THU-70250043-0, Pattern Recognition (Spring 2021) **Homework: 4**

Overfitting, Error Rate Estimation and Linear Classifiers

Lecturer: Changshui Zhang zcs@mail.tsinghua.edu.cn

Hong Zhao vzhao@tsinghua.edu.cn

Student: Jingxuan Yang yangjx20@mails.tsinghua.edu.cn

Overfitting

1. Consider N i.i.d. observations $\boldsymbol{X} = \{x_1, x_2, \ldots, x_N\}$ with corresponding target values $\boldsymbol{T} = \{t_1, t_2, \ldots, t_N\}$. We want to fit these observations into a model

$$t = y(x, \boldsymbol{w}) + \epsilon \tag{1}$$

where $\boldsymbol{w}$ is the model parameter and ϵ is the error term.

1.1 To find $\boldsymbol{w}$, we can minimize the sum of square error

$$E(\boldsymbol{w}) = \frac{1}{2} \sum_{n=1}^{N} \left[y(x_n, \boldsymbol{w}) - t_n \right]^2 \tag{2}$$

Now suppose we believe that the distribution of error term ϵ is Gaussian

$$p(\epsilon|\beta) = \mathcal{N}(\epsilon|0, \beta^{-1}) \tag{3}$$

where $\beta = \dfrac{1}{\sigma^2}$ is the inverse of variance. Using the property of Gaussian distribution, we have

$$p(t|x, \boldsymbol{w}, \beta) = \mathcal{N}(t|y(x, \boldsymbol{w}), \beta^{-1}) \tag{4}$$

Under this assumption, the likelihood function is given by

$$p(\boldsymbol{T}|\boldsymbol{X}, \boldsymbol{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n|y(x_n, \boldsymbol{w}), \beta^{-1}) \tag{5}$$

Show that the problem of finding the maximum likelihood solution for $\boldsymbol{w}$ is equivalent to the problem of minimizing the sum of square error (2).

解: 对数似然函数为

$$\begin{aligned} H(\boldsymbol{\omega}) &= \ln p(\boldsymbol{T}|\boldsymbol{X}, \boldsymbol{w}, \beta) \\ &= \ln \left[\prod_{n=1}^{N} \mathcal{N}(t_n|y(x_n, \boldsymbol{w}), \beta^{-1}) \right] \\ &= \sum_{n=1}^{N} \ln \mathcal{N}(t_n|y(x_n, \boldsymbol{w}), \beta^{-1}) \\ &= \sum_{n=1}^{N} \ln \left[\frac{1}{\sqrt{2\pi\beta^{-1}}} \exp\left(-\frac{[t_n - y(x_n, \boldsymbol{w})]^2}{2\beta^{-1}} \right) \right] \\ &= \sum_{n=1}^{N} \left(\ln \frac{1}{\sqrt{2\pi\beta^{-1}}} - \frac{\beta}{2}[t_n - y(x_n, \boldsymbol{w})]^2 \right) \\ &= -N \ln \sqrt{2\pi\beta^{-1}} - \frac{\beta}{2} \sum_{n=1}^{N} [t_n - y(x_n, \boldsymbol{w})]^2 \end{aligned} \tag{6}$$

所以, $\boldsymbol{w}$ 的最大似然估计为

$$\hat{\boldsymbol{w}} \in \underset{\boldsymbol{w}}{\operatorname{argmax}} \left\{ -N \ln \sqrt{2\pi\beta^{-1}} - \frac{\beta}{2} \sum_{n=1}^{N} [t_n - y(x_n, \boldsymbol{w})]^2 \right\} \tag{7}$$

忽略第一项常数项, 上述问题等价于

$$\hat{\boldsymbol{w}} \in \underset{\boldsymbol{w}}{\operatorname{argmax}} \left\{ -\frac{\beta}{2} \sum_{n=1}^{N} [t_n - y(x_n, \boldsymbol{w})]^2 \right\} \tag{8}$$

又 $\beta > 0$, 则有

$$\hat{\boldsymbol{w}} \in \underset{\boldsymbol{w}}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{n=1}^{N} [t_n - y(x_n, \boldsymbol{w})]^2 \right\} \tag{9}$$

对最小二乘法, 有

$$\tilde{\boldsymbol{w}} \in \underset{\boldsymbol{w}}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{n=1}^{N} [y(x_n, \boldsymbol{w}) - t_n]^2 \right\} \tag{10}$$

比较式 (9) 与式 (10) 可知 $\tilde{\boldsymbol{w}} = \hat{\boldsymbol{w}}$, 即最大似然估计与最小二乘的结果等价.

