Linear Optimization Assignment #2

Due: Sunday, April 15, 23:59:59

Instruction: Write a report and complete code. Download the code from ftp (10.13.71.168). Upload report and code to ftp (10.13.72.84).

- Please name the report as hw2_31xxxxxxxx.pdf and use pdf format; Please name the compressed file as hw2_31xxxxxxxx.zip or hw2_31xxxxxxxx.rar. And put your name into report.
- Upload:
  - Address: 10.13.72.84
  - Username: opt; Passwd: opt18; Port: 21
- Download:
  - Address: 10.13.71.168
  - Username: opt; Passwd: opt18; Port: 21

Problem 1 Short Answers

(a) What is the advantage of using cross-validation over splitting a dataset into dedicated training and test sets? When is that less important?

(b) Describe 3 optimization tricks for speeding up learning in multi-layer perceptron training for a fixed error function and network design.

(c) Describe the training process of RBF shortly.

Solution 1 *Short Answers*

(a) What is the advantage of using cross-validation over splitting a dataset into dedicated training and test sets? 将数据的不同部分轮流作为数据集和验证集, 客观公正地测量模型性能。When is that less important? 测试集较大，较丰富。数据集很大时，进行交叉验证会很耗时。(0.5*2=1 pnts)

(b) mini-batch，momentum，学习率衰减、batch normalize、其他合理答案（0.5*3=1.5 pnts）

(c) 1. 要训练一个 RBF, 首先需要找到基函数的中心, 可以用 K-means 算法; 2. 接着需要计算基函数中的方差项, 可以使用上一步 K-means 的结果, 计算每一类各自的方差, 或者使用不同类中心点间的距离; 3. 然后通过线性回归和最小均方误差准则确定权重 w_i ; 最后需通过交叉验证选择基的个数，以及其他超参数。（0.5*3=1.5 pnts）

Problem 2 Linear Separability

Consider the following two sets of points: $C_1 = \{(0,0),(-1,1),(1,1)\}$, $C_2 = \{(0,2),(-2,0),(2,0)\}$.

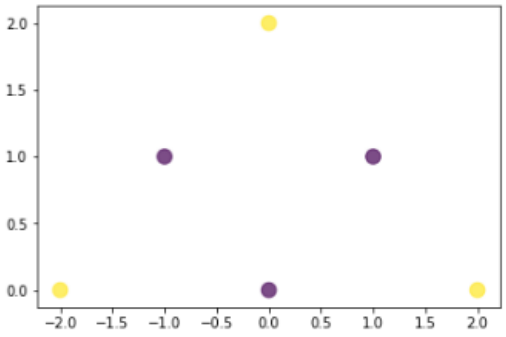
(a) Are these points linearly separable? Why or why not?

(b) Design a MLP that can separate them and plot its decision boundaries.

