清华大学电子工程系

媒体与认知 课堂 2

2022-2023 学年春季学期

作业 3

卜丞科 2023 年 5 月 14 日

理论部分

1 单选题（15 分）

1.1 C

1.2 D

1.3 D

1.4 D

1.5 C

2 计算题（15 分）

2.1 **距离地球很远有一个双星系统，其中有两个星球 A 和 B，从地球观测时，星球 A 和 B 的位置重叠。已知星球 A 有 60% 的部分是海洋，其余是陆地，而星球 B 则全是陆地。某一时刻，观测到星球 A 或 B 的概率相同，假设此时观测到该星球上的陆地，计算该星球是星球 A 的概率。**
(提示：全概率公式 $P(Y)=\sum_{i=1}^{N}P(Y|X_i)P(X_i)$**)**

$$P(\text{观察到A}|\text{观察到陆地})=\frac{P(\text{观察到陆地}|\text{观察到A})P(\text{观察到A})}{P(\text{观察到陆地})}$$

$$=\frac{\frac{2}{5}\times\frac{1}{2}}{\frac{1}{2}(\frac{2}{5}+1)}=\frac{2}{7}$$

2.2 给定 5 维空间中的 6 个样本点，写成如下的 6x5 维矩阵 X，其中每一行代表一个样本点

$$X = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ -3 & -3 & -3 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & -1 & -1 \end{bmatrix}$$

(i) 计算该数据集的均值；

(ii) 求解矩阵 X 的 SVD 分解；（提示：矩阵 X 的 SVD 分解的形式为 $\begin{bmatrix} a & 0 \\ -3a & 0 \\ 2a & 0 \\ 0 & b \\ 0 & -2b \\ 0 & b \end{bmatrix} \times \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix} \times \begin{bmatrix} c & c & c & 0 & 0 \\ 0 & 0 & 0 & d & d \end{bmatrix}$）

(iii) 计算该数据集的第一主成分（即最大特征值对应的归一化特征向量）?

(i) $X_{mean} = [0, 0, 0, 0, 0]$

(ii) $$X = \begin{bmatrix} a\sigma_1 c & a\sigma_1 c & a\sigma_1 c & 0 & 0 \\ -3a\sigma_1 c & -3a\sigma_1 c & -3a\sigma_1 c & 0 & 0 \\ 2a\sigma_1 c & 2a\sigma_1 c & 2a\sigma_1 c & 0 & 0 \\ 0 & 0 & 0 & b\sigma_2 d & b\sigma_2 d \\ 0 & 0 & 0 & -2b\sigma_2 d & -2b\sigma_2 d \\ 0 & 0 & 0 & b\sigma_2 d & b\sigma_2 d \end{bmatrix}$$

$$\therefore \begin{cases} a\sigma_1 c = 1 \\ b\sigma_2 d = -1 \\ a^2 + 9a^2 + 4a^2 = 1 \\ b^2 + 4b^2 + b^2 = 1 \\ 3c^2 = 1 \\ 2d^2 = 1 \end{cases}$$

$$\therefore a = \frac{1}{\sqrt{14}} \quad b = \frac{1}{\sqrt{6}} \quad c = \frac{1}{\sqrt{3}} \quad d = \frac{1}{\sqrt{2}} \quad \sigma_1 = \sqrt{42} \quad \sigma_2 = -2\sqrt{3}$$

$$\therefore X = \begin{bmatrix} \frac{1}{\sqrt{14}} & 0 \\ -\frac{3}{\sqrt{14}} & 0 \\ \frac{2}{\sqrt{14}} & 0 \\ 0 & \frac{1}{\sqrt{6}} \\ 0 & -\frac{2}{\sqrt{6}} \\ 0 & \frac{1}{\sqrt{6}} \end{bmatrix} \begin{bmatrix} \sqrt{42} & 0 \\ 0 & -2\sqrt{3} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$$

(iii) $X_{特} = [\frac{1}{\sqrt{3}} \ \frac{1}{\sqrt{3}} \ \frac{1}{\sqrt{3}} \ 0 \ 0]$

