中信建投证券
CHINA SECURITIES

CHINA SECURITIES RESEARCH

证券研究报告·金融工程深度报告

零基础 python 代码策略模型实战：

——大数据人工智能研究之七

金融工程研究

重要观点

本文概述

本文主要介绍了 python 基础、爬虫、与数据库交互、调用机器学习、深度学习、NLP 等。分别介绍了各个模块的安装，环境的搭建等。并且以机器学习选股为例，把各个模块连贯起来，核心代码基本都有详尽的解释。

大数据 AI 时代，python 无往不胜

Python 的包装能力、可组合性、可嵌入性都很好，可以把各种复杂性包装在 Python 模块里,非常友好的供调用。Python 资源丰富，深度学习如 keras，机器学习如 sk-learn，科学计算如 numpy、pandas，自然语言处理如 jieba 等。

Python 将极大提高工作效率

无论是科学计算，还是图形界面显示；无论是机器学习还是深度学习；无论是操作 excel,txt 等还是连接数据库；无论是搭建网站还是爬虫；无论是自然语言处理还是打包成 exe 执行文件，python 都能快速完成。以最少的代码，最高效的完成。

人人可编写人工智能模型

人工智能给人感觉难于入手，重要原因是机器学习、深度学习、自然语言处理等门槛太高；python 则以最简洁的方式，让你快速使用人工智能相关算法。本文以实战为目的，对模块的安装，搭建环境，核心代码等进行了详细的介绍。

人工智能选股模型策略

以传统因子滚动 12 个月值为特征值，个股下一期按收益大小排序，排名前 30%作为强势股，排名靠后 30%作为弱势股。用机器学习算法进行训练预测。用当期因子作为输入，预测未来一个月个股相对走势的强弱。根据个股的相对强势，我们把排名靠前 20%的作为多头，排名后 20%的作为空头进行了研究，样本外 20090105 到 20171130 期间，行业中性等权年化多空收益差为 16.45%，年化波动率为 7.34%，最大回撤为 10.84%。

丁鲁明

dingluming@csc.com.cn

021-68821623

执业证书编号：S1440515020001

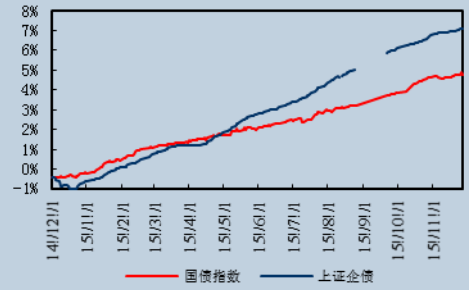
研究助理：喻银尤

yuyinyou@csc.com.cn

021-68821600-808

发布日期： 2018 年 03 月 08 日

市场表现



相关研究报告

18.02.02	大数据人工智能研究之六：机器学习因子有效性分析
17.10.18	大数据研究之五：大数据、机器学习、深度学习在投资领域应用的方法论概述
17.08.16	大数据研究之四：基于新闻热度的周期、成长、消费风格轮动配置
17.03.08	大数据研究之三：新闻情绪选股的多空差策略
17.03.02	大数据研究之指标构建：机器学习之贝叶斯文本分类算法的实现
16.10.12	大数据研究体系之择时篇：基于新闻热度的多空策略

目录

图目录

表目录

一、Python 介绍

Python 是一种面向对象的解释型计算机程序设计语言，由荷兰人 Guido van Rossum 于 1989 年发明。有以下优点简单、易学、免费开源、可移植性、解释性强；缺点为单行语句输出、同 C++和 Matlab 比运行速度较慢。Python 有较为强大的标准库和模块，方便用户进行调用：如科学计算的 Numpy、Pandas、Scipy 库；如机器学习和深度学习的 Scikit-learn、Keras 库；如爬虫的 Pyquery、BeautifulSoup、Scrapy 库。Python 的应用领域较为广泛包括 Web 开发、人工智能、云计算、网络爬虫，游戏开发等。

1.1 Anaconda (Python IDE) 的安装

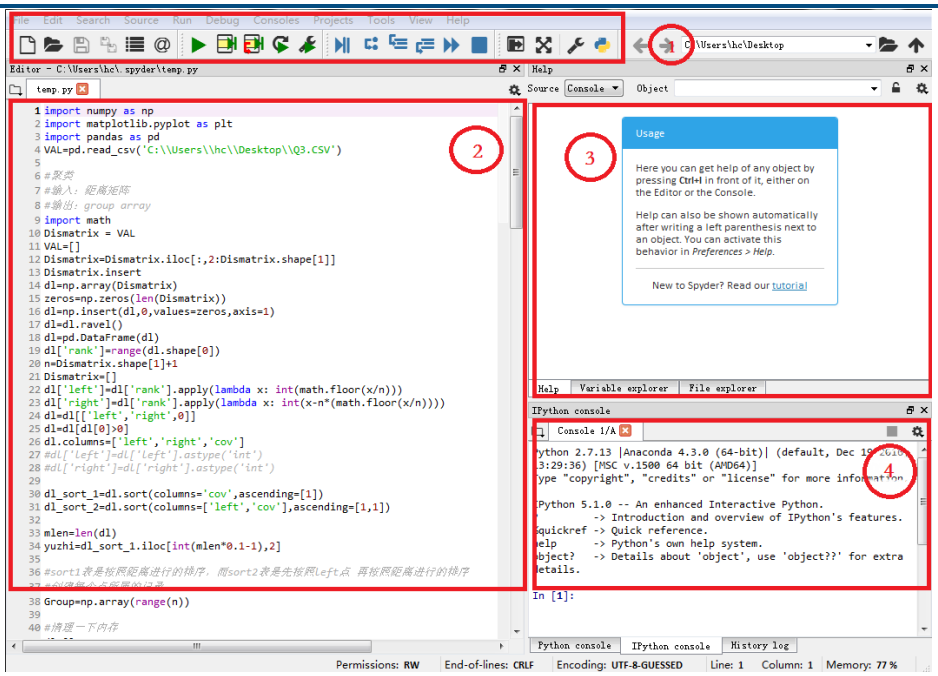
Python 的集成开发环境(IDE)有很多，如 Pycharm、Anaconda、eclipse。其中 Anaconda 是用于科学计算较好的 IDE，支持多种操作系统如 windows、Mac、Linux 等。Anaconda 有如下优势：使用方便，可以自由转化成 Matlab、R 等相关类似界面，非常友好；自带安装了一些计算标准库和模块，可以直接调用而且具有强大的库管理器。

下载以及安装 anaconda 可以从以下资源中获取：

•anaconda 官方网站：https://www.anaconda.com/download/#macos。

下载安装完成后，打开 Spyder 图标，可以看到如下图的界面：其中①为编辑栏菜单，可以进行 run 等各项操作；②为脚本编辑框，即在里面编写代码；③为变量观察框，可以单独查看变量的情况；④为结果控制窗口，可以看到运行结果，也可以在其中执行语句。

图 1：Anaconda 界面介绍



数据来源：中信建投证券研究发展部

1.2 Python 基础知识简介

1）Python 内置数据类型

① Python 的内置数据类型有很多种，包括 Number（数字）、String（字符串）、List（列表）、Tuple（元组）、Dictionary（字典）、Bool（布尔）等。数字类型主要分为 int（整数型）、long（长整型）、float（浮点型）和 complex（复杂型）；列表是一种可修改的集合类型，其元素可以是数字、字符串等基本类型，也可以是列表、元组、字典等集合对象，甚至可以是自定义的类型；元组类型和列表一样，也是一种序列，与列表不同的是，元组是不可修改的；字典类型是一种键值对的集合。

表 1：Python 数据类型

数据类型	举例
Number（数字）	25
String（字符串）	'Hello World!'
List（列表）	['name', 2, [1, 2]]
Tuple（元组）	('age', 1, 2)
Dictionary（字典）	{'a': 1, 2: ['a_a']}
Bool（布尔）	True, False

数据来源：中信建投证券研究发展部

2）Python 函数

函数是组织好的，可重复使用的，用来实现单一或相关联功能的代码段。函数的作用是提高应用的模块性和代码的重复利用率。Python 提供了许多内建函数，比如 print()。但也可以自己创建函数，这被叫做用户自定义函数。定义函数的规则如下：

① 函数代码块以 def 关键词开头，后接函数标识符名称和圆括号()；

② 任何传入参数和自变量必须放在圆括号中间，圆括号中间可以用于定义参数；

③ 函数的第一行语句可以选择性地使用文档字符串—用于存放函数说明；

④ 函数内容以冒号起始，并且缩进；

⑤ return[表达式]结束函数，选择性地返回一个值给调用方，不带表达式的 return 相当于返回 None。

Python 函数的语法格式如下表 2，具体函数举例如表 3：

表 2：Python 函数语法

```
def   function_name(parameters):
    """函数_文档字符串"""
    function_suite
    return [expression]
```

