ECE-6143 Intro to ML
Homework1:
Haotian Yi
Net ID: hy1651

Q1:

Suppose μ is what we wish.

Prove:

$$let\ original\ aim\ function = h(\mu) = \sum_{j=1}^{n} \sum_{i=1}^{d} \left(x_{ji} - \mu_i\right)^2$$

$$= \sum_{i=1}^{d} h(\mu_i)$$

for each attribute x_{ji} ,it has distribution of $\{x_{1i}, x_{2i}, \ldots, x_{ni}\}$

because $E[(x_i - \mu_i)^2] = \frac{\sum_{j=1}^{n} (x_{ji} - \mu_i)^2}{n}$

for each μ_i:

$$h(\mu_i) = \sum_{j=1}^{n} \left(x_{ji} - \mu_i\right)^2 = n\,E[\left(x_{ji} - \mu_i\right)^2]$$

$$= n\,E[{x_i}^2 + {\mu_i}^2 - 2\,\mu_i\,x_i]$$

$$= n\,\{E[{x_i}^2] + {\mu_i}^2 - 2\,\mu_i\,E[x_i]\}$$

to get minimum $h(\mu_i)$:

$$\frac{\partial h(\mu_i)}{\partial \mu_i} = n\,\{E[2\,\mu_i - 2E[x_i]\} = 0$$

so, $\mu_i = E[x_i]$

thus, we can conclude that $\mu = E[x]$

Q2(a):

1)For $\|x\|_2$ and $\|x\|_1$:

because : $\|x\|_2 < \|x\|_1 \;\Leftrightarrow\; \sum_{i=1}^{d} x_i{}^2 < \left(\sum_{i=1}^{d} |x_i|\right)^2$

(do square of both side)

and: $\left(\sum_{i=1}^{d} |x_i|\right)^2 = \sum_{i=1}^{d} |x_i|^2 + \sum c_j\, \alpha$,

$\sum c_j\, \alpha$ represents cross terms of $\left(\sum_{i=1}^{d} |x_i|\right)^2$

and $\sum c_j\, \alpha > 0$ (when x_i are not all equal to 0)

so, $\sum_{i=1}^{d} x_i{}^2 < \left(\sum_{i=1}^{d} |x_i|\right)^2$

so, $\sum_{i=1}^{d} x_i{}^2 \le \left(\sum_{i=1}^{d} |x_i|\right)^2$

(equal when d = 1 or x_i are all equal to 0)

Thus, $\|x\|_2 \le \|x\|_1$

2)For $\|x\|_1$ and $\sqrt{d}\|x\|_2$:

Inequation above is equal when d = 1 or x_i are all 0

when d $\geq$ 2 and x_i are not all 0:

$\left(\sqrt{d}\,\|x\|_2\right)^2 = \sum_{i=1}^{d} x_i{}^2 + (d-1) \sum_{i=1}^{d} x_i{}^2 \ge \sum_{i=1}^{d} x_i{}^2 + 3 \sum_{i=1}^{d} x_i{}^2$

$(\|x\|_1)^2 = \sum_{i=1}^{d} |x_i|^2 + \sum c_j\, \alpha$

$\sum c_j\, \alpha$ represents cross terms of $(\|x\|_1)^2$

for d =2, cross terms of $(|x_1| + |x_2|)^2$ is $2|x_1||x_2|$ and $(2-1)(|x_1|^2$

$+|x_2|^2) > 2|x_1||x_2|$,

for d =3, cross terms of $(|x_1| + |x_2| + |x_3|)^2$ is $2|x_1||x_2| + 2|x_1||x_3|$

$+2|x_2||x_3|$and $(3-1)(|x_1|^2 + |x_2|^2 + |x_3|^2) > 2|x_1||x_2| + 2|x_1||x_3|$

$+2|x_2||x_3|$, according to two instances above, we can infer that:

$(d-1) \sum_{i=1}^{d} x_i{}^2 > \sum c_j\, \alpha$ (when d $\geq$ 2 and x_i are not all 0),

so $\|x\|_1 < \sqrt{d}\|x\|_2$ and then $\|x\|_1 \le \sqrt{d}\|x\|_2$

3) Thus: $\|x\|_2 \le \|x\|_1 \le \sqrt{d}\|x\|_2$

Q2(b):

1) For $\|x\|_\infty$ and $\|x\|_2$:

$$(\|x\|_\infty)^2 = \left(\max_i |x_i|\right)^2$$

$$(\|x\|_2)^2 = \sum_{i=1}^{d} x_i^2 = \sum_{i=1}^{maxi-1} x_i^2 + \left(\max_i |x_i|\right)^2 + \sum_{i=maxi+1}^{d} x_i^2$$

(maxi is the subscript of the $max|x_i|$)

thus $\|x\|_\infty \le \|x\|_2$ (equal when d = 1 or all 0 except $a \max_i |x_i|$)

2) For $\|x\|_2$ and $\sqrt{d}\|x\|_\infty$:

$$\left(\sqrt{d}\|x\|_\infty\right)^2 = d\left(\max_i |x_i|\right)^2$$

$$(\|x\|_2)^2 = \sum_{i=1}^{d} x_i^2 = \sum_{i=1}^{maxi-1} x_i^2 + \left(\max_i |x_i|\right)^2 + \sum_{i=maxi+1}^{d} x_i^2$$

(maxi is the subscript of the $max|x_i|$)

It is obvious: $(d-1)\left(\max_i |x_i|\right)^2 \ge \sum_{i=1}^{maxi-1} x_i^2 + \sum_{i=maxi+1}^{d} x_i^2$

thus $\|x\|_2 \le \sqrt{d}\|x\|_\infty$:

3)Thus: $\|x\|_\infty \le \|x\|_2 \le \sqrt{d}\|x\|_\infty$

Q2(c):

1) For $\|x\|_\infty$ and $\|x\|_1$:

$$\|x\|_1 = \sum_{i=1}^{d} |x_i| = \sum_{i=1}^{maxi-1} |x_i| + \max_i |x_i| + \sum_{i=maxi+1}^{d} |x_i|$$

$$\geq \max_i |x_i| = \|x\|_\infty$$

(maxi is the subscript of the $max|x_i|$)

(equal when d = 1 or all 0 except a $\max_i |x_i|$)

2) For $\|x\|_1$ and $d\|x\|_\infty$:

$$\|x\|_1 = \sum_{i=1}^{d} |x_i| = \sum_{i=1}^{maxi-1} |x_i| + \max_i |x_i| + \sum_{i=maxi+1}^{d} |x_i|$$

$d\|x\|_\infty = (d-1)\max_i |x_i| + \max_i |x_i|$ and it is obviously $\geq \|x\|_1$

(equal when d = 1 or all attributes are equal)

3)Thus: $\|x\|_\infty \leq \|x\|_1 \leq d\|x\|_\infty$

For question 3 and question 4, there are more comments and interpretation in details in .ipynb files.

Q3:

a. Fix d = 3 and generate 10,000 random samples from the standard multi-variate Gaussian distribution defined in Rd.

```
[2] import numpy as np

    mean = np.zeros(3)#d=3
    conv = np.diag(np.ones(3))
    samplenumber = 10000
    sample = np.random.multivariate_normal(mean,conv,samplenumber)
    print("samples:",sample)
```

```
samples: [[-1.99633948  0.19114024  2.05743083]
 [-0.14630086  0.82596652  1.38016821]
 [ 0.76768404  1.12115948  0.5708857 ]
 ...
 [-0.76048075 -0.0235946   0.6739678 ]
 [-0.2612948   0.44252343 -0.04179097]
 [-0.59923705  0.64413611 -0.89987539]]
```