1.2 In order to avoid overfitting, we often add a weight decay term to (2)

$$\tilde{E}(\boldsymbol{w}) = \frac{1}{2} \sum_{n=1}^{N} \left[y(x_n, \boldsymbol{w}) - t_n \right]^2 + \frac{\lambda}{2} \|\boldsymbol{w}\|_2^2 \tag{11}$$

On the other hand, we believe that $\boldsymbol{w}$ has a prior distribution of

$$p(\boldsymbol{w}|\alpha) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \alpha^{-1}\boldsymbol{I})$$

Using Bayes theorem, the posterior distribution for $\boldsymbol{w}$ is proportional to the product of the prior distribution and the likelihood function

$$p(\boldsymbol{w}|\boldsymbol{X},\boldsymbol{T},\alpha,\beta) \propto p(\boldsymbol{T}|\boldsymbol{X},\boldsymbol{w},\beta)p(\boldsymbol{w}|\alpha) \tag{12}$$

Show that the problem of finding Maximum A Posterior (MAP) solution for $\boldsymbol{w}$ (i.e., maximizing (12)) is equivalent to the problem of minimizing (11).

解: 令

$$\begin{aligned}
\tilde{H}(\boldsymbol{\omega}) &\triangleq \ln\left[p(\boldsymbol{T}|\boldsymbol{X},\boldsymbol{w},\beta)p(\boldsymbol{w}|\alpha)\right] \\
&= \ln\left[\prod_{n=1}^{N}\mathcal{N}(t_n|y(x_n,\boldsymbol{w}),\beta^{-1})p(\boldsymbol{w}|\alpha)\right] \\
&= \sum_{n=1}^{N}\ln\mathcal{N}(t_n|y(x_n,\boldsymbol{w}),\beta^{-1}) + \ln p(\boldsymbol{w}|\alpha) \\
&= \sum_{n=1}^{N}\left(\ln\frac{1}{\sqrt{2\pi\beta^{-1}}} - \frac{[t_n - y(x_n,\boldsymbol{w})]^2}{2\beta^{-1}}\right) + \ln\frac{1}{\sqrt{2\pi\alpha^{-1}}} - \frac{\|\boldsymbol{w}\|_2^2}{2\alpha^{-1}} \\
&= -N\ln\sqrt{2\pi\beta^{-1}} - \ln\sqrt{2\pi\alpha^{-1}} - \frac{\beta}{2}\sum_{n=1}^{N}[t_n - y(x_n,\boldsymbol{w})]^2 - \frac{\alpha}{2}\|\boldsymbol{w}\|_2^2
\end{aligned} \tag{13}$$

所以, $\boldsymbol{w}$ 的最大后验估计 (MAP) 为

$$\hat{\boldsymbol{w}} \in \underset{\boldsymbol{w}}{\operatorname{argmax}}\left\{-N\ln\sqrt{2\pi\beta^{-1}} - \ln\sqrt{2\pi\alpha^{-1}} - \frac{\beta}{2}\sum_{n=1}^{N}[t_n - y(x_n,\boldsymbol{w})]^2 - \frac{\alpha}{2}\|\boldsymbol{w}\|_2^2\right\} \tag{14}$$

忽略常数项, 上述问题等价于

$$\hat{\boldsymbol{w}} \in \underset{\boldsymbol{w}}{\operatorname{argmax}}\left\{-\frac{\beta}{2}\sum_{n=1}^{N}[t_n - y(x_n,\boldsymbol{w})]^2 - \frac{\alpha}{2}\|\boldsymbol{w}\|_2^2\right\} \tag{15}$$

又 $\beta > 0$, 则有

$$\hat{\boldsymbol{w}} \in \underset{\boldsymbol{w}}{\operatorname{argmin}}\left\{\frac{1}{2}\sum_{n=1}^{N}[t_n - y(x_n,\boldsymbol{w})]^2 + \frac{\alpha}{2\beta}\|\boldsymbol{w}\|_2^2\right\} \tag{16}$$

对加入正则项的最小二乘法, 有

$$\tilde{\boldsymbol{w}} \in \underset{\boldsymbol{w}}{\operatorname{argmin}}\left\{\frac{1}{2}\sum_{n=1}^{N}[y(x_n,\boldsymbol{w}) - t_n]^2 + \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2\right\} \tag{17}$$

比较式 (16) 与式 (17) 可知, 当 $\lambda = \dfrac{\alpha}{\beta}$ 时, 有 $\tilde{\boldsymbol{w}} = \hat{\boldsymbol{w}}$, 即最大后验估计与正则化最小二乘的结果等价.