数据来源：中信建投证券研究发展部

表 3：Python 函数举例

```
>>>import    math
>>>def    quadratic(a, b, c):
            """二次方程求根"""
            x1 = (-b + math.sqrt(b * b - 4 * a * c)) / (2 * a)
            x2 = (-b - math.sqrt(b * b - 4 * a * c)) / (2 * a)
            return x1, x2
>>>answer = quadratic(1, 2, 1)
>>>print(answer)
(-1.0, -1.0)
```

数据来源：中信建投证券研究发展部

Python 除了正常的函数调用之外，还使用 lambda 来创建匿名函数。其特点如下：1.lambda 只是一个表达式，函数体比 def 简单很多；2.lambda 的主体是一个表达式，而不是一个代码块；3.lambda 函数拥有自己的命名空间，且不能访问自有参数列表之外或全局命名空间里的参数；4.虽然 lambda 函数看起来只能写一行，却不等同于 C 或 C++的内联函数，后者的目的是调用小函数时不占用栈内存从而增加运行效率。lambda 表达式举例如表 4:

表 4：Python lambda 表达式举例

```
>>>y = lambda x: x - 1
>>>print(y(2)):
1
```

数据来源：中信建投证券研究发展部

3）Python 模块

Python 中有一个概念叫做模块（module），这个和 C 语言中的头文件以及 Java 中的包很类似，比如在 Python 中要调用 sqrt 函数，必须导入 math 这个模块。在 Python 中用关键字 import 来导入某个模块，比如 import math 就导入了 math 模块。在调用 math 模块中的函数时，必须这样引用：模块名.函数名。在 Python 中，每个 Python 文件都可以作为一个模块，模块的名字就是文件的名字。

表 5：Python math 模块举例

math.exp(x)	返回 e 的 x 次方	>>> math.exp(2) 7.38905609893065
math.pow(x, y)	返回 x 的 y 次方	>>> math.pow(5,3) 125.0
math.floor(x)	返回不大于 x 的整数	>>> math.floor(5.8) 5.0
math.fabs(x)	返回 x 的绝对值	>>> math.fabs(-5) 5.0
math.fmod(x, y)	返回 x%y（取余）	>>> math.fmod(5,2) 1.0

数据来源：中信建投证券研究发展部

二、Python 的科学计算库

Python 的科学计算包有很多，常见的有 Numpy 库，Pandas 库，Scipy 库和 Matplotlib 库，其中 Numpy 库主要提供矩阵计算的功能；Pandas 库主要提供带标签列(columns)和索引(index)的数据结构之间的计算功能；Scipy 库可以当做是 Numpy 库的升级，是基于 Numpy 构建的一个集成了多种数学算法和方便的函数的 Python 模块；Matplotlib 库主要提供图形可视化的功能。

2.1 Numpy 库

Numpy 是 Python 的一个科学计算的库，提供了矩阵运算的功能，其一般与 Scipy、matplotlib 一起使用。其实，list 已经提供了类似于矩阵的表示形式，不过 Numpy 为我们提供了更多的函数。操作方法有以下几种：

1) 导入模块；
2) 以 list 或 tuple 变量为参数产生一维或者多维度的数组；
3) 通过 reshape 方法，创建一个只改变原数组尺寸的新数组，原数组的 shape 保持不变；
4) arange 函数通过指定开始值、终值（不包括）和步长来创建一维数组；
5) linspace 函数通过指定开始值、终值（包括）和元素个数来创建一维数组;
6) 合并数组可以分为 vstack（垂直方向）和 hstack（水平方向）操作;

表 6：Python numpy 库函数举例

```
>>>import numpy as np
>>>a = np.array([1, 2, 3, 4])
>>>print(a)
[1 2 3 4]
>>>b = a.reshape((2, 2))
>>>print(b)
[[1 2]
[3 4]]
>>>c = np.arange(0, 1, 0.2)
>>>print(c)
[0 0.2 0.4 0.6 0.8]
>>>d = np.linspace(0, 3, 4)
>>>print(d)
[0 1 2 3]
>>>e = np.hstack((b, b))
>>>print(e)
[[1 2 1 2]
 [3 4 3 4]]
```

数据来源：中信建投证券研究发展部

2.2 Pandas 库

pandas 核心为两大数据结构，即 Series 和 DataFrame。数据分析相关的所有事务都是围绕着这两种结构进行，pandas 库的 Series 对象是由一组数据（各种 Numpy 数据类型）以及一组与之相关的数据标签（即 Index）组成。DataFrame 是一个表格型的数据结构，它含有一组有序的列，每列可以是不同的值类型（数值、字符串、布尔值等），DataFrame 既有行索引（Index），也有列索引（Columns）。

具体的 Pandas 相关学习参考资料如下：

《利用 Python 进行数据分析》 Wes McKinney 著（唐学韬译）

Pandas 官方文档：http://pandas.pydata.org/pandas-docs/stable/

Pandas 的相关操作有以下几种：

1) 导入模块（与 Numpy 类似）；
2) 创建 Series，DataFrame；
3) 用 describe()函数对数据的快速汇总；
4) 运用 concat 合并 DataFrame；
5) 进行 groupby 分组；
6) 按值进行排序

表 7：Python pandas 库函数举例

```
>>>import pandas as pd
>>>sr = pd.Series([1, 2], index=('a', 'b'))
>>>print(sr)
a  1
b  2
dtype: int64
>>>df = pd.DataFrame([[1, 2], [3, 4]], index=('row1', 'row2'), columns=('col1', 'col2'))
>>>print(df)
      col1  col2
row1     1     2
row2     3     4
>>>sr.describe()
count   2.0
mean    1.5
std     0.7
min     1.0
25%     1.25
50%     1.50
75%     1.75
max     2.0
dtype: float64
>>>new_df = pd.concat([df, df])
>>>print(new_df)
      col1  col2
row1     1     2
row2     3     4
row1     1     2
row2     3     4
>>>new_df.groupby(by='col1').mean()
col1  col2
   1     2
   3     4
>>>new_df.sort_values(by='col1')
      col1  col2
row1     1     2
row1     1     2
row2     3     4
row2     3     4
```

数据来源：中信建投证券研究发展部

2.3 Scipy 库

SciPy 库是基于 Numpy 构建的一个集成了多种数学算法和方便的函数的 Python 模块。通过给用户提供一些高层的命令和类，SciPy 在 python 交互式会话中，大大增加了操作和可视化数据的能力。

Scipy 库可以提供如下的算法计算：

表 8：Python Scipy 库函数举例

cluster	聚类算法
interpolate	拟合和平滑曲线
linalg	线性代数
stats	统计上的函数和分布
optimize	最优路径选择

数据来源：中信建投证券研究发展部

2.4 Matplotlib 库

Matplotlib 库是一个用于创建出版质量图表的桌面绘图包（主要是 2D 方面），这是一个用 Python 构建的 MATLAB 式的绘图接口，所以库函数的参数以及调用方法大都与 MATLAB 一致。以下是 Matplotlib 使用的简单示例：

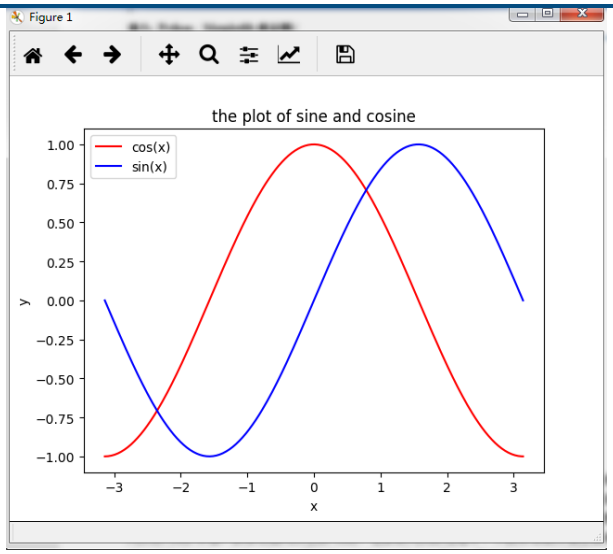
表 9：Python Matplotlib 库示例

```
>>>import numpy as np
>>>import matplotlib.pyplot as plt
>>>x = np.linspace(-np.pi, np.pi, 256)
>>>y = np.cos(x)
>>>z = np.sin(x)
>>>plt.figure()                                  #创建画板
>>>plt.plot(x, y, label="cos(x)", color="red")   #画函数图像
>>>plt.plot(x, z, label="sin(x)", color="blue")  #画函数图像
>>>plt.xlabel("x")                               #设置 x 轴标签
>>>plt.ylabel("y")                               #设置 y 轴标签
>>>plt.title("the plot of sine and cosine")      #设置图标题
>>>plt.legend()                                  #显示图例
>>>plt.show()                                    #显示图表
```