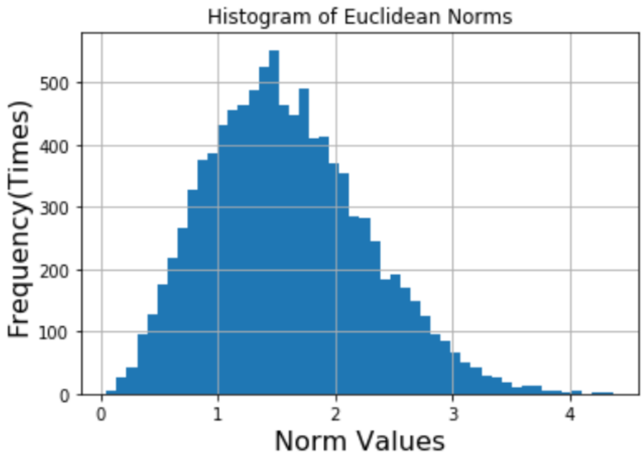
b. Compute and plot the histogram of Euclidean norms of your samples. Also calculate the average and standard deviation of the norms.

```
[ ] #calculate norms and theirs mean and deviation
    norm = np.linalg.norm(sample,ord=2,axis=1)
    print("norms:",norm)
    normmean = np.mean(norm)
    print("mean:",normmean)
    normdev = np.std(norm)
    print("deviation:",normdev)

    import matplotlib.pyplot as plt

    #plot histogram for exploring distribution of norm on frequency
    plt.hist(norm,bins=50)# divide range of norm value into 50 intervals
    plt.grid()
    plt.title('Histogram of Euclidean Norms')
    plt.xlabel('Norm Values',fontsize=16)
    plt.ylabel('Frequency(Times)',fontsize=16)
```

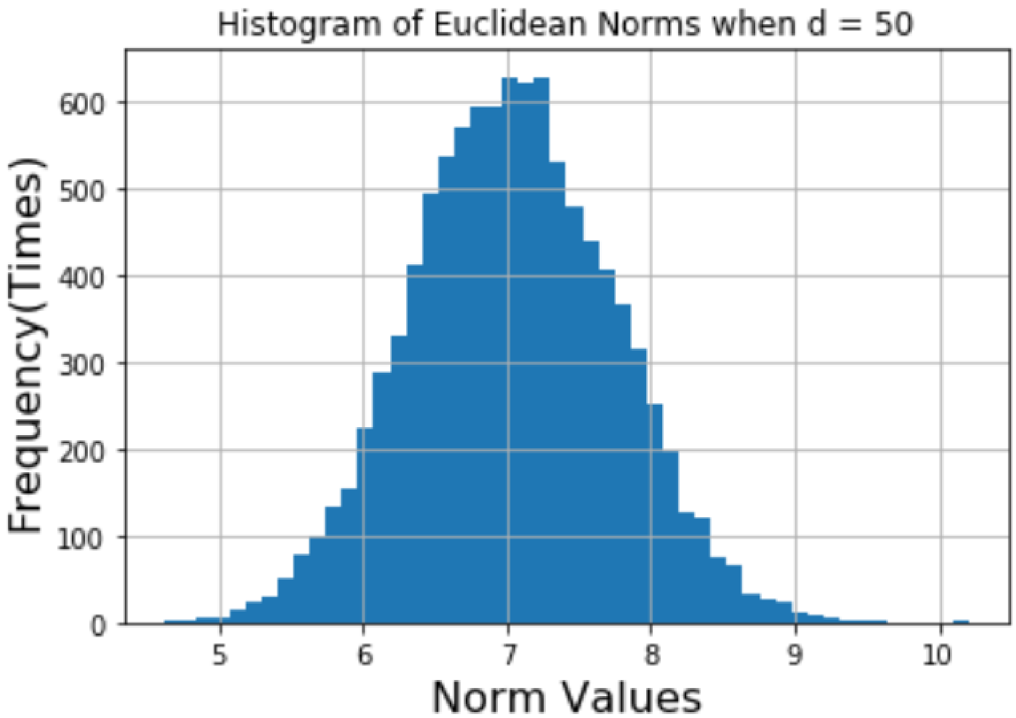
```
norms: [1.27349483 1.89020362 1.43490721 ... 1.17568153 1.88105297 2.20893787]
mean: 1.588787055888442
deviation: 0.6751894171814576
Text(0, 0.5, 'Frequency(Times)')
```

