Final Reports - MATH620152

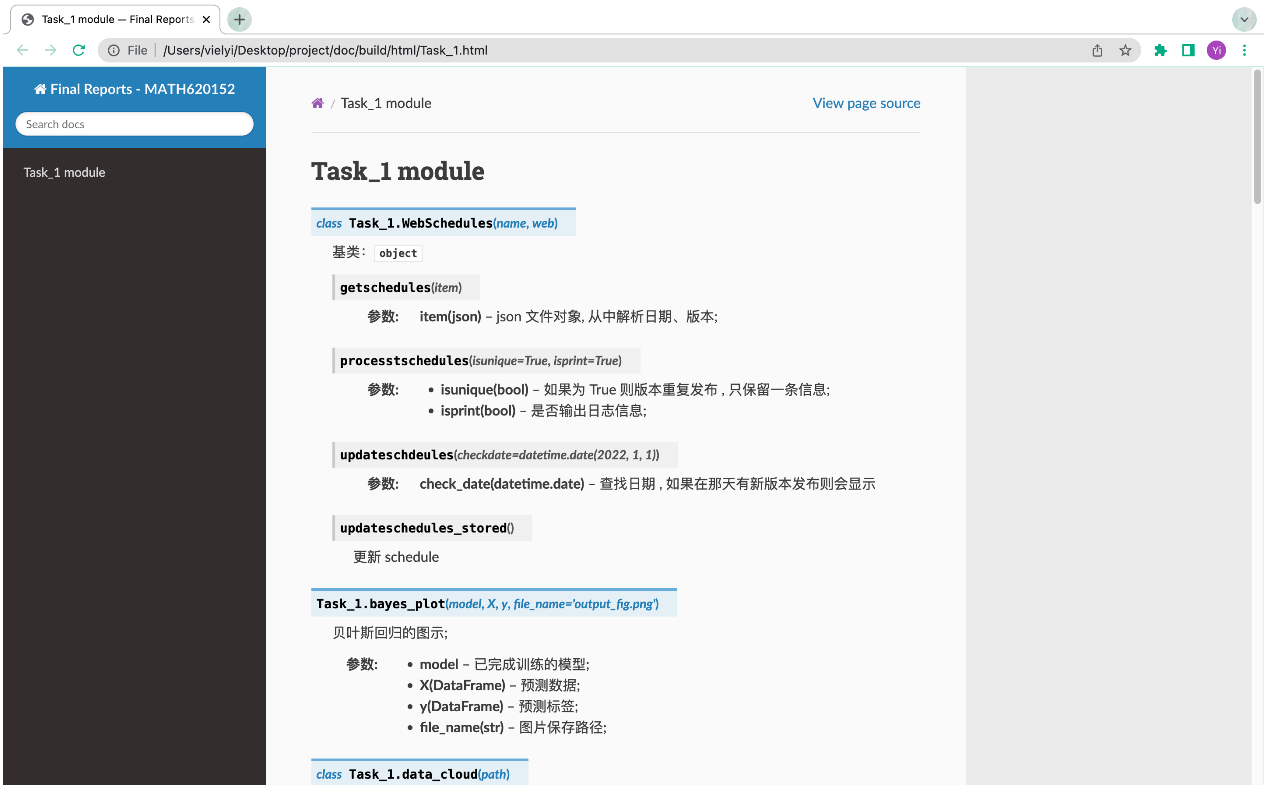
- Author: 吴羿 22210180064

主要程序文件

1. Notebook 程序文件结构：
    - `Task-1.ipynb` 包含：4 个类，18 个各类知识点汇总；
    - `Task-2.ipynb` 包含：2 个类及 `Seaborn` 绘图， 8 个各类知识点汇总；
    - `Task-3.ipynb` 包含：基于Pytorch实现的全连接神经网络，4 个各类知识点汇总；
    - `Task-4.ipynb` 包含：基于gurobipy实现的神经网络，2 个各类知识点汇总；
    - `R_stratup.R` 包含：基于 R 的初学者级的应用
2. 分列的 `torchex_main.py` 以及 `AsData.py` 文件
    - `AsData.py` 提供 Pytorch 的 Dateset 定义模块
    - `torchex_main.py` 方便命令行调用
3. 文件夹 doc 提供了由 **syphinx** （司芬克斯）自动生成并建立的程序说明文档

    syphinx项目地址：https://www.sphinx-doc.org/en/master/

    - 在 `./doc/build/html` 文件夹下，提供了`Task-1.ipynb` 和`Task-2.ipynb` 中共计6个类的说明文档
    - 检阅该HTML可以得到项目的完整说明（该文档提供搜索、目录等各类功能）