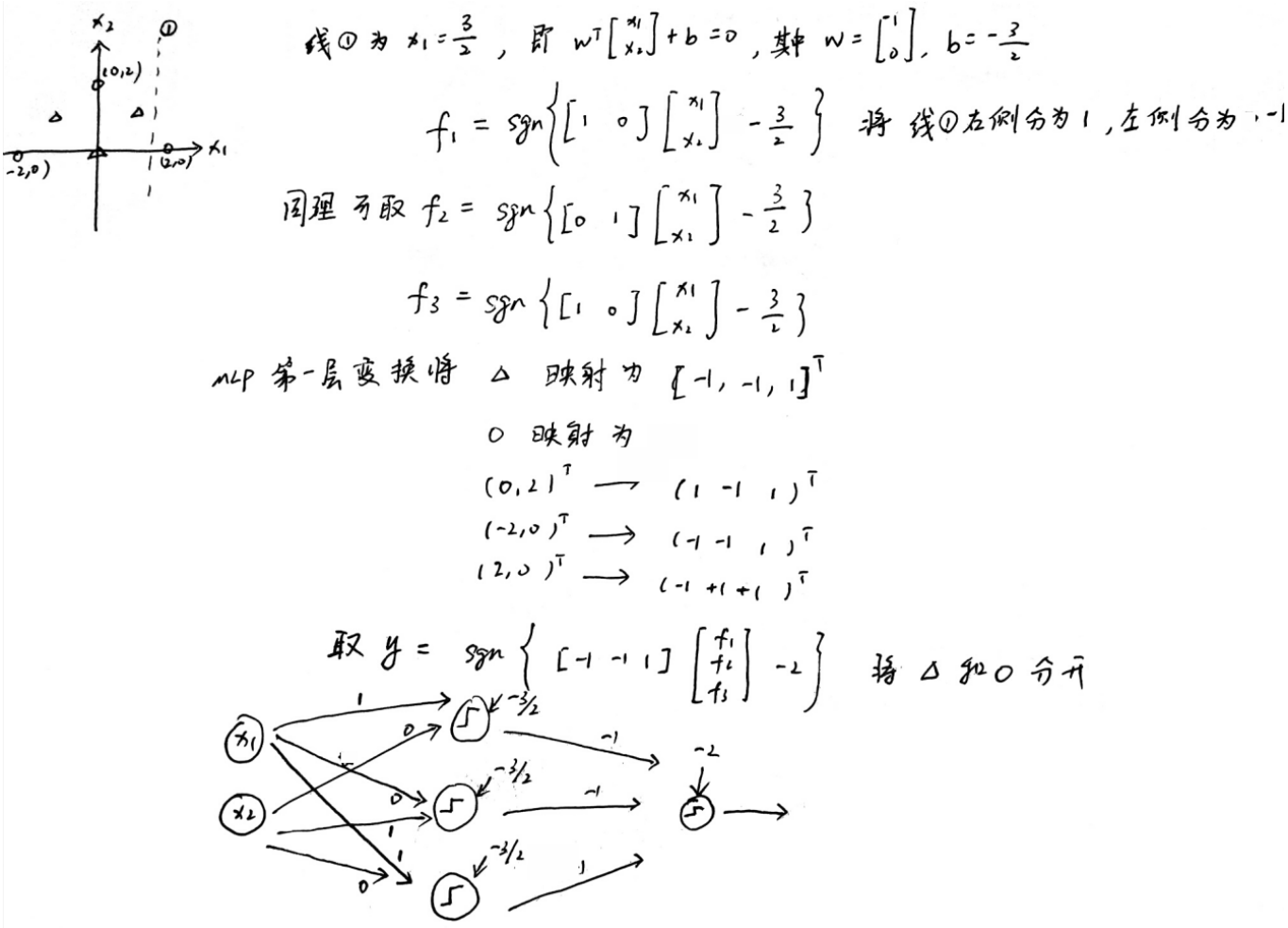
(c) Design a RBF net that can separate them and plot its decision boundaries.

