

8

Data Transformations

数据转换

代数和统计方法处理数据，以便后续回归、分类或聚类



没有数据，就得出结论，这是大错特错。

It is a capital mistake to theorize before one has data.

—— 阿瑟·柯南·道尔 (Arthur Conan Doyle) | 英国小说作家、医生 | 1859 ~ 1930



```
numpy.random.exponential() 产生满足指数分布随机数
pandas.plotting.parallel_coordinates() 绘制平行坐标图
scipy.stats.boxcox() Box-Cox 数据转换
scipy.stats.probplot() 绘制 QQ 图
scipy.stats.yeojohnson() Yeo-Johnson 数据转换
seaborn.distplot() 绘制概率直方图
seaborn.heatmap() 绘制热图
seaborn.jointplot() 绘制联合分布和边际分布
seaborn.kdeplot() 绘制 KDE 核概率密度估计曲线
seaborn.violinplot() 绘制数据小提琴图
sklearn.preprocessing.MinMaxScaler() 归一化数据
sklearn.preprocessing.PowerTransformer() 广义幂变换
sklearn.preprocessing.StandardScaler() 标准化数据
```

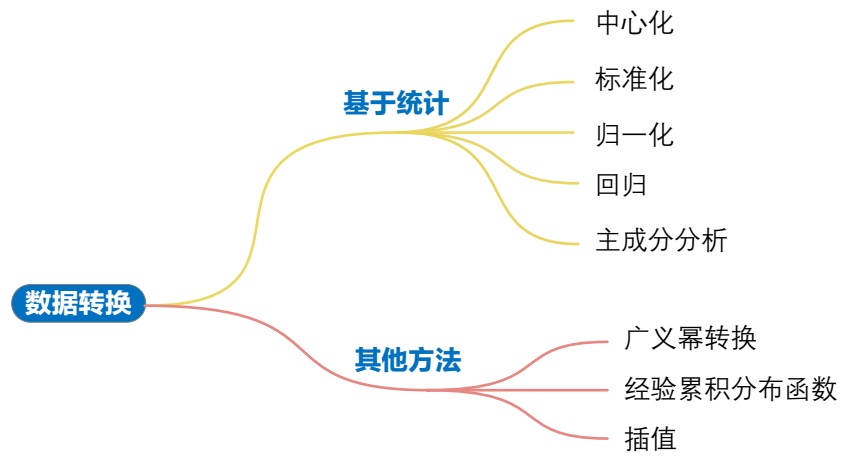
本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com



本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

8.1 数据转换

本章介绍**数据转换** (data transformation) 的常见方法。数据转换是数据预处理的重要一环，用来转换要分析的数据集，使其更方便后续建模，比如回归分析、分类、聚类、降维。注意，数据预处理时，一般先处理缺失值、离群值，然后再数据转换。

数据转换的外延可以很广。函数 (比如指数函数、对数函数)、中心化、标准化、概率密度估计、插值、回归分析、主成分分析、时间序列分析、平滑降噪等，某种意义上都可以看做是数据转换。比如，经过主成分分析处理过的数据可以成为其他算法的输入。

图 1 总结本章要介绍的几种常见数据转换方法。下一章专门介绍插值。图 1 可以用作本章思维导图。

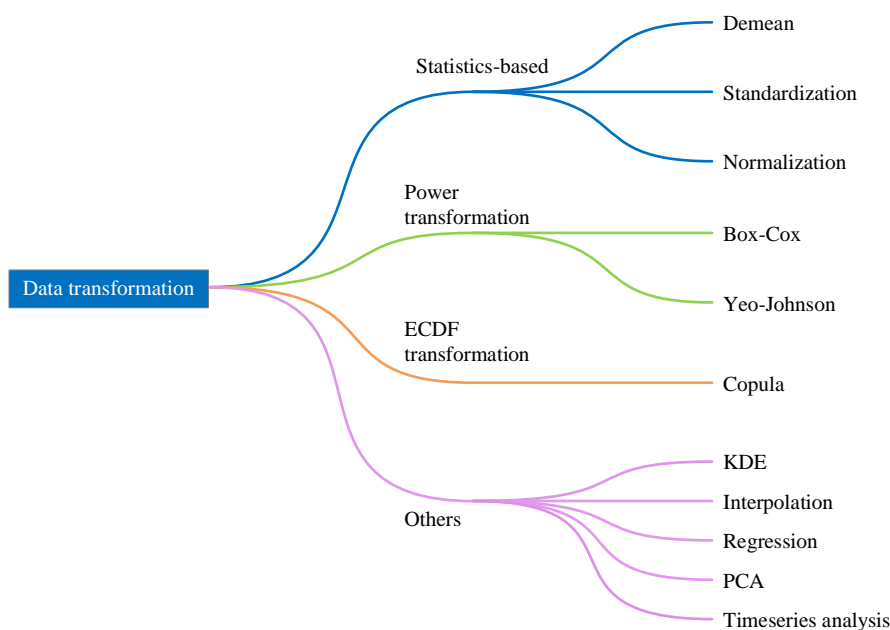


图 1. 常见数据转换方法

8.2 中心化：去均值

数据**中心化** (centralize, demean), 也叫去均值，是基于统计最基本的数据转换。

对于一个给定特征，**去均值数据** (demeaned data, centered data) 的定义为：

$$Y = X - \text{mean}(X) \quad (1)$$

其中， $\text{mean}(X)$ 计算期望值或均值。

一般情况，多特征数据每一列数据代表一个特征。多特征数据的中心化，相当于每一列数据分别去均值。对于均值几乎为 0 的数据，去均值处理效果肯定不明显。

原始数据

本节用四种可视化方案展示数据，它们分别是热图、KDE 分布、小提琴图和平行坐标图。图 2 ~ 图 5 所示为这四种可视化方案展示的鸢尾花原始四个特征数据。

相信丛书读者对前三种可视化方案应该很熟悉。这里简单介绍图 5 所示**平行坐标图** (parallel coordinate plot)。

一个正交坐标系可以用来展示二维或三维数据，但是对于高维多元数据，正交坐标系则显得无力。而平行坐标图，可以用来可视化多特征数据。平行坐标图采用多条平行且等间距的轴，以折线形式呈现数据。图 5 还用不同颜色折线代表分类标签。

