EECE-5644
Intro. to Machine Learning
Spring 2016
Homework #: 5
Submission Date: 4 / 1 / 2016

Name (Last, First): Jortberg, Elise

Homework Submission Rules:

1. All submitted work should be legible. Do not write in small or in overly cursive characters.
2. Your submission of problem solutions must be in the given order, i.e., P1, P2, P3, etc. Do not submit in a random order.
3. Use this cover page as the first page for each homework submission. Homework submitted without a cover page will result in a 10% reduction in the overall score for that homework.
4. All plots must have axis labeled, titles, and legends if applicable.
5. Code specified in the problems must be submitted via the Blackboard site alongside a pdf of your homework solutions. The code must be contained in a zip file with the following file name format: hw_##_lastname.zip. Do not print the code
6. Be concise when writing your solutions and use both sides of the page.

(PLEASE DO NOT WRITE BELOW THIS LINE)

Problem-#	1	2	3	4	5	6	7	8	9	10	TOTAL
Points											

* Please not Matlab indexes from one so Confusion matrix reads 1 → 10 which corresponds to 0 → 9

Elise Jortberg
EECE 5644
HW 5

5.2) SVM for handwritten digit classification.

a) 1. One vs all approach: train one class against all others. Compare the output scores from all models and pick one with highest score.
Advantages: good for cases where classification is exclusive, computationally only an $O(N)$. Performs better than one v one for multiple classes
Disadvantages: requires probabilities all normalized to each other, else risk bad results. With fewer classes, other methods can be just as if not more effective

2. One v One: Each model assigns data to a class. At end, whichever class has the most 'votes' wins the final classification.
Advantages: preserves knowledge about probabilities by assigning proportional votes.
Disadvantages: On order of $O(N^2)$, does not perform as well with multiple classes

3. Error Correction: Use Hamming distance between features in columns to minimize training error.
Advantages: penalizes classification probabilities based on error. Utilizes correlation information across feature columns
Disadvantages: Requires good column separation. Gives incorrect results if features are reflections. More Complex

b, c, d, e) MATLAB - use SVMnumbers.m which calls train.m, test.m *

e) RBF Kernel: $K(x,z) = e^{\frac{-\|x-z\|^2}{c^2}}$

expansion: $K(x,z) = e^{-x^2} e^{-z^2} \sum_{k=0}^{\infty} \frac{2^k x^k z^k}{k!}$

Expansion shows summation over an infinite k dimensions, implying the RBF kernel has an infinite dimensional feature space.

nus 5.1) Bernoulli EM for handwritten digit clustering.

a) Usefulness of unsupervised learning for this application? Other applications?

Having someone manually go through a massive dataset (i.e. handwritten digits) is exhausting and time consuming. For this application, the computer learns the clusters which then can be iterated and improved upon. Other examples include: AI, image classification, pattern recognition in "big data" sets pooled from internet.

b) Prove $\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} E[z_{nk}] x_n$ knowing

$$E[z_{nk}] = \frac{\pi_k p(x_n \mid \mu_k)}{\sum_{l=1}^{K} \pi_l p(x_n \mid \mu_l)}$$

where $p(x \mid \mu_k) = \prod_{d=1}^{D} \mu_{kd}^{x_d} (1-\mu_{kd})^{1-x_d}$

$$\ln(E z_{nk}] = \sum_{n=1}^{N} \sum_{k=1}^{K} E[z_{nk}] \left(\ln \pi_k + \sum_{d=1}^{D} x_{nd} \ln(\mu_{kd}) + (1-x_{nd}) \ln(1-\mu_{kd}) \right.$$

in $p(x \mid \mu_k)$ sum over all d... $x_{nd} \to x_n$

$$\frac{\partial}{\partial \mu_k} E[z_{nk}] = 0$$

$\Rightarrow$

$$\frac{\partial}{\partial \mu_k} \sum_{n=1}^{N} E[z_{nk}] \left(\frac{x_n}{\mu_k} - \frac{1-x_n}{1-\mu_k} \right) = 0$$

$$= \sum_{n=1}^{N} E[z_{nk}] \frac{x_n - \cancel{x_n \mu_k} - \mu_k + \cancel{\mu_k x_n}}{\mu_k(1-\mu_k)}$$

$$= \sum_{n=1}^{N} E[z_{nk}] (x_n - \underbrace{\mu_k}_{\text{not dependent on } n}) = 0$$