Solution 2 *Linear Separability*

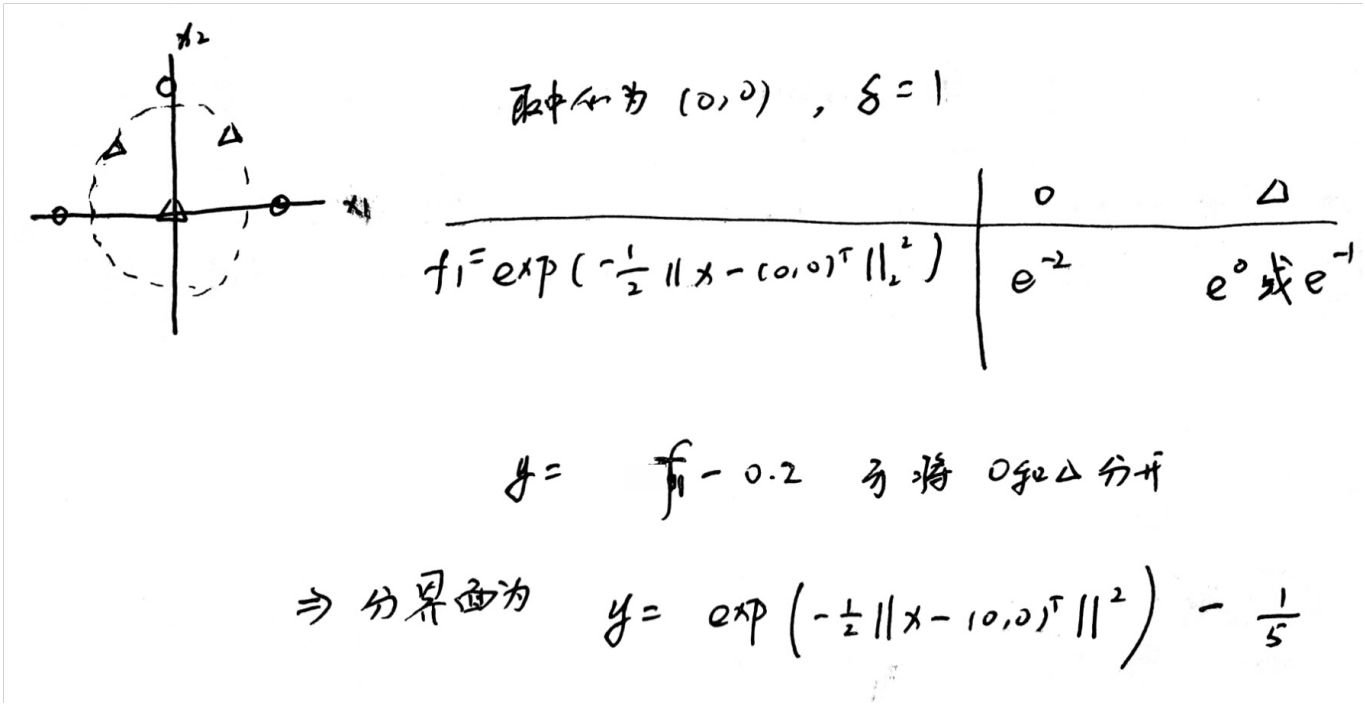
(a) 不是线性可分的, 因为找不到一条直线将它们分在直线两侧 (1 pnts)



(b) 其他 MLP 也正确，手算编程算均可。画出 MLP 结构图 (1 pnt)，画出决策面或写出决策面表达式 (1 pnt)，有求解过程或程序（1 pnt）



(c) 其他 RBF 也正确，手算编程算均可。画出决策面或写出决策面表达式 (1 pnt)，有求解过程或程序（1 pnt）



Problem 3 Duality

(a) Show that the dual LP of $\min\{b^T y; A^T y = c, y \geq 0\}$ is $\max\{c^T x; Ax \leq b\}$.

(b) ***Lagrangian relaxation of Boolean LP*** A Boolean linear program is an optimization problem of the form

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax \leq b \\ & x_i \in \{0,1\}, \quad i = 1, \ldots, n \end{array} \tag{1}$$

and is, in general, very difficult to solve. Relax $x_i \in \{0,1\}$ to $0 \leq x_i \leq 1$, we get LP relaxation of Boolean LP. Relax $x_i \in \{0,1\}$ to $x_i(1-x_i) = 0$ and find its lagrangian dual, we get lagrangian relaxation of this problem.

- Derive the dual of LP relaxation
- Derive the Lagrangian relaxation, *i.e.*, the dual of $x_i(1-x_i) = 0$ relaxation.
- (Bonus, you can choose to skip this question) prove the optimal value for LP relaxation and lagrangian relaxation are the same.

Hint for bonus:

- Derive and use the dual of LP relaxation, since LP satisfies strong duality.

- standard form convex problem is equivalent to its epigraph form, *i.e.*

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.t.} \quad & g_i(x) \leq 0, \quad i = 1, \ldots, n \\ & Ax = b \end{aligned} \tag{2}$$

equivalent to

$$\begin{aligned} \min_{x,t} \quad & t \\ \text{s.t.} \quad & f(x) - t \leq 0 \\ & g_i(x) \leq 0, \quad i = 1, \ldots, n \\ & Ax = b \end{aligned} \tag{3}$$

- To minimize over multiple variables, we can first minimize one variable.