Programming for Error Rate Estimation

2. Consider a two dimensional classification problems: $p(\omega_1) = p(\omega_2) = 0.5$, $p(x|\omega_1) \sim N(\mu_1, \Sigma_1)$, $p(x|\omega_2) \sim N(\mu_2, \Sigma_2)$, where

$$\mu_1 = \begin{bmatrix} -1 \\ 0 \end{bmatrix},\ \mu_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix},\quad \Sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ \Sigma_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{18}$$

2.1 Derive the misclassification rate of the Bayesian classifier theoretically.

解: Bayes 决策边界满足

$$p(x|\omega_1)p(\omega_1) = p(x|\omega_2)p(\omega_2) \tag{19}$$

即

$$\frac{1}{2}\frac{1}{2\pi}\exp\left[-\frac{(x_1+1)^2+x_2^2}{2}\right] = \frac{1}{2}\frac{1}{2\pi}\exp\left[-\frac{(x_1-1)^2+x_2^2}{2}\right] \tag{20}$$

化简得

$$x_1 = 0 \tag{21}$$

所以, 错误率为

$$\begin{aligned} P(e) &= \int_{-\infty}^{\infty}\int_{-\infty}^{0} p(x|\omega_2)p(\omega_2)\mathrm{d}x_1\mathrm{d}x_2 + \int_{-\infty}^{\infty}\int_{0}^{\infty} p(x|\omega_1)p(\omega_1)\mathrm{d}x_1\mathrm{d}x_2 \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{0} \frac{1}{2}\frac{1}{2\pi}\exp\left[-\frac{(x_1-1)^2+x_2^2}{2}\right]\mathrm{d}x_1\mathrm{d}x_2 + \int_{-\infty}^{\infty}\int_{0}^{\infty} \frac{1}{2}\frac{1}{2\pi}\exp\left[-\frac{(x_1+1)^2+x_2^2}{2}\right]\mathrm{d}x_1\mathrm{d}x_2 \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{0} \frac{1}{2\pi}\exp\left[-\frac{(x_1-1)^2+x_2^2}{2}\right]\mathrm{d}x_1\mathrm{d}x_2 \\ &= \int_{-\infty}^{0} \frac{1}{\sqrt{2\pi}}\exp\left[-\frac{(x_1-1)^2}{2}\right]\mathrm{d}x_1 \\ &= \Phi(-1) \\ &= 0.1587 \end{aligned} \tag{22}$$

2.2 Choose a proper n and draw n samples from $p(x|\omega_1)$ and $p(x|\omega_2)$ with labels respectively. Estimate $p_n(x|\omega_1)$ and $p_n(x|\omega_2)$ by Parzen window method, with Gaussian window function and unit hypercube window function. Design Bayesian classifier with your estimated $p_n(x|\omega_1)$ and $p_n(x|\omega_2)$. Compare their misclassification rate with the theoretical optimal Bayesian classifier in theory.

解: 取 $n = 1000$, 使用二维高斯核函数

$$K(x) = \frac{1}{2\pi a^2}\exp\left(-\frac{\|x\|_2^2}{2a^2}\right) \tag{23}$$

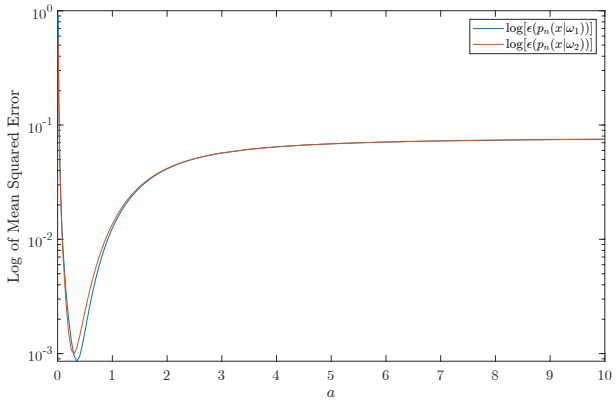

图 1: 高斯估计均方误差与窗宽变化关系

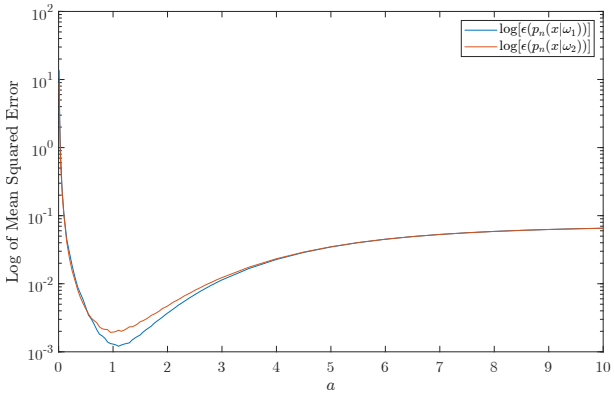

图 2: 方窗估计均方误差与窗宽变化关系

进行估计, 取窗宽 $a = [0.01, 10]$, 做出高斯估计均方误差与窗宽的变化曲线如图 1 所示, 由图可以得出 $n = 1000$ 时高斯估计的最优窗宽为 $a = 0.3$. 在最优窗宽的情况下做 10 次估计, 得到平均均方误差为

$$\epsilon(p_n(x|\omega_1)) = 0.0011, \quad \epsilon(p_n(x|\omega_2)) = 0.0011 \tag{24}$$