数据来源：中信建投证券研究发展部

上面代码输出结果如下：

图 2：Matplotlib 库示例图



数据来源：中信建投证券研究发展部

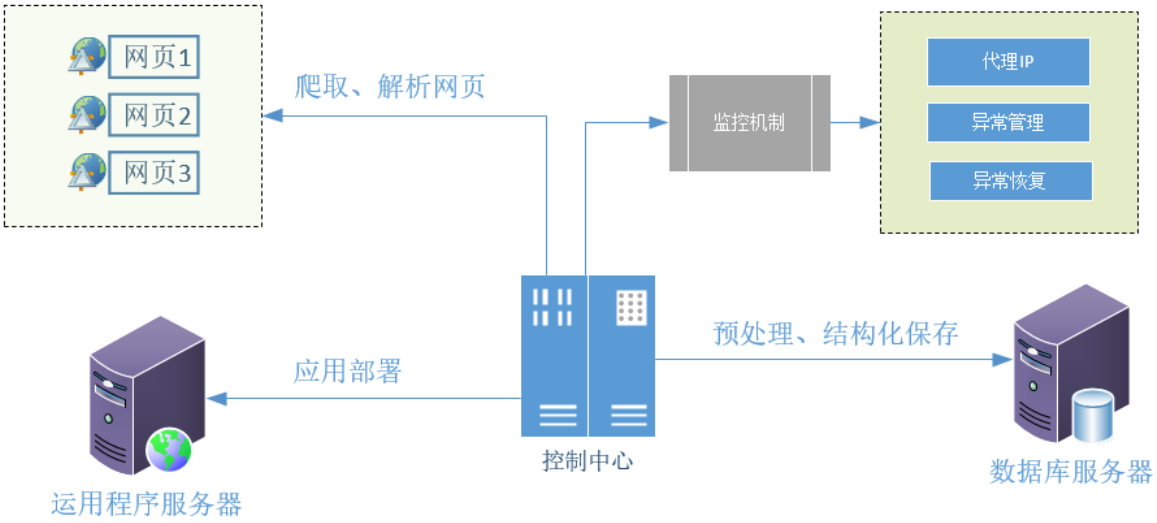
三、Python 的爬虫相关库

网络爬虫（Web crawler），是一种按照一定的规则，自动地抓取万维网信息的程序或者脚本，它们被广泛用于互联网搜索引擎或其他类似网站，可以自动采集所有其能够访问到的页面内容，以获取或更新这些网站的内容和检索方式。从功能上来讲，爬虫一般分为数据采集，处理，储存三个部分。传统爬虫从一个或若干初始网页的 URL 开始，获得初始网页上的 URL，在抓取网页的过程中，不断从当前页面上抽取新的 URL 放入队列，直到满足系统的一定停止条件。聚焦爬虫的工作流程较为复杂，需要根据一定的网页分析算法过滤与主题无关的链接，保留有用的链接并将其放入等待抓取的 URL 队列。然后，它将根据一定的搜索策略从队列中选择下一步要抓取的网页 URL，并重复上述过程，直到达到系统的某一条件时停止。

网络爬虫的基本工作流程如下：

1) 首先选取一部分精心挑选的种子 URL；
2) 将这些 URL 放入待抓取 URL 队列；
3) 从待抓取 URL 队列中取出 URL，解析 DNS，得到主机的 IP，并将 URL 对应的网页下载下来，存储进已下载网页库中。此外，将这些 URL 放进已抓取 URL 队列；
4) 分析已抓取 URL 队列中的 URL，分析其对应网页中的其他子 URL，并且将未抓取过的子 URL 放入待抓取 URL 队列，从而进入下一个循环。

图 3：简单爬虫流程图



数据来源：中信建投证券研究发展部

3.1 Scrapy 库

Scrapy 是 Python 开发的一个快速,高层次的屏幕抓取和 web 抓取框架，用于抓取 web 站点并从页面中提取结构化的数据。Scrapy 用途广泛，可以用于数据挖掘、监测和自动化测试、信息处理和历史档案等大量应用范围内抽取结构化数据的应用程序框架，广泛用于工业和大数据领域。Scrapy 的安装步骤如下:

1) 在 dos 窗口输入：pip install scrapy 回车；
2) 测试 scrapy 是否安装成功,在 dos 窗口输入 scrapy 回车；

Scrapy 主要包括了以下组件：

1) 引擎，用来处理整个系统的数据流处理，触发事务。
2) 调度器，用来接受引擎发过来的请求，压入队列中，并在引擎再次请求的时候返回。
3) 下载器，用于下载网页内容，并将网页内容返回给蜘蛛。
4) 蜘蛛，蜘蛛是主要干活的，用它来制订特定域名或网页的解析规则。
5) 项目管道，负责处理有蜘蛛从网页中抽取的项目，他的主要任务是清洗、验证和存储数据。当页面被蜘蛛解析后，将被发送到项目管道，并经过几个特定的次序处理数据。
6) 下载器中间件，位于 Scrapy 引擎和下载器之间的钩子框架，主要是处理 Scrapy 引擎与 下载器之间的请求及响应。
7) 蜘蛛中间件，介于 Scrapy 引擎和蜘蛛之间的钩子框架，主要工作是处理蜘蛛的响应输入和请求输出。
8) 调度中间件，介于 Scrapy 引擎和调度之间的中间件，从 Scrapy 引擎发送到调度的请求和响应。

表 10：Python Scrapy 库爬虫举例

```
import scrapy
class Sample(scrapy.Spider):
name = "china securities"

    def start_requests(self):
        url = "https://www.csc108.com/"
        yield scrapy.Request(url=url, callback=self.parse)

    def parse(self, response):
        title = response.xpath("//title/text()").extract_first()
        print(title)
```

在 dos 窗口输入：scrapy startproject tutorial

在 tutorial\spiders\路径下新建 py 文件保存上面代码

在 dos 窗口输入：scrapy crawl "china securities"，输出

中信建投证券

数据来源：中信建投证券研究发展部

3.2 BeautifulSoup 库

BeautifulSoup 是一个可以从 HTML 或 XML 文件中提取数据的 Python 库.它能够通过你喜欢的转换器实现惯用的文档导航,查找,修改文档的方式。BeautifulSoup 对象表示的是一个文档的全部内容。大部分时候,可以把它当作 Tag 对象,它支持遍历文档树和搜索文档树中描述的大部分的方法。BeautifulSoup 的安装方法如下：

① 在 dos 窗口输入 pip install beautifulsoup4

② 进入 python，输入 from bs4 import BeautifulSoup 运行，若不报错则安装成功。

表 11：BeautifulSoup 库操作步骤

1.创建对象；

2. 将复杂的 html 文档转换为树形结构（对象可分为 Tag，NavigableString 两种）；

3.搜索文档树，主要通过 find_all 查找；

4.CSS 选择器来筛选

数据来源：中信建投证券研究发展部

表 12：BeautifulSoup 爬虫案例

```
>>>from urllib.request import urlopen
>>>from bs4 import BeautifulSoup
>>>html = urlopen("http://quotes.toscrape.com/page/1/")
>>>b = BeautifulSoup(html.read(), "lxml")
>>>title = b.body.h1.text
>>>print(title)
```

Quotes to Scrape

数据来源：中信建投证券研究发展部

3.3 Pyquery 库

Pyquery 是一个类似 jquery 的库，通过使用 lxml 来处理 xml 和 html. 所以在使用 pyquery 时得先安装 lxml 库,在使用前需要先安装 lxml，下载地址如下：

① http://codespeak.net/lxml/lxml-2.2.8.tgz

② http://pypi.python.org/packages/source/p/pyquery/pyquery-0.6.1.tar.gz

快速简便的安装方法是在 dos 窗口输入：

① pip install libxml2-devel, pip install libxslt (Anaconda 一般默认已安装，直接第下一步便可)

② pip install pyquery

到此时安装已经完成。但以作者的经验，不知是 PyQuery 的 bug 还是什么原因，对于中文是乱码，解决方法如下：

把 pyquery/openers.py 的_requests 函数中的 if encoding: resp.encoding = encoding

换成 resp.encoding = encoding or None

表 13：pyquery 爬虫案例

```
>>>from pyquery import PyQuery as pq
>>>from urllib.request import urlopen
>>>d = pq(url="http://www.baidu.com", opener=lambda url: urlopen(url).read())
>>>print(d.text()[:9])
百度一下，你就知道
```

数据来源：中信建投证券研究发展部

以上是目前主流的一些爬虫框架，相对来说，Scrapy 功能最强大，BeautifulSoup 比较经典，Pyquery 则处理中文更加友好。当然，在爬虫中，会比样例复杂太多，爬虫者对 css,js 等前端技术有了解则更好，有些目标网页可能会有反爬虫机制，这是一个斗智斗勇的过程，你可能需要设置代理 IP,模拟浏览器等相关技术。

四、Python 的数据交互

Python 提供了多种数据的接口包括与 MySQL，SQL Server，Wind 等多类数据库，同时对于小批量的数据格式，例如 csv、excel 和 txt 等文件也可以完成数据的导入和导出。