(c) l_2 **norm soft margin SVMs** If our data is not linearly separable, then we can modify our support vector machine algorithm by introducing an error margin that must be minimized. Specifically, the formulation we have looked at is known as the l_1 norm soft margin SVM. In this problem we will consider an alternative method, known as the l_2 norm soft margin SVM. This new algorithm is given by the following optimization problem (notice that the slack penalties are now squared):

$$\begin{aligned} \min_{w,b,\varepsilon} \quad & 1/2||w||^2 + C/2 \sum_{i=1}^{m} \varepsilon_2^2 \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1 - \varepsilon_i, \quad i = 1, \ldots, m \end{aligned} \tag{4}$$

What is the Lagrangian of the l_2 soft margin SVM optimization problem? What is the dual of the l_2 soft margin SVM optimization problem?

Solution 3 *Duality*

(a), 拉格朗日函数 1 pnt，对偶问题 1pnt，必须注意 λ 是向量，写成 λc 则错误。

原问题

$$\min_{y} \quad b^T y$$
$$\text{s.t.} \quad A^T y = c$$

$$\tilde{L}(\lambda) = \inf_{y \geq 0} \{ b^T y + \lambda^T (A^T y - c) \}$$

$$= \inf_{y \geq 0} \{ (b^T + \lambda^T A^T) y - \lambda^T c \}$$

$$= \begin{cases} -\lambda^T c & b^T + \lambda^T A^T \geq 0 \\ -\infty & b^T + \lambda^T A^T < 0 \end{cases}$$

对偶问题为

$$\max_{\lambda} \tilde{L}(\lambda) \Leftrightarrow \max_{\lambda} \; -\lambda^T c \Leftrightarrow \max_{\lambda} \; -c^T \lambda$$
$$\text{s.t.} \quad b^T + \lambda^T A^T \geq 0 \qquad \text{s.t.} \quad b + \lambda A \geq 0$$

令 $x = -\lambda^*$

$$\Leftrightarrow \max \; c^T x$$
$$\text{s.t.} \quad Ax \leq b$$

(b). The lagrangian L and dual function g of LP relaxation are (2 pnts)

$$\begin{aligned} L(x, u, v, w) &= c^T x + u^T (Ax - b) - v^T x + w^T (x - \mathbf{1}) \\ &= (c + A^T u - v + w)^T x - b^T u - \mathbf{1}^T w \end{aligned} \tag{5}$$

$$g(u, v, w) = \begin{cases} -b^T u - \mathbf{1}^T w & A^T u - v + w + c = 0 \\ -\infty & \text{otherwise.} \end{cases}$$

The dual problem is (1 pnt)

$$\begin{aligned} \max \quad & -b^T u - \mathbf{1}^T w \\ \text{s.t.} \quad & A^T u - v + w + c = 0 \\ & u \geq 0, v \geq 0, w \geq 0 \end{aligned} \tag{6}$$

The dual function $\tilde{L}$ of lagrangian relaxation is (2 pnts), 求解过程过程，比如求导求极值、表达式化简（1 pnt）

$$\tilde{L}(\mu,\nu) = \inf_x \left\{ c^T x + \mu^T(Ax-b) - \nu^T x + x^T \mathrm{diag}(\nu) x \right\}$$

$$= \begin{cases} -\mu^T b - \frac{1}{4}\sum_{i=1}^{n} \left(c_i + a_i^T \mu - \nu_i\right)^2/\nu_i, & \nu \geq 0 \\ -\infty, & \nu < 0 \end{cases}$$

二次型极小值：$L(x,\mu,\nu) = (c^T + \mu^T A - \nu^T)x - \mu^T b + x^T \mathrm{diag}(\nu) x$

