Applied Machine Learning Home Work 2

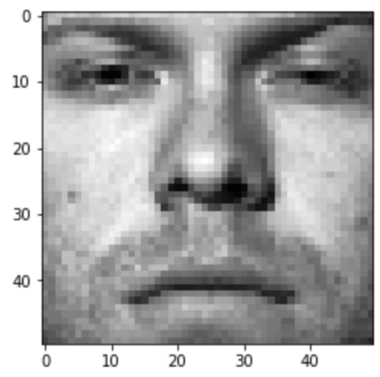
Harrison Zhao, Haochen Jia

September 27th

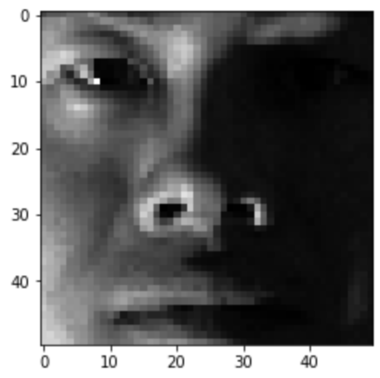
1 Face Recognition

1.1 (b) Load the training set into a matrix X: there are 540 training images in total, each has 50 50 pixels that need to be concatenated into a 2500-dimensional vector. So the size of X should be 540 2500, where each row is a flattened face image. Pick a face image from X and display that image in grayscale. Do the same thing for the test set. The size of matrix Xtest for the test set should be 100 2500.

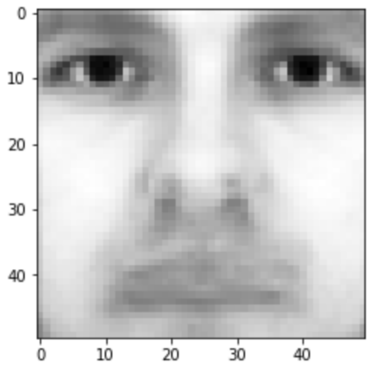
The 10th face from train data



The 10th face from test data

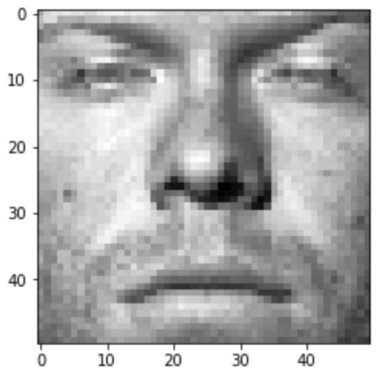


1.2 (c) Average Face. Compute the average face from the whole training set by summing up every column in X then dividing by the number of faces. Display the average face as a grayscale image.

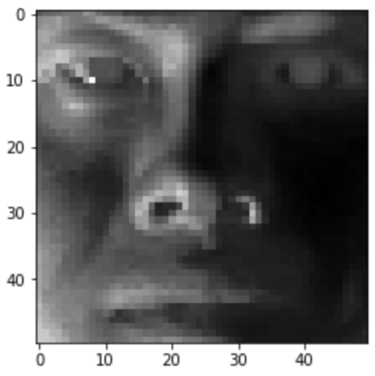


1.3 (d) Mean Subtraction. Subtract average face from every column in X. That is, xi := xi , where xi is the i-th column of X. Pick a face image after mean subtraction from the new X and display that image in grayscale. Do the same thing for the test set Xtest using the pre-computed average face in (c).

The 10th face after subtracting average face in train data.



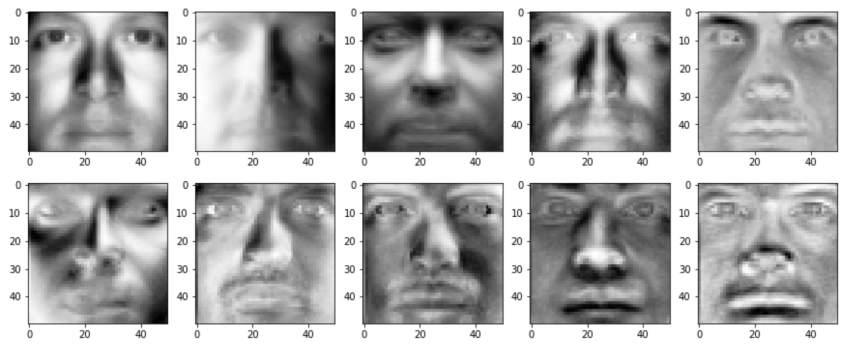
The 10th face after subtracting average face in test data.



