Final Reports - MATH620152

Author: 吴羿 22210180064

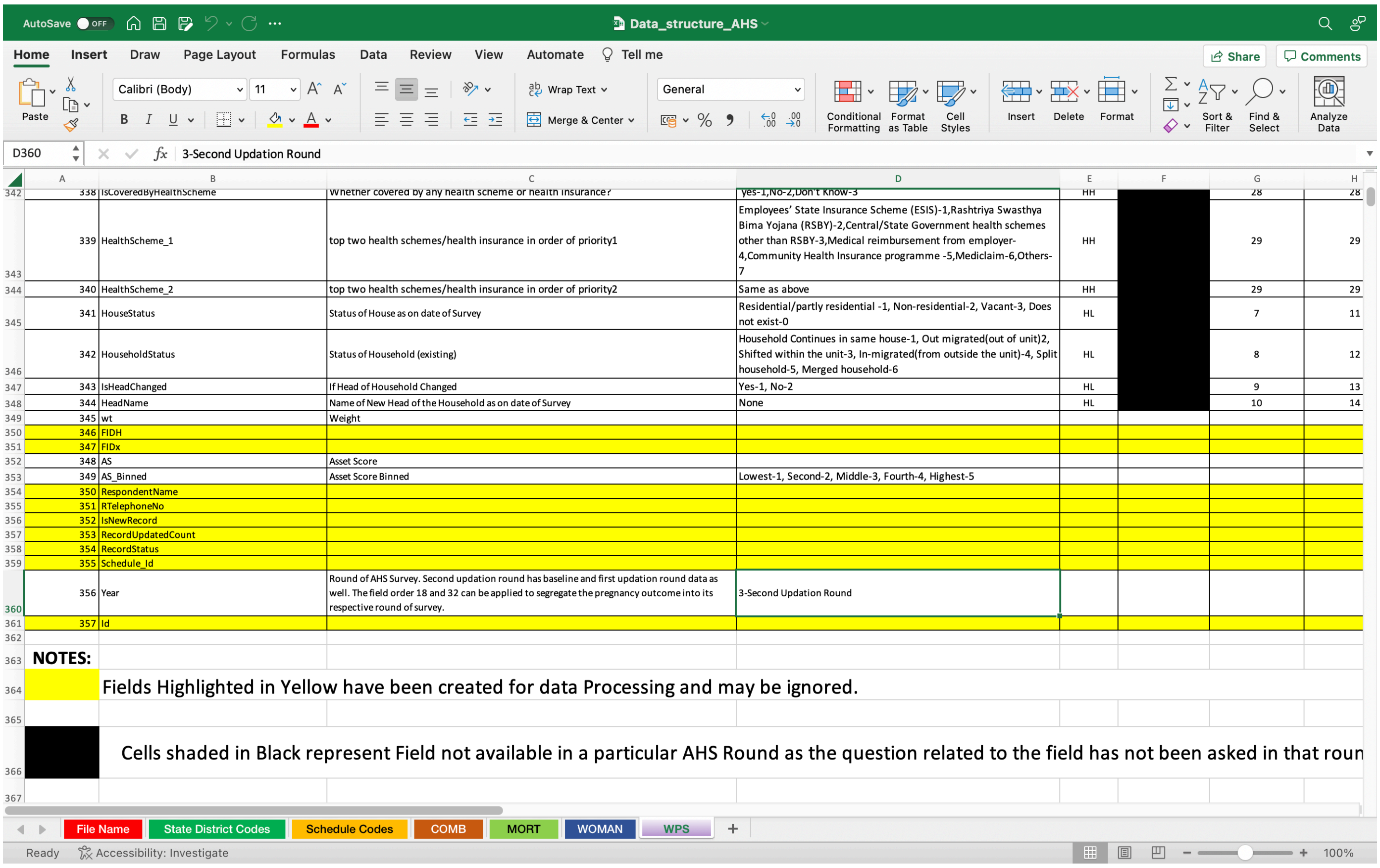
总览

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** 自动生成并建立的程序说明文档（仅限于 python）
4. 文件夹 `__pycache__` 是有 python 运行中生成的缓存文件夹，可以忽略；

Task_1

classTask_1.data_pd(path)

基于源数据`Data_structure_AHS.xlsx`



D360 3-Second Updation Round

	A	B	C	D	E	F	G	H
342	338	IsCoveredByHealthScheme	Whether covered by any health scheme or health insurance?	yes-1,No-2,Don't Know-3	HH	■	28	28
343	339	HealthScheme_1	top two health schemes/health insurance in order of priority1	Employees' State Insurance Scheme (ESIS)-1,Rashtriya Swasthya Bima Yojana (RSBY)-2,Central/State Government health schemes other than RSBY-3,Medical reimbursement from employer-4,Community Health Insurance programme -5,Mediclaim-6,Others-7	HH	■	29	29
344	340	HealthScheme_2	top two health schemes/health insurance in order of priority2	Same as above	HH	■	29	29
345	341	HouseStatus	Status of House as on date of Survey	Residential/partly residential -1, Non-residential-2, Vacant-3, Does not exist-0	HL	■	7	11
346	342	HouseholdStatus	Status of Household (existing)	Household Continues in same house-1, Out migrated(out of unit)2, Shifted within the unit-3, In-migrated(from outside the unit)-4, Split household-5, Merged household-6	HL	■	8	12
347	343	IsHeadChanged	If Head of Household Changed	Yes-1, No-2	HL	■	9	13
348	344	HeadName	Name of New Head of the Household as on date of Survey	None	HL	■	10	14
349	345	wt	Weight					
350	346	FIDH						
351	347	FIDx						
352	348	AS	Asset Score					
353	349	AS_Binned	Asset Score Binned	Lowest-1, Second-2, Middle-3, Fourth-4, Highest-5				
354	350	RespondentName						
355	351	RTelephoneNo						
356	352	IsNewRecord						
357	353	RecordUpdatedCount						
358	354	RecordStatus						
359	355	Schedule_Id						
360	356	Year	Round of AHS Survey. Second updation round has baseline and first updation round data as well. The field order 18 and 32 can be applied to segregate the pregnancy outcome into its respective round of survey.	3-Second Updation Round				
361	357	Id						
362								
363	NOTES:							
364	🟨	Fields Highlighted in Yellow have been created for data Processing and may be ignored.						
365								
366	■	Cells shaded in Black represent Field not available in a particular AHS Round as the question related to the field has not been asked in that roun						
367								

- 实现方法：
  1. 多表数据读入，多级索引结构导入
  2. 表的基础数据清洗 (NA处理、删除非必要数据)
  3. 数据按条件筛选 (`iloc` mask)：处理原表中黄色和黑色的mask逻辑
  4. 数据框整合 (merge连接)
  5. 数据查询，输出保存和简单变形 (两种 `pivot`)

