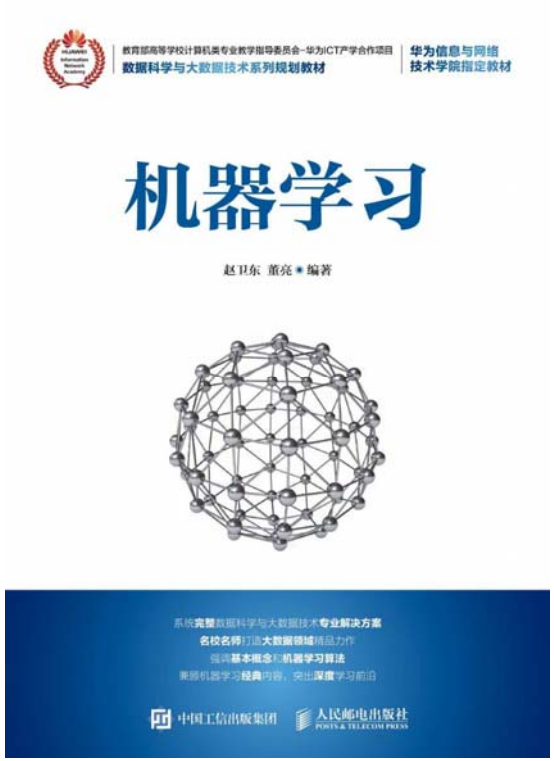


机器学习
可视化分析

复旦大学 **赵卫东** 博士

wdzhao@fudan.edu.cn

章节结构

- 可视化分析
  - 可视化分析的作用
  - 可视化分析方法
  - 可视化分析常用工具
  - 常见的可视化图表
  - 可视化分析面临的挑战

可视化分析

- 可视化分析是一种数据分析方法，利用人类的形象思维将数据关联，并映射为形象的图表。人脑对于视觉信息的处理要比文本信息容易得多，所以可视化图表能够使用户更好地理解信息，可视化分析凭借其直观清晰，能够提供新洞察和发现机会的特点活跃在诸多科学领域

可视化分析的作用

- 在数据分析中，通过绘制图表更容易找到数据中的模式。传统的数据分析方法存在一些局限性，需要借助于分析师丰富的分析经验。可视化分析方法将数据以图像的方式展现，提供友好的交互，还可以提供额外的记忆帮助，对于将要分析的问题，无需事先假设或猜想，可以自动从数据中挖掘出更多的隐含信息
- 在机器学习领域，缺失数据、过度训练、过度调优等都会影响模型的建立，可视化分析可以帮助解决其中一些问题
- 可视化分析在机器学习的数据预处理、模型选择、参数调优等阶段也同样发挥重要作用。在数据建模的过程中，容易辨别出数据的分布、异常、参数取值对模型性能的影响等

可视化分析的作用

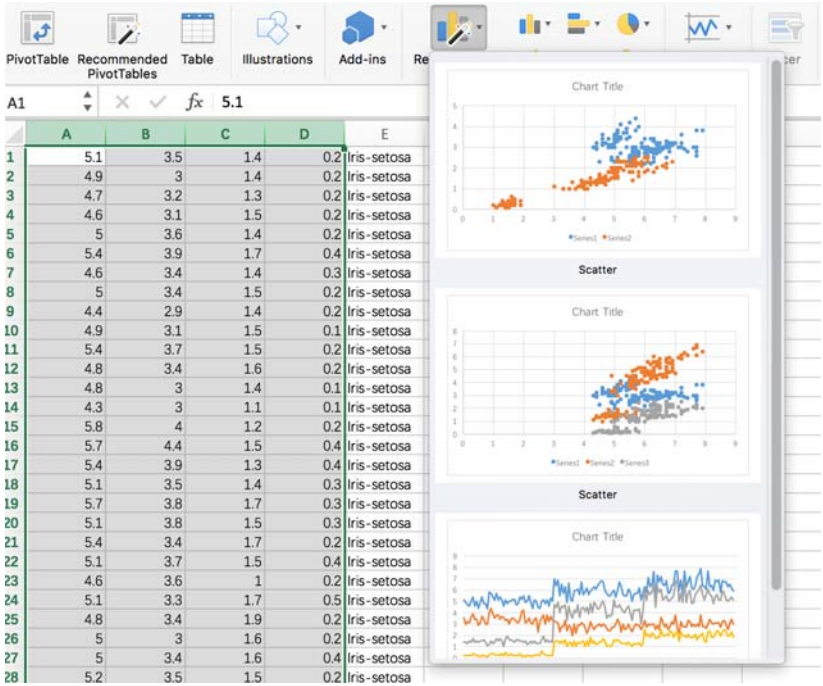
- 在分析结果展示时，通过建立可视化仪表板，组合多幅可视化图表，从不同的角度来描述信息，全方位展示分析结论
- 除了辅助数据分析之外，可视化分析为看似冰冷的数据带来更多趣味性，直观清晰的表达拥有更多的受众。在信息传播领域，可视化结果的独特风格（颜色、线条、轴线、尺寸等）不仅将有用的信息展示出来，更像是种精美的艺术品，让数据展示也变得更加富有情感

