Machine Learning **(Due: 25/12/20)**

Homework #5: Referred Answers

Instructor: Lin, Hsuan Tien *Name:* Hao-Cheng Lo, *Id:* D08227104

Problem 1:

[d] is correct.

The problem is defined as:

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ \text{s.t.} \quad & y_n(\mathbf{w}^T\phi(\mathbf{x}) + b) \geq 1 \ \forall n \end{aligned}$$

Due to the constraints, we should solve the inequality system:

$$\begin{bmatrix} -1 & 2 & -4 & -1 \\ 1 & 0 & 0 & 1 \\ -1 & -2 & -4 & -1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ b \end{bmatrix} \geq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Accordingly, $w_1 \geq 1 - b,\ w_2 \geq 0,\ w_3 \leq -0.5$ and b is a free variable. For minimizing the length of $\mathbf{w}$, let $b = 1$ and we'll get $w_1 = 0,\ w_2 = 0,\ w_3 = -0.5$ which materializes the minimal of length of $\mathbf{w}$.

Problem 2:

[b] is correct.

$\text{margin}(b, \mathbf{w}) = \dfrac{1}{||\mathbf{w}||} = 1/0.5 = 2.$

Problem 3:

[e] is correct.

For 1D large-margin SVM, the margin is defined as a half of distance between support vectors. The support vectors in the problem would be x_M and x_{M+1}. Thus, the margin is $\frac{1}{2}(x_{M+1} - x_M)$.

Problem 4:

[a] is correct.

For the 2 dichotomies that $\{x_1 = +1, x_2 = -1\}$ and $\{x_1 = -1, x_2 = +1\}$, there always a margin-perceptron with margin ρ being able to produce these dichotomies given any distance of x_1 and x_2. The productivity of remaining dichotomies depend on the margin ρ. For $\{x_1 = -1, x_2 = -1\}$, x_1 and x_2 should be both at the LHS of the margin-perceptron with proprobility $(1-2\rho)^2$. For $\{x_1 = +1, x_2 = +1\}$, x_1 and x_2 should be both at the RHS of the margin-perceptron with proprobility $(1-2\rho)^2$. Hence, the total expected number of dichotomies is $2 + 2(1-2\rho)^2$.

Problem 5:

[c] is correct.

The Lagrangian of the uneven SVM optimization problem is:

$\mathcal{L}(\mathbf{w}, b, \alpha) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \sum_{n=1}^{N} \alpha_n(\rho_+[\![y_n = +1]\!] + \rho_-[\![y_n = -1]\!] - y_n(\mathbf{w}^T\mathbf{x}_n + b))$, where $\alpha \geq 0$.

Minimize the Lagrangian with respect to $\mathbf{w}$ and b by taking the their gradients and then setting them equal to 0.

$0 = \nabla_{\mathbf{w}}\mathcal{L} = \mathbf{w} - \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n \rightarrow \mathbf{w} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n$

$0 = \dfrac{\partial \mathcal{L}}{\partial b} = -\sum_{n=1}^{N} \alpha_n y_n \rightarrow \sum_{n=1}^{N} \alpha_n y_n = 0$

The objective function for the dual is (where $\alpha \geq 0$ and $\sum_{n=1}^{N} \alpha_n y_n = 0$):

$\max_\alpha \theta(\alpha) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \sum_{n=1}^{N} \alpha_n(\rho_+[\![y_n = +1]\!] + \rho_-[\![y_n = -1]\!]) - \mathbf{w}^T\mathbf{w}$

$\rightarrow \max_\alpha \theta(\alpha) = -\frac{1}{2}||\sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n||^2 + \sum_{n=1}^{N} \alpha_n(\rho_+[\![y_n = +1]\!] + \rho_-[\![y_n = -1]\!])$

$\rightarrow \min_\alpha \theta(\alpha) = \frac{1}{2}||\sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n||^2 - \sum_{n=1}^{N} \alpha_n(\rho_+[\![y_n = +1]\!] + \rho_-[\![y_n = -1]\!])$

Problem 6:

[idk] is correct.