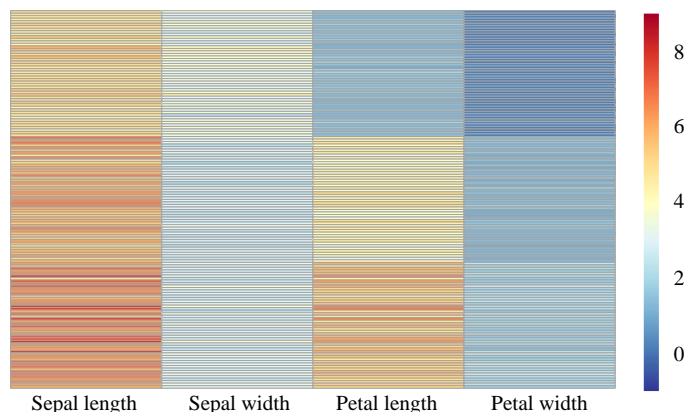


图 2. 鸢尾花数据，原始数据矩阵 X

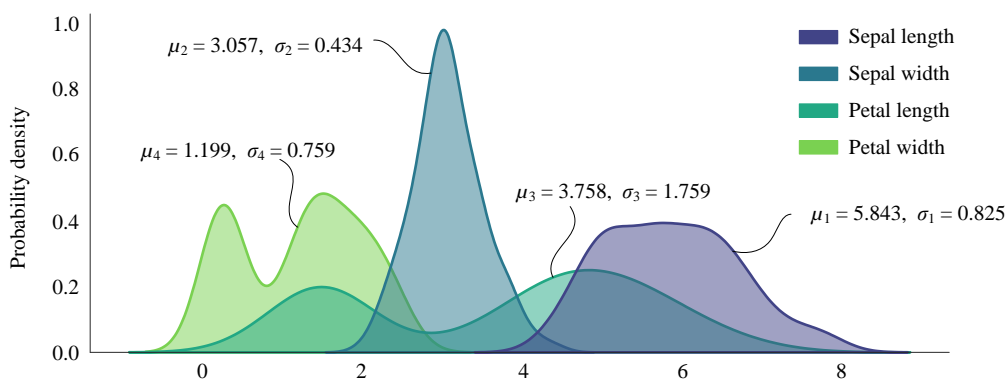


图 3. 鸢尾花数据四个特征上分布，KDE 估计

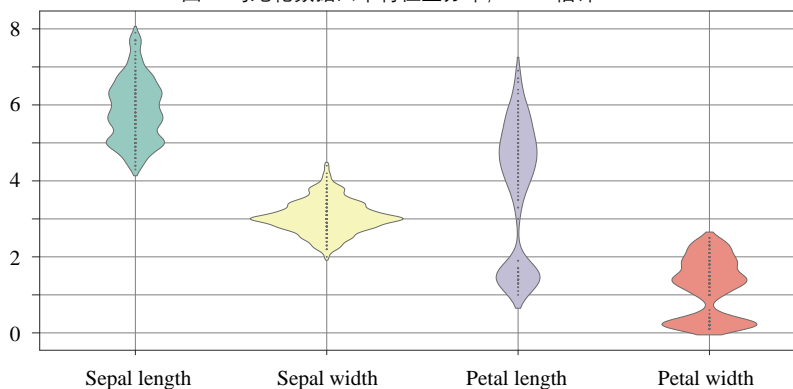


图 4. 鸢尾花原始数据，小提琴图

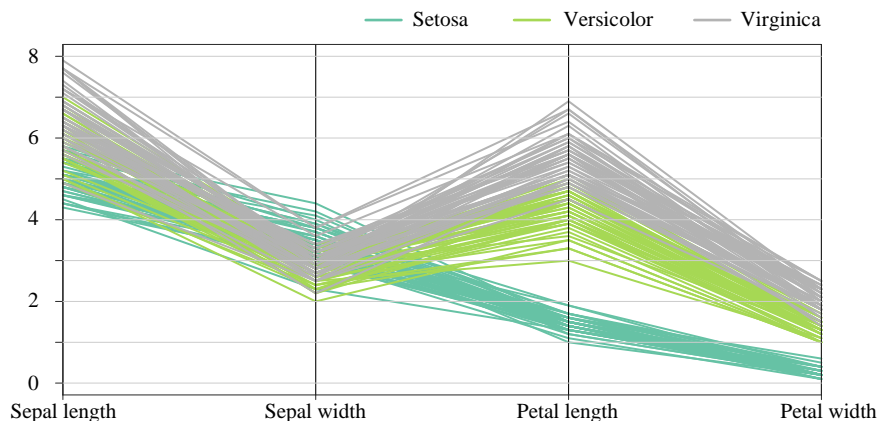


图 5. 鸢尾花数据，平行坐标图

代码 1 绘制图 2 ~ 图 5。下面聊聊其中关键语句。

a 使用 Seaborn 的 `heatmap` 函数绘制热图。参数包括数据 `X`，颜色映射 `cmap` 为 'Red-Yellow-Blue' 反转 (`RdYlBu_r`)，`x` 轴刻度标签为数据集 `X_df` 的列名，颜色条方向为垂直，颜色条的范围为 -1 到 9。

b 使用 Seaborn 的 `kdeplot` 函数绘制核密度估计图。参数包括数据 `X`，设置填充为 `True`，`common_norm` 为 `False` 表示每个数据集的密度将在其自己的范围内进行标准化，`alpha` 设置透明度，`linewidth` 设置线宽度，`palette` 设置颜色映射。

c 使用 Seaborn 的 `violinplot` 函数绘制小提琴图。参数包括数据 `X_df`，颜色映射 `palette` 为 "Set3"，带宽 `bw` 为 0.2，`cut` 为 1 表示在每个小提琴的两端截断，`linewidth` 设置线宽度，`inner` 表示在小提琴内部显示的元素类型，`orient` 为垂直方向。

d 使用 pandas 的 `plotting.parallel_coordinates` 函数绘制平行坐标图。参数包括数据集 `iris_sns`，指定类别变量 `species`，颜色映射为 "Set2"。

```

sns.set_style("ticks")

# 绘制热图
fig, ax = plt.subplots()
a sns.heatmap(X, ax = ax,
               cmap='RdYlBu_r',
               xticklabels=list(X_df.columns),
               cbar_kws={"orientation": "vertical"},
               vmin=-1, vmax=9)
plt.title('X')

# 绘制KDE
fig, ax = plt.subplots()
b sns.kdeplot(data=X, fill=True, ax = ax,
              common_norm=False,
              alpha=.3, linewidth=1,
              palette = "viridis")
plt.title('Distribution of X columns')

# 绘制小提琴图
fig, ax = plt.subplots()
c sns.violinplot(data=X_df, palette="Set3", bw=.2,
                 cut=1, linewidth=0.25, ax = ax,
                 inner="points", orient="v")
ax.grid(linestyle='--', linewidth=0.25, color=[0.5,0.5,0.5])

# 绘制平行坐标图
fig, ax = plt.subplots()
# Make the plot
d pd.plotting.parallel_coordinates(iris_sns,
                                   'species',
                                   colormap=plt.get_cmap("Set2"))

plt.show()

