东北财经大学

DONGBEI UNIVERSITY OF FINANCE & ECONOMICS

AY24/25-S2

机器学习与大数据审计

管理科学与工程学院

谢琛

(71171532K)

自我介绍

东北财经大学 DONGBEI UNIVERSITY OF FINANCE & ECONOMICS

东北财经大学 Dongbei University of Finance and Economics 管理科学与工程学院 School of Management Science and Engineering

请输入关键字 English 新闻资讯 新媒体 MEM招生 联系方式

首页 学院概况 师资队伍 教学管理 科学研究 党团天地 校友 合作交流 资料下载 课程思政

首页 >> 师资队伍 >> 教师风采 >> 大数据管理与应用系（中心） >> 谢琛

谢琛

姓名：谢琛　　性别：男

邮箱：xiechen@dufe.edu.cn　　学位：博士

职称：副教授　　硕士生导师：是

博士生导师：否

研究方向

数据科学、智能计算、金融科技、可解释性人工智能、清洁能源、智慧医疗等。

教育背景

2016.07 – 2022.03 新加坡南洋理工大学 计算机科学与工程 博士学位

2013.07 – 2016.01 新加坡南洋理工大学 电子电气工程 学士学位

简介

谢琛，男，副教授，现就职于东北财经大学-管理科学与工程学院-大数据管理与应用系。主要研究方向为数据科学、智能计算、可解释性人工智能、金融科技、清洁能源、智慧医疗等。本科毕业于新加坡南洋理工大学-电子电气工程专业，博士毕业于新加坡南洋理工大学-计算机科学与工程学院 。 先后在新加坡南洋理工大学智能计算中心、南洋理工大学劳斯莱斯实验室、剑桥大学新加坡先前研究院，以及新加坡胜科能源集团从事科学研发工作，参与多项项目及研究课题并在国际顶级期刊发表学术论文。

工作经历

2023 - 至今 东北财经大学 大数据管理与应用系 副教授

2022 - 2023 胜科能源集团 数据科学与分析部 Data Scientist

2022 - 2022 剑桥大学 新加坡先前研究院 博士后

2021 - 2022 南洋理工大学 劳斯莱斯实验室 科研助理

社会任职

主讲课程

机器学习（本科生）

多模态非结构化数据分析与应用（本科生）

金融科技（硕士生）

机器学习与人工智能前沿（博士生）

主要科研课题

代表性学术成果

Xie Chen, Deepu Rajan, Dilip K. Prasad, and Chai Quek, "An Embedded Deep Fuzzy Association Model for Learning and Explanation" , Appl. Soft. Comput., 2022: 109738.

Xie Chen, Deepu Rajan, and Chai Quek, "An Interpretable Neural Fuzzy Hammerstein-Wiener Network for Stock Price Prediction" , Inf. Sci, 577, 324-335, 2021.

Xie Chen, Deepu Rajan, and Chai Quek, "A Deep Hybrid Fuzzy Neural Hammerstein-Wiener Network for Stock Price Prediction" in International Conference on Artificial Intelligence in Information and Communication (ICAIIC), pp. 288–293, IEEE, 2020.

Nelson M. K., Xie Chen, Qi Cao, Chai Quek, GEMM-SAFIN(FRIE)++: Explainable Artificial Intelligence Visualisation System with Episodic Memory , in Cutting Edge Applications of Computational Intelligence Tools and Techniques, 1st ed., Springer, Dec 2023 [Book Chapter]

竞赛和奖励：

2024年（第十届）全国大学生统计建模大赛，国赛三等奖，第一指导教师

2024年（第十四届）三创赛辽宁赛区选拔赛，省赛一等奖，第一指导教师，竞赛优秀指导教师

2024年 全国大学生数学建模竞赛辽宁赛区， 省赛一等奖，第一指导教师

2024年 大创立项省级1项，第一指导教师

1. 课程介绍

- 课程介绍
- 课程大纲
- 课程安排
- 考核方式
- 课程资源
- 联系方式

	《机器学习与大数据审计》
课程简介	• 主要介绍机器学习算法和AI在大数据审计领域的应用 • 以实践为导向，利用常见的机器学习模型解决审计问题
参考书	周志华编著，机器学习，清华大学出版社，2016. 李贺编著，大数据审计，高等教育出版社，2024.
课程学时	36学时，每周2学时（周一，3-4节） 1-18周，笃行楼602
教材框架	======= 起 ========== 第一章 绪论 第二章 模型评估与选择 第三章 线性模型 第四章 决策树
	======= 承 ========= 第五章 神经网络 第六章 支持向量机 第七章 贝叶斯分类器 第八章 集成学习
	======= 转 ======== 第九章 聚类 第十章 降维 第十一章 特征选择 第十二章 计算学习
	======== 合 ======== 第十三章 大数据审计 第十四章 描述性分析 第十五章 报表识别 第十六章 风险评估

得心应手

2. 课程大纲

课程介绍
课程大纲
课程安排
考核方式
课程资源
联系方式

1 绪论+Python 基础及环境配置
2 模型评估与选择
3 线性模型
4 决策树
5 神经网络
6 聚类
7 大数据审计概论
8 审计数据分析
9 报表识别
10 风险评估与审计报告

3. 课程安排 (线下)

- 课程介绍
- 课程大纲
- 课程安排
- 考核方式
- 课程资源
- 联系方式

=== 起 ===	=== 承 ===	===转 ===	=== 合 ===
课程介绍+绪论 附-Python环境搭建	**神经网络1** - 感知机、MLP	**神经网络2** - 感知机、MLP	**大数据审计概论** - 大数据审计的发展及趋势
模型评估与选择 - ROC与AUC、F1与Precision	**神经网络3** - 分类、预测实践	**神经网络4** - 分类、预测实践	**审计数据分析** - 描述性分析、探索性分析、可视化分析
线性模型 - 线性回归、Linear Programming	**神经网络5** -深度学习概论	**神经网络6** -深度学习实践（CNN）	**报表识别** - 报表、票据、手写记录OCR识别
决策树 - 木、林、森	**集成学习** - Bagging and Boosting, 投票法	**聚类** - KNN、SOM、DB-Scan	**风险评估与审计报告** - 风险评估、审计报告自动生成

4. 考核方式

课程介绍
课程大纲
课程安排
考核方式
课程资源
联系方式

出勤 (10%)

- 课堂出席情况，总出席率 (8%)
- 积极参与分组讨论、与组内同学互相交流（2%）

平时作业 (30%)

- Assignment 1 – 随堂测试 (10%)
- Assignment 2 – （二选一）数据可视化分析案例 (10%)
- Assignment 3 – 财务报表、手写识别案例实践（10%）

课程论文(60%)

- 【小组】从给出的选题方向，根据小组成员兴趣，选择一个选题进行调研和实践，PPT小组汇报 (35%)
- 【个人】将课堂上的个人选题形成报告，可于课程结束时或学期末提交 (25%)

5. 课程资源

课程介绍
课程大纲
课程安排
考核方式
课程资源
联系方式

	《机器学习与大数据审计》- 课程资源 (持续更新中。。。)
中文参考书	• 邱锡鹏编著，神经网络与深度学习，机械工业出版社，2021 • 李轩涯，计湘婷，曹焯然编著，机器学习实践，清华大学出版社，2021
外文参考书	• Simon Haykin, Neural Networks and Learning Machines，Pearson Education, 2008 • Stephen Marsland, Machine Learning: An Algorithmic Perspective, CRC Press, 2014
代码博客	Pytorch写CNN算法的博客： https://blog.paperspace.com/writing-cnns-from-scratch-in-pytorch/
视频课程	《Python编程：从入门到实践》配套视频：https://www.bilibili.com/video/BV14X4y127MT?p=4 吴恩达教授《机器学习》视频课程： https://www.bilibili.com/video/BV1W34y1i7xK/ 吴恩达教授《深度学习》视频课程： https://www.bilibili.com/video/BV16r4y1Y7jv/

6. 联系方式

课程介绍
课程大纲
课程安排
考核方式
课程资源
联系方式

	《机器学习与大数据审计》- 联系方式
邮箱	xiechen@dufe.edu.cn
微信	微信群 - 机器学习与大数据审计25上 (S2)
办公室	东北财经大学 （师言阁313室）
白果云	

东北财经大学

DONGBEI UNIVERSITY OF FINANCE & ECONOMICS

第一章

机器学习绪论
Python 基础及环境配置

管理科学与工程学院

谢琛

课程大纲

Content

1 绪论+Python 基础及环境配置

2 模型评估与选择

3 线性模型

4 决策树

5 神经网络

6 集成学习

7 聚类

8 降维

9 特征选择

10 特征转化

1. 生活中的案例

- 生活案例
- AI 关联
- 引申创新
- 案例反思

生活中的案例：冰糖雪梨 – 该怎么做?

1. 生活中的案例

生活案例
AI 关联
引申创新
案例反思

0. 在什么样的背景下，关注冰糖雪梨?

-- 研究的背景

1. 冰糖雪梨是啥?

-- 研究对象的定义

2. 不同品种或地域的梨做出来有差别吗?

-- 横向对比研究内容及研究背景

3. 大体有几种类型的制作分享(搜索结果)?

-- 有点类似于文献综述的搜索及归纳

4. 每种制作的特点和不足是啥?

-- 分析文献分支的特殊应用场景及限制条件

5. 哪种制作更适合我的食材和制作条件?

-- 对研究环境和条件的评估、采用适合的路线

6. 有没有自己的特色和创新步骤?

-- 继承与创新性

生活中的案例：冰糖雪梨 – 该怎么做?

1. 生活中的案例

生活案例
AI 关联
引申创新
案例反思

0. 在什么样的背景下，关注冰糖雪梨?

-- 研究的背景

1. 冰糖雪梨是啥?

-- 研究对象的定义

2. 不同品种或地域的梨做出来有差别吗?

-- 横向对比研究内容及研究背景

3. 大体有几种类型的制作分享(搜索结果)?

-- 有点类似于文献综述的搜索及归纳

4. 每种制作的特点和不足是啥?

-- 分析文献分支的特殊应用场景及限制条件

5. 哪种制作更适合我的食材和制作条件?

-- 对研究环境和条件的评估、采用适合的路线

6. 有没有自己的特色和创新步骤?

-- 继承与创新性

生活中的案例：冰糖雪梨 – 该怎么做?

冰糖雪梨近年来一直受到我国家庭的广泛青睐，主要是因为它甘甜可口，同时具有生津养胃、清热化痰、止咳的功效。冰糖雪梨是一道常见的甜品，以雪梨和冰糖一起慢火炖制而成，属于粤菜系。雪梨是一种蔷薇科梨属的乔木植物，它在中国主要分布于河北、山东、四川、安徽等地，一般生长在海拔100-2000米干旱寒冷的地区或山坡阳处。从雪梨的产地上看，四川雪梨果实呈扁圆形，山东雪梨果实呈近球形，两者在外观上有些许差别，但它们有相似的药效，都具有主治风热、润肺、凉心、消痰、降火、解毒等药效；现代医学研究证明，雪梨具有润肺清燥、止咳化痰、养血生肌的作用。冰糖雪梨的制作过程包括将雪梨切片与冰糖同时放入炖盅，倒入清水没过雪梨的位置，蒸炖约30分钟后放凉食用。有的做法在食材中加入百合和柠檬，既加强了润肺止咳的功效，同时也增加了柠檬的味道，但百合需要提前浸泡10分钟后与雪梨及其他食材一起蒸炖。还有的做法则使用微波炉代替蒸炖，高火加热20分钟，比蒸炖的方法节省了1/3的时间，但效果和口感可能与蒸炖出的冰糖雪梨相比稍微欠缺一些。考虑到现有的原材料和需要润肺止咳的效果，决定采取蒸炖的方法，加入雪梨和冰糖。由于缺少百合和柠檬，同时希望提高免疫力和减少疲劳，决定额外加入枸杞一起蒸炖。经过实际操作验证和群众反馈，冰糖雪梨口感甘甜可口，止咳效果显著，达到了制作目标。同时，以加入枸杞为创新点的方案具有广泛的拓展和应用性，具有深远的意义!

1. 生活中的案例

生活案例
AI 关联
引申创新
案例反思

0. 在什么样的背景下，关注冰糖雪梨?

-- 研究的背景

1. 冰糖雪梨是啥?

-- 研究对象的定义

2. 不同品种或地域的梨做出来有差别吗?

-- 横向对比研究内容及研究背景

3. 大体有几种类型的制作分享(搜索结果)?

-- 有点类似于文献综述的搜索及归纳

4. 每种制作的特点和不足是啥?

-- 分析文献分支的特殊应用场景及限制条件

5. 哪种制作更适合我的食材和制作条件?

-- 对研究环境和条件的评估、采用适合的路线

6. 有没有自己的特色和创新步骤?

-- 继承与创新性

1. 冰糖雪梨近年来一直受到我国家庭的广泛青睐，主要是因为它甘甜可口，同时具有生津养胃、清热化痰、止咳的功效。
2. 冰糖雪梨是一道常见的甜品，以雪梨和冰糖一起慢火炖制而成，属于粤菜系。
3. 雪梨是一种蔷薇科梨属的乔木植物，它在中国主要分布于河北、山东、四川、安徽等地，一般生长在海拔100-2000米干旱寒冷的地区或山坡阳处。
4. 从雪梨的产地上看，四川雪梨果实呈扁圆形，山东雪梨果实呈近球形，两者在外观上有些许差别，但它们有相似的药效，都具有主治风热、润肺、凉心、消痰、降火、解毒等药效；现代医学研究证明，雪梨具有润肺清燥、止咳化痰、养血生肌的作用。
5. 冰糖雪梨的制作过程包括将雪梨切片与冰糖同时放入炖盅，倒入清水没过雪梨的位置，蒸炖约30分钟后放凉食用。
6. 有的做法在食材中加入百合和柠檬，既加强了润肺止咳的功效，同时也增加了柠檬的味道，但百合需要提前浸泡10分钟后与雪梨及其他食材一起蒸炖，增加了制作的复杂程度。还有的做法则使用微波炉代替蒸炖，高火加热20分钟，比蒸炖的方法节省了1/3的时间，但效果和口感可能与蒸炖出的冰糖雪梨相比稍微欠缺一些。
7. 考虑到现有的原材料和需要润肺止咳的效果，决定采取蒸炖的方法，加入雪梨和冰糖。
8. 由于缺少百合和柠檬，同时希望提高免疫力和减少疲劳，决定额外加入枸杞一起蒸炖。
9. 经过实际操作验证和群众反馈，冰糖雪梨口感甘甜可口，止咳效果显著，达到了制作目标。同时，以加入枸杞为创新点的方案具有广泛的拓展和应用性，具有深远的意义！

1. 生活中的案例 – 制作过程

- 生活案例
- AI 关联
- 引申创新
- 案例反思

生活中的案例：冰糖雪梨 – 制作过程

20分钟

研究和实施的方法反思?

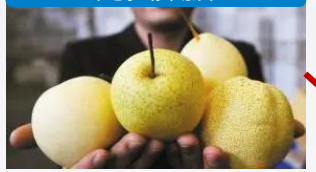
从想吃冰糖雪梨到达成目标，如何参考、设计、操作、实现?

2. 生活中案例与AI的关联

- 生活案例
- AI 关联
- 引申创新
- 案例反思

生活中的案例：冰糖雪梨 – 制作过程

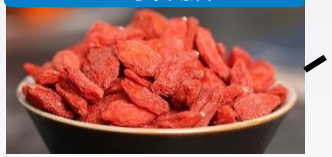
购买数据



获取数据

纯净数据

纯净数据



数据清洗+预处理

数据融合

模型训练

20分钟

模型评估

万物皆想通，容易吧?

数据是模型的**食材**，**训练**是模型的**烹饪**方法

2. 典型的机器学习过程

使用学习算法（*learning algorithm*）

训练数据

类别标记
(*label*)

训练

姓名	职业	年收入	…	好顾客
张三	教师	6 万	…	否
李四	公务员	7 万	…	是
王五	学生	2 万	…	否
周六	企业家	15 万	…	是
董七	演员	13 万	…	否
钱八	教师	8 万	…	是

模型

决策树，神经网络，支持向量机，
Boosting，贝叶斯网络，……

新数据样本

(刘二, 公务员, 8万, …, ?)

? = 是

类别标记
未知

2. 基本术语-数据

特征 ↑　　标记 ↑

编号	色泽	根蒂	敲声	好瓜
1	青绿	蜷缩	浊响	是
2	乌黑	蜷缩	沉闷	是
3	青绿	硬挺	清脆	否
4	乌黑	稍蜷	沉闷	否
1	青绿	蜷缩	沉闷	?

