# 《数据挖掘导论》实验 3: 可视化分析实验

# 一、实验目的：

（1）了解matplotlib的绘图组件

（2）掌握pandas常用的绘图方法

# 二、实验环境：

（1）Anaconda2 开发环境

（2）IDE是ipython notebook

（3）使用的库有numpy，pandas，matplotlib

# 三、实验内容：

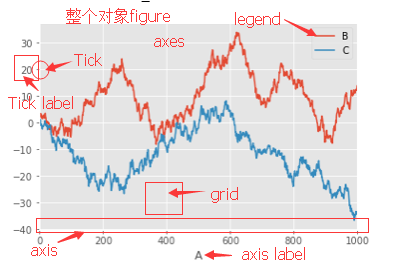
## 1. 使用pandas读入数据。

1. 使用pandasd的read\_csv函数

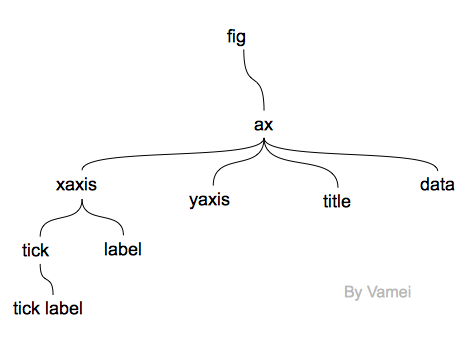
read\_csv(filepath\_or\_buffer, sep=',', header='infer', names=None, index\_col=None, usecols=None)

* 参数
* filepath\_or\_buffer：数据文件的路径
* sep：自定义分隔符，默认为逗号’,’，
* header：用作列名的行号。默认为0（第一行），如果没有header行就应该设置为None
* names：用于结果的列名列表，结合header=None
* index\_col：用作行索引的列标号或列名。可以使单个名称/数字组成的列表（层次化索引）。
* usecols：一维素组，素组元素可以是整数表示字段索引号，也可以字符串表示字段名，返回素组指定的列。
* parse\_dates：True表示将字符串解析成时间对象

## 认识matplotlib的绘图对象



figure是整个绘图对象，axes可理解为一个画板，可以在上面画各种图形，legend可以显示各个曲线的名称，axis可理解为是一个x轴或者y轴的对象，Tick是axis上的刻度，axis有名称，每一个刻度也有名称。grid参数控制是否绘制网格线。他们的包含关系如下图所示：



图片来源：http://www.cnblogs.com/vamei/archive/2013/01/30/2879700.html

1. **Pandas绘图函数**

Pandas绘图函数是对matplotlib绘图函数的一个包装，使得绘图更方便。

1. **plot函数**

plot(x=None, y=None, kind='line', figsize=None, subplots=False, grid=None, legend=True, title=None)

* 参数
* x：x轴的坐标，默认为None，指的是用dataframe的index作为坐标
* y：y轴的坐标，默认为None，指的每个字段认为是一个y的坐标，如绘制所有曲线。
* kind：字符串，str

- 'line' : 默认是绘制折线图

- 'bar' : 绘制柱状图

- 'barh' : 横着的柱状图

- 'hist' : 直方图

- 'box' : 箱图

- 'kde' : 密度图(通过计算可能会产生观测数据的连续概率分布的估计而产生的)

- 'density' : same as 'kde'

- 'area' : 面积图

- 'pie' : 饼图

- 'scatter' : 散点图

- 'hexbin' : 绘制高密度散点图

* figsize：指绘图尺寸，默认为None
* subplots：是否绘制多个子图，默认为False
* grid：是否绘制网格线，默认为False
* legend：是否显示曲线的名称，默认为True
* title：绘图的名称，默认为None
* 返回的是axis对象

1. **scatter\_matrix函数**

pandas绘制直方图的函数不在DataFrame中，而是pandas库中独立的一个方法。

scatter\_matrix(frame, alpha=0.5, figsize=None, ax=None, grid=False, diagonal='hist', marker='.')

