# 数据挖掘-实验2

个人感悟分享

计科210X 甘晴void

2023.10.23

# 再读题目

我对于数据降维的理解：

数据降维是指将高维数据映射到低维空间的过程。

举例：
描述一个人的特征：身高、体重、兴趣、健康状况、性格、成绩……

我现在想用这些特征看一群人的分布

把一个特征想象成一条坐标轴，我们一般能看到的是二维坐标系（x,y）或者三维坐标系（x,y,z），一般来说，我们无法处同时看透这么多特征

降维：在尽可能保留这些特征所附带信息的情况下，把特征的数目减少，以至于甚至用二维坐标轴或者三维坐标轴就能看到这群人的特征。

题目要求：

数据集大小：(13627, 65)

样本

特征

| | Unnamed: 0 | MF: KIRC | MF: BRCA | MF: READ | MF: PRAD | MF: STAD | MF: HNSC | MF: LUAD | MF: THCA | MF: BLCA | ... | CNA: LUAD | CNA: THCA | CNA: BLCA | CNA: ESCA | CNA: LIHC | CNA: UCEC |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | STIM1 | 0.000000 | 0.005282 | 0.000000 | 0.000000 | 0.012993 | 0.000000 | 0.060833 | 0.003601 | 0.000000 | ... | 0.000000 | 0.000000 | 0.000000 | 0.0 | 0.0 | 0.0 |
| 1 | TRPC1 | 0.000000 | 0.005302 | 0.000000 | 0.022758 | 0.000000 | 0.021187 | 0.060202 | 0.000000 | 0.000000 | ... | 0.000000 | 0.000000 | 0.000000 | 0.0 | 0.0 | 0.0 |
| 2 | NOS1 | 0.063345 | 0.015765 | 0.055677 | 0.022690 | 0.048721 | 0.021051 | 0.038974 | 0.000000 | 0.088398 | ... | 0.352785 | 0.000000 | 0.000000 | 0.0 | 0.0 | 0.0 |
| 3 | ATP2B4 | 0.038215 | 0.014866 | 0.053870 | 0.045420 | 0.025206 | 0.016322 | 0.040063 | 0.007193 | 0.057544 | ... | 0.000000 | 0.322222 | 0.272727 | 0.0 | 0.0 | 0.0 |
| 4 | ABCC9 | 0.012769 | 0.031122 | 0.055910 | 0.045414 | 0.187905 | 0.048539 | 0.058909 | 0.010787 | 0.090623 | ... | 0.000000 | 0.000000 | 0.000000 | 0.0 | 0.0 | 0.0 |

## 区分样本与特征

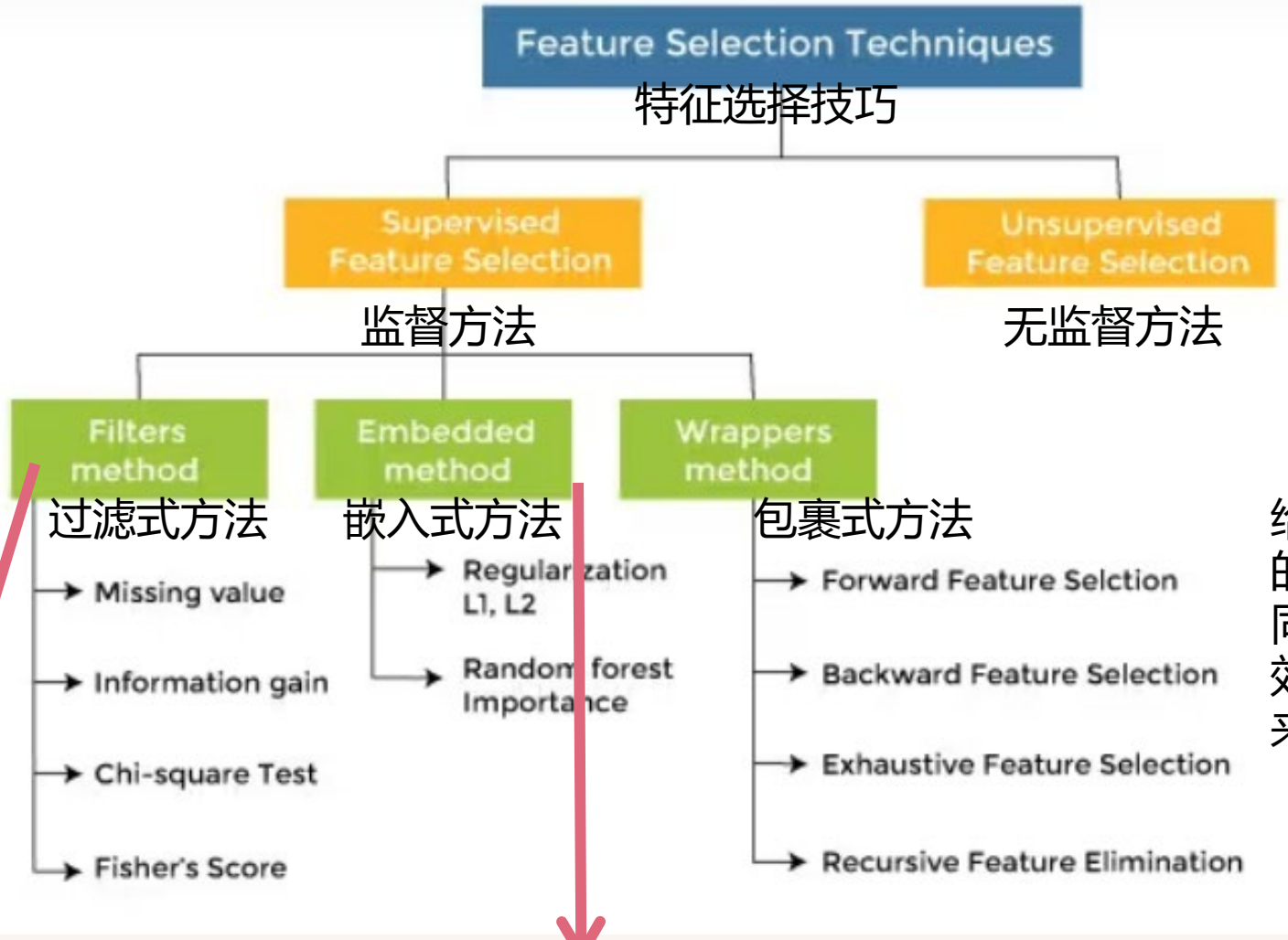
但是一般来说，“行”充当“样本”，
“列”充当“特征“，也就是这样
所以说这里可能涉及到”转置“

| | 0 | 1 | 2 |
|---|---|---|---|
| MF:KIRC | | | |
| MF:BRCA | | | |
| MF:READ | | | |

数据列的所有列名如下

'MF: KIRC', 'MF: BRCA', 'MF: READ', 'MF: PRAD', 'MF: STAD', 'MF: HNSC',
'MF: LUAD', 'MF: THCA', 'MF: BLCA', 'MF: ESCA', 'MF: LIHC', 'MF: UCEC',
'MF: COAD', 'MF: LUSC', 'MF: CESC', 'MF: KIRP', 'METH: KIRC',
'METH: BRCA', 'METH: READ', 'METH: PRAD', 'METH: STAD', 'METH: HNSC',
'METH: LUAD', 'METH: THCA', 'METH: BLCA', 'METH: ESCA', 'METH: LIHC',
'METH: UCEC', 'METH: COAD', 'METH: LUSC', 'METH: CESC', 'METH: KIRP',
'GE: KIRC', 'GE: BRCA', 'GE: READ', 'GE: PRAD', 'GE: STAD', 'GE: HNSC',
'GE: LUAD', 'GE: THCA', 'GE: BLCA', 'GE: ESCA', 'GE: LIHC', 'GE: UCEC',
'GE: COAD', 'GE: LUSC', 'GE: CESC', 'GE: KIRP', 'CNA: KIRC',
'CNA: BRCA', 'CNA: READ', 'CNA: PRAD', 'CNA: STAD', 'CNA: HNSC',
'CNA: LUAD', 'CNA: THCA', 'CNA: BLCA', 'CNA: ESCA', 'CNA: LIHC',
'CNA: UCEC', 'CNA: COAD', 'CNA: LUSC', 'CNA: CESC', 'CNA: KIRP'