训练集 ←　测试集 ←

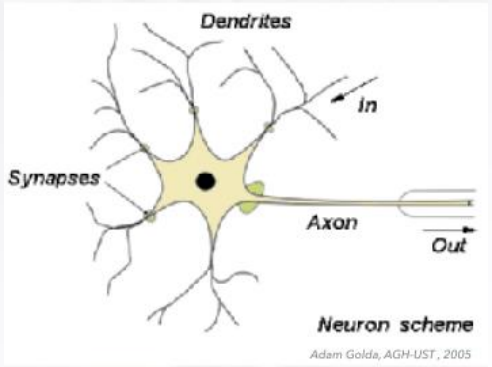
- 预测目标：
  - 分类:离散值
    - 二分类:好瓜;坏瓜
    - 多分类:冬瓜;南瓜;西瓜
  - 回归:连续值

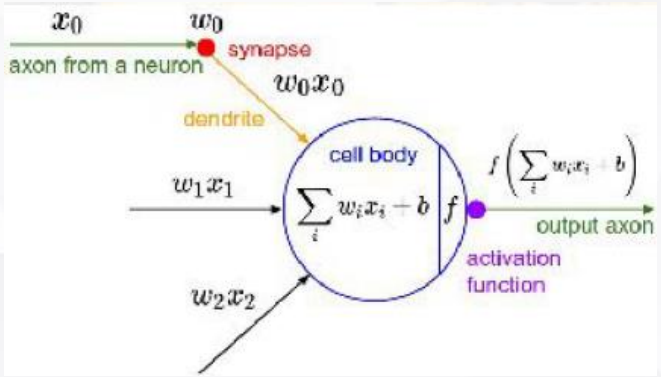
    瓜的成熟度
  - 聚类:无标记信息

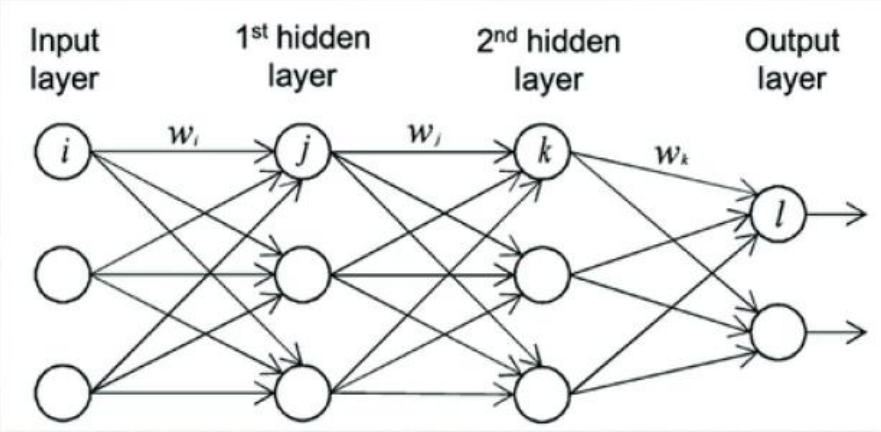
2. 生活中案例与AI的关联

- 生活案例
- AI 关联
- 引申创新
- 案例反思

生活中的案例：冰糖雪梨 – 制作过程







3. 方法的继承与创新

- 生活案例
- AI 关联
- 引申创新
- 案例反思

案例延伸：冰糖雪梨 – 继承与反复 – 秋梨膏

分阶段加入

秋梨膏

方法的继承与创新?

面对已有的方法，**继承+复现+创新**

4. 生活中的案例反思

生活案例
AI 关联
引申创新
案例反思

0. 在什么样的背景下，关注冰糖雪梨?

-- 研究的背景

1. 冰糖雪梨是啥?

-- 研究对象的定义

2. 不同品种或地域的梨做出来有差别吗?

-- 横向对比研究内容及研究背景

3. 大体有几种类型的制作分享(搜索结果)?

-- 有点类似于文献综述的搜索及归纳

4. 每种制作的特点和不足是啥?

-- 分析文献分支的特殊应用场景及限制条件

5. 哪种制作更适合我的食材和制作条件?

-- 对研究环境和条件的评估、采用适合的路线

6. 有没有自己的特色和创新步骤?

-- 继承与创新性

1. 冰糖雪梨近年来一直受到我国家庭的广泛青睐，主要是因为它甘甜可口，同时具有生津养胃、清热化痰、止咳的功效。
2. 冰糖雪梨是一道常见的甜品，以雪梨和冰糖一起慢火炖制而成，属于粤菜系。
3. 雪梨是一种蔷薇科梨属的乔木植物，它在中国主要分布于河北、山东、四川、安徽等地，一般生长在海拔100-2000米干旱寒冷的地区或山坡阳处。
4. 从雪梨的产地上看，四川雪梨果实呈扁圆形，山东雪梨果实呈近球形，两者在外观上有些许差别，但它们有相似的药效，都具有主治风热、润肺、凉心、消痰、降火、解毒等药效；现代医学研究证明，雪梨具有润肺清燥、止咳化痰、养血生肌的作用。
5. 冰糖雪梨的制作过程包括将雪梨切片与冰糖同时放入炖盅，倒入清水没过雪梨的位置，蒸炖约30分钟后放凉食用。
6. 有的做法在食材中加入百合和柠檬，既加强了润肺止咳的功效，同时也增加了柠檬的味道，但百合需要提前浸泡10分钟后与雪梨及其他食材一起蒸炖，增加了制作的复杂程度。还有的做法则使用微波炉代替蒸炖，高火加热20分钟，比蒸炖的方法节省了1/3的时间，但效果和口感可能与蒸炖出的冰糖雪梨相比稍微欠缺一些。
7. 考虑到现有的原材料和需要润肺止咳的效果，决定采取蒸炖的方法，加入雪梨和冰糖。
8. 由于缺少百合和柠檬，同时希望提高免疫力和减少疲劳，决定额外加入枸杞一起蒸炖。
9. 经过实际操作验证和群众反馈，冰糖雪梨口感甘甜可口，止咳效果显著，达到了制作目标。同时，以加入枸杞为创新点的方案具有广泛的拓展和应用性，具有深远的意义！

课程大纲

Content

1 绪论+Python 基础及环境配置

2 模型评估与选择

3 线性模型

4 决策树

5 神经网络

6 集成学习

7 聚类

8 降维

9 特征选择

10 特征转化

2. 安装Python

Windows

Mac OS

Anaconda

推荐第三种安装方式

部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

2. 安装Python

东北财经大学 DONGBEI UNIVERSITY OF FINANCE & ECONOMICS

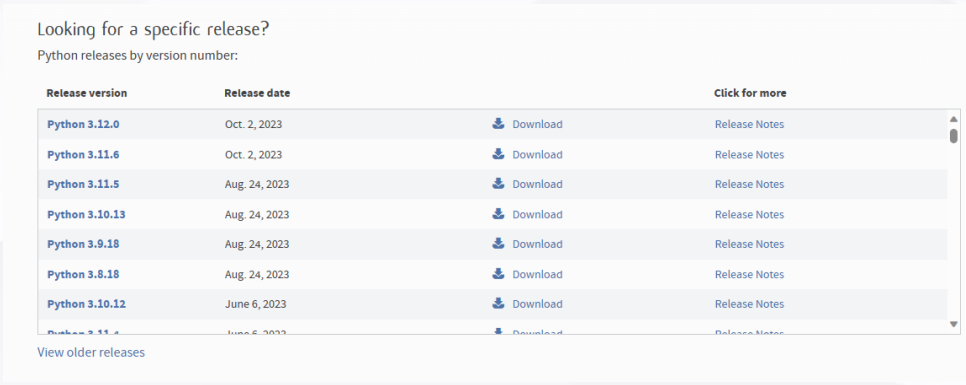
- Windows
- Mac OS
- Anaconda

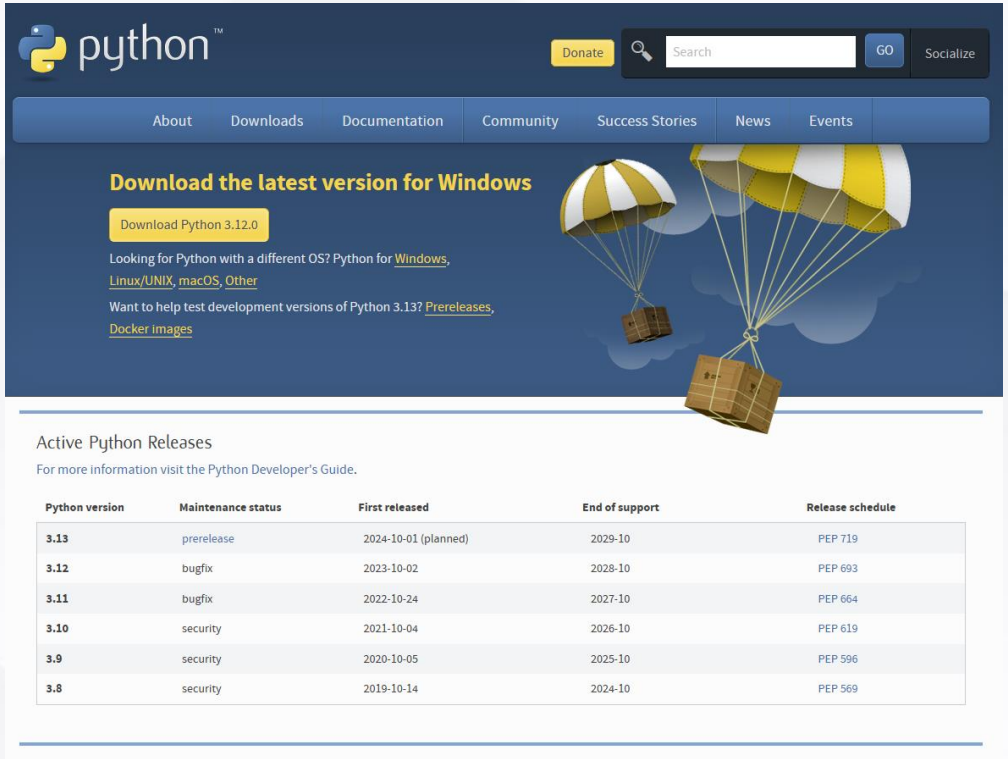
Window安装Python

(1) 访问：

http://www.python.org/download/

(2) 选择版本



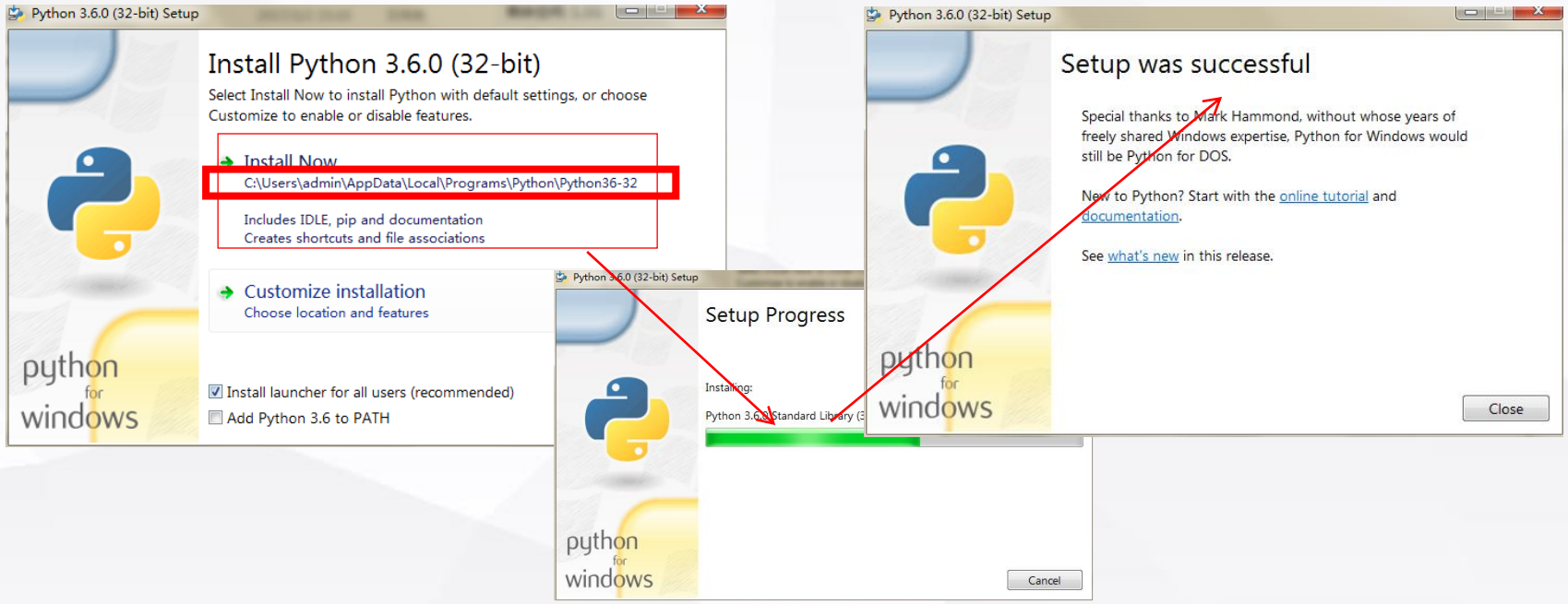


部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

Window安装Python

- Windows
- Mac OS
- Anaconda

(3) 下载并完成Python的安装：



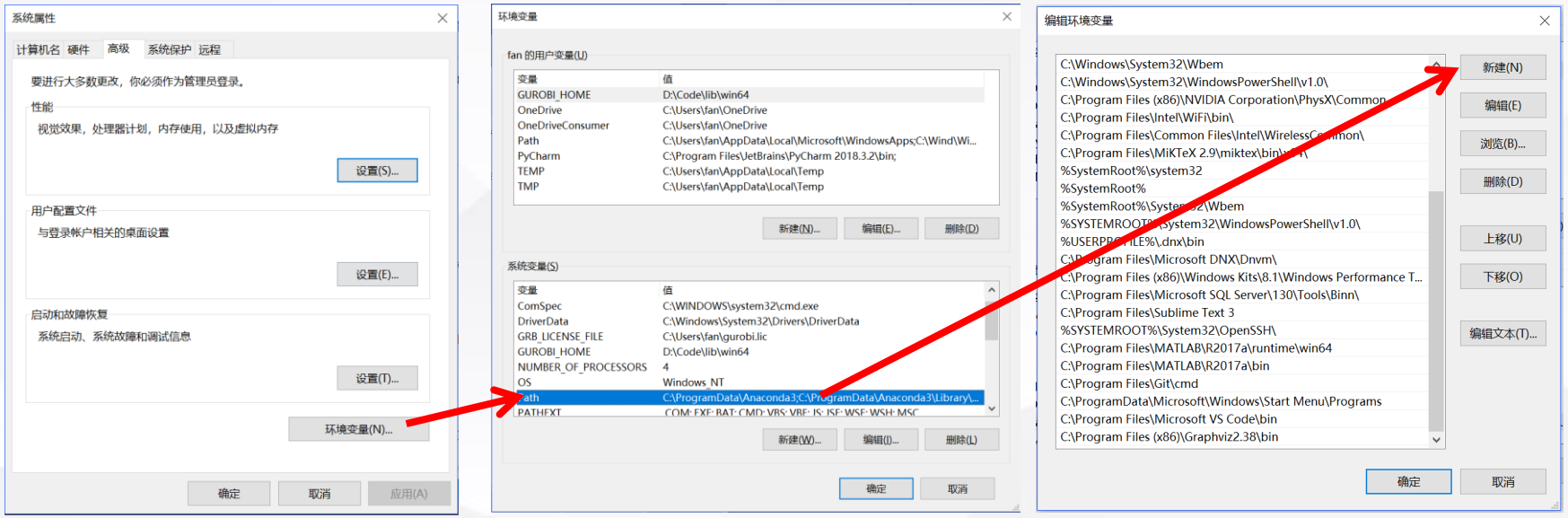
部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

2. 安装Python

Window安装Python

- Windows
- Mac OS
- Anaconda

(4) 配置环境变量： 右键 -> “我的电脑”

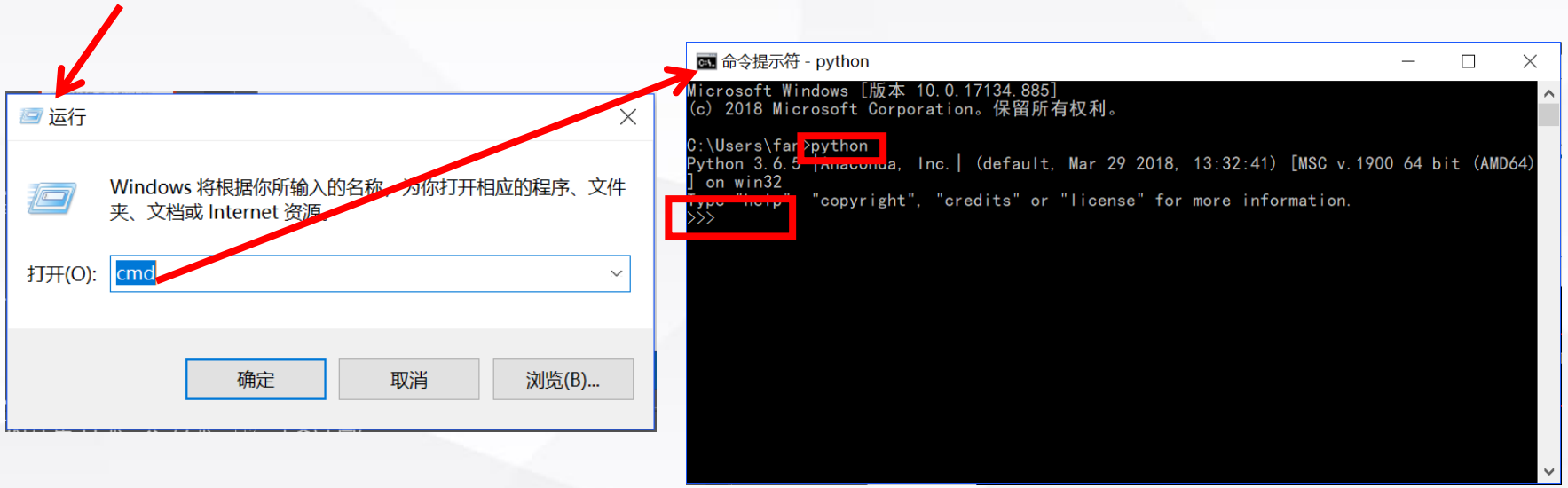


Window安装Python

- Windows
- Mac OS
- Anaconda

(5) 检验环境变量配置：

(Win+R->输入cmd->确定->输入python->Enter)

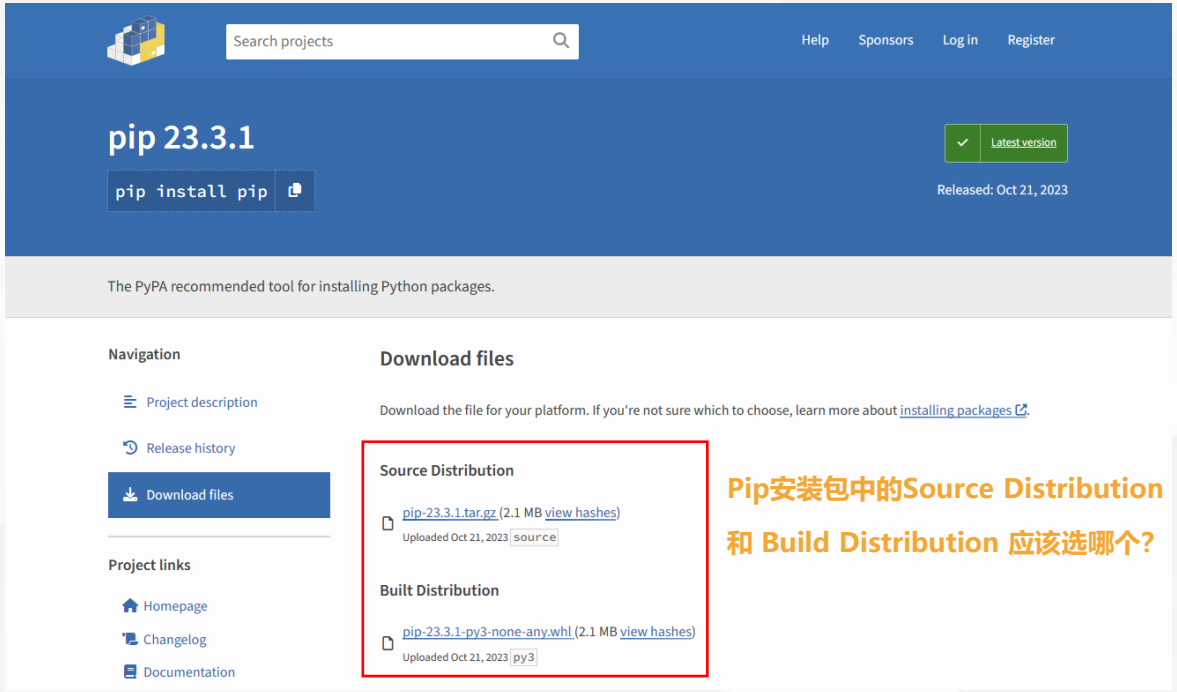


2. 安装Python

Window安装Python

- Windows
- Mac OS
- Anaconda

(6) 下载pip： https://pypi.org/project/pip/#files



Pip安装包中的Source Distribution和 Build Distribution 应该选哪个?