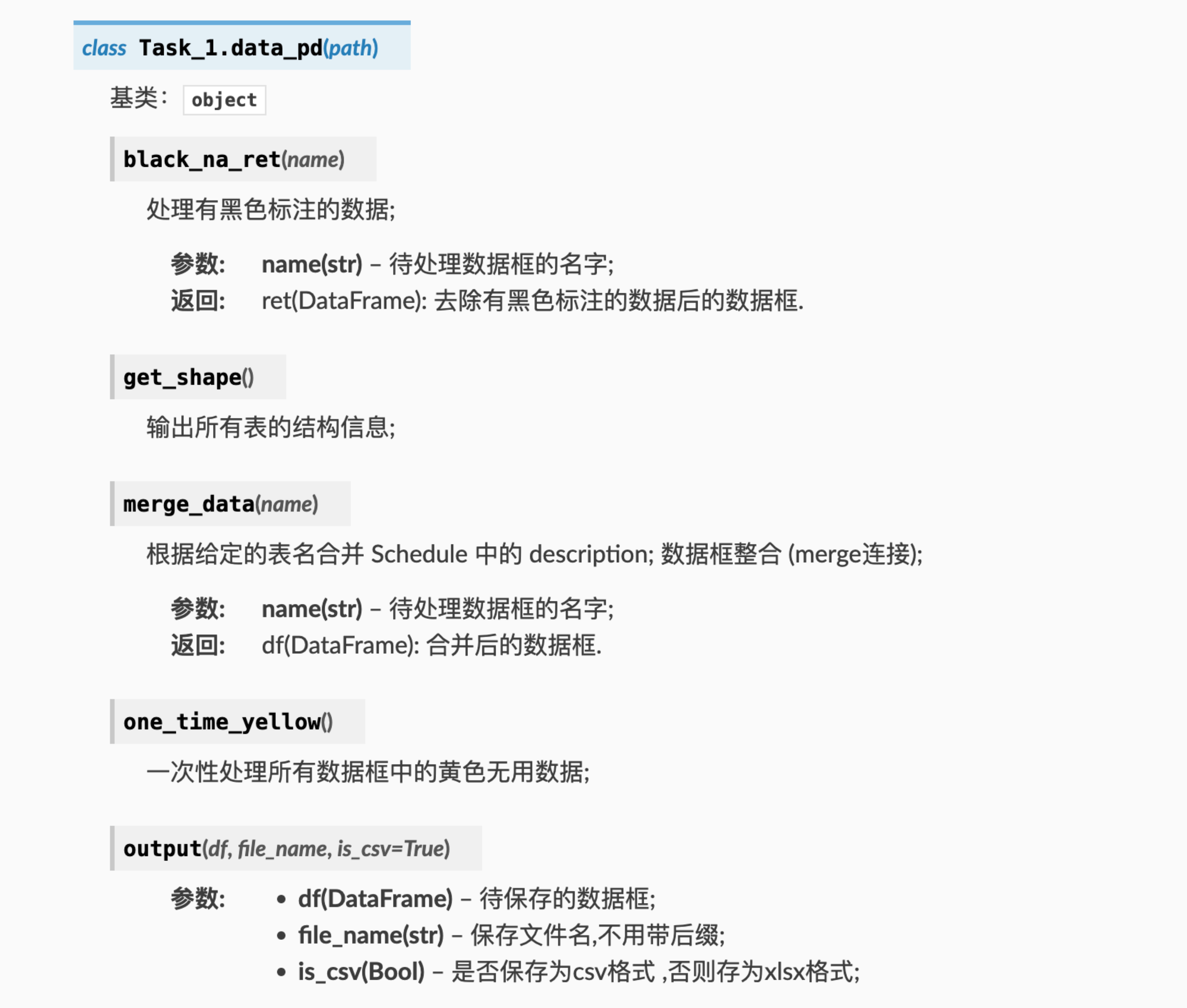
```python
if __name__ == '__main__':
    data = data_pd("./2022期末数据/Data_structure_AHS.xlsx")
    # 读取数据并作基础结构化调整
    data.read_data_all()
    # 清洗数据
    data.one_time_yellow()
    ret = data.black_na_ret('COMB')
    # 数据展示
    # data.show_data(data.data_all['COMB'])
    # 数据查询
    data.search_district(10,2)
    data.search_district(10,40) # not found

    # 数据合并
    df = data.merge_data("ABC") # 报错 Undefined dataframe!
    df = data.merge_data('WOMAN')
    # 输出基本表结构信息
    data.get_shape()
    # 数据保存
    data.output(df, './output/output',False)
```

```
## 输出
You are using pandas:1.4.2 and numpy:1.21.5
Nice to see you! This project is designed as Pandas Class 1.
There are 4 other sheets parsed in xlsx and they are:['COMB', 'MORT', 'WOMAN', 'WPS'].
============================================================
There are 20 data can be ignored since they are marked yellow.
There are 21 data can be ignored since they are marked yellow.
There are 45 data can be ignored since they are marked yellow.
There are 38 data can be ignored since they are marked black.
============================================================
State Code 10 District Code 2 is PURBA CHAMPARAN.
============================================================
ID not found!
============================================================
Undefined dataframe!
============================================================
Merge Succeed!
============================================================
Sheet COMB's shape is (83, 7)
Sheet MORT's shape is (104, 7)
Sheet WOMAN's shape is (158, 7)
Sheet WPS's shape is (324, 7)
```

- 附加：长宽表变形

```
df = df.fillna(method='ffill')
df.pivot(index =["Field Name"],columns = ["Schedule Code"],values = ["First Updation"])
df.pivot_table(index =["Field Name"],columns = ["Schedule Code"],
               values = ["Description"], aggfunc = "count")
```

- 说明文档（部分）



class **Task_1.data_pd**(*path*)

基类： `object`

black_na_ret(*name*)

处理有黑色标注的数据;

参数: **name(str)** – 待处理数据框的名字;
返回: ret(DataFrame): 去除有黑色标注的数据后的数据框.

get_shape()

输出所有表的结构信息;

merge_data(*name*)

根据给定的表名合并 Schedule 中的 description; 数据框整合 (merge连接);

参数: **name(str)** – 待处理数据框的名字;
返回: df(DataFrame): 合并后的数据框.

one_time_yellow()

一次性处理所有数据框中的黄色无用数据;

output(*df, file_name, is_csv=True*)

参数:
- **df(DataFrame)** – 待保存的数据框;
- **file_name(str)** – 保存文件名,不用带后缀;
- **is_csv(Bool)** – 是否保存为csv格式 ,否则存为xlsx格式;

classTask_1.data_cloud(path)