2.3 **设有两类正态分布的样本集，第一类均值为** $\mu_1 = [1, 2]^T$**，第二类均值为** $\mu_2 = [-1, -2]^T$**。两类样本集的协方差矩阵相等** $\Sigma_1 = \Sigma_2 = \Sigma = \begin{bmatrix} 3.0 & 1.0 \\ 1.0 & 2.0 \end{bmatrix}$**，先验概率分别为** $p(\omega_1) = 0.6$**，** $p(\omega_2) = 0.4$**。试计算分类界面，并对样本** $x = [0, 0]^T$ **分类。**

$$g(x_1) = -\frac{1}{2}(x-\mu_1)^T \Sigma_1^{-1}(x-\mu_1) - \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln|\Sigma_1| + \ln P(w_1)$$

$$g(x_2) = -\frac{1}{2}(x-\mu_2)^T \Sigma_2^{-1}(x-\mu_2) - \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln|\Sigma_2| + \ln P(w_2)$$

$\because \Sigma_1 = \Sigma_2 \quad \Sigma_1^{-1} = \Sigma_2^{-1} = \begin{bmatrix} 0.4 & -0.2 \\ -0.2 & 0.6 \end{bmatrix}$

$$\therefore g(x) = g(x_1) - g(x_2)$$

$$= \frac{1}{2}\left[(x-\mu_2)^T\Sigma^{-1}(x-\mu_2) - (x-\mu_1)^T\Sigma^{-1}(x-\mu_1)\right] + \ln\frac{P(w_1)}{P(w_2)}$$

$$= \frac{8}{5}y + \ln\frac{3}{2}$$

$\therefore g(x)$ 为分类平面

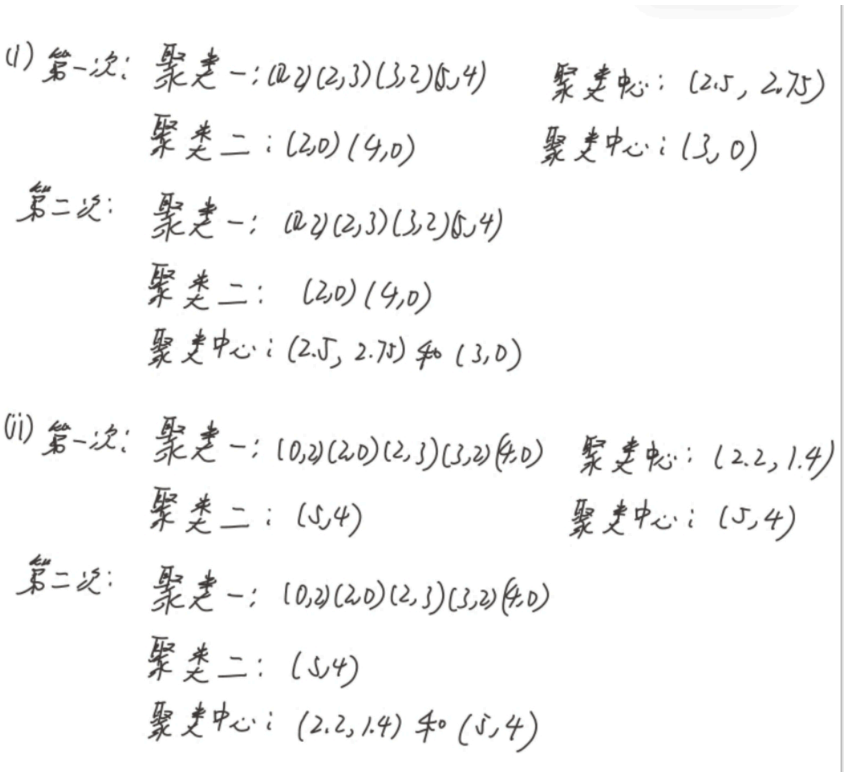
$g(x)|_{x=0,y=0} = \ln\frac{3}{2} > 0$

$\therefore$ 属于第一类

2.4 使用 KMeans 算法对 2 维空间中 6 个点 $(0,2)$，$(2,0)$，$(2,3)$，$(3,2)$，$(4,0)$，$(5,4)$ 进行聚类，距离函数选择哈密顿距离 $d=|x_1-x_2|+|y_1-y_2|$。