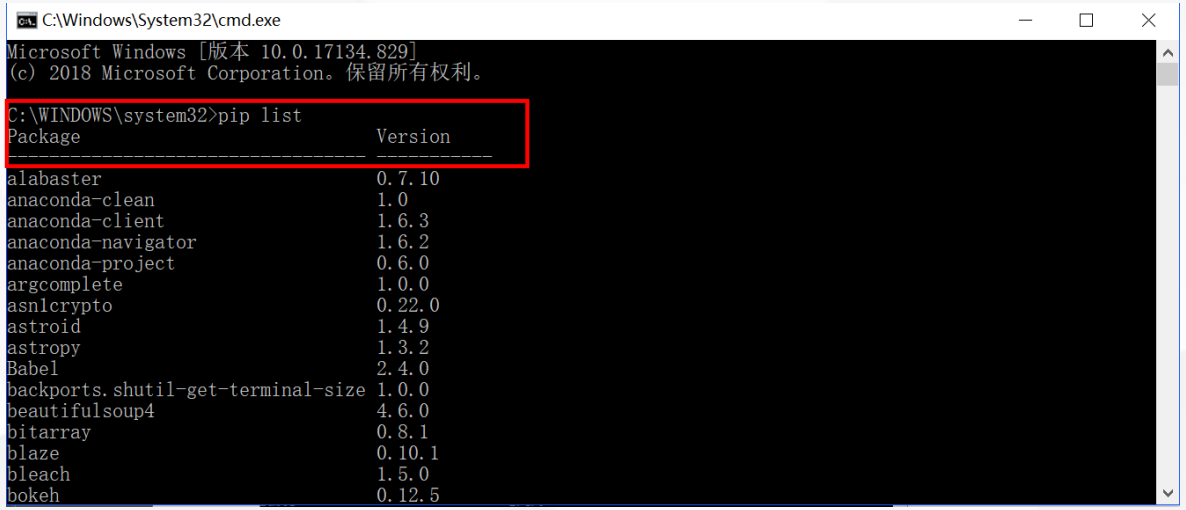
部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

Window安装Python

- Windows
- Mac OS
- Anaconda

(7) 验证pip安装成功：

(Win+R->输入cmd->确定->输入pip list, 出现Package和Version等信息表示pip安装成功)



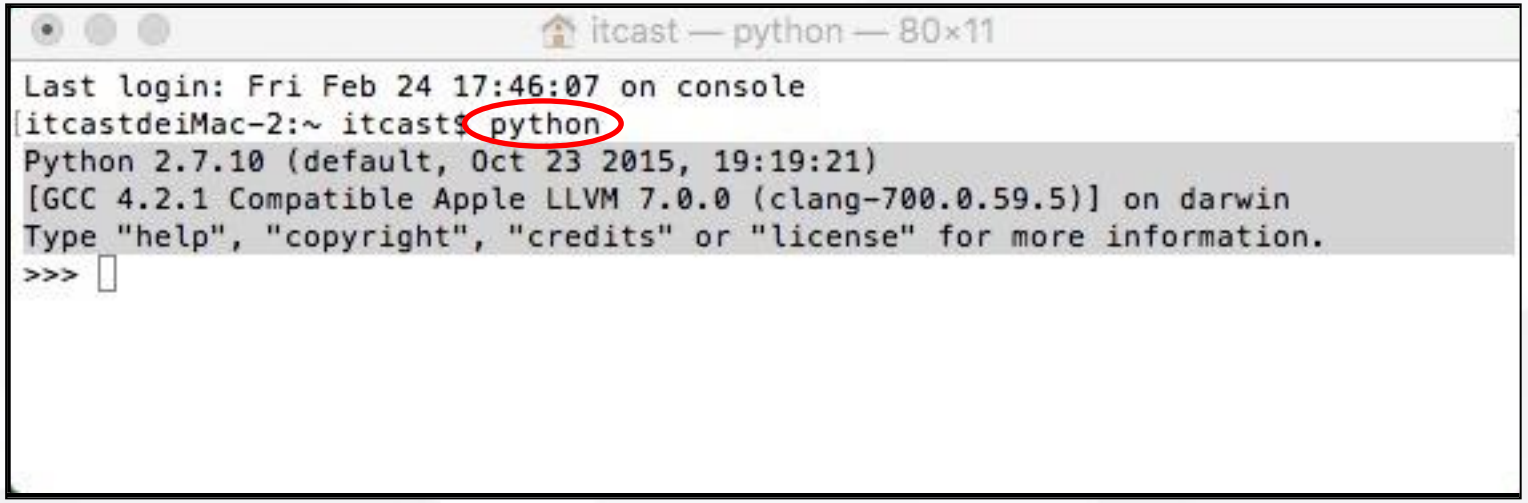
2. 安装Python

东北财经大学 DONGBEI UNIVERSITY OF FINANCE & ECONOMICS

Mac OS安装Python

- Windows
- Mac OS
- Anaconda

Mac系统都自带有Python环境，我们可以在终端输入python命令就可以运行



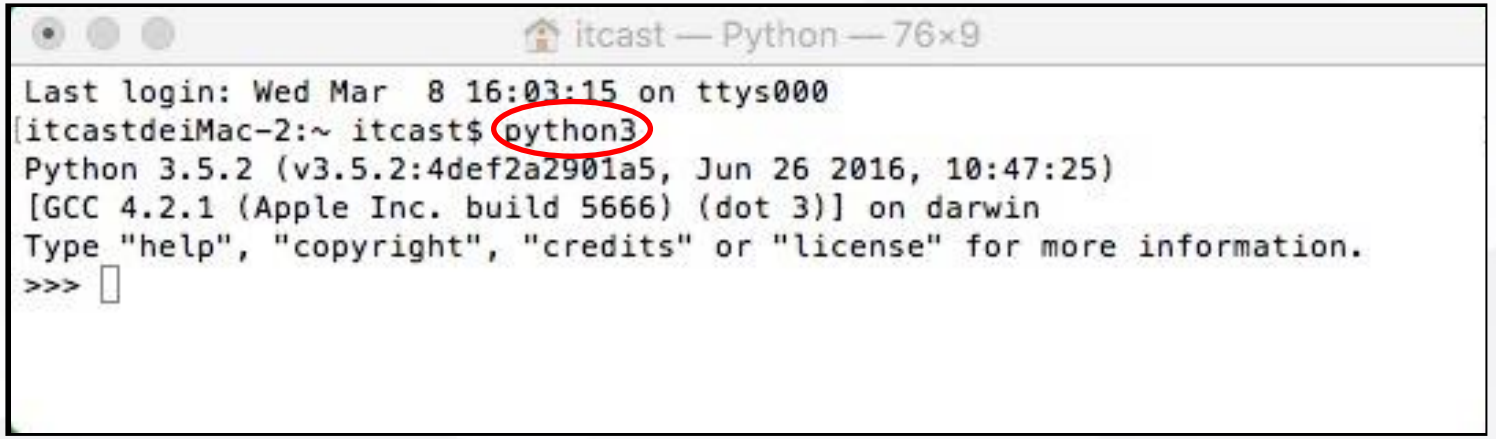
部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

2. 安装Python

Mac OS安装Python

- Windows
- Mac OS
- Anaconda

如果需要安装其他版本的Python，只需要访问网站 http://www.python.org/download/，下载最新版本的dmg文件，双击按照提示完成安装即可



2. 安装Python

Anaconda安装Python

Windows

Mac OS

Anaconda

- 前面的安装方法，**并没有安装丰富的Python工具包**，而且由于**包的依赖关系**和**系统兼容性**问题，很多Python包的安装会很麻烦。
- 对于主要进行数据分析数据挖掘的同学们，**推荐通过Anaconda进行Python安装。**

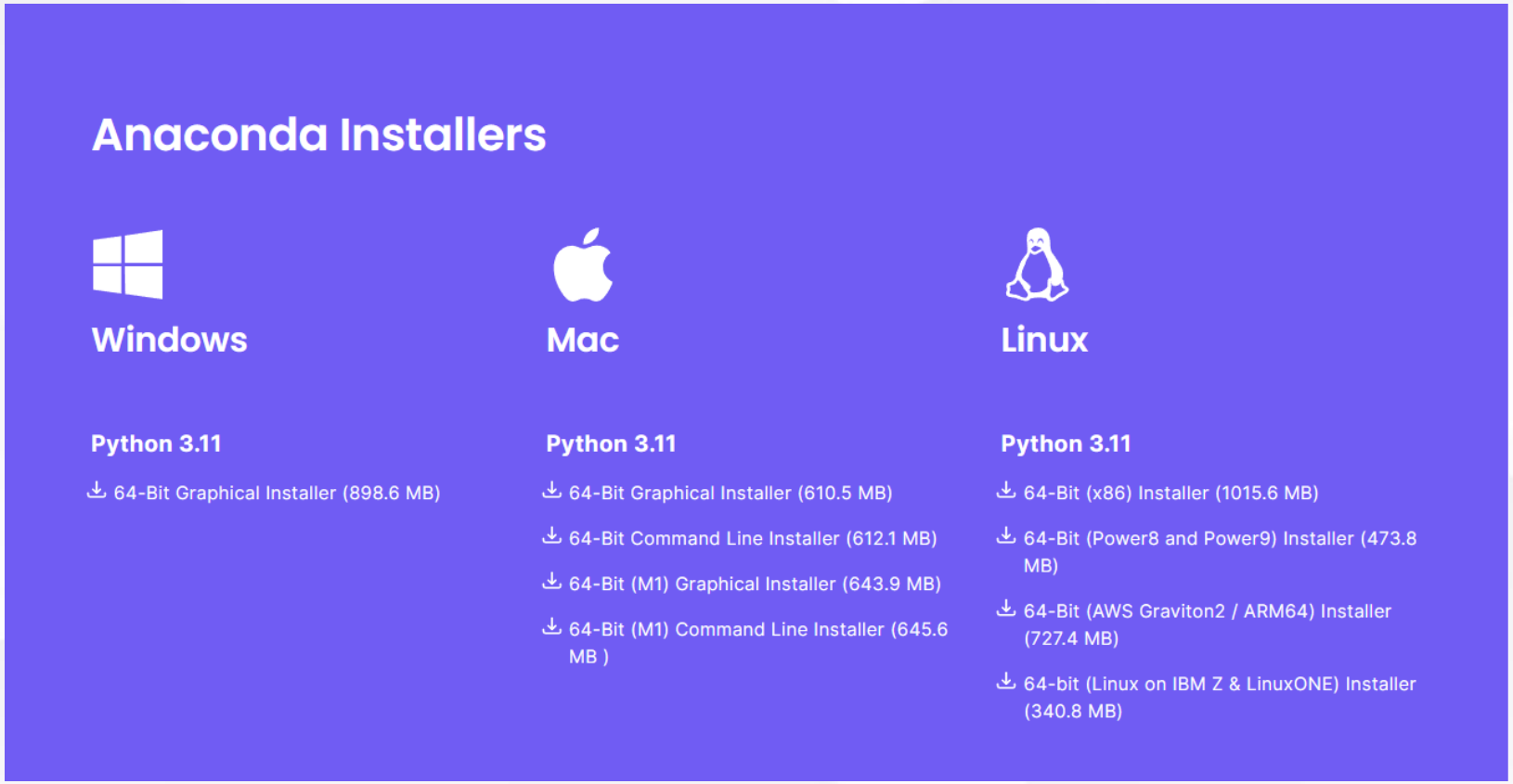
下载地址：https://www.anaconda.com/download

部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

2. 安装Python

Anaconda安装Python

- Windows
- Mac OS
- Anaconda

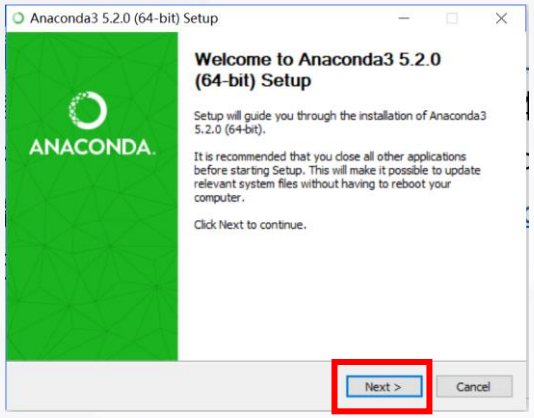


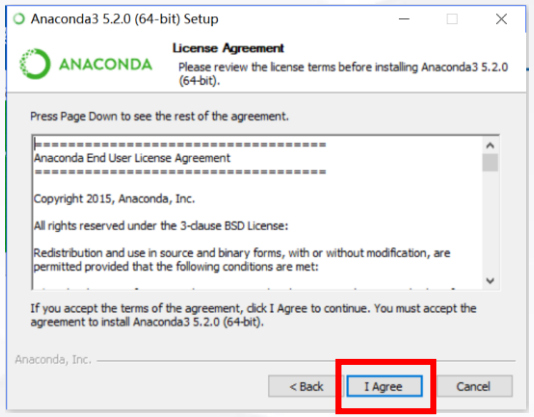
部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