All symbols with superscript h are derived from the even-margin SVM; all symbols with superscript g are derived from the uneven-margin SVM. Given that geographically, $w^g = \frac{\rho_+ + \rho_-}{2} w^h$ and the corresponding support vectors of h and g are identical, $w^g = \sum_{n=1}^{N} \alpha_n^g y_n x_n = \frac{\rho_+ + \rho_-}{2} \sum_{n=1}^{N} \alpha_n^h y_n x_n = \sum_{n=1}^{N} (\frac{\rho_+ + \rho_-}{2} \alpha_n^h) y_n x_n$.

Problem 7:

[d] is correct.

Let $K = \begin{bmatrix} 0.9 & 0.6 \\ 0.6 & 0.4 \end{bmatrix}$, $K' = \log_2 K = \begin{bmatrix} -0.152 & -0.736 \\ -0.736 & -1.321 \end{bmatrix}$
whose eigenvalues are 0.203 and -1.677. Hence K' is not a PSD thus not a valid kernel.

Problem 8:

[c] is correct.

$$
\begin{aligned}
||\phi(\mathbf{x}) - \phi(\mathbf{x}')||^2 &= \phi(\mathbf{x})^T\phi(\mathbf{x}) - 2\phi(\mathbf{x})^T\phi(\mathbf{x}') + \phi(\mathbf{x}')^T\phi(\mathbf{x}') \\
&= \exp\left(-\gamma||\mathbf{x} - \mathbf{x}||^2\right) - 2\exp\left(-\gamma||\mathbf{x} - \mathbf{x}'||^2\right) + \exp\left(-\gamma||\mathbf{x}' - \mathbf{x}'||^2\right) \\
&= 2 - 2\exp\left(-\gamma||\mathbf{x} - \mathbf{x}'||^2\right) \\
&\leq 2 - 2 \times 0 = 2
\end{aligned} \tag{0.1}
$$

Problem 9:

[idk] is correct.

Given $\mathrm{E}_{in}(\hat{h}) = 0$,

$$\begin{aligned} N &= \sum_{i=1}^{N} y_i \mathrm{sign}\Big(\sum_{n=1}^{N} y_n K(\mathbf{x}_n, \mathbf{x}_i)\Big) \\ &\leq \sum_{i=1}^{N} y_i \sum_{n=1}^{N} y_n K(\mathbf{x}_n, \mathbf{x}_i) \\ &\leq \sum_{i=1}^{N} \sum_{n=1}^{N} K(\mathbf{x}_n, \mathbf{x}_i) \\ &\leq N^2 \exp\left(-\gamma\epsilon^2\right) \\ \rightarrow N^{-1} &\leq \exp\left(-\gamma\epsilon^2\right) \\ \rightarrow \frac{\ln(N)}{\epsilon^2} &\geq \gamma \end{aligned} \tag{0.2}$$

Problem 10:

[c] is correct.

$\mathbf{w}_{t+1} := \mathbf{w}_t + y_{n(t)}\phi(\mathbf{x}_{n(t)}) = \sum_{i=1}^{N} \alpha_i \phi(\mathbf{x}_i) + y_{n(t)}\phi(\mathbf{x}_{n(t)}) = \sum_{i=1}^{N}(\alpha_i + y_{n(t)}[\![i = n(t)]\!])\phi(\mathbf{x}_i)$.

Problem 11:

[a] is correct.

$\mathbf{w}_t^T \phi(\mathbf{x}) = \sum_{n=1}^{N} \alpha_{t,n} \phi(\mathbf{x}_n)^T \phi(\mathbf{x}) = \sum_{n=1}^{N} \alpha_{t,n} [\phi(\mathbf{x}_n)^T \phi(\mathbf{x})] = \sum_{n=1}^{N} \alpha_{t,n} K(\mathbf{x}_n, \mathbf{x})$.

Problem 12:

[b] is correct.

Due to the complementary slackness and $\alpha_n = C$, $b = y_s - y_s\xi_s - \mathbf{w}^T\mathbf{z}_s$.

To find the upper bound of b, for those $y_s = 1$, $b = 1 - \xi_s - \mathbf{w}^T\mathbf{z}_s < 1 - \mathbf{w}^T\mathbf{z}_s$. The tightest one would be $\min_{n:y_n>0}(1 - \mathbf{w}^T\mathbf{z}_s) = \min_{n:y_n>0}(1 - \sum_{m=1}^{N} y_m\alpha_m K(\mathbf{x}_n, \mathbf{x}_m))$.