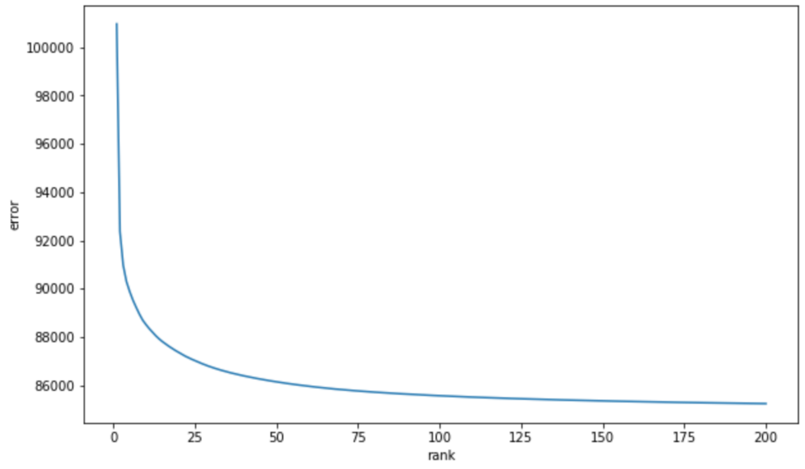
1.4 (e) Eigenface. Perform Singular Value Decomposition (SVD) on training set X (X = UVT) to get matrix VT , where each row of VT has the same dimension as the face image. We refer to vi, the i -th row of VT , as i -th eigenface. Display the first 10 eigenfaces as 10 images in grayscale.

```
U.shape, S.shape, Vt.shape
```

```
((540, 540), (540, 540), (540, 2500))
```



1.5 (f) Low-rank Approximation. Since is a diagonal matrix with non-negative real numbers on the diagonal in non-ascending order, we can use the first r elements in together with first r columns in U and first r rows in VT to approximate X. That is, we can approximate X by Xˆr = U[:,: r] [: r,: r] VT [: r,:]. The matrix Xˆr is called rank-r approximation of X. Plot the rank-r approximation errorXXˆrF 2 a s afunctio no fr whenr =1,2,...,200.



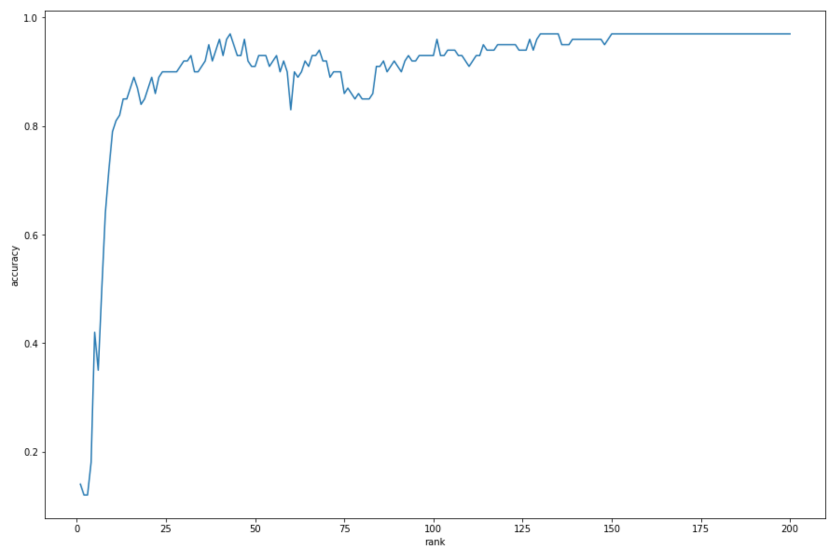
1.6 (g) Eigenface Feature. The top r eigenfaces VT [: r,:] = v1,v2,...,vr T span an r-dimensional linear subspace of the original image space called face space, whose origin is the average face , and whose axes are the eigenfaces v1,v2,...,vr. Therefore, using the top r eigenfaces v1,v2,...,vr,we can represent a 2500-dimensional face image z as an r-dimensional feature vector f: f = VT [: r,:] z = [v1,v2,...,vr]T z. Write a function to generate r-dimensional feature matrix F and Ftest for training images X and test images Xtest, respectively (to get F, multiply X to the transpose of first r rows of VT , F should have same number of rows as X and r columns; similarly for Xtest).