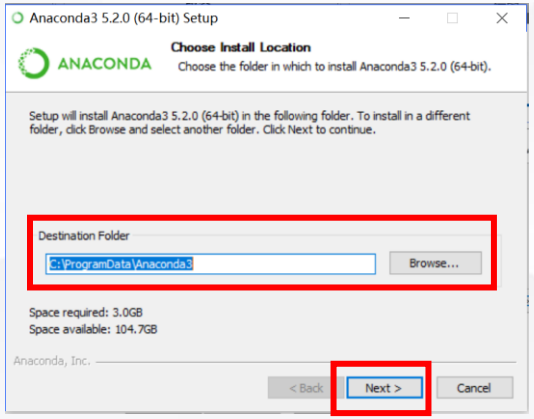
2. 安装Python

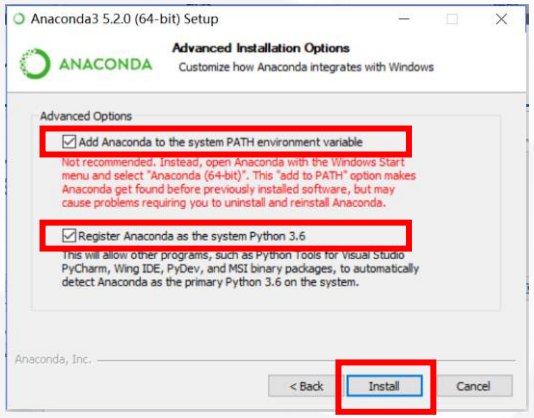
Anaconda安装Python

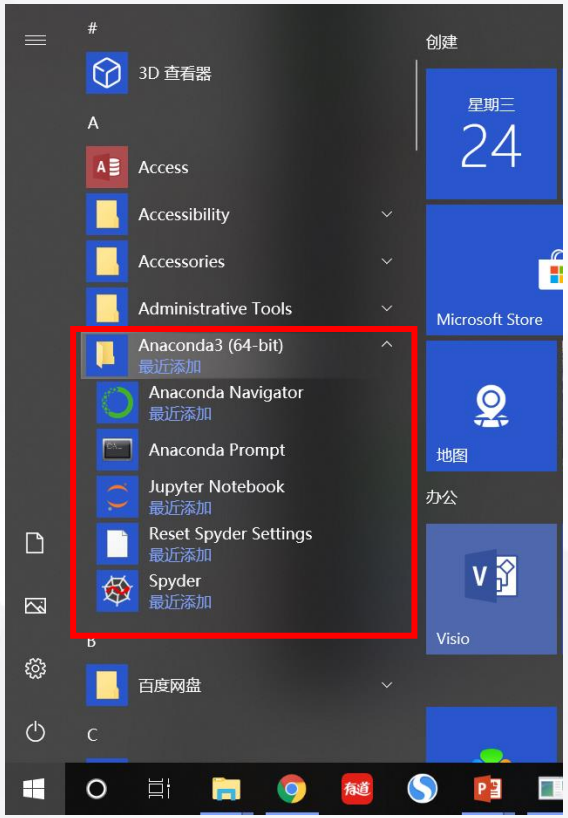
- Windows
- Mac OS
- Anaconda











部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

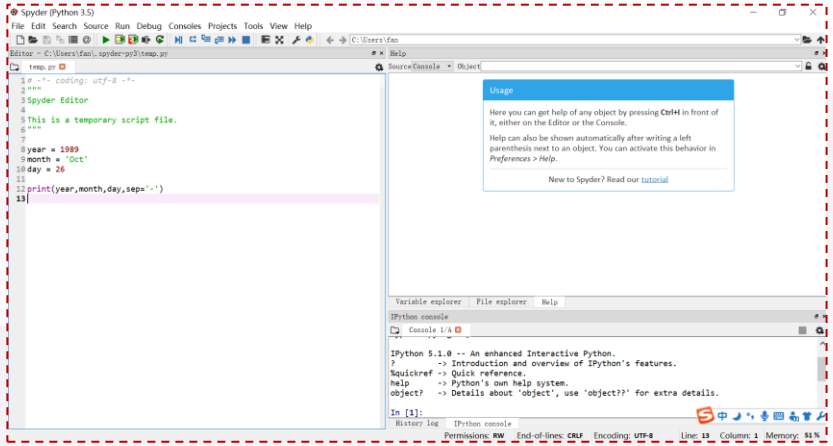
2. 安装Python

Anaconda安装Python

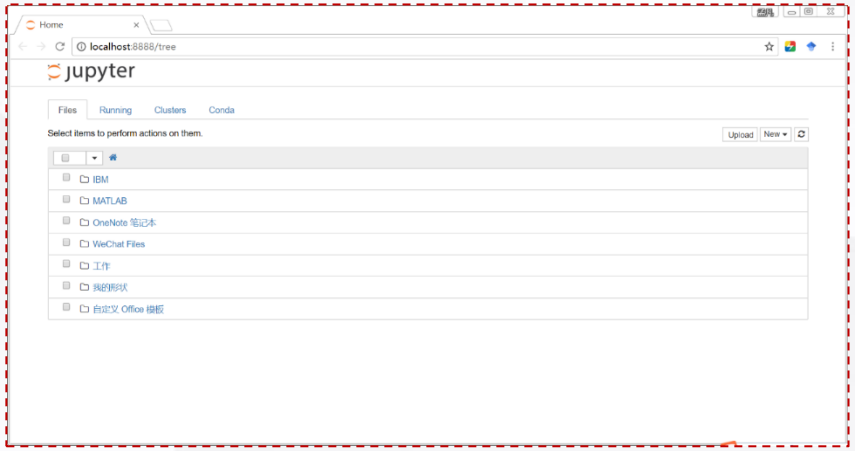
- Windows
- Mac OS
- Anaconda

- 除了丰富的工具包，Anaconda中还集成有丰富的开发工具，包括**集成式**开发环境Spyder，**交互式**开发环境Jupyter。

Spyder



Jupyter Notebook



部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

2. 安装Python

Anaconda安装Python

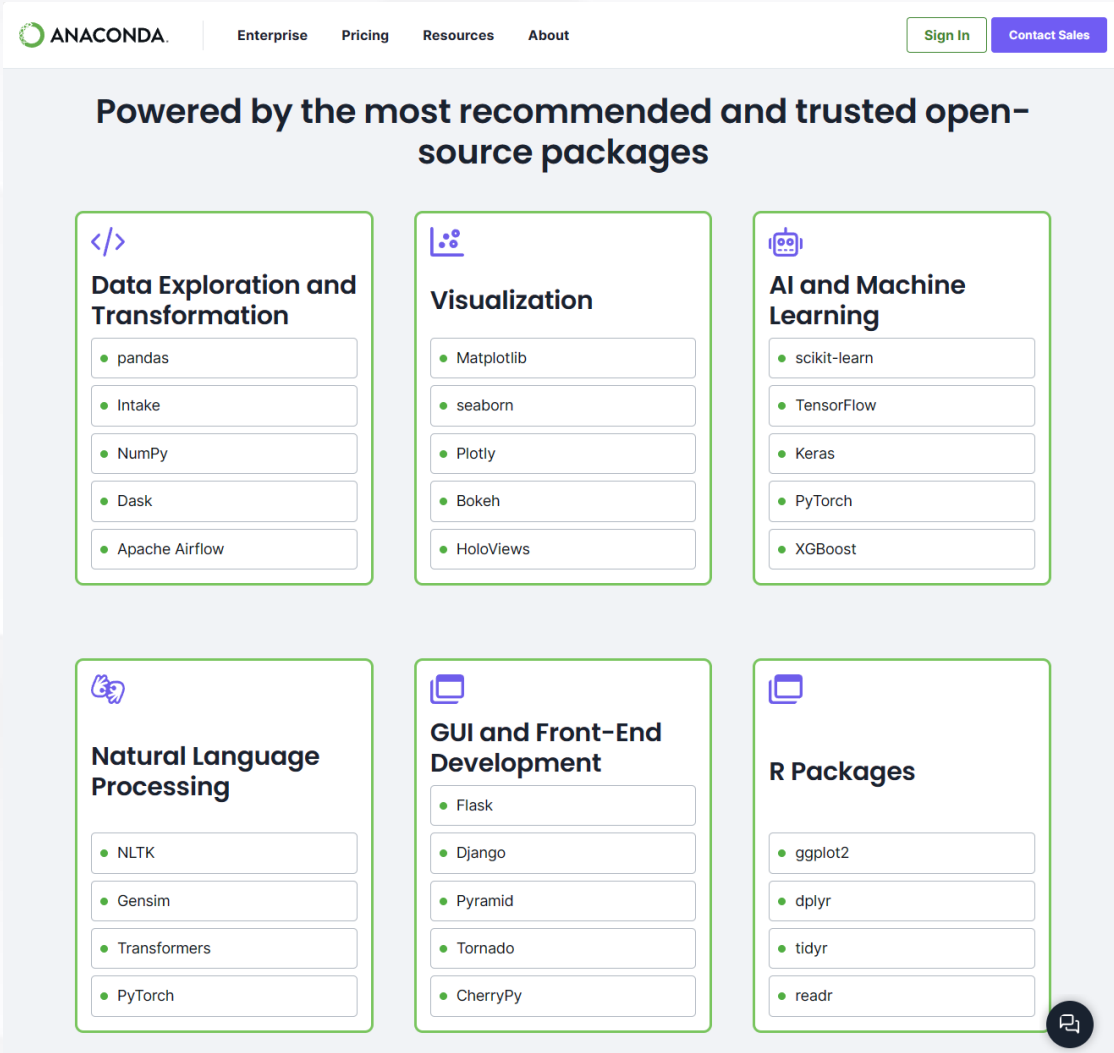
Windows

Mac OS

Anaconda

同时还有很多非常强大的开源包，可以实现：

- 数据分析、数据挖掘、可视化、机器学习模型、自然语言处理，GUI开发等。



部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

2. 安装Python

Windows

Mac OS

Anaconda

Anaconda安装Python

使用conda创建环境：

- conda create -n FinTech python==3.11.4

  Proceed ([y]/n)? -> y

- pip install -r requirements.txt

```
The following NEW packages will be INSTALLED:

  bzip2              pkgs/main/win-64::bzip2-1.0.8-he774522_0
  ca-certificates    pkgs/main/win-64::ca-certificates-2023.08.22-haa95532_0
  libffi             pkgs/main/win-64::libffi-3.4.4-hd77b12b_0
  openssl            pkgs/main/win-64::openssl-3.0.12-h2bbff1b_0
  pip                pkgs/main/win-64::pip-23.3-py311haa95532_0
  python             pkgs/main/win-64::python-3.11.4-he1021f5_0
  setuptools         pkgs/main/win-64::setuptools-68.0.0-py311haa95532_0
  sqlite             pkgs/main/win-64::sqlite-3.41.2-h2bbff1b_0
  tk                 pkgs/main/win-64::tk-8.6.12-h2bbff1b_0
  tzdata             pkgs/main/noarch::tzdata-2023c-h04d1e81_0
  vc                 pkgs/main/win-64::vc-14.2-h21ff451_1
  vs2015_runtime     pkgs/main/win-64::vs2015_runtime-14.27.29016-h5e58377_2
  wheel              pkgs/main/win-64::wheel-0.41.2-py311haa95532_0
  xz                 pkgs/main/win-64::xz-5.4.2-h8cc25b3_0
  zlib               pkgs/main/win-64::zlib-1.2.13-h8cc25b3_0


Proceed ([y]/n)? y


Downloading and Extracting Packages
wheel-0.41.2        | 163 KB    | ######################################################################################## | 100%
openssl-3.0.12      | 7.4 MB    | ##########################################################7                               |  68%
ca-certificates-2023 | 123 KB    | ######################################################################################## | 100%
pip-23.3            | 3.5 MB    | ######################################################################################## | 100%
openssl-3.0.12      | 7.4 MB    | ##########################################################2                               |  71%
```

部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

2. 安装Python

Anaconda安装Python

- Windows
- Mac OS
- Anaconda

4. 使用国内源进行提速

有时候使用pip安装会很慢，此时我们可以设定国内镜像进行提速安装，如下。

- 临时修改

```
pip install -i https://pypi.tuna.tsinghua.edu.cn/simple some-package # 清华源
pip install -i  http://pypi.douban.com/simple some-package #豆瓣镜像
```

比如用国内源码对pip进行升级：

```
pip install -i https://pypi.tuna.tsinghua.edu.cn/simple pip -U
```

部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

2. 安装Python

Anaconda安装Python

Windows

Mac OS

Anaconda

pip install -i https://pypi.tuna.tsinghua.edu.cn/simple some-package # 清华源

pip install -i http://pypi.douban.com/simple some-package #豆瓣镜像

- pip install -i https://pypi.tuna.tsinghua.edu.cn/simple pip -U
- pip install torch===2.1.1 -i https://pypi.tuna.tsinghua.edu.cn/simple

- pip freeze > ./requirements.txt
- pip install –r requirements.txt

部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

2. 安装Python

Anaconda安装Python

- Windows
- Mac OS
- Anaconda

更换清华源

更换为清华源：

```
1 | pip config set global.index-url https://pypi.tuna.tsinghua.edu.cn/simple
```

若要还原，可用下面的命令：

```
1 | pip config unset global.index-url
```

pip config set global.index-url https://pypi.tuna.tsinghua.edu.cn/simple

pip config unset global.index-url

部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

2. 安装Python

东北财经大学 DONGBEI UNIVERSITY OF FINANCE & ECONOMICS

Windows

Mac OS

Anaconda

Pytorch – GPU 配置

Step 1： pip install numba　　Step 2： run test_GPU.py

```
from numba import jit, cuda
import numpy as np
# to measure exec time
from timeit import default_timer as timer

# normal function to run on cpu
def func(a):
    for i in range(10000000):
        a[i]+= 1


# function optimized to run on gpu
@jit(target_backend='cuda')
def func2(a):
    for i in range(10000000):
        a[i]+= 1

if __name__=="__main__":
    n = 10000000
    a = np.ones(n, dtype = np.float64)

    start = timer()
    func(a)
    print("without GPU:", timer()-start)

    start = timer()
    func2(a)
    print("with GPU:", timer()-start)
```

```
@jit(target_backend='cuda')
without GPU: 4.020814199990127
with GPU: 0.4144971999921836
```

Step 3：

```
import torch
torch.cuda.is_available()
```

如果显示以下界面，表示Pytorch-GPU配置成功！

```
torch.cuda.is_available()
✓ 1.4s
True
```

如果显示为False，

```
torch.cuda.is_available()
✓ 0.0s
False
```

在anaconda/envs/FinTech/Lib/sitepackages中删掉：

1. torch
2. torch-2.1.1.dist-info
3. torchvision
4. torchvision-0.16.1.dist-info. 然后继续Step4.

2. 安装Python

东北财经大学 DONGBEI UNIVERSITY OF FINANCE & ECONOMICS

Windows

Mac OS

Anaconda

Pytorch – GPU 配置

Step 4： cmd -> NVIDIA-smi

```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 529.08       Driver Version: 529.08       CUDA Version: 12.0     |
|-------------------------------+----------------------+----------------------+
| GPU  Name            TCC/WDDM | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA RTX A200... WDDM  | 00000000:01:00.0 Off |                  N/A |
| N/A   48C    P0    12W /  39W |      0MiB /  4096MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

Step 5: https://pytorch.org/get-started/locally/#no-cuda-1

PyTorch Build	Stable (2.1.1)		Preview (Nightly)	
Your OS	Linux	Mac	Windows	
Package	Conda	Pip	LibTorch	Source
Language	Python		C++ / Java	
Compute Platform	CUDA 11.8	CUDA 12.1	~~ROCm 5.6~~	CPU
Run this Command:	pip3 install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu118			

Step 6： 复制这个命令，到Anaconda Prompt

1. conda activate FinTech
2. 粘贴命令，回车（1Mb/s的下载速度，时间约为50分钟++）

程序会下载以下三个包：

```
Looking in indexes: https://download.pytorch.org/whl/cu118
Collecting torch
  Downloading https://download.pytorch.org/whl/cu118/torch-2.1.1%2Bcu118-cp311-cp311-win_amd64.whl (2722.7 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.7/2.7 GB 632.4 kB/s eta 0:00:00
Collecting torchvision
  Downloading https://download.pytorch.org/whl/cu118/torchvision-0.16.1%2Bcu118-cp311-cp311-win_amd64.whl (4.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.9/4.9 MB 1.2 MB/s eta 0:00:00
Collecting torchaudio
  Downloading https://download.pytorch.org/whl/cu118/torchaudio-2.1.1%2Bcu118-cp311-cp311-win_amd64.whl (3.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.9/3.9 MB 1.2 MB/s eta 0:00:00
```

显示安装成功.

```
Installing collected packages: torch, torchvision, torchaudio
Successfully installed torch-2.1.1+cu118 torchaudio-2.1.1+cu118 torchvision-0.16.1+cu118
```

最后重复Step3，验证 torch.cuda.is_available()==True.

Step 7: 更新Numpy包，库之间的版本冲突：

```
pip install --upgrade numpy==1.26 -i  https://pypi.tuna.tsinghua.edu.cn/simple
```

2. 安装Python

个人作业 – 需要以下包

Windows

Mac OS

Anaconda

安装命令：pip install folium pandas

```
import folium
import folium.plugins as plugins
import pandas as pd

from folium.plugins import HeatMapWithTime
```

安装命令：pip install numpy matplotlib torch torchvision

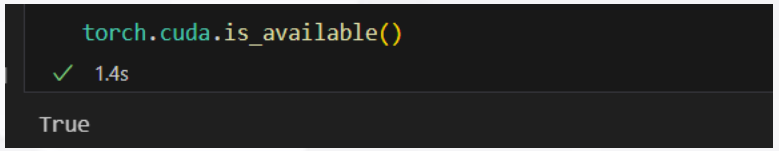
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

import torch
import torch.nn as nn
import torch.optim as optim
import torch.nn.functional as F
from torch.utils.data import DataLoader
import torchvision.datasets as datasets
import torchvision.transforms as transforms
```

Pytorch-GPU安装参考Step5，运行以下命令检测

```
torch.cuda.is_available()
```

如果显示以下界面，表示Pytorch-GPU配置成功！



数据挖掘常用工具包

Python进行数据挖掘的常用工具包主要有：

- **NumPy**
- **Pandas**
- **matplotlib**
- **statsmodels**
- **SciPy**
- **scikit-learn**
- **……**

部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

Pandas

Pandas提供丰富的数据结构和功能，旨在使结构化数据快速、简单、富有表现力。它是使Python成为一个强大且高效的数据分析环境的关键因素之一。Pandas基于两种数据类型：Series与DataFrame。

- Series是一个一维的数据类型，其中每一个元素都有一个标签。Series类似于Numpy中元素带标签的数组。其中，标签可以是数字或者字符串。
- DataFrame是一个二维的表结构。Pandas的DataFrame可以存储许多种不同的数据类型，并且每一个坐标轴都有自己的标签。