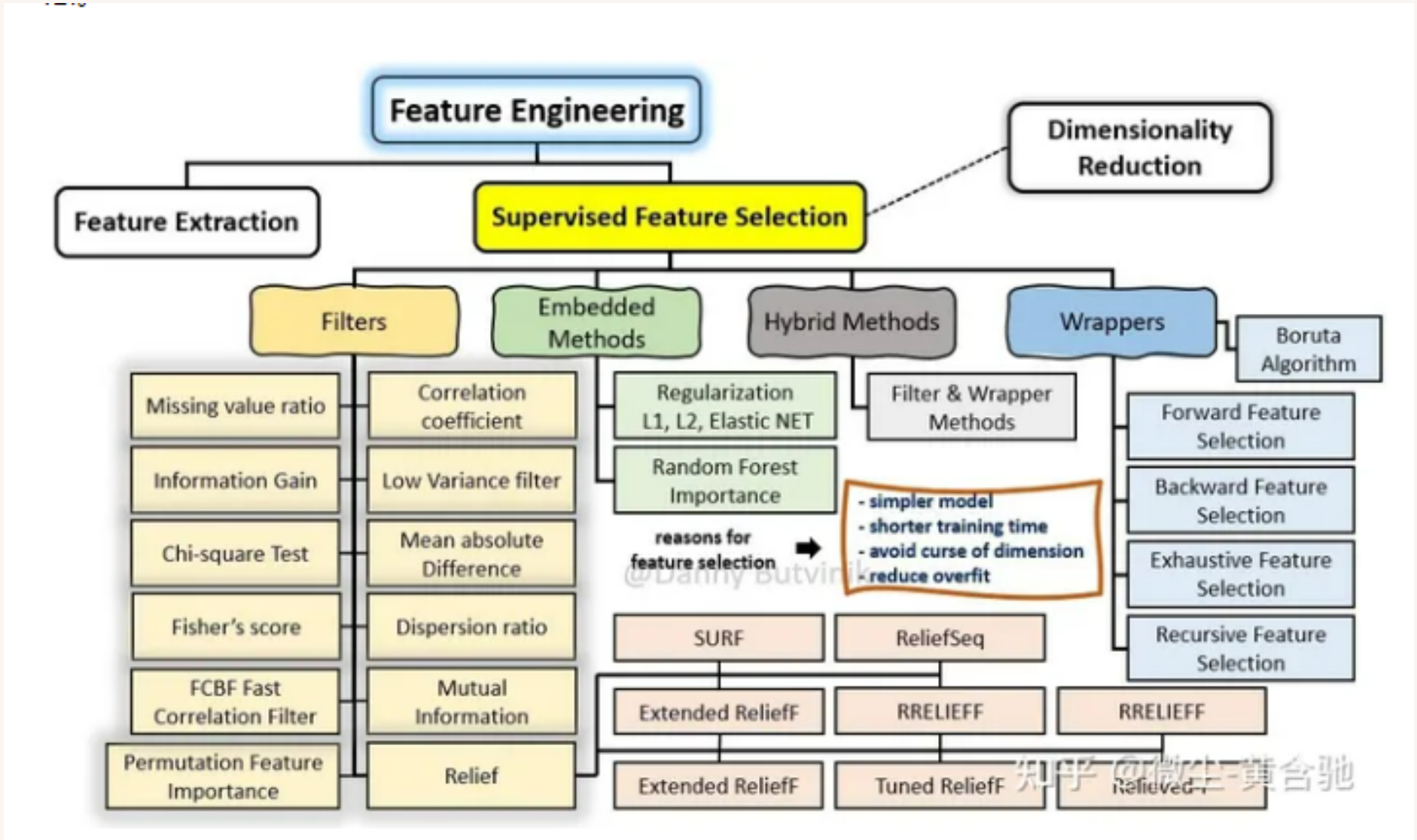
观察数据：
可以发现 MF METH GE CNA 各自领导16个，而后面各自领导4个，由4*16=64
这个在后面分组的时候可以给我们启发




给定了某种模型，及预测效果评价的方法，然后针对特征空间中的不同子集，计算每个子集的预测效果，效果最好的，即作为最终被挑选出来的特征子集

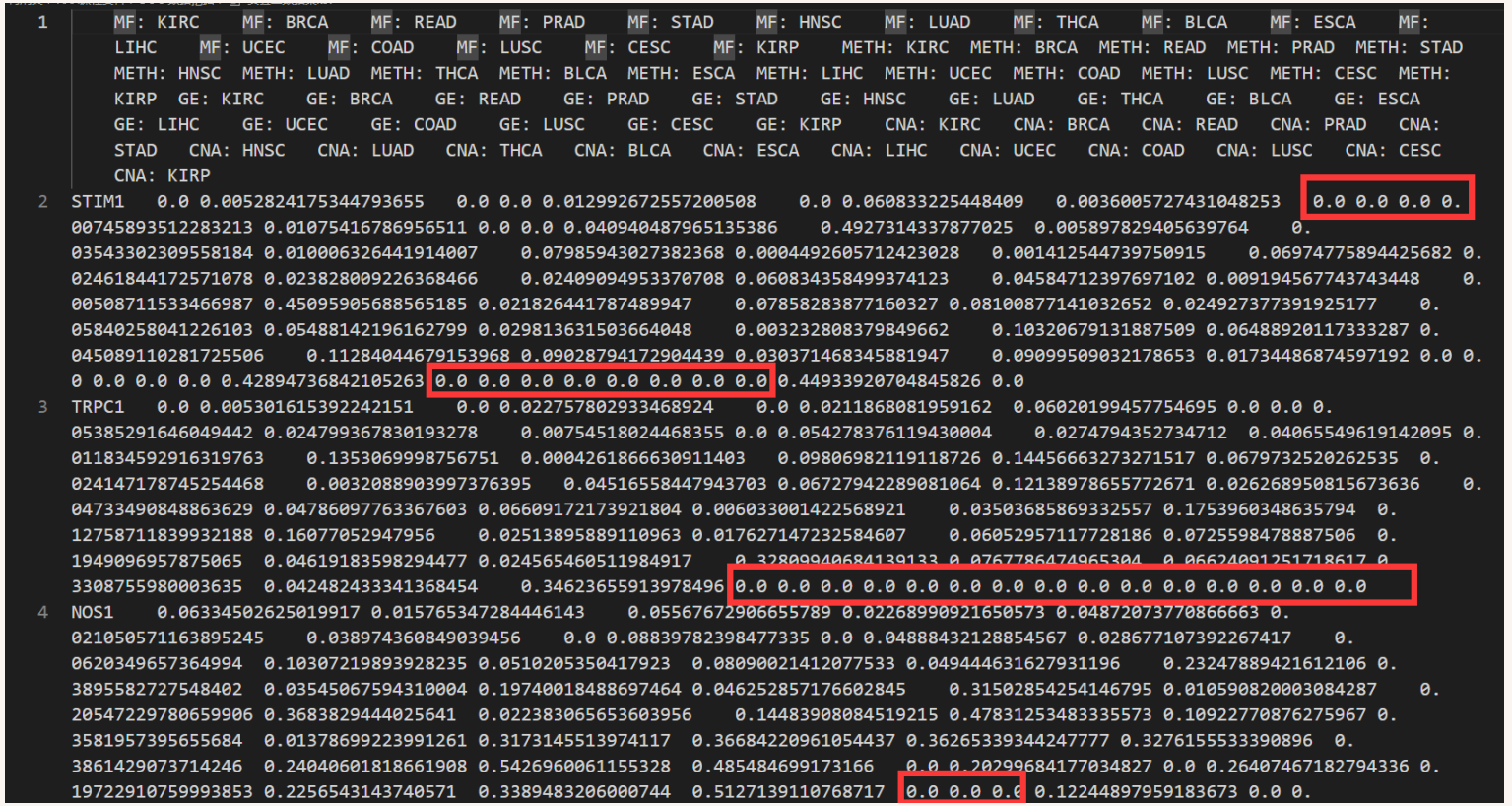
找到一种能度量特征重要性的方法

将特征选择融合在模型训练的过程中


Feature Engineering
Feature Extraction
Supervised Feature Selection
Dimensionality Reduction
Filters
Embedded Methods
Hybrid Methods
Wrappers
Boruta Algorithm
Missing value ratio
Information Gain
Chi-square Test
Fisher's score
FCBF Fast Correlation Filter
Permutation Feature Importance
Correlation coefficient
Low Variance filter
Mean absolute Difference
Dispersion ratio
Mutual Information
Relief
Regularization L1, L2, Elastic NET
Random Forest Importance
Filter & Wrapper Methods
reasons for feature selection
- simpler model
- shorter training time
- avoid curse of dimension
- reduce overfit
Forward Feature Selection
Backward Feature Selection
Exhaustive Feature Selection
Recursive Feature Selection
SURF
ReliefSeq
Extended ReliefF
RRELIEFF
RRELIEFF
Extended ReliefF
Tuned ReliefF
Relieved-F