(i) 起始聚类中心选择 $(2,3)$ 和 $(4,0)$，计算聚类中心；

(ii) 起始聚类中心选择 $(2,0)$ 和 $(5,4)$，计算聚类中心。



2.5 使用 Lagrange 乘子法求 x^2+2y^2 在 $x^2+y^2\le 1$ 区域内的极值。

在边界时

$L = x^2+2y^2-\lambda(x^2+y^2-1)$

$$\begin{cases}\frac{\partial L}{\partial x} = 2x-2x\lambda = 0\\ \frac{\partial L}{\partial y} = 4y-2\lambda y = 0\\ \frac{\partial L}{\partial \lambda} = x^2+y^2-1 = 0\end{cases}$$

∴ 解为：$\begin{cases}x=0\\ y=\pm 1\\ k=2\end{cases}$ $\begin{cases}x=\pm 1\\ y=0\\ k=1\end{cases}$

∴ 当 $(x,y)=(0,\pm 1)$ 时

此时B矩阵负定，$(0,\pm 1)$ 为极大值点，为1

当 $(x,y)=(0,\pm 1)$ 时

此时B矩阵负定，$(0,\pm 1)$ 为极大值点，为2

在内部时

$\frac{\partial L}{\partial x} = 2x = 0$

$\frac{\partial L}{\partial y} = 2y = 0$

∴ $(0,0)$ 为极小值点，值为0

编程部分

3 编程作业报告

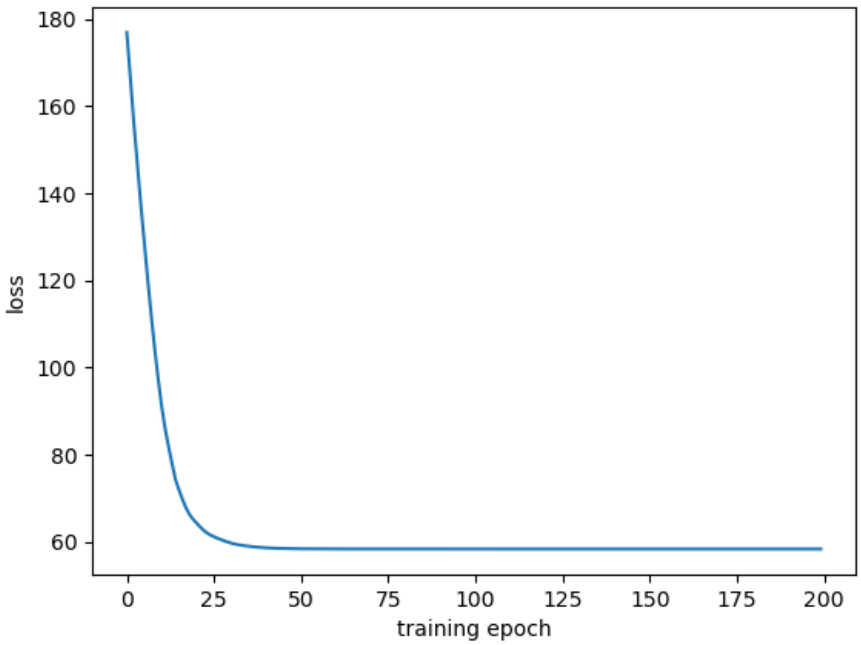
3.1 实现 hinge loss 模拟支持向量机并运行自动评判程序

```
(test) E:\桌面\媒体与认知\HW3>python check.py
Linear successully tested!
Hinge successfully tested!
SVM_HINGE successfully tested!
```