使用二维方窗核函数

$$K(x) = \begin{cases} \dfrac{1}{a^2}, & \|x\|_\infty \leqslant \dfrac{a}{2} \\ 0, & \text{otherwise.} \end{cases} \tag{25}$$

进行估计, 同样取 $a = [0.01, 10]$, 做出方窗估计均方误差与窗宽的变化曲线如图 2 所示, 由图可以得出 $n = 1000$ 时方窗估计的最优窗宽为 $a = 1$. 在最优窗宽的情况下做 10 次估计, 得到平均均方误差为

$$\epsilon(p_n(x|\omega_1)) = 0.0012, \quad \epsilon(p_n(x|\omega_2)) = 0.0010 \tag{26}$$

由于 $p(\omega_1) = p(\omega_2) = 0.5$, 则 Bayes 分类器为

$$\begin{aligned} p_n(x|\omega_1) > p_n(x|\omega_2) &\to x \in \omega_1 \\ p_n(x|\omega_2) > p_n(x|\omega_1) &\to x \in \omega_2 \end{aligned} \tag{27}$$

在最优窗宽的情况下做 10 次估计, 高斯估计和方窗估计的错误率分别如表 1 和表 2 所示, 平均错误率为

$$\bar{P}_{\text{Guassian}}(e) = 0.1569, \quad \bar{P}_{\text{hypercube}}(e) = 0.1587 \tag{28}$$

理论上, 当样本数量 $n \to \infty$ 时, Parzen 窗法得到的概率密度函数估计是没有偏差的, 则错误率也与 Bayes 理论错误率相同. 但是实际估计时样本数量有限, 估计的概率密度函数与理论值之间存在偏差, 则 Parzen 窗法错误率也与 Bayes 理论错误率存在偏差. 通过实验数据比较两种 Parzen 窗法和 Bayes 的理论错误率, 可以看出方窗的平均错误率与 Bayes 理论错误率几乎相同, 高斯窗的平均错误率比 Bayes 理论错误率略小一些.

表 1: 高斯窗估计错误率

No.	$P_1(e)$	$P_2(e)$	$P(e)$
1	0.1389	0.1746	0.1568
2	0.1641	0.1493	0.1567
3	0.1672	0.1493	0.1582
4	0.1392	0.1754	0.1573
5	0.1623	0.1510	0.1566
6	0.1611	0.1516	0.1563
7	0.1518	0.1624	0.1571
8	0.1568	0.1568	0.1568
9	0.1634	0.1495	0.1564
10	0.1446	0.1694	0.1570

表 2: 方窗估计错误率

No.	$P_1(e)$	$P_2(e)$	$P(e)$
1	0.1671	0.1507	0.1589
2	0.1495	0.1671	0.1583
3	0.1642	0.1510	0.1576
4	0.1609	0.1549	0.1579
5	0.1748	0.1426	0.1587
6	0.1434	0.1733	0.1584
7	0.1549	0.1621	0.1585
8	0.1304	0.1898	0.1601
9	0.1632	0.1538	0.1585
10	0.1699	0.1497	0.1598

2.3 From above experiments, what's your suggestion for choosing optimal window function and parameters with given n?

解: 建议选择窗函数和窗宽参数时, 根据估计的概率密度函数的均方误差和相应的错误率来选择, 综合选择均方误差较小且错误率也较小的窗函数和窗宽参数. 根据上述实验结果可知, 本实验中若选择高斯函数, 且最优窗宽选择 $a = 0.3$, 则此时估计的均方误差和错误率相对于其他测试参数而言都是最小的.

2.4 Sample $2n$ points from the Gaussian mixture distribution $p(x)$ without labels. Use EM to estimate μ_1, μ_2, Σ_1, Σ_2 so that we estimate $p_{2n}(x|\omega_1)$ and $p_{2n}(x|\omega_2)$. Which method is more accurate in estimating $p(x|\omega_1)$ and $p(x|\omega_2)$, EM or Parzen window? Prove your statement by experiments.

解: 使用 EM 算法估计 10 次, 其中似然概率最大的一组估计结果为

$$\hat{\mu}_1 = \begin{bmatrix} -1.0212 \\ -0.0259 \end{bmatrix},\ \hat{\mu}_2 = \begin{bmatrix} 0.9645 \\ -0.0234 \end{bmatrix},\quad \hat{\Sigma}_1 = \begin{bmatrix} 1.0156 & -0.0165 \\ -0.0165 & 1.0035 \end{bmatrix},\ \hat{\Sigma}_2 = \begin{bmatrix} 1.0156 & -0.0165 \\ -0.0165 & 1.0035 \end{bmatrix} \tag{29}$$

且估计的均方误差为

$$\epsilon(p_{2n}(x|\omega_1)) = 0.0003, \quad \epsilon(p_{2n}(x|\omega_2)) = 0.0009 \tag{30}$$

通过均方误差的比较来看, EM 算法最好的估计结果比两种 Parzen 窗的估计效果都更加精确.