部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

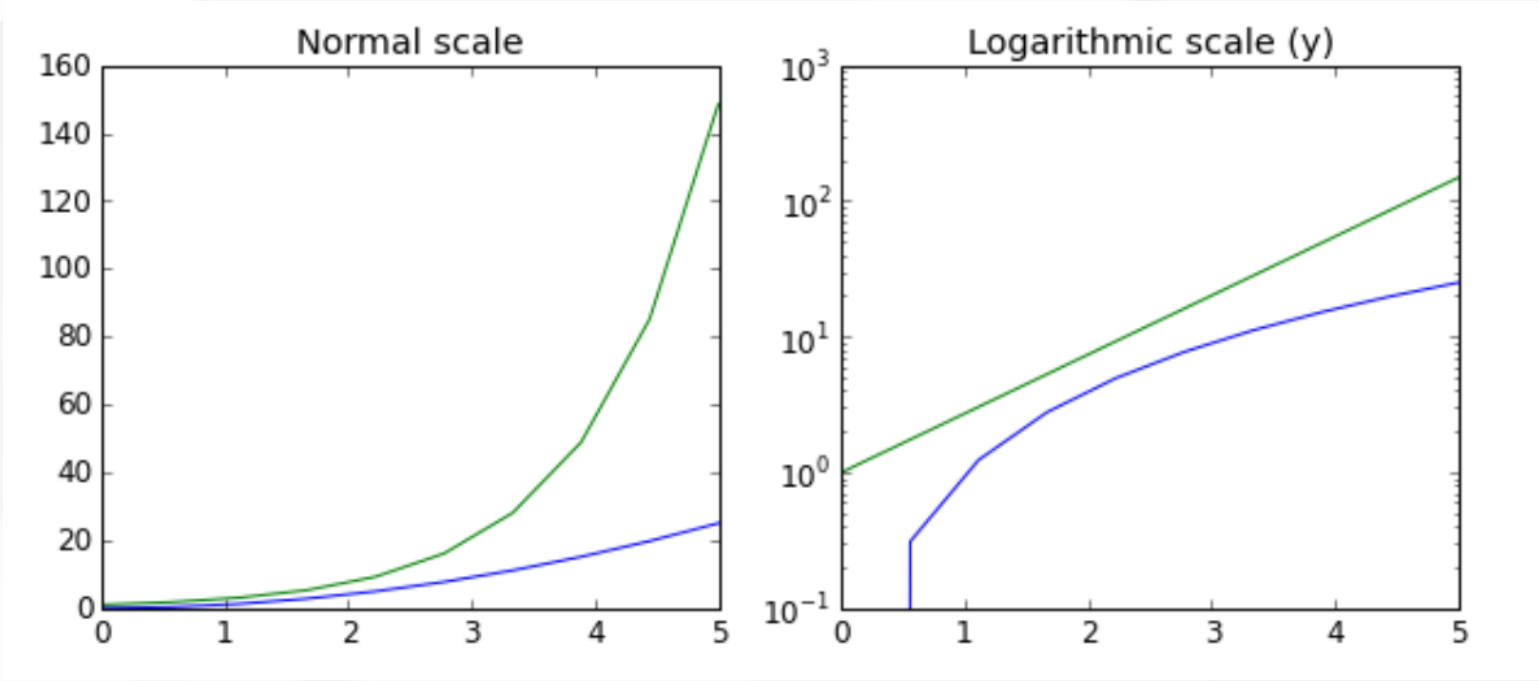
matplotlib

matplotlib是最流行的用于生成绘图和其他2D和3D数据可视化的Python库。它最初是由John d . Hunter(JDH)创建的，现在由一个大型开发团队维护。它非常适合于创建用于发布和展示的图形。

它与IPython集成得很好，从而为绘图和探测数据提供了一个舒适的交互式环境。这些图也具有互动性；可以在plot窗口中使用工具栏来放大图的一部分。

2. Python常用包介绍

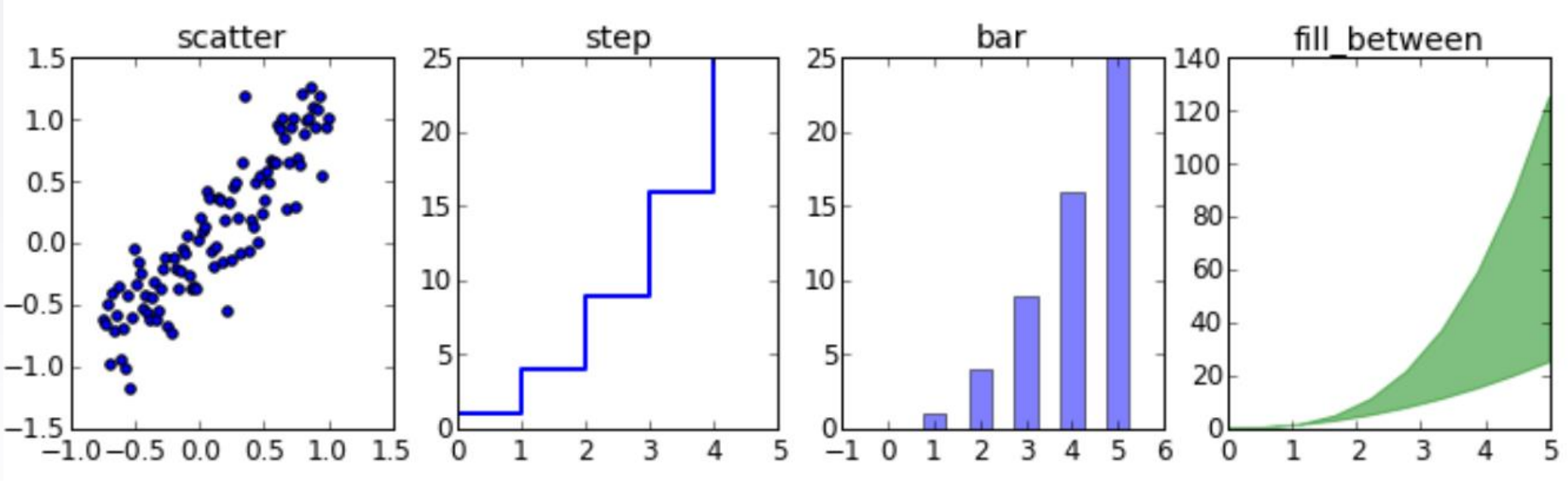
matplotlib



部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

2. Python常用包介绍

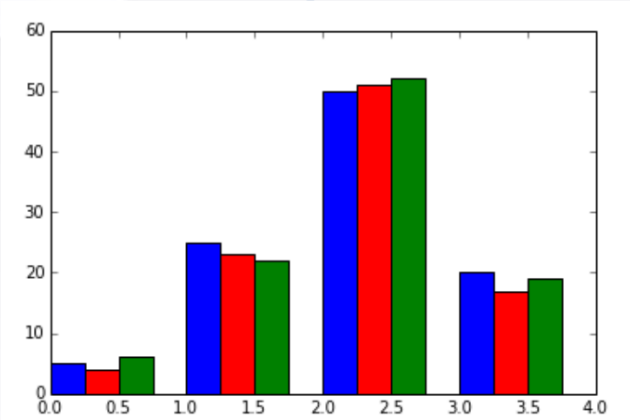
matplotlib

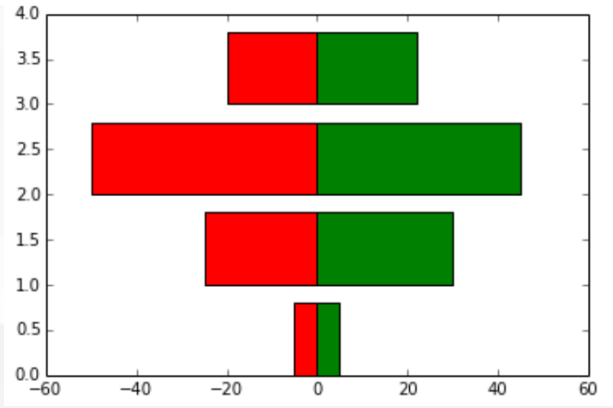


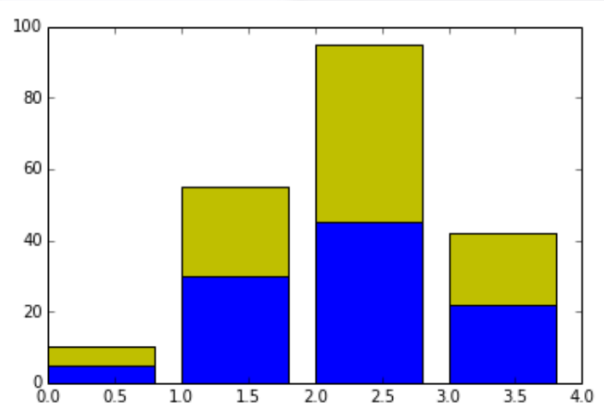
部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

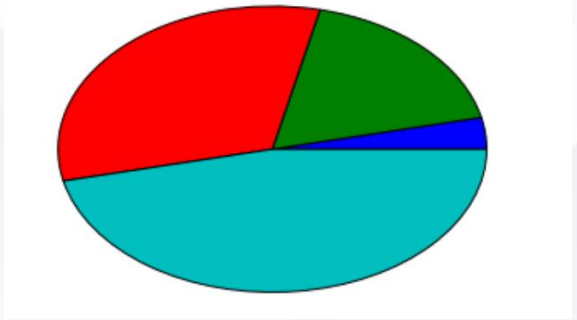
2. Python常用包介绍

matplotlib





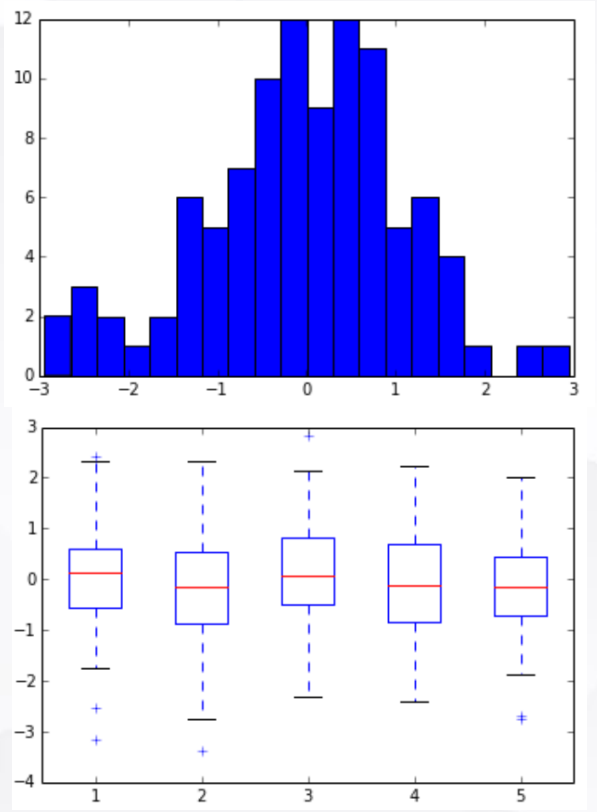


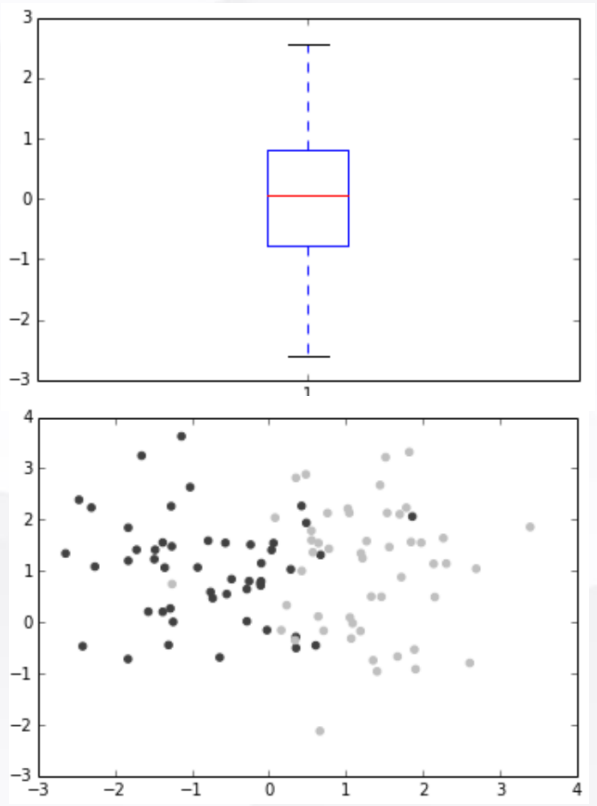


部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

2. Python常用包介绍

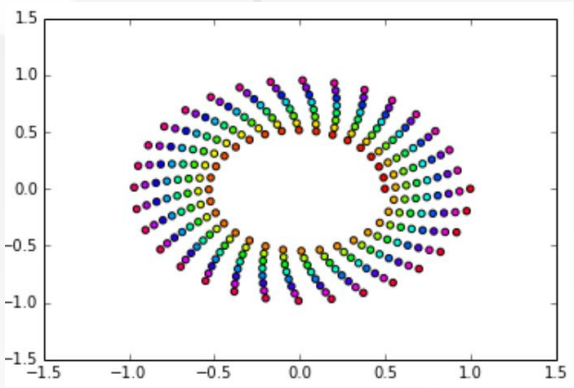
matplotlib

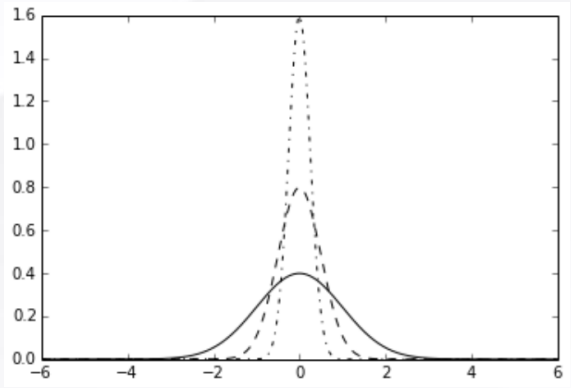


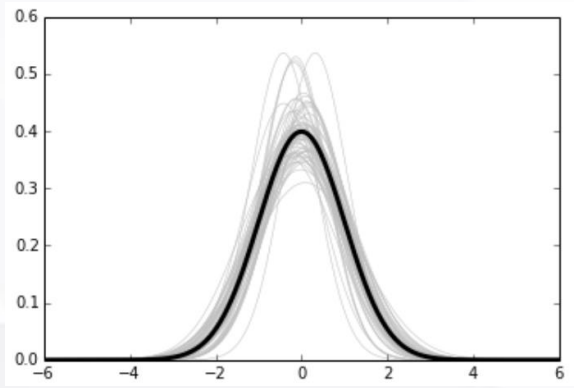


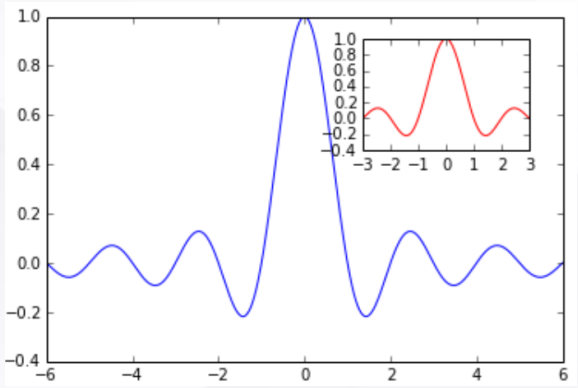
2. Python常用包介绍

matplotlib









部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

statsmodels

Statsmodels是Python的统计建模和计量经济学工具包，包括一些描述统计、统计模型估计和推断。主要功能包括：

- Liner regression models：线性回归模型
- Gneralized linear models：一般线型模型
- Robust linear models：鲁棒线性模型
- Discrete choice models：离散选择模型
- ANOVA：方差分析模型
- Time series analysis：时间序列分析
- Nonparametric estimators：非参检验
- a wide range of statistical tests：各种统计检验
- 以各种方式输出表格：text，latex，html；读取各种格式的数据
- 绘图功能

SciPy

SciPy 是基于Numpy构建在科学计算中处理多个不同标准问题域的包的集合。主要包括以下模块包括：

- scipy.integrate: 数值积分和微分方程求解器
- scipy.linalg: 拓展了numpy.linalg中的线性代数和矩阵分解功能
- scipy.optimize:函数优化器(最小化器)和根查找算法
- scipy.signal: 信号处理工具
- scipy.sparse: 系数矩阵和线性系统求解器
- scipy.special: 对于SPECFUN的封装， SPECFUN库实现了许多常见的数学函数
- scipy.stats: 标准连续和离散概率分布(密度函数，采样器，连续分布函数)，各种统计检验，和更多的描述性统计
- scipy.weave: 使用内联c++代码来加速数组计算的工具

通过结合使用NumPy和SciPy能够实现绝大部分matlab及其工具包的功能。

部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

2. Python常用包介绍

scikit-learn

scikit-learn是Python的一个开源机器学习模块，它建立在NumPy，SciPy和matplotlib模块之上，实现了大量的机器学习算法。包括：

- **Classification**：分类 - SVM, nearest neighbors, random forest, logistic regression, etc.
- **Regression**：回归 - Lasso, ridge regression, etc.
- **Clustering**：聚类 - *k*-means, spectral clustering, etc.
- **Dimensionality** reduction：降维 - PCA, feature selection, matrix factorization, etc.
- **Model selection**：模型选择 - Grid search, cross-validation, metrics
- **Preprocessing**：预处理 - Feature extraction, normalization

部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

2. Python常用包介绍

scikit-learn

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.
Algorithms: SVM, nearest neighbors, random forest, …

— Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, ridge regression, Lasso, …

— Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, …

— Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency
Algorithms: PCA, feature selection, non-negative matrix factorization.

— Examples

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning
Modules: grid search, cross validation, metrics.

— Examples

Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.
Modules: preprocessing, feature extraction.