* 参数
* frame：是一个DataFrame对象
* alpha设置透明度，0到1的浮点数
* figsize：图片尺寸
* ax：可以设置绘图的对象，默认为None，则会新建一个axes对象。
* grid：判断是否设置网格线
* diagonal：设置对角线上图的类型，默认是'hist'，表示绘制直方图。
* marker：设置散点的类型，默认是‘.’，表示绘制实心点。
* 返回的是axis对象

1. **andrews\_curves函数**

pandas绘制调和曲线的函数不在DataFrame中，而是pandas.tools.plotting中。

andrews\_curves(frame, class\_column)

* 参数
* frame，DataFrame对象
* class\_column，字符串表示字段名，指示类标所在的列。
* 返回的是axis对象

## 案例一，股票数据勘探

1. **读取股票数据**

有四只美国股票价格从2003年到2011年的数据，数据的格式如下，第一行是字段名字，第一列是日期：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **AAPL** | **MSFT** | **XOM** | **SPX** |
| **2003-01-02** | 7.40 | 21.11 | 29.22 | 909.03 |

使用如下函数读入数据，返回的是一个DataFrame对象，parse\_dates为true会把时间解析成datetime时间对象，index\_col是指将第一列（索引为0的一列），即时间那一列作为索引，索引类型为DatetimeIndex。

|  |
| --- |
| close\_px\_all = pd.read\_csv('stock\_px.csv', parse\_dates=True, index\_col=0) |

使用head()函数查看前5行数据：

|  |
| --- |
| close\_px\_all.head() |

结果如下表所示：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **AAPL** | **MSFT** | **XOM** | **SPX** |
| **2003-01-02** | 7.40 | 21.11 | 29.22 | 909.03 |
| **2003-01-03** | 7.45 | 21.14 | 29.24 | 908.59 |
| **2003-01-06** | 7.45 | 21.52 | 29.96 | 929.01 |
| **2003-01-07** | 7.43 | 21.93 | 28.95 | 929.93 |
| **2003-01-08** | 7.28 | 21.31 | 28.83 | 909.93 |

1. **绘制折线图**

绘制股票价格变化折线图

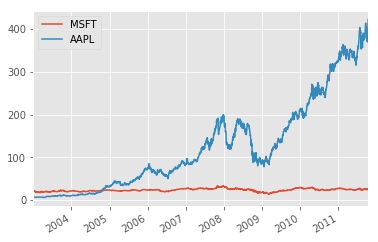
导入绘图的库

|  |
| --- |
| import matplotlib.pyplot as plt  import matplotlib  #使绘图内嵌到notebook中  %matplotlib inline |

使用DataFrame对象的plot函数绘制'MSFT'和'AAPL'股票的价格曲线

|  |
| --- |
| close\_px\_all.plot(y=['MSFT', 'AAPL'], kind='line', subplots=True) |

结果如下图所示：



1. **绘制柱状图**

绘制股票每年平均价格柱状图

从DatetimeIndex中可以取出每个时间的年份

|  |
| --- |
| # 在读取表格的时候，已经将时间的数据转换为  # DatetimeIndex对象了,并作为DataFrame的index，现在可以  # 取出每个时间的year属性值，用print语句查看DataFrame  # 的index的每个时间对象的年份属性。  print close\_px\_all.index.year  # 输出为：  # array([2003, 2003, 2003, ..., 2011, 2011, 2011]) |

使用dataFrame对象的groupby方法统计每只股票每年平均股票价格

|  |
| --- |
| # 根据close\_px\_all.index.year返回的年份作来进行分组，用  # mean函数求每组的股票的平均值  meanPrice = close\_px\_all.groupby(close\_px\_all.index.year).mean() |

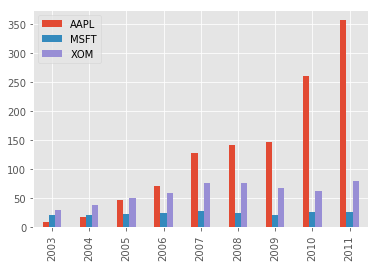
统计的平均价格如下表所示：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **AAPL** | **MSFT** | **XOM** | **SPX** |
| **2003** | 9.272619 | 20.59512 | 30.21111 | 965.2275 |
| **2004** | 17.76389 | 21.85044 | 38.87544 | 1130.649 |
| **2005** | 46.67595 | 23.07242 | 51.04548 | 1207.229 |
| **2006** | 70.81064 | 23.75936 | 58.45841 | 1310.462 |
| **2007** | 128.2739 | 27.90442 | 75.76713 | 1477.184 |
| **2008** | 141.979 | 24.76059 | 76.52597 | 1220.042 |
| **2009** | 146.8141 | 21.8854 | 67.12496 | 948.0464 |
| **2010** | 259.8425 | 26.26262 | 63.06798 | 1139.966 |
| **2011** | 356.5268 | 25.82593 | 79.04266 | 1276.093 |

