层次聚类

1. 实验目的

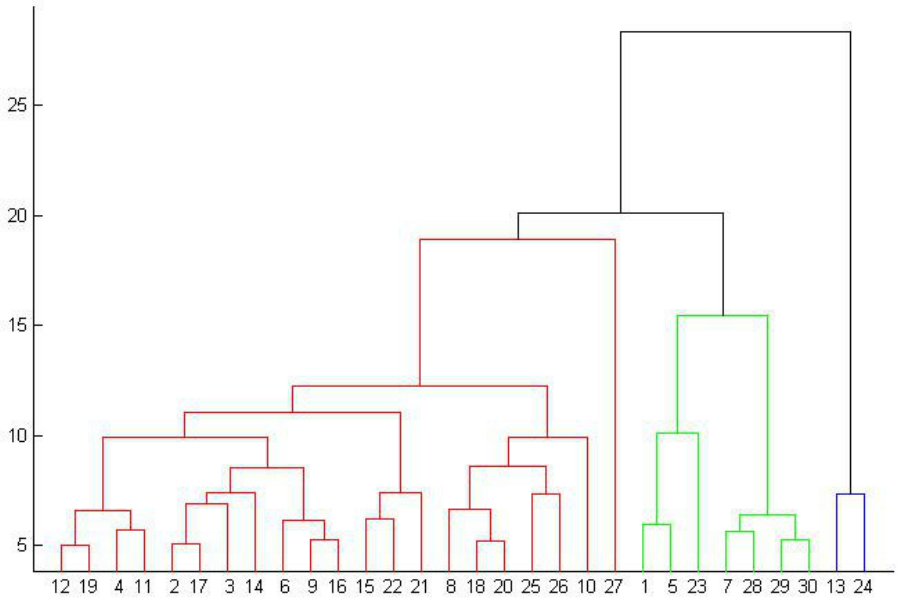
了解层次聚类算法的原理，并且可以简单应用

2. 算法原理

1）PCA 用来做什么

层次聚类，是一种很直观的算法。顾名思义就是要一层一层地进行聚类，可以从下而上地把小的 cluster 合并聚集，也可以从上而下地将大的 cluster 进行分割。似乎一般用得比较多的是从下而上地聚集，因此这里我就只介绍这一种。

所谓从下而上地合并 cluster，具体而言，就是每次找到距离最短的两个 cluster，然后进行合并成一个大的 cluster，直到全部合并为一个 cluster。整个过程就是建立一个树结构，类似于下图。



那么，如何判断两个 cluster 之间的距离呢？一开始每个数据点独自作为一个类，它们的距离就是这两个点之间的距离。而对于包含不止一个数据点的 cluster，就可以选择多种方法了。最常用的，就是 average-linkage，即计算两个 cluster 各自数据点的两两距离的平均值。类似的 还有 single-linkage/complete-linkage，选择两个 cluster 中距离最短/最长的一对数据点的距离作为类的距离。个人经 验 complete-linkage 基本没用，single-linkage 通过关注局域连接，可以得到一些形状奇特的 cluster，但是因为太过极 端，所以效果也不是太好。

层 次聚类较大的**优点**，就是它一次性地得到了整个聚类的过程，只要得到了上面那样的聚类树，想要分多少个 cluster 都可以直接根据树结构来得到结果，改变 cluster 数目不需要再次计算数据点的归属。层次聚类的缺点是计算量比较大，因为要每次都要计算多个 cluster 内所有数据点的两两距离。另外，

