数据结构

c=[1,’abc’,[1,2]] #列表#

d={‘today’:20,’tomorrow’:30} #字典#

s={1,2,3} #集合#

------------------------------------------------------------------------------

a=2

b=a\*\*3 #平方#

print(b)

------------------------------------------------------------------------------

s='I like python'

t=s+' very much' #链接字符串#

print(t)

s.split(' ') #切分字符串#

s='I:like:python'

t=s+' very much'

print(t)

s.split(':')

------------------------------------------------------------------------------

if a==2:

print(a)

else:

print ('a不等于1')

------------------------------------------------------------------------------

s=0

k=0

while k<100:

k=k+1

s=s+k

print (s)

s=0

for k in range(101):

s=s+k

print(s)

------------------------------------------------------------------------------

def mylove(x,y):

return x\*\*2,y+5

a,b=mylove(2,3)

print(a,b)

f=lambda x,y:x\*\*2+y # 定义函数f(x,y)=x^2+y #

print(f(2,3))

------------------------------------------------------------------------------

a=[1,2,3]

b=map(lambda x:x+2,a)

b=list(b)

from fuctools import reduce

reduce(lambda x,y:x\*y,range(1,n+1))

b=filter(lambda x:x>5 and x<8,range(10))

b=list(b)

------------------------------------------------------------------------------

a=[1,'a',2,8]

b=a

b=a[:]

------------------------------------------------------------------------------

len(a)

max(a)

min(a)

sum(a)

sorted(a)

a.append(3)

a.count(3)

a.extend([2,4])

a.index(3)

a.insert(2,7)

a.pop(3)

a=[2,3,4]

b=[]

for i in a:

b.append(i+2)

a=[2,3,4]

b=[i+3 for i in a]

------------------------------------------------------------------------------

a={1,2,2,3}

b={2,3,4,5}

o=a|b

p=a&b

q=a-b

r=a^b

------------------------------------------------------------------------------

import math as m

m.sin(1)

m.exp(1)

m.pi

from math import exp as e

e(1)

------------------------------------------------------------------------------

pip install numpy

------------------------------------------------------------------------------

import numpy as np

b=np.array([[1,2,3],[4,5,6]])

print(b)

print(b\*\*2)

------------------------------------------------------------------------------

numpy

scipy

matplotlib

pandas

statsmodels

scikit-learn

keras

gensim

------------------------------------------------------------------------------

import numpy as np

import matplotlib.pyplot as plt

x=np.linspace(0,10,1000)

y=np.sin(x)+1

z=np.cos(x\*\*2)+1

plt.figure(figsize=(8,4))

plt.plot(x,y,label='$\sin x+1$',color='red',linewidth=2)

plt.plot(x,z,'b--',label='$\cos x^2+1$')

plt.xlabel('Time(s)')

plt.ylabel('Volt')

plt.title('A Simple Example')

plt.ylim(0,2.2) #y轴范围#

plt.legend() #显示图例#

plt.show()

plt.rcParams[‘font.sans-serif’]=[‘SimHei’] #正常显示中文标签#

plt.rcParams[‘axes.unicode\_minus’]=False #解决图像不能显示负号问题#

------------------------------------------------------------------------------

pip install xlrd #添加读取Excel功能#

pip install xlwt #添加写入Excel功能#

----------------------------------------------------------------------------------

import pandas as pd

s=pd.Series([1,2,3],index=['a','b','c'])

d=pd.DataFrame([[1,2,3],[4,5,6]],columns=['a','b','c'])

d2=pd.DataFrame(s)

d.head()

d.describe()

pd.read\_excel (‘data.xls’)

pd.read\_csv (‘data.csv’,encoding=’utf-8’)

----------------------------------------------------------------------------------

平稳性检验

from statsmodels.tsa.stattools import adfuller as ADF

import numpy as np

ADF(np.random.rand(1000))

----------------------------------------------------------------------------------

SVM模型

from sklearn import datasets

iris=datasets.load\_iris()

print(iris.data.shape)

from sklearn import svm

clf=svm.LinearSVC()

clf.fit(iris.data,iris.target)

clf.predict([[5.0,3.6,1.3,0.25]])

clf.coef\_

----------------------------------------------------------------------------------