Check it in the code. With r=20

1.7 (h) Face Recognition. Extract training and test features for r = 10. Train a Logistic Regression model using F and test on Ftest. Report the classification accuracy on the test set. Plot the classification accuracy on the test set as a function of r when r = 1,2,...,200. Use "one-vs- rest" logistic regression, where a classifier is trained for each possible output label. Each classifier is trained on faces with that label as positive data and all faces with other labels as negative data. sklearn calls this "ovr" mode.

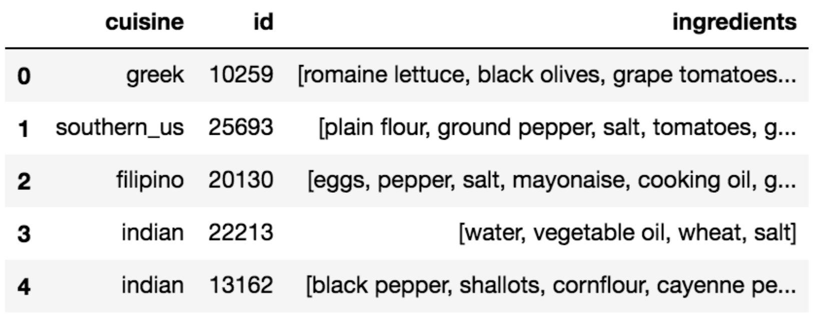
Check it in the code. When r = 10, the accuracy is 0.79000000000000004.

For r = 1:200, the accuracy shows as follow.



2 Whats' Cooking

2.1 (a) Join the What's Cooking competition on Kaggle. Download the training and test data (in .json). The competition page describes how these files are formatted.

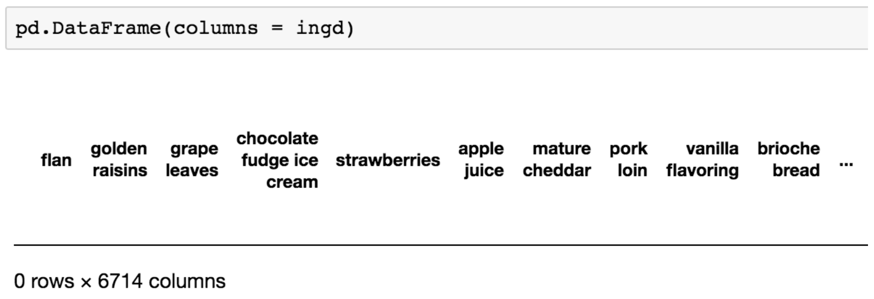


	cuisine	id	ingredients
0	greek	10259	[romaine lettuce, black olives, grape tomatoes...
1	southern_us	25693	[plain flour, ground pepper, salt, tomatoes, g...
2	filipino	20130	[eggs, pepper, salt, mayonaise, cooking oil, g...
3	indian	22213	[water, vegetable oil, wheat, salt]
4	indian	13162	[black pepper, shallots, cornflour, cayenne pe...

2.2 (b) Tell us about the data. How many samples (dishes) are there in the training set? How many categories (types of cuisine)? Use a list to keep all the unique ingredients appearing in the training set. How many unique ingredients are there?

There are 39774 samples in the the training set.
There are 20 types of cuisines in the the training set.
There are 6714 unique ingredients in the the training set.

2.3 (c) Represent each dish by a binary ingredient feature vector. Suppose there are d different ingredients in total from the training set, represent each dish by a 1d binary ingredient vector x, where xi = 1 if the dish contains ingredient i and xi = 0 otherwise. For example, suppose all the ingredients we have in the training set are beef, chicken, egg, lettuce, tomato, rice and the dish is made by ingredients chicken, lettuce, tomato, rice , then the dish could be represented by a 6 1 binary vector [0, 1, 0, 1, 1, 1] as its feature or attribute. Use n d feature matrix to represent all the dishes in training set and test set, where n is the number of dishes.



```
pd.DataFrame(columns = ingd)
```

flan	golden raisins	grape leaves	chocolate fudge ice cream	strawberries	apple juice	mature cheddar	pork loin	vanilla flavoring	brioche bread	...