2.5 Design Bayesian classifier with the estimated $p_{2n}(x|\omega_1)$ and $p_{2n}(x|\omega_2)$ by EM. Analyze its performance, i.e., the expectation and variance of misclassification rate and compare them with that of optimal Bayesian classifier.

解: 由于 $p(\omega_1) = p(\omega_2) = 0.5$, 则 Bayes 分类器为

$$\begin{aligned} p_{2n}(x|\omega_1) > p_{2n}(x|\omega_2) &\to x \in \omega_1 \\ p_{2n}(x|\omega_2) > p_{2n}(x|\omega_1) &\to x \in \omega_2 \end{aligned} \tag{31}$$

使用 EM 算法估计 10 次, 错误率如表 3 所示.

表 3: EM 算法估计错误率

No.	$P_1(e)$	$P_2(e)$	$P(e)$
1	0.2229	0.3611	0.2920
2	0.1777	0.1732	0.1754
3	0.1819	0.1695	0.1757
4	0.1546	0.1570	0.1558
5	0.1611	0.1504	0.1557
6	0.2014	0.2941	0.2478
7	0.1749	0.1480	0.1615
8	0.1582	0.1528	0.1555
9	0.1782	0.1728	0.1755
10	0.1552	0.1563	0.1558

错误率的期望和方差分别为

$$\mathbb{E}[P(e)] = 0.1851, \quad \mathrm{Var}[P(e)] = 0.0022 \tag{32}$$

由实验结果可知, EM 算法的平均错误率比 Bayes 分类的理论错误率高, 方差较小但还是有一些, 主要是因为 EM 算法偶尔会收敛到局部最优解.

2.6 Conclude your results. Which method is your favorite to estimate parameters and which classifier is your favorite classifier? Why?

解: 这个实验内容包含了之前所学的统计决策方法和概率密度函数的估计两部分内容. 首先, 在概率分布模型以及参数都已知时, 可以直接使用 Bayes 分类, 得到最小错误率的分类器. 若已知概率分布模型, 但是不知道具体参数值, 则可以选择 EM 算法等参数方法进行参数估计, 再进行 Bayes 分类; 当概率分布模型也未知时, 可以选用 Parzen 窗等非参数方法进行估计, 之后进行 Bayes 分类.

相对而言, 我更倾向于选择 EM 算法等参数化方法来估计概率模型的参数并使用 Bayes 分类器, 这是因为 Parzen 窗等非参数方法存在维数爆炸的问题, 同时运算量很大, 还要选择合适的窗函数以及最优窗宽大小; 而参数方法相对比较简单, 且运算量更小.

Programming for Single Sample Correction Algorithm

3. The training process of the Single Sample Correction Algorithm (Algorithm 1) can be regarded as a searching process for a solution in feasible solution region, whereas no strict restrictions are demanded for the capacity of this solution. The solution only needs to satisfy $a^\top y_n > 0$, where a is the weight vector of the perceptron, and y_n is the normalized augmented sample vector. However, the margin perceptron (Algorithm 2) requires the finally converged hyperplane possesses a margin ($> \gamma$), where γ is a predefined positive scalar. It means that the final solution of perceptron need to satisfy $a^\top y_n > \gamma$.

Thus, there are two types of "mistakes" during the training of perceptron, namely (1) the prediction mistake and (2) margin mistake (i.e., its prediction is correct, but its margin is not large enough).

Algorithm 1 Fixed-Increment Single Sample Correction Algorithm

initialize a, $k \leftarrow 0$
repeat
  $k \leftarrow (k+1) \bmod n$
  if y_k is misclassified by a **then**
    $a \leftarrow a + y_k$
  end if
until all patterns are properly classified
return a

Algorithm 2 Single Sample Correction Algorithm With Margin

initialize a, $k \leftarrow 0$, $\gamma > 0$
repeat
  $k \leftarrow (k+1) \bmod n$
  if $a^\top y_k \leqslant \gamma$ **then**
    $a \leftarrow a + y_k$
  end if
until all patterns are properly classified with a large enough margin γ
return a

3.1 Please generate 200 data points in the 2D plane, among which 100 data points are labeled as 1 and the remaining 100 are labeled as -1. Make sure that these 200 data points are linearly separable. Plot these 200 data points in a 2D plane.

解: 设定超平面保证两类数据点线性可分, 生成的 200 个数据点保存在 `percepData.mat` 文件, 如图 3 所示.