```

代码 1. 四个可视化方案 |  Bk6_Ch08_01.ipynb

中心化数据

图 6 ~ 图 9 则用这四种可视化方案展示去均值后鸢尾花数据。

➡ 《矩阵力量》介绍过，对于多特征数据，去均值相当于将数据质心移动到 0 ，但是对数据分布的离散度没有任何影响。

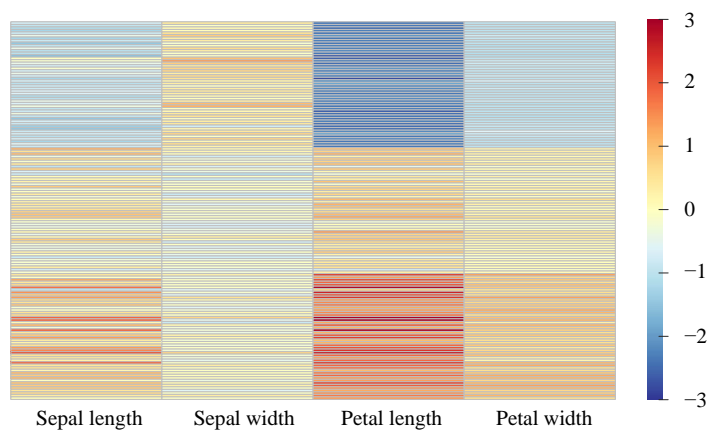


图 6. 数据热图，去均值

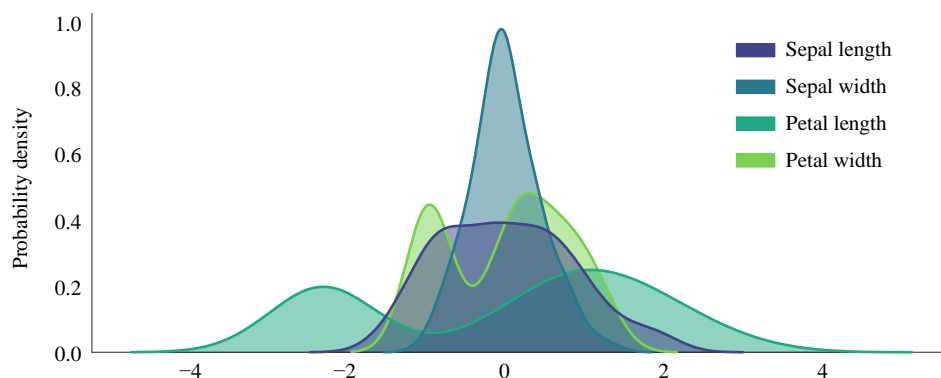


图 7. 数据 KDE 分布估计，去均值

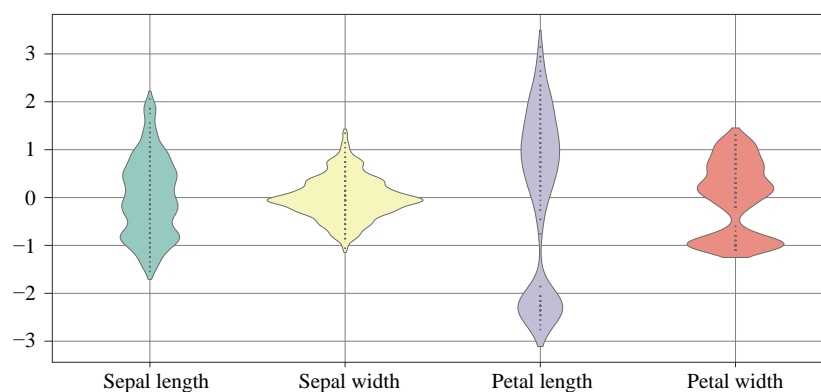


图 8. 小提琴图，去均值

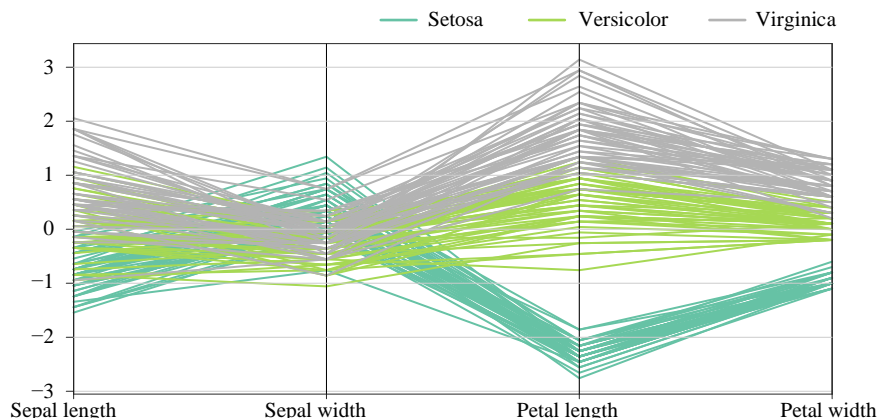


图 9. 平行坐标图，去均值

8.3 标准化：Z 分数

标准化 (standardization) 对原始数据先去均值，然后再除以标准差：

$$Z = \frac{X - \text{mean}(X)}{\text{std}(X)} \quad (2)$$

处理得到的数值实际上是原始数据的 Z 分数，表达若干倍的标准差偏移。比如，某个数值处理后结果为 3，这代表数据距离均值 3 倍标准差偏移。

⚠ 注意，Z 分数的正负代表偏离均值的方向。

在机器学习中，standardization 和 normalization 通常分别翻译为标准化和归一化。这两种预处理方法的主要区别在于对数据的缩放方式不同。

标准化通常是指将数据缩放到均值为 0，标准差为 1 的标准正态分布上。标准化可以通过先减去均值，再除以标准差来实现。标准化可以使得不同特征之间的数值尺度相同，避免某些特征对模型的影响过大，从而提高模型的鲁棒性和稳定性。

归一化 (normalization) 通常是指将数据缩放到 [0,1] 或 [-1,1] 的区间上。归一化可以通过线性变换、MinMaxScaler 等方法来实现。归一化可以使得不同特征的权重相同，避免某些特征对模型的影响过大，从而提高模型的准确性和泛化能力。

