COMS 4721 HW #2
Machine Learning for Data Science

Name: Sung-Yen Liu
UNI: sl3763

Problem 1.

$$P(y|x, w_1, \dots, w_K) = \prod_{i=1}^{k} \left(\frac{e^{x^T w_i}}{\sum_{j=1}^{k} e^{x^T w_j}} \right)^{1(y=i)} \Leftrightarrow \left(P(y=i|x, w_1, \dots, w_K) = \frac{e^{x^T w_i}}{\sum_{j=1}^{k} e^{x^T w_j}} \right)$$

likelihood of generating each data

$\Rightarrow$ for $(x_1, y_1), \dots, (x_n, y_n)$, the likelihood $= \prod_{m=1}^{n} \prod_{i=1}^{k} \left(\frac{e^{x^T w_i}}{\sum_{j=1}^{k} e^{x^T w_j}} \right)^{1(y=i)}$ n: number of data, k: number of classes

$$\Rightarrow \text{log-likelihood } \mathcal{L} = \sum_{m=1}^{n} \left(\sum_{i=1}^{k} [y=i] x^T w_i - \log \sum_{j=1}^{k} (e^{x^T w_j}) \right)$$

$$\nabla_{w_i} \mathcal{L} = \frac{\partial}{\partial w_i} \mathcal{L} = \sum_{m=1}^{n} \left([y=i] x^T - \frac{e^{x^T w_i}}{\sum_{j=1}^{k} (e^{x^T w_j})} \cdot x^T \right) = \sum_{m=1}^{n} x^T \left([y=i] - \frac{e^{x^T w_i}}{\sum_{j=1}^{k} (e^{x^T w_j})} \right) = \sum_{m=1}^{n} x^T ([y=i] - P(y=i|x_i))$$

$$\nabla_{w_i}^2 \mathcal{L} = \frac{\partial}{\partial w_l} (\nabla_{w_i} \mathcal{L}) = \frac{\partial}{\partial w_l} \left(\sum_{m=1}^{n} x^T ([y=i] - P(y=i|x)) \right)$$

$$= \sum_{m=1}^{n} \frac{\partial \left(-P(y=i|x) \right)}{\partial w_l} x^T = \sum_{m=1}^{n} \left(\frac{e^{x^T w_l}}{\sum_{j=1}^{k} (e^{x^T w_j})} \right) \left(1 - \frac{e^{x^T w_l}}{\sum_{j=1}^{k} (e^{x^T w_j})} \right) x \cdot x^T$$

2. By framing original problem to probability question

$$\phi_t(u) = \frac{1}{(2\pi\beta')^{d/2}} \exp\left\{\frac{-\|u-t\|^2}{2\beta'}\right\} = \frac{1}{\beta'^{\frac{d-1}{2}}} \mathcal{N}(t|u,\Sigma)$$

$$\phi_t(v) = \frac{1}{(2\pi\beta')^{d/2}} \exp\left\{\frac{-\|v-t\|^2}{2\beta'}\right\} = \frac{1}{\beta'^{\frac{d-1}{2}}} \mathcal{N}(t|v,\Sigma)$$

$$\Rightarrow k(u,v) = \int \phi_t(u)\phi_t(v)\,dt = \frac{1}{\beta'^{d-1}} \int \mathcal{N}(t|u,\Sigma)\mathcal{N}(t|v,\Sigma)$$

$$\text{for } \mathcal{N}(t|u,\Sigma)\mathcal{N}(t|v,\Sigma) = \frac{1}{(2\pi)^{\frac{d}{2}}|2\Sigma|^{\frac{1}{2}}} \exp\left\{-\frac{(u-v)^T(2\Sigma)^{-1}(u-v)}{2}\right\} \cdot \mathcal{N}(t|\mu,\sigma^2)$$

$$k(u,v) = \frac{1}{\beta'^{d-1}} \times \frac{1}{(2\pi)^{\frac{d}{2}}|2\Sigma|^{\frac{1}{2}}} \exp\left\{-\frac{(u-v)^T(2\Sigma)^{-1}(u-v}{2}\right\} \int \mathcal{N}(t|\mu,\sigma^2)\,dt$$

$$= \frac{1}{\beta'^{d-1}} \times \frac{1}{(2\pi)^{\frac{d}{2}}(2\beta')^{\frac{1}{2}}} \exp\left\{-\frac{(u-v)^T(u-v)}{2\times 2\beta'}\right\}$$

$$\Rightarrow \alpha = \frac{1}{\sqrt{2}} \times \frac{1}{\beta'^{\frac{d-1}{2}}} \times \frac{1}{(2\pi)^{\frac{d}{2}}}, \quad \beta = 4\beta'$$