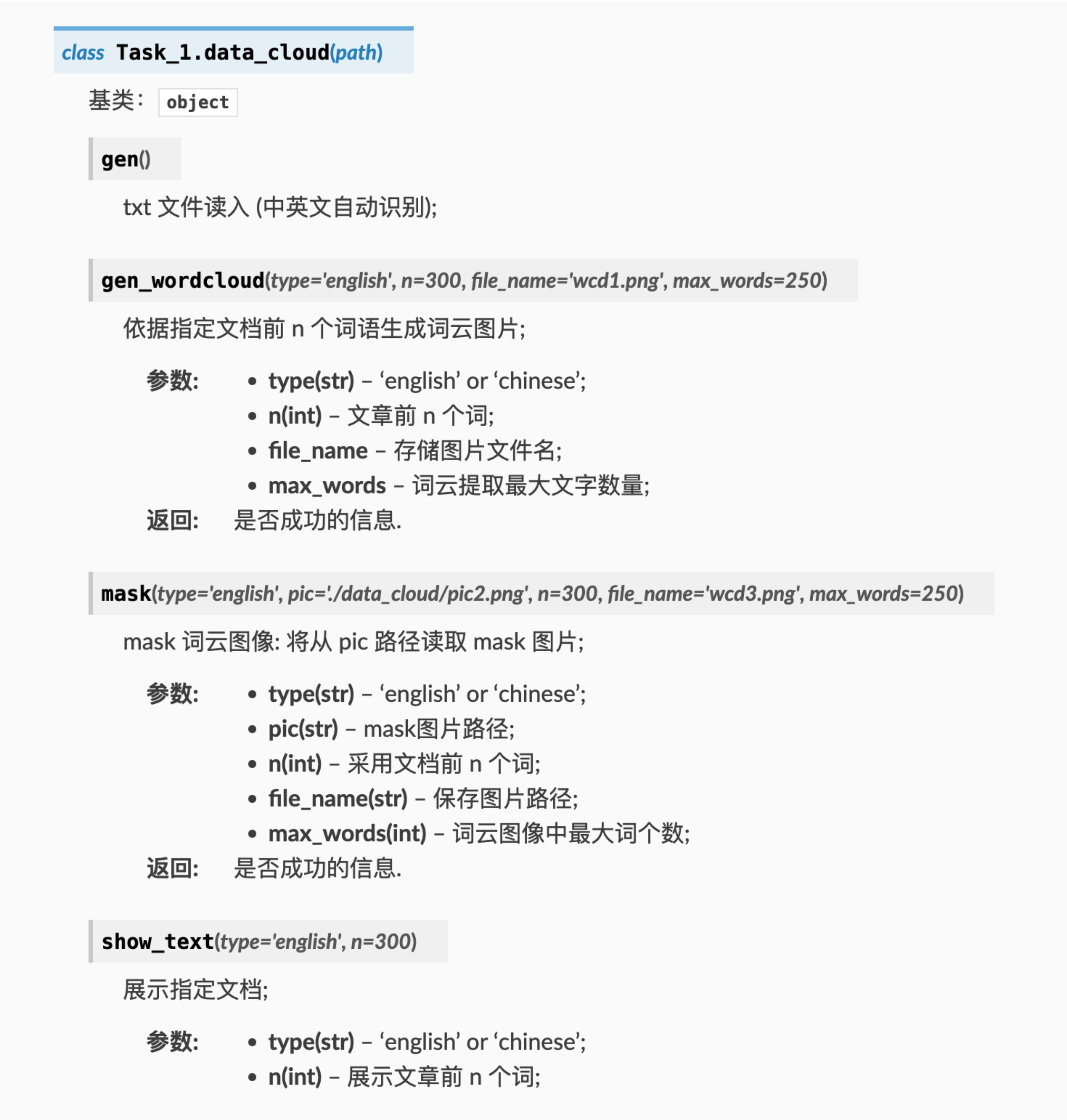
基于源数据 `小说.txt`

- 实现方法
  1. txt 文件读入 (中英文自动识别)
  2. jieba分词（在词处理过程中使用）
  3. 中英文基础词云图片展示
  4. 中英文mask-词云图片展示
  5. 中文使用过滤停词库的词云图展示

```
if __name__ == '__main__':
    data = data_cloud('./2022期末数据/小说.txt')

    # 展示英文文字前300个
    data.show_text()

    # 分别输出中英文词云图
    data.gen_wordcloud('english',n = -1,file_name = "./output/wcd1.png")
    data.gen_wordcloud('chinese',n = 20000,file_name = "./output/wcd2.png")

    # 输出中文有 mask 的词云图
    data.mask(type = 'chinese', n = 10000, file_name = "./output/wcd3.png")

    # 输出中文有 mask 的过滤停词库的词云图
    data.stop_words_mask(n = 10000, file_name = "./output/wcd4.png")
```

```
## 输出
You are using jieba:0.42.1.
Nice to see you! This project is designed as WorldCloud Class.
File Closed!
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
The Wolf and the Lamb WOLF, meeting  with a  Lamb astray  from the  fold, resolved
 not to lay violent hands on him, but to find some plea  to justify to the Lamb the
Wolf's right  to eat him.  He thus  addressed him: "Sirrah, last  year you grossly
insulted  me." "Indeed," bleated the Lamb in a mou
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
The txt's length is 193374.
Building prefix dict from the default dictionary ...
Loading model from cache
/var/folders/zz/c_1__h650c74spfy3w9n5_4r0000gn/T/jieba.cache
Loading model cost 0.495 seconds.
Prefix dict has been built successfully.
```

- 说明文档（部分）



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

基类： `object`

gen()

txt 文件读入 (中英文自动识别);

gen_wordcloud(*type='english', n=300, file_name='wcd1.png', max_words=250*)

依据指定文档前 n 个词语生成词云图片;

参数:
- **type(str)** – ‘english’ or ‘chinese’;
- **n(int)** – 文章前 n 个词;
- **file_name** – 存储图片文件名;
- **max_words** – 词云提取最大文字数量;

返回: 是否成功的信息.

mask(*type='english', pic='./data_cloud/pic2.png', n=300, file_name='wcd3.png', max_words=250*)

mask 词云图像: 将从 pic 路径读取 mask 图片;

参数:
- **type(str)** – ‘english’ or ‘chinese’;
- **pic(str)** – mask图片路径;
- **n(int)** – 采用文档前 n 个词;
- **file_name(str)** – 保存图片路径;
- **max_words(int)** – 词云图像中最大词个数;

返回: 是否成功的信息.

show_text(*type='english', n=300*)

展示指定文档;

参数:
- **type(str)** – ‘english’ or ‘chinese’;
- **n(int)** – 展示文章前 n 个词;

classTask_1.data_learn(path)