⚠ 很多文献混用 standardization 和 normalization，大家注意区分。

图 10、图 11 和图 12 分别展示的是经过标准化处理的鸢尾花数据的热图、KDE 分布曲线和平行坐标图。



《统计至简》一册讲过，**主成分分析** (Principal Component Analysis, PCA) 之前，一般会先对数据进行标准化。经过标准化后的数据，再求协方差矩阵，得到的实际上是原始数据的相关性系数矩阵。

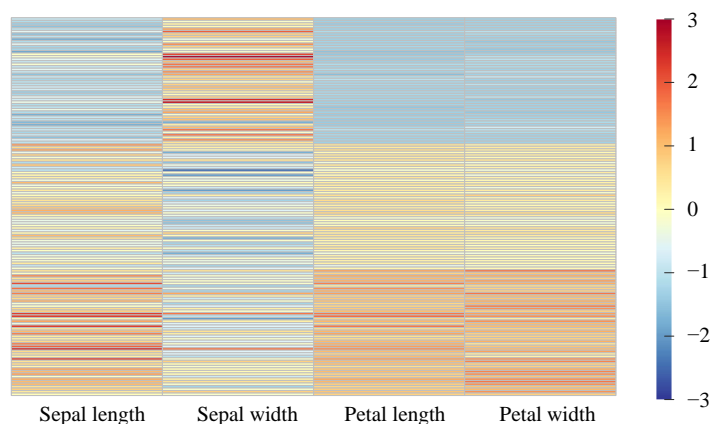


图 10. 热图，标准化

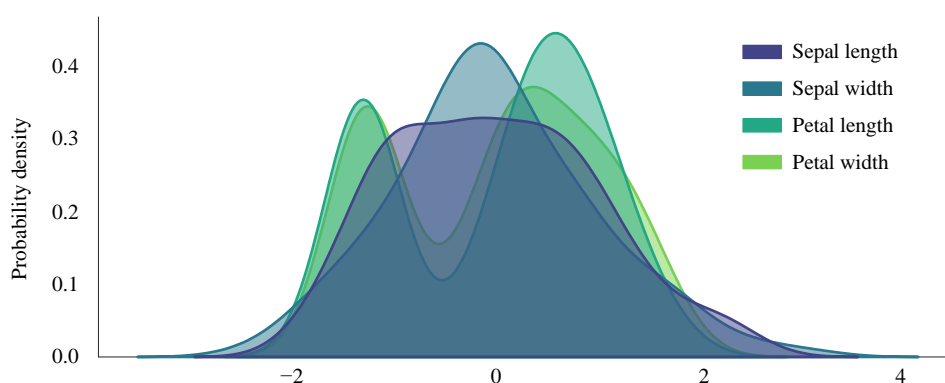


图 11. KDE 分布估计，标准化

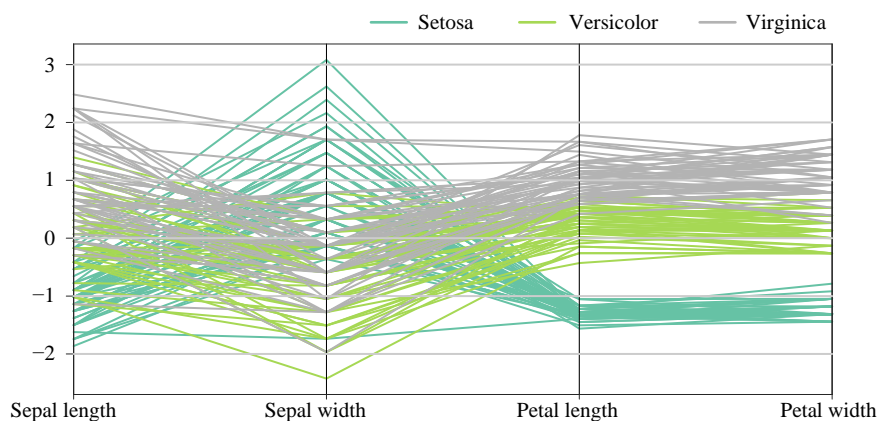


图 12. 平行坐标图，标准化

8.4 归一化：取值在 0 和 1 之间

归一化 (normalization) 常指数据首先减去其最小值，然后再除以 $\text{range}(X)$ ，即 $\max(X) - \min(X)$ ：