可视化分析方法

- 为了获得易于理解的可视化结果，人机交互很重要。可视化分析的常用方法大致可以划分为三个层次：领域方法、基础方法以及方法论基础
- 领城方法领域方法是根据数据的来源领域以及数据的性质进行可视化，包括地理信息可视化、空间数据可视化、文本数据可视化、跨媒体数据可视化、实时数据可视化等
- 可视化基础方法基础方法包括统计图表、视觉隐喻。常见的统计图表有柱状图、折线图、饼图、箱图、散点图、韦恩图、气泡图、雷达图、热地图、等值线等，不同的统计图表有各自的适用场合
- 可视化分析的方法论基础是视觉编码，视觉编码是指受众对于接收到的视觉刺激进行编码，所以视觉编码的关键在于使用符合目标用户人群视觉感知习惯的表达方法，鉴于视觉感知习惯往往与一个人的知识、经验、心理等多种特异性的因素相关，而且视觉感知是一种视觉信息直接映射与信息提取、转换、存储、处理、理解等后续活动结合而成的过程

可视化分析常用工具

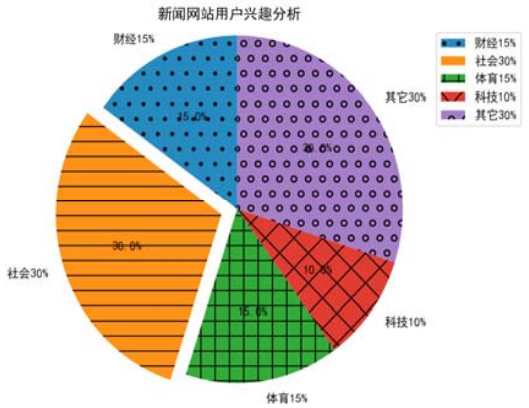
- Excel
- Tableau
- Raw
- Chart.js
- Processing
- Wordle
- Orange
- Facets
- Python、R语言库：
  - matplotlib、Seaborn、Pyecharts、ggplosts



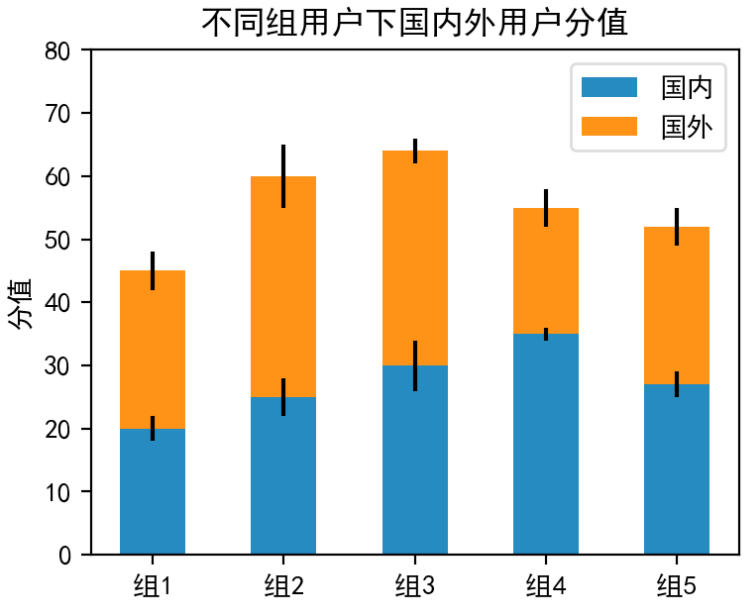
常见可视化图表

```
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] #用来正常显示中文
plt.rcParams['axes.unicode_minus']=False #用来正常显示负号

labels = '财经15%', '社会30%', '体育15%','科技10%', '其它30%‘  #初始化参数autopct为显示的百分比样式
sizes = [15, 30, 15, 10, 30]
explode = (0, 0.1, 0, 0,0)#突出第2项
fig1, ax1 = plt.subplots()
pie = ax1.pie(sizes, explode=explode, labels=labels, autopct='%1.1f%%',shadow=False,
startangle=90)
patches = pie[0]   #设置分块的填充模式
patches[0].set_hatch('.')
patches[1].set_hatch('-')
patches[2].set_hatch('+')
patches[3].set_hatch('x')
patches[4].set_hatch('o')
plt.legend(patches, labels)
ax1.axis('equal')
plt.title('新闻网站用户兴趣分析')
plt.show()
```



