CSC 474 DATA MINING ASSIGNMENT #2 HONGHU LIN V00804639

Question # 1

3 Classes — rectangle, triangle, circle

50% training ← same distribution

50% testing ← Balanced

Q1.1 33% tri   33% tri   33% tr , ...

Error rate = 1 − Accuracy

actual data

	rect	tri	circle
rect	1 A	0 B	0 C
tri	1 D	0 E	0 F
circle	1 G	0 H	0 I

$$\text{Accuracy} = \frac{(A+E+I)}{(A+B+C+D+E+F+G+H+I)} = \frac{TN+TP}{\text{total}}$$

Accuracy $= 1/3$ error rate $= 1 - 1/3 = 2/3$

Q2.1

actual data

	rect	tri	circle
rect	0.7	0.3	0
tri	0.7	0.3	0
circle	0.7	0.3	0

$$\text{Accuracy} = \frac{(0.7 + 0.3 + 0)}{3} = 1/3$$

error rate $= 1 - 1/3 = 2/3$

Q3.1

	rect	tri	circle
rect	1	0	0
tri	1	0	0
circle	1	0	0

$$\text{Accuracy} = 1\left(\tfrac{1}{2}\right) + 0\left(\tfrac{1}{4}\right) + 0\left(\tfrac{1}{4}\right) = \tfrac{1}{2}$$

$$\text{error rate} = 1 - \tfrac{1}{2} = \tfrac{1}{2}.$$

Q4.1

	rect	tri	circle
rect	0.7	0.3	0
tri	0.7	0.3	0
circle	0.7	0.3	0

$$\begin{aligned}\text{Accuracy} &= 0.7\left(\tfrac{1}{2}\right) + 0.3\left(\tfrac{1}{4}\right) + 0\left(\tfrac{1}{4}\right)\\ &= \tfrac{7}{20} + \tfrac{3}{40}\\ &= \tfrac{17}{40}\end{aligned}$$

$$\text{error rate} = 1 - \tfrac{17}{40} = \tfrac{23}{40}$$

CSC 474 DATA MINING ASSIGNMENT #2 HONGHU LIN V00804639

<u>Question # 2</u>

Q2.1

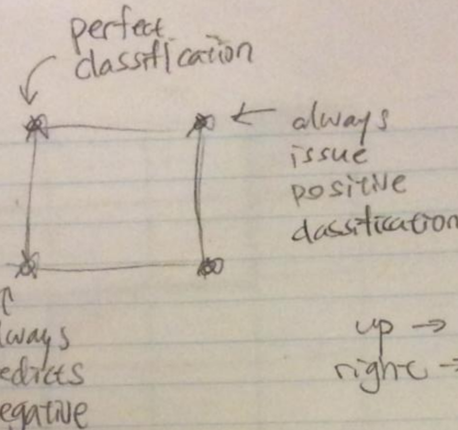
mark right → ↙ pass threshold

TP :

FP : mark wrong

FN : no pass threshold mark wrong

TN :



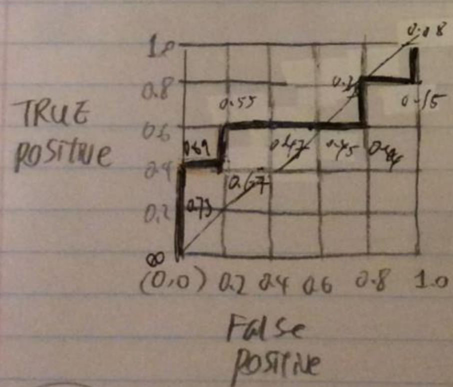
Precision = TP/(TP+FP)

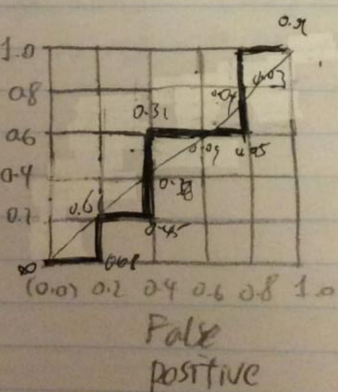
Recall = TP/(TP+FN)

up → positive

right → negative

	correct	not correct
selected	tp	fp
not selected	fn	tn





0.73	P	0.68	N
0.69	P	0.61	P
0.67	N	0.45	N
0.55	P	0.38	P
0.47	N	0.31	P
0.45	N	0.09	N
0.44	N	0.05	N
0.35	P	0.04	P
0.15	N	0.03	P
0.08	P	0.01	N