4.1 Python 与数据库的交互

PyMySQL 是用于 Python 连接 MySQL 数据库的接口，在使用之前需要安装，安装的语句为：

pip install pymysql

其主要操作如下：

1) 查询记录；
2) 插入数据；
3) 更新数据；
4) 删除数据

表 14：Python 与 MySQL 的交互操作条目

```
import pymysql                                                    #导入交互模块
localhost = 'host'   #主机地址，比如本地为：127.0.0.1
user = 'user'     #数据库登录帐号
pwd = 'pwd'   #数据库登录密码
db = 'db'     #要操作的数据库
table = 'table'     #要操作的表名
#连接数据库
conn = pymysql.connect(host=localhost,user=user,passwd=pwd,db=db,charset='utf8')
cur =conn.cursor()
sql = "SELECT field FROM " + table       #查询语句
sql =    "INSERT INTO "+ table +    " VALUES (值 1，值 2,....) "       #插入语句
sql =    "UPDATE "+ table + "SET  列名称  =  新值  WHERE  列名称  =  某值"   #更新语句
sql =    "DELETE FROM" + table + "WHERE  列名称  =  值"     #删除语句
cur.execute(sql)   #执行语句
row = cur.fetchone()   #获得结果
#操作数据库完毕后，关闭
cursor.close()
conn.close()
```

数据来源：中信建投证券研究发展部

Pyodbc 是用于 Python 连接 SQL Sever 数据库的接口，在使用之前需要安装，安装的语句为：

```
pip install pyodbc
```

其主要操作如下：

1) 查询记录；
2) 插入数据；
3) 更新数据；
4) 删除数据

表 15：Python 与 SQL Server 的交互操作条目

```
import pyodbc                                                    #导入交互模块
port = 'port' #端口号：sqlserver 一般为 1433
conInfo = 'driver={SQL Server};server='+host+','+port+';database='+db+';uid='+user+';pwd='+pwd
conn = pyodbc.connect(conInfo)
cur = conn.cursor()
别的操作基本和 mysql 完全一致
```

数据来源：中信建投证券研究发展部

4.2 Python 与 csv、excel 和 txt 文件的交互

对于 csv、excel 和 txt 格式的文件，Pandas 和 Python 内置函数提供了导入和保存数据的方法，具体如下：

图 4：csv、excel 数据示例

	A	B	C
1	s_info_windcode	trade_dt	s_dq_close
2	000001.SH	20170915	3353.6192
3	000001.SH	20170918	3362.8587
4	000001.SH	20170919	3356.8446
5	000001.SH	20170920	3365.9959
6			
7			

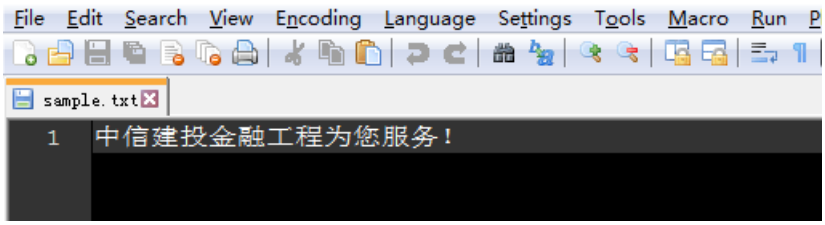
数据来源：中信建投证券研究发展部

表 16：Python 与 csv、excel 的交互

```
>>>import pandas as pd
>>>csv_df = pd.read_csv("C:/Users/Desktop/000001.SH.csv")
>>>print(csv_df)
   s_info_windcode    trade_dt    s_dq_close
0          000001.SH    20170915     3353.6192
1          000001.SH    20170918     3362.8587
2          000001.SH    20170919     3356.8446
3          000001.SH    20170920     3365.9959
>>>csv_df.to_csv("C:/Users/Desktop/index.csv")          #将 csv_df 以 csv 格式保存在指定目录下
>>>excel_df = pd.read_excel("C:/Users/Desktop/000001.SH.xlsx")
>>>print(excel_df)
   s_info_windcode    trade_dt    s_dq_close
0          000001.SH    20170915     3353.6192
1          000001.SH    20170918     3362.8587
2          000001.SH    20170919     3356.8446
3          000001.SH    20170920     3365.9959
>>>excel_df.to_excel("C:/Users/Desktop/index.xlsx")      #将 excel_df 以 excel 格式保存在指定目录下
```

数据来源：中信建投证券研究发展部

图 5：txt 数据示例



数据来源：中信建投证券研究发展部

表 17：Python 与 txt 的交互

```
>>>df = open("C:/Users/Desktop/sample.txt", encoding="utf-8")
>>>df.read()
'中信建投金融工程为您服务！'
```

数据来源：中信建投证券研究发展部

4.3 Python 与 Wind 客户端的交互

1) 正常 WindPy 接口安装

打开 Wind 资讯终端，点击“量化”选项，出现下方的界面，点击“Python 插件”，会弹出广告说明；

2) 特殊安装 WindPy 方式

假设 Wind 终端安装在 C:\Wind\Wind.NET.Client\WindNET 目录（目录下有 bin 等等子目录），Python 安装在 C:\python28 目录。首先通过 Windows 进入 cmd 命令，然后输入如下命令即可：

C:\Python28\python.exeC:\Wind\Wind.NET.Client\WindNET\bin\installWindPy.py C:\wind\wind.net.client\windnet

按任意键 WindPy 安装过程结束。

说明：以上接口安装说明来自于 wind 客户端，若有更新，请参考最新 wind 说明。

表 18：Python 与 windPy 的交互

```
from WindPy import w      #导入模块
#获取交易时间列表
outdata = w.tdays(date_begin, date_end, "")
print(outdata.Times)
strTemp = ''
for i in range(0,len(outdata.Times)):
    #2017-03-27 00:00:00.005000[0:10]，只截取前 10 个字符，即 2017-03-27
    strTemp=strTemp+str(outdata.Times[i])[0:10]+' '
    #把 strTemp 分割到 list 中
listInfo = strTemp.split()
```

数据来源：中信建投证券研究发展部

说明：接口调用方式请参考 D:\WindNET\bin 目录下的 WindNavigator.exe 或 wind 官方说明

五、Python 自然语言的处理

5.1 jieba 库

jieba 是一个 python 实现的分词库，对中文有着很强大的分词能力。Jieba 安装方法为：

pip install jieba

jieba 库的优点如下：

1) 支持三种分词模式；
精确模式，试图将句子最精确地切开，适合文本分析；
全模式，把句子中所有的可以成词的词语都扫描出来，速度非常快，但是不能解决歧义；
搜索引擎模式，在精确模式的基础上，对长词再次切分，提高召回率，适合用于搜索引擎分词。
2) 支持繁体分词
3) 支持自定义词典

表 19：Python jieba 库常见的操作

```
1.分词
>>>import jieba
>>>sentense = "中信建投金融工程组诚挚服务您"
>>>new_sentense = jieba.cut(sentense)
>>>list(new_sentense)
['中信', '建投', '金融', '工程','组', '诚挚', '服务', '您']
2.加入自定义字典
>>>jieba.load_userdict("C:/Users/Desktop/dict.txt")
>>>new_sentense = jieba.cut(sentense)
>>>list(new_sentense)
['中信建投', '金融工程组', '诚挚', '服务', '您']
3.词性标注
>>>import jieba.posseg as pseg
>>>words = pseg.cut(sentense)
>>>for w in words:
        print(w.word, w.flag)
中信建投    x
金融工程组  x
诚挚        a
服务        vn
您          zg
```

数据来源：中信建投证券研究发展部

除了 jieba 分词外，还有以外分词库等也非常流行：

1） NLTK 库

在使用 Python 处理自然语言的工具中也处于领先的地位。它提供了 WordNet 这种方便处理词汇资源的接口，以及分类、分词、词干提取、标注、语法分析、语义推理等类库。

网站

http://www.nltk.org/

安装 ：

pip install -U nltk

2） TextBlob 库

TextBlob 是一个处理文本数据的 Python 库。它提供了一个简单的 api 来解决一些常见的自然语言处理任务，例如词性标注、名词短语抽取、情感分析、分类、翻译等等。

网站：
http://textblob.readthedocs.org/en/dev/
安装：
pip install -U textblob

六、Python 与机器学习

基于 python 的机器学习库非常多，主要有以下几类：

1）Scikit-learn 是一个简单且高效的数据挖掘和数据分析工具，易上手，可以在多个上下文中重复使用。它基于 NumPy, SciPy 和 matplotlib，开源，可商用（基于 BSD 许可）。

2）Statsmodels 是一个 Python 模块，可以用来探索数据，估计统计模型，进行统计测试。对于不同类型的数据和模型估计，都有描述性统计，统计测试，绘图功能和结果统计的详细列表可用。

3）Shogun 是一个机器学习工具箱，它提供了很多统一高效的机器学习方法。这个工具箱允许多个数据表达，算法类和通用工具无缝组合。

等等。以下主要介绍 Scikit-learn。

6.1 Scikit-learn 库

Scikit-learn 是 Python 里面一个机器学习相关的库，它是构建于 NumPy, SciPy, and matplotlib 基础上的简单高效的数据挖掘和数据分析工具，而且是开源的，内部自带的算法包较多。