$$\frac{X - \min(X)}{\max(X) - \min(X)} \quad (3)$$

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

通过上式归一化得到的数据取值范围在 $[0, 1]$ 之间。

图 13、图 14 分别展示归一化鸢尾花数据的小提琴图和平行坐标图。

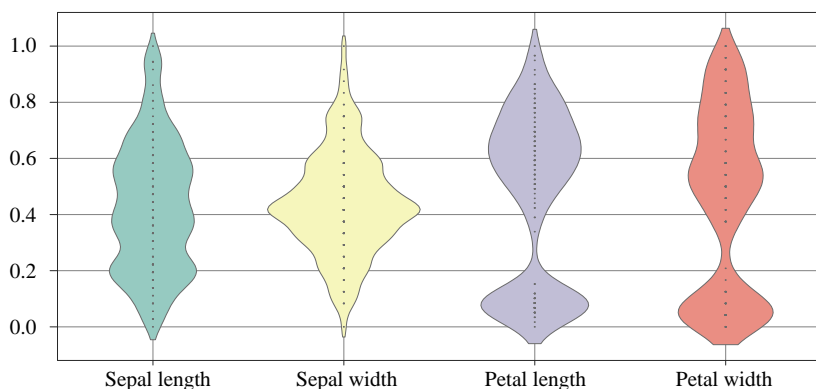


图 13. 小提琴图，归一化

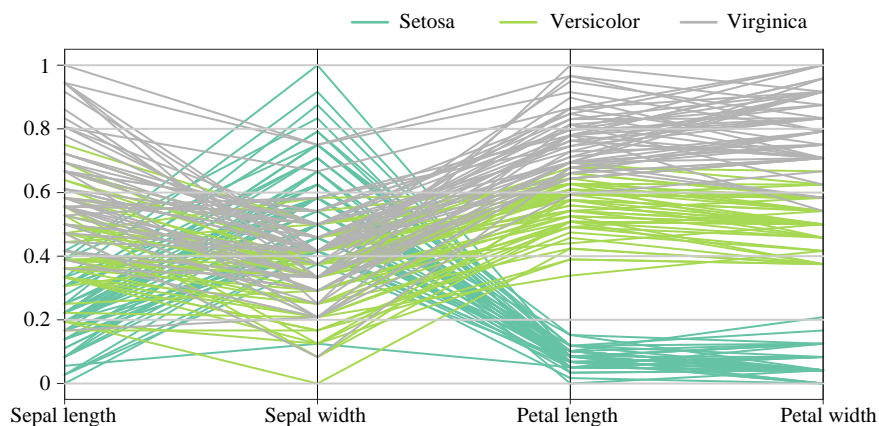


图 14. 平行坐标图，归一化

其他转换

另外一种类似归一化的数据转换方式，数据先去均值，然后再除以 $\text{range}(X)$ ：

$$\tilde{x} = \frac{x - \text{mean}(X)}{\max(X) - \min(X)} \quad (4)$$

这种数据处理的特点是，处理得到的数据取值范围约在 $[-0.5, 0.5]$ 之间。

还有一种数据转换使用箱型图的**四分位间距** (interquartile range) 作为分母，来缩放数据：

$$\frac{X - \text{mean}(X)}{IQR(X)} \quad (5)$$

其中 $IQR = Q_3 - Q_1$ 。



Bk6_Ch08_01.ipynb 绘制本章之前几乎所有图像。

8.5 广义幂转换

广义幂转换 (power transform), 也称 Box-Cox, 是一种用于对非正态分布数据进行转换的方法。Box-Cox 转换通过一系列参数 λ 的取值, 将数据的概率密度函数进行幂函数变换, 使得变换后的数据更加接近正态分布。

Box-Cox 转换可以通过最大似然估计或数据探索的方式来确定最优的 λ 值。Box-Cox 转换可以帮助我们改善非正态分布数据的统计性质, 如方差齐性、线性关系和偏度等, 从而提高模型的准确性和稳定性。Box-Cox 转换广泛应用于回归分析、时间序列分析、贝叶斯分析等领域。

Box-Cox 转换具体为:

$$x^{(\lambda)} = \begin{cases} \frac{x^\lambda - 1}{\lambda} & \lambda \neq 0 \\ \ln x & \lambda = 0 \end{cases} \quad (6)$$

其中, x 为原始数据, $x^{(\lambda)}$ 代表经过 Box-Cox 转换后的新数据, λ 为转换参数。

⚠ 注意, Box-Cox 转换要求参与转换的数据为正数。

在进行 Box-Cox 转换之前, 需要确保数据都是正数。如果数据包含负数或零, 可以先对数据进行平移或加上一个较小的正数, 使得数据都变成正数, 然后再进行 Box-Cox 转换。另外, 如果数据中存在较小的负数或零, 也可以考虑使用其他的转换方法, 如 Yeo-Johnson 转换, 它可以处理包含负数的数据。

实际上, Box-Cox 转换代表一系列转换。其中, $\lambda = 0.5$ 时, 叫平方根转换; $\lambda = 0$ 时, 叫对数转换; $\lambda = -1$ 时, 为倒数转换。大家观察上式可以发现, 它无非就是两个单调递增函数。

Box-Cox 转换通过优化 λ 参数, 让转换得到的新数据明显地展现出**正态性** (normality)。

正态性指的是一个随机变量服从高斯分布的特性。正态分布是一种常见的概率分布, 其概率密度函数呈钟形曲线, 具有单峰性、对称性和连续性。如果一个数据集或随机变量的分布近似于正态分布, 那么它就具有正态性, 也称为正态分布性。正态性在统计分析中非常重要, 因为很多经典的统计方法, 如 t 检验、方差分析等, 都基于正态分布的假设。如果数据不服从正态分布, 可能会影响到模型的可靠性和精度, 需要采取相应的数据预处理或选择适当的非参数方法。

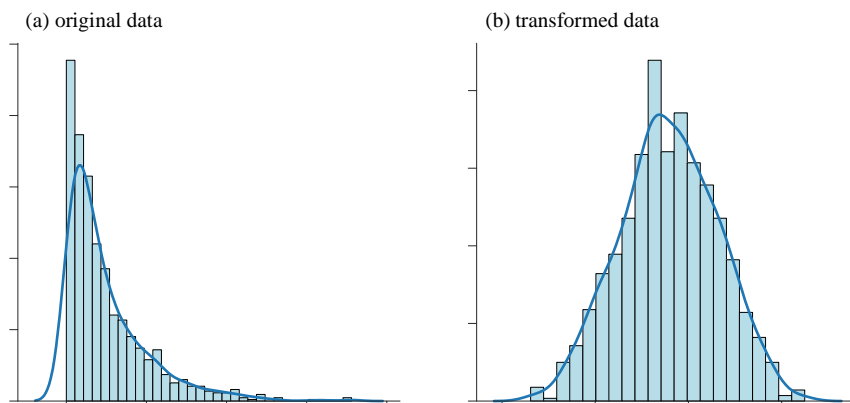


图 15. 原始数据和转换数据的直方图

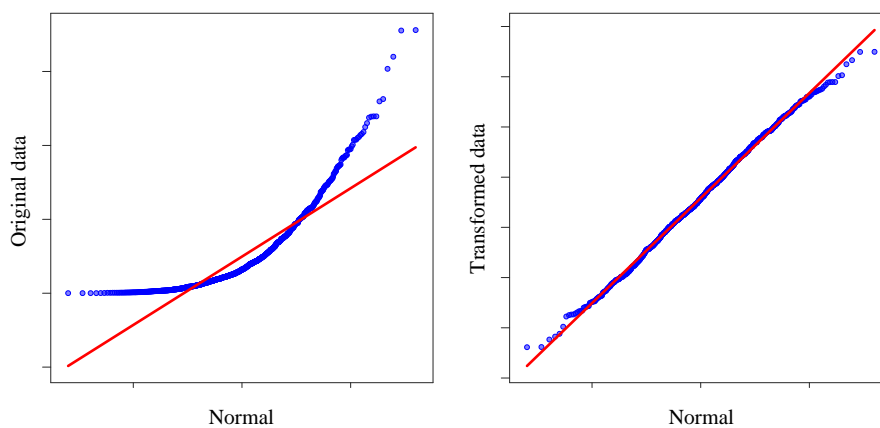


图 16. 原始数据和转换数据的 QQ 图

Yeo-Johnson 转换

前文提过 Yeo-Johnson 可以处理负值，具体数学工具为：

$$x^{(\lambda)} = \begin{cases} \frac{(x+1)^\lambda - 1}{\lambda} & \lambda \neq 0, x \geq 0 \\ \ln(x+1) & \lambda = 0, x \geq 0 \\ -\frac{((-x+1)^{2-\lambda} - 1)}{2-\lambda} & \lambda \neq 2, x < 0 \\ -\ln(-x+1) & \lambda = 2, x < 0 \end{cases} \quad (7)$$

Bk6_Ch08_01.ipynb 绘制图 15 和图 16。sklearn.preprocessing.PowerTransformer() 函数同时支持'yeo-johnson'和'box-cox'两种方法。下面聊聊 Bk6_Ch08_01.ipynb 中关键语句。

