Assignment I

Part I

1，设计高斯分布

多元高斯分布的概率密度可表示为

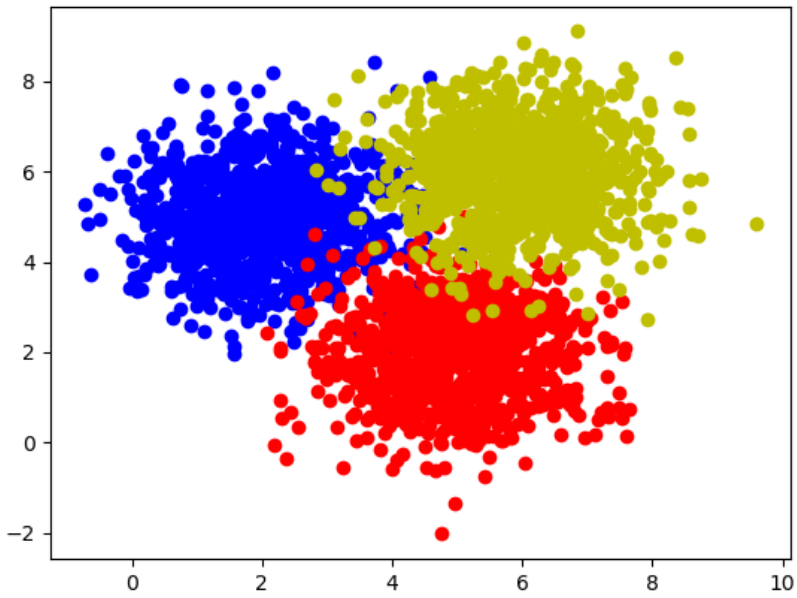
$$\boldsymbol{p}(x) = \frac{\mathbf{1}}{(\mathbf{2\pi})^{D/2}} \frac{\mathbf{1}}{|\Sigma|^{D/2}} \exp\{-\frac{\mathbf{1}}{\mathbf{2}}(x-\boldsymbol{\mu_k})^T \Sigma^{-1}(x-\boldsymbol{\mu_k})\}$$

对于 A 类，$\boldsymbol{\mu_A} = \begin{pmatrix}2\\5\end{pmatrix}$，对于 B 类，$\boldsymbol{\mu_B} = \begin{pmatrix}5\\2\end{pmatrix}$，对于 C 类，$\boldsymbol{\mu_C} = \begin{pmatrix}6\\6\end{pmatrix}$

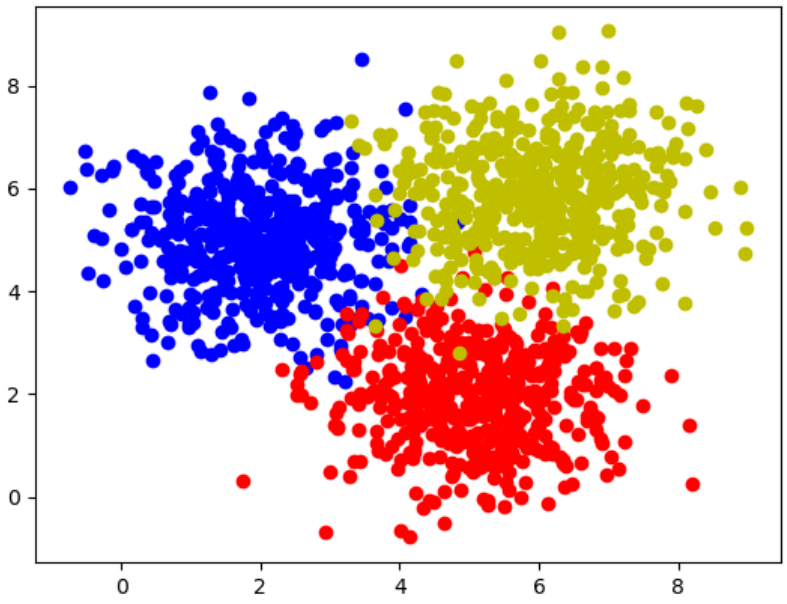
它们共同的协方差矩阵为：$\boldsymbol{\Sigma} = \begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}$