安装方法如下：

Anaconda 一般都包含了这些包，但有时可能需要更新，更新方法为：
pip install -U scikit-learn

主要内容如下：

1) 按算法的功能分类，分为分类（classification），回归（regression），聚类（clustering），降维，预处理等。
sklearn 提供了很全面的算法实现；
（具体清单可以参考 http://scikit-learn.org/stable/index.html）
2) 测试数据集，比如 iris，boston 房价等，总共 10 个左右；
3) 数据预处理，比如二值化，正规化，特征提取；
4) 测试数据选择、测试算法以及确定参数，甚至 pipeline 化的支持；
5) 其他支持功能，比如评分 matrix；

使用 sklearn 进行计算的主要步骤：

1) 数据准备。需要把数据集整理为输入 X[sample_count, feature_count]，结果 y[label_count]的格式，其中 sample_count 应该等于 label_count；
2) [可选的降维过程]，原始数据维度大可能会出现 The curse of dimensionality 问题，严重影响性能和算法的扩展性，sklearn 会以降维（PCA 等）或者一些原型算法（Kmeans，Lasso 等，也叫 shrinkage）去掉贡献度低的一些维度；

3) 学习以及预测的过程。生成一个算法的预测器 Estimator，同时可以自己设置参数，比如 K 近邻聚类；调用该预测器的 fit(x,y)函数对输入数据和结果 label 进行学习，从中得到学习的结果，即分类器的各种参数；对未知数据进行预测；

4) 反复学习的过程。仅仅使用一个预测器，或者使用一个预测器的一种参数，对未知数据进行预测可能会有不准确性，所以会使用多种策略：把已知的数据分为多份进行多次计算，常用的是 k-fold ,k-label-fold, leave-1, leave-1-lable 等；多个预测器进行预测，或者独立进行预测，或者组合预测；对一个预测器设置不同的参数进行多次进行预测，同时把数据分组。基本上每一种分类器都有 cross-validation（交叉验证）版本，即把预测器加上 cv，比如 LassoCV。

表 20：sklearn 库示例(以线性回归为例子)

```
>>>from sklearn.linear_model import LinearRegression      #导入模型
>>>x = np.array([[1, 3], [2, 4]])
>>>y = np.array([2, 3])
>>>clf = LinearRegression()                               #创建模型
>>>clf.fit(x, y)                                          #模型拟合
>>>print(clf.coef_)                                       #输出模型拟合系数
[0.5, 0.5]
>>>clf.predict([[1, 5], [3, 5]])                          #模型预测值
array([3, 4])
```

数据来源：中信建投证券研究发展部

6.2 分类算法，以朴素 Bayes 为例

单一的分类方法主要包括：决策树、贝叶斯、人工神经网络、K-近邻、支持向量机和基于关联规则的分类等；另外还有用于组合单一分类方法的集成学习算法，如 Bagging 和 Boosting 等。以下主要介绍朴素贝叶斯。

朴素贝叶斯方法，其中朴素指的是特征条件独立，贝叶斯指的是基于贝叶斯定理。分类：通过学到的概率，给定未分类新实例 X，就可以通过上述概率进行计算，得到该实例属于各类的后验概率 $p(y=c_k \mid X)$，因为对所有的类来说，公式（2）中分母的值都相同，所以只计算分子部分即可，具体步骤如下：先计算该实例属于 $y=c_k$ 类的概率 $p(y=c_k \mid X)=p(y=c_k)\prod_{j=1}^{n} p(X^{(j)}=x^{(j)} \mid y=c_k)$（5）再确定该实例所属的分类 y，其中 $y=\arg\max p(y=c_k \mid X)$。

下面用具体的 sklearn 中的案例来说明朴素 Bayes 分类的应用：

表 21：朴素 Bayes 算法 sklearn 库实现

```
import numpy as np   #导入包
X = np.array([[-1, -1], [-2, -1], [-3, -2], [1, 1], [2, 1], [3, 2]])   '样本数据'
Y = np.array([1, 1, 1, 2, 2, 2])
from sklearn.naive_bayes import GaussianNB   #导入模型
clf = GaussianNB()    #创建模型
clf.fit(X, Y) #模型拟合
print(clf.predict_proba([[-0.8, -1]]))   #输出分别属于两者的概率
#输出两类别的概率：[[   9.99999949e-01    5.05653254e-08]]
```

数据来源：中信建投证券研究发展部

6.3 回归算法：以 Logistic 为例

回归算法是试图采用对误差的衡量来探索变量之间的关系的一类算法。常见的回归算法包括：最小二乘法（Ordinary Least Square），逻辑回归（Logistic Regression），逐步式回归（Stepwise Regression），多元自适应回归样条（Multivariate Adaptive Regression Splines）以及本地散点平滑估计（Locally Estimated Scatterplot Smoothing）。以下主要介绍 Logistic 回归。

Logistic 回归是研究二分类观察结果 y 与一些影响因素 $(x_1, x_2, ..., x_n)$ 之间关系的一种多变量分析方法。通常的问题是，研究某些因素条件下某个结果是否发生。根据线性回归可以预测连续的值，对于分类问题，我们需要输出 0 或者 1。所以，在分类模型中需要将连续值转换为离散值。我们可以预测：当 h_θ 大于等于 0.5 时，输出为 y=1；当 h_θ 小于 0.5 时，输出为 y=0。

Logistic 回归模型的输出变量范围始终在 0 和 1 之间，Logistic 回归模型的假设为：$h_\theta(x) = g(\theta^T x)$，其中：x 代表特征向量，g 代表逻辑函数（Logistic Function），常用逻辑函数为 S 形函数（Sigmoid Function），函数公式为：$g(z) = \frac{1}{1+e^{-z}}$。所以，整个模型的假设为：$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$，该假设函数 $h_\theta(x)$ 的作用是，对于给定的输入变量，根据已经训练好的模型参数计算出输出变量=1 的可能性（estimated probability），即 $h_\theta(x) = P(y=1 \mid x;\theta)$。

在线性回归中，我们将代价函数定义为模型所有误差的平方和。而在逻辑回归中沿用这个定义得到的代价函数是一个非凸函数，这难以用梯度下降法求局部最小值，因此需重新定义逻辑回归的代价函数：$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} Cost(h_\theta(x^{(i)}), y^{(i)})$，其中 Cost()函数定义为：

$$Cost(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & y=1 \\ -\log(1-h_\theta(x)) & y=0 \end{cases}$$

,Cost()函数简化如下：$Cost(h_\theta(x), y) = -y\log(h_\theta(x)) - (1-y)\log(1-h_\theta(x))$

带入代价函数得到代价函数表达式为：$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log(h_\theta(x^{(i)})) + (1-y^{(i)}) \log(1-h_\theta(x^{(i)}))]$。代价函数的求解仍然采用梯度下降法求代价函数的局部最小值：$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$，求导后得到迭代过程：

$\theta_j := \theta_j - \alpha \sum_{i=1}^{m} (h_\theta(x^{(i)} - y^{(i)}) x_j^{(i)}$,Simultaneously update θ_j for j=0,1…n。

表 22：Logistic 回归算法 sklearn 库实现

```
import numpy as np
from urllib.request import urlopen
url = "http://archive.ics.uci.edu/ml/machine-learning-databases/pima-indians-diabetes/pima-indians-diabetes.data"
raw_data = urlopen(url)
dataset = np.loadtxt(raw_data, delimiter=",")
x = dataset[:, 0:7]                                  #从网站下载得到解释变量数据
y = dataset[:, 8]                                    #从网站下载得到被解释变量数据
from sklearn.linear_model import LogisticRegression  #导入模型包
from sklearn import metrics                          #导入模型包
model = LogisticRegression()                         #创建模型
model.fit(x, y)                                      #模型拟合
expected = y
predicted = model.predict_proba(x)                       #模型预测
print(predicted)                                         #输出预测概率
```

输出结果：

```
[[ 0.36782623    0.63217377]
 ...,
 [ 0.88852685    0.11147315]]
```

结果说明：比如第一行，表示第一个样本属于第一类的概率为 0. 36782623，属于第二类的概率为 0. 63217377

数据来源：中信建投证券研究发展部

6.4 聚类算法：以 k-means 为例

聚类指事先并不知道任何样本的类别标号，通过某算法来把一组未知类别的样本划分成若干类别，叫作 unsupervised learning （无监督学习）。在本文中，我们主要介绍一个比较简单的聚类算法：k-means 算法。

我们把样本间的某种距离或者相似性来定义聚类，即把相似的（或距离近的）样本聚为同一类，而把不相似的（或距离远的）样本归在其他类。

k-means 算法是一种很常见的聚类算法，它的基本思想是：通过迭代寻找 k 个聚类的一种划分方案，使得用这 k 个聚类的均值来代表相应各类样本时所得的总体误差最小。k-means 算法的基础是最小误差平方和准则。

k-means 算法的主要实现步骤如下：

1) 从 N 个数据对象中随机挑选 k 个对象当作聚类的初始聚类的中心，即种子点；
2) 分别计算剩下其它对象与这些聚类的中心的相似度即其距离，然后将其分别将它们划分给与相似最多的聚类心；
3) 计算该聚类中所有相关对象的平均值，即点群中心点，然后种子点移动到属于他的“点群”的中心；