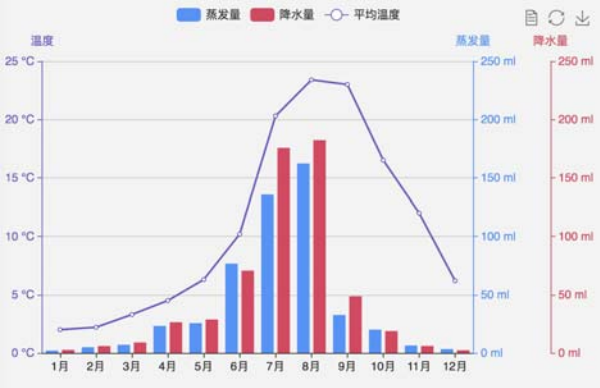
常见可视化图表

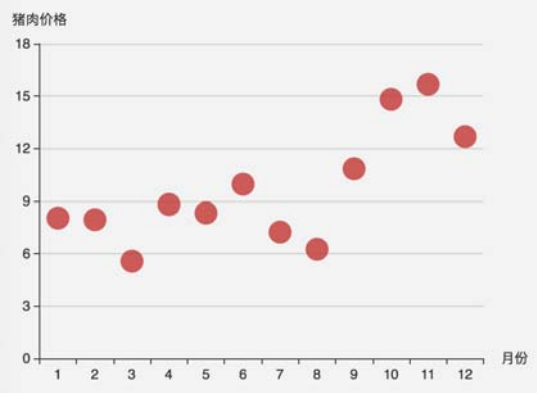



```
import numpy as np
N=5
inMeans=(20,25,30,35,27)
outMeans=(25,35,34,20,25)
inStd = (2,3,4,1,2)
outStd=(3,5,2,3,3)
ind=np.arange(N)    #Bar坐标位置
width=0.5      #Bar的宽度
#使用plt.bar()方法生成两个国人和国外两组柱子
p1=plt.bar(ind, inMeans, width, yerr=inStd)
P2=plt.bar(ind,outMeans,width,bottom=inMeans,yerr=outStd)
#查看不同组用户的总分值基础上，查看组内不同类别的用户分值占比情况
plt.ylabel('分值')
plt.title('不同组用户下国内外用户分值')
plt.xticks(ind, ('组1', '组2', '组3', '组4', '组5'))
plt.yticks(np.arange(0, 81, 10))
plt.legend((p1[0], p2[0]), ('国内', '国外'))
plt.show()
```