Task_1 module — Final Reports
File
/Users/vielyi/Desktop/project/doc/build/html/Task_1.html
Final Reports - MATH620152
Search docs
Task_1 module
Task_1 module
View page source

Task_1 module

class **Task_1.WebSchedules**(*name, web*)

基类： `object`

getschedules(*item*)

参数: **item(json)** – json 文件对象, 从中解析日期、版本;

processtschedules(*isunique=True, isprint=True*)

参数:
- **isunique(bool)** – 如果为 True 则版本重复发布，只保留一条信息;
- **isprint(bool)** – 是否输出日志信息;

updateschdeules(*checkdate=datetime.date(2022, 1, 1)*)

参数: **check_date(datetime.date)** – 查找日期，如果在那天有新版本发布则会显示

updateschedules_stored()

更新 schedule

Task_1.bayes_plot(*model, X, y, file_name='output_fig.png'*)

贝叶斯回归的图示;

参数:
- **model** – 已完成训练的模型;
- **X(DataFrame)** – 预测数据;
- **y(DataFrame)** – 预测标签;
- **file_name(str)** – 图片保存路径;

class **Task_1.data_cloud**(*path*)

    - 在 `./doc/code` 路径下可以找到这两个包含所有类的 `.py` 文件

支持文件

支持文件目录为 `./support` 和 `./output`

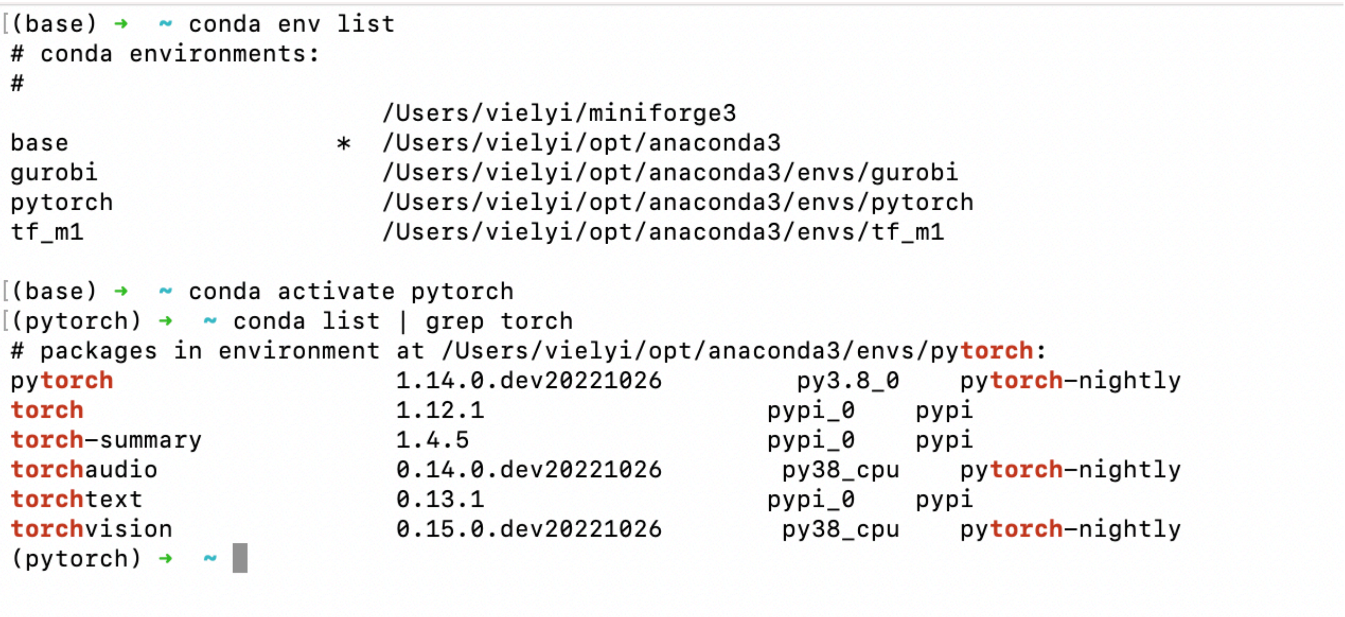
- `./output` 是程序文件默认的输出目录
- `./support` 主要包含两部分内容：**data_cloud** （词云类的必要支持文件）和 **MD_pic** （Markdown文档的图片路径）