4） 重复 2,3 过程，一直到其标准测度的函数开始收敛结束，即种子点没有移动。

表 23：kmeans 算法 sklearn 库实现

```
>>>import matplotlib.pyplot as plt
>>>from sklearn.cluster import KMeans
>>>from sklearn.datasets import make_blobs
>>>n_samples = 1500
>>>random_state = 170
>>>X, y = make_blobs(n_samples=n_samples, random_state=random_state)
>>>y_pred = KMeans(n_clusters=2, random_state=random_state).fit_predict(X) #聚类，分成 2 类
>>>plt.scatter(X[:, 0], X[:, 1], c=y_pred) #散点图
>>>plt.show() #展示
```

数据来源：中信建投证券研究发展部

七、Python 与深度学习

目前常用的深度学习框架包括 Caffe、CNTK、TensorFlow、Theano 和 Torch，keras 等。

Caffe 开始于 2013 年底,具有出色的卷积神经网络实现。在计算机视觉领域 Caffe 依然是最流行的工具包，但对递归网络和语言建模的支持很差。在 Caffe 中图层需要使用 C++定义，而网络则使用 Protobuf 定义。

CNTK 中网络会被指定为向量运算的符号图，运算的组合会形成层。CNTK 通过细粒度的构件块让用户不需要使用低层次的语言就能创建新的、复杂的层类型。

TensorFlow 是一个理想的 RNN（递归神经网络） API 和实现，TensorFlow 使用了向量运算的符号图方法，使得新网络的指定变得相当容易，但 TensorFlow 并不支持双向 RNN 和 3D 卷积，同时公共版本的图定义也不支持循环和条件控制。

Theano 支持大部分先进的网络，现在的很多研究想法都来源于 Theano，它引领了符号图在编程网络中使用的趋势。Theano 的符号 API 支持循环控制，让 RNN 的实现更加容易且高效。

Torch 对卷积网络的支持非常好。Torch 通过时域卷积的本地接口使得它的使用非常直观。Torch 通过很多非官方的扩展支持大量的 RNN，同时网络的定义方法也有很多种。与 Caffe 相比，在 Torch 中定义新图层非常容易，不需要使用 C++编程，图层和网络定义方式之间的区别最小。

本文主要介绍 keras，原因如下：

1) 纯 Python，方便查看/修改源代码
2) 支持 theano 和 Tensorflow 两种模式
3) 配置非常简单，可快速搭建自己的模型
4）文档齐全，社区非常活跃
5）封装的非常好，简单好用

7.1 Keras 框架

1）简单介绍

Keras 是基于 Theano、TensorFlow 的一个深度学习框架，它的设计参考了 Torch，用 Python 语言编写，是

一个高度模块化的神经网络库，支持 GPU 和 CPU（使用的文档为 http://keras.io/）。以下是深度学习几个说明：

激活函数：加入非线性因素的，因为线性模型的表达能力不够

放弃层(Dropout)：防止过拟合

损失函数：模型试图最小化的目标函数，衡量模型预测的好坏

池化层：Mean pooling(均值采样)、Max pooling(最大值采样)、Overlapping (重叠采样)、L2 pooling(均方采样)、Local Contrast Normalization(归一化采样)、Stochasticpooling(随即采样)、Def-pooling(形变约束采样)。其中最经典的是最大池化，作用是降维,可以扩大感知野

优化算法：最常用的为 SGD 算法，也就是随机梯度下降算法

全连接层（Dense）:负责分类或者回归；全连接层会丢失一些特征位置信息，矩阵乘法，相当于一个特征空间变换，可以把有用的信息提取整合；维度变换，尤其是可以把高维变到低维，同时把有用的信息保留下来。

2）框架搭建

1) 以 Windows 版本作为基础环境
2) 目前 Tensorflow 不支持 Windows 版本，所以本文选用 Theano 安装
3) Python 环境建议使用 Anaconda3
4) 安装 Theano ,首先要安装 C++ 编译器，因为 windows 下面没有，所以首先安装 MinGw，这是一个 GCC 的编译环境:
   1、在 cmd 中输入 conda install mingw libpython
   2、配置环境变量：path：C:\Anaconda\MinGW\bin;C:\Anaconda\MinGW\x86_64-w64-mingw32\lib;
   3)、path 中还要加入：C:\Anaconda2;C:\Anaconda2\Scripts;
   注：以以路径请修改为自己的 anaconda 所在位置
5) 安装 theano 库
   pip install theano
6) 安装 keras 库
   pip install keras
7) 在用户文件夹的.keras 子文件夹下找到 keras.json，然后记事本编辑改'tensorflow'为'theano'（不能为 Theano，必须全部小写，否则报错）
8) 验证 keras 是否安装成功
   >>> import keras
   Using Theano backend.

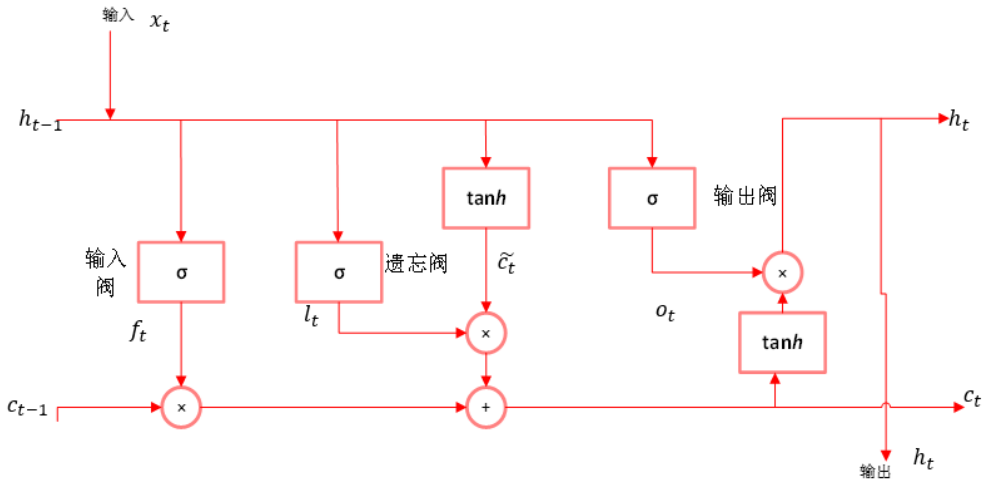
没有报错则恭喜您，深度学习环境已搭建成功！

7.2 长短期记忆网络

经济学家靠 ARMA 模型预测的时间序列模型。该模型对小数据集效果很好，可容纳时间序列的记忆效应，如持久性、均值回归、季节性等。在深入学习中，长短期记忆（Long short-term memory，LSTM）可类比于 ARIMA。

LSTM 是一个循环神经网络，能记忆通过网络预先输入的信息。LSTM 对 RNN 进行了结构上的修改，来避免长期依赖问题。

图 6：LSTM 神经元结构



数据来源：中信建投证券研究发展部

这里，ht 代表神经元的输出，而 Ct 代表神经元的状态。系统的运行分为以下步骤：

A.计算需要被细胞丢弃的信息：$f_t = \sigma\left(W_f\left[h_{t-1}, x_t\right] + b_f\right)$

B.计算需要添加到细胞中的信息 $i_t = \sigma\left(W_i\left[h_{t-1}, x_t\right] + b_i\right)\ \tilde{C}_t = \tanh\left(W_C \cdot\left[h_{t-1}, x_t\right] + b_C\right)$

C.更新细胞状态 $C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$

D.计算输出信息 $o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)\ \text{and}\ h_t = o_t * \tanh\left(C_t\right)$

表 24：LSTM 示例

```
from keras.models import Sequential
from keras.layers import LSTM, Dense
import numpy as np
from numpy.random import seed
seed(1) #种子点可以是指定数字，比如“1”，来保证每次代码运行时生成相同的随机数序列
data_dim = 4   #类似因子数量
timesteps = 3   #类似日期数量
num_classes = 2 #标签分类
num_samples = 5 #训练样本数
num_predict = 3   #待预测数
#生成随机数
x_train = np.random.random((num_samples, timesteps, data_dim))
#y_train = np.random.random((num_samples, num_classes))
y_train = np.random.randint(num_classes, size=(num_samples, num_classes))#真实数据应该为 0 或 1，表示类别
x_test = np.random.random((num_predict, timesteps, data_dim))
model = Sequential()
#将三个 LSTM 堆叠在一起，是该模型能够学习更高层次的时域特征表示
model.add(LSTM(32, return_sequences=True,input_shape=(timesteps, data_dim)))
model.add(LSTM(32, return_sequences=True))
model.add(LSTM(32))
model.add(Dense(num_classes, activation='softmax')) #全连接层，激活函数为 softmax
#设置目标函数即 loss 函数，优化算法，及衡量指标
model.compile(loss='categorical_crossentropy',
                    optimizer='rmsprop',
                    metrics=['accuracy'])
model.fit(x_train, y_train,batch_size=100, epochs=50)#训练模型，设置批大小及训练次数
y   = model.predict(x_test) #预测
print(y )
[[ 0.5076285     0.49237153]
 [ 0.50342733    0.4965727 ]
 [ 0.50616741    0.49383259]]
```