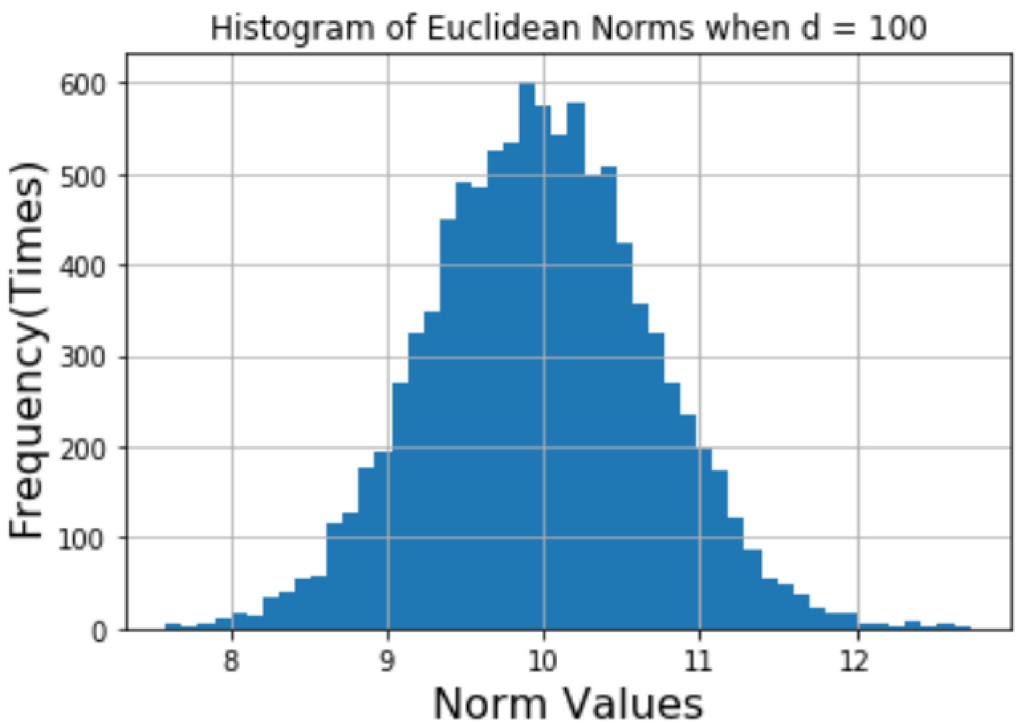


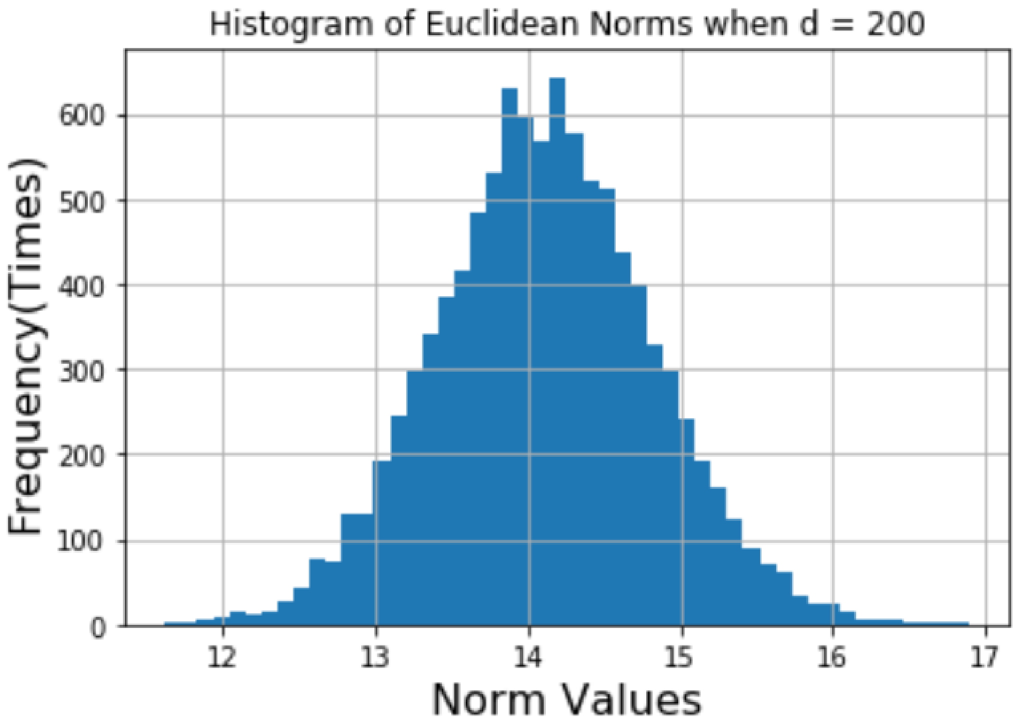
c. Increase d on a coarsely spaced log scale all the way up to d = 1000 (say d = 50; 100; 200; 500; 1000), and repeat parts (a) and (b). Plot the variation of the average and the standard deviation of Euclidean norm of the samples with increasing d.