令 $d^T = c^T + \mu^T A - \nu^T$

$L(x,\mu,\nu) = d^T x - \mu^T b + x^T \mathrm{diag}(\nu) x$

$\frac{\partial L}{\partial x} = 0 \Rightarrow x = -\frac{1}{2}(\mathrm{diag}(\nu))^{-1} d.$

代回 L 得，$\tilde{L}(\mu,\nu) = -\frac{1}{4} d^T \mathrm{diag}(\nu) d - \mu^T b$

当 $\frac{\partial^2 L}{\partial^2 x} \geq 0$ 时，上式为最小值

The lagrangian relaxation of Boolean LP is （1 pnt）

$$\begin{aligned} \max \quad & -b^T\mu - \frac{1}{4}\sum_{i=1}^{n}(c_i + a_i^T\mu - \nu_i)^2/\nu_i \\ \text{s.t.} \quad & \nu \geq 0, \mu \geq 0 \end{aligned} \tag{7}$$

Bonus 部分：

To prove they are equivalent, we can first eliminate ν from the lagrangian relaxation according to use hint (3). （+1 pnt）求解过程过程，比如求导求极值、表达式化简（+0.5）

$\sum_{\nu_i \geq 0}(-(c_i + a_i^T\mu - \nu_i)^2/\nu_i) = \min\{0, 4(c_i + a_i^T\mu)\}$

$$\begin{aligned} \max \quad & -b^T\mu + \sum_{i=1}^{n}\min\{0, c_i + a_i^T\mu\} \\ \text{s.t.} \quad & \mu \geq 0 \end{aligned} \tag{8}$$

Let $-\omega_i = \min\{0, c_i + a_i^T\mu\}$, and use hint (2), equation 8 is exactly equation 7 (+1 pnt)

如果是用其他的方法的，Bonus 部分 +2 pnt 即可

(c) 拉格朗日函数 1 pnt，KKT 条件 1 pnt，求导正确 1 pnt，对偶问题 1 pnt

$$L(w,b,\xi,\alpha) = \frac{1}{2}\|w\|^2 + \frac{C}{2}\sum_{i=1}^{m}\xi_i^2 + \sum_{i=1}^{m}\alpha_i\left[1-\xi_i-y_i(w^T x_i+b)\right]$$

$$\frac{\partial L}{\partial w}=0 \Rightarrow w=\sum_{i=1}^{m}\alpha_i y_i x_i$$

$$\frac{\partial L}{\partial b}=0 \Rightarrow \sum_{i=1}^{m}-\alpha_i y_i = 0$$

$$\frac{\partial L}{\partial \xi_i}=0 \Rightarrow C\xi_i=\alpha_i$$

代入 L $\Rightarrow$ $\hat{L}(\alpha) = -\frac{1}{2}\|w\|^2 + \sum_{i=1}^{m}\alpha_i - \frac{1}{2C}\sum_{i=1}^{m}\alpha_i^2$

$$= -\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j x_i^T x_j + \sum_{i=1}^{m}\alpha_i - \frac{1}{2C}\sum_{i=1}^{m}\alpha_i^2$$

对偶问题为

$$\max_{\alpha} \quad -\frac{1}{2}\sum_i\sum_j \alpha_i\alpha_j y_i y_j x_i^T x_j + \sum_i \alpha_i - \frac{1}{2C}\sum_i \alpha_i^2$$

$$\text{s.t.} \quad \alpha_i \geq 0,\ i=1,\dots,m$$

$$\sum_i \alpha_i y_i = 0$$

Problem 4 RBF on Double Moon

Double Moon data is not linearly separable, ref to fig 1.