结果说明：比如第一行可理解为第一只个股未来表现下跌的概率为 0. 5076285，表现上涨的概率为 0. 49237153

数据来源：中信建投证券研究发展部

7. 3 卷积神经网络

卷积神经网络（Convolutional Neural Network，CNN）一种专门处理图像的特殊的多层神经网络，包括卷积层(alternating convolutional layer)和池层(pooling layer)。CNN 的基本结构一般包括两层，一为特征提取层，每个神经元的输入与前一层的局部接受域相连，并提取该局部的特征。随着该局部特征被提取，它与其它特征间的位置关系也确定下来。二是特征映射层，网络的每个计算层由多个特征映射组成，每个特征映射是一个平面，

平面上所有神经元的权值相等。特征映射结构采用影响函数核小的 sigmoid 函数作为卷积网络的激活函数，使得特征映射具有位移不变性。

表 25：卷积神经网络代码示例

```
from keras.models import Sequential
from keras.layers import Dense, Dropout
from numpy.random import seed
seed(1) #种子点可以是指定数字，比如“1”，来保证每次代码运行时生成相同的随机数序列
from keras.layers import Conv1D, GlobalAveragePooling1D
import numpy as np
num_classes = 2    #类别数
num_samples = 30 #训练样本数
num_predict = 3    #待预测数
x_train = np.random.random((num_samples, 20))
y_train = np.random.randint(num_classes, size=(num_samples, 1))
x_test = np.expand_dims(np.random.random((num_predict,20)), axis=2)
row,col = x_train.shape
#expand_dims：把 a*b 的 array 变换成 a 个 b*1 的 array，成为深度学习需要的输入形式
x_train = np.expand_dims(x_train, axis=2)
model = Sequential()
model.add(Conv1D(64, 3, activation='relu', input_shape=(col,1)))#加入卷积层，激活函数为 relu
model.add(GlobalAveragePooling1D())#加入池化层
#放弃层，训练过程中每次更新参数时随机断开一定百分比的输入神经元，防止过拟合。这里是断开 50%的输入神经元。
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid')) #全连接层
#设置目标函数即 loss 函数，优化算法，及衡量指标
model.compile(loss='binary_crossentropy',
                    optimizer='rmsprop',
                    metrics=['accuracy'])
history = model.fit(x_train, y_train, batch_size=16, epochs=3) #训练模型
y    = model.predict(x_test) #预测
print(y )
输出结果：
[[ 0.51958132]
 [ 0.51438302]
 [ 0.51856101]]
结果说明：因为此处是二分类，目标标签为 0 和 1，比如第一个结果 0. 51958132 说明该样本有 0. 51958132 概率上涨。
```

数据来源：中信建投证券研究发展部

八、人工智能因子打分策略

8.1 策略代码实战

整体策略模型，我们基本都是模块化编写，即各功能都是严格分开编写，方便后续修改，也方便代码重用。各大模块代码实战分析：

1) 提取数据库模块代码分析

该模块功能主要用于从数据库提取数据，但不作任何处理。

表 26：提取数据库核心代码

```
import pyodbc
import configInfo
import time
import numpy as np
#连接数据库函数
def conFactor():
    con = {}
    conInfo = 'driver={SQL Server};server='+host+','+port+';database='+db_yyy_Factor+';uid='+user+';pwd='+pwd
    conn = pyodbc.connect(conInfo)
    cur = conn.cursor()
    con['conn'] = conn
    con['cur'] = cur
    return con

#关闭数据库
def close(cursor,conn):
    #关闭
    cursor.close()
    conn.close()

#查询数据库
def queryFactors(list_dates,list_code):
    list_temp = []
    mycon = conFactor()
    conn = mycon['conn']
    cur = mycon['cur']
    #一年的数据
    sql_data = "SELECT S_INFO_WINDCODE,TRADE_DT "+ str_factors +" FROM " + factor_month +" where
        TRADE_DT    in ("+strdate+") and S_INFO_WINDCODE in ("+strcode+")";
    cur.execute(sql_data)
    for row in cur.fetchall():
        dic_temp = {}
        dic_temp['stockcode'] = row[0]
        dic_temp['TRADE_DT'] = row[1]
        for i in range(len(factors)):
            dic_temp[factors[i]] = row[2+i] #根据因子名动态生成 dic
        list_temp.append(dic_temp)
    close(cur,conn)
    return list_temp
```

数据来源：中信建投证券研究发展部

核心代码分析：

def conFactor():

该函数为数据库连接模块，顾名思义，就是连接数据库模块

import pyodbc：导入连接数据库驱动模块，pyodbc 用于 Python 连接 sql server 数据库

import configInfo：导入参数模块，为了方便修改，我们把所有参数放在该模块

sql_data：查询语句，此处功能是查询指定个股指定时间的指定因子值

cur.execute(sql_data)：执行查询语句

for row in cur.fetchall():用于获取所有查询到的数据，然后进行封装。

Python 需要严格缩进。否则代码则出错。比如函数体都需要缩进，for 循环等也需要缩进。

2) 数据预处理模块

a.没满一年的新股不进行机器学习因子计算：因为需要用最近历史一年的数据作为训练。

b.对于缺失值，用平均值代替，当缺失达到 10%，则该因子丢弃。

c. z-score 标准化，要求原始数据的分布可以近似为高斯分布，否则效果不好。

对 a_value,turnover_1 等这一类不不符合高斯分布因子，需要用 ln(t1/t0)（同一个股当期与上期比值的对数）进行处理，才近似高斯分布。但对 sec_return_1,MACD 等这一类变化率等相关因子，直接用原始值便可以，因为他们本身已经近似符合正态分布。

预处理之所有没有处理掉极值和去掉涨跌停个股因子，原因是因为此处只是训练特征，而不是最终选股。再次，我们所选的因子是经过人工核对的，基本没有太多相似性，故也没有降维这一步。

表 27：数据预处理核心代码

```
import numpy as np
from utils import publicVariables
from sklearn import preprocessing
temp_factor[i,:] = np.log(factor[i]/temp[i-1])
array_data = preprocessing.scale(array_data)
```

数据来源：中信建投证券研究发展部

核心代码分析：

from utils import publicVariables：有些公用变量，建议单独建在一个模块，这样方便调用。

from sklearn import preprocessing：导入预处理模块。

temp_factor[i,:] = np.log(factor[i]/temp[i-1])：经验告诉我们，有些因子已经近似为高斯分布，比如动量因子，普通的因子经过取对数后与高斯分布比较接近，比如流动市值因子。

array_data = preprocessing.scale(array_data)：z-score 标准化，要求原始数据的分布可以近似为高斯分布，否则效果不好。

3) 中性化处理模块

中性化处理我们包含二层含义，一是市值中性化，二是行业中性化。

首先，我们都知道，市值因子对个股的影响十分显著，如果不考虑市值带来的干扰，则我们的策略可能被市值因子带来严重的影响。为此，我们市值分成 20 组，分别在不同市值组各选取 20%作为策略多头与空头，使多头与空头有相同的市值分布，以消除市值可能带来的影响。

其次，众所周知，不同行业，因子特征可能差异明显，放在一起可能不具备可比性。为了去除行业带来的影响，我们也分别在不同行业选取 20%作为我们的空头与多头，使多头与空头保持同样的行业暴露，以消除行业带来的影响。

中性化处理代码相对比较简单，且基本是按照逻辑编写便可，在此不作详细介绍。

4) 机器学习模块

机器学习模块是核心，也是重点，也最简单，因为一般情况下，我们没有特殊要求，直接调用现在的机器学习包便可。重点要注意的是处理好数据成机器学习输入的形式就可以了。一般机器学习算法的输入形式为：自变量为 n*m 数组，n 为样本数量，m 为因子个数，因变量为对应的标签列表。

表 28：机器学习模块（以 knn 为例）

```
import numpy as np
import math
from sklearn.neighbors import KNeighborsClassifier
k = math.floor(np.sqrt(len(array_data)))  #经验参数，n 一般少于训练集的平方根
knn = KNeighborsClassifier(n_neighbors=k,weights='distance')
knn.fit(array_data,list_target)
list_predict = list(knn.predict_proba(array_predict)[:,1])
```

数据来源：中信建投证券研究发展部

核心代码分析：

k = math.floor(np.sqrt(len(array_data)))：经验参数，n 一般少于训练集的平方根。表示 n 个邻近。

knn = KNeighborsClassifier(n_neighbors=k,weights='distance')：实例化算法，weights 为加权方式，weights='distance',对距离加权，可以降低 k 值设定的影响。

knn.fit(array_data,list_target) ：模型训练拟合。

list_predict = list(knn.predict_proba(array_predict)[:,1])：模型预测结果。

5) 策略计算模块

该模块主要负责计算策略整体评估功能，主要包括：计算多空收益差模块，计算 IC 模块，分 N 组计算各组收益模块。主要考虑了以下几种情况：

a. 当期单个因子在全市场缺失达 40%时，则该因子丢弃，不进行计算。
b. 调仓当天停牌，涨停，跌停个股剔除。
c. 新股一个月之内不能作为候选股（上市小于 20 个交易日）。