绘制'AAPL', 'MSFT'和'XOM'股票每年平均价格的柱状图

|  |
| --- |
| meanPrice.plot(kind='bar', y=['AAPL', 'MSFT', 'XOM']) |

结果如下图所示：

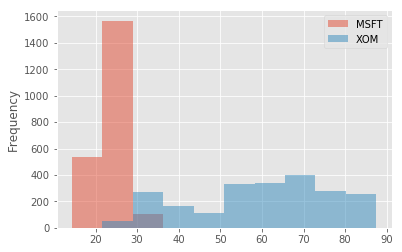


1. **绘制直方图**

绘制'MSFT'和'XOM'股票价格的直方图，可以了解到股票价格在各个价格区间的次数。

|  |
| --- |
| # 这里的alpha参数是用于设置透明度的  close\_px\_all.plot(y=['MSFT', 'XOM'], kind='hist', alpha=0.5) |

结果如下图所示：

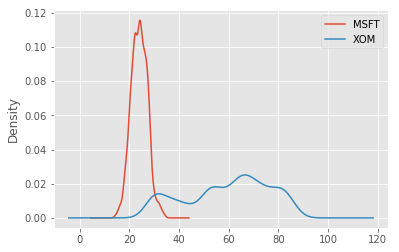


1. **绘制密度图**

例绘制'MSFT'和'XOM'股票价格的密度图，即价格的连续概率分布的估计，可以看出查看股票价格在各个价格区间的频率分布。

使用DataFrame的plot函数， 设置kind的值为’kde’。

close\_px\_all.plot(y=['MSFT', 'XOM'], kind='kde')



1. **绘制饼图**

例绘制四只股票在2010年，根据每个月平均值占12个月价格总和的比例绘制饼图。

筛选出2010年的数据

|  |
| --- |
| year2010 = close\_px\_all.ix[close\_px\_all.index.year==2010] |

求每个月平均值

|  |
| --- |
| year2010\_month\_mean =year2010.groupby(year2010.index.month).mean() |

每个月平均值占12个月价格总和的比例

|  |
| --- |
| year2010\_rate = year2010\_month\_mean / year2010\_month\_mean.sum() |

结果如下表所示：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **AAPL** | **MSFT** | **XOM** | **SPX** |
| **1** | 0.066837 | 0.091866 | 0.086061 | 0.082183 |
| **2** | 0.063967 | 0.0866 | 0.083143 | 0.079665 |
| **3** | 0.071887 | 0.089573 | 0.084752 | 0.084265 |
| **4** | 0.080812 | 0.093446 | 0.087069 | 0.087576 |
| **5** | 0.080913 | 0.086242 | 0.080659 | 0.082291 |
| **6** | 0.084029 | 0.078175 | 0.077782 | 0.079241 |
| **7** | 0.082035 | 0.077105 | 0.075542 | 0.078981 |
| **8** | 0.080868 | 0.076178 | 0.077807 | 0.079528 |
| **9** | 0.088117 | 0.076035 | 0.079026 | 0.082073 |
| **10** | 0.096832 | 0.077855 | 0.083937 | 0.085694 |
| **11** | 0.10026 | 0.081227 | 0.090334 | 0.087691 |
| **12** | 0.103445 | 0.085698 | 0.093887 | 0.09081 |

