anaconda.com/pkgs/msys2
5 show_channel_urls: true
6
```

换成下图这个

```
! .condarc
1 channels:
2   - conda-forge
3   - defaults
4 show_channel_urls: true
5 default_channels:
6   - https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main
7   - https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/r
8   - https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/msys2
9 custom_channels:
10  conda-forge: https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud
11  pytorch: https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud
12 channel_priority: strict
13
```

(2) 理解 conda-forge:

①核心概念:

conda 是一个跨平台的包和环境管理系统，可用于安装、运行和更新软件包及其依赖项。而 conda-forge 就是基于 conda 的一个特定软件包源，它包含了大量由社区贡献和维护的软件包。

②主要特点:

**社区驱动:** conda-forge 由全球范围内的开发者社区共同维护和贡献。这种社区驱动的模式使得它能够快速响应新的软件需求和更新，软件包的更新频率通常较高，能及时提供最新版本的软件。

**多平台支持:** conda-forge 支持多种操作系统，包括 Windows、macOS 和 Linux，确保了软件包在不同平台上的兼容性和可用性。

**丰富的软件包:** 仓库中涵盖了各种领域的软件包，如科学计算、数据科学、机器学习、深度学习等。像 numpy、pandas、scikit-learn 等常见的科学计算库在 conda-forge 上都能找到。

**严格的构建和测试流程:** 为了保证软件包的质量和稳定性，conda-forge 有一套严格的构建和测试流程。每个软件包在被添加到仓库之前，都要经过自动化测试，以确保其不同平台上都能正常工作。

是一个由社区驱动的、用于 conda 包管理器的第三方软件包仓库。

(3) 配置 Anaconda 清华镜像后，安装 polars

```
$ conda install polars
Channels:
- conda-forge
- defaults
- https://repo.anaconda.com/pkgs/main
- https://repo.anaconda.com/pkgs/r
- https://repo.anaconda.com/pkgs/msys2
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done
```

注意：去除软件包，不要删除，可能会造成更大的瘫痪，但是可以把整个环境全部删掉，然后再继续安装

8 使用 `pip install` 命令往 Conda 环境里安装 Python 软件包，并验证版本

```
86157@LAPTOP-GMTRB58B MINGW64 ~
$ pip install tushare
Collecting tushare
  Downloading tushare-1.4.19-py3-none-any.whl.met
Collecting pandas (from tushare)
  Downloading pandas-2.2.3-cp312-cp312-win_amd64.
Requirement already satisfied: requests in d:\861
```

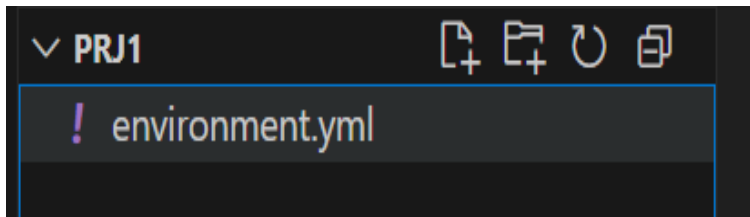
9 根据 文档 配置 PyPI 清华镜像，加快 `pip install` 安装软件包的速度

```
86157@LAPTOP-GMTRB58B MINGW64 ~
$ pip install tushare
Looking in indexes: https://mirrors.tuna.tsinghua.edu.cn/pypi/web/simple
Collecting tushare
  Downloading https://mirrors.tuna.tsinghua.edu.cn/pypi/web/packages/2b/b1/8aa06e9
3150a691f71edc1/tushare-1.4.19-py3-none-any.whl (142 kB)
Collecting pandas (from tushare)
```

10 能够导出 `environment.yml` Conda 环境配置文件，能够删除 Conda 环境，能够用 `environment.yml` 配置文件重建 Conda 环境

(1) 导出可以保存和分享，备份：

```
86157@LAPTOP-GMTRB58B MINGW64 ~
$ conda env export -f environment.yml
(prj1)
86157@LAPTOP-GMTRB58B MINGW64 ~
```



(2) 重建 Conda 环境，重新出现 `prj1`

