5228 - Midterm Notes

Type of Attributes

Nominal（名义变量）：

- 定义: 这种数据类型只用于标识类别或分类，不能排序或计算数值差异。
- 例子: 性别（男、女）、颜色（红、绿、蓝）、国家（中国、美国、法国）。

Ordinal（有序变量）：

- 定义: 这种数据类型不仅可以分类，而且这些类别之间有明确的顺序关系，但不能量化类别之间的差异。
- 例子: 教育程度（小学、初中、高中、大学）、服务评级（差、一般、好）。

Interval（区间变量）：

- 定义: 区间变量是有顺序且可以量化差异的数据类型，但没有绝对零点，不能进行倍数比较。
- 例子: 温度（摄氏度、华氏度）、年份（1900年、2000年），我们可以计算两个值之间的差异，但不能说温度是“倍数”关系。

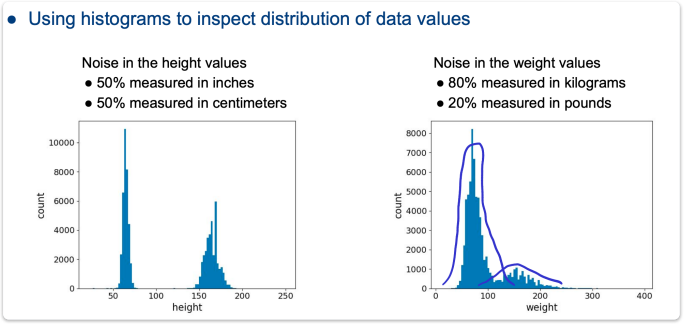
Ratio（比例变量）：

- 定义: 比例变量是有绝对零点的数据类型，不仅可以比较差异，还可以进行倍数的比较。
- 例子: 身高、体重、年龄、收入。这些数据有绝对的零点，可以说一个人重是另一个人的两倍。

Data Quality

- **Data = true signal + noise**
- Outlier: Data point with attribute values considerably different from other points
- 两种情况：
  - outlier是noise，需要移除
  - outlier是target(Credit card fraud)

EDA



Data Preprocessing

- Data Reduction:
  - 减少data的数量（sampling）
  - 减少attribute （remove irrelevant attribute, reduce dimension）
  - noise removal (Aggregation or generalization)
- Data transformation:
  - Normalization
  - One-Hot Encoding (将分类性的column转换成0，1，2之类的)

Clustering

- Goal of clustering:
  - Maximum intra-cluster similarity
  - Minimize inter-cluster similarity
- Similarity Measurement:
  - Euclidean Distance
  - Jaccard Similarity
  - Cosine Similarity
- Type of clusters
  - Well-separated:
    - 每个在cluster中的点到相同cluster的点的距离永远小于到其他cluster的点的距离
  - Center-based:
    - 每个在cluster中的点到相同cluster的中心点的距离永远小于到其他cluster的中心点的距离
    - Cluster center commonly called centroid
    - K-means is center based
  - Contiguity-based:
    - 基于邻近性的聚类方法通常依赖于点与点之间的距离或相邻关系来确定聚类。具体来说，它关注样本点之间的地理邻接性。
  - Density-based:
    - 基于密度的聚类方法通过寻找高密度区域来形成聚类，而低密度区域则被视为噪声或不同的聚类之间的边界。