— Examples

部分课件改写自中国大学慕课《数据挖掘与Python实践》- 中央财经大学，李爱华

3. Numpy-介绍

Numpy 简介

Numpy

Pandas

Matplotlib

- Numpy是Python用于科学计算的基础包，也是大量Python数学和科学计算包的基础，不少数据处理及分析包都是在Numpy基础上开发的，比如后面介绍的pandas包就是在其基础上开发的。
- Numpy的核心基础是ndarray（N-dimensional array，N维数组), 即由数据类型相同的元素组成的N维数组。
- 可利用Numpy包提供的数组定义函数array()将数据转化为数组的形式

部分课件改写自《Python金融数据分析与挖掘实战》- 黄桓秋

3. Numpy-介绍

Numpy 简介

基于array()函数，可以将列表、元组、嵌套列表、嵌套元组等给定的数据结构转化为数组，值得注意的是利用array函数之前，事先要导入Numpy

- Numpy
- Pandas
- Matplotlib

示例代码如下：

```
#1.先预定义列表d1，元组d2，嵌套列表d3、d4和嵌套元组d5
d1=[1,2,3,4,0.1,7]          #列表
d2=(1,2,3,4,2.3)            #元组
d3=[[1,2,3,4],[5,6,7,8]]    #嵌套列表，元素为列表
d4=[(1,2,3,4),(5,6,7,8)]    #嵌套列表，元素为元组
d5=((1,2,3,4),(5,6,7,8))    #嵌套元组

#2.导入Numpy，并调用其中的array函数，创建数组
import numpy as np
d11=np.array(d1)
d21=np.array(d2)
d31=np.array(d3)
d41=np.array(d4)
d51=np.array(d5)
```

部分课件改写自《Python金融数据分析与挖掘实战》- 黄桓秋

3. Numpy-介绍

- Numpy
- Pandas
- Matplotlib

Numpy 简介

利用内置函数，可以创建一些特殊的数组.

np.ones(n,m) -> 创建n行m列元素全为1的数组、
np.zeros(n,m) -> 创建n行m列元素全为0的数组，
np.arange(a,b,c) -> 创建以a为初始值，(b-1)为末值，c为步长的一维数组。
其中a和c参数可省，默认值：a=0，c=1。

示例代码如下

```
z1=np.ones((3,3))        #创建3行3列元素全为1的数组
z2=np.zeros((3,4))       #创建3行4列元素全为0的数组
z3=np.arange(10)         #创建默认初始值为0，默认步长为1，末值为9的一维数组
z4=np.arange(2,10)      #创建默认初始值为2，默认步长为1，末值为9的一维数组
z5=np.arange(2,10,2)  #创建默认初始值为2，步长为2，末值为9的一维数组
```

部分课件改写自《Python金融数据分析与挖掘实战》- 黄桓秋

3. Numpy-介绍

- Numpy
- Pandas
- Matplotlib

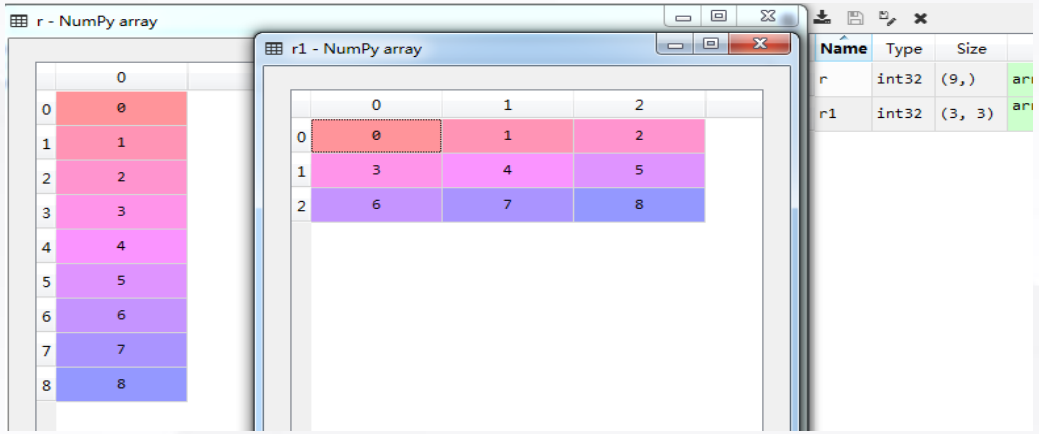
Numpy 简介

数组尺寸，也称为数组的大小，通过行数和列数来表现。

np.shape -> 返回数组的尺寸，其返回值为元组, e.g.(2,5)。
如果是一维数组，返回的元组中仅一个元素，代表这个数组的长度。
如果是二维数组，元组中有两个值，第一个值为行数，第二个值为列数。

在程序应用过程中，有时候需要将数组进行重排
可以通过np.reshape()函数来实现：

```
r=  np.array(range(9))  #一维数组
r1=r.reshape((3,3))   #重排为3行3列
```



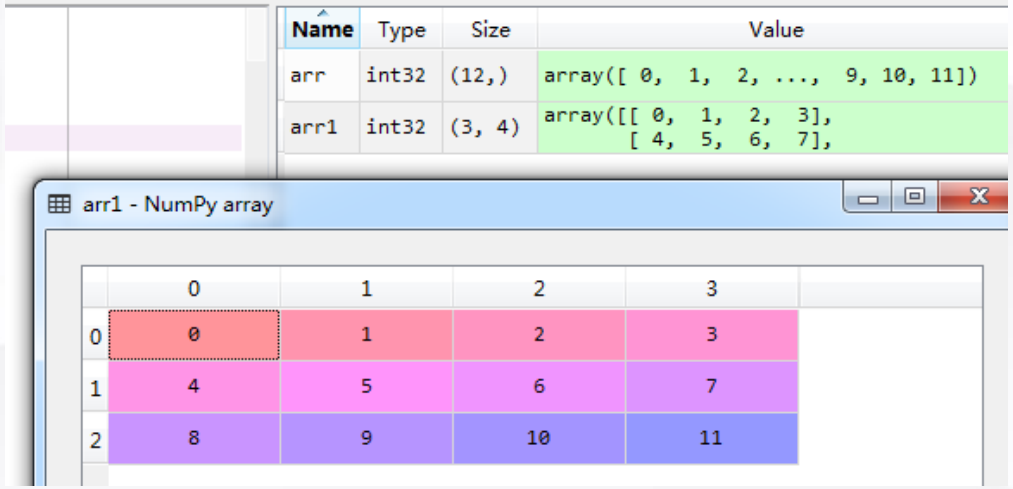
部分课件改写自《Python金融数据分析与挖掘实战》- 黄桓秋

- Numpy
- Pandas
- Matplotlib

Numpy 简介

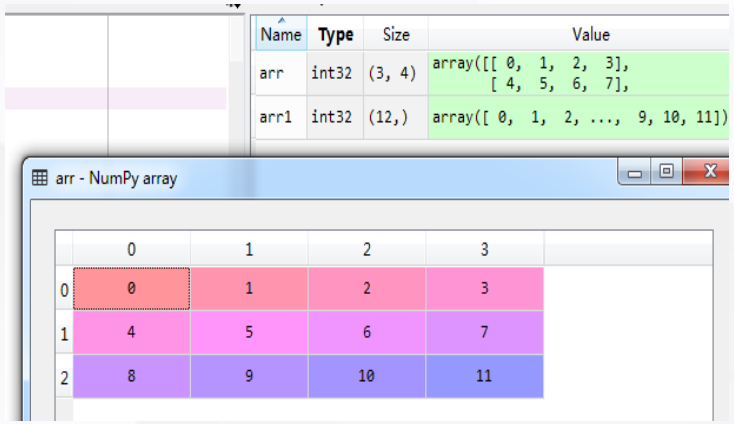
Numpy提供了reshape方法用于改变数组的形状，reshape方法仅改变原始数据的形状，不改变原始数据的值。示例代码如下：

```
import numpy as np
arr = np.arange(12)  # 创建一维ndarray
arr1 = arr.reshape(3, 4)  # 设置ndarray的维度，改变其形态
```



将二维数组形态展平变换为一维数组，通过ravel()函数即可实现。示例代码如下：

```
import numpy as np
arr = np.arange(12).reshape(3, 4)
arr1=arr.ravel()
```



部分课件改写自《Python金融数据分析与挖掘实战》- 黄桓秋

3. Numpy-介绍

Numpy 简介

数学运算

- Numpy
- Pandas
- Matplotlib

```
D=np.array([[1,2,3,4],[5,6,7,8],[9,10,11,12],[13,14,15,16]])  #定义数组D

#数学运算
E1=np.sqrt(D)            #数组D中所有元素取平方根，结果赋给变量E1
E2=np.abs([1,-2,-100])    #取绝对值
E3=np.cos([1,2,3])        #取cos值
E4=np.sin(D)              #取sin值
E5=np.exp(D)              #取指数函数值
```

3. Numpy-介绍

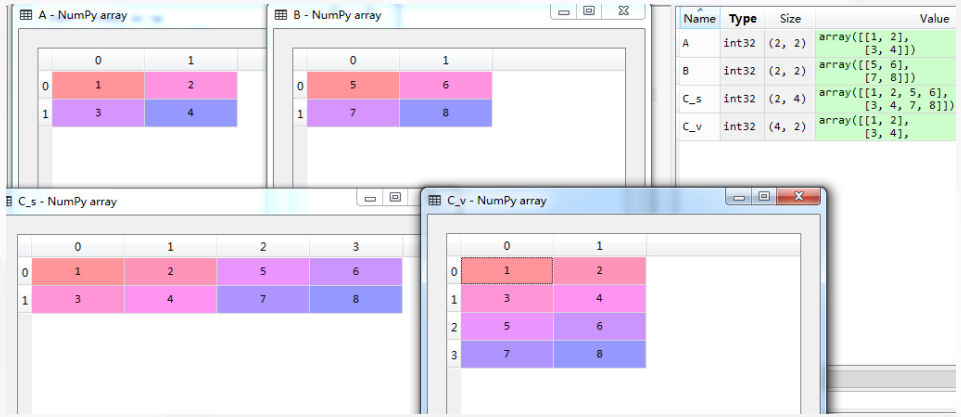
Numpy 简介

在数据处理中，多个数据源的集成整合是经常发生的。
水平连接函数用hstack()、
垂直连接函数用vstack()实现。注意输入参数为两个待连接数组组成的元组。

- Numpy
- Pandas
- Matplotlib

```
import numpy as np

A=np.array([[1,2],[3,4]])      #定义二维数组A
B=np.array([[5,6],[7,8]])      #定义二维数组B
C_s=np.hstack((A,B))           #水平连接要求行数相同
C_v=np.vstack((A,B))           #垂直连接要求列数相同
```



部分课件改写自《Python金融数据分析与挖掘实战》- 黄桓秋

3. Numpy-介绍

- Numpy
- Pandas
- Matplotlib

Numpy 简介

通过Numpyt提供的sort函数，可以对数组元素值按从小到大进行直接排序，示例代码如下：

```
import numpy as np
arr = np.array([5,2,3,3,1,9,8,6,7])
arr1=np.sort(arr)
```

Name	Type	Size	Value
arr	int32	(9,)	array([5, 2, 3, 3, 1, 9, 8, 6, 7])
arr1	int32	(9,)	array([1, 2, 3, 3, 5, 6, 7, 8, 9])

通过Numpy提供的argmax和argmin函数，可以返回待搜索数组最大值和最小值元素的索引值，如果存在多个最大值或最小值，则返回第一次出现的索引。对于二维数组而已，可以通过设置axis=0或1返回各列或者各行最大值或最小值索引。需要注意的是，索引从0开始。

```
import numpy as np
arr = np.array([5,2,3,3,1,1,9,8,6,7,8,8])
arr1=arr.reshape(3,4)
maxindex = np.argmax(arr)
minindex  = np.argmin(arr)
maxindex1=np.argmax(arr1,axis=0) #返回各列最大值索引
minindex1=np.argmin(arr1,axis=1)  #返回各行最小值索引
```

Name	Type	Size	Value
arr	int32	(12,)	array([5, 2, 3, ..., 7, 8, 8])
arr1	int32	(3, 4)	array([[5, 2, 3, 3], [1, 1, 9, 8],
maxindex	int64	1	6
maxindex1	int64	(4,)	array([2, 2, 1, 1], dtype=int64)
minindex	int64	1	4
minindex1	int64	(3,)	array([1, 0, 0], dtype=int64)

arr1 - NumPy array

	0	1	2	3
0	5	2	3	3
1	1	1	9	8
2	6	7	8	8

部分课件改写自《Python金融数据分析与挖掘实战》- 黄桓秋

3. Pandas-介绍

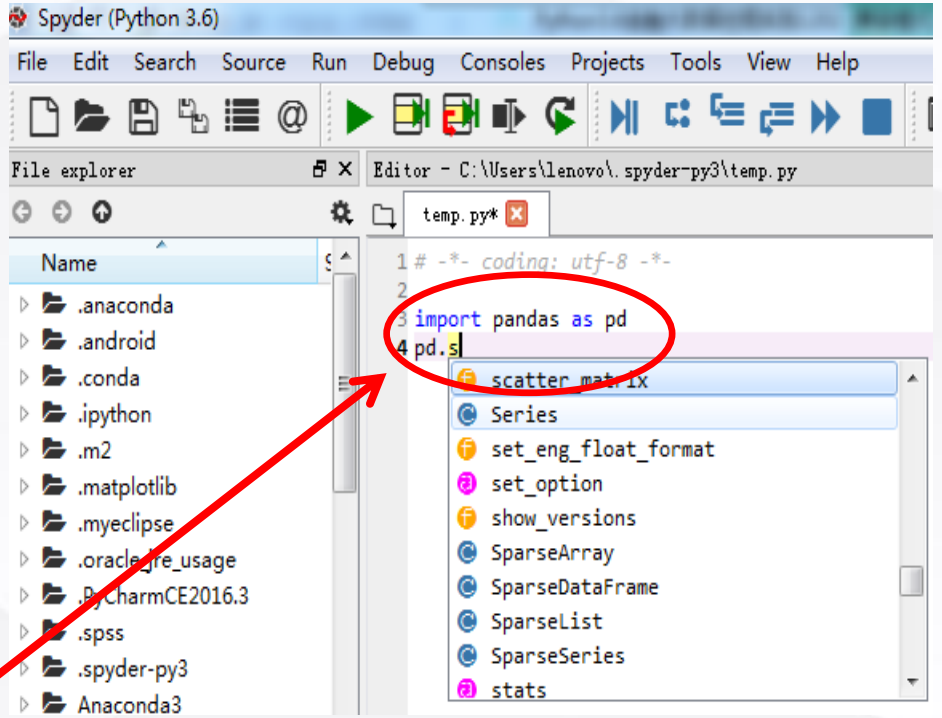
Numpy

Pandas

Matplotlib

Pandas 简介

- Pandas 是基于Numpy开发的一个Python数据分析包，由AQR Capital Management于2008年4月开发，并于2009年底开源出来。
- Pandas作为Python数据分析的核心包，提供了大量的数据分析函数，包括数据处理、数据抽取、数据集成、数据计算等基本的数据分析手段。
- Pandas核心数据结构包括**序列**和**数据框**，序列储存一维数据，而数据框则可以存储更复杂的多维数据。这里主要介绍二维数据（类似于数据表）及其相关操作。
- 在Anaconda发行版中，Pandas包已经集成在系统中，无需另外安装。在使用过程中直接导入该包即可，导入方法为import pandas as pd。



部分课件改写自《Python金融数据分析与挖掘实战》- 黄桓秋

3. Pandas-介绍

Pandas 核心数据结构1 – 序列 (Series)

- 序列是Pandas中非常重要的一个数据结构，由两部分组成，一部分是索引index，一部分是对应的值。
- 序列对象的创建通过Pandas包中的Series()函数来实现。

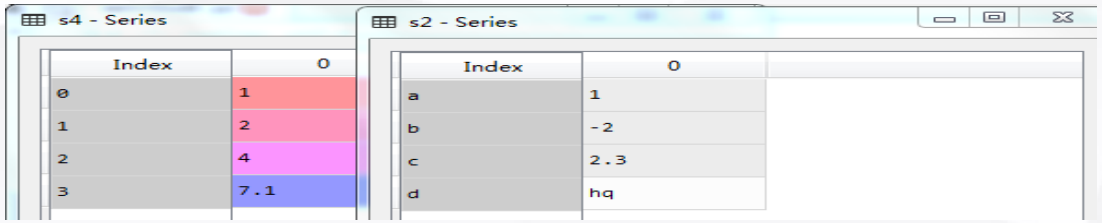
Numpy

Pandas

Matplotlib

```
import pandas as pd          #导入Pandas库
import numpy as np           #导入Numpy库

s1=pd.Series([1,-2,2.3,'hq'])                          #指定列表创建默认序列
s2=pd.Series([1,-2,2.3,'hq'],index=['a','b','c','d'])  #指定列表和索引，创建个性化序列
s3=pd.Series((1,2,3,4,'hq'))                           #指定元组创建默认序列
s4=pd.Series(np.array([1,2,4,7.1]))                    #指定数组创建默认序列
```



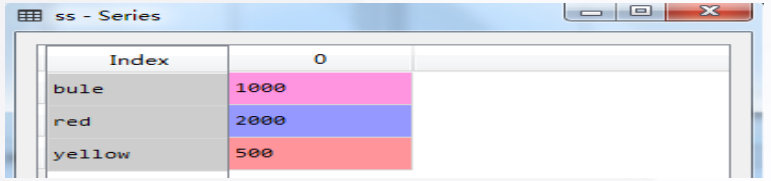

```
print(s1[3])  ->  hq
print(s2['c']) ->  2.3
```

部分课件改写自《Python金融数据分析与挖掘实战》- 黄桓秋

3. Pandas-介绍

Pandas 核心数据结构1 – 序列 (Series)

- Numpy
- Pandas
- Matplotlib




```
#通过字典创建序列
mydict={'red':2000,'bule':1000,'yellow':500}    #定义字典
ss=pd.Series(mydict)                            #指定字典创建序列
```

```
import pandas as pd
s1=pd.Series([1,-2,2.3,'hq'])  #创建序列s1
va1=s1.values                  #获取序列s1中的值，赋给变量va1
in1=s1.index                   #获取序列s1中的索引，赋给变量in1
print(va1)
print(in1)
```

执行结果如下：

```
[1 -2 2.3 'hq']
RangeIndex(start=0, stop=4, step=1)
```

部分课件改写自《Python金融数据分析与挖掘实战》- 黄桓秋

3. Pandas-介绍

- Numpy
- Pandas
- Matplotlib

Pandas 核心数据结构1 – 序列 (Series)

1. unique() <= 唯一性

```
import pandas as pd
s5=[1,2,2,3,'hq','hq','he']      #定义列表s5
s5=pd.Series(s5)                 #将列表s5转换为序列
s51=s5.unique()                  #调用unique()方法去重
print(s51)
```

执行结果如下：

[1 2 3 'hq' 'he']

2. isin() <= 存在性

如果存在则返回True，否则为False。

如判断元素0和'he'这两个元素是否存在于s5序列中。

示例代码如下：

```
s52=s5.isin([0,'he'])
print(s52)
```

```
0    False
1    False
2    False
3    False
4    False
5    False
6     True
dtype: bool
```

3. value_counts() <= 出现次数

```
s53=s5.value_counts()
```

其中索引（index）为原序列元素的值，其值部分则为出现的次数。本函数在实际应用中，有时也起到与unique相同的效果，即去掉序列数据中的重复值，保障了数据的唯一性，而且还获得了重复的次数，在金融数据处理中用得非常广泛。

s53 - Series

Index	0
hq	2
2	2
he	1
3	1
1	1

部分课件改写自《Python金融数据分析与挖掘实战》- 黄桓秋

Pandas 核心数据结构1 – 序列 (Series)