基于源数据 `NMDS_coordinates.csv`

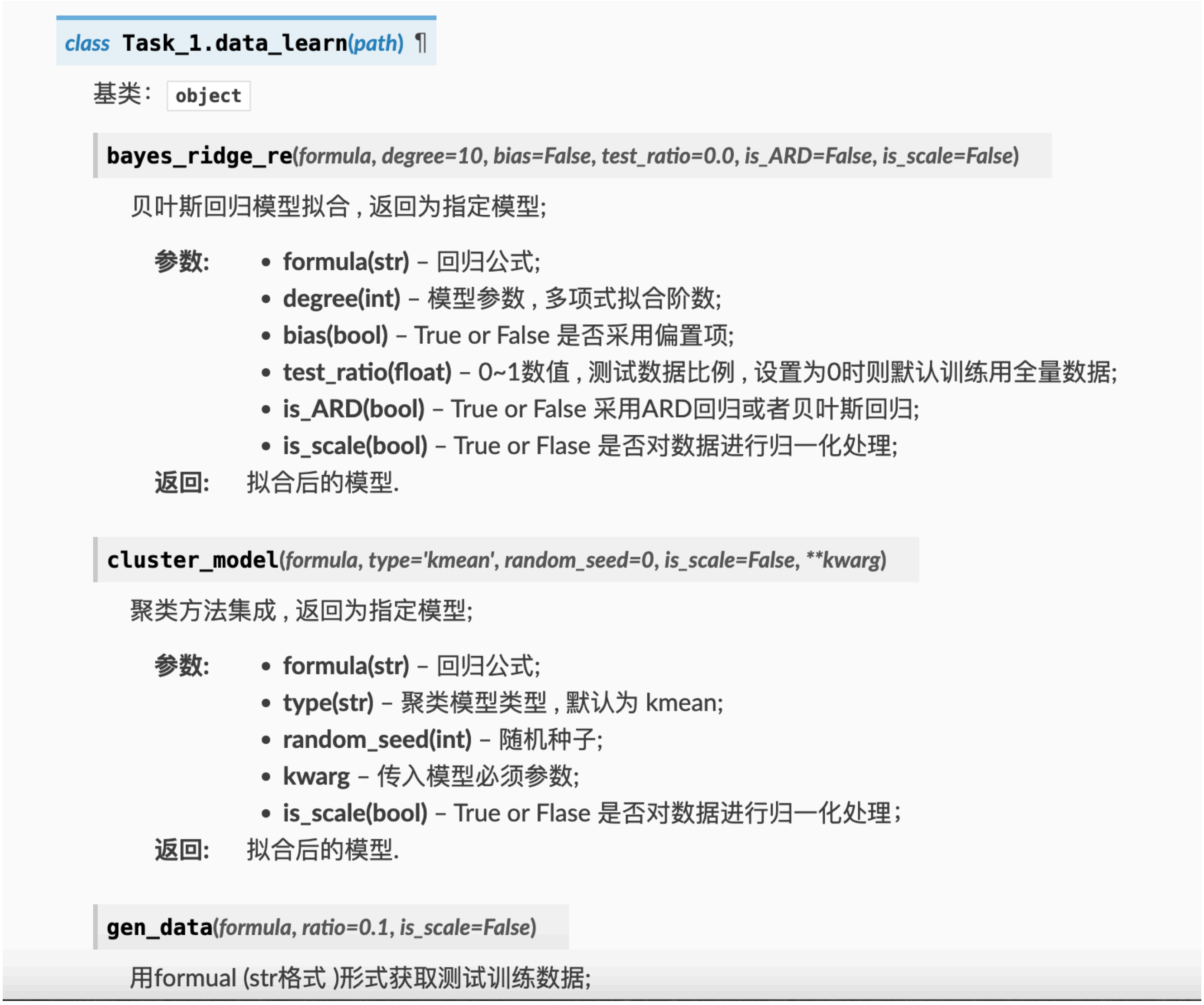
- 实现方法

  1. 用formual (`str`)形式获取测试、训练数据
  2. 贝叶斯岭回归以及ARD回归
  3. 贝叶斯岭回归结果的 `seaborn` 图示
  4. kMeans方法及预测
  5. DBSCAN方法及预测
  6. 支持向量机（SVC）方法及预测
  7. 支持超参的网格搜索方式

```
if __name__ == '__main__':
    data = data_learn('./2022期末数据/NMDS_coordinates.csv')

    # 获取bayes回归模型
    model_bayes = data.bayes_ridge_re('NMDS1~X_Vel',test_ratio = 0.1)
    # 获取测试集数据
    _,_,X_test,y_test = data.gen_data('NMDS1~X_Vel',ratio = 0.1)
    # 可视化
    bayes_plot(model_bayes,X_test,y_test,file_name="./output/output_fig.png")

    # 获取Kmeans模型
    model_kmean = data.cluster_model(formula = 'mean_rug+sd_rug',
                                     type = 'kmean' ,n_cluster=3)
    print(f"Predict class for ([14.1234,1.2666]) is
          {model_kmean.predict([[14.1234,1.2666]])}") # 使用模型进行预测

    # 获取DBSCN模型
    model_dbscan = data.cluster_model(formula = 'mean_rug+sd_rug',
                                      type = 'DBSCAN' ,eps=0.1)
    print(f"DBSCAN predict {np.unique(model_dbscan.labels_)} classes.")

    # 获取SVC模型并且进行归一化处理
    model_svc = data.svm_svc('SEASON~X_Vel+MidPt',kernal='rbf'
                             ,test_ratio = 0.1,is_scale= True)
    # 获取测试集数据
    _,_,X_test,y_test = data.gen_data('SEASON~X_Vel+MidPt',
                                      ratio = 0.1,is_scale= True)
    y_pred = model_svc.predict(X_test)

    # 搜索上一个模型的最优超参
    param_grid = dict(gamma = np.logspace(-2,1,2), coef0 = np.logspace(0,5,10))
    _ = data.search_hyper_para(model_svc,param_grid,n_split = 10)
```

```
## 输出
You are using sklearn:1.0.2.
Nice to see you! This project is designed as Sklearn Class.
Notice: This is unsupervised learning since y is omitted!
Kmeans Score is -2749.8407722635643
Predict class for ([14.1234,1.2666]) is [1]
Notice: This is unsupervised learning since y is omitted!
DBSCAN predict [ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21
22 23
 24] classes.
Self predict score is 0.5168667810177244
Notice: Please establish the model first!
The best parametes are {'coef0': 1.0, 'gamma': 10.0} with a score of
0.5610169491525424
```

- 说明文档（部分）



class **Task_1.data_learn**(*path*) ¶

基类： `object`

bayes_ridge_re(*formula, degree=10, bias=False, test_ratio=0.0, is_ARD=False, is_scale=False*)

贝叶斯回归模型拟合，返回为指定模型;

参数:
- **formula(str)** – 回归公式;
- **degree(int)** – 模型参数，多项式拟合阶数;
- **bias(bool)** – True or False 是否采用偏置项;
- **test_ratio(float)** – 0~1数值，测试数据比例，设置为0时则默认训练用全量数据;
- **is_ARD(bool)** – True or False 采用ARD回归或者贝叶斯回归;
- **is_scale(bool)** – True or Flase 是否对数据进行归一化处理;

返回: 拟合后的模型.

cluster_model(*formula, type='kmean', random_seed=0, is_scale=False, **kwarg*)

聚类方法集成，返回为指定模型;

参数:
- **formula(str)** – 回归公式;
- **type(str)** – 聚类模型类型，默认为 kmean;
- **random_seed(int)** – 随机种子;
- **kwarg** – 传入模型必须参数;
- **is_scale(bool)** – True or Flase 是否对数据进行归一化处理;

返回: 拟合后的模型.

gen_data(*formula, ratio=0.1, is_scale=False*)

用formual (str格式)形式获取测试训练数据;

classTask_1.WebSchedules(name, web)