Problem 3

a) K-NN classifier

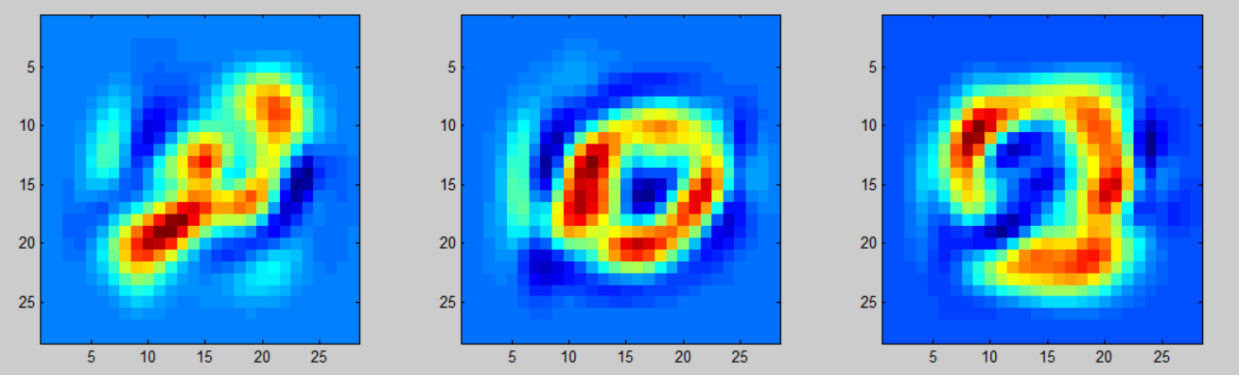
Code: (Algorithm part)

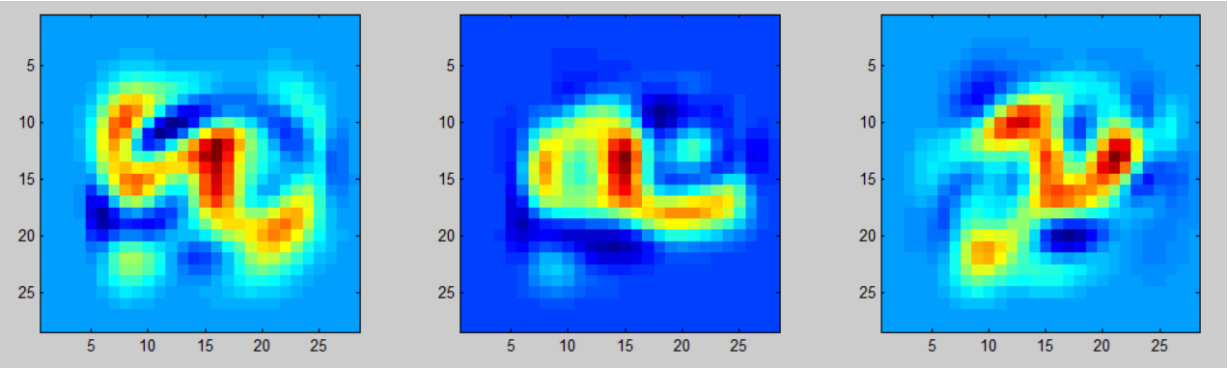
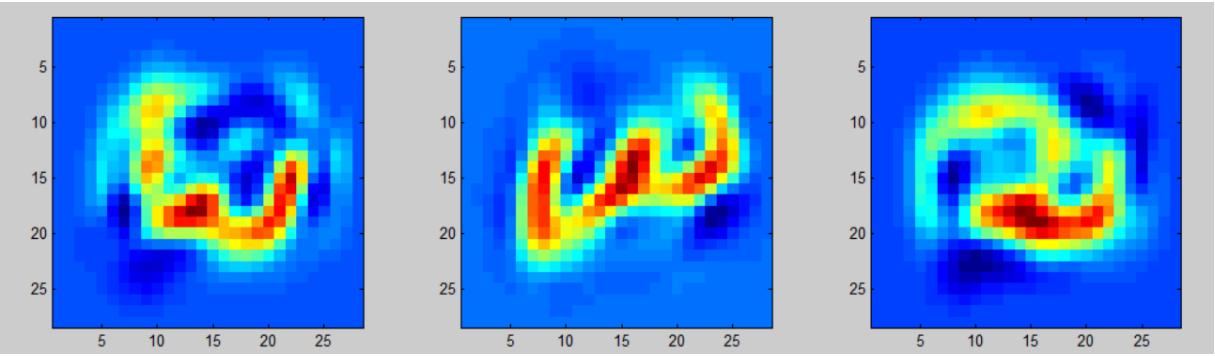
```
% problem 3.1
conf = zeros(10,10,5);
idx = 1;

for ts = 1:500
        xte = Xtest(:,ts);
        xterp = repmat(xte,[1 5000]);
        diff = Xtrain - xterp;
        diff = sum(diff.*diff);

        [dist, order] = sort(diff);
        knn = label_train(order(1:5));
        for k = 1:5
                countMax = mode(knn(1:k));
                conf(label_test(ts)+1,countMax+1,k) = conf(label_test(ts)+1,countMax+1,k)+1;
        end
end
```