# 初步降维

# （1）对数据进行初步降维：参考思路



```
1    MF: KIRC  MF: BRCA  MF: READ  MF: PRAD  MF: STAD  MF: HNSC  MF: LUAD  MF: THCA  MF: BLCA  MF: ESCA  MF:
     LIHC  MF: UCEC  MF: COAD  MF: LUSC  MF: CESC  MF: KIRP  METH: KIRC  METH: BRCA  METH: READ  METH: PRAD  METH: STAD
     METH: HNSC  METH: LUAD  METH: THCA  METH: BLCA  METH: ESCA  METH: LIHC  METH: UCEC  METH: COAD  METH: LUSC  METH: CESC  METH:
     KIRP  GE: KIRC  GE: BRCA  GE: READ  GE: PRAD  GE: STAD  GE: HNSC  GE: LUAD  GE: THCA  GE: BLCA  GE: ESCA
     GE: LIHC  GE: UCEC  GE: COAD  GE: LUSC  GE: CESC  GE: KIRP  CNA: KIRC  CNA: BRCA  CNA: READ  CNA: PRAD  CNA:
     STAD  CNA: HNSC  CNA: LUAD  CNA: THCA  CNA: BLCA  CNA: ESCA  CNA: LIHC  CNA: UCEC  CNA: COAD  CNA: LUSC  CNA: CESC
     CNA: KIRP
2    STIM1  0.0 0.0052824175344793655  0.0 0.0 0.012992672557200508  0.0 0.060833225448409  0.0036005727431048253  0.0 0.0 0.0 0.
     00745893512283213 0.01075416786956511 0.0 0.0 0.040940487965135386  0.4927314337877025  0.005897829405639764  0.
     03543302309558184 0.010006326441914007  0.07985943027382368 0.0004492605712423028  0.001412544739750915  0.06974775894425682 0.
     024618441725710 78 0.023828009226368466  0.02409094953370708 0.060834358499374123  0.04584712397697102 0.009194567743743448  0.
     00508711533466987 0.45095905688565185 0.021826441787489947  0.07858283877160327 0.08100877141032652 0.024927377391925177  0.
     05840258041226103 0.05488142196162799 0.029813631503664048  0.003232808379849662  0.10320679131887509 0.06488920117333287 0.
     045089110281725506  0.11284044679153968 0.09028794172904439 0.030371468345881947  0.09099509032178653 0.01734486874597192 0.0 0.
     0 0.0 0.0 0.0 0.42894736842105263 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.44933920704845826 0.0
3    TRPC1  0.0 0.005301615392242151  0.0 0.022757802933468924  0.0 0.0211868081959162  0.06020199457754695 0.0 0.0 0.
     05385291646049442 0.024799367830193278  0.00754518024468355 0.0 0.054278376119430004  0.0274794352734712  0.04065549619142095 0.
     011834592916319763  0.1353069998756751  0.0004261866630911403  0.09806982119118726 0.14456663273271517 0.0679732520262535  0.
     024147178745254468  0.0032088903997376395  0.04516558447943703 0.06727942289081064 0.12138978655772671 0.026268950815673636  0.
     04733490848863629 0.04786097763367603 0.06609172173921804 0.006033001422568921  0.03503685869332557 0.1753960348635794  0.
     12758711839932188 0.16077052947956  0.02513895889110963 0.017627147232584607  0.06052957117728186 0.0725598478887506  0.
     1949096957875065  0.04619183598294477 0.024565460511984917  0.32809940684139133 0.076778647496530 0.06624091251718617 0.
     3308755980003635  0.042482433341368454  0.34623655913978496 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
4    NOS1  0.06334502625019917 0.015765347284446143  0.05567672906655789 0.02268990921650573 0.04872073770866663 0.
     02105057116389 5245  0.038974360849039456  0.0 0.08839782398477335 0.0 0.04888432128854567 0.028677107392267417  0.
     0620349657364994  0.10307219893928235 0.0510205350417923  0.08090021412077533 0.049444631627931196  0.23247889421612106 0.
     3895582727548402  0.03545067594310004 0.19740018488697464 0.046252857176602845  0.31502854254146795 0.010590820003084287  0.
     20547229780659906 0.3683829444025641  0.022383065653603956  0.14483908084519215 0.47831253483335573 0.10922770876275967 0.
     3581957395655684  0.01378699223991261 0.3173145513974117  0.36684220961054437 0.36265339344247777 0.3276155533390896  0.
     3861429073714246  0.24040601818661908 0.5426960061155328  0.485484699173166  0.0 0.20299684177034827 0.0 0.26407467182794336 0.
     19722910759993853 0.2256543143740571  0.3389483206000744  0.5127139110768717  0.0 0.0 0.0 0.12244897959183673 0.0 0.
```

点开数据看就不发现存在很多的0值和很小很小的值，往往一个特征列上都是这样的值，对于这些几乎不变的值，是否可以把它们删去？这是不是一个值得思考的角度

（1）对数据进行初步降维：参考思路

【《清洗数据》这一章详情看书】

针对本题我们可以从以下角度思考：

| | |
|---|---|
| 删除未使用的列？ | 所有列均使用 |
| 删除具有缺失值的列？ | 无缺失值 |
| 不相关的特征？ | 暂时看不出来 |
| 低方差特征？ | ★这个角度可以思考，因为确实出现很多几乎不变的数据 |
| 多重共线性？ | 相关性系数过高（0.8），是一个可选角度 |
| 方差膨胀因子 (VIF)？ | 衡量多重共线性的指标，是一个可选角度 |

……

结合问题（初步降维，简单处理即可）、数据（0或者很小的数据较多）
我初步的想法是使用方差阈值特征选择来进行降维
简单来说就是筛选掉一些阈值

```python
import pandas as pd
from sklearn.feature_selection import VarianceThreshold

# 读取数据集
data = pd.read_csv('实验二数据集.tsv', delimiter='\t', index_col=0)

# 转置数据以使样本在行上，特征在列上
data = data.T

# 1. 方差阈值特征选择
variance_threshold = VarianceThreshold(threshold=0.035)  # 调整阈值
data_variance_selected = variance_threshold.fit_transform(data)

# 获取选择的列索引
selected_columns = data.columns[variance_threshold.get_support()]

# 保存选择的列名到CSV文件，以逗号分隔
selected_columns_text = ','.join(selected_columns)
with open('selected_columns.csv', 'w') as file:
    file.write(selected_columns_text)
```

© sklearn.feature_selection._variance_threshold.VarianceThreshold

```
def __init__(self, threshold: Any = 0.0) -> None
```

Feature selector that removes all low-variance features.

This feature selection algorithm looks only at the features (X), not the desired outputs (y), and can thus be used for unsupervised learning.

Read more in the User Guide .

**See Also**

SelectFromModel
: Meta-transformer for selecting features based on importance weights.

SelectPercentile
: Select features according to a percentile of the highest scores.

SequentialFeatureSelector
: Transformer that performs Sequential Feature Selection.

**Notes**

Allows NaN in the input. Raises ValueError if no feature in X meets the variance threshold.

**Examples**

The following dataset has integer features, two of which are the same in every sample. These are removed with the default setting for threshold:

```
>>> from sklearn.feature_selection import VarianceThreshold
```

删除所有低方差特征的特征选择器

| 方差阈值 | 结果维度 |
| --- | --- |
| 0.07 | 23 |
| 0.05 | 176 |
| 0.04 | 446 |
| 0.035 | 678 |
| 0.03 | 1039 |
| 0.01 | 6658 |

经过一些尝试，我发现方差阈值设定为0.035是比较好的，这样出来的结果维度为678维，处于要求的500-1000范围内。

这一步剔除了所有方差小于0.035的特征列，保证了剩下的方差都是较大的，这也是数据作为分类特征的价值所在。

# 进一步降维

## 2.使用无监督数据降维方法，比如PCA，ICA、UMap等进行降维

主成分分析（Principal Component Analysis，PCA）

独立成分分析（Independent Component Analysis，ICA）

UMAP（Uniform Manifold Approximation and Projection）

# （1）主成分分析（Principal Component Analysis，PCA）

主成分分析基本原理老师上课讲过了，不再赘述，这里关注评价指标

方差解释比例（Variance Explained Ratio）：每个主成分都能够解释原始数据中的一定比例方差。这个比例通常以百分比表示，例如，第一个主成分可能能够解释数据总方差的30%，第二个主成分能够解释15%，以此类推。

累计方差解释比例（Cumulative Variance Explained Ratio）：累计方差解释比例是指前n个主成分（或因子）的方差解释比例之和。它告诉我们，在保留了这些主成分的情况下，原始数据中的总方差的多少被解释了。通常，我们希望保留足够多的主成分，以使累积方差解释比例达到某个预定的阈值，以确保保留了足够的信息，同时降低数据维度。

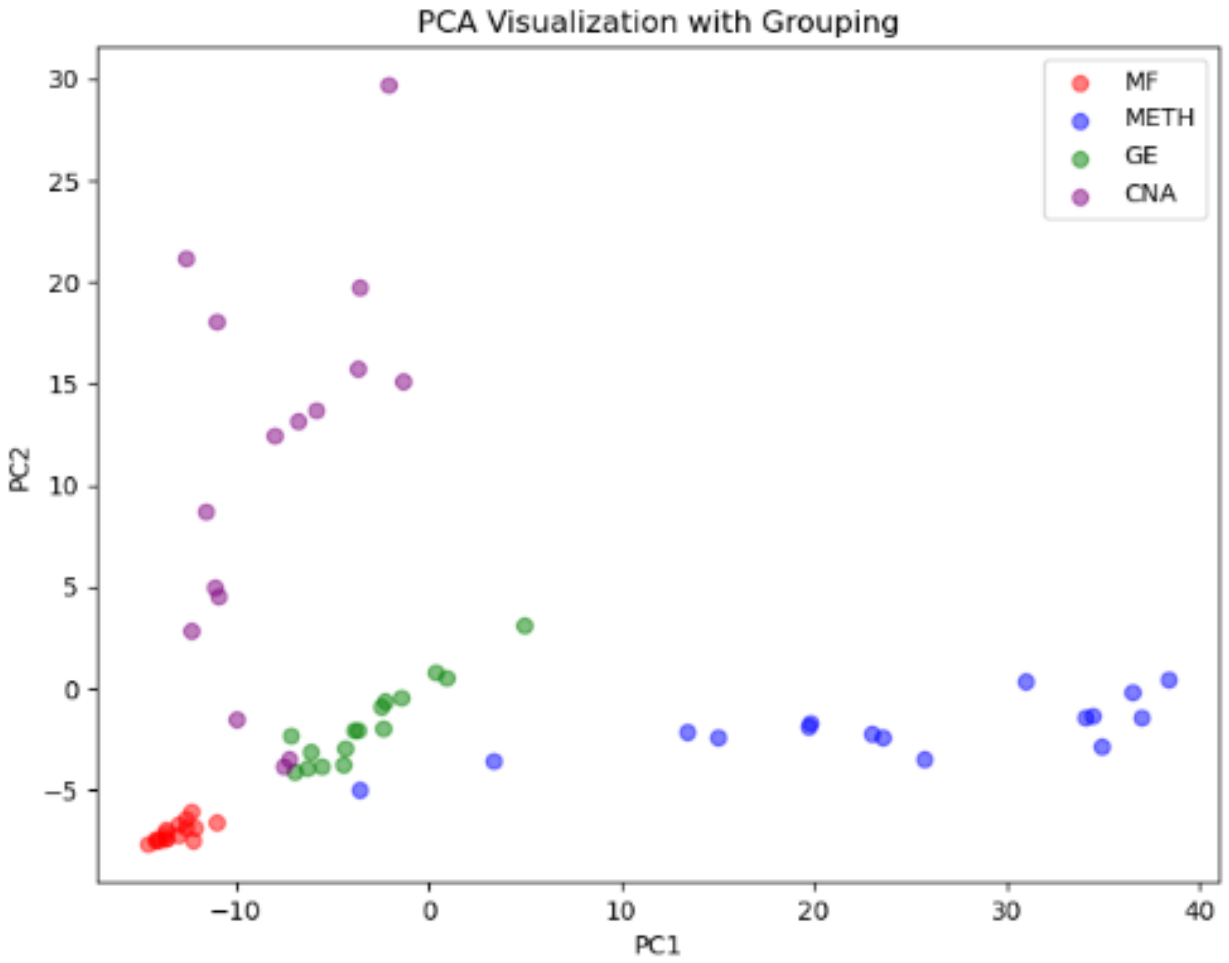
选择主成分数量：根据累积方差解释比例，可以决定保留多少主成分。一般来说，当累积方差解释比例达到一个满意的水平（通常在70%到95%之间）时，可以考虑停止增加主成分的数量，因为这足够解释大部分数据的方差。