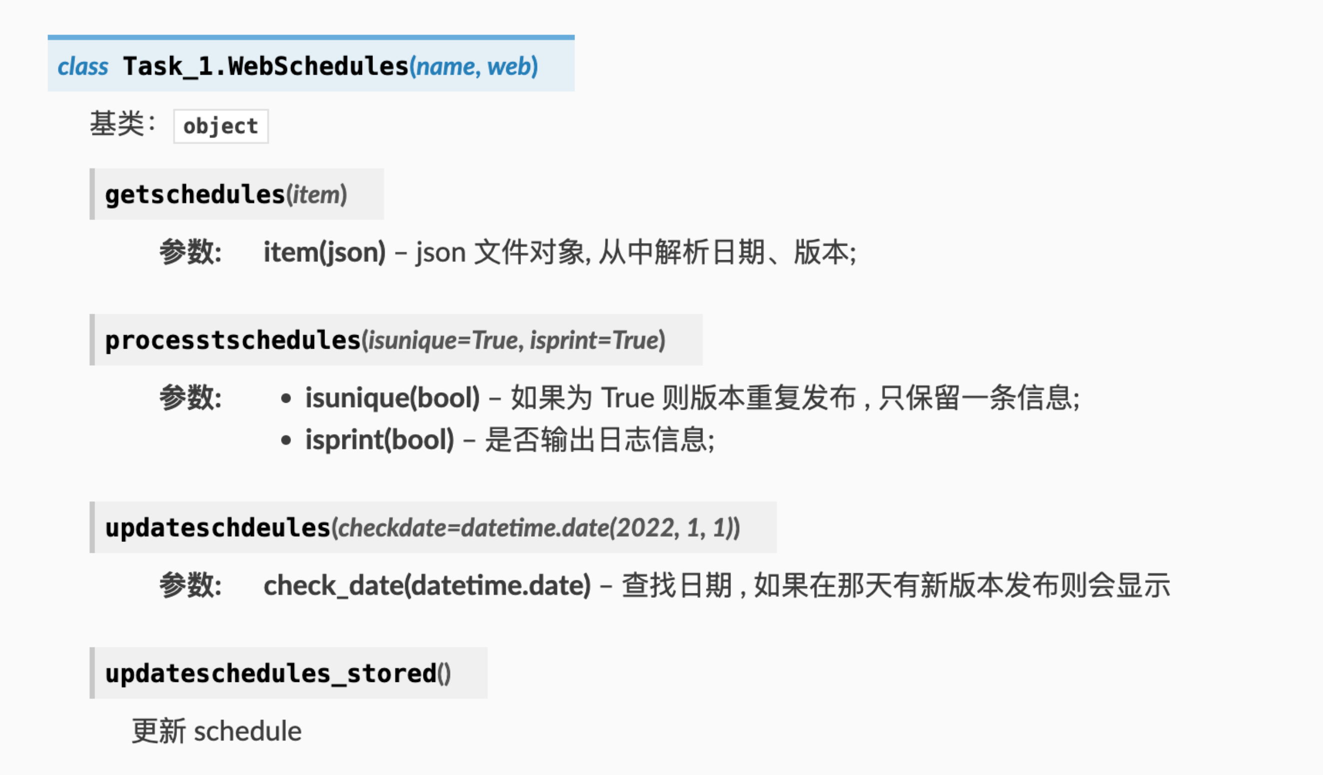
- 实现方法

1. 请求 Chrome 浏览器发布页面并解析

```
if __name__ == '__main__':
    now = datetime.datetime.now()
    headers = {...}
    # Chrome schedule
    chrome_schedules = WebSchedules('chrome_schedules','chrome')
    chrome_stable_release = WebSchedules('chrome_stable_release','chrome')
    response = requests.get(url_chrome_schedule_API, headers=headers)
    if response.status_code == 200:
        chrome_schedule = response.json()
        for item in chrome_schedule['items']:
            if 'Warning: avoid committing large changes,' not in item['summary']:
                chrome_schedules.getschedules(item)
                if re.match('M.* Stable Release', item['summary'], re.I):
                    chrome_stable_release.getschedules(item)

        chrome_schedules.processtschedules(isprint = False, isunique= False)
        chrome_schedules.updateschedules_stored()

        # Get stable release + stable cut + 1st beta info.
        chrome_stable_release.processtschedules(isprint = True, isunique= False)
        # stable release
        print("----------")
        check_date = pd.to_datetime(datetime.date(2023, 10, 31))
        chrome_stable_release.updateschdeules(checkdate= check_date)
```

```
## 输出
Hi~Yi. This class is designed for getting chrome-version schedule!
You are using requests:2.27.1.
 2022-12-03 20:09:33.839277
UPDATE:
CHROME SCHEDULE:
                           version
date
...
2023-10-03  M118 Stable Release
2023-10-31  M119 Stable Release
----------
CHROME M119 Stable Release RELEASED TODAY!
```

- 说明文档



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_2

classTask_2.elastic(path)

基于源数据 `confirmes.csv`

- 实现方法
  1. 弹性网络模型拟合及预测
  2. 多任务弹性网络模型拟合及预测

```
if __name__ == '__main__':
    data = elastic("./2022期末数据/confirmes.csv")

    # 弹性网络模型拟合
    params = {"alpha":1.0, "l1_ratio":0.5, "fit_intercept":True,
              "precompute":False, "max_iter":1000, "copy_X":True,
              "tol":1e-4, "warm_start":False, "positive":False, \
              "selection":"cyclic"}
    model = data.elsastic_net(target = ['contact'],x = ['mort','gueri'],
                              params=params, test_ratio=0.3)
    print(f"Predict value when ['mort','gueri'] = [250,10000]:
          {model.predict([[250,10000]])}")

    # 多任务弹性网络模型拟合
    params = {"alpha":1.0, "l1_ratio":0.5, "fit_intercept":True,
              "max_iter":1000, "copy_X":True, "tol":1e-4, "warm_start":False,
              "random_state":False, "selection":"cyclic"}
    model_m = data.multi_elsastic_net(target = ['cas','contact'],
                                      x = ['communautaire','mort','gueri'],
                                      params=params, test_ratio=0.3)
    print(f"Predict values when ['communautaire','mort','gueri'] = [4,250,10000]:
          {model_m.predict([[4,250,10000]])}")
```

```
## 输出
You are using sklearn:1.0.2.
Nice to see you! This project is designed as Sklearn Class 2.
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
You are using n_samples: 235 and n_features: 2 to train!
Hyper parameters: ['alpha', 'l1_ratio', 'fit_intercept', 'precompute', 'max_iter',
'copy_X', 'tol', 'warm_start', 'positive', 'selection']
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Score on test set is 0.021467686178646694
Predict value when ['mort','gueri'] = [250,10000]:[32.82221635]
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
You are using n_samples: 235 and n_features: 3 to train!
```

```
Hyper parameters: ['alpha', 'l1_ratio', 'fit_intercept', 'max_iter', 'copy_X',
'tol', 'warm_start', 'random_state', 'selection']
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Score on test set is 0.3560457943019035З
Predict values when ['communautaire','mort','gueri'] = [4,250,10000]:[[27.59085898
20.3436984 ]]
```

- 说明文档

class Task_2.elastic(*path*)

基类： `object`

elsastic_net(*target, x, params, test_ratio=0.0, random_seed=0*)

生成弹性网络模型，返回为指定模型;

参数:
- **list)** (*x(str)*) – 回归因变量(此时只能有一个);
- **list)** – 回归自变量;
- **test_ratio(float)** – 0~1数值，测试数据比例，设置为0时则默认训练用全量数据;
- **random_seed(int)** – 随机种子;
- **params(dict)** – 所使用的参数字典;

返回: 拟合后的模型enet

multi_elsastic_net(*target, x, params, test_ratio=0.0, random_seed=0*)

生成多任务弹性网络模型，返回为指定模型;

参数:
- **list)** (*x(str)*) – 回归因变量;
- **list)** – 回归自变量;
- **test_ratio(float)** – 0~1数值，测试数据比例，设置为0时则默认训练用全量数据;
- **random_seed(int)** – 随机种子;
- **params(dict)** – 所使用的参数字典;

返回: 拟合后的模型mten

classTask_2.Linear()

基于源数据 `zone_c_confirmesGeo.csv`

- 实现方法

1. 支持多元线性回归模型的拟合（区间估计）
2. 支持各种拟合优度数据的输出：AIC, BIC, R2, F统计量等
3. 支持预测（区间估计）
4. 支持带有线性约束条件的回归
5. `statsmodels` 库 `api` 的实现和比较