- Run 'main.py', the code is runnable and supposed to be bug free. If success, you can see fig 2. What you need to do is to answer the question and improve the code.
- Read the code, the code is not commented, you need to understand it by yourself. RBF in the code uses mean square loss and least square method to calculate weight and bias. Please improve code, *i.e.*, do one or more of the following:
  - add l_2 regularizetion (0.5 pnt)

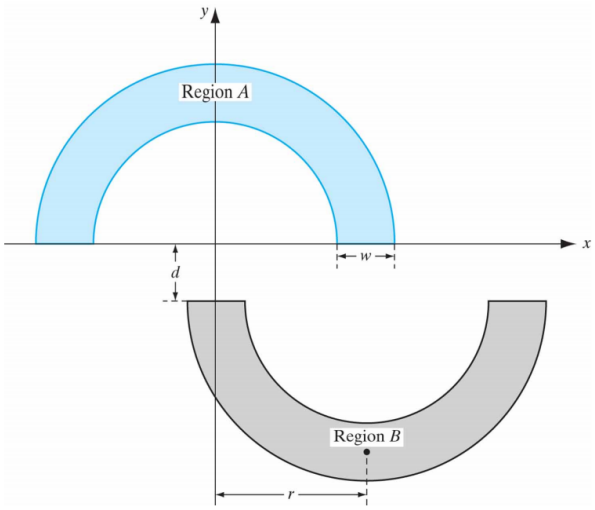


Figure 1: Double moon

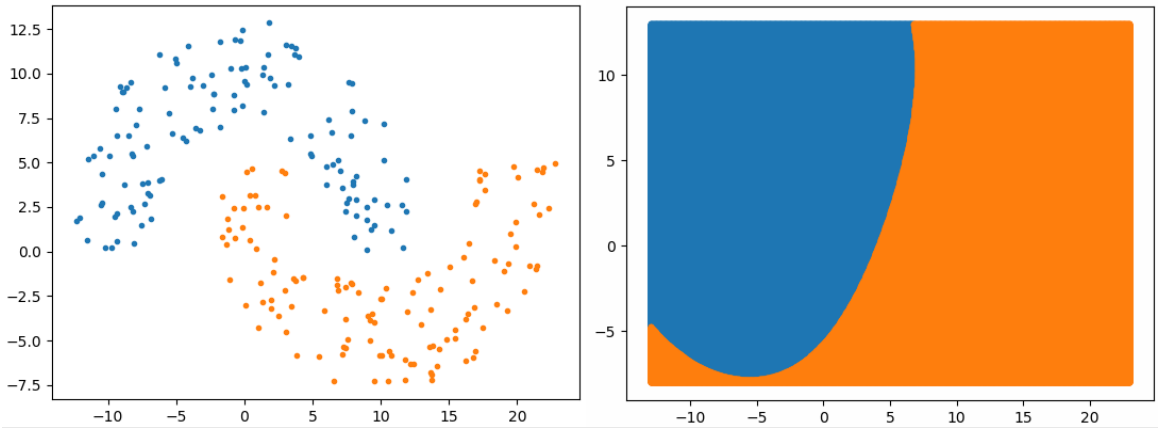


Figure 2: Results. Note that: the results are got from random seed = 16. **Left:** Training data. **Right:** Decision boundary.

加上 l_2 正则化项后，希望最小化的损失函数变为

$$E = \frac{1}{2}||y-d||_2^2 + \frac{1}{2}||w||_2^2$$

这时通过最小二乘法解出的权重为

$$w = 2[2X^TX + \text{diag}(1)]^{-1}X^Td$$

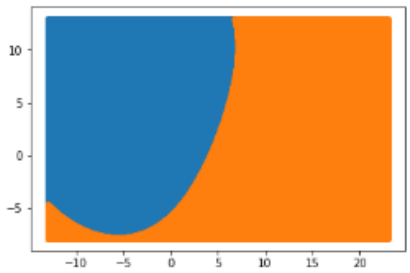


图 5: n_clusters=2

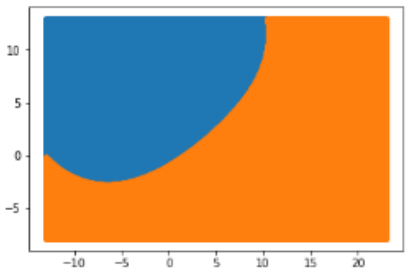


图 6: n_clusters=5

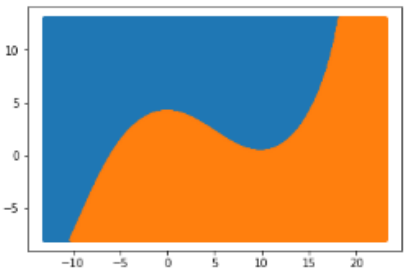


图 7: n_clusters=8

使用这个式子计算权重得到的分类结果与原来并没有很大的差别

– gradient based optimization method
可以不更新 center，sigma （2 pnt），也可以求梯度 center sigma （3 pnt）