| 目标维度<BR>（N_COMPONENTS） | 累计方差解释比例<BR>（CUMULATIVE VARIANCE EXPLAINED） |
| --- | --- |
| 64 | 100%(未开始降维) |
| 50 | 99.63% |
| 40 | 97.94% |
| 30 | 94.11% |
| 20 | 87.13% |
| 15 | 82.25% |
| 14 | 81.12% |
| 13 | 79.88% |
| 10 | 75.32% |
| 5 | 62.69% |
| 1 | 36.66% |

一般来说，累计方差解释比例低于50%是不可信的。在50%到80%时一般可信。在80%以上则称为可信。

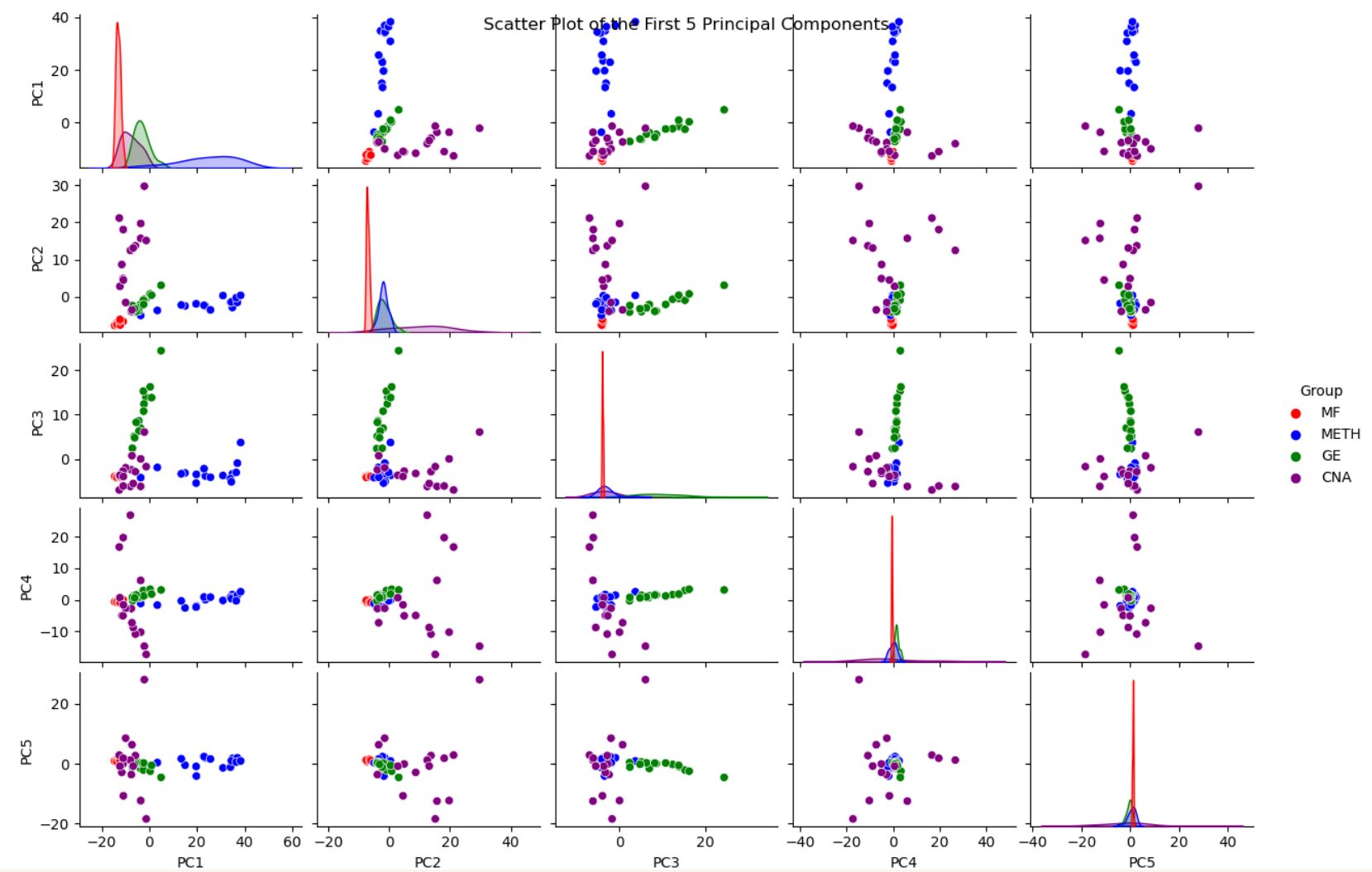
按照这种观点来看，我们可以选择14维作为目标维度，使用PCA进行降维，并利用降维的结果绘制部分主成分之间的三点关系图。

这里我们考虑到数据的“数据来源”与“数据类别”两个标签，其中“数据类别”有16种，不太适合分组呈现，故我这里就“数据来源”的不同取值“MF”，“METH”，“GE”，“CNA”进行分组分颜色显示。