4.空值处理方法

① isnull(): 判断序列中是否有空值 (nan值); 空值 ->True，非空值 ->False.
② notnull(): 判断序列中的非空值; 非空值->True，空值(nan值)->False.
③ dropna(): 清洗序列中的空值(nan值). 可以配合使用空值处理函数，实现对空值的清洗.

```
import pandas as pd
import numpy as np

ss1=pd.Series([10,'hq',60,np.nan,20])   #定义序列ss1，其中np.nan为空值（nan值）
tt1=ss1[~ss1.isnull()]                  #~为取反，采用逻辑数组进行索引获取数据

tt2=ss1[ss1.notnull()]
tt3=ss1.dropna()
```

tt2和tt3的结果与tt1一样。

ss1 - Series

Index	0
0	10
1	hq
2	60
3	nan
4	20

tt - Series

Index	0
0	10
1	hq
2	60
4	20

3. Pandas-介绍

- Numpy
- Pandas
- Matplotlib

Pandas 核心数据结构1 – 序列 (Series)

批量获取序列中的值：

```
import numpy as np

s1=pd.Series([1,-2,2.3,'hq'])
s2=pd.Series([1,-2,2.3,'hq'],index=['a','b','c','d'])
s4=pd.Series(np.array([1,2,4,7.1]))

s22=s2[['a','d']]      #取索引号为字符a,b的元素
s11=s1[0:2]            #索引为连续的数组
s12=s1[[0,2,3]]        #索引为不连续的数组
s41=s4[s4>2]           #索引为逻辑数组

print(s22)
print('-'*20)
print(s11)
print('-'*20)
print(s12)
print('-'*20)
print(s41)
```

执行结果如下：

```
a     1
d    hq
dtype: object
--------------------
0     1
1    -2
dtype: object
--------------------
0     1
2   2.3
3    hq
dtype: object
--------------------
2   4.0
3   7.1
dtype: float64
```

3. Pandas-介绍

Pandas 核心数据结构1 – 序列 (Series)

- Numpy
- Pandas
- Matplotlib

序列中的求和、平均值、最大值、最小值、方差、标准差等：

```
import pandas as pd

s=pd.Series([1,2,4,5,6,7,8,9,10])
su=s.sum()
sm=s.mean()
ss=s.std()
smx=s.max()
smi=s.min()
```

Name	Type	Size	Value
s	Series	(9,)	Series object of pandas.core.serie
sm	float	1	5.777777777777778
smi	int64	1	1
smx	int64	1	10
ss	float	1	3.0731814857642954
su	int	1	52

3. Pandas-介绍

Pandas 核心数据结构2 – 数据框 (DataFrame)

Numpy

Pandas

Matplotlib

- Pandas中的数据框，由多个序列按照相同的index组织在一起，形成一个二维表.
- 数据框的每一列为序列.
- 数据框的属性包括index、列名和值.
- 许多外部数据文件读取到Python中, 采用数据框的来存取 (e.g. excel, csv, txt文件).
- 数据框提供了极为丰富的方法用于处理数据，是完成数据处理及分析的重要方法之一.

3. Pandas-介绍

Numpy

Pandas

Matplotlib

Pandas 核心数据结构2 – 数据框 (DataFrame)

基于字典，利用Pandas库中的DataFrame函数，可以创建数据框。

- 字典的键(key)为列名
- 字典的值(value)为列值
- 索引(index)为默认值　(即从0开始从小到大排列)

```
import pandas as pd
import numpy as np

data={'a':[2,2,np.nan,5,6],
        'b':['kl','kl','kl',np.nan,'kl'],
        'c':[4,6,5,np.nan,6],
        'd':[7,9,np.nan,9,8]}

df = pd.DataFrame(data)
```

df - DataFrame

Index	a	b	c	d
0	2	kl	4	7
1	2	kl	6	9
2	nan	kl	5	nan
3	5	nan	nan	9
4	6	kl	6	8

数据框对象具有三个属性:

1. 列名 df.columns
2. 索引 df.index
3. 值　df.values

部分课件改写自《Python金融数据分析与挖掘实战》- 黄桓秋

3. Pandas-介绍

- Numpy
- Pandas
- Matplotlib

Pandas 核心数据结构2 – 数据框 (DataFrame)

1. dropna() <= 去掉数据集中的空值（nan值）

```
df1=df.dropna()
```

df - DataFrame

Index	a	b	c	d
0	2	kl	4	7
1	2	kl	6	9
2	nan	kl	5	nan
3	5	nan	nan	9
4	6	kl	6	8

df1 - DataFrame

Index	a	b	c	d
0	2	kl	4	7
1	2	kl	6	9
4	6	kl	6	8

2. fillna() <=对数据框中的空值进行填充

```
df2=df.fillna(0)          #所有空值元素填充0
df3=df.fillna('Kl')       #所有空值元素填充Kl
df4=df.fillna({'a':0,'b':'kl','c':0,'d':0})
df5=df.fillna({'a':0,'b':'kl'  })    #部分列填充
```

df2 - DataFrame

Index	a	b	c	d
0	2	kl	4	7
1	2	kl	6	9
2	0	kl	5	0
3	5	0	0	9
4	6	kl	6	8

df3 - DataFrame

Index	a	b	c	d
0	2	kl	4	7
1	2	kl	6	9
2	Kl	kl	5	Kl
3	5	Kl	Kl	9
4	6	kl	6	8

df4 - DataFrame

Index	a	b	c	d
0	2	kl	4	7
1	2	kl	6	9
2	0	kl	5	0
3	5	kl	0	9
4	6	kl	6	8

df5 - DataFrame

Index	a	b	c	d
0	2	kl	4	7
1	2	kl	6	9
2	0	kl	5	nan
3	5	kl	nan	9
4	6	kl	6	8

部分课件改写自《Python金融数据分析与挖掘实战》- 黄恒秋

3. Pandas-介绍

- Numpy
- Pandas
- Matplotlib

Pandas 核心数据结构2 – 数据框 (DataFrame)

3. sort_values() <= 指定列按值进行排序

```
import pandas as pd
data={'a':[5,3,4,1,6],'b':['d','c','a','e','q'],'c':[4,6,5,5,6]}
Df=pd.DataFrame(data)
Df1=Df.sort_values('a',ascending=False)
#默认按升序，这里设置为降序
```

Df - DataFrame

Index	a	b	c
0	5	d	4
1	3	c	6
2	4	a	5
3	1	e	5
4	6	q	6

Df1 - DataFrame

Index	a	b	c
4	6	q	6
0	5	d	4
2	4	a	5
1	3	c	6
3	1	e	5

4. sort_index() <=对索引进行排序

```
Df2=Df1.sort_index(ascending=False)
#默认按升序，这里设置为降序
```

Df1 - DataFrame

Index	a	b	c
4	6	q	6
0	5	d	4
2	4	a	5
1	3	c	6
3	1	e	5

Df2 - DataFrame

Index	a	b	c
4	6	q	6
3	1	e	5
2	4	a	5
1	3	c	6
0	5	d	4

部分课件改写自《Python金融数据分析与挖掘实战》- 黄恒秋

3. Pandas-介绍

- Numpy
- Pandas
- Matplotlib

Pandas 核心数据结构2 – 数据框 (DataFrame)

5. head() <=取前N行

H4=Df2.head(4)　#取前4行

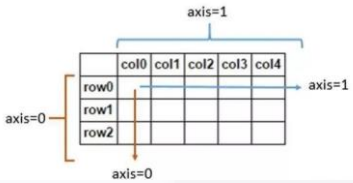
H4 - DataFrame

Index	a	b	c
4	6	q	6
3	1	e	5
2	4	a	5
1	3	c	6

6. drop() <=删掉数据集中的指定列或行

H41=H4.drop('b', axis=1)

列：axis=1　行：axis=0



H41 - DataFrame

Index	a	c
4	6	6
3	1	5
2	4	5
1	3	6

7. join() <=对两个数据框之间的水平连接

Df3=pd.DataFrame({'d':[1,2,3,4,5]})

Df4=Df.join(Df3)

Df - DataFrame

Index	a	b	c
0	5	d	4
1	3	c	6
2	4	a	5
3	1	e	5
4	6	q	6

Df4 - DataFrame

Index	a	b	c	d
0	5	d	4	1
1	3	c	6	2
2	4	a	5	3
3	1	e	5	4
4	6	q	6	5

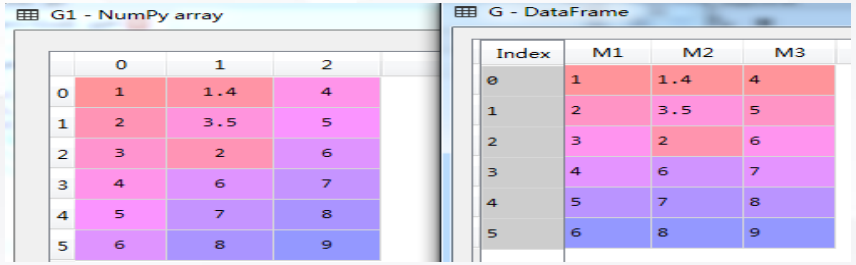
部分课件改写自《Python金融数据分析与挖掘实战》- 黄恒秋

Pandas 核心数据结构2 – 数据框 (DataFrame)

8. as_matrix() <= 数据框转换为Numpy数组

```
import pandas as pd
list1=['a','b','c','d','e','f']
list2=[1,2,3,4,5,6]
list3=[1.4,3.5,2,6,7,8]
list4=[4,5,6,7,8,9]
list5=['t',5,6,7,'k',9.6]

G={'M1':list2,'M2':list3,'M3':list4}
#定义字典G，值为纯数值数据
G=pd.DataFrame(G)
G1=G.as_matrix()
```



	0	1	2
0	1	1.4	4
1	2	3.5	5
2	3	2	6
3	4	6	7
4	5	7	8
5	6	8	9

Index	M1	M2	M3
0	1	1.4	4
1	2	3.5	5
2	3	2	6
3	4	6	7
4	5	7	8
5	6	8	9

```
D={'M1':list1,'M2':list2,'M3':list3,'M4':list4,'M5':list5}
    #定义字典D，值为字符、数值混合数据

D=pd.DataFrame(D)

D1=D.as_matrix()
```

```
D= [['a' 1 1.4 4 't']
    ['b' 2 3.5 5 5]
    ['c' 3 2.0 6 6]
    ['d' 4 6.0 7 7]
    ['e' 5 7.0 8 'k']
    ['f' 6 8.0 9 9.6]]
```

Pandas 核心数据结构2 – 数据框 (DataFrame)

9. to_excel() <= 将数据框导出为Excel文件

D.to_excel('D.xlsx')

	M1	M2	M3	M4	M5
0	a	1	1.4	4	t
1	b	2	3.5	5	5
2	c	3	2	6	6
3	d	4	6	7	7
4	e	5	7	8	k
5	f	6	8	9	9.6

10. to_csv()

```
D={'M1':list1,'M2':list2,'M3':list3,'M4':list4,'M5':list5}
    #定义字典D，值为字符、数值混合数据

D=pd.DataFrame(D)

D1=D.as_matrix()
```

```
D= [['a' 1 1.4 4 't']
    ['b' 2 3.5 5 5]
    ['c' 3 2.0 6 6]
    ['d' 4 6.0 7 7]
    ['e' 5 7.0 8 'k']
    ['f' 6 8.0 9 9.6]]
```

3. Pandas-介绍

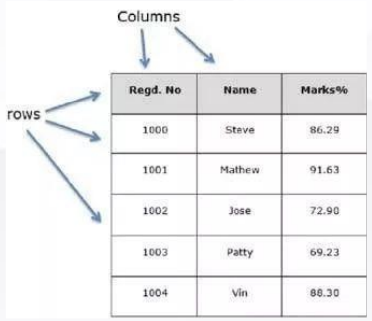
Pandas 核心数据结构2 – 数据框 (DataFrame)

11. DF.iloc[①,②]

①为对DF的行(row)下标控制，②为对DF的列(columns)下标控制

```
# iloc for positional indexing
c3=df2.iloc[1:3,2]
c4=df2.iloc[1:3,0:2]
c5=df2.iloc[1:3,:]
c6=df2.iloc[[0,2,3],[1,2]]

TF=[True,False,False,True,True]
c7=df2.iloc[TF,[1]]
```



df2 - DataFrame

Index	a	b	c	d
0	2	k1	4	7
1	2	k1	6	9
2	0	k1	5	0
3	5	0	0	9
4	6	k1	6	8

c3 - Series

Index	c
1	6
2	5

c4 - DataFrame

Index	a	b
1	2	k1
2	0	k1

c6 - DataFrame

Index	b	c
0	k1	4
2	k1	5
3	0	0

c7 - DataFrame

Index	b
0	k1
3	0
4	k1

Pandas 核心数据结构2 – 数据框 (DataFrame)

12. DF.loc[①,②]

①为对DF的条件，②为对DF的列下标控制

```
# loc for label based indexing
c8=df2.loc[df2['b'] == 'kl',:];
c9=df2.loc[df2['b'] == 'kl',:].head(3);
c10=df2.loc[df2['b'] == 'kl',['a','c']].head(3);
c11=df2.loc[df2['b'] == 'kl',['a','c']];
```

df2 - DataFrame

Index	a	b	c	d
0	2	kl	4	7
1	2	kl	6	9
2	0	kl	5	0
3	5	0	0	9
4	6	kl	6	8

c10 - DataFrame

Index	a	c
0	2	4
1	2	6
2	0	5

c9 - DataFrame

Index	a	b	c	d
0	2	kl	4	7
1	2	kl	6	9
2	0	kl	5	0

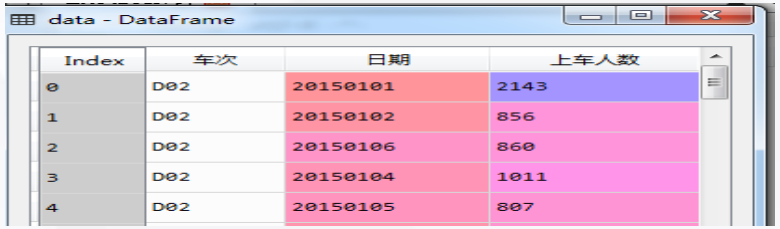
3. Pandas-介绍

Pandas 核心数据结构2 – 数据框 (DataFrame)

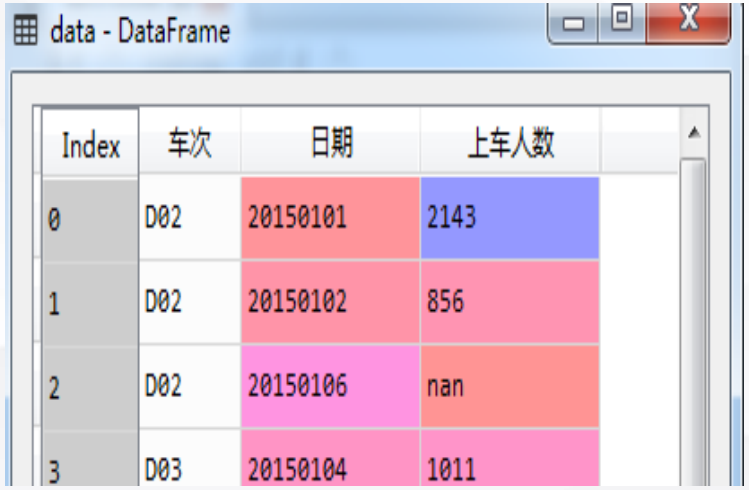
- Numpy
- Pandas
- Matplotlib

13. pd.read_excel()

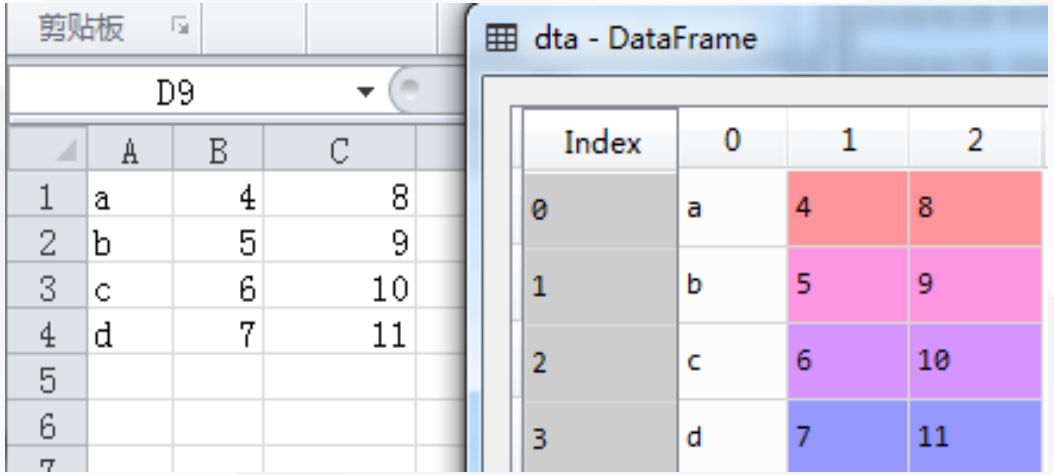
path='一、车次上车人数统计表.xlsx';
data=pd.read_excel(path)



#读取Sheet2里面的数据
data=pd.read_excel(path,'Sheet2')



dta=pd.read_excel('dta.xlsx',header=None) #无表头



3. Pandas-介绍

- Numpy
- Pandas
- Matplotlib

Pandas 核心数据结构2 – 数据框 (DataFrame)