0 rows × 6714 columns

The matrix shape is :(39774, 6714)

2.4 (d) Using Naïve Bayes Classifier to perform 3 fold cross-validation on the training set and report your average classification accuracy. Try both Gaussian distribution prior assumption and Bernoulli distribution prior assumption.

Gaussian:
[0.37901644290239855, 0.38293860310755767, 0.37765877206215115]
Bernoulli:
[0.68419067732689698, 0.67951425554382261, 0.68690601900739179]

2.5 (e) For Gaussian prior and Bernoulli prior, which performs better in terms of cross-validation accuracy? Why? Please give specific arguments.

Answer: Bernoulli Naive Bayes Classifier perform better. Reasons: Gaussian NB assumes the data is normally distributed but the data we have is binary so the Bernoulli will have better fit. Gaussian NB assumes the data is continuous with order but our data is discontinuous and doesn't have order. So, Bernoulli NB should perform better.

2.6 (f) Using Logistic Regression Model to perform 3 fold cross-validation on the training set and report your average classification accuracy.

[0.77590888520138779, 0.77213757731181176, 0.7786242268818826]

2.7 (g) Train your best-performed classifier with all of the training data, and generate test labels on test set. Submit your results to Kaggle and report the accuracy.

Check the out put in the code. the accuracy is 0.783

3 Written Exercises

3.1 Exercise 4.1

Show how to solve the generalized eigenvalue problem $max[a^T Ba]$ subject to $[a^T Wa] = 1$ by transforming to a standard eigenvalue problem.

- Proof:Let us define the Lagrangian L with multiplier λ as following:

  $$L(a, \lambda) = a^T Ba + \lambda(a^T Wa - 1)$$

  Which has the same value as $[a^T Ba]$

- To achieve the maximum, let us fix its derivative to zero:

  $$\frac{\partial L}{\partial a} = \frac{\partial a^T Ba}{\partial a} + \frac{\partial a^T Wa}{\partial a} = 0$$

- Because $\forall X$ we have $\frac{\partial a^T Xa}{\partial a} = a^T * (X + X^T) or (X + X^T) * a$ The right hand side of function can be changed to

  $$(B + B^T + \lambda(W + W^T) * a$$

  Because B,W are Hermitian matrices, $B^T = B, W^T = W$

- So, it is $2[Ba + \lambda(Wa)] = 0$
- So, $Ba = -\lambda Wa$
- So, $W^{-1}Ba = -\lambda a$

3.2 Exercise 4.2

Suppose we have features$x \in R_p$, a two-class response, with class sizes N1,N2, and the target coded as N/N1,N/N2.

- Part a: According to the logic -ratio upon class density when G= 1 or 2, its log ration should be following:

$$log\frac{f(G=2)}{f(G=1)} - 0.5(\mu_1 + \mu_2)^T \sum^{-1}(\mu_2 - \mu_1) + x^T \sum^{-1}(\mu_1 + \mu_2)$$

- Lda rule will go to class 1 if this function smaller than 0 and go to class 2 otherwise.
- in this equation, Lda will go to 0 if the function greater than 0: Then, we try to estimate the Gaussian distribution with $\hat{\mu_i and} \hat{\sum}$

$$log\frac{f(G=2)}{f(G=1)} - 0.5(\mu_1 + \mu_2)^T \hat{\sum}^{-1}(\mu_2 - \mu_1) + x^T \hat{\sum}^{-1}(\mu_1 + \mu_2)$$

so,

$$x^T \hat{\sum}^{-1}(\hat{\mu_1} + \hat{\mu_2}) > -log\frac{f(G=2)}{f(G=1)} + 0.5(\hat{\mu_1} + \hat{\mu_2})^T \hat{\sum}^{-1}(\hat{\mu_2} - \hat{\mu_1})$$

because$f(G=1) = (N/N1)$ and $f(G=2) = (N/N2)$ and the right hand side is

$$0.5\hat{\mu_2}^T \hat{\sum}^{-1} \hat{\mu_2} - 0.5\hat{\mu_1}^T \hat{\sum}^{-1} \hat{\mu_1} + log(\frac{N_1}{N}) - log(\frac{N_2}{N})$$