```

86157@LAPTOP-QMTRB58B MINGW64 ~
$ conda deactivate
(base)
86157@LAPTOP-QMTRB58B MINGW64 ~
$ conda env list
# conda environments:
#
base                  * D:\86157\anaconda3
prj1                  D:\86157\anaconda3\envs\prj1
prj2                  D:\86157\anaconda3\envs\prj2
(base)
86157@LAPTOP-QMTRB58B MINGW64 ~
$ conda env remove -n prj1
Remove all packages in environment D:\86157\anaconda3\envs\prj1:
## Package Plan ##

```

## 11 理解 Conda 与 Python 的关系，理解 Conda-Forge 与 Conda 的关系，理解 Python 解释器、第三方软件包、PyPI 软件仓库、以及程序/软件包的路径问题

### (1) 理解 Conda 与 Python 的关系

Conda 是一个跨平台的开源软件包管理系统和环境管理系统，可用于安装、更新和卸载各类软件包，同时能创建和管理不同的运行环境。Python 是一种广泛使用的高级编程语言。

Conda 和 Python 的关系在于，Conda 可对 Python 进行管理。你能借助 Conda 创建不同 Python 版本的虚拟环境，在这些环境里安装和管理 Python 软件包。例如，你既能创建一个使用 Python 3.7 的环境，也能创建一个使用 Python 3.9 的环境，不同环境相互独立，互不干扰。

### (2) 理解 Conda - Forge 与 Conda 的关系

Conda - Forge 是一个社区驱动的 Conda 软件包仓库，它提供了大量的软件包供 Conda 使用。Conda 默认的软件包仓库是 Anaconda 仓库，不过 Conda - Forge 提供了更多的软件包选择，而且更新速度更快。你可以把 Conda - Forge 添加到 Conda 的软件包仓库列表中，这样就能使 Conda 从 Conda - Forge 安装软件包了。

### (3) 理解 Python 解释器、第三方软件包、PyPI 软件仓库

**Python 解释器：**它是一个程序，用于执行 Python 代码。当你编写好 Python 代码后，Python 解释器会读取代码并将其转换为计算机能够理解的指令。常见的 Python 解释器有 CPython（官方解释器）、Jython（运行在 Java 虚拟机上的 Python 解释器）等。

**第三方软件包：**是由 Python 社区开发者编写的额外软件包，能帮助你更高效地完成各种任务。例如，numpy 可用于科学计算，pandas 可用于数据处理和分析。

**PyPI 软件仓库：**Python Package Index（Python 包索引），是 Python 官方的软件包仓库，存储了大量的第三方 Python 软件包。你可以使用 pip（Python 的包管理工具）从 PyPI 下载和安装软件包。

### (4) 程序/软件包的路径问题

在 Python 中，程序和软件包的路径至关重要，因为 Python 解释器需要知道在哪里找到这些文件。

**sys.path：**Python 解释器在查找模块时会参考 sys.path 列表。该列表包含了一系列目录，Python 会按顺序在这些目录中查找所需的模块。你可以通过以下代码查看 sys.path 的内容：

☰ ☷ ☶ ☵ ☴ ☳ ☲ ☱

总结来说，Conda 可管理 Python 环境和软件包，Conda - Forge 为 Conda 提供更多软件包选择，Python 解释器执行 Python 代码，第三方软件包可扩展 Python 功能，PyPI 是官方软件包仓库，而程序和软件包的路径问题需要通过 sys.path 和虚拟环境路径来管理。

### (1) 先生成一个配置文件

The screenshot shows the RRPP application interface. On the left, the '打开的编辑器' (Opened Editors) panel lists the following editors:

- 欢迎 (Welcome)
- environment.yml
- RRPP
  - myproject
  - environment.yml

On the right, the content of the selected 'environment.yml' file is displayed:

```
1 name: myproject
2 channels:
3   - conda-forge
4 dependencies:
5   - python=3.12
```

```
(base) 86157@LAPTOP-GMTRB58B MINGW64 /d/86157/rrpp/myproject
$ ped
bash: ped: command not found

done

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

```
(base) 86157@LAPTOP-GMTRB58B MINGW64 /d/86157/rrpp/myproject
$ conda activate myproject
(myproject)
86157@LAPTOP-GMTRB58B MINGW64 /d/86157/rrpp/myproject
$ conda list
# packages in environment at D:\86157\anaconda3\envs\myproject:
#
# Name                               Version                               Build      Chann
el
bzip2                                1.0.8                                h2466b09_7  con
da-forge
ca-certificates                      2025.1.31                             h56e8100_0  con
da-forge
```