常见可视化图表

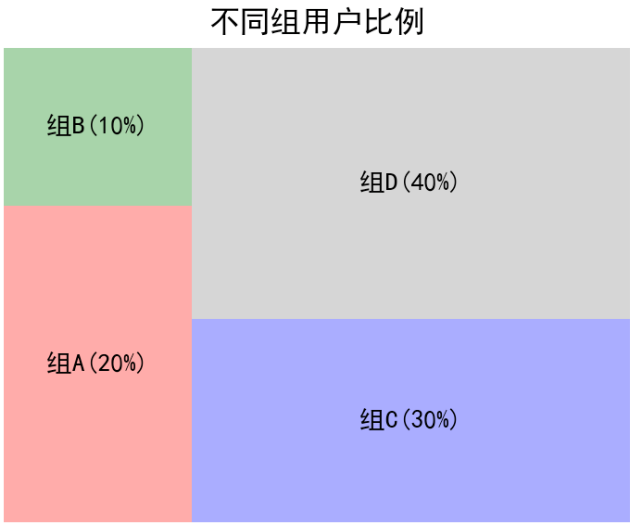
- 时间序列可视化

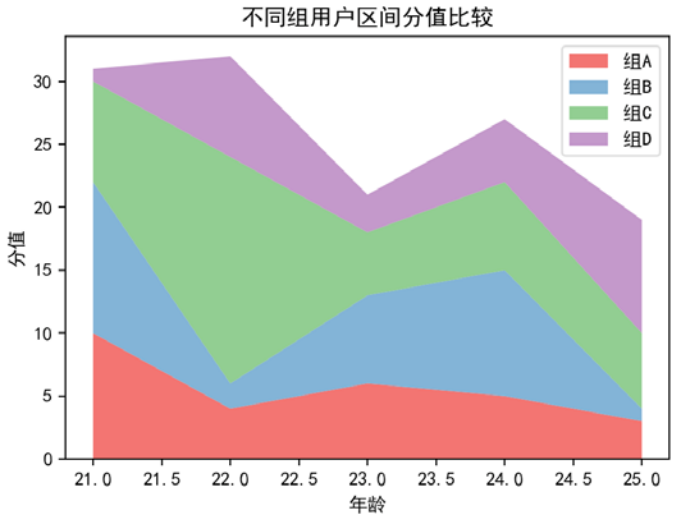




常见可视化图表

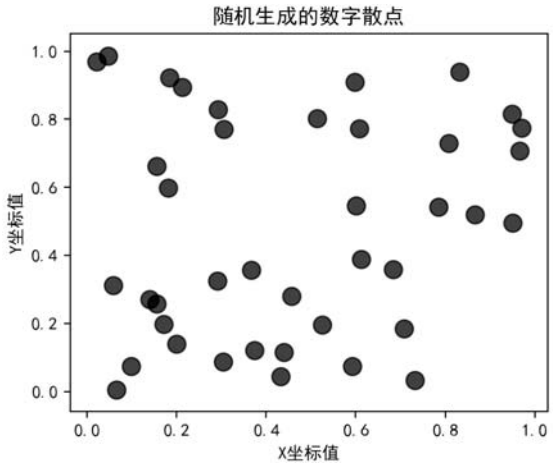
- 比例的可视化

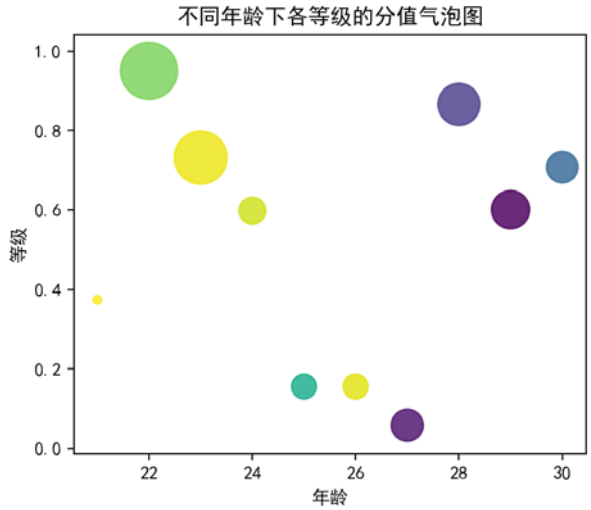




常见可视化图表

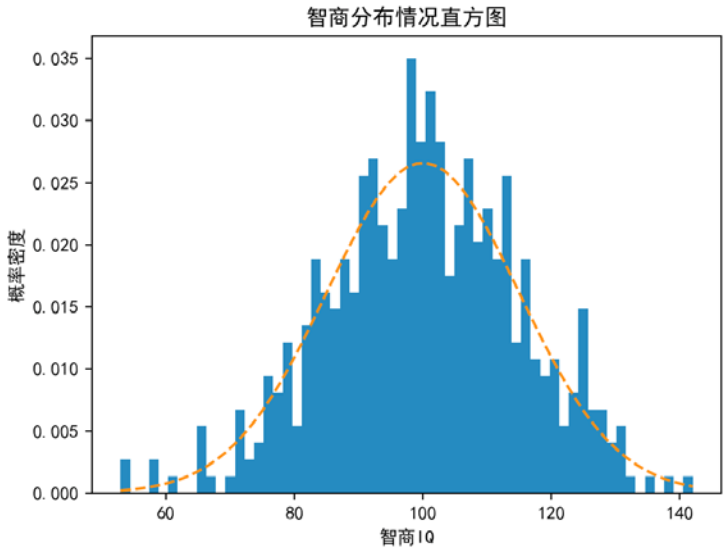
- 关系可视化

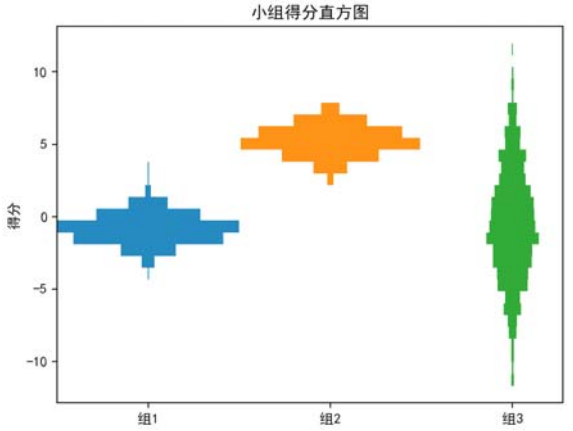




常见可视化图表

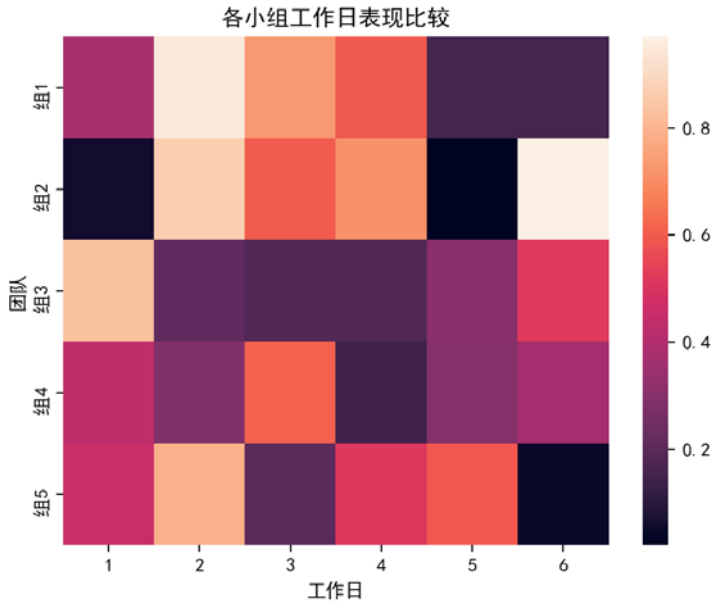
- 关系可视化