- Part b: Commonly, we assume

$$\hat{Y} = X^T \times \hat{\beta}$$

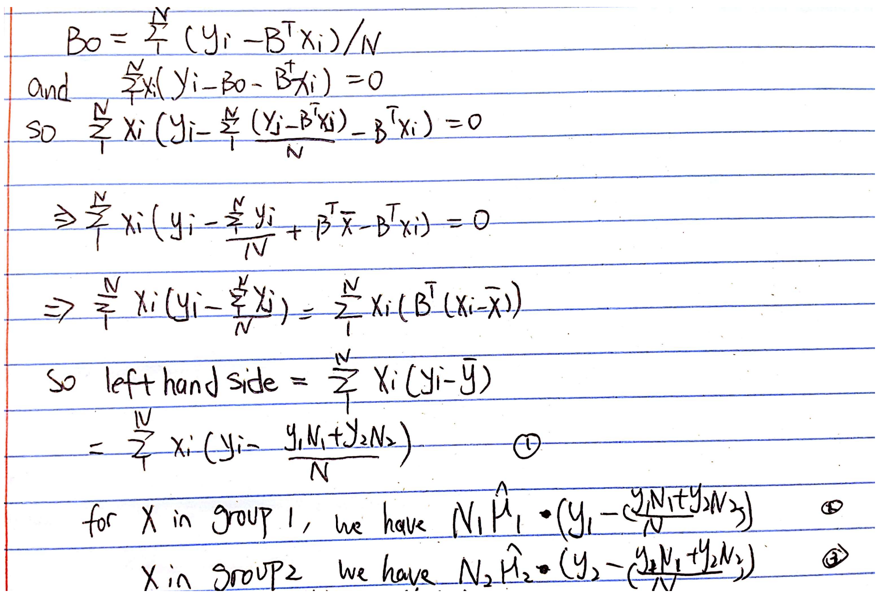
In this case,

$$X^T = (x_1, x_2 \ldots x_N) and \hat{\beta} = (\beta_0, \beta_1 \ldots \beta_N)^T = (\beta_0, \beta)$$

- Because we consider the minimize the least square criterion, thus, the derivative of $\beta_0 or \beta$ in the function (4.55) RSS should be strictly equal to 0:

$$\frac{\partial RSS}{\partial \beta_0} = -2\sum(y_i - \beta_0 - \beta^T x_i) = 0$$

$$\frac{\partial RSS}{\partial \beta} = -2 \sum x_i(y_i - \beta_0 - \beta^T x_i) = 0$$



$B_0 = \sum_1^N (y_i - B^T x_i)/N$

and $\sum_1^N x_i(y_i - B_0 - B^T x_i) = 0$

so $\sum_1^N x_i \left(y_i - \sum_1^N \frac{(y_j - B^T x_j)}{N} - B^T x_i\right) = 0$

$\Rightarrow \sum_1^N x_i \left(y_i - \frac{\sum_1^N y_j}{N} + B^T \bar{x} - B^T x_i\right) = 0$

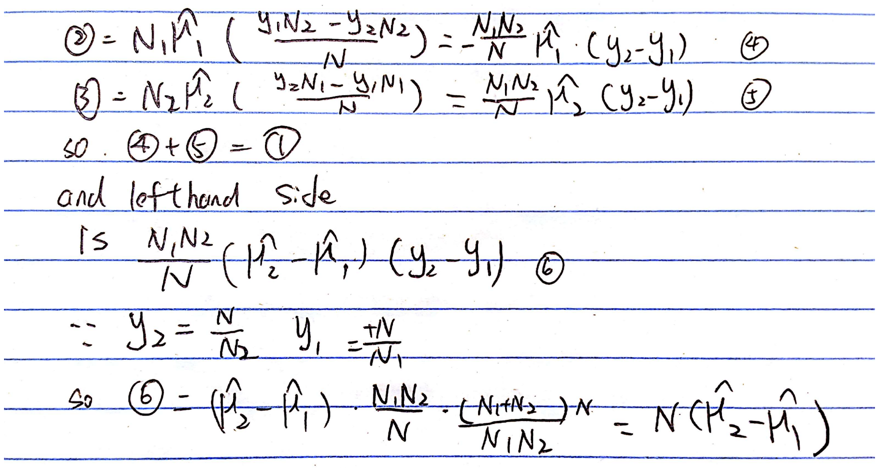
$\Rightarrow \sum_1^N x_i \left(y_i - \frac{\sum_1^N y_j}{N}\right) = \sum_1^N x_i (B^T (x_i - \bar{x}))$