Clustering Algorithms

K-means

K-means 算法步骤

1. **确定簇的数量 (K)**：
   - 选择要分成的簇数 (K)，这是用户需要手动设定的参数。
2. **初始化簇中心**：
   - 随机选择 (K) 个数据点作为初始的簇中心（centroids）。这些初始点可以从数据集中随机选取，或通过其他启发式方法（如 K-means++）选定，以更好地选择初始点。
3. **分配数据点到最近的簇**：
   - 对每个数据点，计算它与所有簇中心的距离（通常是欧氏距离），并将该数据点分配到与其距离最近的簇中。每个簇代表一组离簇中心最近的点。
4. **更新簇中心**：
   - 对于每个簇，计算该簇中所有点的均值，并将这个均值作为该簇的新簇中心（centroid）。即：
     [
     \mu*k = \frac{1}{|C_k|} \sum*{x \in C_k} x
     ]
     其中 (C_k) 表示簇 (k) 中的数据点集合，(|C_k|) 为簇中的数据点数。
5. **重复步骤 3 和 4**：
   - 重复分配数据点到簇和更新簇中心的步骤，直到簇中心不再发生变化，或者变化小于设定的阈值（收敛条件）。
6. **终止条件**：
   - 当簇中心不再移动，或者变化很小，算法收敛，K-means 终止。
   - 另一种终止条件是达到预设的最大迭代次数。
7. **输出结果**：
   - 最终的输出是 (K) 个簇，每个簇包含一组数据点，以及每个簇的中心。
     K-means 流程图总结：
8. 确定簇数 (K)。
9. 初始化 (K) 个随机簇中心。
10. 重复：

    - 分配数据点到最近的簇。
    - 计算簇的中心（均值）。
11. 直到簇中心不再变化或达到最大迭代次数。
12. 输出结果：K 个簇及其中心。

K-Means Varients

K-Means++:

- K-means++ 是对传统 K-means 算法的改进，主要在初始簇中心的选择上进行了优化。K-means 的一个主要问题是初始簇中心的选择会极大地影响聚类结果，而 K-means++ 提供了一种启发式的方法，使得初始簇中心选择更具代表性，减少聚类结果的不确定性。
    - 从数据集中随机选择一个点作为第一个簇中心。
    - 对于每个数据点 xi，计算它与最近的已选择簇中心的距离 D(xi)
    - 根据与现有簇中心的距离，以概率分布选择下一个簇中心。这意味着离现有簇中心越远的点更有可能被选为新的簇中心。
    - 重复以上两个步骤，指导选出K个簇中心
    - 使用这些初始化的簇中心，运行传统K-Means
- 优点：
  - 有效避免KMeans由于随机选择初始中心导致的局部最优问题
  - 提高收敛速度和聚类质量

  ### X-means：
- 从小的K值开始运行KMeans
- 对每个簇，尝试将其进一步划分为两个簇，计算划分前后的BIC或AIC分数
- 根据分数的变化，决定是否接受新的簇划分
- 重复该过程，直到找到最优的K值
- 优点：
    - 自动化确定聚类数量K
- 缺点：
  - 计算复杂度较高

  ### K-medoids

  K-medoids 是 K-means 的一种变体，区别在于它使用数据集中实际存在的点作为聚类中心，而不是使用均值计算得到的中心点。它通常通过最小化簇内点与中心点的绝对距离来进行聚类，因此对噪声和离群点更鲁棒。
  K-medoids 的主要流程：

1. 随机选择 KKK 个点作为初始的中心（称为 medoids）。
2. 分配每个数据点到最近的 medoid，形成簇。
3. 对每个簇，计算不同点作为 medoid 的代价，选择代价最小的点作为新的 medoid。
4. 重复分配和 medoid 更新的过程，直到收敛。
   优点：

- 对于离群点更鲁棒，因为使用实际的数据点作为簇中心，避免了离群点对均值的影响。
- 可以处理非球形分布的数据。
  缺点：
- 比 K-means 计算复杂度更高，尤其是在大数据集上，更新 medoid 的过程较慢。

AGNES

AGNES（Agglomerative Nesting）是一种层次聚类算法，其全名为“凝聚式层次聚类”（Agglomerative Hierarchical Clustering）。这是一个自底向上的聚类方法，主要用于无监督学习任务中的数据聚类分析。以下是AGNES的基本概念和工作原理：

1. 基本概念

AGNES是一种层次聚类算法，属于凝聚式方法。这意味着它一开始将每个数据点视为一个单独的簇（cluster），然后逐步将相似的簇合并，直到满足停止条件或者所有数据点都归为一个簇。

