Assignment 1

17307130019 张孝亮

Part I

本部分需要设计三个高斯分布，考虑后续进行不同线性模型的分类，为了实验过程的直观以及可视化可易于理解的考虑，生成标签为 A、B、C 的二维高斯分布。

生成二维高斯分布使用 numpy 库中自带的随机函数，如下：

numpy.random.multivariate_normal(mean, cov, (size..))

所以目前主要可控制的参数就是分布的期望 mean 以及协方差矩阵 cov，由于目前没有参考，所以在设计完成分类模型后进行调整。此外由于采样大小的限制（不超过 20Kb），所以对每一个分布进行 150 次采样（19Kb 大小），结构为（标签，分布向量）的元组结构，一行为一个样本。并且计划使用其中 120 次（80%）进行训练，30 次（20%）作为测试样本。

实际在实验中，直接调用 generate_gaussian_data(NA，NB，NC)就可以直接生成.data 数据集文件，其中参数分别是期望的三个数据集的规模大小，文件中每一行的纪录为：label x1 x2。同样调用 process_gaussian_data()可以将.data 文件解析并且划分为三个训练集 A，B，C 以及测试集 T 和 T 中每个元素对应的标签数组 label，为了方便，顺便返回测试集中每个标签元素的数量。

Part II

线性判别模型：

目标应该是完成三类别的区分，但是先考量两类别分类问题，首先采用 Fisher Linear Discriminant 方法，分析如下:

假设有一组 n 个的 d 维样本 $X(x_1,x_2\cdots\cdots x_n)$，前 n_1 个属于 w_1，后 n_2 个属于 w_2，分别是符合定义的高斯分布的抽样样本。目标是寻找一个超平面将这两类样本区分，考虑寻找该超平面的法线，并且将数据点投影到该直线上，以此降低数据维度，然后利于进行进一步处理，该法线可表示为 $y = w^Tx + b$，其中 w 就是相当于特征向量。

为了描述不同类别的投影的特征，于是有了定义样本类内离散矩阵 Si 以及总的类内离散矩阵 Sw 来描述同一样本的松散程度：

样本均值向量定义为 $m_i = 1 / n_i * (\sum x_j)$

$Si = (\sum (x_j - m_i) (x_j - m_i)^T)$

$Sw = \sum Si$　　其中 i = 1,2

同时定义类间离散矩阵用来定义不同类之间的区别度：

$Sb = (m_1 - m_2)(m_1 - m_2)^T$

问题就可以转换为寻找合适的投影 w，使得降维后数据在法线上的投影 Y_1 和 Y_2 两类具有最大的类间距离以及最小的类内距离，进一步可以使用两者的比值进行描述，使得其比值最大，有如下结论：

对于新的集合 Y_1 和 Y_2 有：

$m_i' = 1 / n_i *(\sum y)$　　$S_i' = \sum (y - m_i')$　　$S_b' = (m_1' - m_2')(m_1' - m_2')^T$

得到比值表示为：$J(w) = ((m_1' - m_2')(m_1' - m_2')^T) / (S_1' + S_2')$

由于需要获得最优的 w 使得上式最大，所以还要进一步处理：

根据 $y = w^Tx$，经过计算，用 X 的结论和 w 表示为：

$m_i' = w^T m_i$　　$S_i' = w^T S_iw$　　$S_w' = w^T S_ww$　　$S_b' = w^T S_bw$

$J(w) = w^T S_bw / (w^T S_ww)$

计算$\frac{\partial J(w)}{\partial w} = 0$（拉格朗日乘子法，$S_bw = \lambda S_ww$），最终得到结论（附完整推导图片）：

$w = S_w^{-1}(m_1 - m_2)$

至此我得到了最优的划分的法线，对于测试集合的分类，可以采用计量在发现上投影与两类均值在法线上的投影的距离，距离近的就认为该测试点属于该类别。实际应该设计判别函数为 $S_w^{-1}(T - 1/2 * (m_1 - m_2))$，用该函数的值与 0 进行比较，表达的意思是相同的。

代码的实现则可以利用 numpy 中的矩阵操作，可以容易完成矩阵、向量的求转置、求逆和加减乘除运算，得到目标参数。

接下来使用不同方法将两类问题推广到多类（N 类）问题上：

One versus the rest：

通过选择所有类别中的一类和其他所有类，将多类判别问题转换为两类的判别问题，进行 N 次分类操作，将测试样例判别到分类次数唯一且最多的一类，舍弃其他情况，实际上只有三类所以直接使用判断语句。

One versus one：

选择已知类别中的所有两对的组合，进行两两分类并且记录被分到的类别，一共进行 $N(N - 1) / 2$ 次分类（意味着相同数量的分类器），得到对应次数的投票，选择其中次数最多的类别，同样也有被舍弃的情况。

Argmax：

使用 M 个分类器，同时使用合适的判别函数，只进行 M 次判别，并且选择其中数值最大的判别结果对应的分类结果作为测试样例的所属分类，不会产生漏判的情况，也不会有过多的判别次数。