表 29：计算 IC 核心代码

```
import pandas as pd
order_data = pd.Series(order_data)
order_ret = pd.Series(order_ret)
IC = order_data.corr(order_ret)
```

数据来源：中信建投证券研究发展部

核心代码分析：

import pandas as pd：导入 pandas 模块

order_data = pd.Series(order_data)：原型为 pd.Series(data, index=index)， data 是数据源，可以是 Python 字典类型，ndarray 或者标量值。index 代表轴标签，传递列表类型。若 index 省略，则默认为[0,1,2,…,len(data)-1]

order_data.corr(order_ret)：计算序列 order_data 与序列 order_ret 的相关系数。

6) 结果入库模块

该模块功能简单明了，即只是把计算结果保存到数据库。

表 30：结果入库核心代码

```
sql = "insert into "+table+tableField+" values("+factors_str+")";
try:
    cur.execute(sql)
    conn.commit()
except pyodbc.Error as e:
    print (sql)
```

数据来源：中信建投证券研究发展部

核心代码分析：

sql = "insert into "+table+tableField+" values("+factors_str+")"：插入数据库语句

cur.execute(sql)：执行插入数据库语句

conn.commit()：提交执行插入数据库语句

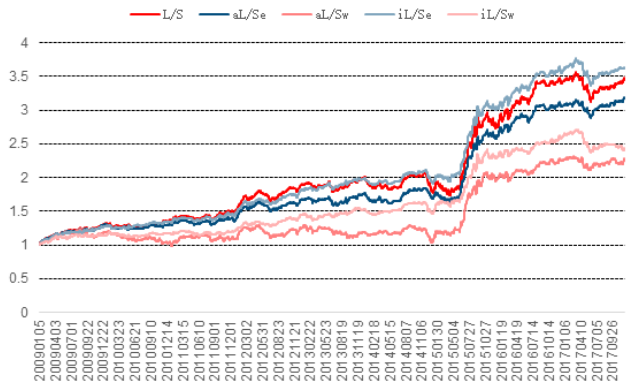
except pyodbc.Error as e:

print (sql) ：若执行错误则输出。目的是为了调试，出了问题也方便第一时间找到原因。

8.2 策略结果(以 Logistic 为例)

在 20090105 到 20171130 期间，我们分别进行了全市场选股，市值中性选股，行业中性选股，五种情况表现如下：

图 7：人工智能选股多空收益差策略净值



数据来源：wind 资讯，中信建投证券研究发展部

相关说明如下：

1） 所用因子：全市场训练得到的个股未来相对强势值。
2） L/S：全市场选股多空收益差净值。相对强势值排名靠前 20%作为多头，相对强势值排名后 20%作为空头。
3） aL/Se:市值等权多空收益差净值。分 20 小组，分别在组内选前 20%作为多头，后 20%作为空头，最后各组等权。
4） aL/Sw:市值加权多空收益差净值。分 20 小组，分别在组内选前 20%作为多头，后 20%作为空头，最后各组以市值组权重加权得到多空组合。
5） iL/Se:行业等权多空收益差净值。在中信一级行业，分别在行业内选前 20%作为多头，后 20%作为空头，最后各行业以等权到多空组合。
6） iL/Sw:行业加权多空收益差净值。在中信一级行业，分别在行业内选前 20%作为多头，后 20%作为空头，最后各行业以沪深 300 行业内权重加权得到多空组合。

九、Python 股票策略打包成 Exe 文件

9.1 运用 pyinstaller 打包成 exe

在实际运行中，我们可能希望我们的模型能定期在后台运行，比较简单的方法就是打包成 exe 可执行文件，双点便可在后台运行。Python 中提供了可以将.py 文件打包成可执行文件 exe 的 pyinstall 包，其具体使用步骤如下:

1) 安装：pip install pyinstaller
2) Pyinstaller 打包：
D:\Anaconda3\Scripts\pyinstaller.exe -F -w C:\Users\ Desktop\work\testPyinstaller\src\test.py
注：Pyinstaller 的根目录：D:\Anaconda3\Scripts\pyinstaller.exe
需要打包的文件目录：C:\Users\ Desktop\work\testPyinstaller\src\test.py
实际运行请对应修改。
3) C:\Users\下面增加了 build 和 dist 文件夹，需要的 test.exe 在 dist 文件夹内。双击即可在后台运用

更多说明请参考链接：http://www.pyinstaller.org/

十、模型代码编写建议

作为一位量化投资相关人员，虽然编写代码不是核心，但拥有一个良好的编写代码习惯也是提高工作效率的关键。在此给出几点个人的代码编写建议。

1） 在开始编码之前，一定要有个大致的设计，比如模型分几大块：控制层模块（相当于总设计），读取数据模块、数据预处理模块、模型核心算法模块、数据结果保存或者展示模块等（通过情况下，我们一个简单的策略都可以分成这几大块）。
2） 优秀的代码文档跟编程语句一样重要。在代码源文件中，应该为主要的代码段添加注释，解释代码的基本逻辑，若模型交接方便接手人了解代码也方便日后自己回顾。若是日后修正代码，则应该注明修改日期，以及修改的原因。
3） 最好有一个 README 文件，注明每个源文件、数据文件等的作用。整个模型流程及功能及模型需要注意的事项也应该加以注明。

4） 变量名和函数名称尽可能写得有意义。例如，收盘价数组可定义为 array_close，而不是直接写 close 或者 c，array_close 非常直观的可以让人知道这是一个 array 数组，用来存放 close 的。

5） 重复的代码一定不要出现。可用函数形式，用到时调用。这样方便日后修改，也不会出现某处忘记修改的情况。

6） 最后一项，也不是最不重要的一项，每次修改版本之前，一定要备份，否则出了问题，无法回到之前版本，那将是令人抓狂的一件事情。

分析师介绍

丁鲁明：同济大学金融数学硕士，中国准精算师，现任中信建投证券研究发展部金融工程方向负责人，首席分析师。9 年证券从业，历任海通证券研究所金融工程高级研究员、量化资产配置方向负责人；先后从事转债、选股、高频交易、行业配置、大类资产配置等领域的量化策略研究，对大类资产配置、资产择时领域研究深入，创立国内“量化基本面”投研体系。多次荣获团队荣誉：新财富最佳分析师 2009 第 4、2012 第 4、2013 第 1、2014 第 3 等；水晶球最佳分析师 2009 第 1、2013 第 1 等。

研究助理 喻银尤：021-68821600-808 yuyinyou@csc.com.cn
复旦大学计算机硕士，通过 CFA 三级，两年上交所相关部门工作经验，专注于大数据、多因子、人工智能等相关策略研究。

研究服务

社保基金销售经理

彭砚苹 010-85130892pengyanping@csc.com.cn

姜东亚 010-85156405jiangdongya@csc.com.cn

机构销售负责人

赵海兰 010-85130909zhaohailan@csc.com.cn

北京非公募组

张博 010-85130905zhangbo@csc.com.cn

李祉瑶 010-85130464lizhiyao@csc.com.cn

周瑞 010-85130749zhourui@csc.com.cn

刘凯 010-86451013liukaizgs@csc.com.cn

张勇 zhangyongzgs@csc.com.cn

北京公募组

黄玮 010-85130318huangwei@csc.com.cn

朱燕 85156403zhuyan@csc.com.cn

任师蕙 010-8515-9274renshihui@csc.com.cn

黄杉 010-85156350huangshan@csc.com.cn

王健 010-65608249wangjianyf@csc.com.cn

上海地区销售经理

黄方禅 021-68821615huangfangchan@csc.com.cn

戴悦放 021-68821617daiyuefang@csc.com.cn

邓欣 dengxin@csc.com.cn

谈祺阳 tanqiyang@csc.com.cn

翁起帆 wengqifan@csc.com.cn

深广地区销售经理

胡倩 0755-23953981huqian@csc.com.cn

许舒枫 xushufeng@csc.com.cn

程一天 chengyitian@csc.com.cn

曹莹 caoyingzgs@csc.com.cn

张苗苗 zhangmiaomiao@csc.com.cn

廖成涛 liaochengtao@csc.com.cn

陈培楷 chenpeikai@csc.com.cn

评级说明

以上证指数或者深证综指的涨跌幅为基准。

买入：未来 6 个月内相对超出市场表现 15%以上；

增持：未来 6 个月内相对超出市场表现 5—15%；

中性：未来 6 个月内相对市场表现在-5—5%之间；

减持：未来 6 个月内相对弱于市场表现 5—15%；

卖出：未来 6 个月内相对弱于市场表现 15%以上。

重要声明

中信建投证券研究发展部

北京

东城区朝内大街 2 号凯恒中心 B 座 12 层（邮编：100010）

电话：(8610) 8513-0588

传真：(8610) 6560-8446

上海

浦东新区浦东南路 528 号上海证券大厦北塔 22 楼 2201 室（邮编：200120）

电话：(8621) 6882-1612

传真：(8621) 6882-1622

深圳

福田区益田路 6003 号荣超商务中心 B 座 22 层（邮编：518035）

电话：（0755）8252-1369

传真：（0755）2395-3859