3.2 Implement the classical perceptron algorithm and run it on the above generated data points. Plot the classification boundary and these data points in one figure.

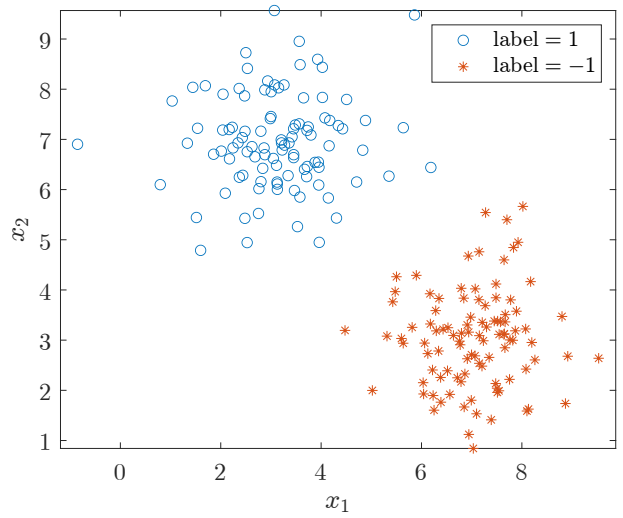

图 3: 200 个数据点, 分为两类

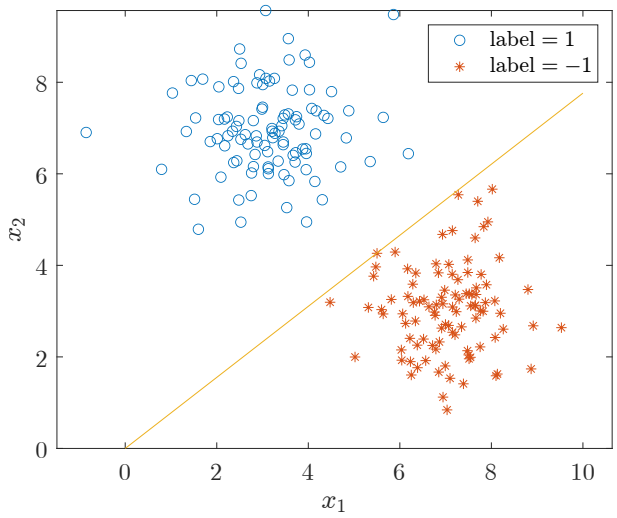

图 4: 经典感知器分类结果

解: 经典感知器的分类边界如图 4 所示.

3.3 Implement the margin perceptron algorithm and run it on the above generated data points. Plot the classification boundary and these data points in one figure. Analyze the impacts of γ on algorithm convergence and the classification boundary.

解: 取 $\gamma = 10$, margin 感知器的分类边界如图 5 所示.

γ 大小对算法迭代次数的影响如图 6 所示, 可知随着 γ 的增大, 算法迭代次数不断震荡波动但是大体上逐渐增加, 因此总得来说增加 margin 对算法收敛速度是不利的.