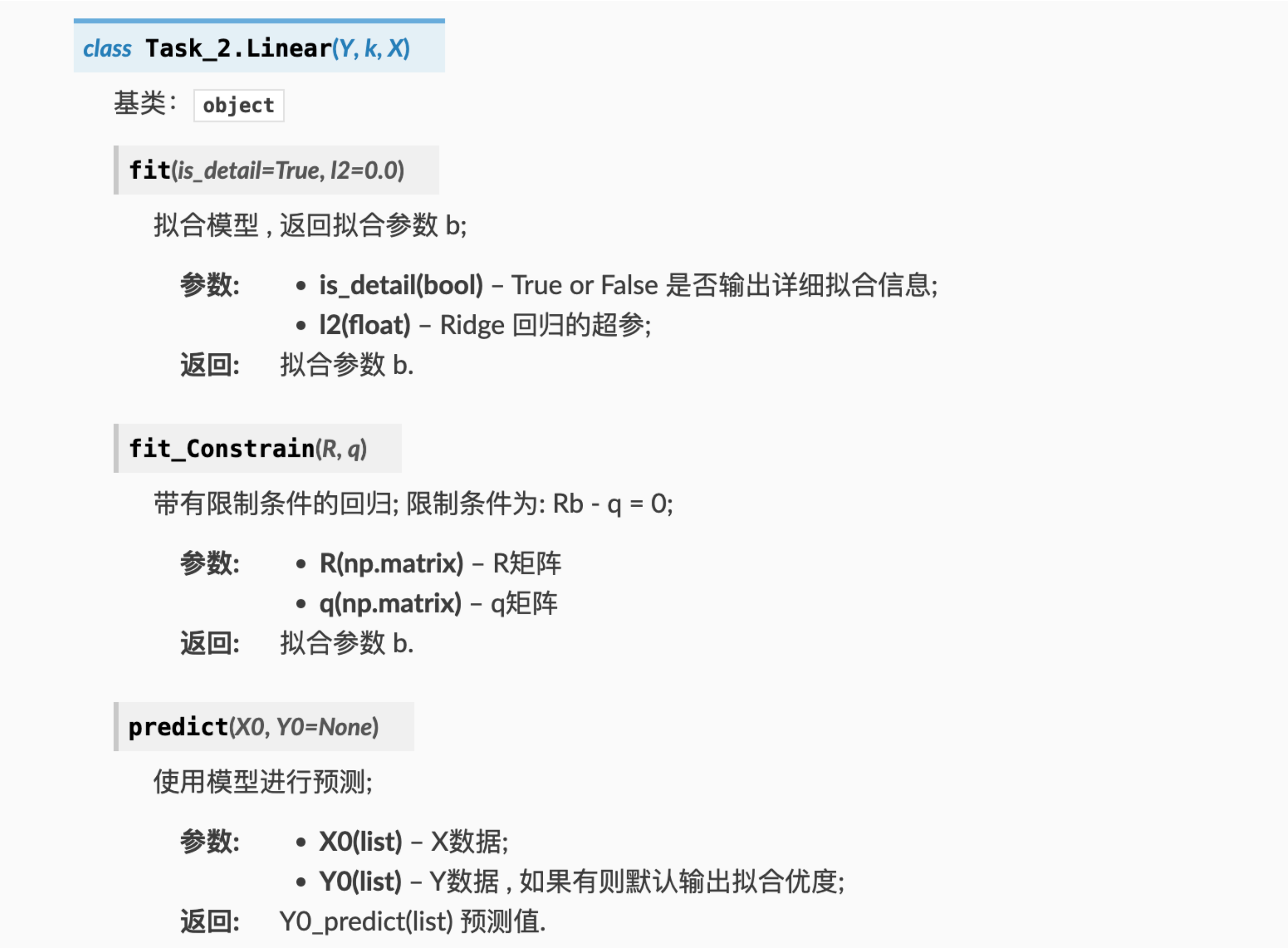
注意：

1. 输出中的 Rc 就是 R-squared
2. 在小样本下，区间估计会稍有出入，因为采用的是正态分布分位数而非精确T分布
3. 本实例仅依靠 `numpy` 进行矩阵向量算法
4. 实际上，本实例也实现了 Ridge 岭回归，但 `statsmodels` 库并没有直接的实现方式

```
file = pd.read_csv("./2022期末数据/zone_c_confirmesGeo.csv")
X, y = np.array(file['lat']), np.array(file['lon'])
X = sm.add_constant(X)
reg_learn = Linear(y,2,X)
reg_learn.fit()
```

```
## 输出
OLS Progressing...
OLS Regression Results
============================================================
const = -5.511065436104366 , [0.025 0.975]:[-11.171965634004241,0.1498347617955087]
t test = -1.908122008363944
x_1 = -0.7714117686812897 , [0.025 0.975]:[-1.1584240710853375,-0.3843994662772418]
t test = -3.9067674521540314
============================================================
Ruc = 0.9899007999359839
Rc = 0.07621400226135157
F-statistic = 15.2628319252101
Log-Likelihood = -364.1299342197638
AIC = 732.2598684395276
BIC = 738.7220856732368
============================================================
matrix([[-5.51106544],
        [-0.77141177]])
```

- 说明文档



class **Task_2.Linear**(*Y, k, X*)

基类： `object`

fit(*is_detail=True, l2=0.0*)

拟合模型，返回拟合参数 b;

参数:
- **is_detail(bool)** – True or False 是否输出详细拟合信息;
- **l2(float)** – Ridge 回归的超参;

返回: 拟合参数 b.

fit_Constrain(*R, q*)

带有限制条件的回归; 限制条件为: Rb - q = 0;

参数:
- **R(np.matrix)** – R矩阵
- **q(np.matrix)** – q矩阵

返回: 拟合参数 b.

predict(*X0, Y0=None*)

使用模型进行预测;

参数:
- **X0(list)** – X数据;
- **Y0(list)** – Y数据，如果有则默认输出拟合优度;

返回: Y0_predict(list) 预测值.

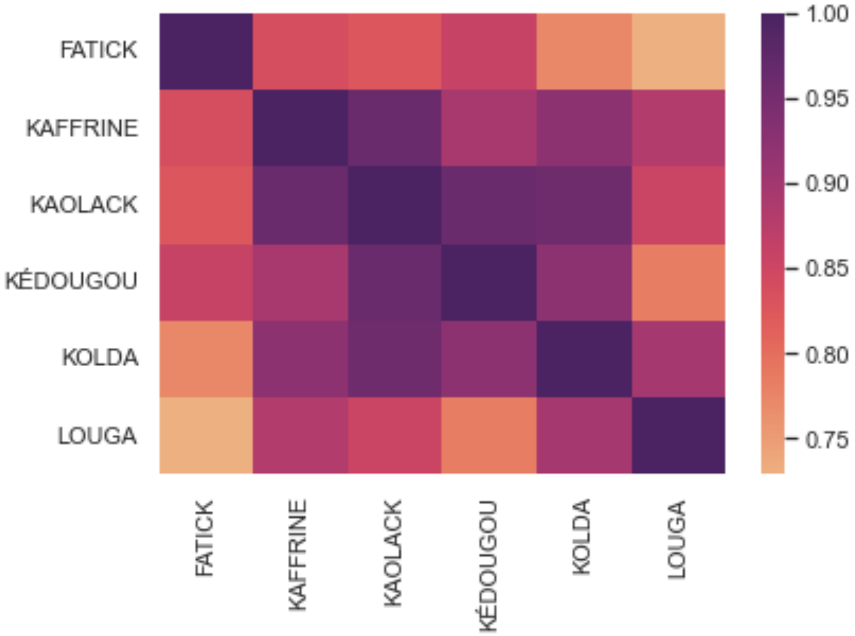
Seaborn

- 实现各种数据分布可视化
- 基于源数据

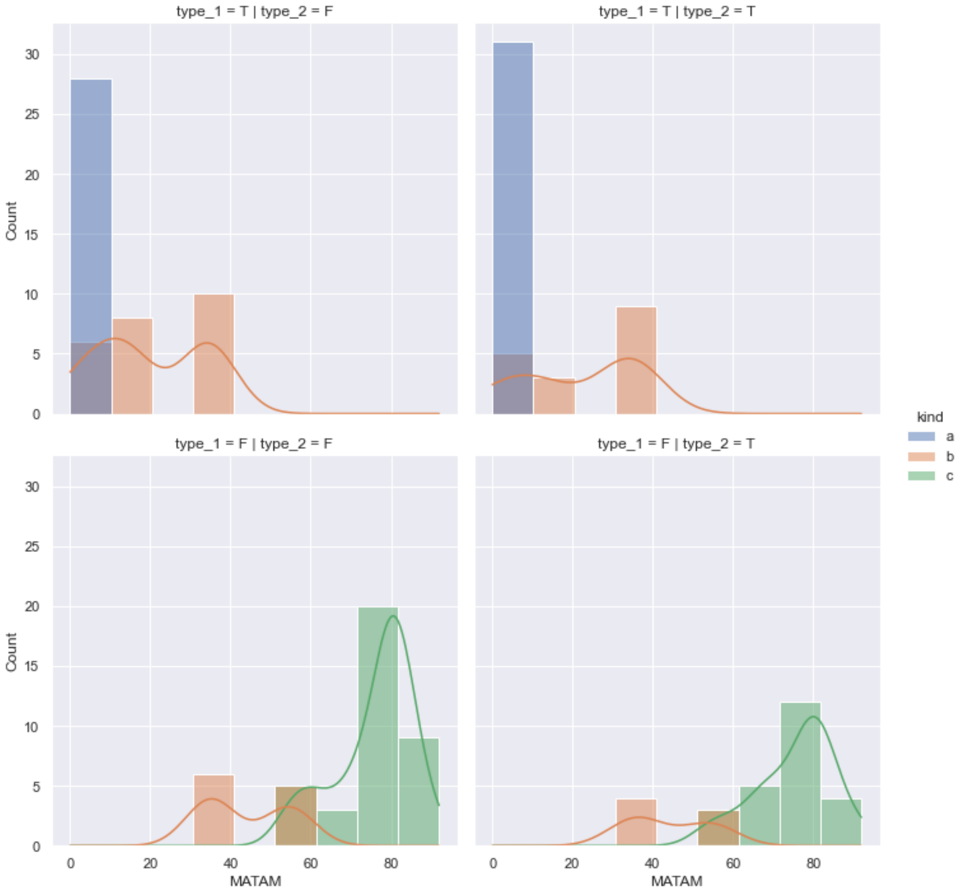
```
file = pd.read_csv("./2022期末数据/regions_cas.csv")

# 添加分类数据
file["kind"] = 59 * ["a"] + 59 * ["b"] + 61 * ["c"]
file["type_1"] = 100 * ["T"] + 79 * ["F"]
file = file.iloc[np.random.permutation(len(file))] # 随机打乱
file["type_2"] = 100 * ["F"] + 79 * ["T"]
```