线性生成模型：

先行生成模型中首先我们需要选择一个分布模型，然后利用已有的数据计算分布所需要的参数，然后分别计算给定的数据属于已有的类别的概率，取其中概率最大的作为分类结果返回。

对于二维高斯分布，需要的数据只有每个类别的出现概率，每个类别的均值μ，每个类别的协方差矩阵，此处是二维矩阵，可以通过 get_parameter()获得，在这部分实际上是求取函数参数的的最大似然估计，此处根据结论：

$\mu = 1 / n * \sum x$

$\sigma = 1 / n * \sum (x - \mu)(x - \mu)^T$

通过简单计算得到条件最优的参数。

接下来需要计算判别函数，此处利用了贝叶斯定理：

$p(C_k|x) = p(x|C_k)\, p(C_k) / \sum (p(x|C_j)\, p(C_j)) = \text{softmax}(a)$

s.t. $a = [\ln(p(x|C_k)\, p(C_k))]$

同时此处的先验概率可以由已获得的分布生成概率，于是目前计算所需的变量参数齐全，通过计算就会得到关于 $C_{A,B,C}$ 三个集合的后验概率，通过比较概率的大小，取最大值尽心分类即可，最后输出分类的准确性。

线性 vs 生成：

在所有测试数据中，生成模型要优于所有使用的线性模型，而在线性模芯中，一对其他的方法要差得多，正确率低而且有较多的漏判，而一对一和 argmax 方法正确率相差不大，但是一对一可能有漏判。

目前测试中由于数据维度较低而且结果简单，相对区分度比较高，所以两个模型得到的正确率都比较高，差别在一定范围之内，只有比较少的边界值容易造成判断错误，对数据的调整将在 Part III 中进行。

生成模型相比于模型，需要实现估计数据的分布情况，比如此处就直接假设高斯分布，因为真实数据就是二维高斯分布，所以会有较高的精确度，否则如果使用不当的分布函数生

成，会产生较差的结果，所以对于未知分布的数据集，还要多次尝试不同的概率分布。而生成模型只要数据有一定的区分度就可以直接使用，完成判别。

同时良好的生成模型不仅可以对不同数据进行分类，而且可以模拟生成数据，而判别模型则不能做到；但是因为生成模型额外的能力，需要更多的参数以及计算来支持，与之相比，判别模型则能提供更快更轻便的处理仅需分类的问题。

判别模型需要考虑边界处的判别问题，可能选择不同的策略对最终的结果有很大的影响，并且可能会发生漏判（无法判断）的情况，而生成模型则不会有这方面的考虑。

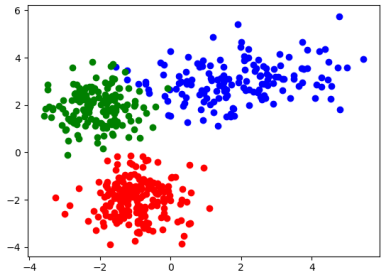
Part III

不同的数据集特征会给分类工作带来不一样的影响，以下列举几种典型的分类情况：

1、 不同的标签的数据分布分散，相同标签的数据分布比较聚集：

```
Discriminative Model
--------- One versus the rest-------
Accuracy of A class:  1.0
Accuracy of B class:  0.8
Accuracy of C class:  0.8666666666666667
Accuracy of all class:  0.9
Expected A B C:  40 30 30
Class A B C:     40 24 29 7
--------- One versus one -----------
Accuracy of A class:  1.0
Accuracy of B class:  0.8
Accuracy of C class:  1.0
Accuracy of all class:  0.94
Expected A B C:  40 30 30
Class A B C:     40 24 36 0
--------- argmax ------------------
Accuracy of A class:  1.0
Accuracy of B class:  0.8666666666666667
Accuracy of C class:  0.9666666666666667
Accuracy of all class:  0.95
Expected A B C:  40 30 30
Class A B C:     41 26 33 0

Generative Model
Accuracy of A class:  1.0
Accuracy of B class:  0.9333333333333333
Accuracy of C class:  1.0
Accuracy of all class:  0.98
Expected A B C:  40 30 30
Class A B C:     40 28 32 0
```

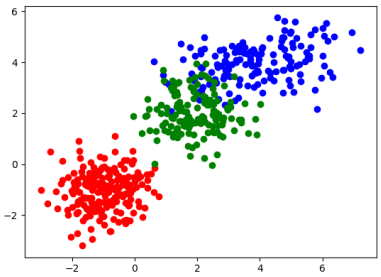


可以看出三类标签的数据存在明显的划分，所以无论是生成模型，还是判别模型的三种方法都能得到 90% 以上的正确率，依然生成模型效果最好，而一对其他的方法任然有漏判。