$$-N_k \mu_k + \sum_{n=1}^{N} E[z_{nk}] x_n = 0$$

$$\sum_{n=1}^{N} E[z_{nk}] x_n = N_k \mu_k$$

$$\therefore \quad \mu_k = \frac{1}{N_k} \sum_{n=1}^{N} E[z_{nk}] x_n$$

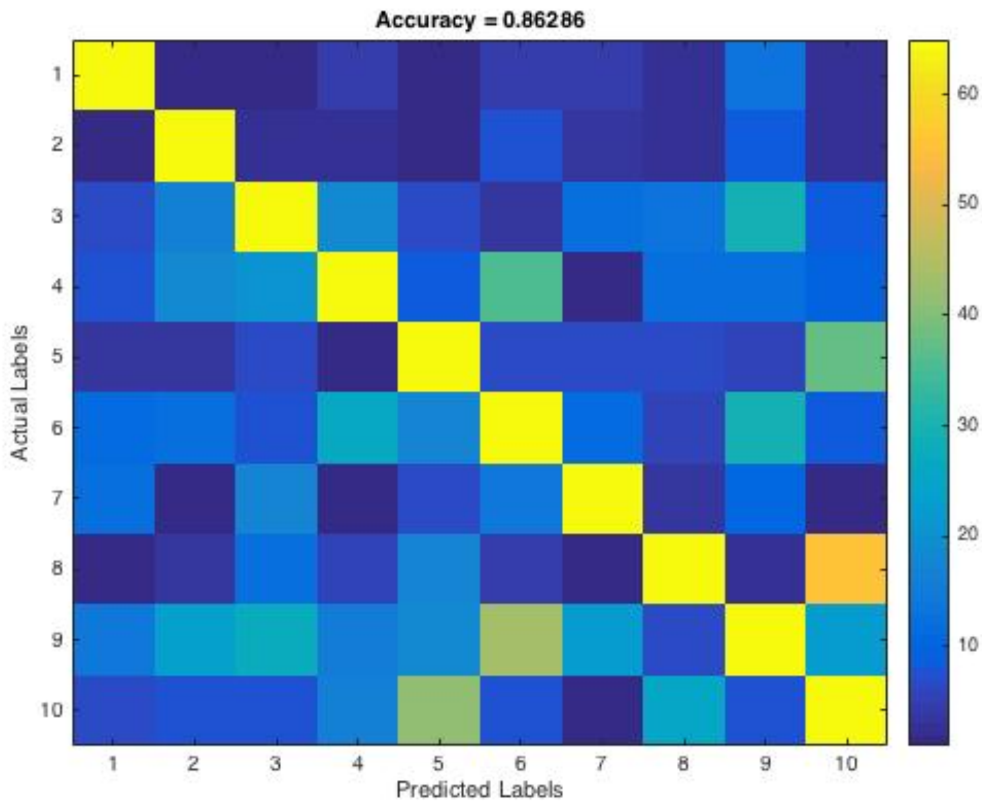
c) MATLAB function - use hw5.m which calls EMClustering-Bernoulli.m

d) digits = 0, 1, 4. K = 3
The cluster means vary in accuracy compared to the true means for each digit. EM performed best on 1, decently on 0 and the worst on 4.