首先获取最大的两个主成分PC1和PC2的散点关系图，可以发现MF的聚类情况表现地较明显，即MF来源的数据相似情况较大。

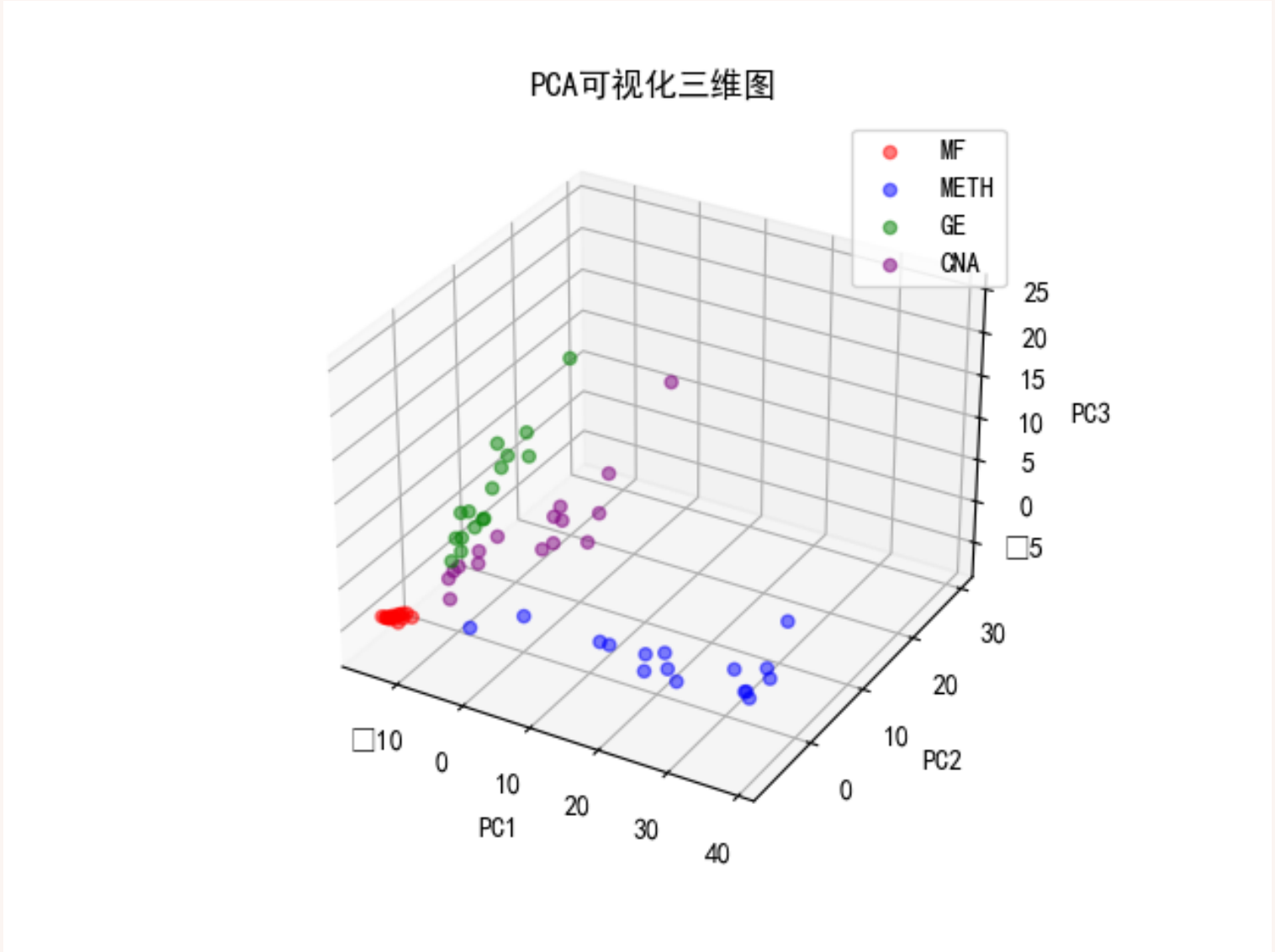
接下来查看前5个PCA主成分之间的相互散点关系图




分别表示不同主成分作为坐标轴的散点图，同一主成分表示不同类别（颜色）的点的聚合程度

可以发现，仅使用PC1与PC2主成分已经能够较为完美地完成相似数据的分类任务。

进一步，我们可以在三维图视下查看前三个主成分（PC1、PC2、PC3）的互相关系。



这里具体执行会遇到一个小问题，即n_components，看上去PC的值并没有改变。实际上这里是因为变化过于微小导致看上去并没有变化，如果我们将小数位数放多一些，实际上还是有明显的变化的。

n_components = 5

```
Explained Variance Ratio for PC1: 36.6633189992%
Explained Variance Ratio for PC2: 9.6806797212%
Explained Variance Ratio for PC3: 6.5627968456%
Explained Variance Ratio for PC4: 5.6423145826%
Explained Variance Ratio for PC5: 4.1373794292%
Cumulative Variance Explained by 5 Principal Components:
62.69%
```

可见数据还是改变了。

原因可能在于数据中的特征之间没有足够的差异，或者特征之间的相关性非常高，导致 PCA 的主成分没有多大变化，初步判定应该是数据本身的问题，主成分的方法没有太大的问题。

n_components = 14

```
Explained Variance Ratio for PC1: 36.6633189992%
Explained Variance Ratio for PC2: 9.6806797201%
Explained Variance Ratio for PC3: 6.5627968242%
Explained Variance Ratio for PC4: 5.6423145516%
Explained Variance Ratio for PC5: 4.1373791936%
Explained Variance Ratio for PC6: 3.3504752123%
Explained Variance Ratio for PC7: 2.8403813497%
Explained Variance Ratio for PC8: 2.4943752057%
Explained Variance Ratio for PC9: 2.0477662556%
Explained Variance Ratio for PC10: 1.9012579517%
Explained Variance Ratio for PC11: 1.6803878986%
Explained Variance Ratio for PC12: 1.5090011367%
Explained Variance Ratio for PC13: 1.3680188469%
Explained Variance Ratio for PC14: 1.2435843400%
Cumulative Variance Explained by 14 Principal Components:
81.12%
```

# （2）独立成分分析（Independent Component Analysis，ICA）

模型简要介绍：

建立混合模型：定义混合模型，假设混合信号是独立成分的线性组合。这个模型通常表示为X = AS，其中：

- X 是观测到的混合信号矩阵，每一列代表一个观测时间点或传感器通道。
- A 是混合矩阵，包含了混合系数，表示混合成分与观测信号之间的关系。
- S 是独立成分矩阵，包含了独立成分的时间序列或通道。

估计混合矩阵 A 和独立成分矩阵 S。这通常涉及到最大独立性估计（maximum likelihood estimation for independent sources, maximum entropy ICA）等方法。ICA算法的目标是找到A和S，使得S中的各行（独立成分）是统计上不相关的。

# 部分评价指标

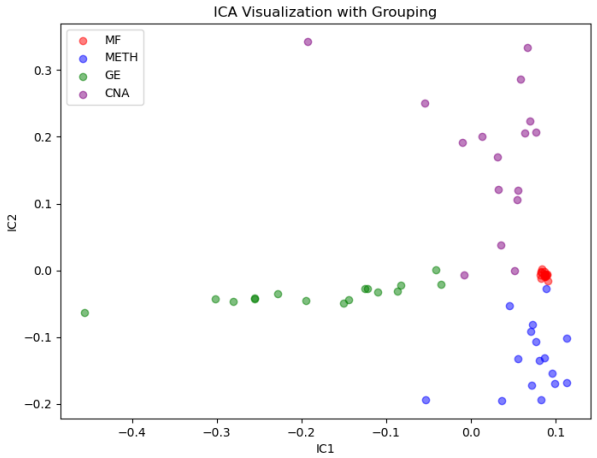
SNR（信噪比）：用于衡量ICA分离的信号成分与噪声之间的相对强度。在ICA的背景下，这意味着SNR用于度量独立成分的清晰度，即成分中信号与噪声的比例。
SNR(dB) = 10 * log10(信号功率 / 噪声功率)

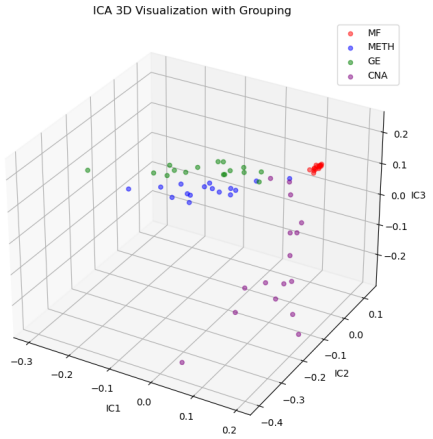
MI（互信息）：用于衡量ICA分离的成分与原始信号之间的信息传输量。它可以帮助确定分离的成分是否包含原始信号的信息。
MI(X, Y) = ∫∫ p(x, y) * log(p(x, y) / (p(x) * p(y))) dx dy，其中 X 表示原始信号，Y 表示ICA成分。

峰度（Kurtosis)：用于描述概率分布尾部（尤其是高阶短尾或长尾）相对于正态分布的“尖锐度”或“平缓度”的统计量。它用于度量分布中数据点分布的尖峰程度。
Kurt(X) = E[(X - μ)^4] / (σ^4)，其中 X 是数据集，μ 是均值，σ 是标准差。

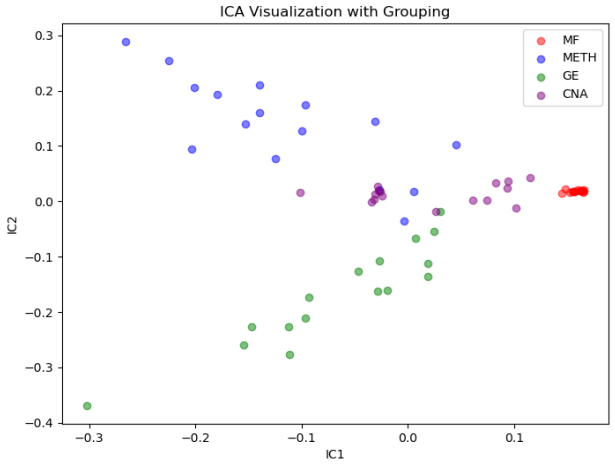
偏度（Skewness）:用于描述数据分布的不对称性，即数据在分布中的偏向。正偏度表示数据右偏，负偏度表示数据左偏，零偏度表示分布对称。
Skew(X) = E[(X - μ)^3] / (σ^3)，其中 X 是数据集，μ 是均值，σ 是标准差。