3.2 训练/验证/可视化/比较

3.2.1 Hinge loss 模拟 SVM 的训练及验证

使用 hinge loss 模拟 SVM，按缺省参数训练和验证，可得 loss 函数如下图：

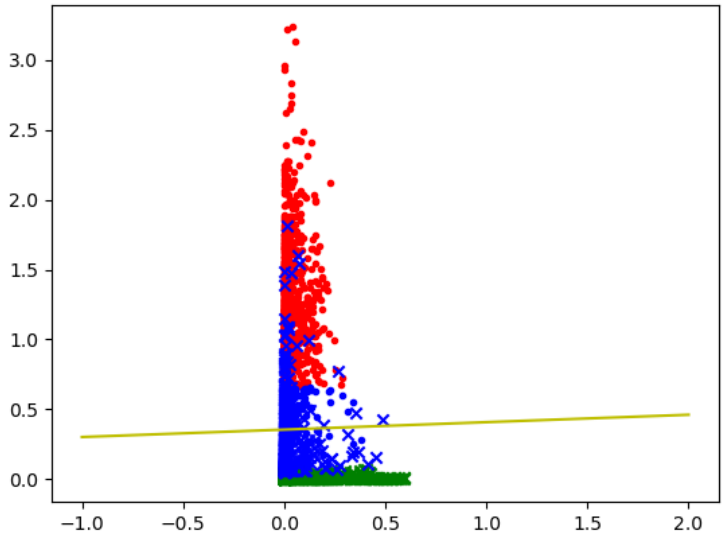


训练结果的准确率也比较可观，为百分之 92.8

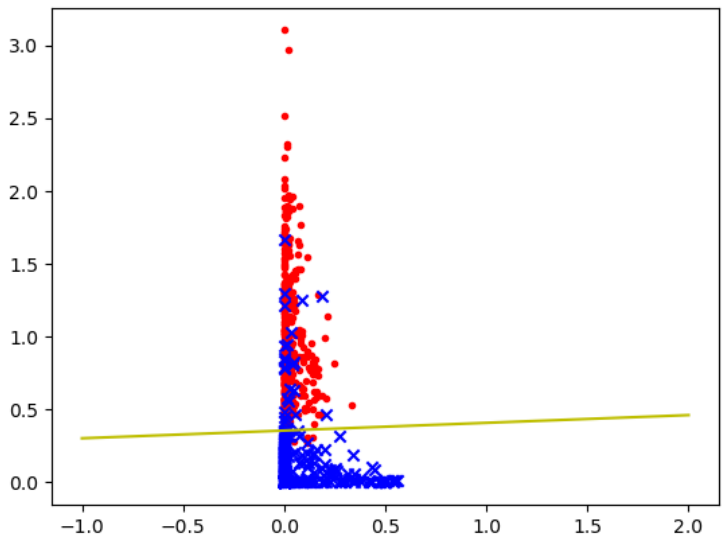
```
Epoch 197: loss = 58.385
Epoch 198: loss = 58.385
Epoch 199: loss = 58.385
Epoch 200: loss = 58.385
Epoch 200: validation accuracy = 92.8%
Model saved in saved_models/recognition.pth
```

3.2.2 可视化分类结果

使用 hinge loss 模拟 SVM 实现分类：
训练集可视化结果：



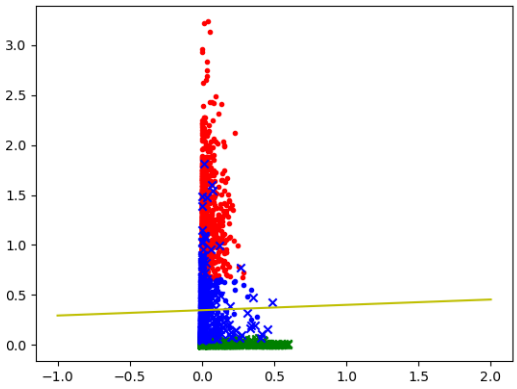
测试集可视化结果：



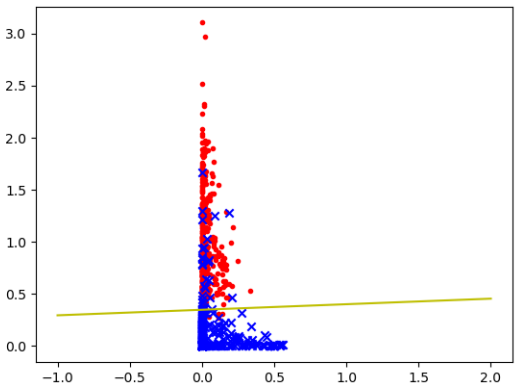
使用 libsvm 库实现分类：

```
(test) E:\桌面\媒体与认知\HW3>python classify_hw.py --mode baseline
*
optimization finished, #iter = 380
nu = 0.266243
obj = -58.385376, rho = 1.178906
nSV = 641, nBSV = 638
Total nSV = 641
Accuracy = 92.75% (742/800) (classification)
```

训练集可视化结果：



测试集可视化结果：



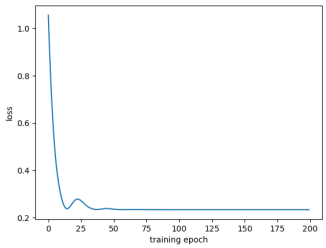
使用 hinge 模拟和使用 libsvm 库得到的结果完全一致，说明我们可以使用 hinge loss 模拟支持向量机。