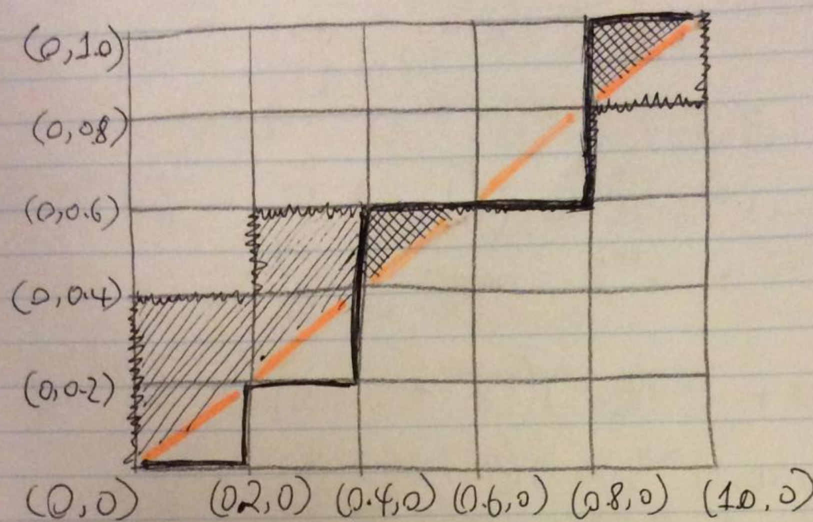
Q2.2

$$\text{Precision} = \frac{3}{3+1} = \frac{3}{4}$$

$$\text{Recall} = \frac{3}{3+2} = \frac{3}{5}$$

$$F = 2\left(\frac{P\cdot R}{P+R}\right) = 2\left(\frac{\frac{9}{20}}{\frac{15+12}{20}}\right) = 2\left(\frac{\frac{9}{20}}{\frac{27}{20}}\right) = 2\left(\frac{1}{3}\right) = 2/3$$

CSC 474 DATA MINING ASSIGNMENT #2 HONGHU LIN V00804639



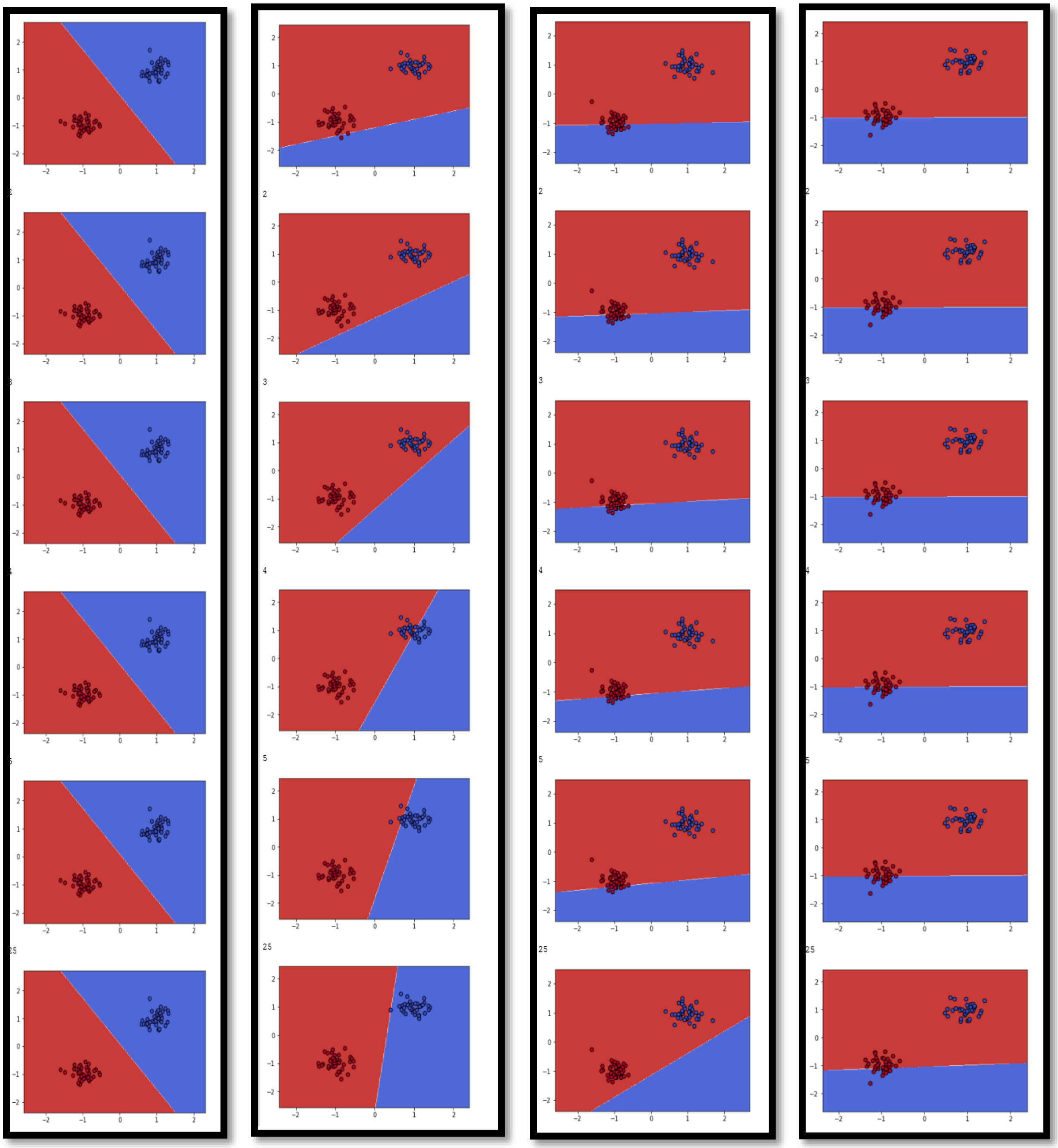
		Ranges	
	Classifier A	[1.0, 0.45] ⇒ 0.45 ~ 1.0	
	Classifier B	[0.38, 0.09] ⇒ 0.09 ~ 0.38	[0.04, 0.01] ⇒ 0.01 ~ 0.04