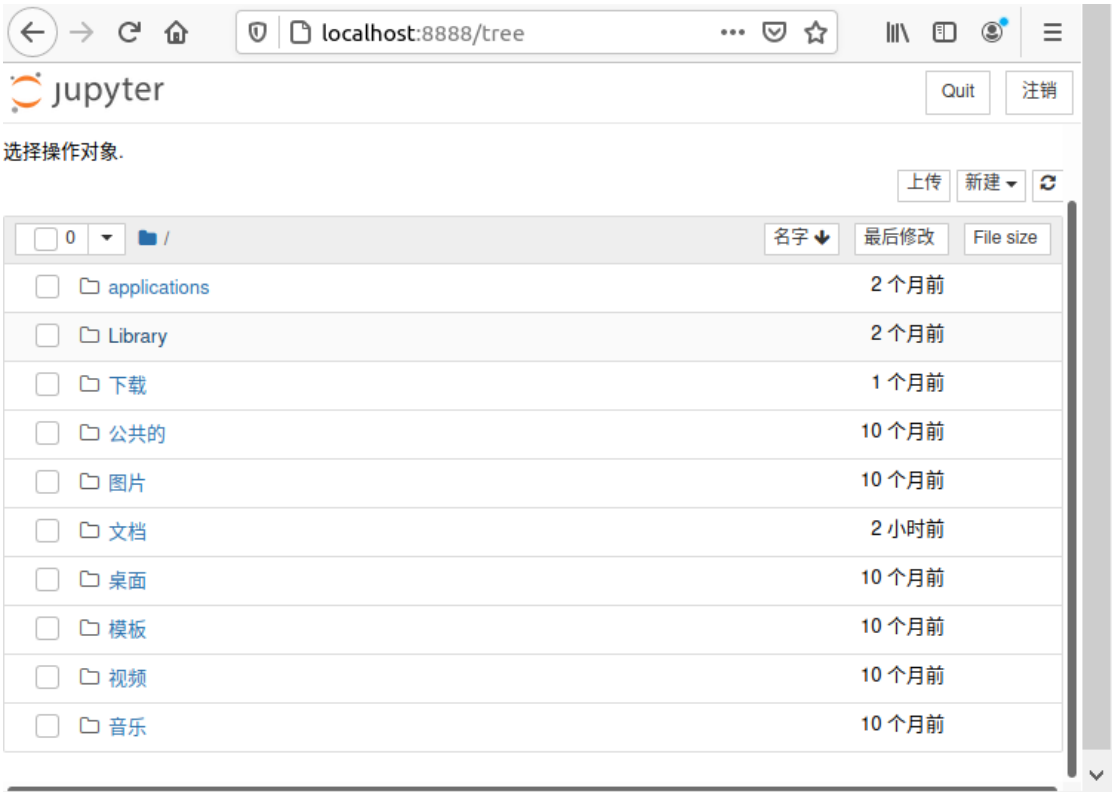
由 于层次聚类使用的是贪心算法，得到的显然只是局域最优，不一定就是全局最优，这可以通过加入随机效应解决，这就是另外的问题了。

3. 实验环境

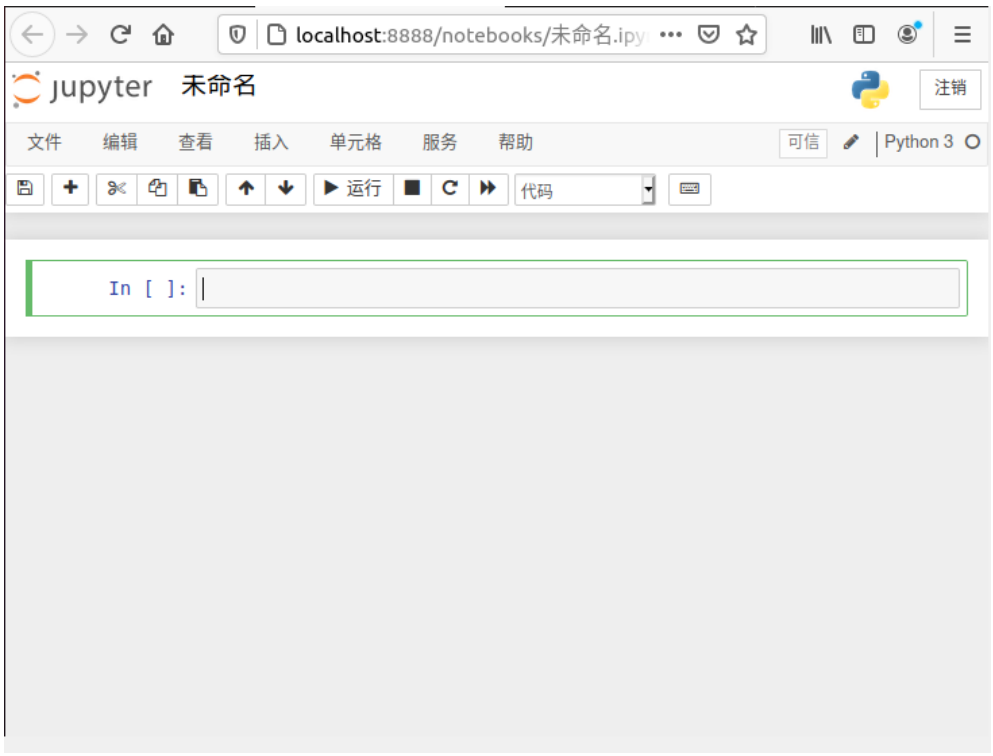
Ubuntu 20.04
Python 3.6
Jupyter notebook