画箱线图

#-\*- coding: utf-8 -\*-

import pandas as pd

catering\_sale = 'C:/Users/penny/Desktop/useful/code/chapter3/data/catering\_sale.xls' #餐饮数据

data = pd.read\_excel(catering\_sale, index\_col = u'日期') #读取数据，指定“日期”列为索引列

import matplotlib.pyplot as plt #导入图像库

plt.rcParams['font.sans-serif'] = ['SimHei'] #用来正常显示中文标签

plt.rcParams['axes.unicode\_minus'] = False #用来正常显示负号

plt.figure() #建立图像

p = data.boxplot() #画箱线图，直接使用DataFrame的方法

x = p['fliers'][0].get\_xdata() # 'flies'即为异常值的标签

y = p['fliers'][0].get\_ydata()

y.sort() #从小到大排序，该方法直接改变原对象

#用annotate添加注释

#其中有些相近的点，注解会出现重叠，难以看清，需要一些技巧来控制。

#以下参数都是经过调试的，需要具体问题具体调试。

for i in range(len(x)):

if i>0:

plt.annotate(y[i], xy = (x[i],y[i]), xytext=(x[i]+0.05 -0.8/(y[i]-y[i-1]),y[i]))

else:

plt.annotate(y[i], xy = (x[i],y[i]), xytext=(x[i]+0.08,y[i]))

plt.show() #展示箱线图

----------------------------------------------------------------------------------

#-\*- coding: utf-8 -\*-

#餐饮销量数据统计量分析

from \_\_future\_\_ import print\_function

import pandas as pd

catering\_sale = 'D:/python/chapter3/data/catering\_sale.xls' #餐饮数据

data = pd.read\_excel(catering\_sale, index\_col = u'日期') #读取数据，指定“日期”列为索引列

data = data[(data[u'销量'] > 400)&(data[u'销量'] < 5000)] #过滤异常数据

statistics = data.describe() #保存基本统计量

statistics.loc['range'] = statistics.loc['max']-statistics.loc['min'] #极差

statistics.loc['var'] = statistics.loc['std']/statistics.loc['mean'] #变异系数

statistics.loc['dis'] = statistics.loc['75%']-statistics.loc['25%'] #四分位数间距

print(statistics)

----------------------------------------------------------------------------------

#-\*- coding: utf-8 -\*-

#菜品盈利数据 帕累托图

from \_\_future\_\_ import print\_function

import pandas as pd

#初始化参数

dish\_profit = 'D:/python/chapter3/data/catering\_dish\_profit.xls' #餐饮菜品盈利数据

data = pd.read\_excel(dish\_profit, index\_col = u'菜品名')

data = data[u'盈利'].copy()

data.sort(ascending = False)

import matplotlib.pyplot as plt #导入图像库

plt.rcParams['font.sans-serif'] = ['SimHei'] #用来正常显示中文标签

plt.rcParams['axes.unicode\_minus'] = False #用来正常显示负号

plt.figure()

data.plot(kind='bar')

plt.ylabel(u'盈利（元）')

p = 1.0\*data.cumsum()/data.sum()

p.plot(color = 'r', secondary\_y = True, style = '-o',linewidth = 2)

plt.annotate(format(p[6], '.4%'), xy = (6, p[6]), xytext=(6\*0.9, p[6]\*0.9), arrowprops=dict(arrowstyle="->", connectionstyle="arc3,rad=.2")) #添加注释，即85%处的标记。这里包括了指定箭头样式。

plt.ylabel(u'盈利（比例）')

plt.show()

----------------------------------------------------------------------------------

#-\*- coding: utf-8 -\*-

#餐饮销量数据相关性分析

from \_\_future\_\_ import print\_function

import pandas as pd

catering\_sale = 'D:/python/chapter3/data/catering\_sale\_all.xls' #餐饮数据，含有其他属性

data = pd.read\_excel(catering\_sale, index\_col = u'日期') #读取数据，指定“日期”列为索引列

data.corr() #相关系数矩阵，即给出了任意两款菜式之间的相关系数

data.corr()[u'百合酱蒸凤爪'] #只显示“百合酱蒸凤爪”与其他菜式的相关系数

data[u'百合酱蒸凤爪'].corr(data[u'翡翠蒸香茜饺']) #计算“百合酱蒸凤爪”与“翡翠蒸香茜饺”的相关系数

----------------------------------------------------------------------------------

d.sum()

d.mean()

d.var()

d.std()