CSC 474 DATA MINING ASSIGNMENT #2 HONGHU LIN V00804639

Question # 3

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_blobs

np.random.seed(666)

def make_meshgrid(x, y, h=.02):
    x_min, x_max = x.min() - 1, x.max() + 1
    y_min, y_max = y.min() - 1, y.max() + 1
    xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
                         np.arange(y_min, y_max, h))
    return xx, yy


def plot_boundary(clf, X_train, Y_train, xx, yy):
    Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)
    plt.contourf(xx, yy, Z, cmap=plt.cm.coolwarm)
    plt.scatter(X_train[:, 0], X_train[:, 1], c=Y_train,
                cmap=plt.cm.coolwarm,
                edgecolors='k')
    plt.show()


class Perceptron():

    def __init__(self, learning_rate=0.01):
        self.lr = learning_rate
        self.weights = np.array([[5., 0., 5.]])

    def fit_epoch(self, X, Y):
        X = np.hstack((np.ones((X.shape[0], 1)), X))
        shuff = np.random.permutation(len(Y))
        X, Y = X[shuff], Y[shuff]

        a = np.hstack((np.ones((X.shape[0], 0)), X))
        a = np.sign(np.dot(a, self.weights.T))

        for iter in range (len(a)):
            if a[iter] != Y[iter] :
                self.weights = self.weights + self.lr * ((Y[iter]-a[iter]) * X[iter])


    def predict(self, X):
        asd = np.hstack((np.ones((X.shape[0], 1)), X))
        return np.sign(np.dot(asd, self.weights.T))


if __name__ == '__main__':
    N, M = 40, 2
    X_train = np.r_[np.random.randn(N, M) + [1, 1], np.random.randn(N, M) + [10, 10]]
    X_train = (X_train - X_train.mean(axis=0)) / X_train.std(axis=0)
    Y_train = np.array([1]*N + [-1]*N)

    xx, yy = make_meshgrid(X_train[:, 0], X_train[:, 1])

    n_epochs = 25

    clf = Perceptron(learning_rate=0.001)
    for iter in range(n_epochs):
        clf.fit_epoch(X_train,Y_train)