13. pd.read_table() <= 读取TXT文本数据

```
import pandas as pd
dta1=pd.read_table('txt1.txt',header=None)
#分隔默认为Tab键，设置无表头
```

txt1 - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

8.35	23.53	7.51	8.62	17.42	10.00	1.04	11.21
9.25	23.75	6.61	9.19	17.77	10.48	1.72	10.51
8.19	30.50	4.72	9.78	16.28	7.60	2.52	10.32
7.73	29.20	5.42	9.43	19.29	8.49	2.52	10.00
9.42	27.93	8.20	8.14	16.17	9.42	1.55	9.76
9.16	27.98	9.01	9.32	15.99	9.10	1.82	11.35
10.06	28.64	10.52	10.05	16.18	8.39	1.96	10.81
9.09	28.12	7.40	9.62	17.26	11.12	2.49	12.65

dta1 - DataFrame

Index	0	1	2	3	4	5	6	7
0	8.35	23.53	7.51	8.62	17.42	10	1.04	11.21
1	9.25	23.75	6.61	9.19	17.77	10.48	1.72	10.51
2	8.19	30.5	4.72	9.78	16.28	7.6	2.52	10.32
3	7.73	29.2	5.42	9.43	19.29	8.49	2.52	10
4	9.42	27.93	8.2	8.14	16.17	9.42	1.55	9.76
5	9.16	27.98	9.01	9.32	15.99	9.1	1.82	11.35

```
dta2=pd.read_table('txt2.txt',sep='\s+')
#分隔为空格，带表头
```

txt2 - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
a b c d e
1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
```

dta2 - DataFrame

Index	a	b	c	d	e
0	1	2	3	4	5
1	6	7	8	9	10
2	11	12	13	14	15

部分课件改写自《Python金融数据分析与挖掘实战》- 黄恒秋

3. Pandas-介绍

- Numpy
- Pandas
- Matplotlib

Pandas 核心数据结构2 – 数据框 (DataFrame)

13. pd.read_table() <= 读取TXT文本数据

```
import pandas as pd
dta3=pd.read_table('txt3.txt',sep=',',header=None)
#分隔为逗号，设置无表头
```

Index	0	1	2	3	4
0	2	3	4	5	6
1	7	8	9	10	11
2	12	13	14	15	16

部分课件改写自《Python金融数据分析与挖掘实战》- 黄恒秋

Pandas 滚动计算 - rolling

- Numpy
- Pandas
- Matplotlib

13. pd.rolling_mean(P,N) <= 滚动求平均值

P为待求的数据列，N为滚动计算的长度。

P可以是Numpy或DataFrame，但是不能是list或者tuple

```
import pandas as pd
import numpy as np

L=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15]  #列表
T=(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15)  #元组

A=np.array(L)                    #将列表L转换为数组，赋给变量A
S=pd.Series(L)                   #将列表L转换为序列，赋给变量S
#avg_L=pd.rolling_mean(L,10)     #报错
#avg_T=pd.rolling_mean(T,10)     #报错
avg_S=pd.rolling_mean(S,10)
avg_A=pd.rolling_mean(A,10)
```

avg_A - NumPy array

	0
0	nan
1	nan
2	nan
3	nan
4	nan
5	nan
6	nan
7	nan
8	nan
9	5.5
10	6.5
11	7.5
12	8.5
13	9.5
14	10.5

avg_S - Series

Index	0
0	nan
1	nan
2	nan
3	nan
4	nan
5	nan
6	nan
7	nan
8	nan
9	5.5
10	6.5
11	7.5
12	8.5
13	9.5
14	10.5

Pandas 滚动计算 - rolling

- Numpy
- Pandas
- Matplotlib

13. pd.rolling_sum(P,N) <= 滚动求和

pd.rolling_min(P,N) <= 滚动求最小值

pd.rolling_max(P,N) <= 滚动求最大值

```
sum_S=pd.rolling_sum(S,10)
sum_A=pd.rolling_sum(A,10)

Min_S=pd.rolling_min(S,10)
Min_A=pd.rolling_min(A,10)

Max_S=pd.rolling_max(S,10)
Max_A=pd.rolling_max(A,10)
```

P为待求的数据列，N为滚动计算的长度。

P可以是Numpy或DataFrame，但是不能是list或者tuple

```
import pandas as pd
import numpy as np

L=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15]  #列表
T=(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15)  #元组

A=np.array(L)          #将列表L转换为数组，赋给变量A
S=pd.Series(L)         #将列表L转换为序列，赋给变量S
```

3. Matplotlib-介绍

Matplotlib 简介

- Matplotlib是Python中一个二维绘图包，能够非常简单的实现数据可视化。Matplotlib最早由John Hunter于2002年启动开发，其目的是为了构建一个Matlab式的绘图函数接口。
- Matplotlib图像大致可以分为如下4个层次结构：

Numpy

Pandas

Matplotlib

1. canvas（画板）

位于最底层，导入matplotlib库时自动存在

2. figure（画布）

建立在canvas之上，
从这一层就能开始设置其参数。

3.axes（子图）

将figure分成不同块，
实现分面绘图。

4.图表信息（构图元素）

添加或修改axes上的图形信息，
优化图表的显示效果。

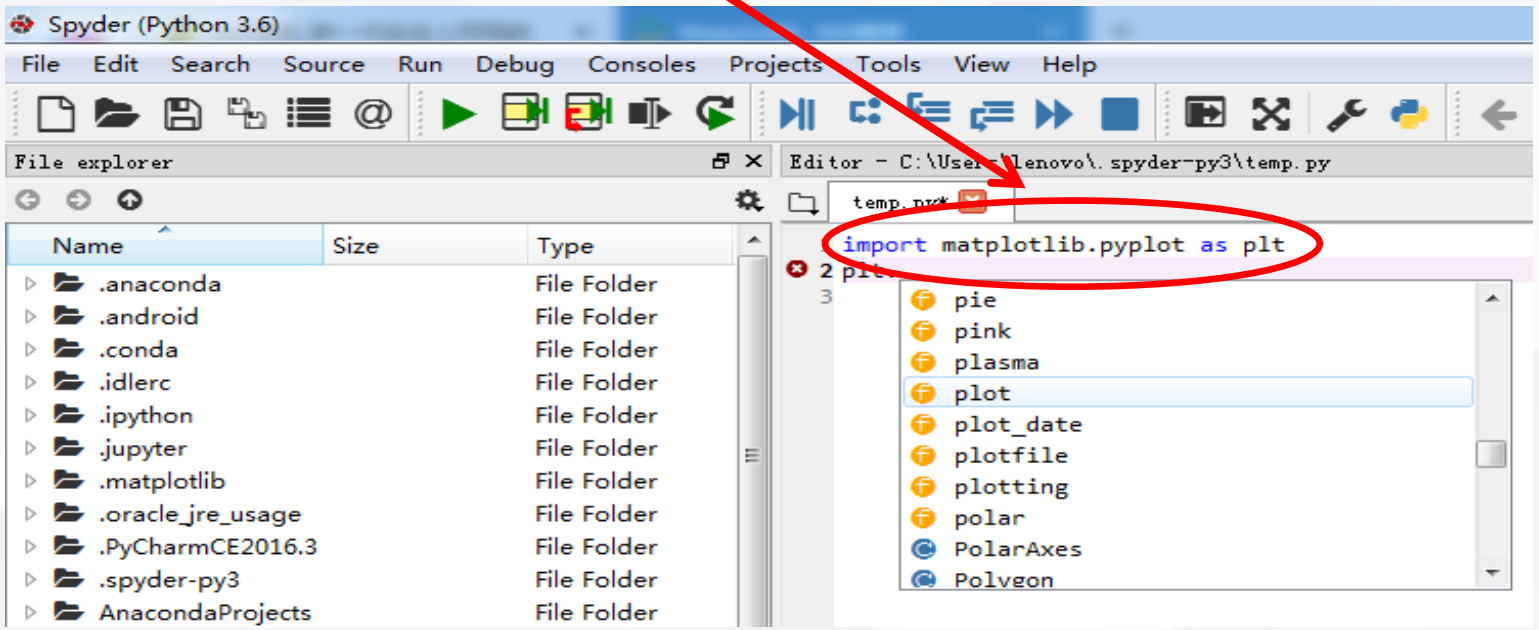
部分课件改写自《Python金融数据分析与挖掘实战》- 黄桓秋

3. Matplotlib-介绍

Matplotlib 简介

- Numpy
- Pandas
- Matplotlib

在Anaconda发行版中已经集成了Matplotlib库，直接导入pyplot模块就可以使用了。导入方法为：import matplotlib.pyplot as plt



部分课件改写自《Python金融数据分析与挖掘实战》- 黄桓秋

3. Matplotlib-介绍

Matplotlib – 画布、绘图、标签

```
plt.figure(1)                # 创建第一个画布
plt.subplot(2, 1, 1)         # 画布划分为2×1图形阵，选择第1张图片
```

添加画布内容。第二部分是绘图的主体部分。添加标题、坐标轴名称等步骤与绘制图形是并列的，没有先后顺序，可以先绘制图形，也可以先添加各类标签。

函数名称	函数作用
title	在当前图形中添加标题，可以指定标题的名称、位置、颜色、字体大小等参数
xlabel	在当前图形中添加x轴名称，可以指定位置、颜色、字体大小等参数
ylabel	在当前图形中添加y轴名称，可以指定位置、颜色、字体大小等参数
xlim	指定当前图形x轴的范围，只能确定一个数值区间，而无法使用字符串标识
ylim	指定当前图形y轴的范围，只能确定一个数值区间，而无法使用字符串标识
xticks	指定x轴刻度的数目与取值
yticks	指定y轴刻度的数目与取值
legend	指定当前图形的图例，可以指定图例的大小、位置、标签

3. Matplotlib-介绍

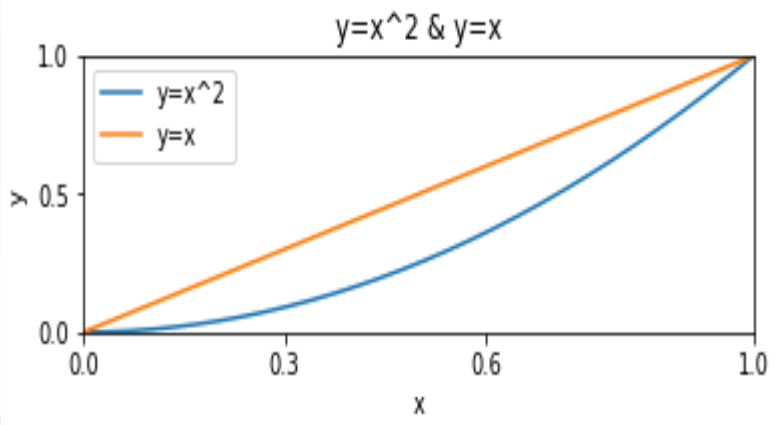
Numpy

Pandas

Matplotlib

Matplotlib – 第一幅图

- 最后是图形保存与展示。绘制图形之后，可使用matplotlib.pyplot.savefig()函数保存图片到指定路径
- 使用matplotlib.pyplot.show()函数展示图形。综合整体流程绘制函数“y=x^2”与“y=x”



图像示例代码如下：

```
import matplotlib.pyplot as plt
import numpy as np
plt.figure(1)          # 创建画布
x = np.linspace(0, 1, 1000)
plt.subplot(2, 1, 1) # 分为2×1图形阵，选择第1张图片绘图
plt.title('y=x^2 & y=x')  # 添加标题
plt.xlabel('x')        # 添加x轴名称 ‘x’
plt.ylabel('y')        # 添加y轴名称 ‘y’
plt.xlim((0, 1))       # 指定x轴范围（0,1）
plt.ylim((0, 1))       # 指定y轴范围（0,1）
plt.xticks([0, 0.3, 0.6, 1])  # 设置x轴刻度
plt.yticks([0, 0.5, 1]) # 设置y轴刻度
plt.plot(x, x ** 2)
plt.plot(x, x)
plt.legend(['y=x^2', 'y=x'])  # 添加图例
plt.savefig('1.png')  # 保存图片
plt.show()
```

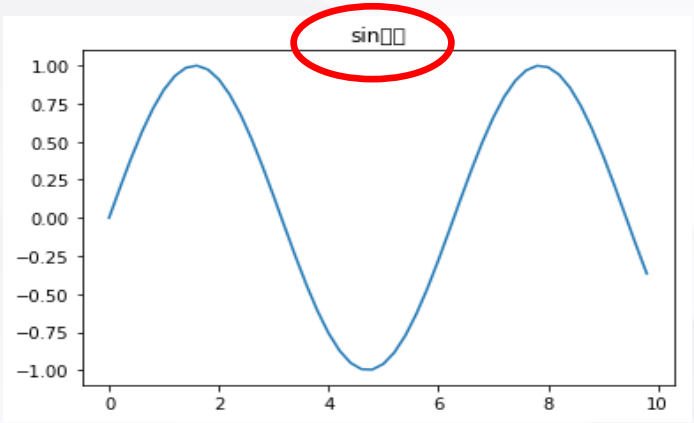
Matplotlib – 解决中文显示问题

- Numpy
- Pandas
- Matplotlib

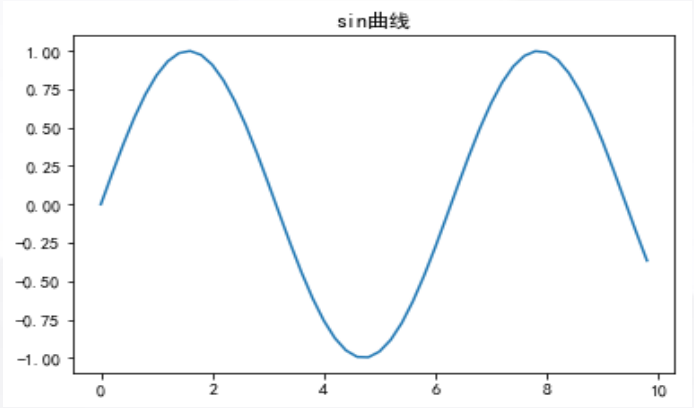
示例代码如下：

```
import numpy as np
import matplotlib.pyplot as plt

x = np.arange(0, 10, 0.2)
y = np.sin(x)
plt.title('sin曲线')
plt.plot(x, y)
plt.savefig('2.png')
plt.show()
```




```
x = np.arange(0, 10, 0.2)
y = np.sin(x)
plt.rcParams['font.sans-serif'] = 'SimHei'  # 设置字体为SimHei
plt.rcParams['axes.unicode_minus'] = False  # 解决负号“-”显示异常
plt.title('sin曲线')
plt.plot(x, y)
plt.savefig('2.png')
plt.show()
```



部分课件改写自《Python金融数据分析与挖掘实战》- 黄桓秋

3. Matplotlib-介绍

Numpy

Pandas

Matplotlib

Matplotlib – 解决坐标轴标注问题

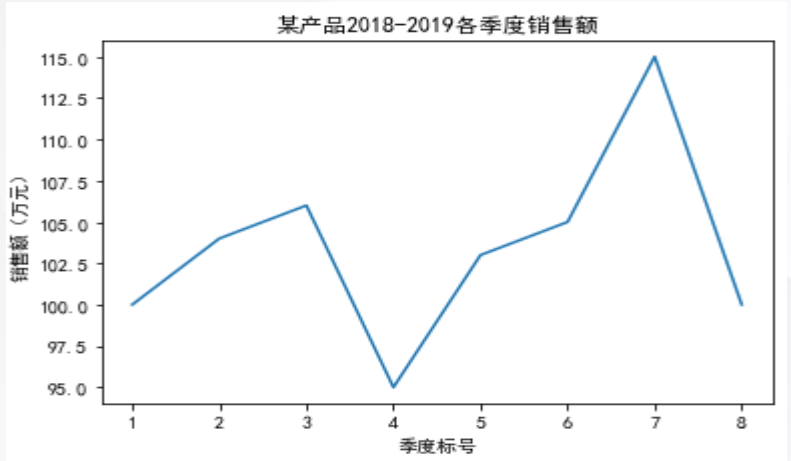
示例代码如下：

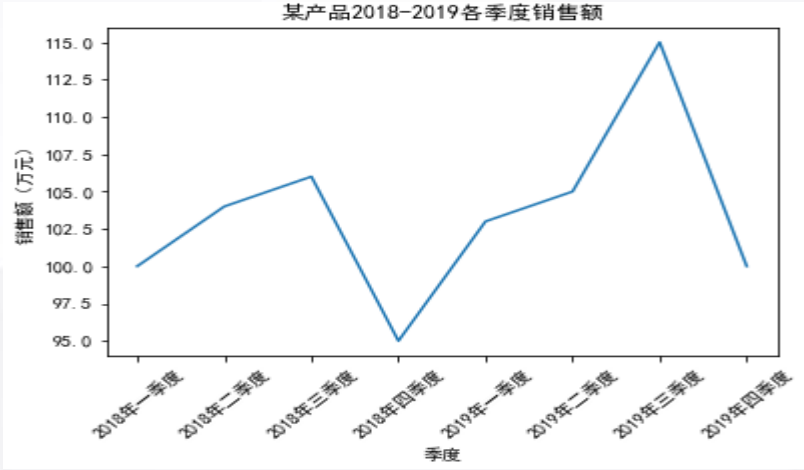
```
import numpy as np
import matplotlib.pyplot as plt

x = np.array([1,2,3,4,5,6,7,8])                    #季度标号
y = np.array([100,104,106,95,103,105,115,100])   #销售额
```

```
plt.rcParams['font.sans-serif'] = 'SimHei'          # 设置字体为SimHei
plt.title('某产品2018-2019各季度销售额')
plt.plot(x, y)
plt.xlabel('季度标号')
plt.ylabel('销售额（万元）')
plt.show()
```

```
v=['2018年一季度','2018年二季度','2018年三季度','2018年四季度',
   '2019年一季度','2019年二季度','2019年三季度','2019年四季度']
plt.rcParams['font.sans-serif'] = 'SimHei'     # 设置字体为SimHei
plt.title('某产品2018-2019各季度销售额')
plt.plot(x, y)
plt.xlabel('季度')
plt.xticks(x, v, rotation = 45) #v为与x对应的字符刻度，rotation为旋转角度
plt.ylabel('销售额（万元）')
plt.show()
```





部分课件改写自《Python金融数据分析与挖掘实战》- 黄恒秋

感谢倾听