2. 工作原理

AGNES的工作过程可以概括为以下几个步骤：

1. **初始化**：将每个数据点视为一个单独的簇。因此，若有N个数据点，则初始化时有N个簇。
2. **计算距离**：根据某种距离度量（例如欧几里得距离、曼哈顿距离等），计算每对簇之间的相似度或距离。
3. **合并簇**：找到最相似的两个簇，将它们合并为一个簇。
4. **重复步骤2-3**：不断重复计算距离和合并簇的过程，直到达到某个停止条件，例如只剩一个簇或达到预定的簇数量。

3. 距离度量方法

在AGNES中，簇之间的相似度计算方式是关键因素，常见的距离度量方法包括：

- **最短距离（单链法，Single Linkage）**：两个簇中距离最近的两个数据点之间的距离。
- **最长距离（全链法，Complete Linkage）**：两个簇中距离最远的两个数据点之间的距离。
- **平均距离（Average Linkage）**：簇中所有数据点之间的平均距离。
- **质心距离（Centroid Distance）**：两个簇的质心之间的距离。

4. 特点

- **优点**：AGNES生成的层次结构可以通过树状图（Dendrogram）来可视化，这有助于理解数据的结构。此外，它不需要预先指定簇的数量。
- **缺点**：它的计算复杂度较高，尤其是在处理大规模数据集时，因为每一步都需要计算簇之间的距离。此外，AGNES的结果对噪声和离群点较为敏感。

Association Rule Mining

关联规则挖掘（Association Rule Mining）是一种用于**发现数据集中不同变量或项之间的有趣关系或模式**的技术。它常用于大规模数据分析中，尤其在购物篮分析（Market Basket Analysis）中非常常见。关联规则挖掘的目标是找到**频繁出现的项集**及其之间的关联关系，从而帮助揭示隐藏在数据中的模式。

1. 基本概念

关联规则挖掘的主要目标是从一个大型数据集中挖掘出形式为`A → B`的规则。这条规则的含义是：“如果事件A发生，那么事件B很可能也会发生”。比如，在购物篮分析中，规则`{牛奶} → {面包}`表示“如果顾客购买了牛奶，他们很有可能也会购买面包”。

2. 关联规则的组成部分

关联规则表示不同项集之间的条件关系，通常形式为`A → B`，意思是“如果出现A，通常也会出现B”。这类规则旨在揭示数据集中某些项之间的模式关联关系。关联规则通常有三个关键指标来衡量它们的有效性：

- **支持度（Support）**：某项集在总交易中出现的频率。表示的是规则在数据集中有多常见。
  Support(A→B) = 包含A和B的交易数 / 总交易数
  例如，如果有1000笔交易，其中100笔交易同时包含牛奶和面包，那么支持度为`100/1000 = 0.1`。
- **置信度（Confidence）**：在包含A的交易中，B出现的频率。衡量的是规则的可靠性。
  Confidence(A→B) = 包含A和B的交易数包含A的交易数 / 包含A的交易数
  如果有200笔交易包含牛奶，而其中100笔交易也包含面包，那么置信度为`100/200 = 0.5`，即50%的概率。

- **提升度（Lift）**：衡量A和B之间的相关性强度，值越大表示关联性越强。如果提升度大于1，则表明A和B之间有正相关性。
  Lift(A→B)=Confidence(A→B) / Support(B)
  如果提升度等于1，意味着A和B是独立的。