        if ((iter<5) or (iter==24)) :
            print (iter + 1)
            plot_boundary(clf, X_train, Y_train, xx, yy)
```

CSC 474 DATA MINING ASSIGNMENT #2 HONGHU LIN V00804639



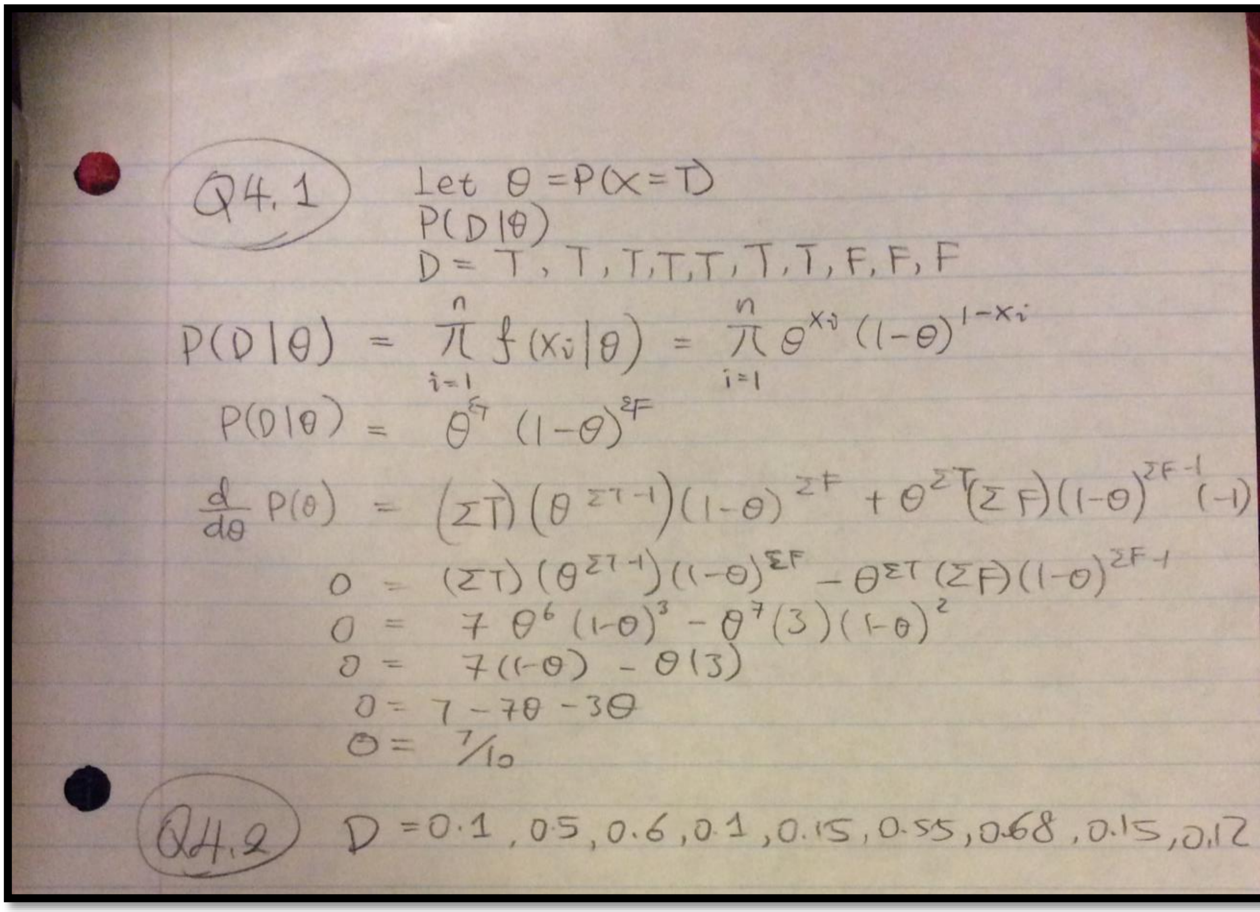
0.1 **0.01** **0.001** **0.0001**

3.1) as iteration increases, the boundary gets more accurate. Clearly the final guessed boundary line after the fifth iterations does separate the training data points correctly.

3.2) the observation for different learning rate is, as learning rate grows, the change is getting more obvious, whereas when the LR is small. The change is small, decision boundary is hard to divide.

CSC 474 DATA MINING ASSIGNMENT #2 HONGHU LIN V00804639

Question # 4



Q4.1

Let $\theta = P(X=T)$

$P(D|\theta)$

$D = T, T, T, T, T, T, T, F, F, F$

$$P(D|\theta) = \prod_{i=1}^{n} f(x_i|\theta) = \prod_{i=1}^{n} \theta^{x_i}(1-\theta)^{1-x_i}$$

$$P(D|\theta) = \theta^{\Sigma T}(1-\theta)^{\Sigma F}$$

$$\frac{d}{d\theta}P(\theta) = (\Sigma T)(\theta^{\Sigma T-1})(1-\theta)^{\Sigma F} + \theta^{\Sigma T}(\Sigma F)(1-\theta)^{\Sigma F-1}(-1)$$

$$0 = (\Sigma T)(\theta^{\Sigma T-1})(1-\theta)^{\Sigma F} - \theta^{\Sigma T}(\Sigma F)(1-\theta)^{\Sigma F-1}$$

$$0 = 7\theta^{6}(1-\theta)^{3} - \theta^{7}(3)(1-\theta)^{2}$$

$$0 = 7(1-\theta) - \theta(3)$$

$$0 = 7 - 7\theta - 3\theta$$

$$\theta = 7/10$$

Q4.2

$D = 0.1, 0.5, 0.6, 0.1, 0.15, 0.55, 0.68, 0.15, 0.12$

ANSWER TO 4.2

Let's assume the numbers are equally split to

- ⇨ X > 0.5
- ⇨ X <= 0.5

	A	B
P(X > 0.5)	0	3/21
P(X <= 0.5)	1	18/21

MLE A	P(D|D<=0.5) = (22/23)^6 *(1/23)^3 = 6.294852537 * 10^(-5)
MLE B	P(D|D<=0.5) = (18/21)^6 *(3/21)^3 = 1.156179174 * 10^(-5)

1.156179174 * 10^ (-5)   IS **GREATER** THAN 6.294852537 * 10^ (-5)

Therefore model B is well explained for this data.