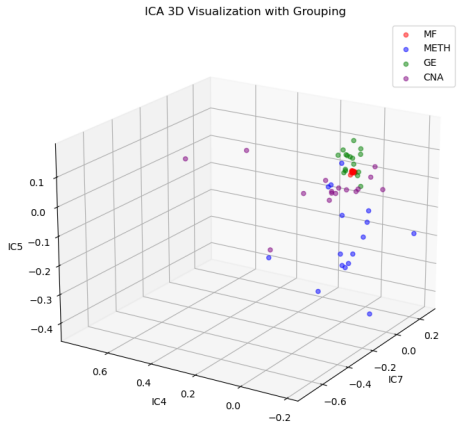
## n_components =3时




## n_components =10时




## n_components =3时

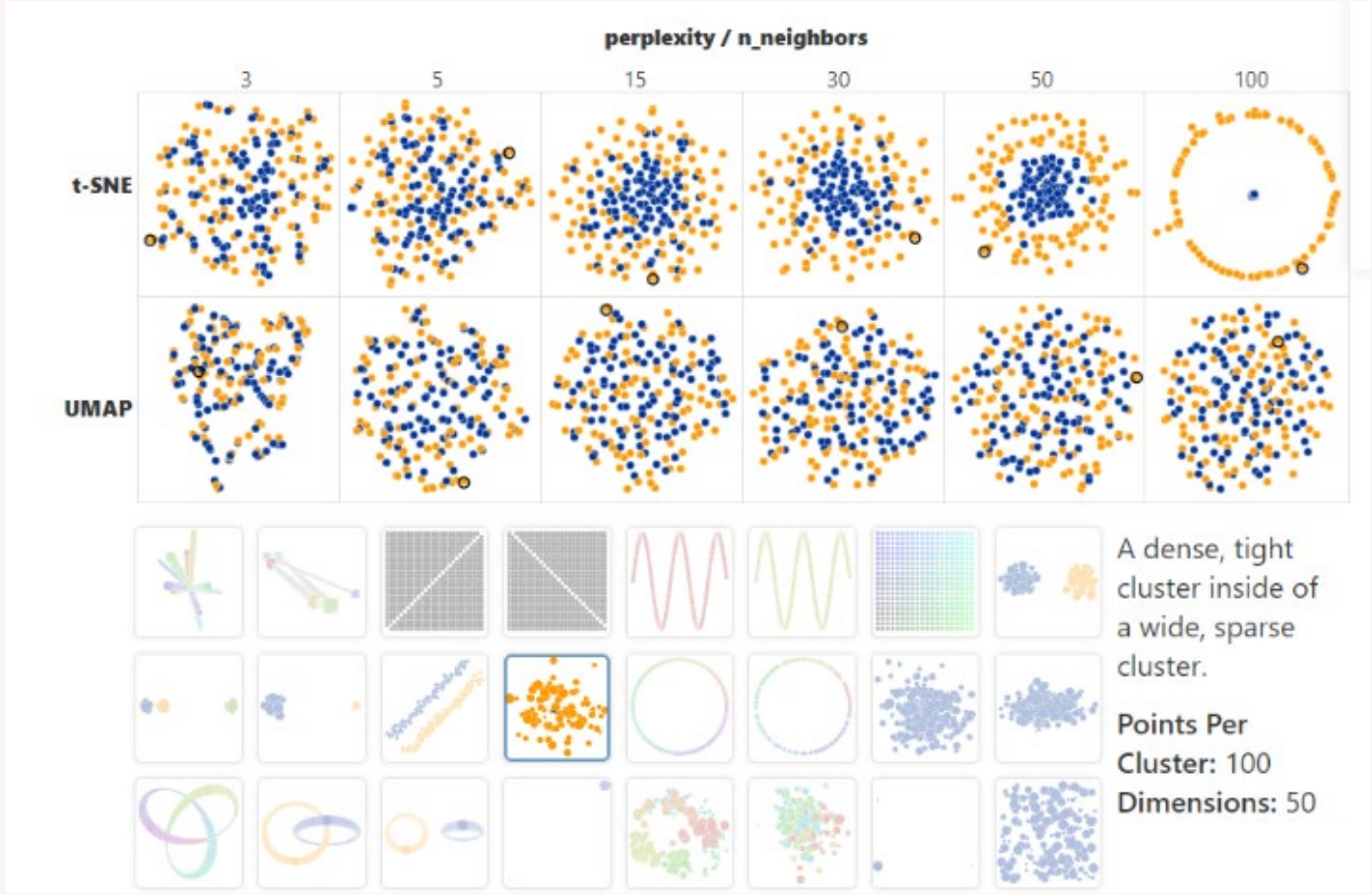
```
Signal-to-Noise Ratio (SNR): 53.28
Mutual Information (MI) with True Signal: 3.6379
  Component  Kurtosis  Skewness
0       IC1  1.811055 -1.561985
1       IC2  0.604004  0.946129
2       IC3 -1.067555  0.467046
```

## n_components =10时

```
Signal-to-Noise Ratio (SNR): 2.57
Mutual Information (MI) with True Signal: 3.6379
  Component   Kurtosis  Skewness
0       IC1  -0.798939 -0.287743
1       IC2   0.744784 -0.457725
2       IC3   5.696764  2.139519
3       IC4  20.054037 -3.384852
4       IC5  40.362070 -6.036482
5       IC6  41.987941  6.188286
6       IC7  20.493603 -4.103255
7       IC8  26.287692 -4.377644
8       IC9  14.829184 -3.959355
9      IC10  15.423052  3.054299
```

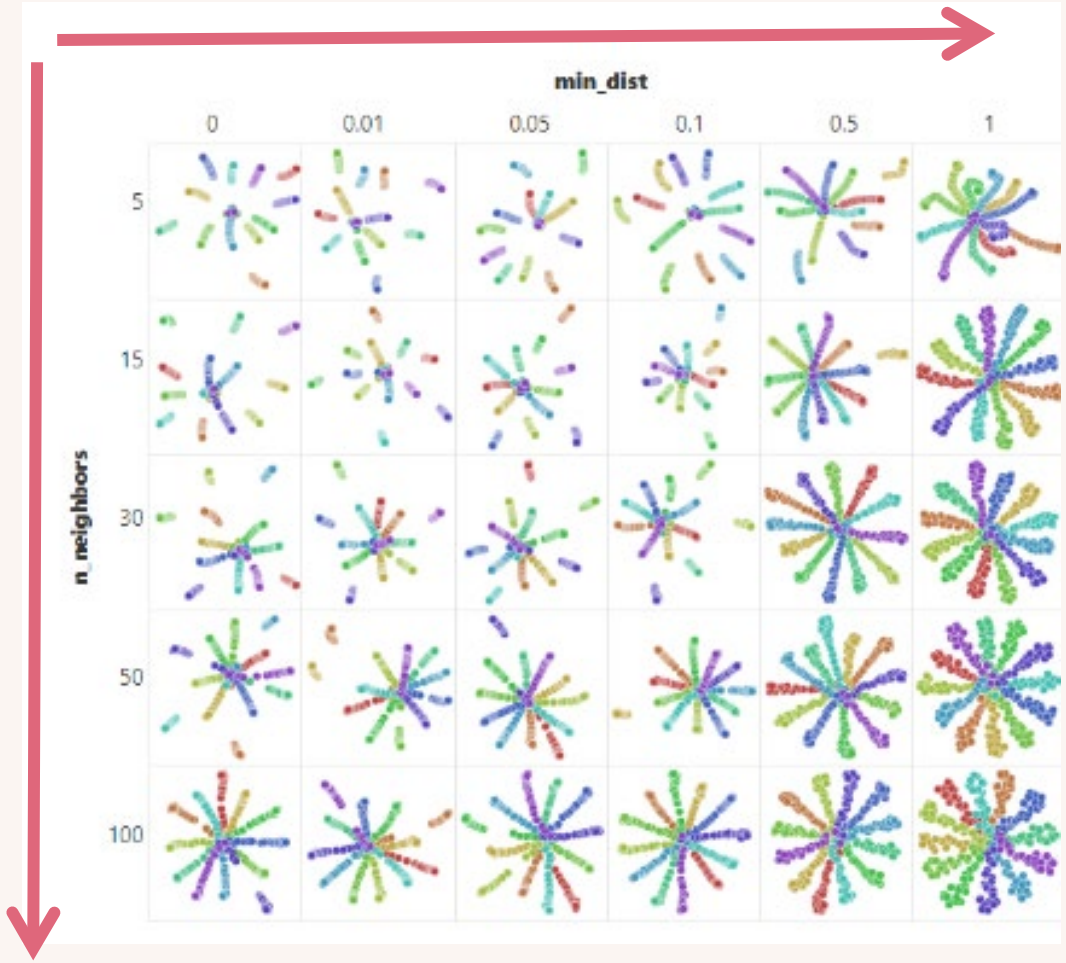
挑选IC4,IC5,IC7作为三个轴

# （3）UMAP（Uniform Manifold Approximation and Projection）




统一流形逼近与投影，UMAP（Uniform Manifold Approximation and Projection）是一种非线性降维技术，用于将高维数据映射到低维空间以进行数据可视化、聚类和降维分析。UMAP是一种基于流形学习的方法，旨在保留数据中的局部结构和全局结构，并在降维后尽量保持数据点之间的拓扑关系。

UMAP与t-SNE之间有相似之处。简单来说，UMAP首先构建数据的高维图表示，然后优化低维图以使其在结构上尽可能相似。




UMAP两个重要的参数

n_neighbors

最重要的参数是n_neighbors，用于构造初始高维图的近似最近邻的数量。它有效地控制UMAP如何平衡局部结构与全局结构：较小的值将通过限制在分析高维数据时考虑的相邻点的数量来推动UMAP更多地关注局部结构，而较大的值将推动UMAP代表全局结构，同时失去了细节。

min_dist

我们将研究的第二个参数是 min_dist，即低维空间中点之间的最小距离。此参数控制UMAP将点聚集在一起的紧密程度，较低的值会导致嵌入更紧密。较大的 min_dist值将使UMAP将点更松散地打包在一起，而是专注于保留广泛的拓扑结构。