若使用基于梯度的优化方法，那么网络中径向基函数的中心 x_j，方差 σ_j 和隐藏层到输出层的权重 w_j 都需要经过梯度下降法来获得。损失函数为均方误差，那么对上述变量求导得到

$$E = \frac{1}{2}||y-d||_2^2 \qquad y = b + \sum_j w_j \exp(-\frac{1}{2\sigma_j^2}||x-x_j||^2)$$

$$\frac{\partial E}{\partial w_j} = (y-d)\exp(-\frac{1}{2\sigma_j^2}||x-x_j||^2)$$

$$\frac{\partial E}{\partial \sigma_j} = (y-d)\frac{w_j||x-x_j||^2}{\sigma_j^3}\exp(-\frac{1}{2\sigma_j^2}||x-x_j||^2)$$

$$\frac{\partial E}{\partial x_j} = (y-d)\frac{w_j(x-x_j)}{\sigma_j^2}\exp(-\frac{1}{2\sigma_j^2}||x-x_j||^2)$$

$$\frac{\partial E}{\partial b} = (y-d)$$

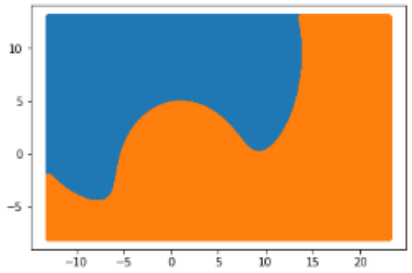


图 16: n_clusters=2, 20000 times, mse=6

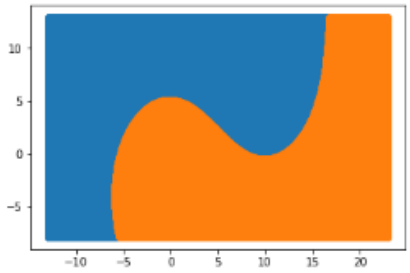


图 17: n_clusters=5, 2000 times, mse=2.76

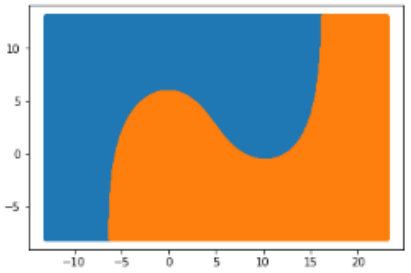


图 18: n_clusters=10, 200 times, mse=2.37

  - use logistic regression to calculate weight and bias. (Thus you may have to implement gradient based optimization method) 2 pnt

- (Bonus: answer one or more questions shown below) Do something extra surrounding the topics in this assignment, using the code you developed.

  For example, is there some other interesting question we could have asked? Is there any insightful visualization you can plot?

  Explain the principle of function 'cal_distmat', Profile and compare with other potential implementation of 'cal_distmat'. 2 pnt

函数 “cal_distmat” 的作用是计算输入 X 中每个子向量和输入 Y 中每个子向量间的距离。若 X 和 Y 都是一维向量，那么它们间的距离很容易计算，即

$$X = [x_1\ x_2\ ...\ x_m], Y = [y_1\ y_2\ ...\ y_m]$$
$$\text{distance}_{XY} = \sqrt{\sum_i (x_i - y_i)^2} = \sqrt{\sum_i (x_i^2 + y_i^2 - 2x_i y_i)}$$