#相关系数#

d.corr(method=’pearson’)

s1.corr(s2,method=’pearson’)

import pandas as pd

D=pd.DataFrame([range(1,8),range(2,9)]) #生成样本D，第一行为1-7，第二行为2-8#

D.corr(method='pearson')

S1=D.loc[0] #提取第一行#

S2=D.loc[1] #提取第二行#

S1.corr(S2,method='pearson') #计算第一行和第二行的相关系数#

#协方差#

d.cov()

s1.cov(s2)

import numpy as np

D=pd.DataFrame(np.random.randn(6,5)) #生成6\*5的随机矩阵#

D.cov()

D[0].cov(D[1]) #计算第一列和第二列的协方差#

D.skew()/D.kurt() #偏度/峰度#

D.describe()

D.cumsum()

D.cumprod()

D.cummax()

D.cummin()

----------------------------------------------------------------------------------

#rolling 系列函数#

rolling\_sum()

rolling\_mean()

rolling\_var()

rolling\_std()

rolling\_corr()

rolling\_cov()

rolling\_skew()

rolling\_kurt()

D=pd.Series(range(0,20))

pd.rolling\_sum(D,2)

----------------------------------------------------------------------------------

#作图#

plt.plot(x,y,S)

plt.pie(size)

plt.hist(x,y)

D.plot(kind=’box’) #line, bar, barh, hist, box, kde, area, pie#

D.boxplot()

D.plot(logx=Ture)/D.plot(logy=True)

D.plot(yerr=error)

#线性图#

import numpy as np

x=np.linspace(0,2\*np.pi,50)

y=np.sin(x)

plt.plot(x,y,’bp--’)

plt.show()

#饼图#

import matplotlib.pyplot as plt

labels='Frogs','Hogs','Dogs','Logs'

sizes=[15,30,45,10]

colors=['yellowgreen','gold','lightskyblue','lightcoral']

explode=(0,0.1,0,0)

plt.pie(sizes,explode=explode,labels=labels,colors=colors,autopct='%1.1f%%',shadow=True,startangle=90)

plt.axis('equal')

plt.show()

#直方图#

import matplotlib.pyplot as plt

import numpy as np

x=np.random.randn(1000)

plt.hist(x,10) #分成10组绘制直方图#

plt.show()

#箱线图#

import matplotlib.pyplot as plt

import numpy as np

import pandas as pd

x=np.random.randn(1000)

D=pd.DataFrame([x,x+1]).T

D.plot(kind='box')

plt.show()

#误差图#

import matplotlib.pyplot as plt

import numpy as np

import pandas as pd

plt.rcParams['font.sans-serif']=['SimHei']

plt.rcParams['axes.unicode\_minus']=False

error=np.random.randn(10)

y=pd.Series(np.sin(np.arange(10)))

y.plot(yerr=error)

plt.show()

----------------------------------------------------------------------------------

#拉格朗日插值代码

import pandas as pd #导入数据分析库Pandas

from scipy.interpolate import lagrange #导入拉格朗日插值函数

inputfile = 'D:/python/chapter3/data/catering\_sale.xls' #销量数据路径

outputfile = 'D:/python/output/sales.xls' #输出数据路径

data = pd.read\_excel(inputfile) #读入数据

data[u'销量'][(data[u'销量'] < 400) | (data[u'销量'] > 5000)] = None #过滤异常值，将其变为空值

#自定义列向量插值函数

#s为列向量，n为被插值的位置，k为取前后的数据个数，默认为5

def ployinterp\_column(s, n, k=5):

y = s[list(range(n-k, n)) + list(range(n+1, n+1+k))] #取数

y = y[y.notnull()] #剔除空值

return lagrange(y.index, list(y))(n) #插值并返回插值结果

#逐个元素判断是否需要插值

for i in data.columns:

for j in range(len(data)):

if (data[i].isnull())[j]: #如果为空即插值。

data[i][j] = ployinterp\_column(data[i], j)

data.to\_excel(outputfile) #输出结果，写入文件

----------------------------------------------------------------------------------