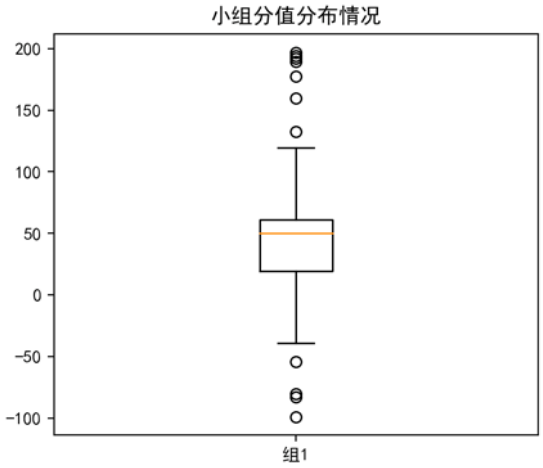




常见可视化图表

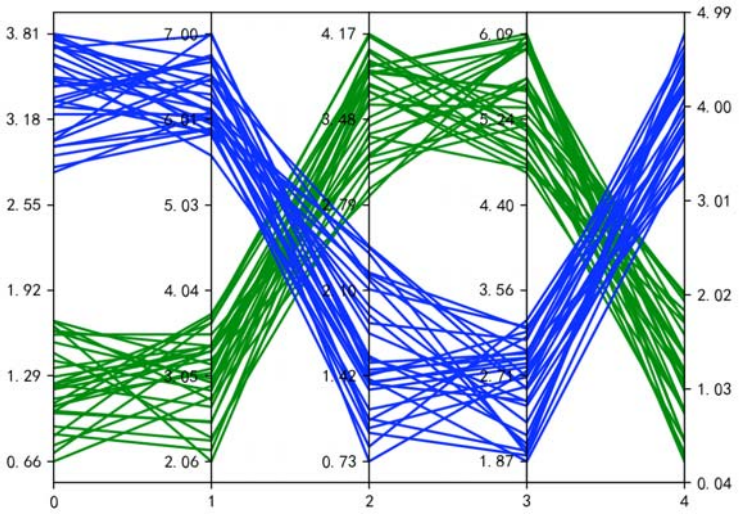
- 差异可视化

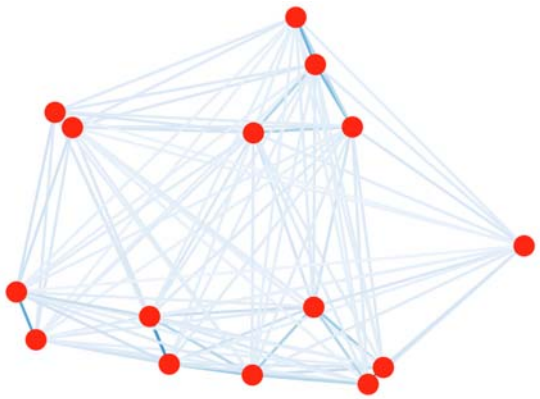




常见可视化图表

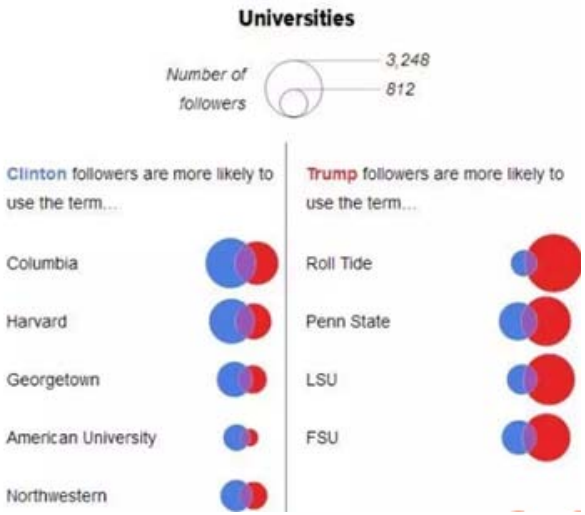
- 差异可视化





常见可视化图表

- 空间关系可视化



income

	clinton	trump	other/no answer
under $30,000 **17%**	53%	41%	6%
$30k-$49,999 **19%**	51%	42%	7%
$50k-$99,999 **31%**	46%	50%	4%
$100k-$199,999 **24%**	47%	48%	5%
$200k-$249,999 **4%**	48%	49%	3%
$250,000 or more **6%**	46%	48%	6%

24537 respondents

可视化分析面临的挑战

- 进行可视化分析时挑战主要来自于两个方面：数据和可视化结果
- 数据层面的挑战包括数据的来源不唯、数据质量良莠不齐、数据整合困难等挑战。信息时代数据更新飞快、体量大，对可视化分析速度要求越来越高。分析过程涉及领域广而繁杂，对于数据的专业解读带来挑战
- 在可视化结果层面，数据集中样本的相关性导致视觉噪声的大量出现，面临降噪的挑战。 受限于设备的长宽比、分辨率、现实世界的感受等，可视化图表中大型图像的感知的挑战；受限于可视化的算法以及硬件的性能，及时响应，高速图像变换的挑战；专业领域不同带来的可视化需求不同，最大限度地满足受众视觉喜好的挑战
- 此外还有可视化分析流程的优化，可视化分析工具的可操作性等等。