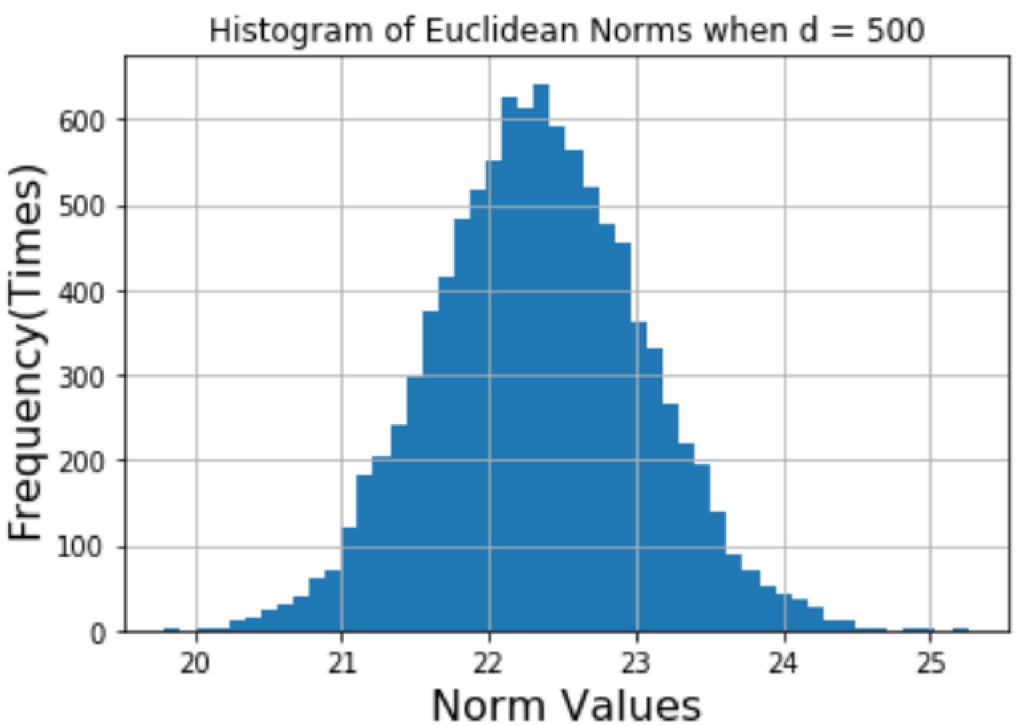
Write a for loop, achieve goals according to the method of part a and b, here are results below:

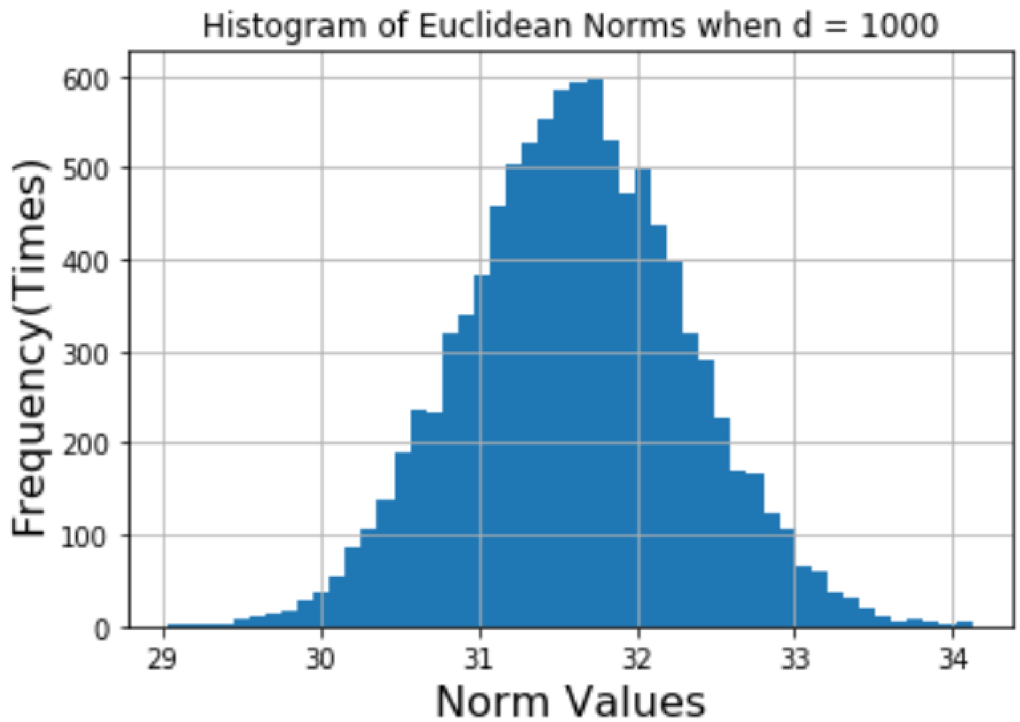
```
norms when d =  50  : [6.53910849 5.63162623 6.19677787 ... 6.38595451 6.84422402 5.67247905]
mean of norms when d =  50  : 7.040054850879492
deviation of norms when d =  50  : 0.7038537219946317
norms when d =  100  : [ 9.88349894  9.47851887 10.14300275 ... 10.44087788  9.73454083
 10.03696166]
mean of norms when d =  100  : 9.979845021709012
deviation of norms when d =  100  : 0.7067810503347449
norms when d =  200  : [13.88327155 14.95508868 13.68816151 ... 14.47573589 14.00834985
 14.48744089]
mean of norms when d =  200  : 14.110547045599942
deviation of norms when d =  200  : 0.7075301652511011
norms when d =  500  : [22.47678157 22.63272807 21.72342571 ... 22.47750188 23.31387571
 21.65822913]
mean of norms when d =  500  : 22.335521539668985
deviation of norms when d =  500  : 0.708271631676346
norms when d =  1000  : [32.91980399 31.52243523 31.43687957 ... 32.43494827 32.65579205
 29.68628511]
mean of norms when d =  1000  : 31.611496517868176
deviation of norms when d =  1000  : 0.7046380157251774
```