(4) 创建简单程序

```
(myproject)
86157@LAPTOP-GMTRB58B MINGW64 /d/86157/rrpp/myproject
$ python main.py
Hello, conda!
(myproject)
```

(5) 加入 pandas，之后用 `conda env update` 命令更新，这样就可以运行成功

```
- python=3.12
- pandas
```

```
86157@LAPTOP-GMTRB58B MINGW64 /d/86157/rrpp/myproject
$ python main.py
Hello, conda!
(myproject)
```

(6) 在 python 加入几行，然后在输出，就会看到不同的结果

```
def main():
    print("Hello, conda!")
    print(pd.__version__)
    print(pd.__file__)
    wheel = True
    _name_ = str
    if __name__ == "__main__":
        main()

da-forge 14.3 hbf610ac_24
da-forge 14.42.34438 hfd919c2_24
da-forge 0.45.1 pyhd8ed1ab_1
(myproject)
86157@LAPTOP-GMTRB58B MINGW64 /d/86157/rrpp/myproject
$ python main.py
Hello, conda!
Traceback (most recent call last):
  File "D:\86157\rrpp\myproject\main.py", line 11, in <module>
    main()
  File "D:\86157\rrpp\myproject\main.py", line 6, in main
    print(pd.__version__)
    ~~~~~^
AttributeError: module 'pandas' has no attribute '__version__'
d you mean: '__version__'?
(myproject)
```

(7) 按照教程，安装完网址内容之后，扩展 rainbow.csv 就会变成彩色

```
1 OBJECTID,GEOID10,GEOID20,STATEFP,COUNTYFP,TRACTCE,BLKGRPCE,CSA,CSA_Name,CBSA,CBSA_Name,CB
2 1,4,8113E+11,4,8113E+11,48,113,7825,4,206,"Dallas-Fort Worth, TX-OK",19100,"Dallas-Fort w
3 2,4,8113E+11,4,8113E+11,48,113,7825,3,206,"Dallas-Fort Worth, TX-OK",19100,"Dallas-Fort w
4 3,4,8113E+11,4,8113E+11,48,113,7825,3,206,"Dallas-Fort Worth, TX-OK",19100,"Dallas-Fort w
5 4,4,8113E+11,4,8113E+11,48,113,7824,1,206,"Dallas-Fort Worth, TX-OK",19100,"Dallas-Fort w
6 5,4,8113E+11,4,8113E+11,48,113,7824,2,206,"Dallas-Fort Worth, TX-OK",19100,"Dallas-Fort w
7 6,4,8113E+11,4,8113E+11,48,113,7827,1,206,"Dallas-Fort Worth, TX-OK",19100,"Dallas-Fort w
8 7,4,8113E+11,4,8113E+11,48,113,9201,2,206,"Dallas-Fort Worth, TX-OK",19100,"Dallas-Fort w
9 8,4,8113E+11,4,8113E+11,48,113,1102,2,206,"Dallas-Fort Worth, TX-OK",19100,"Dallas-Fort w
10 9,4,8113E+11,4,8113E+11,48,113,11401,2,206,"Dallas-Fort Worth, TX-OK",19100,"Dallas-Fort
11 10,4,8113E+11,4,8113E+11,48,113,11401,3,206,"Dallas-Fort Worth, TX-OK",19100,"Dallas-Fort
12 11,4,8113E+11,4,8113E+11,48,113,11500,2,206,"Dallas-Fort Worth, TX-OK",19100,"Dallas-Fort
13 12,4,8113E+11,4,8113E+11,48,113,11500,1,206,"Dallas-Fort Worth, TX-OK",19100,"Dallas-Fort
14 13,4,8113E+11,4,8113E+11,48,113,12301,2,206,"Dallas-Fort Worth, TX-OK",19100,"Dallas-Fort
15 14,4,8113E+11,4,8113E+11,48,113,12301,3,206,"Dallas-Fort Worth, TX-OK",19100,"Dallas-Fort
16 15,4,8329E+11,4,8329E+11,48,329,304,3,372,"Midland-Odessa, TX",33260,"Midland, TX",169800
17 16,4,8329E+11,4,8329E+11,48,329,305,2,372,"Midland-Odessa, TX",33260,"Midland, TX",169800
```

(8) 输出结果，解读此代码，也可以将代码问大模型

```
86157@LAPTOP-GMTRB58B MINGW64 /d/86157/rrpp/myproject
$ python main.py
7.45% of U.S. residents live in highlywalkable neighborhoods.
```