使用DataFrame的plot函数， 设置kind的值为’pie’。

新建4个axes画图对象

|  |
| --- |
| fig, ax = plt.subplots(2,2,figsize=(10,10)) |

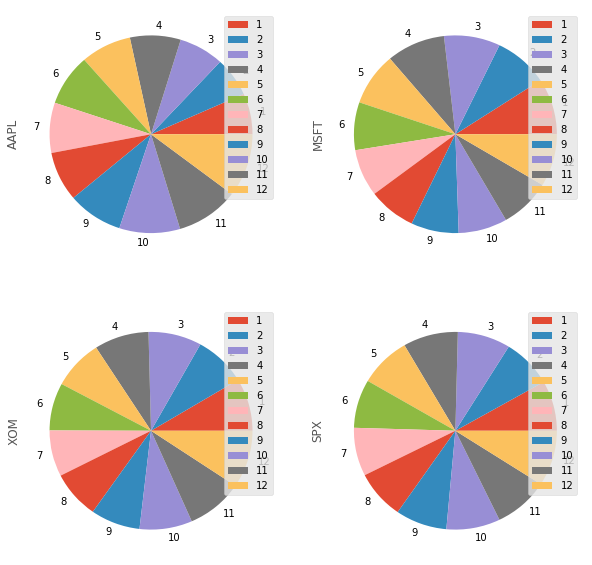
将画图数组改成列表形式

|  |
| --- |
| ax\_list = [ax[0,0], ax[0,1], ax[1,0], ax[1,1]] |

绘制饼图，参数ax设置绘图的axes对象，subplots为True设置绘制多个子图。

|  |
| --- |
| year2010\_rate.plot( kind='pie', ax=ax\_list, subplots=True) |

结果如下图所示：



## 5.案例二，鸢尾花数据勘探

### 1) 读取鸢尾花数据

鸢尾花数据的格式如下表所示，第一行是字段名字，花萼的长，花萼的宽，花瓣的长，花瓣的宽和花的类别，其中花的类别有'setosa', 'versicolor', 'virginica'。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| sepal length (cm) | sepal width (cm) | petal length (cm) | petal width (cm) | class |
| 5.1 | 3.5 | 1.4 | 0.2 | setosa |

使用如下函数读入数据，返回的是一个DataFrame对象。

|  |
| --- |
| iris = pd.read\_csv('iris.csv') |

使用head()函数查看前5行数据：

|  |
| --- |
| iris.head() |

前5行数据如下表所示

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | sepal length (cm) | sepal width (cm) | petal length (cm) | petal width (cm) | class |
| 0 | 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| 1 | 4.9 | 3 | 1.4 | 0.2 | setosa |
| 2 | 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| 3 | 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| 4 | 5 | 3.6 | 1.4 | 0.2 | setosa |

### 2) 绘制散点图

根据花瓣的长和宽绘制散点图

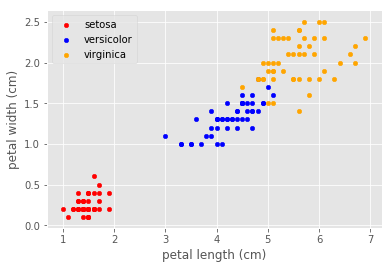
导入绘图的库

|  |
| --- |
| import matplotlib.pyplot as plt  import matplotlib  #使绘图内嵌到notebook中  %matplotlib inline |

使用DataFrame对象的plot函数根据花瓣的长和宽绘制散点图，参数x是设置用作x轴坐标的字段，参数y设置用作y轴坐标的字段，参数kind设置绘图的类型，参数c设置颜色，ax设置画板（默认为None），label设置散点的名称。

|  |
| --- |
| ax = iris[iris['class']=='setosa'].plot(x='petal length (cm)', y='petal width (cm)', kind='scatter', c='red', ax=None, label='setosa')  iris[iris['class']=='versicolor'].plot(x='petal length (cm)', y='petal width (cm)', kind='scatter', c='blue' , ax=ax, label='versicolor')  iris[iris['class']=='virginica'].plot(x='petal length (cm)', y='petal width (cm)', kind='scatter', c='orange', ax=ax, label='virginica') |