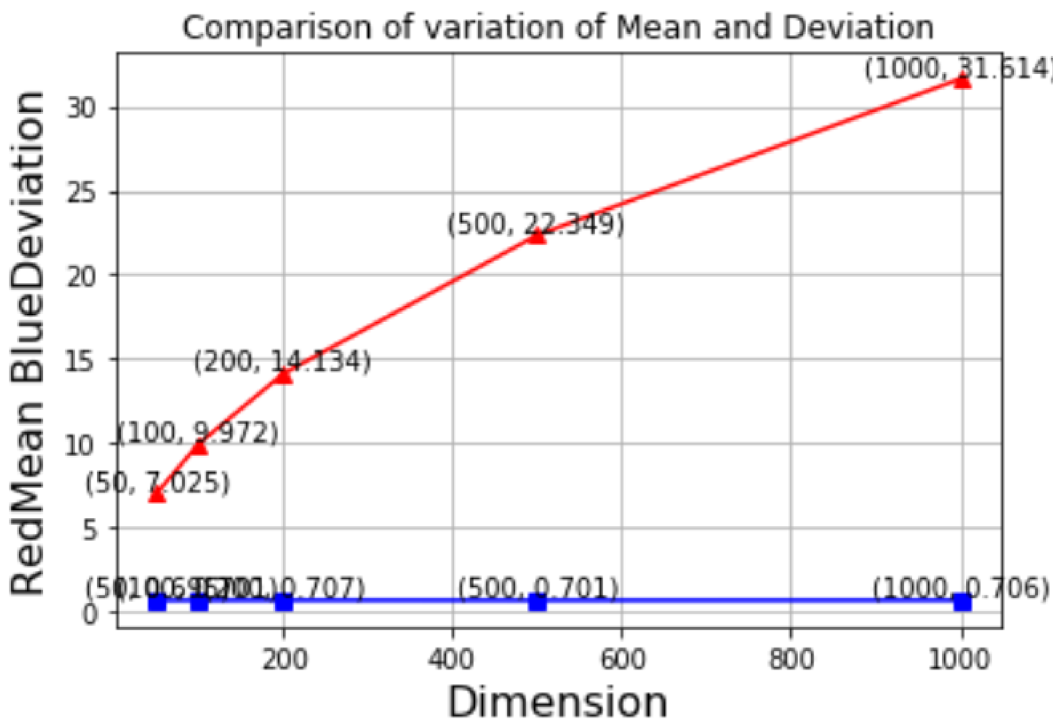

Histogram of Euclidean Norms when d = 50
Frequency(Times)
Norm Values
0
100
200
300
400
500
600
5
6
7
8
9
10

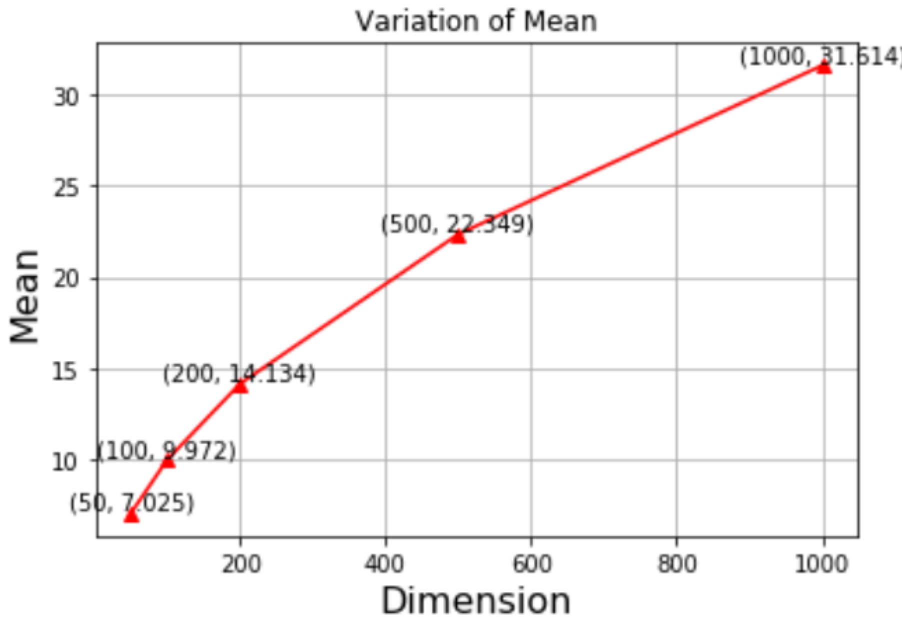

Histogram of Euclidean Norms when d = 100
Frequency(Times)
Norm Values
0
100
200
300
400
500
600
8
9
10
11
12