2，数据采样

训练集：A 类 1000 个数据，B 类 1000 个数据，C 类 1000 个数据，均为按概率分布随机生成。散点图如下：



测试集：A 类 500 个数据，B 类 500 个数据，C 类 500 个数据，同样按概率分布随机生成。散点图如下：



Part II

1，generative model

为行文方便，我们使用C_0表示 A 类，使用C_1表示 B 类，使用C_2表示 C 类。对于类C_k，我们首先对其先验概率和条件概率建模，使用最大似然法得到各参数，然后使用贝叶斯定理得到其后验概率。对于测试集中的数据，将它们分别代入三个类的模型中得到对应的后验概率，哪一个最大就将数据分入对应的类中。

数据集形式为$\{\boldsymbol{x_n}, \boldsymbol{t_n}\}$，其中$n = \{1,2,3, \dots\dots, N\}$；$\boldsymbol{t_n} = (1,0,0)^T$表示 A 类，$\boldsymbol{t_n} = (0,1,0)^T$表示 B 类，$\boldsymbol{t_n} = (0,0,1)^T$表示 C 类。令$p(C_0) = \pi_0$，$p(C_1) = \pi_1$，$p(C_2) = \pi_2$。对于来自$C_k$的数据$\boldsymbol{x_n}$，有

$$p(\boldsymbol{x_n}, C_k) = p(C_k)p(\boldsymbol{x_n}|C_k) = \pi_k N(\boldsymbol{x_n}|\boldsymbol{\mu_k}, \boldsymbol{\Sigma})t_{nk}$$

则似然函数为

$$p(\boldsymbol{t}|\boldsymbol{\pi}, \boldsymbol{\mu_1}, \boldsymbol{\mu_2}, \boldsymbol{\mu_3}, \boldsymbol{\Sigma}) = \prod_{n=1}^{N}\prod_{k=0}^{2} t_{nk}\pi_k N(\boldsymbol{x_n}|\boldsymbol{\mu_k}, \boldsymbol{\Sigma})$$

其中$\boldsymbol{t} = (\boldsymbol{t_1}, \boldsymbol{t_2}, \boldsymbol{t_3}, \dots\dots, \boldsymbol{t_N})^{\boldsymbol{T}}$，$\boldsymbol{\pi} = (\pi_0, \pi_1, \pi_2)^{\boldsymbol{T}}$。

对似然函数取对数，再令偏导等于零可依次求得各参数：

$$\pi_k = \frac{N_k}{N_0 + N_1 + N_2}$$

其中N_k为训练集中属于C_k的数据个数。

$$\boldsymbol{\mu_k} = \frac{1}{N_k}\sum_{n=1}^{N} t_{nk}\boldsymbol{x_n}$$

$$\boldsymbol{\Sigma} = \frac{1}{N}\sum_{n=1}^{N}\sum_{k=0}^{2} t_{nk}(\boldsymbol{x_n} - \boldsymbol{\mu_k})(\boldsymbol{x_n} - \boldsymbol{\mu_k})^{\boldsymbol{T}}$$

有了这些参数后，后验概率可表示为：

$$p(C_k|\boldsymbol{x_n}) = \frac{p(\boldsymbol{x_n}|C_k)p(C_k)}{\sum_{\boldsymbol{j}} p(\boldsymbol{x_n}|C_j)p(C_j)} = \frac{\exp(a_k)}{\sum_{\boldsymbol{j}} \exp\,(a_j)}$$

其中

$$a_k = \ln p(\boldsymbol{x_n}|C_k)p(C_k)$$

决策函数为

$$i = \underset{k}{\operatorname{argmax}}\, p(C_k|\boldsymbol{x_n}) = \underset{k}{\operatorname{argmax}}\, a_k$$

则$\boldsymbol{x_n}$应被分入类C_i。

2，discriminative model

数据集形式为$\{\widehat{\boldsymbol{x_n}}, \boldsymbol{t_n}\}$，其中$n = \{1,2,3,\dots\dots,N\}$；$\widehat{\boldsymbol{x_n}} = (\boldsymbol{x_{n1}}, \boldsymbol{x_{n2}}, 1)$为增广特征向量；$\boldsymbol{t_n}$同样为 one-hot 向量。对于数据$\boldsymbol{x_n}$，模型预测其属于类$C_k$的概率为：

$$p(C_k|\boldsymbol{x_n}) = \frac{\exp(\boldsymbol{w_k}^T\boldsymbol{x_n})}{\sum_{\boldsymbol{j}} \exp\,(\boldsymbol{w_j}^T\boldsymbol{x_n})}$$