Problem 13:

[e] is correct.

The primal Lagrangian for the problem is:

$\mathcal{L}(\mathbf{w}, b, \xi, \alpha) = \dfrac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{n=1}^{N}\xi_n^2 - \sum_{n=1}^{N}\alpha_n[y_n(\mathbf{w}^T\mathbf{z}_n + b) - 1 + \xi_n]$, where $\alpha \geq 0$.

Minimize the Lagrangian with respect to $\mathbf{w}$, ξ, and b by taking the their gradients and then setting them equal to 0.

$0 = \nabla_{\mathbf{w}}\mathcal{L} = \mathbf{w} - \sum_{n=1}^{N}\alpha_n y_n \mathbf{z}_n$

$0 = \nabla_{\xi}\mathcal{L} = 2C\xi - \alpha$

$0 = \dfrac{\partial\mathcal{L}}{\partial b} = -\sum_{n=1}^{N}\alpha_n y_n$

Substitute such relations into the primal to obtain the dual objective function (where $\alpha \geq 0$ and $\sum_{n=1}^{N}\alpha_n y_n = 0$), whose derivation is similar to problem 5:

$$\begin{aligned}
\mathcal{L}(\mathbf{w}, b, \xi, \alpha) &= -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_i\alpha_j y_i y_j \mathbf{z}_i^T\mathbf{z}_j + \sum_{n=1}^{N}\alpha_n - \frac{1}{2}\sum_{n=1}^{N}\frac{\alpha_n^2}{2C} \\
&= -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_i\alpha_j y_i y_j[K(\mathbf{x}_n, \mathbf{x}_m) + \frac{1}{2C}[\![n = m]\!]] + \sum_{n=1}^{N}\alpha_n \qquad (0.3) \\
\min_{\alpha} -\mathcal{L}(\mathbf{w}, b, \xi, \alpha) &= \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_i\alpha_j y_i y_j[K(\mathbf{x}_n, \mathbf{x}_m) + \frac{1}{2C}[\![n = m]\!]] - \sum_{n=1}^{N}\alpha_n
\end{aligned}$$

Problem 14:

[e] is correct.

By $0 = 2C\xi - \alpha$, we will get $\xi = \dfrac{1}{2C}\alpha$.

In [1]:
```
import numpy as np
from random import sample
from svmutil import *
```

In [22]:
```
y, x = svm_read_problem('hw5_train.txt')
yt, xt = svm_read_problem('hw5_test.txt')
```

Q15

[d] = 8.5

In [3]:
```
y15 = [1 if i == 3 else -1 for i in y]
x15 = x
```

In [4]:
```
prob15  = svm_problem(y15, x15)
param15 = svm_parameter('-t 0 -c 10')
model15 = svm_train(prob15, param15)
```

In [5]:
```
svc15 = model15.get_sv_coef()
sv15 = model15.get_SV()
print(model15.get_nr_sv())
```

```
500
```

In [6]:
```
#sv15 = [[float(v) for k,v in i.items()] for i in sv15]
#svc15 = [float(i[0]) for i in svc15]
```

In [7]:
```
w = []
for i in range(1,37):
    wi = 0
    for j in range(500):
        if i in sv15[j].keys():
            wi += float(sv15[j][i])*float(svc15[j][0])
    w.append(wi)
w = np.array(w)
np.sqrt(w.T@w)
```

Out[7]:
```
8.45708429836768
```

Q16

[b] = 2 versus not 2

Q17

[c] = 700

In [8]:
```
for t in range(1,6):
    y16 = [1 if i == t else -1 for i in y]
    x16 = x
    prob16  = svm_problem(y16, x16)
    param16 = svm_parameter('-t 1 -g 1 -r 1 -d 2 -c 10')
    model16 = svm_train(prob16, param16)
    p_label, p_acc, p_val = svm_predict(y16, x16, model16)
    print(p_acc)
    nr_sv = model16.get_nr_sv()
    print(nr_sv)
```

```
Accuracy = 99.9324% (4432/4435) (classification)
(99.93235625704622, 0.002705749718151071, 0.9963197311370662)
145
Accuracy = 100% (4435/4435) (classification)
(100.0, 0.0, 1.0)
87
Accuracy = 97.7678% (4336/4435) (classification)
(97.76775648252537, 0.08928974069898535, 0.8750113320392298)
433
Accuracy = 95.9865% (4257/4435) (classification)
(95.98647125140924, 0.16054114994363022, 0.5651265925930234)
712
Accuracy = 99.3236% (4405/4435) (classification)
(99.32356257046223, 0.02705749718151071, 0.929443549395688)
259
```