So left hand side $= \sum_1^N x_i (y_i - \bar{y})$

$= \sum_1^N x_i \left(y_i - \frac{y_1 N_1 + y_2 N_2}{N}\right)$ $\textcircled{1}$

for X in group 1, we have $N_1 \hat{\mu}_1 \cdot \left(y_1 - \left(\frac{y_1 N_1 + y_2 N_2}{N}\right)\right)$ $\textcircled{2}$

X in group2 we have $N_2 \hat{\mu}_2 \cdot \left(y_2 - \left(\frac{y_1 N_1 + y_2 N_2}{N}\right)\right)$ $\textcircled{3}$



$\textcircled{2} = N_1 \hat{\mu}_1 \left(\frac{y_1 N_2 - y_2 N_2}{N}\right) = -\frac{N_1 N_2}{N} \hat{\mu}_1 \cdot (y_2 - y_1)$ $\textcircled{4}$

$\textcircled{3} = N_2 \hat{\mu}_2 \left(\frac{y_2 N_1 - y_1 N_1}{N}\right) = \frac{N_1 N_2}{N} \hat{\mu}_2 (y_2 - y_1)$ $\textcircled{5}$

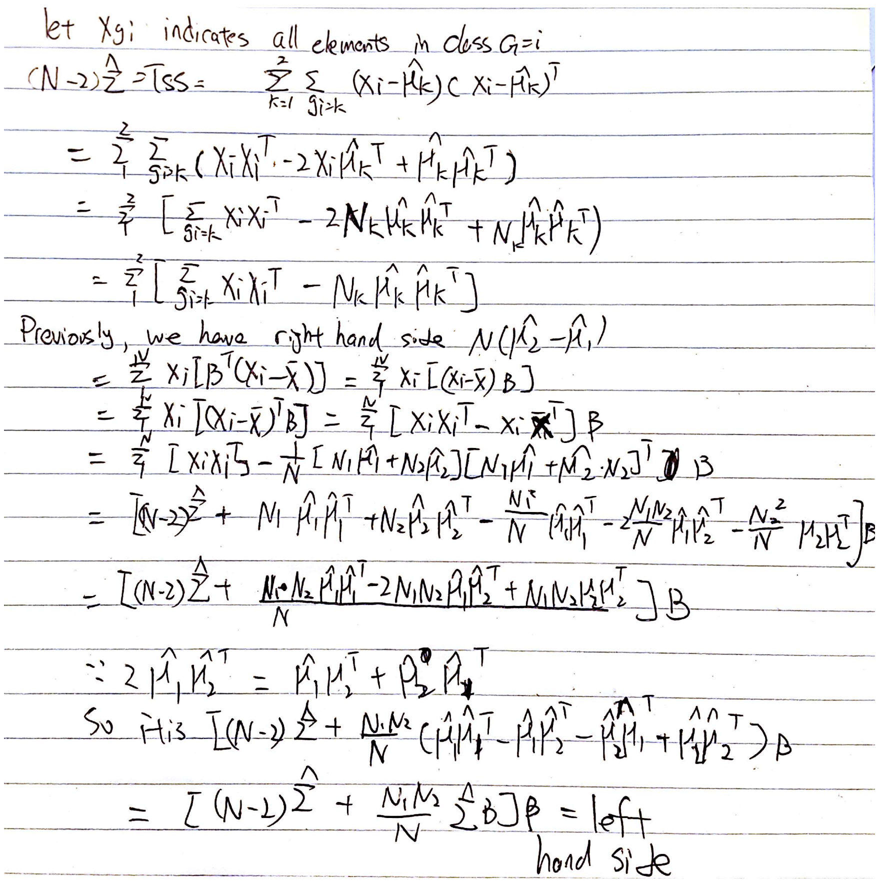
so. $\textcircled{4} + \textcircled{5} = \textcircled{1}$

and lefthand side

is $\frac{N_1 N_2}{N} (\hat{\mu}_2 - \hat{\mu}_1)(y_2 - y_1)$ $\textcircled{6}$

$\because y_2 = \frac{N}{N_2} \quad y_1 = \frac{-N}{N_1}$

so $\textcircled{6} = (\hat{\mu}_2 - \hat{\mu}_1) \cdot \frac{N_1 N_2}{N} \cdot \frac{(N_1 + N_2) N}{N_1 N_2} = N(\hat{\mu}_2 - \hat{\mu}_1)$