- 使用举例：

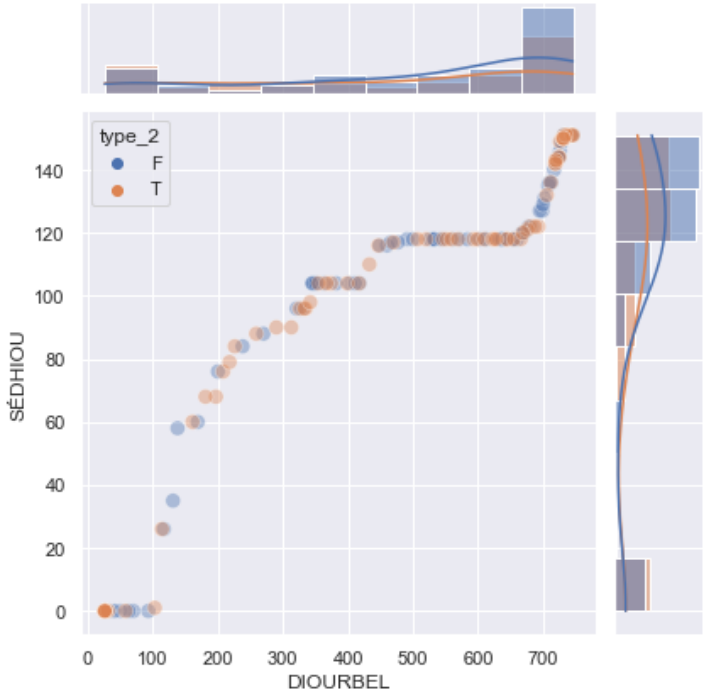
```
sns.heatmap(data = file[file.columns[3:9]].corr(), cmap = 'flare')
```



```
sns.displot(data = file,x="MATAM",row="type_1",
                                  col="type_2",hue="kind",kde = True)
```



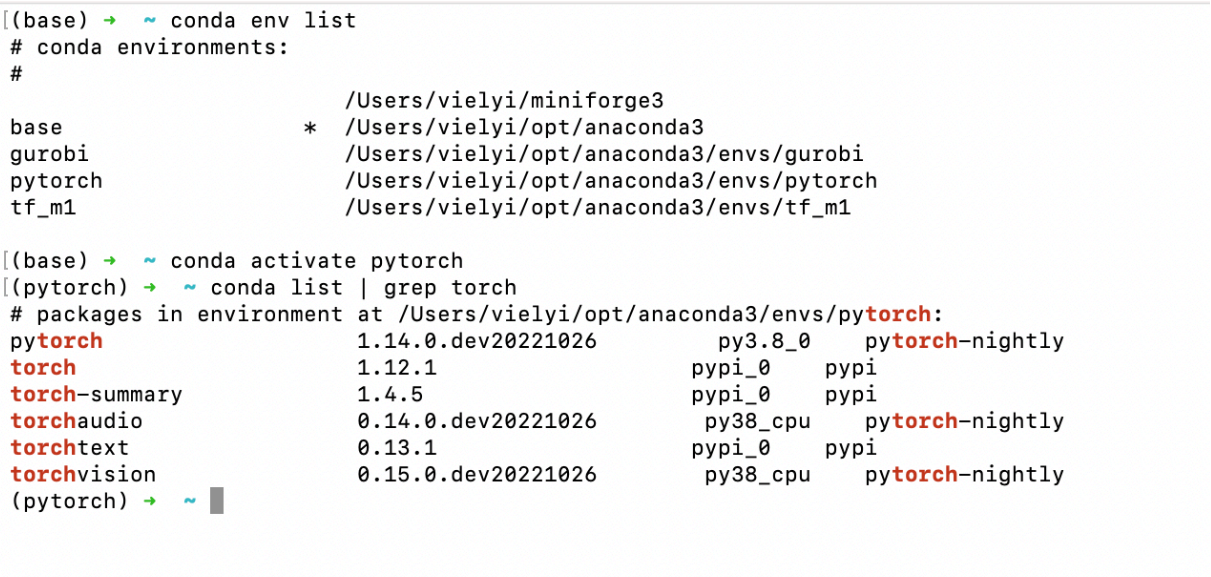
```
g = sns.JointGrid(data = file, x = "DIOURBEL", y = "SÉDHIOU",hue = 'type_2')
g.plot_joint(sns.scatterplot, s = 80, alpha =.4) # 点的大小和透明度
g.plot_marginals(sns.histplot, kde=True)
```



Task_3

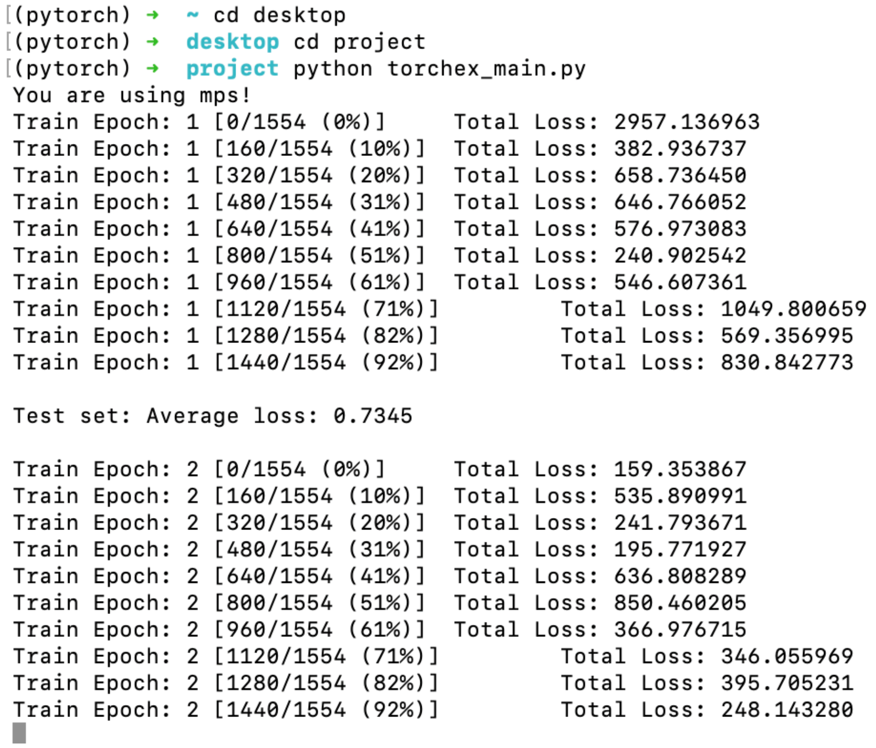
Pytorh 实例

- 基于源数据：`NMDS_coordinates.csv`
- 实现：
  1. `Dataset` 自定义，(实现在 `AsData.py` 中，需要在这里import)
  2. 全连接神经网络 `Net()` 类 (继承自 `nn.Module`)
  3. 网络训练和测试
  4. 使用 `argparse` 提供自定义参数设计并提供保存模型端口等
- Torch 环境：