Q18

[d] or [e]

In [9]:
```
for t in range(-2,3):
    y18 = [1 if i == 6 else -1 for i in y]
    x18 = x
    prob18  = svm_problem(y18, x18)
    param18 = svm_parameter('-t 2 -g 10 -c {}'.format(10**t))
    model18 = svm_train(prob18, param18)
    yt18 = [1 if i == 6 else -1 for i in yt]
    xt18 = xt
    p_label, p_acc, p_val = svm_predict(yt18, xt18, model18)
    ACC, MSE, SCC = evaluations(yt18, p_label)
    print(ACC, MSE)
```

```
Accuracy = 76.5% (1530/2000) (classification)
76.5 0.94
Accuracy = 83.65% (1673/2000) (classification)
83.65 0.654
Accuracy = 89.35% (1787/2000) (classification)
89.35 0.426
Accuracy = 90.3% (1806/2000) (classification)
90.3 0.388
Accuracy = 90.3% (1806/2000) (classification)
90.3 0.388
```

Q19

[b]

In [10]:
```
for t in range(-1,4):
    y18 = [1 if i == 6 else -1 for i in y]
    x18 = x
    prob18  = svm_problem(y18, x18)
    param18 = svm_parameter('-t 2 -g {} -c 0.1'.format(10**t))
    model18 = svm_train(prob18, param18)
    yt18 = [1 if i == 6 else -1 for i in yt]
    xt18 = xt
    p_label, p_acc, p_val = svm_predict(yt18, xt18, model18)
    ACC, MSE, SCC = evaluations(yt18, p_label)
    print(ACC, MSE)
```

```
Accuracy = 90.15% (1803/2000) (classification)
90.14999999999999 0.394
Accuracy = 93% (1860/2000) (classification)
93.0 0.28
Accuracy = 83.65% (1673/2000) (classification)
83.65 0.654
Accuracy = 76.5% (1530/2000) (classification)
76.5 0.94
Accuracy = 76.5% (1530/2000) (classification)
76.5 0.94
```

Q20

[b]

In [27]:
```
gammas = []
for exp in range(1000):
    temp = 0
    y18 = np.array([1 if i == 6 else -1 for i in y])
    x18 = np.array(x)
    indices = sample(range(len(y18)),200)
    x18v = x18[indices]
    x18 = np.delete(x18,indices)
    y18v = y18[indices]
    y18 = np.delete(y18,indices)
    for t in range(-1,4):
        prob18  = svm_problem(y18, x18)
        param18 = svm_parameter('-t 2 -g {} -c 0.1'.format(10**t))
        model18 = svm_train(prob18, param18)
        p_label, p_acc, p_val = svm_predict(y18v, x18v, model18)
        if p_acc[0] > temp:
            temp = p_acc[0]
            tempans = 10**t
    gammas.append(tempans)
```

```
Accuracy = 78.5% (157/200) (classification)
Accuracy = 78.5% (157/200) (classification)
Accuracy = 94.5% (189/200) (classification)
Accuracy = 94% (188/200) (classification)
Accuracy = 82% (164/200) (classification)
Accuracy = 75% (150/200) (classification)
Accuracy = 75% (150/200) (classification)
Accuracy = 91.5% (183/200) (classification)
Accuracy = 94% (188/200) (classification)
Accuracy = 86% (172/200) (classification)
Accuracy = 79.5% (159/200) (classification)
Accuracy = 79.5% (159/200) (classification)
Accuracy = 93.5% (187/200) (classification)
Accuracy = 94% (188/200) (classification)
Accuracy = 81.5% (163/200) (classification)
Accuracy = 74.5% (149/200) (classification)
Accuracy = 74.5% (149/200) (classification)
Accuracy = 90.5% (181/200) (classification)
Accuracy = 93.5% (187/200) (classification)
Accuracy = 88% (176/200) (classification)
```

In [30]:
```
import statistics as st
st.mode(gammas)
```

Out[30]: 1