- a** 用 `numpy.random.exponential()` 生成一个包含 1000 个随机指数分布的数据样本，存储在 `original_X` 中。
- b** `scipy.stats` 中 `boxcox` 函数对 `original_X` 进行 Box-Cox 变换，将变换后的数据存储在 `new_X` 中，并返回变换的 Lambda 值 (`fitted_lambda`)，Lambda 值用于标识 Box-Cox 变换中的幂。
- c** 在第一个子图上绘制原始数据的直方图，包括核密度估计曲线。`kde=True` 表示同时显示核密度估计，`label="Original"` 用于图例标签。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

- d 在第二个子图上绘制变换后的数据的直方图，也包括核密度估计曲线。
- e 用 `scipy.stats.probplot` 用于生成 QQ 图，`dist=stats.norm` 表示使用正态分布作为比较对象，`plot=ax[0]` 表示在第一个子图上绘制。
- f 在第二个子图上绘制变换后的数据的 QQ 图。

```

a original_X = np.random.exponential(size = 1000)

# Box-Cox tpower transformation
b new_X, fitted_lambda = stats.boxcox(original_X)

# 直方图
fig, ax = plt.subplots(1, 2)

c sns.histplot(original_X,
               kde = True,
               label = "Original", ax = ax[0])

d sns.histplot(new_X,
               kde = True,
               label = "Original", ax = ax[1])

# QQ图
fig, ax = plt.subplots(1, 2)

e stats.probplot(original_X, dist=stats.norm, plot=ax[0])
ax[0].set_xlabel('Normal')
ax[0].set_ylabel('Original data')
ax[0].set_title('')

f stats.probplot(new_X, dist=stats.norm, plot=ax[1])
ax[1].set_xlabel('Normal')
ax[1].set_ylabel('Transformed data')
ax[1].set_title('')

```

代码 2. 广义幂转换 |  Bk6_Ch08_02.ipynb

8.6 经验累积分布函数

➡ 《统计至简》第 9 章一册提到，经验累积分布函数 (Empirical Cumulative Distribution Function, ECDF) 实际上也是一种重要的数据转换函数。ECDF 是一种非参数的数据转换方法。

ECDF 的特点是简单易懂，不需要对数据进行任何假设或参数估计，适用于任何类型的数据分布，包括连续型和离散型数据。通过将原始数据转换为概率分布函数，可以更好地理解数据的分布情况，并与理论分布进行比较，从而判断数据是否符合某种分布模型。

图 17 所示为样本数据和其经验累积分布的关系。

如图 18 所示， $u = \text{ECDF}(x)$ 代表经验累积分布函数；其中， x 为原始样本数值， u 为其 ECDF 值。 u 的取值范围为 $[0, 1]$ 。 $u = \text{ECDF}(x)$ 具有单调递增特性。

$u = \text{ECDF}(x)$ 对应 Scikit-learn 中的 `sklearn.preprocessing.QuantileTransformer()` 函数。

图 19 所示为鸢尾花数据四个特征的 ECDF 图像。

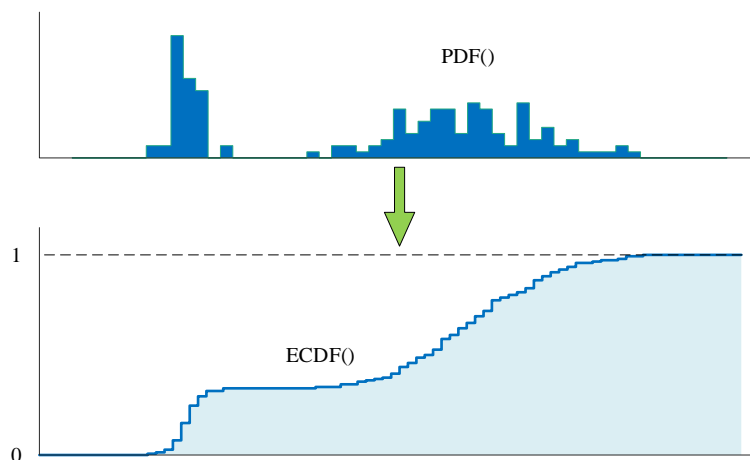


图 17. ECDF 函数转换样本数据

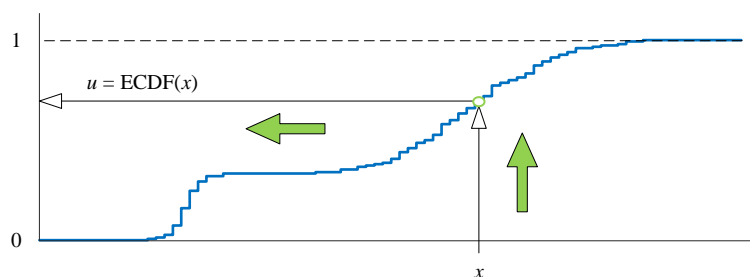


图 18. ECDF 函数原理

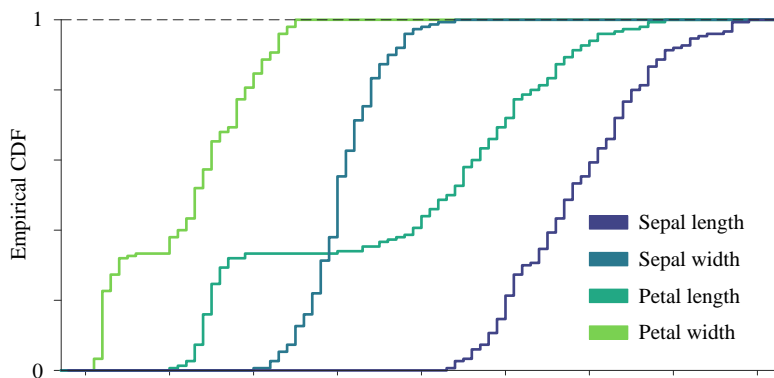


图 19. 鸢尾花数据四个特征的 ECDF

散点图

如图 19 所示，经过 ECDF 转换，鸢尾花四个特征的样本数据都变成了 $[0, 1]$ 区间的数据。这组数据肯定也有自己的分布特点。

图 20 所示为花萼长度、花萼宽度 ECDF 散点图和概率密度等高线。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