结果如下图所示，从下图来看，使用花瓣数据对花的区分度挺好的：



**3) 绘制散布图**

绘制散布图需要用到pandas的scatter\_matrix函数。

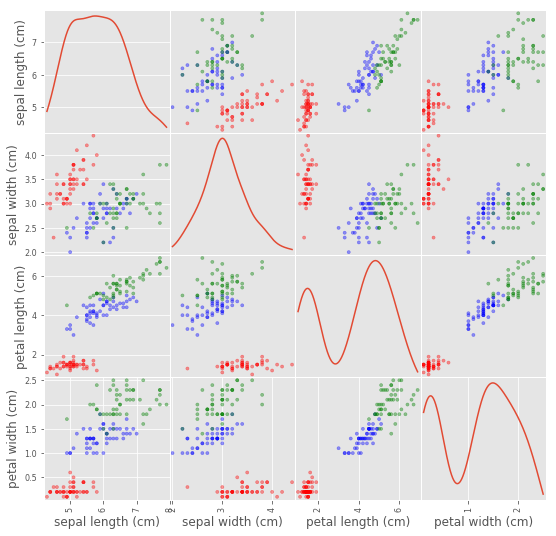
设置一个颜色字典，键是花的种类，值是颜色

|  |
| --- |
| color = {'setosa':'red', 'versicolor':'blue', 'virginica':'green'} |

iris.iloc[:,:-1]是指取出除了最后一列外的数据，figsize参数设置绘图大小，diagonal设置对角线上绘制的图的类型，lambda x: color[x]是一个匿名函数，根据参数返回字典中对应的值，iris['class'].apply(lambda x: color[x])返回各个类别对应的颜色。

|  |
| --- |
| pd.scatter\_matrix(iris.iloc[:,:-1], figsize=(9, 9), diagonal='kde', marker='o', s=40, alpha=0.4, c=iris['class'].apply(lambda x: color[x])) |

使用该函数绘制的散布图如下所示：



**4) 绘制调和曲线图**

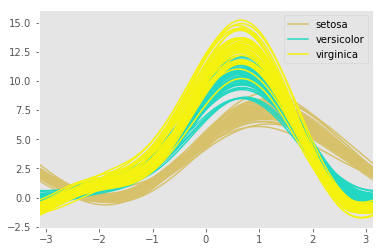
导入函数

|  |
| --- |
| from pandas.tools.plotting import andrews\_curves |

使用andrews\_curves绘制调和曲线，第一个参数是DataFrame对象，第二个参数是DataFrame对象中存放类别的列的名字。

|  |
| --- |
| andrews\_curves(iris, 'class') |

结果如下图所示：



## 实践任务

从商店客流量数据可视化和皮马印第安人糖尿病数据可视化中选择一个来完成。

1. **商店客流量数据可视化**
2. **数据来源**

商店数据来自天池口碑商家客流量预测比赛，这里只筛选了一部分数据。

<https://tianchi.aliyun.com/competition/introduction.htm?spm=5176.100066.333.10.W7qorD&raceId=231591>

“shop\_payNum.csv”的数据各个字段的含义

|  |  |  |
| --- | --- | --- |
| **Field** | **Sample** | **Description** |
| pay\_num | 1 | 客流量（每天使用支付宝在该店消费的人数） |
| shop\_id | 1 | 商家id，与shop\_info对应 |
| time\_stamp | 2016/1/1 | 支付时间 |
| cate\_2\_name | 快餐 | 商店的二级分类名称 |

1. **实验要求：**

参考案例一从以下任务中任选3个绘制不同图形的任务：

（1）绘制所有便利店的10月的客流量折线图。

（2）绘制每类商家10月份的日平均客流量折线图。

（3）选择一个商家，统计每月的总客流量，绘制柱状图。

（4）选择一个商家，统计某个月中，周一到周日的每天平均客流量，并绘制柱状图。

（5）选择一个商家，绘制客流量直方图。

（6）选择一个商家，绘制客流量密度图。

（7）统计某个月各个类别商店总客流量占该月总客流量的比例，绘制饼图。

**2）皮马印第安人糖尿病数据可视化**

**a) 数据来源：** <http://archive.ics.uci.edu/ml/datasets/Pima+Indians+Diabetes>

“pima.csv”数据前9个字段的含义：

(1) Number of times pregnant

(2) Plasma glucose concentration a 2 hours in an oral glucose tolerance test

(3) Diastolic blood pressure (mm Hg)

(4) Triceps skin fold thickness (mm)

(5) 2-Hour serum insulin (mu U/ml)

(6) Body mass index (weight in kg/(height in m)^2)

(7) Diabetes pedigree function

(8) Age (years)

(9) Class variable (0 or 1)

**b) 实验要求：**

参考案例二完成一下任务：

（1）任选两个字段绘制散点图。

（2）使用全部或者部分特征绘制散布图。

（3）绘制调和曲线图。