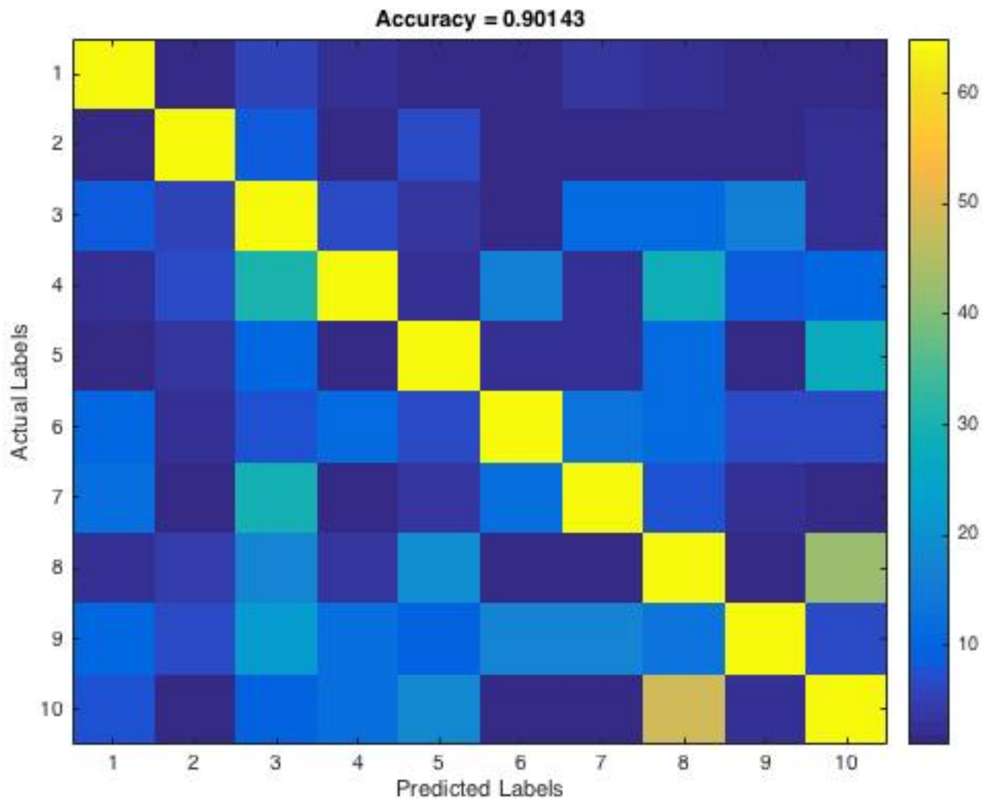
e) K = 5, 10, 20 for all digits (0-9)
NOTE - titles added when digits where discernable from EM results.
As K increases the quality of the means appears to get better, sharper. K=10 did not identify all of the digits, repeating some. K=20 did find all the digits (some repeated). I think the problem is that some digits are classified as two different clusters because of variation in their morphology, i.e. someone's 3 could look like someone else's 8. I don't think GMM would solve the problem and may make the means fuzzier since the distribution is continuous instead of binary. GMM better for segmentation in fuzzy images

Elise Jortberg
HW 5
EECE 5644

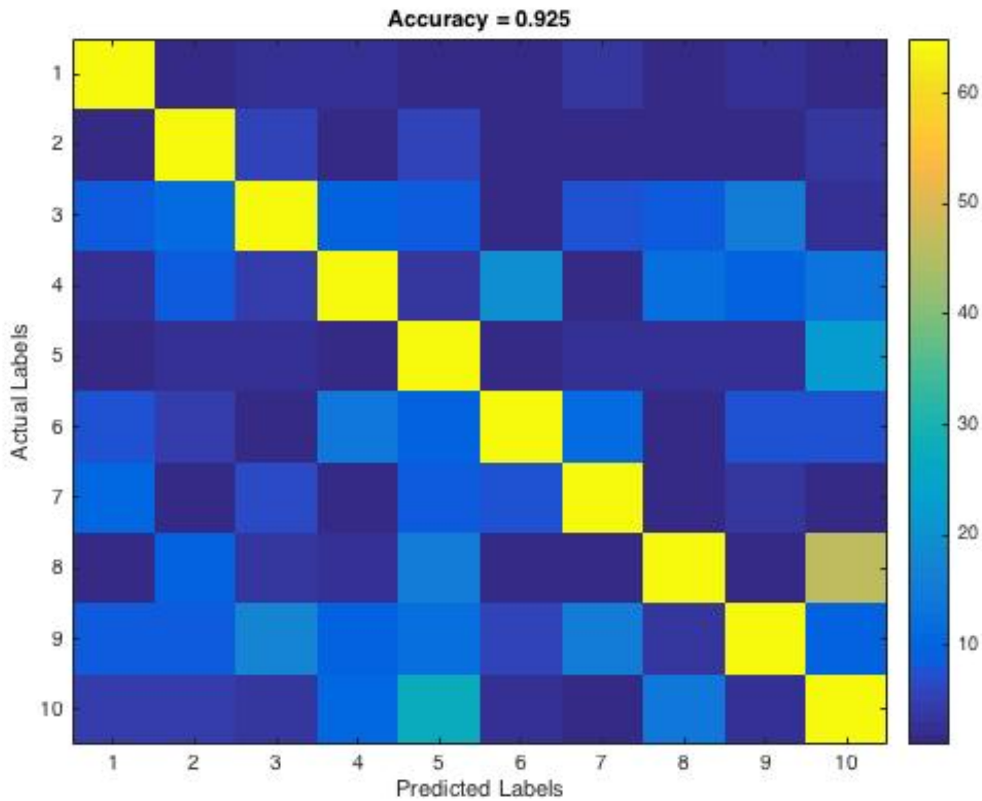
Problem 5.2
b)



c)