3.2.3 调整正则化系数 C，体会不同的 C 对分类效果的影响

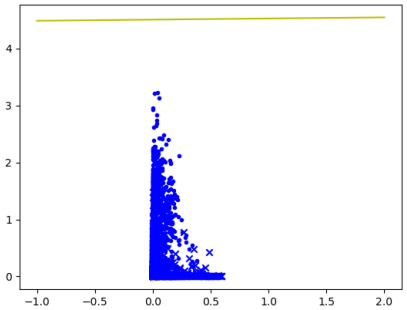
C=0.0001 时准确率：

```
Epoch 197: loss = 0.234
Epoch 198: loss = 0.234
Epoch 199: loss = 0.234
Epoch 200: loss = 0.234
Epoch 200: validation accuracy = 50.0%
Model saved in saved_models/recognition.pth
```

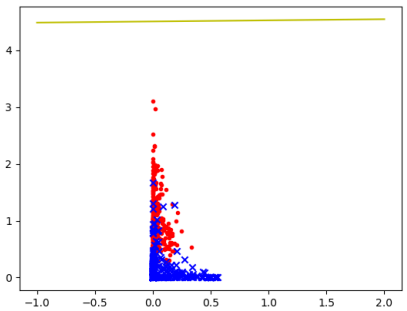
loss 函数：



训练集可视化结果：



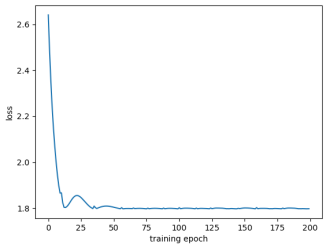
测试集可视化结果：



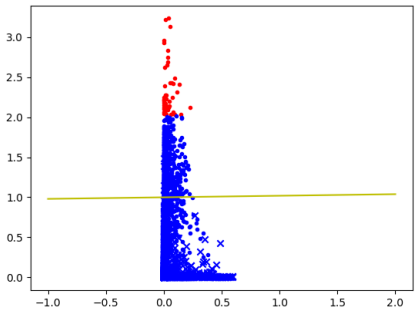
C=0.001 时准确率：

```
Epoch 198: loss = 1.797
Epoch 199: loss = 1.797
Epoch 200: loss = 1.797
Epoch 200: validation accuracy = 67.6%
Model saved in saved_models/recognition.pth
```

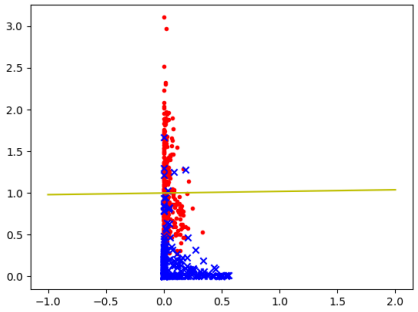
loss 函数：



训练集可视化结果：



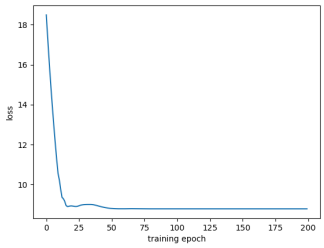
测试集可视化结果：



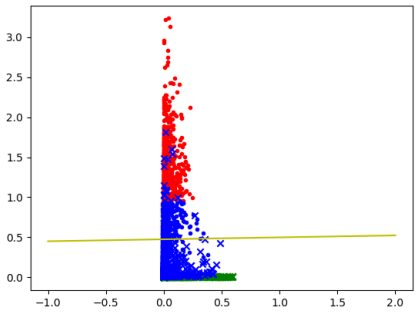
C=0.01 时准确率：

```
Epoch 198: loss = 8.787
Epoch 199: loss = 8.787
Epoch 200: loss = 8.787
Epoch 200: validation accuracy = 91.4%
Model saved in saved_models/recognition.pth
```