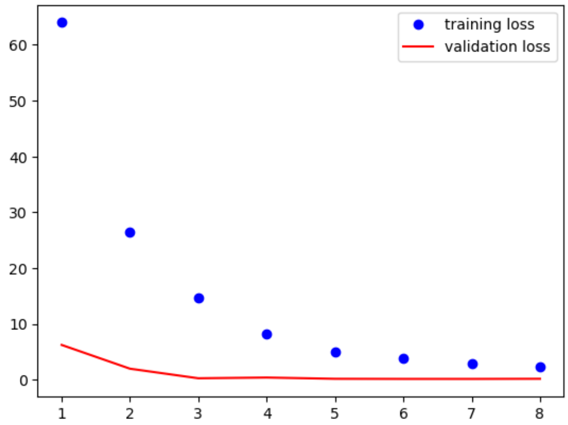
```
[(base) ➜  ~ conda env list
# conda environments:
#
                         /Users/vielyi/miniforge3
base                  *  /Users/vielyi/opt/anaconda3
gurobi                   /Users/vielyi/opt/anaconda3/envs/gurobi
pytorch                  /Users/vielyi/opt/anaconda3/envs/pytorch
tf_m1     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

- 程序使用：

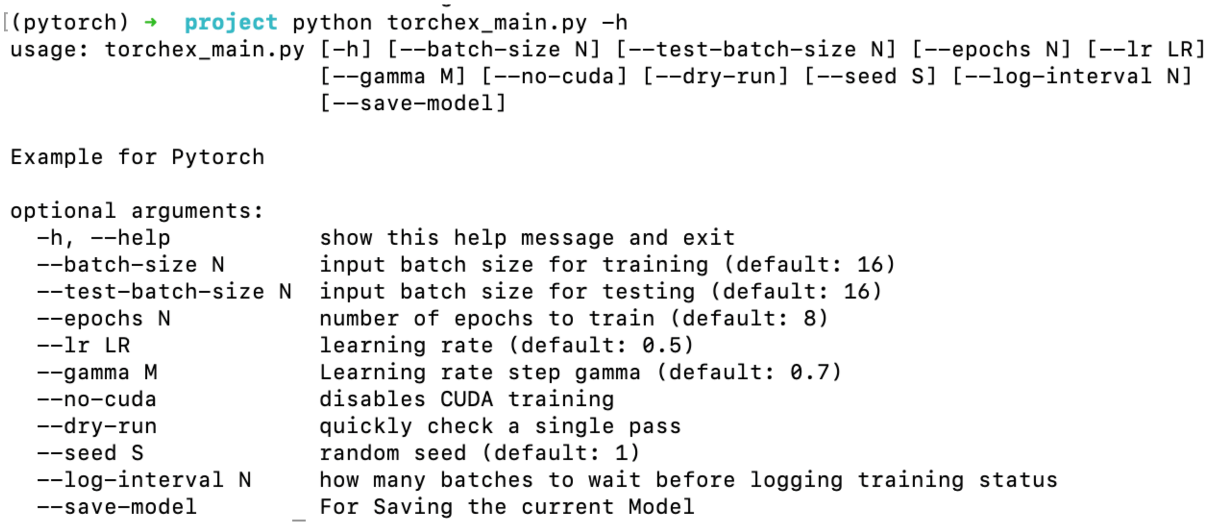


```
[(pytorch) ➜  ~ cd desktop
[(pytorch) ➜  desktop cd project
[(pytorch) ➜  project python torchex_main.py
You are using mps!
Train Epoch: 1 [0/1554 (0%)]     Total Loss: 2957.136963
Train Epoch: 1 [160/1554 (10%)]  Total Loss: 382.936737
Train Epoch: 1 [320/1554 (20%)]  Total Loss: 658.736450
Train 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
```

- 输出结果：



- 其他可获取参数



```
(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
```

Task_4

Python最优化算法

1. 入门级实例：线性规划问题实例(及非线性约束)
2. 更多约束：TSP问题求解

- 基于数据源

```
data = pd.read_csv("./2022期末数据/11 (1).tsv" , sep = "\t")
```

- 举例：

```
# 配置模型
model = gurobipy.Model("Model 2")

# 定义自变量
tpm = model.addVar(lb=0.0, ub = gurobipy.GRB.INFINITY,
vtype=gurobipy.GRB.CONTINUOUS, name = "tpm")
tmm = model.addVar(lb=0.0, ub = gurobipy.GRB.INFINITY,
vtype=gurobipy.GRB.CONTINUOUS, name = "tmm")

# 设置目标函数
model.setObjective(20 * tpm + 10 * tmm, gurobipy.GRB.MAXIMIZE) # 最大化
# 设置非线性约束
for i in range(5):
    model.addConstr(float(data.iloc[i][['tpm']].values) * tpm**2
                        + float(data.iloc[i][['tmm']].values) * tmm**2<= 100)
    # f非线性约束中必须是整数幂 , float 类型会报错
    model.addConstr(float(data.iloc[i+5][['tpm']].values) * tpm
                    + float(data.iloc[i+5][['tmm']].values) * tmm >= 1)
model.update()

# 显示求解过程（套路代码）
model.Params.LogToConsole = True
# 开始优化求解
```

```
## 输出
Gurobi Optimizer version 9.5.2 build v9.5.2rc0 (mac64[rosetta2])
Thread count: 8 physical cores, 8 logical processors, using up to 8 threads
Optimize a model with 5 rows, 2 columns and 10 nonzeros
Model fingerprint: 0xe06fe8d1
Model has 5 quadratic constraints
Coefficient statistics:
  Matrix range     [1e-02, 6e-01]
  QMatrix range    [3e-02, 2e-01]
  Objective range  [1e+01, 2e+01]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+00, 1e+00]
  QRHS range       [1e+02, 1e+02]
Presolve time: 0.00s
Presolved: 20 rows, 15 columns, 33 nonzeros
Presolved model has 5 second-order cone constraints
Ordering time: 0.00s

Barrier statistics:
 Dense cols : 1
 AA' NZ     : 1.080e+02
 Factor NZ  : 2.120e+02
 Factor Ops : 2.874e+03 (less than 1 second per iteration)
 Threads    : 1

                  Objective                Residual
...

Barrier solved model in 7 iterations and 0.02 seconds (0.00 work units)
Optimal objective 5.95073794e+02
```

```
print(f"optimal objective value is {model.objVal}.")
# 查看变量取值
for v in model.getVars():
    print('%s %g' % (v.varName, v.x))
```

```
## 输出
optimal objective value is 595.0737937549338.
tpm 14.3987
tmm 30.71
```

- TSP 问题

```
model = gurobipy.Model('TSP')

x = model.addVars(city_num, city_num, vtype = gurobipy.GRB.BINARY, name = 'x')
utility = model.addVars(city_num, lb = 0,
                        vtype = gurobipy.GRB.CONTINUOUS, name = 'utility')

model.setObjective(sum(x[i,j] * distance[i,j]
                       for i in range(city_num) for j
                       in range(city_num)), GRB.MINIMIZE)

# 设置约束
for j in range(city_num):
    model.addConstr(gurobipy.quicksum(x[i,j] for i in range(city_num)) == 1)
for i in range(city_num):
    for j in range(1, city_num):
        if (i != j):
            model.addConstr(utility[i] - utility[j]
                            + city_num * x[i,j] <= city_num - 1)
model.update()

# 显示求解过程
model.Params.LogToConsole = True
# 限制求解时间
model.Params.TimeLimit =50
# 开始优化求解
model.optimize()

print(f"optimal objective value is {model.objVal}.")

# 将结果输出成路径
u_value = []
for i in range(city_num):
    u_value.append(utility[i].x) # 加载 utility[i] 的值
sequence = sorted(enumerate(u_value), key = lambda y:y[1]) # 按照 utility 值排序
for item in sequence:
    print(item[0] + 1,'->',end=' ')
print(1)
```

```
optimal objective value is 1.136092581758076.
1 -> 5 -> 26 -> 7 -> 30 -> 25 -> 28 -> 29 -> 2 -> 22 -> 3 -> 4 -> 6 -> 21 -> 23 ->
27 -> 12 -> 24 -> 11 -> 13 -> 14 -> 8 -> 9 -> 10 -> 20 -> 16 -> 17 -> 15 -> 18 ->
19 -> 1
```

R_startup

主要内容：

- 主成分分析
- 支持向量机的应用和自动调参设计
- R 神经网络的应用和结果评估
- 决策树生成和绘图
- 广义线性回归应用
- R集成学习：梯度提升树
- 基本作图：包括柱状图、箱形图、饼状图等
- 偏最小二乘回归PLS和主成分回归PCR