Confusion Matrix:

k	1	2	3	4	5
Accuracy	0.948	0.93	0.938	0.946	0.946

Misclassified:

K=1	11	20	457
True class	0	0	9
Predicted	5	3	6
			
K=3	194	206	261
True class	3	4	5

Predicted	1	9	4
			
K=5	155	166	323
True class	3	3	6
Predicted	7	8	4
			

b) Multivariate Gaussian

Code: (Algorithm part)

```
% problem 3.2
mu = zeros(20,10);
cov = zeros(20,20,10);
invcov = zeros(20,20,10);
detcov = zeros(10,1);
conf = zeros(10,10);

for lb = 0:9
        xtr = Xtrain(:,label_train==lb);
        mu(:,lb+1) = mean(xtr,2);
        murep = repmat(mu(:,lb+1),[1 500]);
        cov(:,:,lb+1) = ((xtr-murep)*(xtr-murep)')./500;
        invcov(:,:,lb+1) = inv(cov(:,:,lb+1));
        detcov(lb+1) = det(cov(:,:,lb+1));
end
```

```
for ts = 1:500
        x = Xtest(:,ts);
        plugin = zeros(10,1);
        for lb = 1:10
                temp = reshape(invcov(:,:,lb),20,20);
                plugin(lb) = (1/sqrt(detcov(lb)))*exp(-0.5*(x-mu(:,lb))'*temp*(x-mu(:,lb)));
        end
        maxidx = find(plugin==max(plugin));
        conf(label_test(ts)+1,maxidx) = conf(label_test(ts)+1,maxidx) + 1;
end
```

Based on the derivation which has been mentioned in lecture slide for Multivariate Gaussian parameter:

$$\mu_y = \frac{1}{n_y}\sum_{i=1}^{n} 1[\![y_i = y]\!]\, x_i$$

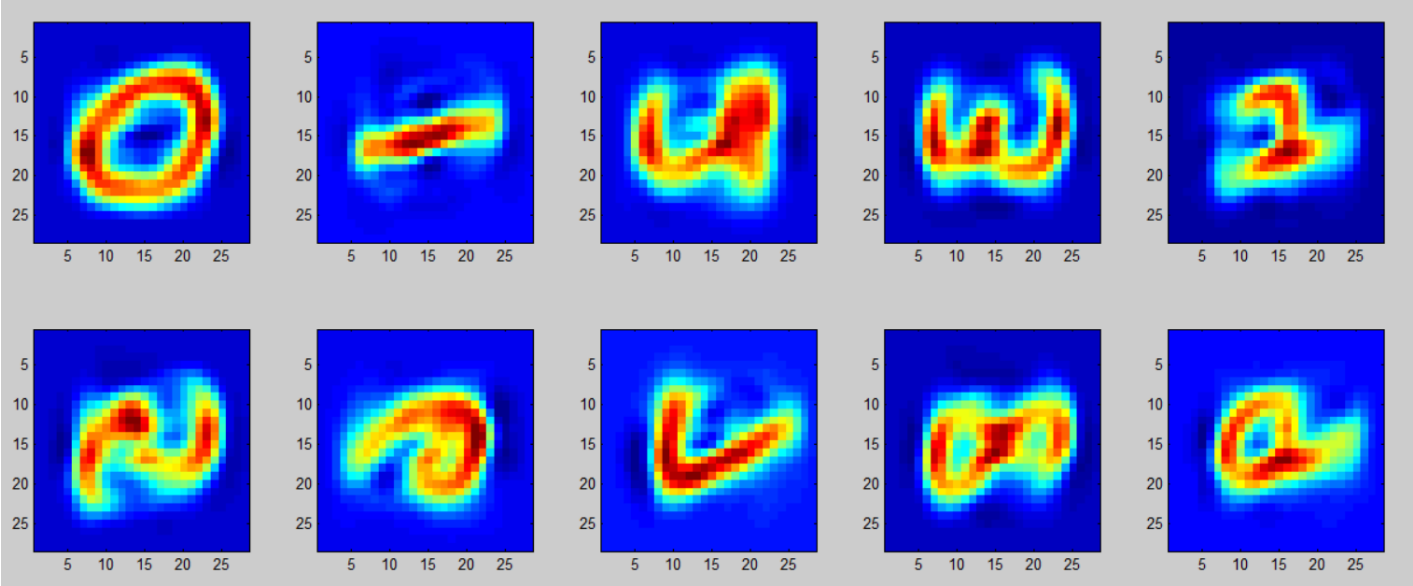
$$\Sigma_y = \frac{1}{n_y}\sum_{i=1}^{n} 1[\![y_i = y]\!]\,(x_i - \mu_y)(x_i - \mu_y)^T$$

Confusion Matrix:

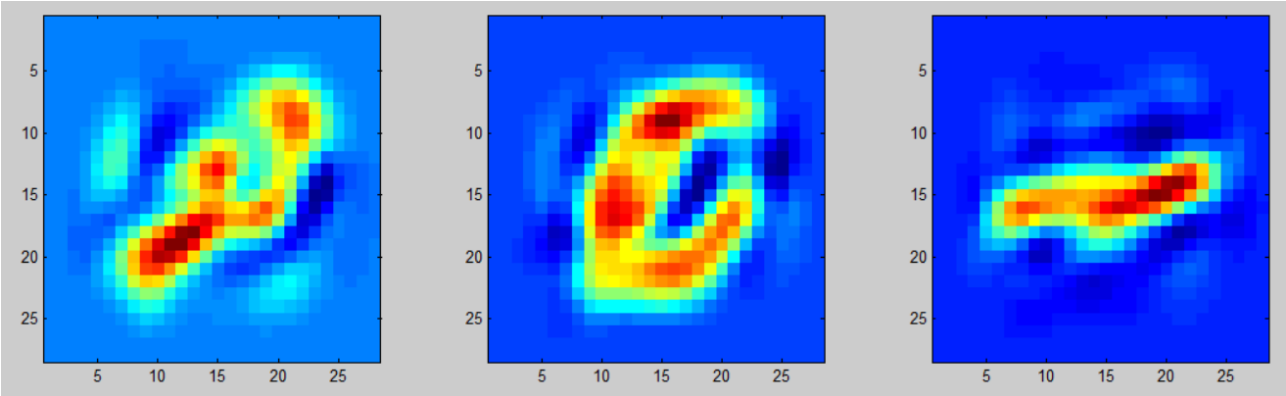
	1	2	3	4	5	6	7	8	9	10
1	48	0	0	1	0	1	0	0	0	0
2	0	49	0	0	0	0	0	0	1	0
3	0	0	48	0	1	0	1	0	0	0
4	0	0	1	47	0	0	0	0	2	0
5	0	0	0	0	48	0	0	0	1	1
6	0	0	0	1	0	45	2	0	1	1
7	0	0	0	0	1	5	43	0	0	1
8	0	0	2	0	2	0	0	46	0	0
9	0	0	1	0	0	1	0	0	47	1
10	1	0	0	0	2	0	0	0	0	47