3. 关联规则挖掘的算法

常用的关联规则挖掘算法包括：

- **Apriori算法**：**priori算法**是一种经典的关联规则挖掘算法，通过减少不必要的项集计算，大幅提升效率。它的核心思想是利用**频繁项集的子集也必须是频繁项集**这一性质，来减少要处理的项集数量，从而提高效率。这种属性被称为“**Apriori原理**”。
  - 算法步骤如下：
    1. **生成频繁1项集**：首先扫描整个数据集，找到支持度大于或等于阈值的1项集
    2. **生成候选k项集**：在k-1项集基础上，合并生成候选的k项集。
    3. **剪枝**：根据Apriori原理，移除那些有子集不是频繁项集的候选项集。
    4. **计算支持度**：扫描数据集，计算每个候选项集的支持度，保留支持度大于或等于阈值的项集。
    5. **重复步骤2-4**：重复生成更高阶的频繁项集，直到没有新的频繁项集可以生成为止。
    6. **生成关联规则**：通过从频繁项集中生成所有可能的关联规则，并计算其置信度。
  - Apriori原理：如果一个项集是频繁的，那么它的所有子集也是频繁的。因此，若某个项集的子集不频繁，算法可以直接跳过该项集的进一步扩展，节省计算时间。
  - **优点**：通过“剪枝”（删除不频繁项集的子集），大大减少了需要处理的项集数量，提高了效率；能处理大规模数据集。
  - **缺点**：尽管比Brute-force高效，Apriori算法仍需要多次扫描数据库，尤其在大规模数据集下，仍有较高的计算成本；生成候选项集的过程在项集规模较大时会变得复杂。
- **Brute-Force算法**：一种直接且简单的方式，用于从数据集中生成关联规则。它的工作方式是通过**枚举所有可能的项集组合**，然后计算每个项集的支持度、置信度等度量指标，以此确定频繁项集和有效的关联规则。
  - 算法步骤如下：
    1. **生成所有可能的项集**：给定数据集`D`中的`n`个项，它会生成所有可能的项集。这意味着如果有`n`个项，可能的项集组合数是`2^n`，包括所有的子集。
    2. **计算每个项集的支持度**：对于每个项集，算法遍历数据集来计算其支持度（支持计数/总交易数）。
    3. **筛选频繁项集**：根据设定的支持度阈值，筛选出支持度大于或等于阈值的频繁项集。
    4. **生成关联规则**：通过从频繁项集中生成所有可能的关联规则，计算每条规则的置信度，保留置信度高于阈值的规则。
  - **优点**：概念简单，直接枚举所有可能的组合，容易实现。
  - **缺点**：计算效率极低，尤其在数据集较大时，可能的项集数量随着项的数量呈指数增长，导致计算成本过高。因此，Brute-force方法在实际应用中并不常用。

5. 实际例子

假设有以下四笔交易记录：
交易1: {牛奶, 面包}
交易2: {牛奶, 啤酒}
交易3: {牛奶, 面包, 黄油}
交易4: {面包, 黄油}

- **Brute-force算法**会枚举所有可能的项集组合，如`{牛奶}`，`{面包}`，`{牛奶, 面包}`，`{牛奶, 啤酒}`，直到`{牛奶, 面包, 黄油}`，然后计算支持度和置信度。
- **Apriori算法**首先找出所有的1项集，如`{牛奶}`，`{面包}`，`{啤酒}`，`{黄油}`，然后生成频繁的2项集（如`{牛奶, 面包}`），通过剪枝减少生成非频繁的项集，直到最终找到所有频繁项集。

定义

- Itemset: 项集是指一个数据集中一组共同出现的物品（项）的集合。每个项集可以包含一个或多个物品。例如，在购物篮分析中，`{牛奶，面包}`就是一个项集，表示顾客同时购买了牛奶和面包。
- K-Itemset: k项集表示包含k个项的项集。也就是说，项集中的项的数量是k。例如:
  - `{牛奶}` 是一个1项集（1-itemset）。
  - `{牛奶，面包}` 是一个2项集（2-itemset），因为它包含2个项。
  - `{牛奶，面包，黄油}` 是一个3项集（3-itemset），包含3个项。
- Support count: 支持计数指的是在数据集中某个项集出现的次数。比如，如果在1000笔交易中有100笔交易包含了`{牛奶，面包}`，那么这个项集的支持计数为100。
- Support: 支持度衡量某个项集在总交易中出现的频率，它是支持计数相对于总交易数的比率。support(X) = 支持计数(X) / 总交易数
- Frequent itemset: 频繁项集是指支持度高于某个预设阈值的项集。阈值可以是用户设定的，表示对频繁模式的定义。例如，如果支持度阈值设定为5%，那么所有支持度高于5%的项集被视为频繁项集。
  - 举例：假设阈值为0.05（即5%），如果`{牛奶，面包}`的支持度为0.1，它就是一个频繁项集。