let X_{gi} indicates all elements in class $G=i$

$$(N-2)\hat{\Sigma} = TSS = \sum_{k=1}^{2}\sum_{g_i=k}(x_i-\hat{\mu}_k)(x_i-\hat{\mu}_k)^T$$

$$= \sum_{1}^{2}\sum_{g_i=k}(x_i x_i^T - 2x_i\hat{\mu}_k^T + \hat{\mu}_k\hat{\mu}_k^T)$$

$$= \sum_{1}^{2}\left[\sum_{g_i=k} x_i x_i^T - 2N_k\hat{\mu}_k\hat{\mu}_k^T + N_k\hat{\mu}_k\hat{\mu}_k^T\right)$$

$$= \sum_{1}^{2}\left[\sum_{g_i=k} x_i x_i^T - N_k\hat{\mu}_k\hat{\mu}_k^T\right]$$

Previously, we have right hand side $N(\hat{\mu}_2-\hat{\mu}_1)$

$$= \sum^{N} x_i[\beta^T(x_i-\bar{x})] = \sum_{1}^{N} x_i[(x_i-\bar{x})\beta]$$

$$= \sum_{1}^{N} x_i[(x_i-\bar{x})^T\beta] = \sum_{1}^{N}[x_i x_i^T - x_i\bar{x}^T]\beta$$

$$= \sum_{1}^{N}[x_i x_i^T] - \frac{1}{N}[N_1\hat{\mu}_1 + N_2\hat{\mu}_2][N_1\hat{\mu}_1 + \hat{\mu}_2 \cdot N_2]^T]\beta$$

$$= [(N-2)\hat{\Sigma} + N_1\hat{\mu}_1\hat{\mu}_1^T + N_2\hat{\mu}_2\hat{\mu}_2^T - \frac{N_1^2}{N}\hat{\mu}_1\hat{\mu}_1^T - 2\frac{N_1N_2}{N}\hat{\mu}_1\hat{\mu}_2^T - \frac{N_2^2}{N}\hat{\mu}_2\hat{\mu}_2^T]\beta$$

$$= [(N-2)\hat{\Sigma} + \frac{N_1 \cdot N_2\hat{\mu}_1\hat{\mu}_1^T - 2N_1N_2\hat{\mu}_1\hat{\mu}_2^T + N_1N_2\hat{\mu}_2\hat{\mu}_2^T}{N}]\beta$$

$$\because 2\hat{\mu}_1\hat{\mu}_2^T = \hat{\mu}_1\hat{\mu}_2^T + \hat{\mu}_2\hat{\mu}_1^T$$

So $\hat{\mu}$ is $[(N-2)\hat{\Sigma} + \frac{N_1N_2}{N}(\hat{\mu}_1\hat{\mu}_1^T - \hat{\mu}_1\hat{\mu}_2^T - \hat{\mu}_2\hat{\mu}_1^T + \hat{\mu}_2\hat{\mu}_2^T)]\beta$

$$= [(N-2)\hat{\Sigma} + \frac{N_1N_2}{N}\hat{\Sigma}_B]\beta = \text{left hand side}$$

- Part c:

$$\hat{\sum_B}\beta = (\hat{\mu_2}-\hat{\mu_1})(\hat{\mu_2}-\hat{\mu_1})^T\beta$$

Consider $\hat{\beta'} = (\hat{\mu_2}-\hat{\mu_1})^T\beta$, it is clear that

$$\hat{\sum_B}\beta = (\hat{\mu_2}-\hat{\mu_1}) * \hat{\beta'}$$

So, $\hat{\beta'}$works as scalar and $\hat{\sum}_B\beta$ is in the direction of $\hat{\mu_2}-\hat{\mu_1}$ Further more

$$\hat{\beta} = (\frac{1}{N-2})\hat{\sum}^{-1}[N(\hat{\mu_2}-\hat{\mu_1}) - \frac{N_1N_2}{N}\hat{\sum_B}\beta]$$

$$= (\frac{1}{N-2})[N - \frac{N_1N_2}{N}\hat{\beta'}]\hat{\sum}^{-1}(\hat{\mu_2}-\hat{\mu_1})$$

