实验三 基于单高斯模型的二分类

PB19071509 王瑞哲

>>> 实验目标

用多元高斯模型解决二分类问题。假设两类有相同的先验概率。每个观测样本的特征是3维向量。基于训练数据 `Train.txt` 创建单高斯模型，然后用估计的模型对测试数据进行分类 `Test.txt`

>>> 实验原理

多元高斯分布：

$$G(\vec{x}|\vec{\mu},\Sigma)=\frac{1}{(2\pi)^{d/2}\det(\Sigma)^{1/2}}\exp\left(-\frac{1}{2}(\vec{x}-\vec{\mu})^T\Sigma^{-1}(\vec{x}-\vec{\mu})\right)$$

其中$\vec{x}$为d维随机向量

根据最大似然估计，利用多个样本来估计该多元高斯分布概率分布时，需知道其均值$\vec{\mu}$和协方差矩阵Σ。其公式分别为：

$$\vec{\mu}=\frac{1}{m}\sum_{i=1}^{m}\vec{x}_i$$

$$\Sigma=\frac{1}{m-1}\sum_{i=1}^{m}(\vec{x}-\vec{\mu})(\vec{x}-\vec{\mu})^T$$

>>> 实验过程

导入必要的包；定义导入训练、测试数据的函数

In [ ]:

```
import numpy as np
import re

def dataloader(filename):
    f = open(filename, "r")
    A_Set = []
    B_Set = []
    for line in f.readlines():
        cls = 1 if line[0]=='A' else 0
        data = re.search(r"(?<=\().*(?=\))", line).group()  # Using regular expressio
        if cls:
            A_Set.append(data.split(' '))
        else:
            B_Set.append(data.split(' '))
    # assert input matrix: [dimension*num_samples] every col represents a sample
    A_Set = np.array(A_Set).astype(float).T     # 1.Change list to np.array; 2.Change
    B_Set = np.array(B_Set).astype(float).T
    return A_Set, B_Set
```

定义一个多元高斯分布模型类，该类接收训练数据，计算其均值与方差的最大似然估计，以此生

成模型，并可以根据输入数据来计算模型计算结果

In [ ]:
```
from scipy.stats import multivariate_normal

class Gaussian_model():
    def __init__(self) -> None:
        self.mu = 0
        self.sigma = 1
        self.dimention = 1

    def getModel(self, trainData):
        # assert input matrix: [dimension*num_samples] every col represents a sample
        self.mu = np.mean(trainData, axis=1)
        self.sigma = np.cov(trainData)
        self.dimention = len(self.mu)
        return self

    def forward(self, input):
        assert len(input)==self.dimention, "Dimension dismatching error! This is a {}
        model = multivariate_normal(self.mu, self.sigma)        # Fuction for multiva
        return model.pdf(input)
```

定义评估函数，来判断训练得到的结果在测试集上的准确率

In [ ]:
```
def evaluation(modelA, modelB, filename):
    Test_A, Test_B = dataloader(filename)
    acc_num = 0
    for i in Test_A.T:      # switch set to [num_samples*dimension] for convinence
        if modelA.forward(i) >= modelB.forward(i):
            acc_num += 1
    for i in Test_B.T:
        if modelB.forward(i) >= modelA.forward(i):
            acc_num += 1
    acc = acc_num / (Test_A.shape[1] + Test_B.shape[1])
    return acc
```

正式开始训练与推断

In [ ]:
```
# train
Train_set_A, Train_set_B = dataloader("Train.txt")
G_A = Gaussian_model()
G_B = Gaussian_model()
G_A = G_A.getModel(Train_set_A)
G_B = G_B.getModel(Train_set_B)
# infer
acc = evaluation(G_A, G_B, "Test.txt")
print("The acc of this Gaussian Model is {:.3f}%".format(100*acc))
```

```
The acc of this Gaussian Model is 75.663%
```

>>> 实验总结

通过本次实验，我了解了多元高斯模型的基本理论基础，学会了利用代码编写训练单个多元高斯模型的过程，并将其应用于模型评测中

除此之外，本次代码中多采用类、函数来实现功能，这为接下来将本次实验所定义的内容转移到下一次实验多个多元高斯模型将有很大帮助。

同时注意到，本次实验的准确率预测结果约为75%，还有进步的空间，这一点可能会在下一次混合高斯模型的实验中所体现。