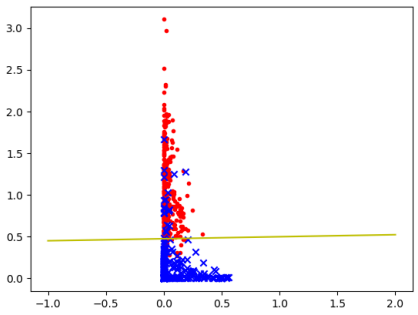
loss 函数：



训练集可视化结果：



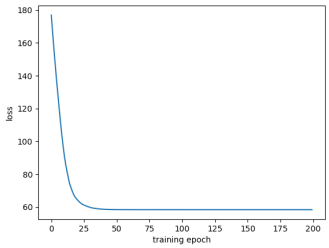
测试集可视化结果：



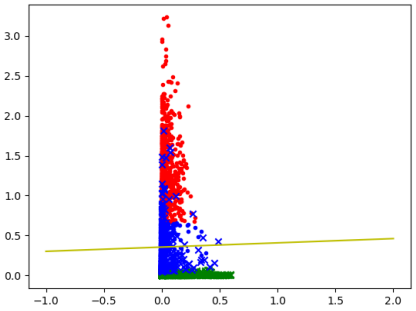
C=0.1 时准确率：

```
Epoch 197: loss = 58.385
Epoch 198: loss = 58.385
Epoch 199: loss = 58.385
Epoch 200: loss = 58.385
Epoch 200: validation accuracy = 92.8%
Model saved in saved_models/recognition.pth
```

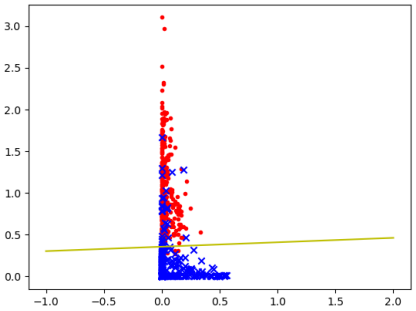
loss 函数：



训练集可视化结果：



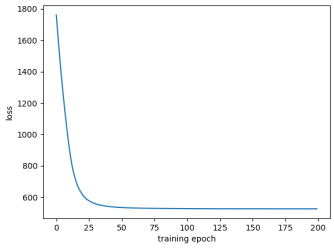
测试集可视化结果：



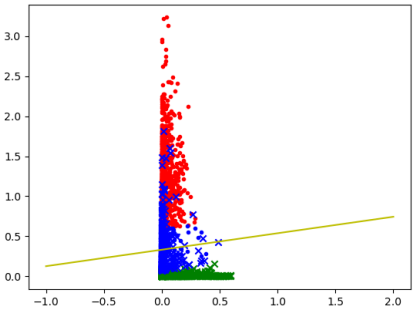
C=1 时准确率：

```
Epoch 197: loss = 525.906
Epoch 198: loss = 525.906
Epoch 199: loss = 525.905
Epoch 200: loss = 525.904
Epoch 200: validation accuracy = 92.4%
Model saved in saved_models/recognition.pth
```

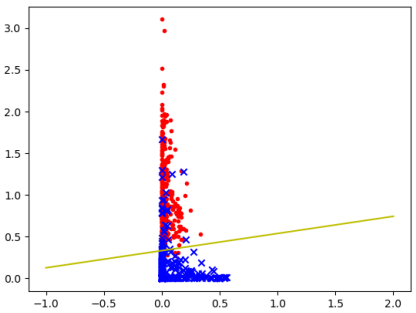
loss 函数：



训练集可视化结果：



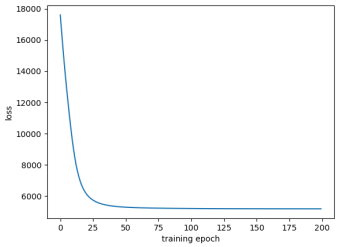
测试集可视化结果：



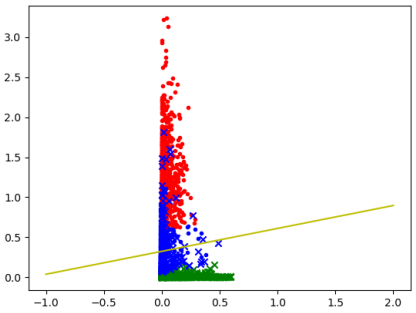
C=10 时准确率：

```
Epoch 197: loss = 5183.135
Epoch 198: loss = 5183.091
Epoch 199: loss = 5183.045
Epoch 200: loss = 5182.997
Epoch 200: validation accuracy = 92.4%
Model saved in saved_models/recognition.pth
```