Accuracy: 0.936

Mean of each digit:



Misclassified example:

True	0	0	1
Predict	3	5	8
			

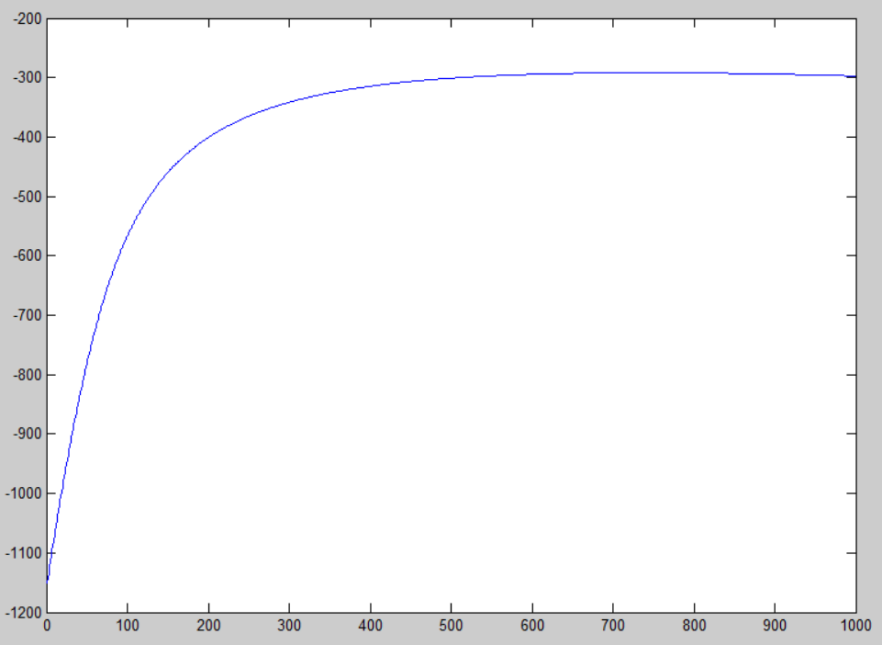
c) Multi-class Logistic Regression

Code: (Algorithm part)

```
% problem 3.3
conf = zeros(10,10);
w = zeros(21,10);
nextw = zeros(21,10);
smxbase = zeros(5000,10);
Xtrain(21,:) = 1;
Xtest(21,:) = 1;
```

```
for ite = 1:1000
        smxbase = sum(exp(Xtrain'*w),2); % 5000*1, sum of the value generate by 10 classifiers (each data0)
        for j = 0:9
                xtr = Xtrain(:,label_train==j); % 21*500, for a specific class
                tmp = repmat(exp((xtr')*w(:,j+1))./ smxbase(j*500+1:(j+1)*500), [1 21]); % coefficient
                like = (tmp.*(xtr')); % 500*21
                nextw(:,j+1) = w(:,j+1) + 0.1*(sum(xtr'-like))'/5000; % w(class, t+1) = w(class, t) + n*grad
        end
        w = nextw;
end

for ts = 1:500
        xts = Xtest(:,ts); % 21*1
        plugin = zeros(10,1);
        for j = 1:10
                plugin(j) = exp(xts'*w(:,j))/ sum(exp(xts'*w)); % (1*21)*(21*1) / sum((1*21)*(21*10))
        end
        maxidx = find(plugin==max(plugin));
        conf(label_test(ts)+1,maxidx) = conf(label_test(ts)+1,maxidx) + 1;
end
```

Log-Likelihood:



Confusion Matrix:

	1	2	3	4	5	6	7	8	9	10
1	43	0	1	0	0	5	1	0	0	0
2	0	40	0	0	0	2	0	0	8	0
3	1	0	36	3	0	0	3	0	7	0
4	1	0	1	38	0	3	0	0	7	0
5	0	0	2	0	40	1	0	0	2	5
6	0	1	0	6	2	38	0	0	1	2
7	0	0	1	0	8	4	35	0	2	0
8	0	0	2	0	1	0	0	42	4	1
9	0	0	0	0	0	3	0	0	46	1
10	0	0	1	0	2	1	0	0	1	45

Accuracy: 0.806

Misclassified example:

True	0	0	0
Predict	5	5	2