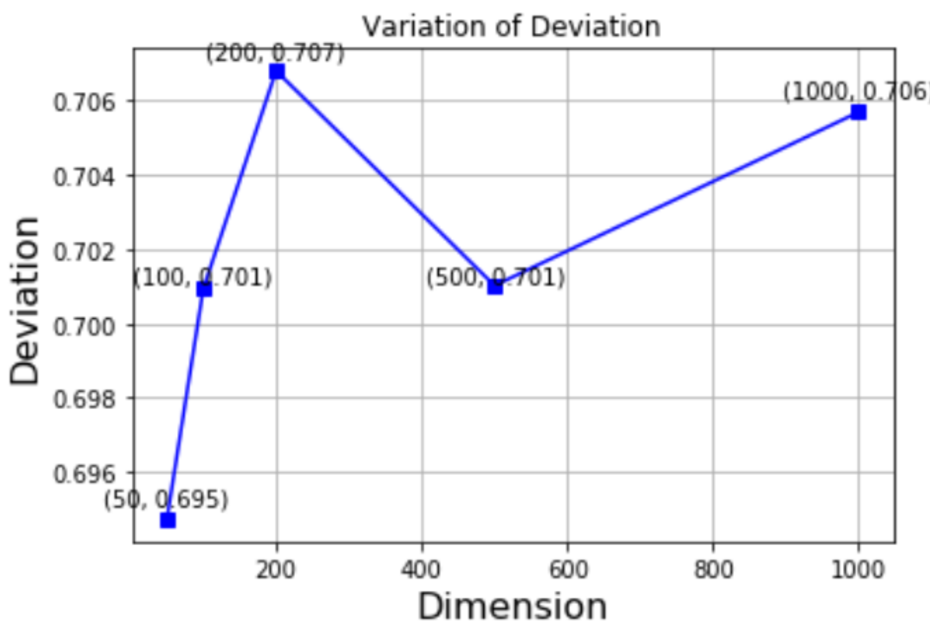

Histogram of Euclidean Norms when d = 200
Frequency(Times)
Norm Values
0
100
200
300
400
500
600
12
13
14
15
16
17


Histogram of Euclidean Norms when d = 500
Frequency(Times)
Norm Values
0
100
200
300
400
500
600
20
21
22
23
24
25


Histogram of Euclidean Norms when d = 1000
Frequency(Times)
Norm Values
0
100
200
300
400
500
600
29
30
31
32
33
34


Comparison of variation of Mean and Deviation
RedMean BlueDeviation
Dimension
(1000, 31.614)
(500, 22.349)
(200, 14.134)
(100, 9.972)
(50, 7.025)
(200, 0.707)
(500, 0.701)
(1000, 0.706)
0
5
10
15
20
25
30
200
400
600
800
1000





d. What can you conclude from your plot from part (c)?

According to diagram above, I discover that:

1.in high dimension, most of mass is concentrated near the mean of Euclidean norms,

2.mean of norms increase with dimension,

3.deviation is approximately not changed with dimension.

Q4:

a. Write a small parser to read each document and convert it into a vector of words.

define an function to read each document and convert it into a vector of words and obtain appearance time of each word, tf of each word and a list of all kinds of word in each .txt document saving as dictionaries

b. Compute tf-idf values for each word in every document as well as the query.

b1. write a function to obtain result of function "preprocess" on each document from paths provided in a list and put appearance counter of words, tf of words in every document and a collection of all kinds of words appear in all documents respectively into three lists : counter, tfdic, all_words

b2.write a function to unify the format of counter and tfdic to make every dictionary have same attributes and dimension, put them respectively into two lists: counters, tfdics

b3. write a function to obtain idf of each words in every document and put them together into one list: idfdics , actually they are all the same but I save them as the same format and container as tf

b4. write a function to obtain tf-idf of each words in every document and put them together into one list: tfidfdics

c. Compute the cosine similarity between tf-idf vectors of each document and the query.

define a function that can calculate cosine similarity between two dictionaries of tf-idf

d. Report the document with the maximum similarity value.

define a recognition function that can use function defined above to compare cosine similarity among documents and return sequence number of result document and maximum value of similarity

results:

d 4 is the document with maximum cosine similarity value of 0.19616295821215432

Q5: approximately 21 hours for whole homework