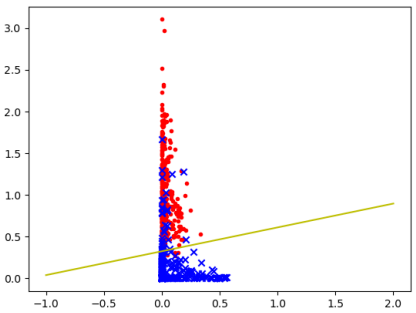
loss 函数：



训练集可视化结果：



测试集可视化结果：



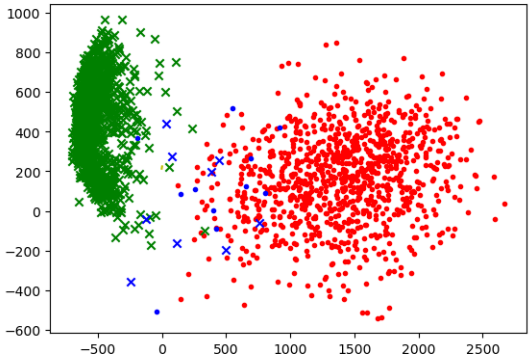
由此可见，当 C 比较小的时候，松弛变量限制过紧，hinge 模拟的效果很差，尤其是当 C=0.0001 的时候，训练模型时将所有点都视为了支持向量，训练出的模型也没有任何判断能力，所有样本点都划入了同一类，模型欠拟合严重。而随着 C 增大，松弛变量的约束放宽松，训练的准确率越来越高，大约在 C=0.01 的时候几乎达到 libsvm 库训练的模型。但是在这之后即使继续增大 C 的值，对于准确率的提升也变得十分有限。

3.3 使用 SVM 算法在 MNIST 数据集上进行分类

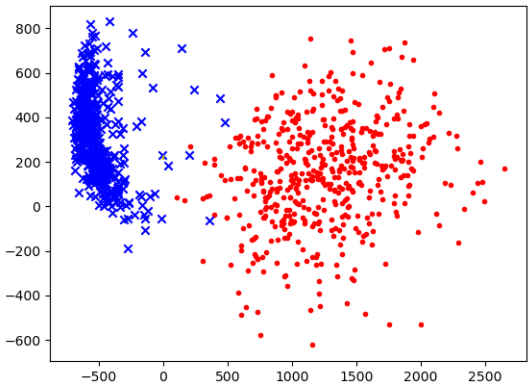
提取数字 0 和数字 1 的图片特征进行分类：该模型的效果非常好，准确率可以达到百分之 99.8，参数如下：

```
optimization finished, #iter = 3611083
nu = 0.006703
obj = -1.344896, rho = -0.422849
nSV = 20, nBSV = 9
Total nSV = 20
Accuracy = 99.8% (998/1000) (classification)
```

分布图：
训练集：



测试集



与之前相比，特征降维后的样本点分类更加清晰，并且可以利用更少的样本点作为支持向量来计算决策面，准确率相较之前也有了大幅度的提升。

3.4 作业中遇到的问题、解决方法以及建议

在运行指令

```
python process_mnist.py --class_0 0 --class_1 1 --feat_dim 10s
```

有如下的报错：

```
(test) E:\桌面\媒体与认知\HW3>python process_mnist.py --class_0 0 --class_1 1 --feat_dim 10
OMP: Error #15: Initializing libiomp5md.dll, but found libiomp5md.dll already initialized.
OMP: Hint This means that multiple copies of the OpenMP runtime have been linked into the program. That is dangerous, since it
can degrade performance or cause incorrect results. The best thing to do is to ensure that only a single OpenMP runtime is link
ed into the process, e.g. by avoiding static linking of the OpenMP runtime in any library. As an unsafe, unsupported, undocumen
ted workaround you can set the environment variable KMP_DUPLICATE_LIB_OK=TRUE to allow the program to continue to execute, but
that may cause crashes or silently produce incorrect results. For more information, please see http://www.intel.com/software/pr
oducts/support/.
```

后来经过查资料发现只要在 process_mnist.py 的文件头加入
os.environ[”KMP_DUPLICATE_LIB_OK”]=”TRUE”
即可解决报错。