Classification & Regression

分类（Classification）和回归（Regression）是监督学习中两种常见的机器学习任务，它们之间的主要区别在于输出变量的类型和任务的目标。以下是它们的详细区别：

1. 输出变量的类型

- 分类：分类任务的输出是离散的，通常是有限个类别中的某一个。模型的目标是将输入数据分类到预定义的类别中。例如，给定一张图片，分类模型可能输出“猫”、“狗”或“鸟”这样的类别。
  - 例子：预测一封电子邮件是“垃圾邮件”还是“正常邮件”（二元分类），或预测一张图片中的动物是“猫”、“狗”还是“兔子”（多类分类）。
- 回归：回归任务的输出是连续的，通常是实数。模型的目标是预测一个数值输出。例如，给定一个人的房间面积、位置等信息，回归模型可以预测房价。
  - 例子：预测股票价格、房屋价格或天气温度等。

2. 任务目标

- 分类：任务的目标是找到一个决策边界，将数据划分到不同的类别中。模型的训练过程旨在最小化分类错误，即尽量正确地将每个输入样本分到正确的类别。
  - 常见的算法：逻辑回归（Logistic Regression）、支持向量机（SVM）、K近邻（K-NN）、决策树、随机森林、神经网络等。
- 回归：任务的目标是预测一个连续值，模型的训练过程旨在最小化预测值与真实值之间的误差（例如，最小化均方误差）。回归模型关心的是输出数值的准确性。
  - 常见的算法：线性回归（Linear Regression）、多元回归、岭回归（Ridge Regression）、Lasso回归、支持向量回归（SVR）、决策树回归等。

3. 评估指标

- 分类：评估分类模型的常用指标有准确率（Accuracy）、精确率（Precision）、召回率（Recall）、F1分数（F1 Score）和ROC-AUC曲线等。这些指标用于衡量模型在分类任务中的表现如何。
  - 例子：在垃圾邮件分类中，准确率用于衡量正确分类的比例。
- 回归：评估回归模型的常用指标包括均方误差（Mean Squared Error, MSE）、均绝对误差（Mean Absolute Error, MAE）、R^2（决定系数）等。这些指标用于衡量模型在数值预测任务中的误差大小。
  - 例子：在房价预测中，均方误差可以用于衡量预测值与真实值的差异。

4. 模型应用的场景

- 分类：分类模型常用于图像识别、文本分类（如垃圾邮件识别）、疾病诊断、金融风险评估等任务。
  - 示例场景：诊断一位病人是否患有某种疾病（是或否），识别一张图片中的物体是猫还是狗。
- 回归：回归模型常用于价格预测、销售预测、温度预测、需求预测等任务。
  - 示例场景：根据房子的特征（位置、面积等）预测房价。

5. 模型输出形式

- 分类：模型的输出是一个类别标签。例如，“猫”或“狗”。某些分类模型也可以输出每个类别的概率。例如，一个二分类模型可能输出“猫的概率为0.7，狗的概率为0.3”，然后将输入样本归类为猫。
- 回归：模型的输出是一个实数值。例如，给定房屋的特征，模型可能输出房屋价格为“100万”。

混淆矩阵

混淆矩阵（Confusion Matrix） 是一种用于评估分类模型性能的工具，特别适用于二分类或多分类任务。它能够帮助我们直观地了解模型在不同类别上的预测结果分布，从而进一步计算准确率、精确率、召回率等指标。
混淆矩阵的每一行表示模型的实际类别，每一列表示模型的预测类别。对于二分类问题，混淆矩阵通常是一个2x2的矩阵，结构如下：