图 21 所示为鸢尾花数据 ECDF 的成对特征图。

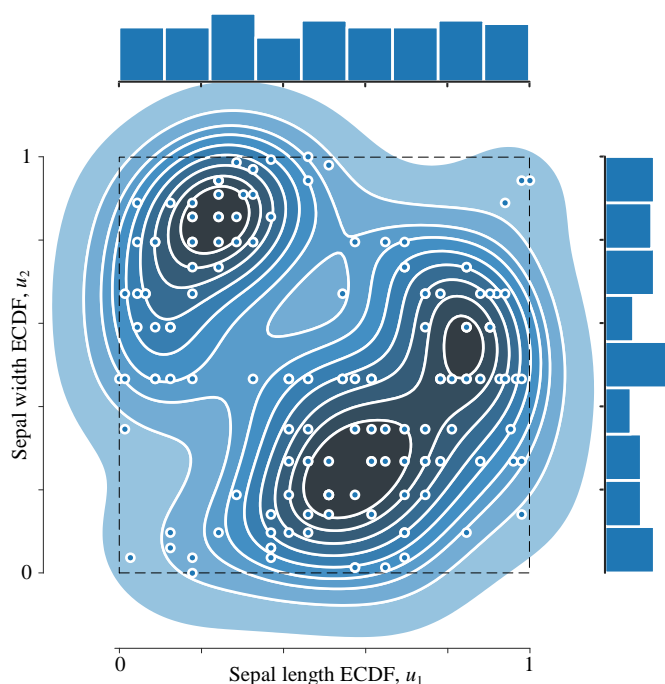


图 20. 鸢尾花花萼长度、花萼宽度 ECDF 散点图

容易发现 parametric (theoretical) CDF 和 empirical CDF 的取值范围都是 $[0, 1]$ ，而且是一一对应关系，这就是我们反复提到过的，CDF 曲线是很好的映射函数，可以将任意取值范围的数值映射到 $(0, 1)$ 区间，而且得到的具体数值有明确的含义，即累积概率值，可以解释。

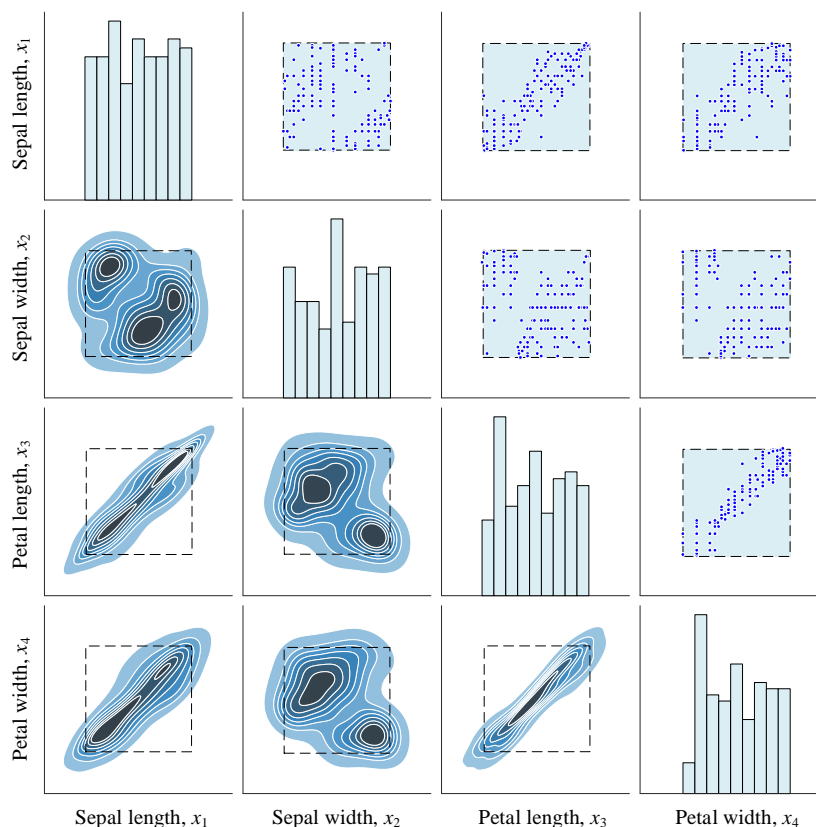


图 21. 鸢尾花数据 ECDF 的成对特征图

Bk6_Ch08_01.ipynb 绘制图 20 和图 21。下面聊聊其中关键语句。

- a** 从 `scikit-learn` 库中导入 `QuantileTransformer` 类，该类用于对数据进行分位数转换。
- b**：创建 `QuantileTransformer` 的实例 `qt`，并设置分位数的数量为数据集 `X_df` 的长度，`random_state` 为 `0` 是为了保证可重复性。
- c** 使用 `QuantileTransformer` 对数据集 `X_df` 进行拟合和转换，得到经过分位数转换的结果 `ecdf`。
- d** 将转换后的结果 `ecdf` 转换为 `DataFrame`，列名保持与原始数据集 `X_df` 一致。
- e** 使用 Seaborn 的 `jointplot` 函数创建一个二维联合图，其中横轴是 `feature_names[0]`，纵轴是 `feature_names[1]`，并限制轴的范围在 `[0, 1]` 之间。
- f** 在联合图上绘制核密度估计图，使用蓝色渐变颜色映射。`zorder=0` 表示将核密度估计放置在最底层，`levels=10` 表示绘制 10 个等高线，`fill=True` 表示填充核密度估计图的区域。


```

a from sklearn.preprocessing import QuantileTransformer
b qt = QuantileTransformer(n_quantiles=len(X_df),
                           random_state=0)
c ecdf = qt.fit_transform(X_df)
d ecdf_df = pd.DataFrame(ecdf,
                        columns = X_df.columns)

# 可视化
e g = sns.jointplot(data=ecdf_df, x=feature_names[0],
                   y=feature_names[1],
                   xlim = [0,1], ylim = [0,1])
f g.plot_joint(sns.kdeplot, cmap="Blues_r", zorder=0,
              levels=10, fill = True)