2、 上一个条件更进一步，所有数据分类的均值大致在一条直线的方向上：

```
Discriminative Model
--------- One versus the rest-------
Accuracy of A class:  1.0
Accuracy of B class:  0.0
Accuracy of C class:  0.4666666666666667
Accuracy of all class:  0.54
Expected A B C:  40 30 30
Class A B C:     40 0 14 46
--------- One versus one -----------
Accuracy of A class:  1.0
Accuracy of B class:  0.8666666666666667
Accuracy of C class:  1.0
Accuracy of all class:  0.96
Expected A B C:  40 30 30
Class A B C:     40 26 34 0
--------- argmax ------------------
Accuracy of A class:  1.0
Accuracy of A class:  1.0
Accuracy of B class:  1.0
Accuracy of C class:  0.7333333333333333
Accuracy of all class:  0.92
Expected A B C:  40 30 30
Class A B C:     40 38 22 0

Generative Model
Accuracy of A class:  1.0
Accuracy of B class:  0.9333333333333333
Accuracy of C class:  0.9
Accuracy of all class:  0.95
Expected A B C:  40 30 30
Class A B C:     40 31 29 0
```

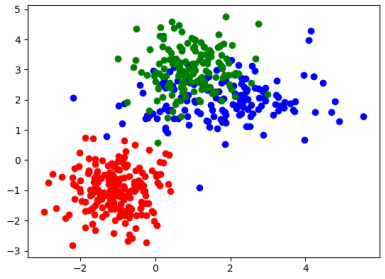


可以发现这时候一对一的判别模型反而效果最好，而一对其他的判别模型基本上失去了作用，这是因为此时两两之间的划分最为清晰，而如果直接使用一对其他的方法，绿色点的数据基本无法区分。

3、 存在两个甚至多个标签的数据高度重合，紧密分布，类间散度较低：

```
Discriminative Model
--------- One versus the rest-------
Accuracy of A class:  1.0
Accuracy of B class:  0.13333333333333333
Accuracy of C class:  0.1
Accuracy of all class:  0.47
Expected A B C:  40 30 30
Class A B C:     42 4 5 49
--------- One versus one -----------
Accuracy of A class:  1.0
Accuracy of B class:  0.7333333333333333
Accuracy of C class:  0.8333333333333334
Accuracy of all class:  0.87
Expected A B C:  40 30 30
Class A B C:     42 26 32 0
--------- argmax -------------------
Accuracy of A class:  1.0
Accuracy of B class:  0.6333333333333333
Accuracy of C class:  0.9333333333333333
Accuracy of all class:  0.87
Expected A B C:  40 30 30
Class A B C:     42 20 38 0

Generative Model
Accuracy of A class:  1.0
Accuracy of B class:  0.9
Accuracy of C class:  0.8666666666666667
Accuracy of all class:  0.93
Expected A B C:  40 30 30
Class A B C:     41 30 29 0
```

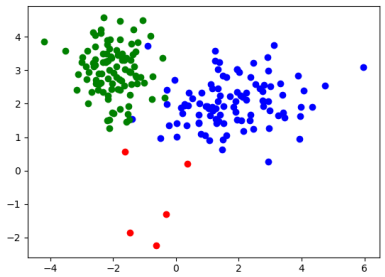


此时 B 和 C 标签的数据比较重合，在判别模型中，正确率就有了不小的下降，而且对于 A 的分类正确率比较高而 B、C 基本分类效果很差，而一对其他的方法下还产生了大量的漏判。但是在生成模型中任然能保持较高的正确率。当数据进一步重合，判别模型还会进一步快速下降。

4、 在 1 的条件下，但是存在某个标签的数据量远小于其他数据量：

```
Discriminative Model
--------- One versus the rest-------
Accuracy of A class:  1.0
Accuracy of B class:  0.9
Accuracy of C class:  1.0
Accuracy of all class:  0.9512195121951219
Expected A B C:  1 20 20
Class A B C:     1 18 21 1
--------- One versus one -----------
Accuracy of A class:  1.0
Accuracy of B class:  0.95
Accuracy of C class:  1.0
Accuracy of all class:  0.975609756097561
Expected A B C:  1 20 20
Class A B C:     1 19 21 0
--------- argmax -------------------
Accuracy of A class:  1.0
Accuracy of B class:  0.95
Accuracy of C class:  1.0
Accuracy of all class:  0.975609756097561
Expected A B C:  1 20 20
Class A B C:     1 19 21 0

Generative Model
Accuracy of A class:  0.0
Accuracy of B class:  0.95
Accuracy of C class:  1.0
Accuracy of all class:  0.9512195121951219
Expected A B C:  1 20 20
Class A B C:     0 20 21 0
```



此时 A 标签的数据只有 5 个，发现虽然所有模型对于该情况的分类效果均比较好，但那是对于 B、C 标签分类结果占的比重较大，在判别模型中对于 A 能较好地分类，但是在生成模型中对于 A 标签判断的成功率为 0，那是因为数据过少，这种标签样本学习的机会很少，不能学习到正确的模型参数，导致分类失败。

代码运行方法：

python source.py