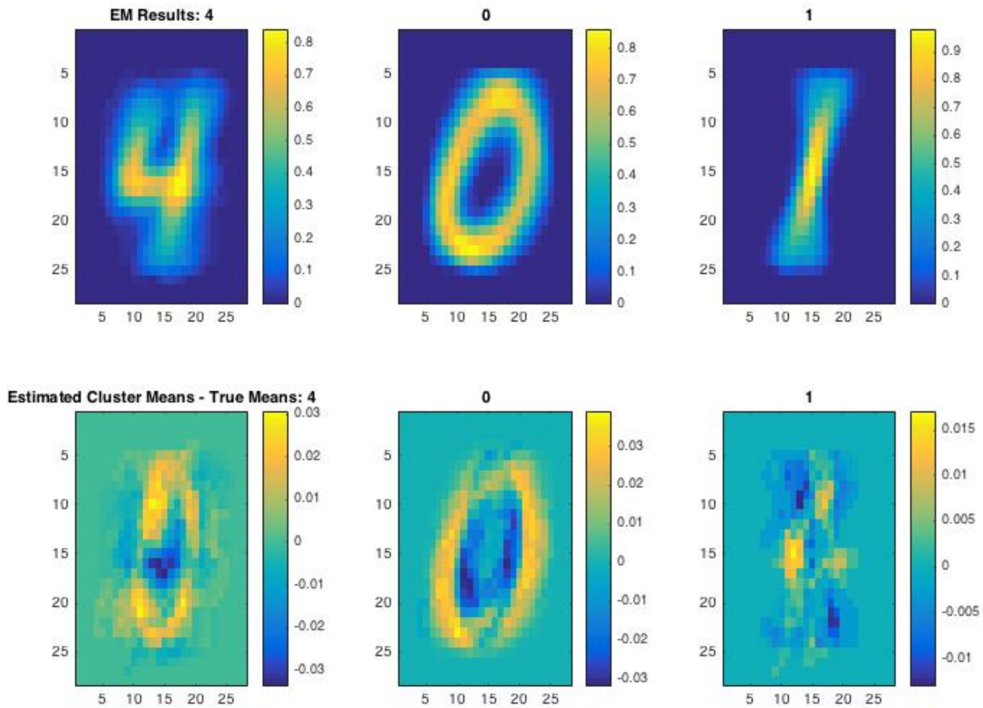

Accuracy = 0.90143
Actual Labels
Predicted Labels
1
2
3
4
5
6
7
8
9
10
10
20
30
40
50
60

d)

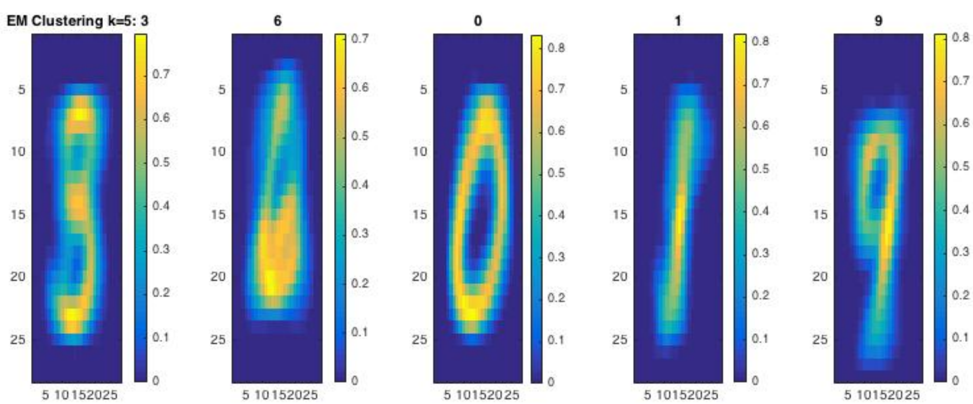


Accuracy = 0.925
Actual Labels
Predicted Labels
1
2
3
4
5
6
7
8
9
10
60
50
40
30
20
10