	预测为正类	预测为负类
实际正类	TP（真正例）	FN（假负例）
实际负类	FP（假正例）	TN（真负例）

1. 混淆矩阵中的四个元素

- TP（True Positive，真正例）：模型正确预测为正类的样本数。例如，实际为正类（例如“癌症”）的病人被正确预测为正类。
- TN（True Negative，真负例）：模型正确预测为负类的样本数。例如，实际为负类（例如“没有癌症”）的病人被正确预测为负类。
- FP（False Positive，假正例）：模型错误地将负类样本预测为正类的样本数（又叫“I型错误”或“误报”）。例如，实际没有癌症的病人被错误预测为有癌症。
- FN（False Negative，假负例）：模型错误地将正类样本预测为负类的样本数（又叫“II型错误”或“漏报”）。例如，实际患有癌症的病人被错误预测为没有癌症。

2. 常用指标的计算

通过混淆矩阵中的四个元素，通常可以计算出以下几个常见的分类性能评估指标：

1. 准确率（Accuracy）

准确率衡量的是模型的整体分类正确率，即模型预测正确的样本所占的比例。公式如下：

Accuracy= (TP+TN) / (TP+TN+FP+FN)

准确率适用于数据分布相对平衡的情况。

2. 精确率（Precision）

精确率衡量的是模型预测为正类的样本中，实际为正类的比例。公式如下：

Precision=TP / (TP+FN)

精确率高表示模型在预测为正类时，错误较少，适用于关注误报（FP）较少的场景。

3. 召回率（Recall）

召回率衡量的是实际正类的样本中，被正确预测为正类的比例。公式如下：

Recall=TP / (TP+FN)

召回率高表示模型在正类样本中，漏报较少，适用于关注漏报（FN）较少的场景。

4. F1分数（F1 Score）

F1分数是精确率和召回率的调和平均数，用来综合衡量模型的精确率和召回率。公式如下：

F1 Score=2× ((Precision×Recall) / (Precision+Recall))

F1分数适用于数据不平衡的情况，能够平衡精确率和召回率。

5. 特异性（Specificity）

特异性衡量的是实际负类样本中，被正确预测为负类的比例。公式如下：

Specificity=TN / (TN+FP)

特异性高表示模型对负类样本的区分能力较好。

3. 混淆矩阵举例

假设有一个二分类问题，分类任务是判断某个病人是否患有某种疾病。现在有一个包含20个病人的测试数据集，模型的预测结果如下：

- 10个病人实际患病，其中8个被正确预测为患病（TP = 8），2个被错误预测为未患病（FN = 2）。
- 10个病人实际未患病，其中7个被正确预测为未患病（TN = 7），3个被错误预测为患病（FP = 3）。

混淆矩阵表示如下：

	预测患病	预测未患病
实际患病	8（TP）	2（FN）
实际未患病	3（FP）	7（TN）

- 准确率 = (8 + 7) / (8 + 7 + 3 + 2) = 15 / 20 = 0.75
- 精确率 = 8 / (8 + 3) = 8 / 11 ≈ 0.727
- 召回率 = 8 / (8 + 2) = 8 / 10 = 0.8
- F1分数 = 2 *(0.727* 0.8) / (0.727 + 0.8) ≈ 0.761

KNN

K近邻算法（K-Nearest Neighbors, KNN）是一种基于实例的非参数监督学习算法，常用于分类和回归任务。KNN的核心思想是，通过测量与新样本在特征空间中的距离，找到与新样本最近的K个邻居，基于这些邻居的类别或数值来进行预测。以下是KNN算法的详细解释：

1. KNN算法的核心思想

KNN算法的关键概念是相似性。它假设相似的数据点可能属于同一类或者具有相似的输出。给定一个未标记的样本，KNN通过以下步骤来进行预测：

1. 计算距离：根据某种距离度量方法（如欧几里得距离、曼哈顿距离等），计算该样本与训练集中所有数据点的距离。
2. 选择K个最近的邻居：选择距离最近的K个数据点作为邻居。
3. 投票或加权平均：

- **分类**：K个邻居中占多数类别的标签被分配给新样本，通常称为“多票表决”。
- **回归**：K个邻居的输出值取平均值，作为新样本的预测值。

2. K值的选择

- **K的大小**是KNN算法的一个重要超参数，它决定了参与预测的邻居数量。
  - 如果K值过小（如K=1），算法可能会对噪声数据非常敏感，导致过拟合。
  - 如果K值过大，算法可能会忽略局部模式，导致预测过于平滑（欠拟合）。
  - 通常，K值需要通过交叉验证或实验来选择，确保模型在不同K值下的表现最优。