# 评价指标

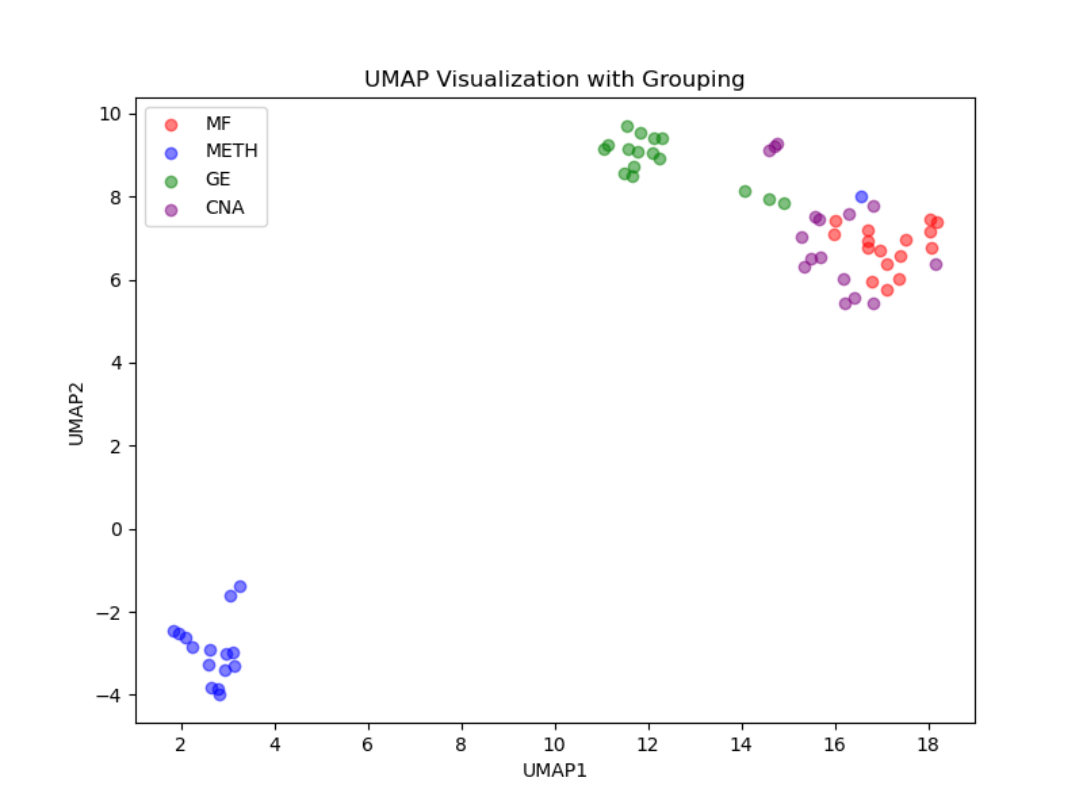
1. 高维数据表示：UMAP从高维数据集开始，通常以N×D的形式表示，其中N是样本数量，D是特征维度。

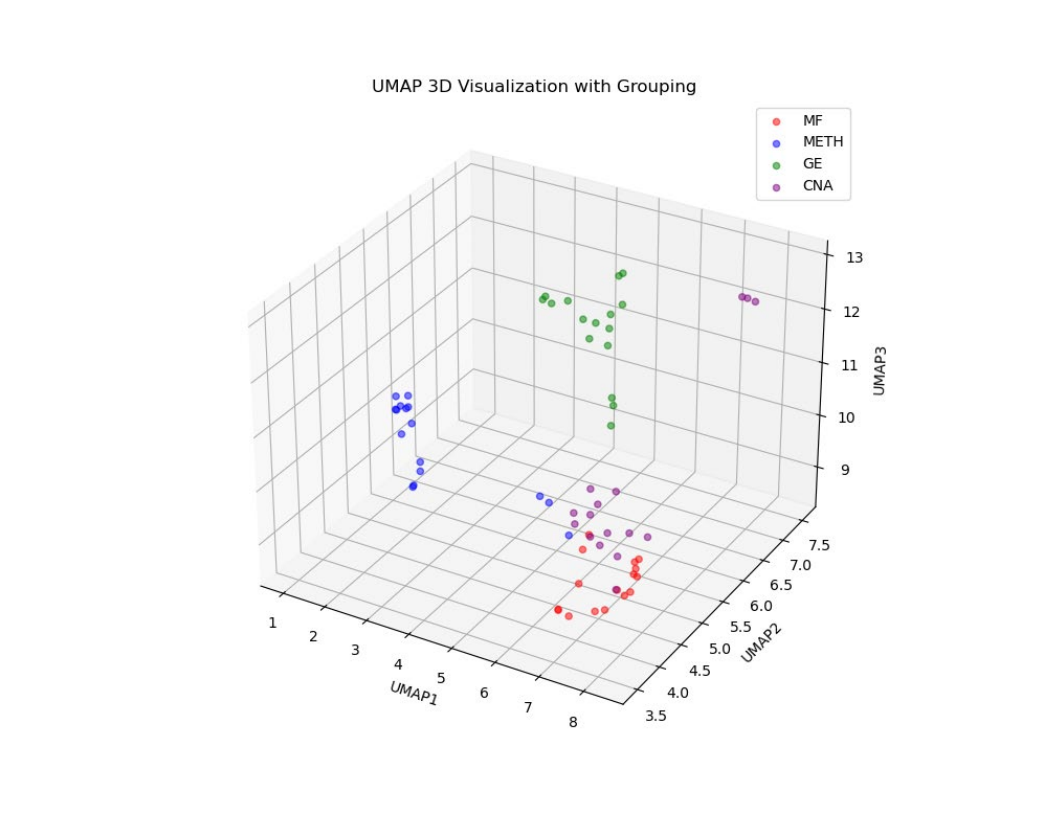
2. 构建连通图：UMAP首先构建一个表示数据点之间连接的权重图。这一步骤包括以下子步骤：
   - 确定邻近性：对于每个数据点，UMAP确定其在高维空间中的k个最近邻居。这是通过计算数据点之间的距离来完成的。
   - 权重计算：UMAP计算每对邻近数据点之间的权重，反映它们之间的连接强度。UMAP使用距离度量来计算权重，通常采用高斯核函数来赋予邻近点更高的权重，而远离点较低的权重。

3. 优化连通图：UMAP使用拓扑优化技术，如随机梯度下降，来最小化在低维空间的连通图与高维连通图之间的拓扑误差。这有助于保留数据的全局结构。

4. 低维嵌入：UMAP将优化后的高维连通图映射到低维空间。通常，UMAP将数据映射到2D或3D空间以进行可视化。映射是通过优化低维坐标以最小化高维图与低维图之间的拓扑误差来实现的。

# UMAP二维/三维图像


UMAP Visualization with Grouping
MF
METH
GE
CNA
UMAP2
UMAP1
10
8
6
4
2
0
−2
−4
2
4
6
8
10
12
14
16
18


UMAP 3D Visualization with Grouping
MF
METH
GE
CNA
UMAP1
UMAP2
UMAP3

代码讲解：

ICA

```python
# 提取特征（所有列）
features = data.iloc[:, :]

# 对特征进行标准化
scaler = StandardScaler()
scaled_features = scaler.fit_transform(features)

# 指定降维后的维度
n_components = 3  # 降维后的维度

# 创建ICA模型并进行降维
ica = FastICA(n_components=n_components)
ica_result = ica.fit_transform(scaled_features)
```

PCA

```python
# 提取特征（所有列）
features = data.iloc[:, :]

# 对特征进行标准化
scaler = StandardScaler()
scaled_features = scaler.fit_transform(features)

# 指定降维后的维度
n_components = 14  # 降维后的维度

# 创建PCA模型并进行降维
pca = PCA(n_components=n_components)
pca_result = pca.fit_transform(scaled_features)
```