4. 实验步骤

1)打开终端，然后输入 jupyter notebook，出现如下界面



2)选定特定文件夹，新建 ipynb 文件，在未命名出可重命名文件



5. 实操

Step 1:数据预处理

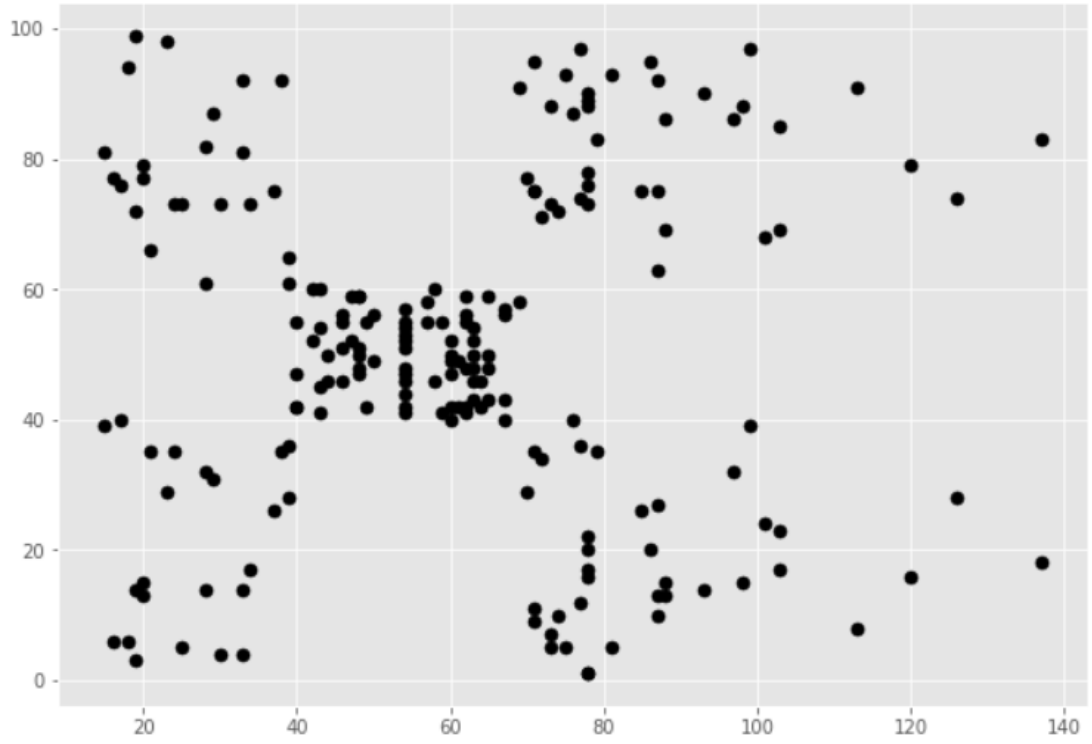
1. 导入库
2. 导入数据集
3. 数据集信息
4. 数据集绘制

```
#导入库
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

#导入数据集
# read the dataset
dataset = pd.read_csv('Mall_Customers.csv')
dataset.head()


# 数据集信息
dataset.info()
X = dataset.iloc[:, [3, 4]].values
X.shape


# 数据集绘制
plt.scatter(X[:, 0], X[:, 1], s = 50, c = 'black')
plt.show()
```