其中w_k为第 k 类的增广权重向量。

模型的决策函数可表示为：

$$i = \underset{k}{\operatorname{argmax}}\, p(C_k|\boldsymbol{x_n}) = \underset{k}{\operatorname{argmax}}(\boldsymbol{w_k}^T\boldsymbol{x_n})$$

则$\boldsymbol{x_n}$应被分入类C_i。

模型使用交叉熵损失函数来学习最优的参数矩阵$\boldsymbol{W}$，风险函数为：

$$R(\boldsymbol{W}) = -\frac{1}{N}\sum_{n=1}^{N}\sum_{k=0}^{2} t_{nk}\ln y_{nk} = -\frac{1}{N}\sum_{n=1}^{N}\boldsymbol{t_n}^T\ln\boldsymbol{y_n}$$

其中$\boldsymbol{y_n} = \mathrm{softmax}(\boldsymbol{W}^T\boldsymbol{x_n})$为样本$\boldsymbol{x_n}$在每个类别的后验概率。

风险函数$R(\boldsymbol{W})$关于$\boldsymbol{W}$的梯度为：

$$\frac{\partial R(\boldsymbol{W})}{\partial \boldsymbol{W}} = -\frac{1}{N}\sum_{n=1}^{N}\boldsymbol{x_n}(\,\boldsymbol{t_n} - \boldsymbol{y_n})^{\boldsymbol{T}}$$

则采用梯度下降法，模型的训练过程为：初始化$\boldsymbol{W_0} \leftarrow \boldsymbol{0}$，然后通过下式进行迭代更新：

$$\boldsymbol{W}_{t+1} \leftarrow \boldsymbol{W}_t + \alpha\left[\frac{1}{N}\sum_{n=1}^{N}\boldsymbol{x_n}\big(\,\boldsymbol{t_n} - \boldsymbol{y_n}^{(\boldsymbol{W}_t)}\big)^{\boldsymbol{T}}\right]$$

其中α是学习率，取为 1，$\boldsymbol{y}_n^{(\boldsymbol{W}_t)}$是当参数为$\boldsymbol{W}_t$时，softmax 函数的输出。

3，比较

（1）通过本作业

理论上来说，两种模型在多类问题上的的区别在于获得 softmax 函数中参数矩阵的方式不同。对于生成模型而言，首先需要通过最大似然求出先验概率和条件概率表达式中的各参数，然后再用先验概率和条件概率来确定参数矩阵，当然，实际上确定参数矩阵这一步并不是必要的，通过先验概率和条件概率可以直接得到后验概率，写成$\text{softmax}(\boldsymbol{W}^T\boldsymbol{x}_n)$这种形式可以告诉我们决策边界是线性的；而对于判别模型，确定先验概率和条件概率是非必要的，直接构建$\text{softmax}(\boldsymbol{W}^T\boldsymbol{x}_n)$的交叉熵损失函数，通过梯度下降迭代获取参数矩阵。此时写成$\text{softmax}(\boldsymbol{W}^T\boldsymbol{x}_n)$这种形式不仅告诉我们决策边界是线性的，同样也是计算后验概率的必须工具。理论上判别模型显得更加直接。

实践上来说，生成模型由于采用最大似然来学习参数，并且有极值的解析解，故相当于仅仅计算了以样本集为自变量的几个函数值，并没有迭代过程，显得较为直接。而判别模型使用梯度下降，计算开销相对更大，训练耗时更长。

（2）通过课堂和资料

生成模型:
i 可以生成样本数据
ii 生成数据的能力与分类能力正相关
iii 有更多的参数：$2M + \frac{M(M+1)}{2} + 1$
iv 需要较好的便于处理的类分布

判别模型
i 只能用来分类
ii 并不需要做目标范围之外的工作，相对更加简单
iii 有较少的参数：$M + 1$
iv 需要较好的模型来判断决策边界

Part III

1， 初始运行结果
初始时设置判别模型的学习率$\alpha = 0.1$，迭代 1000 次，结果如下:

```
generative model:
correct/total
1443 / 1500
accuracy rate: 0.962
discriminative model:
correct/total
1440 / 1500
accuracy rate: 0.96
```

2， 改变数据规模:

将训练集和测试集同时扩大一倍，结果如下:

```
generative model:
correct/total
2884 / 3000
accuracy rate: 0.9613333333333334
discriminative model:
correct/total
2886 / 3000
accuracy rate: 0.962
```