文档汇总

所有的文档文件已经汇总到目录 `./PDFandHTML` 文件夹下

环境说明

由于使用了 **Pytorch** ，非常有必要在此声明所使用的环境



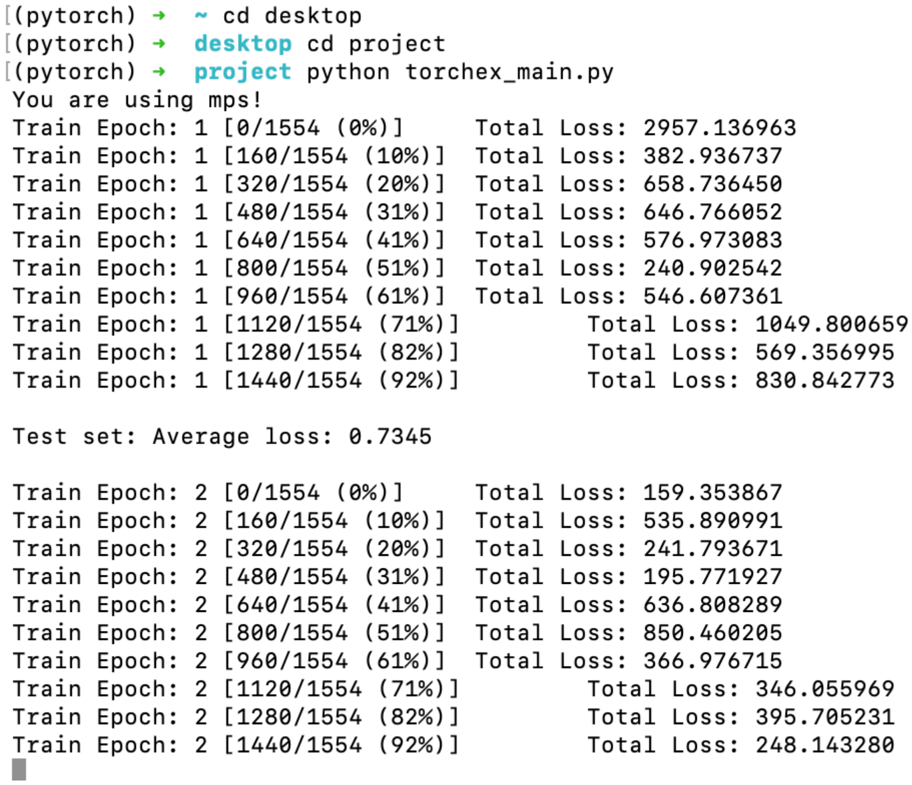
```
[(base) ➜  ~ conda env list
# conda environments:
#
                         /Users/vielyi/miniforge3
base                  *  /Users/vielyi/opt/anaconda3
gurobi                   /Users/vielyi/opt/anaconda3/envs/gurobi
pytorch                  /Users/vielyi/opt/anaconda3/envs/pytorch
tf_m1                    /Users/vielyi/opt/anaconda3/envs/tf_m1

[(base) ➜  ~ conda activate pytorch
[(pytorch) ➜  ~ conda list | grep torch
# packages in environment at /Users/vielyi/opt/anaconda3/envs/pytorch:
pytorch                   1.14.0.dev20221026         py3.8_0    pytorch-nightly
torch                     1.12.1                   pypi_0    pypi
torch-summary             1.4.5                    pypi_0    pypi
torchaudio                0.14.0.dev20221026        py38_cpu    pytorch-nightly
torchtext                 0.13.1                   pypi_0    pypi
torchvision               0.15.0.dev20221026        py38_cpu    pytorch-nightly
(pytorch) ➜  ~ █
```

该版本的 **Pytorch** （nightly）为 MacOS M1架构进行设计，程序中 `torch.device("mps" if use_cuda else "cpu")` 是在 MacOS 中启用 GPU 的标准格式，如果报错，请确认GPU空间充足或者直接使用CPU进行训练；