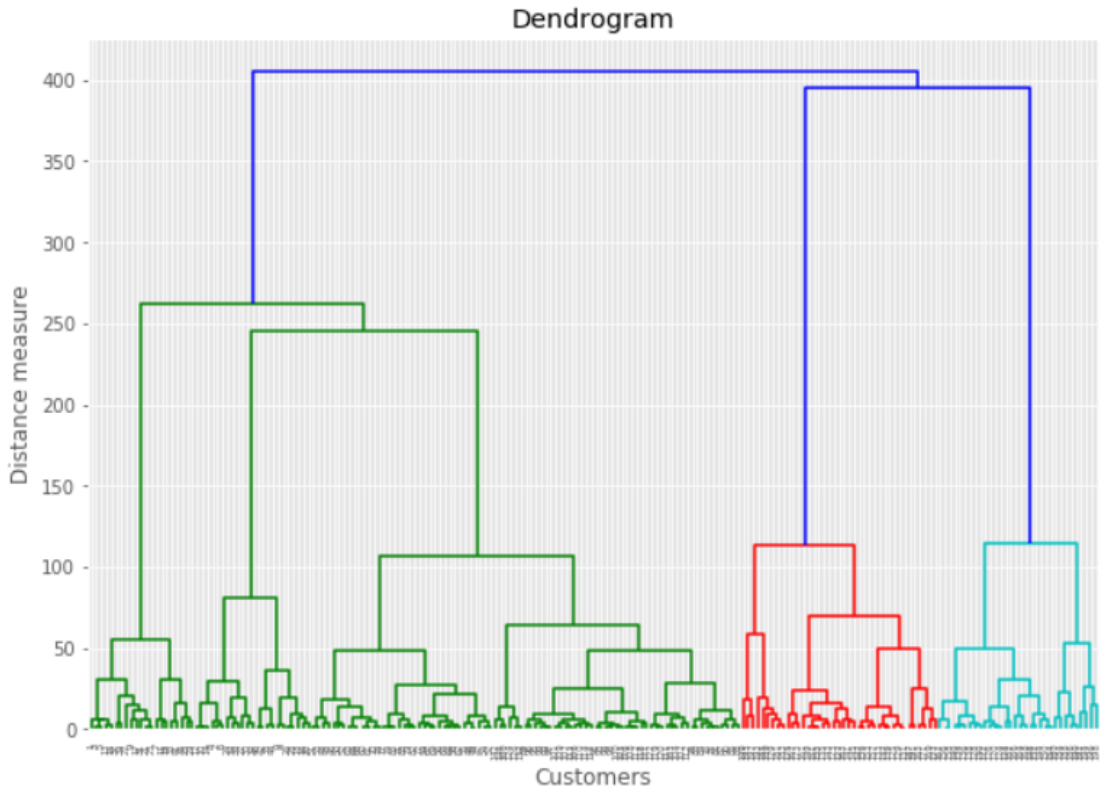


Step 2:层次聚类模型模型

```
# Using the dendrogram to find the optimal number of clusters
import scipy.cluster.hierarchy as sch

dendrogram = sch.dendrogram(sch.linkage(X, method = 'ward'))

plt.title('Dendrogram')
plt.xlabel('Customers')
plt.ylabel('Distance measure')
plt.show()
```



Step 3:聚类操作与结果

```
# Fitting Hierarchical Clustering to the dataset
from sklearn.cluster import AgglomerativeClustering
hc = AgglomerativeClustering(n_clusters = 5, affinity = 'euclidean', li
nkage = 'ward')
y_hc = hc.fit_predict(X)

# Visualising the clusters
plt.scatter(X[y_hc == 0, 0], X[y_hc == 0, 1], s = 100, c = 'red', label
 = 'Cluster 1')
plt.scatter(X[y_hc == 1, 0], X[y_hc == 1, 1], s = 100, c = 'blue', labe
l = 'Cluster 2')
plt.scatter(X[y_hc == 2, 0], X[y_hc == 2, 1], s = 100, c = 'green', lab
el = 'Cluster 3')
plt.scatter(X[y_hc == 3, 0], X[y_hc == 3, 1], s = 100, c = 'cyan', labe
l = 'Cluster 4')
plt.scatter(X[y_hc == 4, 0], X[y_hc == 4, 1], s = 100, c = 'magenta', l
abel = 'Cluster 5')
plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```


Clusters of customers
Spending Score (1-100)
Annual Income (k$)
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster 5
0
20
40
60
80
100
120
140