So, it goes to $\hat{\sum}^{-1}(\hat{\mu_2}-\hat{\mu_1})$

- Part d: From part b, we know that $\frac{N_1N_2}{N}(y_2-y_1)(\hat{\mu_2}-\hat{\mu_1}) = N(\hat{\mu_2}-\hat{\mu_1})$ So, we can just replace this function to the function in part c:

  $$\hat{\beta} = (\frac{1}{N-2})[\frac{N_1N_2}{N}(y_2-y_1) - \frac{N_1N_2}{N}\hat{\beta}']\hat{\sum}^{-1}(\hat{\mu_2}-\hat{\mu_1})$$

  It goes to $\hat{\sum}^{-1}(\hat{\mu_2}-\hat{\mu_1})$

- Part e: Just consider when $y_i = 0$:we have

  $$-2\sum(y_i - \beta_0 - \beta^T x_i) = 0$$

  , so we can estimate $\hat{\beta}_0$ as $\frac{1}{N}\sum(\beta^T x_i) = \mu^T\hat{\beta}$ Then, if $N_1 = N_2$, it is clear that$\mu = 0.5(\mu_1+\mu_2)$ So, $\hat{f} = (x-\hat{\mu})^T\hat{\beta}$ which is a function of $\hat{\sum}^{-1}(\hat{\mu_2}-\hat{\mu_1}) = x^T\hat{\sum}^{-1}(\hat{\mu_2}-\hat{\mu_1}) - 0.5(\mu_1+\mu_2)^T\hat{\sum}^{-1}(\hat{\mu_2}-\hat{\mu_1})$ So, it follows the Lda rule. Otherwise, it is clear that if $N_1$is not equal toN_2, this will not hold.

3.3 Consider matrix M. It has rank 2, as you can see by observing that there times the first column minus the other two columns is 0.

(a) Compute the matrices M^TM and MM^T. (b) Find the eigenvalues for your matrices of part(a). (c) Find the eigenvectors for the matrices of part(a). (d) Find the SVD for the original matrix M from parts (b) and (c). Note that there are only two nonzero eigenvalues, so your matrix $\sum$ should have only two singular values, while U and V have only two columns. (e) Set your smaller singular value to 0 and compute the one-dimensional approximation to the matrix M.

- a):M^TM:

  $$\begin{bmatrix} 39 & 57 & 60 \\ 57 & 118 & 53 \\ 60 & 53 & 127 \end{bmatrix}$$

  MM^T:

  $$\begin{bmatrix} 10 & 9 & 26 & 3 & 26 \\ 9 & 62 & 8 & -5 & 85 \\ 26 & 8 & 72 & 10 & 50 \\ 3 & -5 & 10 & 2 & -1 \\ 26 & 85 & 50 & -1 & 138 \end{bmatrix}$$

- b,c) eigenvalue of M^TM

  [0.90453403 0.01460404 0.42615127] [-0.30151134 0.72859799 0.61500884] [-0.30151134 -0.68478587 0.66344497] eigenvector of M^TM
  [0 69.3295108 214.670489]

eigenvector of MM^T:
[0.94963468 0.03803537 0.09764055 0.24497323 -0.16492942]
[-0.04825172 0.28679929 0.6981977 -0.45330644 -0.47164732]
[-0.3024112 -0.07162025 0.31974279 0.82943965 -0.33647055]
[-0.05303102 0.94252712 -0.28285243 0.16974659 -0.00330585]
[-0.04001143 -0.15103855 -0.56634384 -0.13310656 -0.79820031]

eigenvalue of MM^T:
[0 0 0 69.329510 214.670489]

- d)
  SVD:
  D:
  [0.24497323 -0.16492942]
  [-0.45330644 -0.47164732]
  [0.82943965 -0.33647055]
  [0.16974659 -0.00330585]
  [-0.13310656 -0.79820031]
  σ:
  [14.6516378 0]
  [0 8.32643446]
  V:
  [0.01460404 0.42615127]
  [0.72859799 0.61500884]
  [-0.68478587 0.66344497]

- e) the new matrix is following when we eliminate the small singular
  [1.02978864, 1.48616035, 1.60320558]
  [2.94487812 4.24996055 4.58467382]
  [2.10085952 3.031898 3.27068057]
  [0.02064112 0.02978864 0.0321347]
  [4.9838143 7.19249261 7.75895028]