```

代码 3. 经验累积分布函数 |  Bk6_Ch08_03.ipynb

连接函数

大家肯定会问，有没有一种分布可以描述图 20、图 21 所示概率分布？答案是肯定的！

这就是**连接函数** (copula)。连接函数是一种描述**协同运动** (co-movement) 的方法。定义向量：

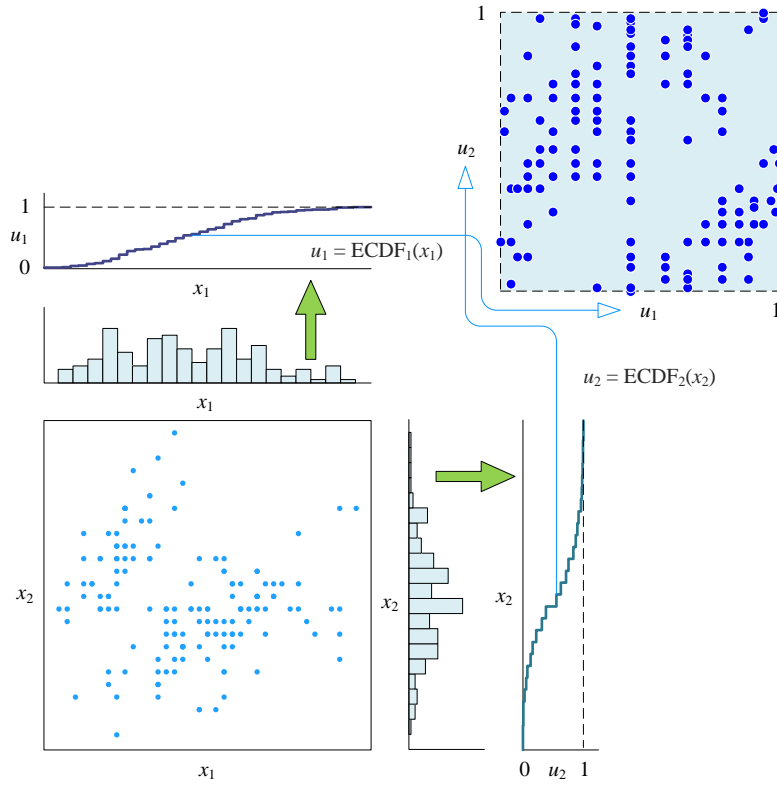
$$[x_1 \ x_2 \ \cdots \ x_D] \quad (8)$$

它们各自的边缘经验累积概率分布值可以构成如下向量：

$$[u_1 \ u_2 \ \cdots \ u_D] = [\text{ECDF}_1(x_1) \ \text{ECDF}_2(x_2) \ \cdots \ \text{ECDF}_D(x_D)] \quad (9)$$

其中 $u_j = \text{ECDF}_j(x_j)$ 为 X_j 的边缘累积概率分布函数， u_j 的取值范围为 $[0, 1]$ 。

图 22 所示为以二元为例展示原数据和 ECDF 的关系。

图 22. x_1 和 x_2 , 和 u_1 和 u_2 的关系

反方向来看 (9):

$$\begin{bmatrix} x_1 & x_2 & \cdots & x_D \end{bmatrix} = \begin{bmatrix} \text{ECDF}_1^{-1}(u_1) & \text{ECDF}_2^{-1}(u_2) & \cdots & \text{ECDF}_D^{-1}(u_D) \end{bmatrix} \quad (10)$$

其中, $x_j = \text{ECDF}_j^{-1}(u_j)$ 为**逆累积概率分布函数** (inverse empirical cumulative distribution function), 也就是累积概率分布函数 $u_j = \text{ECDF}_j(x_j)$ 的反函数。

连接函数 C 可以被定义为:

$$C(u_1, u_2, \dots, u_D) = \text{ECDF}(\text{ECDF}_1^{-1}(u_1), \text{ECDF}_2^{-1}(u_2), \dots, \text{ECDF}_D^{-1}(u_D)) \quad (11)$$

连接函数的概率密度函数, 也就是 copula PDF 可以通过下式求得:

$$c(u_1, u_2, \dots, u_D) = \frac{\partial^D}{\partial u_1 \cdot \partial u_2 \cdot \dots \cdot \partial u_D} C(u_1, u_2, \dots, u_D) \quad (12)$$

图 23 展示的是几种常见连接函数, 其中最常用的是**高斯连接函数** (Gaussian copula)。本书不做展开讲解, 请感兴趣的读者自行学习。

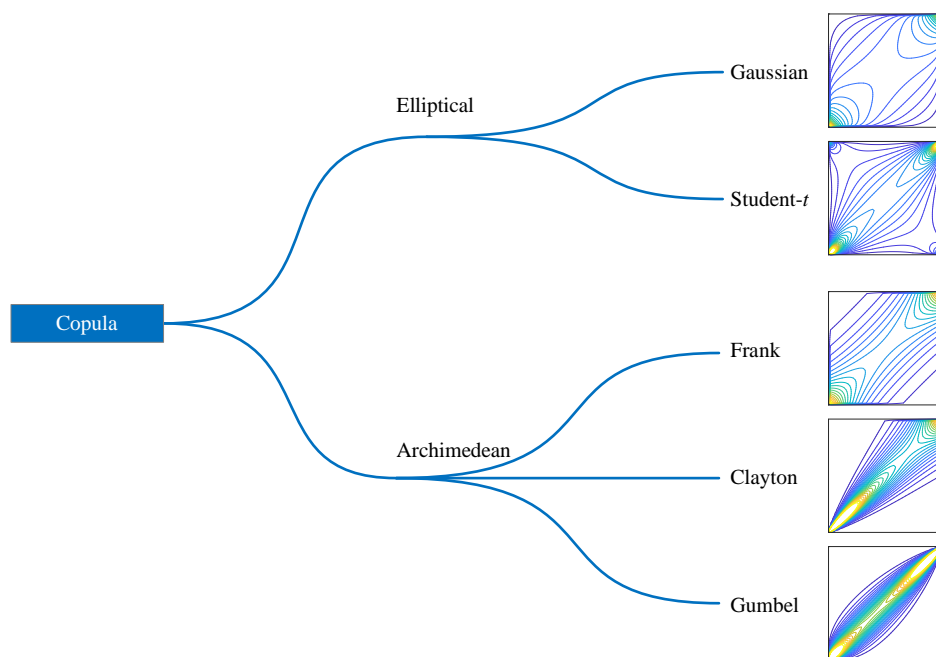


图 23. 常见连接函数

在机器学习中，数据转换是将原始数据进行处理或转换，以更好地适应模型的需求。常用的数据转换方法包括中心化、标准化、归一化、对数转换、指数转换和广义幂转换等方法。这些方法可以根据数据的分布特点、度量单位、取值范围和变量之间的关系进行选择和应用。

正确的数据转换可以提高模型的预测精度，从而提高模型的应用效果。然而，不同的数据转换方法可能对同一数据集产生不同效果，需要进行比较和评估。

如下网页专门介绍 Scikit-learn 预处理，请大家参考：

<https://scikit-learn.org/stable/modules/preprocessing.html>

此外，Scikit-learn 有大量的数据转换函数，请大家学习如下两例：

https://scikit-learn.org/stable/auto_examples/preprocessing/plot_all_scaling.html

https://scikit-learn.org/stable/auto_examples/preprocessing/plot_map_data_to_normal.html

Statsmodels 支持连接函数，请大家参考：

<https://www.statsmodels.org/dev/examples/notebooks/generated/copula.html>