在 `torchex_main.py` 中使用了 `argparse` 库以保证可以以更加标准和规范的方式使用 **Pytorch**

- 你可以在命令行中用如下命令以默认参数的方式运行该程序

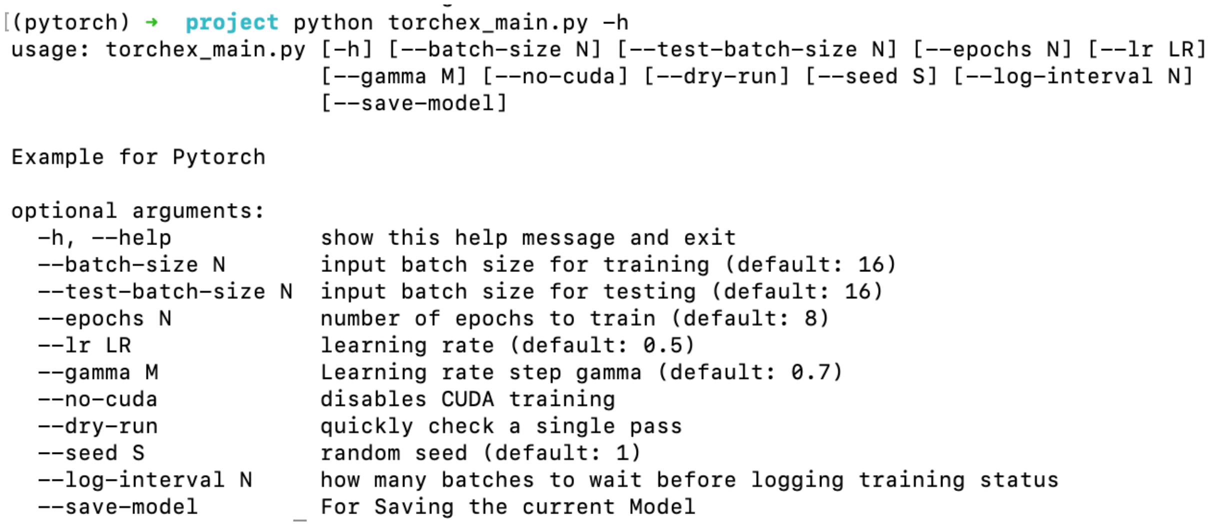


```
[(pytorch) ➜  ~ cd desktop
[(pytorch) ➜  desktop cd project
[(pytorch) ➜  project python torchex_main.py
 You are using mps!
 Train Epoch: 1 [0/1554 (0%)]     Total Loss: 2957.136963
 Train Epoch: 1 [160/1554 (10%)]  Total Loss: 382.936737
 Train Epoch: 1 [320/1554 (20%)]  Total Loss: 658.736450
 Train Epoch: 1 [480/1554 (31%)]  Total Loss: 646.766052
 Train Epoch: 1 [640/1554 (41%)]  Total Loss: 576.973083
 Train Epoch: 1 [800/1554 (51%)]  Total Loss: 240.902542
 Train Epoch: 1 [960/1554 (61%)]  Total Loss: 546.607361
 Train Epoch: 1 [1120/1554 (71%)]         Total Loss: 1049.800659
 Train Epoch: 1 [1280/1554 (82%)]         Total Loss: 569.356995
 Train Epoch: 1 [1440/1554 (92%)]         Total Loss: 830.842773

 Test set: Average loss: 0.7345

 Train Epoch: 2 [0/1554 (0%)]     Total Loss: 159.353867
 Train Epoch: 2 [160/1554 (10%)]  Total Loss: 535.890991
 Train Epoch: 2 [320/1554 (20%)]  Total Loss: 241.793671
 Train Epoch: 2 [480/1554 (31%)]  Total Loss: 195.771927
 Train Epoch: 2 [640/1554 (41%)]  Total Loss: 636.808289
 Train Epoch: 2 [800/1554 (51%)]  Total Loss: 850.460205
 Train Epoch: 2 [960/1554 (61%)]  Total Loss: 366.976715
 Train Epoch: 2 [1120/1554 (71%)]         Total Loss: 346.055969
 Train Epoch: 2 [1280/1554 (82%)]         Total Loss: 395.705231
 Train Epoch: 2 [1440/1554 (92%)]         Total Loss: 248.143280
```

- 关于参数的说明也可以查看：



```
[(pytorch) ➜  project python torchex_main.py -h
usage: torchex_main.py [-h] [--batch-size N] [--test-batch-size N] [--epochs N] [--lr LR]
                       [--gamma M] [--no-cuda] [--dry-run] [--seed S] [--log-interval N]
                       [--save-model]

Example for Pytorch

optional arguments:
  -h, --help           show this help message and exit
  --batch-size N       input batch size for training (default: 16)
  --test-batch-size N  input batch size for testing (default: 16)
  --epochs N           number of epochs to train (default: 8)
  --lr LR              learning rate (default: 0.5)
  --gamma M            Learning rate step gamma (default: 0.7)
  --no-cuda            disables CUDA training
  --dry-run            quickly check a single pass
  --seed S             random seed (default: 1)
  --log-interval N     how many batches to wait before logging training status
  --save-model         For Saving the current Model
```