实验目标

1. 任务：对包含许多样本的数据集，将相似的样本进行合理聚类，将样本集化成多个簇，每个簇内的样本相似度较高
    1. 实现聚类算法，实现上述的聚类
    2. 找到合适的评估标准，对聚类结果进行评估
2. 编程语言：$Python\ 3$
3. 使用库：$pandas$，$numpy$
4. 数据集：KDD Cup 1999 Data（KDD Cup 1999 Data (uci.edu)）

   https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

算法设计

step1: 数据预处理

1. 使用 `read_csv` 读入数据并分出特征矩阵X和标签y.
2. 把特征矩阵X的非数值列进行编码.
3. 将矩阵X归一化处理.
4. 把y的标签归为4大类.
5. 取出60%的数据作为训练集，剩下40%作为验证集.
6. 由于数据分布不均衡，对训练集过采样.

step2: 聚类算法

参数

```
:param n_clusters: 聚类簇数
:param max_iter: 最大迭代次数
:param eps: 精度
:param distance: 距离计算方式
:param random_state: 随机种子
:param Mu: 聚类中心
:param label: 聚类标签
```

拟合函数

输入: 样本集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
必连约束集合 $\mathcal{M}$;
勿连约束集合 $\mathcal{C}$;
聚类簇数 k.

过程:

从 D 中随机选取 k 个样本作为初始均值向量$\{\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\ldots,\boldsymbol{\mu}_k\}$;
repeat
  $C_j=\varnothing\ (1\leqslant j\leqslant k)$;
  for $i=1,2,\ldots,m$ **do**
    计算样本 $\boldsymbol{x}_i$ 与各均值向量 $\boldsymbol{\mu}_j\ (1\leqslant j\leqslant k)$ 的距离: $d_{ij}=\|\boldsymbol{x}_i-\boldsymbol{\mu}_j\|_2$;
    $\mathcal{K}=\{1,2,\ldots,k\}$;
    is_merged=false;
    while ¬ is_merged **do**
      基于 $\mathcal{K}$ 找出与样本 $\boldsymbol{x}_i$ 距离最近的簇: $r=\arg\min_{j\in\mathcal{K}} d_{ij}$;
      检测将 $\boldsymbol{x}_i$ 划入聚类簇 C_r 是否会违背 $\mathcal{M}$ 与 $\mathcal{C}$ 中的约束;
      if ¬ is_voilated **then**
        $C_r=C_r\bigcup\{\boldsymbol{x}_i\}$;
        is_merged=true
      else
        $\mathcal{K}=\mathcal{K}\setminus\{r\}$;
        if $\mathcal{K}=\varnothing$ **then**
          break并返回错误提示
        end if
      end if
    end while
  end for
  for $j=1,2,\ldots,k$ **do**
    $\boldsymbol{\mu}_j=\frac{1}{|C_j|}\sum_{\boldsymbol{x}\in C_j}\boldsymbol{x}$;
  end for
until 均值向量均未更新

输出: 簇划分 $\{C_1,C_2,\ldots,C_k\}$

部分代码如下

```
def fit(self, X, y):
    np.random.seed(self.random_state)
    Mu = X[np.random.choice(
        X.shape[0], self.n_clusters, replace=False
    )]
    mp = {
        y.unique()[i]: i
        for i in range(y.unique().shape[0])
    }
    mp.update({
        i: y.unique()[i]
        for i in range(y.unique().shape[0])
    })
    label_ = np.array([
        np.array([0] * y.unique().shape[0])
```

```
            for _ in range(self.n_clusters)
        ])
        for n_iter in range(self.max_iter):
            Mu_copy = Mu.copy()
            C = [[] for _ in range(self.n_clusters)]
            label = label_.copy()

            for i in range(X.shape[0]):
                dist = np.array([self.distance(X[i], mu) for mu in Mu])
                for _ in range(self.n_clusters):
                    r = np.argmin(dist)
                    if not self.violated(mp[y[i]], r, label):
                        C[r].append(i)
                        label[r][mp[y[i]]] += 1
                        break
                    else:
                        dist[r] = np.inf
            print(f'iter: {n_iter + 1}')
            print(label)
            for j in range(self.n_clusters):
                Mu[j] = np.mean(X[C[j]], axis=0)
            if np.all(np.abs(Mu_copy - Mu) < self.eps):
                break

        self.label = np.array(
            [mp[np.random.choice(np.where(s == s.max())[0], 1)[0]] for s in
label]
        )
        self.Mu = Mu
```

Step3: 预测

计算并选出距离最近的聚类中心点, 预测为该中心点的聚类簇的标签众数.

注：尝试过依概率预测, 但是结果并不好, 可能是数据分布不平衡导致的.

模型评价与存取

模型使用在验证集上分类正确的占比作为评价指标.

模型将类的字典使用 `DataFrame` 保存到 *model.csv* 文件.

模型读取 *.csv* 文件后使用 `eval` 函数返回给类的属性.

参考实现

```
    def to_DataFrame(self):
        kv = self.__dict__.copy()
        kv['distance'] = self.distance.__name__
        kv['Mu'] = kv['Mu'].tolist()
        kv['label'] = kv['label'].tolist()
        return pd.DataFrame([kv])

    def saveModel(self):
        self.to_DataFrame().to_csv('Models/cluster.csv', index=False)

    def loadModel(self):
        with open(f'Models/cluster.csv', 'r') as f:
            kv = pd.read_csv(f).iloc[0].to_dict()
        self.__init__(**kv)
```

一些说明

1. 本实验的距离模型使用 $Euclidean$ 距离 $d_{x_1,x_2} = ||x_1 - x_2||_2$
2. 本实验的归一化采用 $Min\text{-}Max$ 方式 $\frac{x-min(x)}{max(x)-min(x)}$. 本实验为减少过小的数运算带来的精度问题，遂对前式乘上10^6常数.
3. 由于数据集中的数据分布极其不平衡（如下图），故本实验采取过采样的方式（即把少数类重复取出），使得类与类之间的数目差异不太悬殊.

超参数的选取

本实验的超参数主要有 聚类簇数 n 与 随机种子 $random\_state$.

由于笔者实验时数据基本都能在15次左右的迭代后收敛，遂不对$iter$展开讨论.

$n = [5, 6, 7, 8, 9, 10, 11, 12]$.

$random\_state = [18, 24, 36, 42, 47]$.

笔者对$n \times random_state$共计40种方案进行尝试，最终得出在$n = 6, random_state = 24$时具有最好的表现.

参考文献

[1]. 周志华.《机器学习》. 2016. 清华大学出版社.