3. 距离度量

KNN算法的基础在于**距离计算**，常用的距离度量方法包括：

- **欧几里得距离（Euclidean Distance）**：
  适用于连续数值型数据，衡量两个点在特征空间中的直线距离。
- **曼哈顿距离（Manhattan Distance）**：
  测量的是两个点在坐标轴上移动的总距离。
- **闵可夫斯基距离（Minkowski Distance）**：欧几里得距离和曼哈顿距离的推广形式，可以根据不同的p值调整。当p=2时，它变为欧几里得距离，当p=1时，它变为曼哈顿距离。

4. KNN的分类和回归

- **KNN分类**：在KNN分类任务中，算法找到K个最近的邻居，并根据邻居中的多数类别为新样本分配类别。例如，如果K=5，且其中3个邻居属于类别A，2个属于类别B，则该样本将被预测为类别A。
- **KNN回归**：在KNN回归任务中，算法找到K个最近的邻居，并取这些邻居的平均值或加权平均值作为新样本的预测值。

5. KNN的优缺点

优点：

1. **简单易懂**：KNN算法非常直观，且实现容易，不需要进行复杂的模型训练。
2. **无需假设数据分布**：KNN算法是非参数模型，意味着它不需要假设数据具有某种特定的分布（例如线性或高斯分布），适用于任意数据分布。
3. **适用于多分类问题**：KNN能够处理二分类和多分类问题，而且在数据量不大时效果不错。

缺点：

1. **计算成本高**：KNN是懒惰学习算法，在预测阶段需要遍历整个训练集来计算距离，特别在数据集较大时，计算效率较低。
2. **内存需求大**：因为KNN不存储模型，它需要在内存中存储所有训练数据，这对大规模数据集可能会带来问题。
3. **对数据归一化敏感**：KNN依赖于距离计算，不同特征的量纲可能会对距离产生显著影响，因此在使用KNN时通常需要对特征进行归一化或标准化处理。
4. **对噪声敏感**：KNN对异常值（噪声）敏感，尤其是当K值较小时，少量的噪声可能会严重影响预测结果。

6. KNN的应用场景

- **图像分类**：KNN常用于简单的图像分类任务，例如手写数字识别。
- **文本分类**：在自然语言处理中，KNN可以用于文本的分类任务。
- **推荐系统**：在协同过滤推荐系统中，KNN可以用于计算用户或物品的相似性，进而推荐相关内容。
- **金融预测**：KNN可用于金融领域的回归任务，如股票价格预测等。

7. KNN算法的工作流程示例

假设有一个简单的二分类问题，我们的目标是预测某个新样本属于哪一类，数据集如下：
训练数据：
样本1：[2, 4] 类别：A
样本2：[4, 6] 类别：A
样本3：[4, 2] 类别：B
样本4：[6, 4] 类别：B

测试数据：
样本：[5, 5] 类别：未知

步骤：

1. 计算测试样本与每个训练样本的欧几里得距离：
   - 与样本1的距离：$\sqrt{((5-2)^2 + (5-4)^2)} = \sqrt{(9 + 1)} = \sqrt{10} \approx 3.16$
   - 与样本2的距离：$\sqrt{((5-4)^2 + (5-6)^2)} = \sqrt{(1 + 1)} = \sqrt{2} \approx 1.41$
   - 与样本3的距离：$\sqrt{((5-4)^2 + (5-2)^2)} = \sqrt{(1 + 9)} = \sqrt{10} \approx 3.16$
   - 与样本4的距离：$\sqrt{((5-6)^2 + (5-4)^2)} = \sqrt{(1 + 1)} = \sqrt{2} \approx 1.41$
2. 设定K=3，选择距离最近的3个邻居，它们的类别分别是A、B、B。
3. 进行多数投票：类别B出现2次，类别A出现1次，所以新样本被分类为类别B。