基本上没有太大区别，就是在超参数的选择上，以及有自己的特性（UMAP）

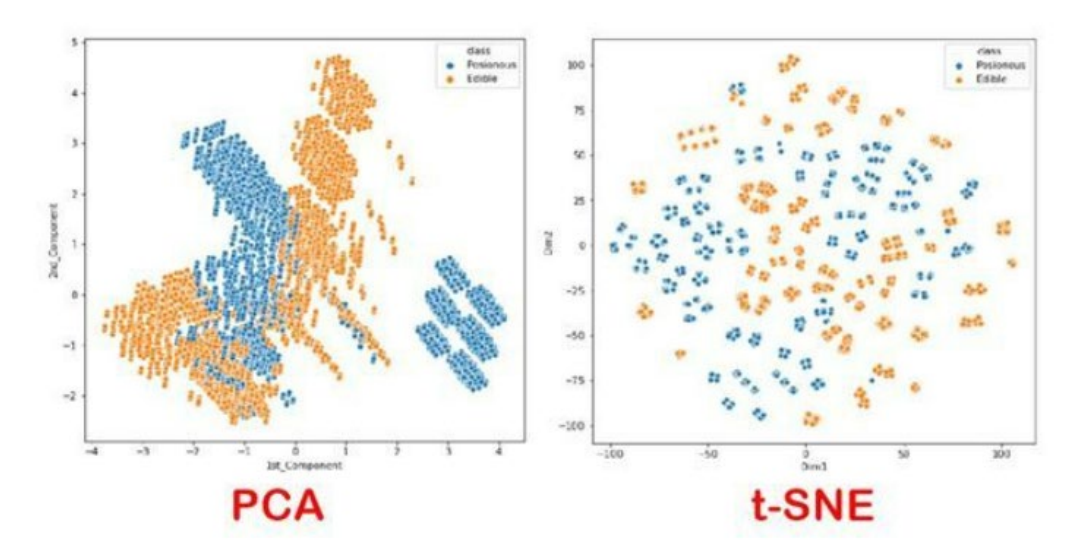
UMAP

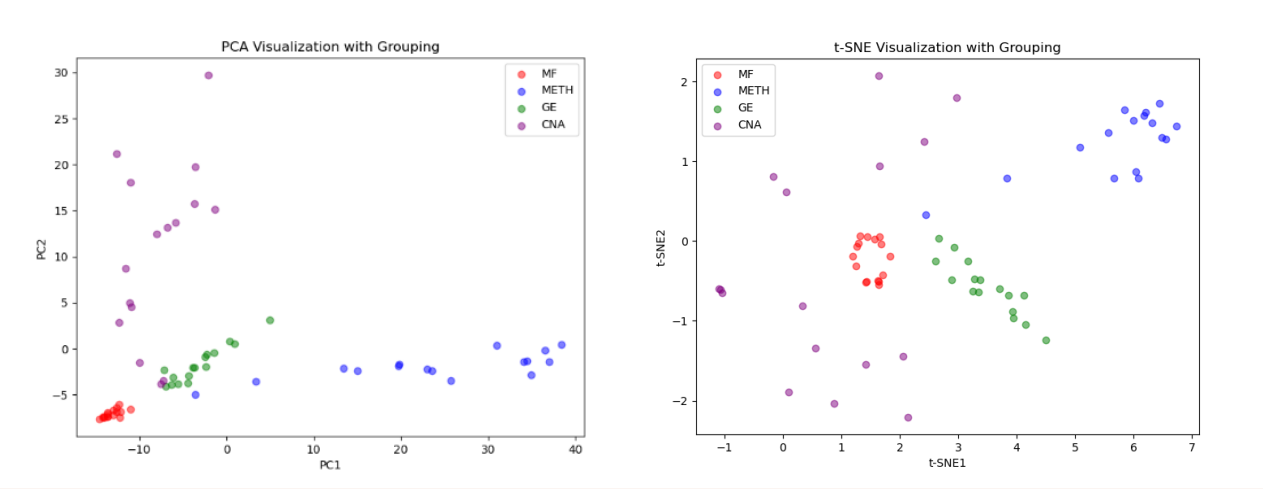
```python
# 提取特征（所有列）
features = data.iloc[:, :]

# 对特征进行标准化
scaled_features = StandardScaler().fit_transform(features)

# 创建UMAP模型并进行降维
n_components = 2  # 降维后的维度
umap_model = umap.UMAP(n_neighbors=4, n_components=n_components)
umap_result = umap_model.fit_transform(scaled_features)
```

# t-SNE



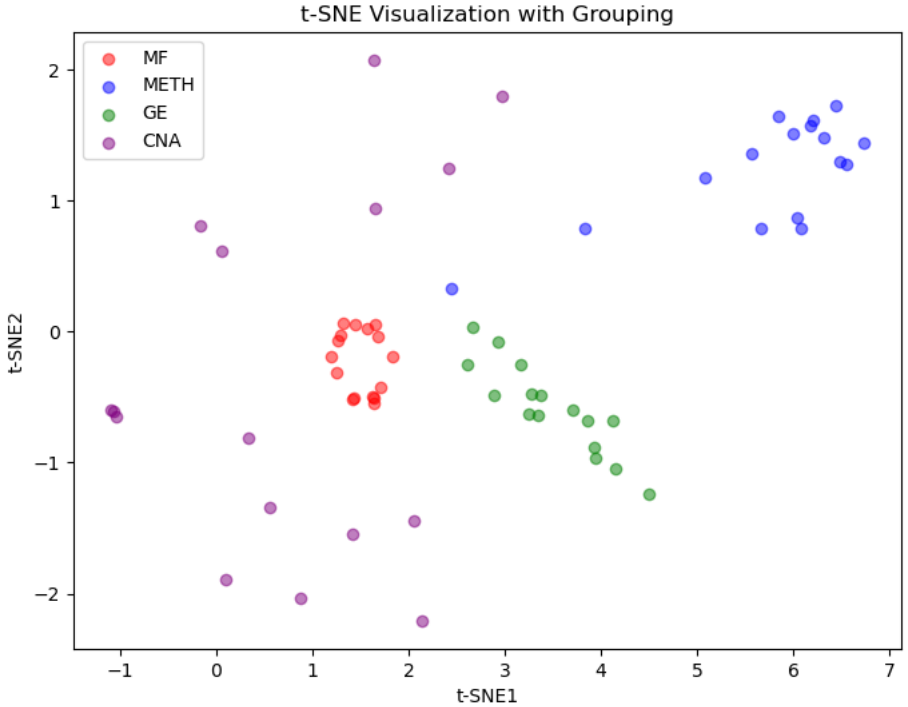

t-SNE 要保留数据的局部特征
应该满足这样的要求：原先距离近的数据，降维之后距离应该也很近；原先距离远的数据，降维之后距离应该也很远。

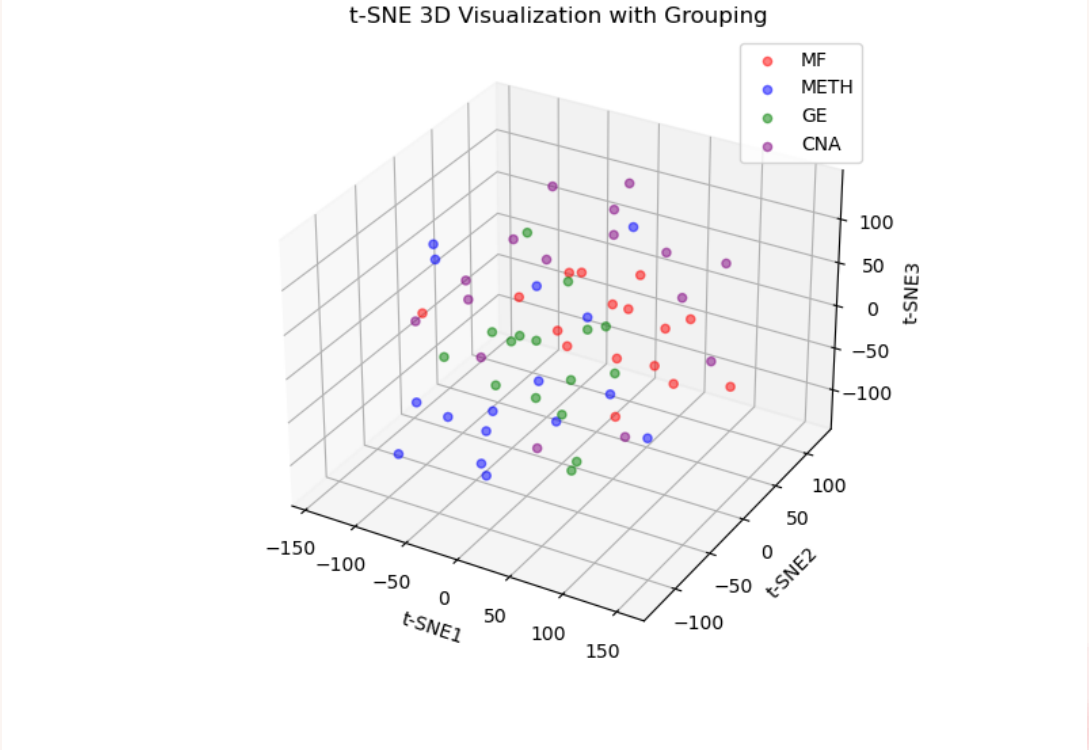
那么我们怎么去做到这一点呢？t-SNE 中主要是将“距离的远近关系”转化为一个概率分布，每一个概率分布就对应着一个“样本间距离远近”的关系。而降维前后的数据都各自对应着一个概率分布，我们就希望这两个概率分布足够的接近。

再回想一下UMAP
UMAP基于流形学习的思想，它通过优化流形上的连接性来构建一个低维表示

一般来说，UMAP比t-SNE 要快很多（5倍以上）

在第一步PCA降维到14维的基础上进一步进行降维，读取PCA的14维结果进行进一步降维，最终降到2或3维。



代码讲解：指定超参数即可，没有太多的变化

右边是绘图部分，一般照抄模板再根据自己需要进行微调即可

```
# 提取特征（所有列）
features = data.iloc[:, :]

# 创建t-SNE模型并进行降维
n_components = 2  # 降维后的维度
tsne_model = TSNE(n_components=n_components, random_state=7)
tsne_result = tsne_model.fit_transform(features)
```

```
# 创建一个新列，用于标识数据行所属的部分
tsne_df['Group'] = None
tsne_df.loc[0:16, 'Group'] = 'MF'  # 第一部分
tsne_df.loc[16:32, 'Group'] = 'METH'  # 第二部分
tsne_df.loc[32:48, 'Group'] = 'GE'  # 第三部分
tsne_df.loc[48:64, 'Group'] = 'CNA'  # 第四部分

# 定义颜色映射
colors = {'MF': 'red', 'METH': 'blue',
'GE': 'green', 'CNA': 'purple'}

# 根据分组使用不同颜色绘制点
plt.figure(figsize=(8, 6))
for group, color in colors.items():
    group_data = tsne_df[tsne_df['Group'] ==
group]
    plt.scatter(group_data[selected_components[0]],
group_data[selected_components[1]], c=color,
label=group, alpha=0.5)

plt.xlabel(selected_components[0])
plt.ylabel(selected_components[1])
plt.title('t-SNE Visualization with Grouping')
plt.legend()
plt.show()
```

# 感谢指导