主成分分析

- 主成分分析是最常用的线性降维方法，它的目标是通过某种线性投影，将高维的数据映射到低维的空间中，并期望在所投影的维度上数据的方差最大，以此使用较少的维度，同时保留较多原数据的维度
- 尽可能如果把所有的点都映射到一起，那么几乎所有的区分信息都丢失了，而如果映射后方差尽可能的大，那么数据点则会分散开来，特征更加明显。PCA是丢失原始数据信息最少的一种线性降维方法，最接近原始数据
- PCA算法目标是求出样本数据的协方差矩阵的特征值和特征向量，而协方差矩阵的特征向量的方向就是PCA需要投影的方向。使样本数据向低维投影后，能尽可能表征原始的数据。协方差矩阵可以用散布矩阵代替，协方差矩阵乘以（n-1）就是散布矩阵，n为样本的数量。协方差矩阵和散布矩阵都是对称矩阵，主对角线是各个随机变量（各个维度）的方差

主成分分析

- 设有m条n维数据，PCA的一般步骤如下
  - 将原始数据按列组成n行m列矩阵X
  - 计算矩阵X中每个特征属性（n维）的平均向量M（平均值）
  - 将X的每行（代表一个属性字段）进行零均值化，即减去M
  - 按照公式$C = \frac{1}{m} XX^T$求出协方差矩阵
  - 求出协方差矩阵的特征值及对应的特征向量
  - 将特征向量按对应特征值从大到小按行排列成矩阵，取前k（k < n）行组成基向量P
  - 通过$Y = PX$计算降维到k维后的样本特征

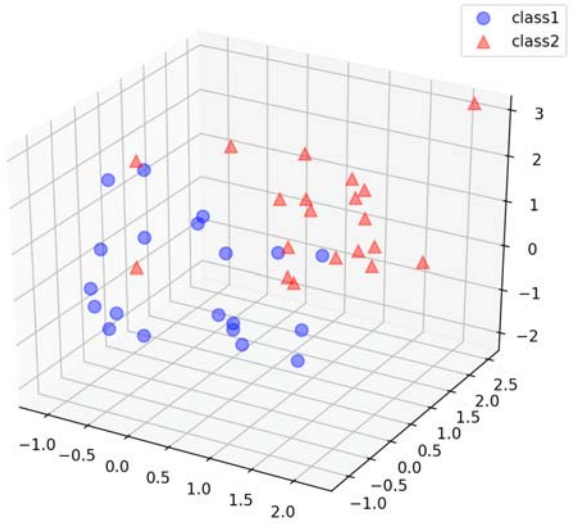
主成分分析

- 基于sklearn（Python语言下的机器学习库）和numpy随机生成2个类别共40个3维空间的样本点，生成的代码如下：

```
mu_vec1 = np.array([0,0,0])
cov_mat1 = np.array([[1,0,0],[0,1,0],[0,0,1]])
class1_sample = np.random.multivariate_normal(mu_vec1, cov_mat1, 20).T
mu_vec2 = np.array([1,1,1])
cov_mat2 = np.array([[1,0,0],[0,1,0],[0,0,1]])
class2_sample = np.random.multivariate_normal(mu_vec2, cov_mat2, 20).T
```

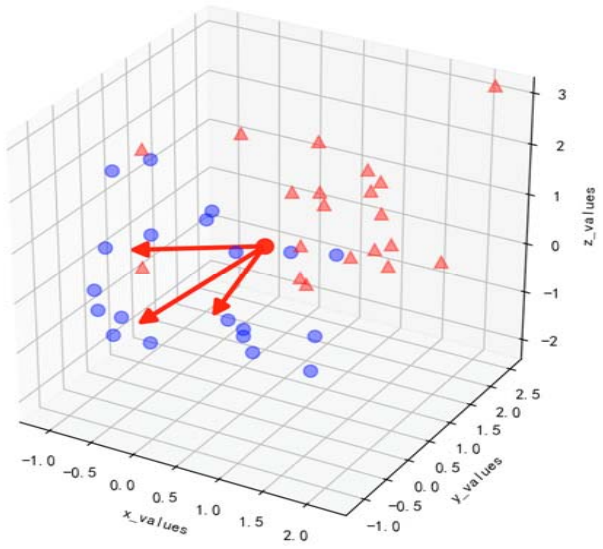
主成分分析

- 生成的两个类别class1_sample和class2_sample的样本数据维度为3维，即样本数据的特征数量为3个，将其置于3维空间中展示



主成分分析

- 计算40个点在3个维度上的平均向量



主成分分析

- 二维空间分布