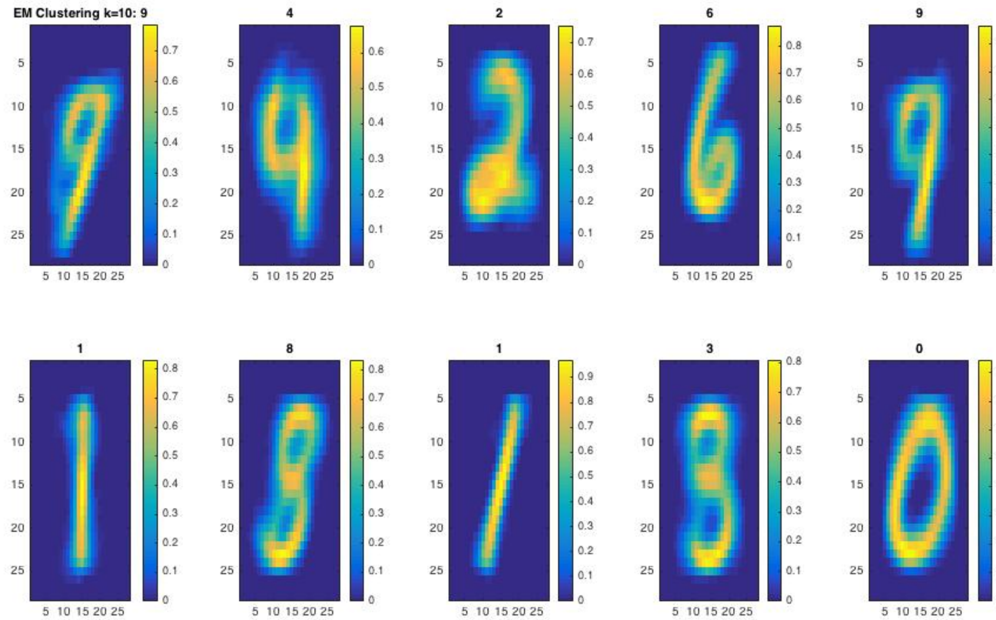
Problem 5.1)
d)


EM Results: 4
0
1
Estimated Cluster Means - True Means: 4
0
1

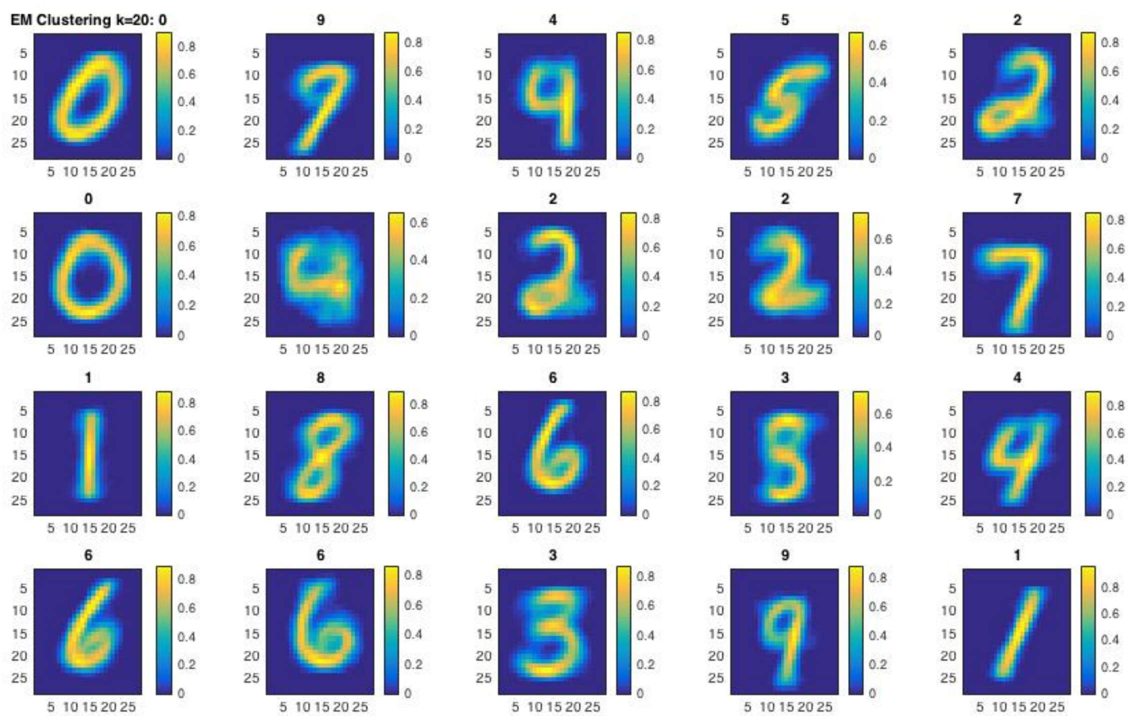
e) K=5


EM Clustering k=5: 3
6
0
1
9

K=10


EM Clustering k=10: 9
4
2
6
9
1
8
1
3
0

K=20


EM Clustering k=20: 0
9
4
5
2
0
2
2
7
1
8
6
3
4
6
6
3
9
1