当 X 和 Y 都含有多个子向量时（分别为 n 个样本点和 k 个中心点），就需要得到每两两组合间的距离，而代码就是按上述的计算方法实现的，即

$$X = \begin{bmatrix} x_{11} & \cdots & x_{1m} \\ x_{21} & \cdots & x_{2m} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nm} \end{bmatrix},\ Y = \begin{bmatrix} y_{11} & \cdots & y_{1m} \\ \vdots & \ddots & \vdots \\ y_{k1} & \cdots & y_{km} \end{bmatrix}$$

代码中 “(X ** 2).sum(axis=1).reshape(-1, 1) + (Y ** 2).sum(axis=1).reshape(1, -1)” 计算了两个平方项的和，结果为

$$\begin{bmatrix} \sum_{i=1}^m (x_{1i}^2 + y_{1i}^2) & \sum_{i=1}^m (x_{1i}^2 + y_{2i}^2) & \cdots & \sum_{i=1}^m (x_{1i}^2 + y_{ki}^2) \\ \sum_{i=1}^m (x_{2i}^2 + y_{1i}^2) & \sum_{i=1}^m (x_{2i}^2 + y_{2i}^2) & \cdots & \sum_{i=1}^m (x_{2i}^2 + y_{ki}^2) \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^m (x_{ni}^2 + y_{1i}^2) & \sum_{i=1}^m (x_{ni}^2 + y_{2i}^2) & \cdots & \sum_{i=1}^m (x_{ni}^2 + y_{ki}^2) \end{bmatrix}$$

矩阵 X 有 n 个子向量，矩阵 Y 有 k 个子向量，代码执行时会自动进行复制，最后得到的结果为 $n \times k$ 维，可见包含了所有两两子向量间的平方和。接下来的 “- 2 * np.dot(X, Y.T)” 则减去了向量间的相乘项，结果为

$$-2 \begin{bmatrix} \sum_{i=1}^m (x_{1i} y_{1i}) & \sum_{i=1}^m (x_{1i} y_{2i}) & \cdots & \sum_{i=1}^m (x_{1i} y_{ki}) \\ \sum_{i=1}^m (x_{2i} y_{1i}) & \sum_{i=1}^m (x_{2i} y_{2i}) & \cdots & \sum_{i=1}^m (x_{2i} y_{ki}) \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^m (x_{ni} y_{1i}) & \sum_{i=1}^m (x_{ni} y_{2i}) & \cdots & \sum_{i=1}^m (x_{ni} y_{ki}) \end{bmatrix}$$

故与上一个结果矩阵相加后，再开平方，即可得到 X 和 Y 中子向量间所有组合的距离。

Is the code robust to all exceptions and/or elegant with enough documents/comments? May comment for it and describe the training process of RBF. 1 pnt

Using ‘np.ndarray’ maybe lengthy, may try ‘np.matrix’ instead? 1 pnt

How to treat bias as weight by $[w^T, b]^T$ notation in gradient based optimization method? 1 pnt

What would happen if ‘train_pnts’ is not shuffled? 1 pnt

当 ‘train_pnts’ 没有进行打乱时, 就会按照原来的先正样本后负样本的顺序排列。

如果使用的是全批量学习, 每次都用所有的样本进行训练, 故对结果不会有什么影响。

但当我们使用的 batch size 为一个或者一小批样本的时候, 是否打乱就会对结果有较大的影响, 我的理解为, 如果总是用一批正样本对网络进行训练, 那么结果网络会对正样本分的很好,

接下来如果调换一下, 总是用负样本对网络进行训练, 那么网络又会偏向于对负样本分的很好, 最后会丢失起初对正样本分的很好的特性。

相比起来, 如果对样本进行打乱, 那么每次对网络进行训练时都含有正负样本, 每次网络都考虑要同时分好正负样本, 这时的分界面将会变得较为合理, 不过分偏向任何一方。

How about using other hyperparameter**s** (*e.g.* 'n_clusters')? 1 pnt

Can you implement kmeans from scratch? *etc.* 1 pnt