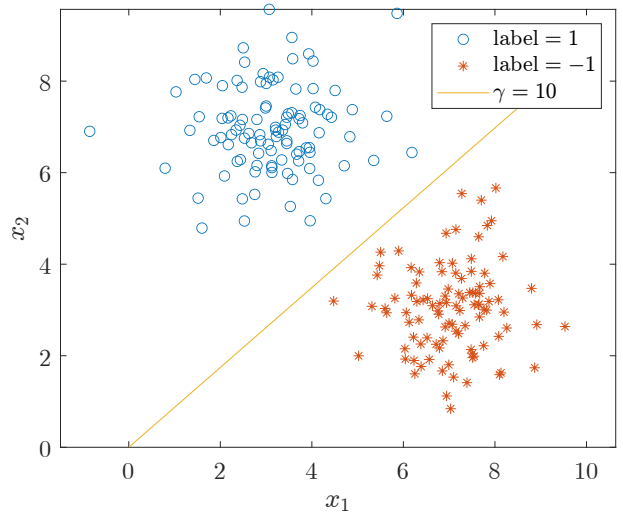

图 5: 感知器分类结果, margin 为 10

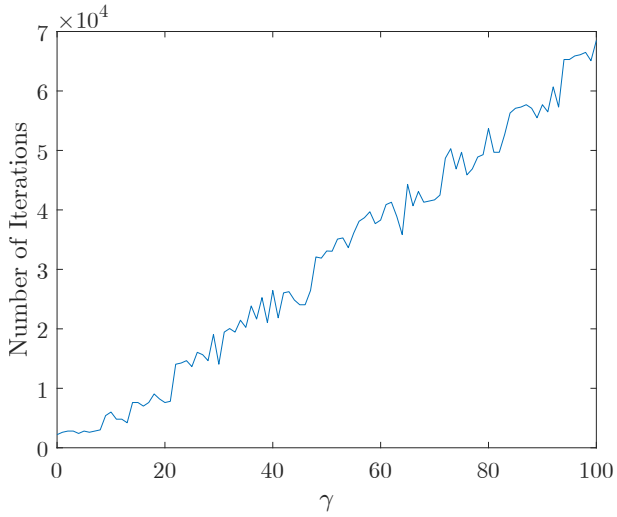

图 6: margin 大小对算法迭代次数的影响

取 $\gamma = 0, 10, \ldots, 80$, 分别绘制分类边界如图 7 所示, 可以看出分类边界的位置随着 γ 的增大而改变, 分类边界到与其最近的数据点的距离随着 γ 的增大而有所增大, 逐渐趋于两类之间居中的位置.

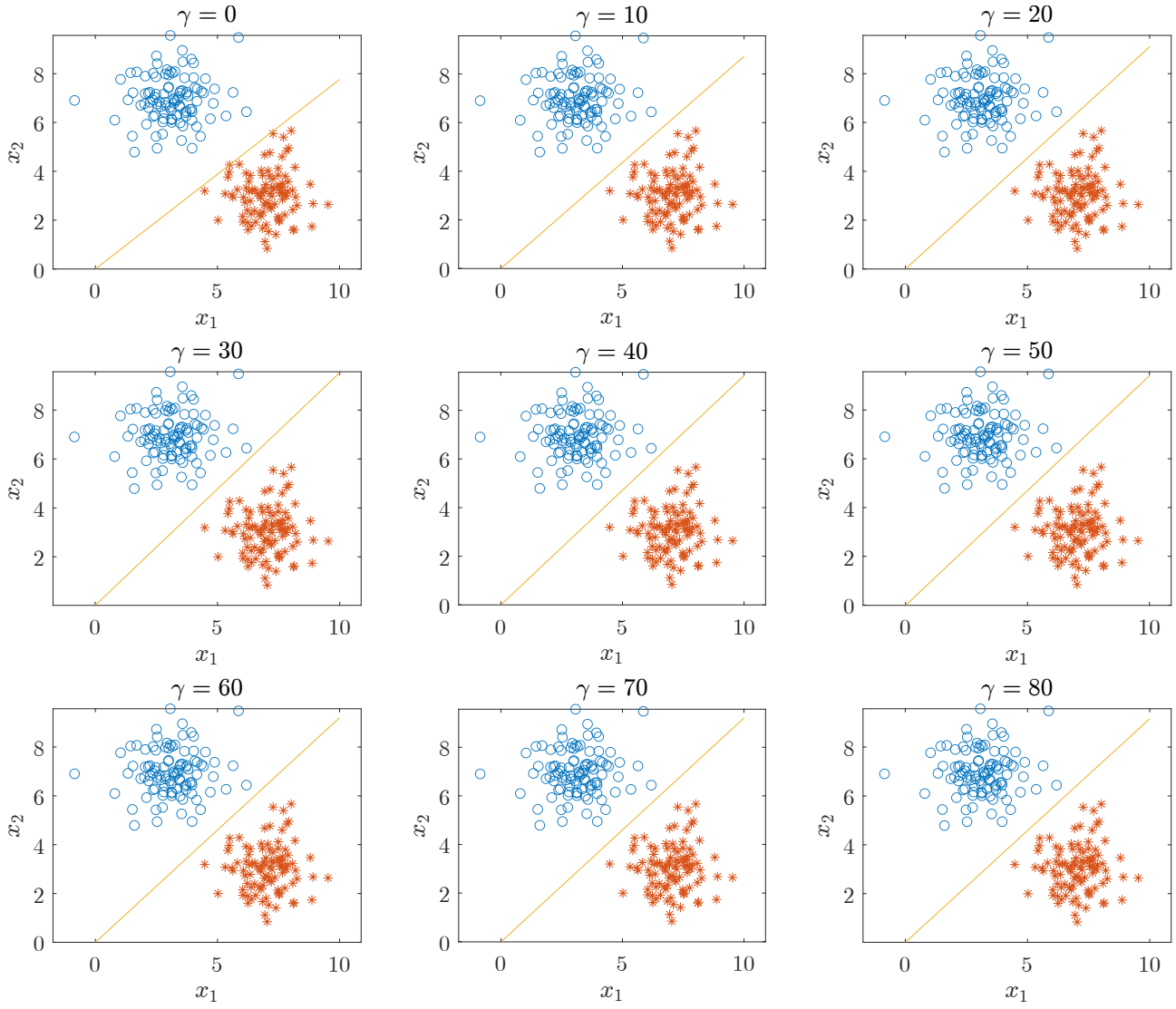

图 7: 感知器分类结果, margin 从 0 变化到 80

Proof of Single Sample Correction Algorithm

4. Suppose we have N points x_i in $\mathbb{R}^p$ with class labels $\omega_i \in \{-1, 1\}$. Prove that the perceptron algorithm converges to a separating hyperplane in finite steps:

- Denoting a hyperplane by $f(x) = a_1^\top x + a_0 = 0$, or in more compact notation $a^\top y = 0$, where $y = (x, 1)$ and $a = (a_1, a_0)$. Let $z_i = \dfrac{y}{\|y\|_2}$. Show that separability implies there exists a a_{sep} such that $\omega_i a_{\text{sep}}^\top z_i \geqslant 1$, $\forall\ i = 1, 2, \ldots, N$.
- Given a current a_{old}, the Algorithm 1 identifies a point z_i that is misclassified, and produces the update $a_{\text{new}} = a_{\text{old}} + \omega_i z_i$. Show that $\|a_{\text{new}} - a_{\text{sep}}\|_2^2 \leqslant \|a_{\text{old}} - a_{\text{sep}}\|_2^2 - 1$, and hence the algorithm converges to a separating hyperplane in no more than $\|a_{\text{start}} - a_{\text{sep}}\|_2^2$ steps.

证明: 假设数据是线性可分的, 则 $\exists\ \bar{a} \in \mathbb{R}^{p+1}$ 使得

$$\omega_i \bar{a}^\top y_i > 0, \quad \forall\ i = 1, 2, \ldots, N \tag{33}$$

其中 $y_i = (x_i, 1)$, 则 $\|y_i\|_2 > 0, \forall\ i = 1, 2, \ldots, N$, 所以

$$\omega_i \bar{a}^\top \frac{y_i}{\|y_i\|_2} > 0, \quad \forall\ i = 1, 2, \ldots, N \tag{34}$$

令 $z_i = \dfrac{y_i}{\|y_i\|_2}$, 则有

$$\omega_i \bar{a}^\top z_i > 0, \quad \forall\ i = 1, 2, \ldots, N \tag{35}$$

所以

$$\omega_i \bar{a}^\top z_i \geqslant \min_i \{\omega_i \bar{a}^\top z_i\} > 0, \quad \forall\ i = 1, 2, \ldots, N \tag{36}$$

因此

$$\frac{\omega_i \bar{a}^\top z_i}{\min_i \{\omega_i \bar{a}^\top z_i\}} \geqslant 1, \quad \forall\ i = 1, 2, \ldots, N \tag{37}$$

取

$$a_{\text{sep}} = \frac{\bar{a}}{\min_i \{\omega_i \bar{a}^\top z_i\}}$$

则有

$$\omega_i a_{\text{sep}}^\top z_i \geqslant 1, \quad \forall\ i = 1, 2, \ldots, N \tag{38}$$

假设点 z_i 被 a_{old} 错分, 即有

$$\omega_i a_{\text{old}}^\top z_i < 0 \tag{39}$$

则算法下一步更新

$$a_{\text{new}} = a_{\text{old}} + \omega_i z_i \tag{40}$$

结合以上三式可得

$$\begin{aligned} \|a_{\text{new}} - a_{\text{sep}}\|_2^2 &= \|a_{\text{old}} - a_{\text{sep}} + \omega_i z_i\|_2^2 \\ &= \|a_{\text{old}} - a_{\text{sep}}\|_2^2 + \|\omega_i z_i\|_2^2 + 2\omega_i (a_{\text{old}} - a_{\text{sep}})^\top z_i \\ &= \|a_{\text{old}} - a_{\text{sep}}\|_2^2 + 1 + 2\omega_i a_{\text{old}}^\top z_i - 2\omega_i a_{\text{sep}}^\top z_i \\ &\leqslant \|a_{\text{old}} - a_{\text{sep}}\|_2^2 + 1 + 0 - 2 \\ &= \|a_{\text{old}} - a_{\text{sep}}\|_2^2 - 1 \end{aligned} \tag{41}$$

假设算法从 a_{start} 开始, 经 n 步更新得到 $a^{(n)}$, 则由上式可知

$$0 \leqslant \|a^{(n)} - a_{\text{sep}}\|_2^2 \leqslant \|a^{(n-1)} - a_{\text{sep}}\|_2^2 - 1 \leqslant \ldots \leqslant \|a_{\text{start}} - a_{\text{sep}}\|_2^2 - n \tag{42}$$

因此有

$$n \leqslant \|a_{\text{start}} - a_{\text{sep}}\|_2^2 \tag{43}$$

即算法更新次数不会超过 $\|a_{\text{start}} - a_{\text{sep}}\|_2^2$, 所以算法不超过 $\|a_{\text{start}} - a_{\text{sep}}\|_2^2$ 步就会收敛到一个分离超平面. □