#-\*- coding: utf-8 -\*-

#数据规范化

import pandas as pd

datafile = 'D:/python/chapter4/data/normalization\_data.xls' #参数初始化

data = pd.read\_excel(datafile, header = None) #读取数据

(data - data.min())/(data.max() - data.min()) #最小-最大规范化

(data - data.mean())/data.std() #零-均值规范化

data/10\*\*np.ceil(np.log10(data.abs().max())) #小数定标规范化

----------------------------------------------------------------------------------

#-\*- coding: utf-8 -\*-

#数据规范化

import pandas as pd

datafile = 'D:/python/chapter4/data/discretization\_data.xls' #参数初始化

data = pd.read\_excel(datafile) #读取数据

data = data[u'肝气郁结证型系数'].copy()

k = 4

d1 = pd.cut(data, k, labels = range(k)) #等宽离散化，各个类比依次命名为0,1,2,3

#等频率离散化

w = [1.0\*i/k for i in range(k+1)]

w = data.describe(percentiles = w)[4:4+k+1] #使用describe函数自动计算分位数

w[0] = w[0]\*(1-1e-10)

d2 = pd.cut(data, w, labels = range(k))

from sklearn.cluster import KMeans #引入KMeans

kmodel = KMeans(n\_clusters = k, n\_jobs = 4) #建立模型，n\_jobs是并行数，一般等于CPU数较好

kmodel.fit(data.reshape((len(data), 1))) #训练模型

c = pd.DataFrame(kmodel.cluster\_centers\_).sort(0) #输出聚类中心，并且排序（默认是随机序的）

w = pd.rolling\_mean(c, 2).iloc[1:] #相邻两项求中点，作为边界点

w = [0] + list(w[0]) + [data.max()] #把首末边界点加上

d3 = pd.cut(data, w, labels = range(k))

def cluster\_plot(d, k): #自定义作图函数来显示聚类结果

import matplotlib.pyplot as plt

plt.rcParams['font.sans-serif'] = ['SimHei'] #用来正常显示中文标签

plt.rcParams['axes.unicode\_minus'] = False #用来正常显示负号

plt.figure(figsize = (8, 3))

for j in range(0, k):

plt.plot(data[d==j], [j for i in d[d==j]], 'o')

plt.ylim(-0.5, k-0.5)

return plt

cluster\_plot(d1, k).show()

cluster\_plot(d2, k).show()

cluster\_plot(d3, k).show()

----------------------------------------------------------------------------------

#-\*- coding: utf-8 -\*-

#线损率属性构造

import pandas as pd

#参数初始化

inputfile= '../data/electricity\_data.xls' #供入供出电量数据

outputfile = '../tmp/electricity\_data.xls' #属性构造后数据文件

data = pd.read\_excel(inputfile) #读入数据

data[u'线损率'] = (data[u'供入电量'] - data[u'供出电量'])/data[u'供入电量']

data.to\_excel(outputfile, index = False) #保存结果

----------------------------------------------------------------------------------

#主成分#

sklearn.decomposition.PCA(n\_components=None, copy=True, whiten=False)

#-\*- coding: utf-8 -\*-

#主成分分析 降维

import pandas as pd

#参数初始化

inputfile = 'D:/python/chapter4/data/principal\_component.xls'

outputfile = 'D:/python/output/dimention\_reducted.xls' #降维后的数据

data = pd.read\_excel(inputfile, header = None) #读入数据

from sklearn.decomposition import PCA

pca = PCA(3)

pca.fit(data)

low\_d=pca.transform(data)

pd.DataFrame(low\_d).to\_excel(outputfile)

pca.components\_ #返回模型的各个特征向量

pca.explained\_variance\_ratio\_ #返回各个成分各自的方差百分比

pca.inverse\_transform(low\_d)

----------------------------------------------------------------------------------

from scipy.interpolate import\*

f=scipy.interpolate.lagrange(x,y)

np.unique(D)

D.unique()

D.isnull/D.notnull

np.random.rand(k,m,n……)

np.random.randn(k,m,n……)

----------------------------------------------------------------------------------