发现准确率几乎没变

将训练集规模减小一倍，结果如下:

```
generative model:
correct/total
1440 / 1500
accuracy rate: 0.96
discriminative model:
correct/total
1438 / 1500
accuracy rate: 0.9586666666666667
```

发现准确率略微下降

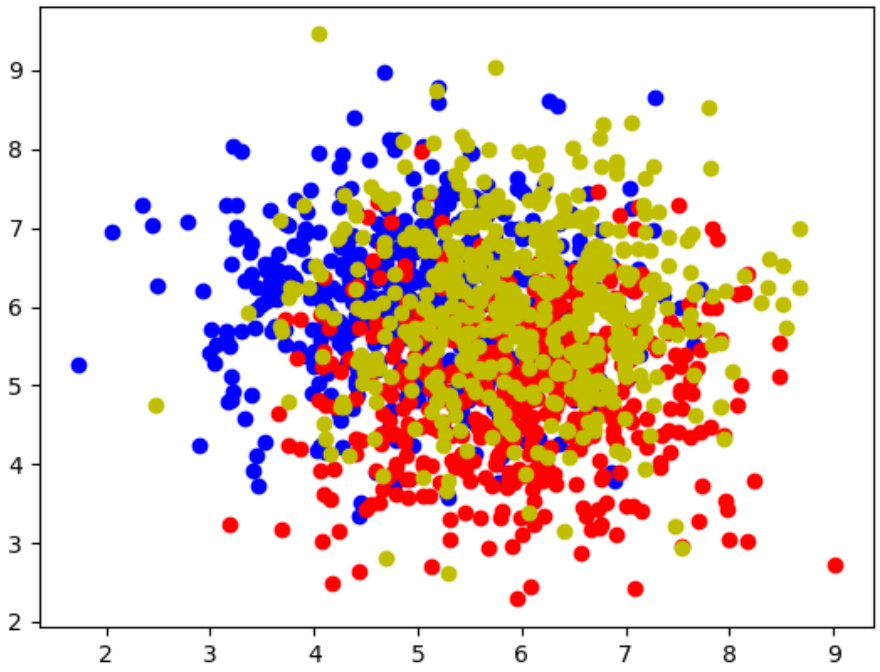
将训练集规模减小两倍，结果如下:

```
generative model:
correct/total
1434 / 1500
accuracy rate: 0.956
discriminative model:
correct/total
1433 / 1500
accuracy rate: 0.9553333333333334
```

发现准确率进一步下降，可见训练集规模太小会减小模型的准确性，但是当训练集规模大到一定程度后对于本作业中的模型准确性的影响就很小了。

3， 改变分布

修改$\boldsymbol{\mu_A},\boldsymbol{\mu_B},\boldsymbol{\mu_C}$，改后，$\boldsymbol{\mu_A}=\begin{pmatrix}5\\6\end{pmatrix}$，对于 B 类，$\boldsymbol{\mu_B}=\begin{pmatrix}6\\5\end{pmatrix}$，对于 C 类，$\boldsymbol{\mu_C}=\begin{pmatrix}6\\6\end{pmatrix}$，协方差矩阵不变。
修改后测试集的散点图如下：



此时的运行结果为：

```
generative model:
correct/total
860 / 1500
accuracy rate: 0.5733333333333334
discriminative model:
correct/total
705 / 1500
accuracy rate: 0.47
```

此时的准确率只有 0.5 左右，分类效果很差。从上面的散点图可以预计到这一点，可以看到散点图上三类几乎重叠在一起，无法使用线性边界将它们分开，本作业中的线性分类模型失效。

4， 改变判别模型学习率和迭代次数

将判别模型的学习率由 1 改为 0.1，结果如下：

```
discriminative model:
correct/total
1424 / 1500
accuracy rate: 0.9493333333333334
```

改为 0.01，结果如下：

```
discriminative model:
correct/total
1355 / 1500
accuracy rate: 0.9033333333333333
```

可以看到准确率进一步降低

再将学习率改为 0.05，迭代次数改为 5000 次，结果如下：

```
discriminative model:
correct/total
1433 / 1500
accuracy rate: 0.9553333333333334
```

发现准确率又回升到一开始的水平

综合以上可以推断，学习率与迭代次数要相匹配，较小的学习率就需要更高的迭代次数。这也是可以预期的，较小的学习率意味着每一次迭代参数矩阵的改变都较小，这样如果迭代次数不够，就会导致还未收敛就结束了学习过程，自然会导致模型准确率的下降。