1. (20 points) [Data Attributes] [Angela Zhang] Classify the following attributes 1) as nominal, ordinal, interval, or ratio and 2) as binary, discrete, or continuous. Some cases may have more than one interpretation, so briefly justify your answer if you think there may be some ambiguity.

(a) (1 point) Temperature measured in Kelvin.

- **Ratio**
- **Continuous**

(b) (1 point) City/town (Raleigh, Durham, Cary, etc.).

- **Nominal (Since it is only distinct. Cannot be ordered, and addition, multiplication does not apply)**
- **Discrete**

(c) (1 point) Annual income (US currency).

- **Ratio (Distinctness, Order, Addition, & Multiplication)**
- **Continuous**

(d) (1 point) Eye color (Blue, Green, Brown, etc.).

- **Nominal (distinctness)**
- **Discrete**

(e) (1 point) Temperature measured in Celsius.

- **Interval (Distinctness, order & addition)**
- **Continous**

(f) (1 point) The amount of Calories found on a nutrition label.

- **Ratio (Distinctness, Order, Addition, Multiplication)**
- **Continuous**

(g) (1 point) Binary numbers (0, 1, 10, 11, 100, 101, ...).

- **Ordinal (You can order it since 0 comes before 1)**
- **Discrete**

(h) (1 point) Blood type (O+, O-, A+, A-, B+, B-, AB+, AB-).

- **Nominal (Distinctness)**
- **Discrete**

(i) (1 point) Letter grade assigned to coursework (A+, A, A-, B+, ..., F).

- **Ordinal (Distinctness & order)**
- **Discrete**

(j) (1 point) The distance traveled by a jogger measured in kilometers.

- **Ratio (Distinctness, Order, Addition, & Multiplication)**
- **If considering whole numbers (i.e. no decimals) then the answer is discrete. If we are to consider decimal values too, then the answer would be continuous.**

(k) (1 point) Unix time (also known as Epoch time, POSIX time, UNIX Epoch time, etc.).

- **Interval (You can order it, and also add to it, but not multiply)**
- **This would be discrete if you only consider whole seconds, and not microseconds or the numbers after the decimal point. If you want to**

consider the microseconds, then continuous. Generally (non-technically) speaking time is continuous.

(l) (1 point) True or false.

- **Nominal (not considering the moral superiority for ordering, i.e. True !> False)**
- **Discrete (there is nothing in between)**

(m) (1 point) CPU processing speed (actual or base) measured in Gigahertz.

- **Ratio**
- **If we are not considering decimal values, and just a whole number then the answer would be discrete. If we are considering decimal values then the answer would be continuous.**

(n) (1 point) A person's height measured in meters.

- **Ratio (Height can be twice of a value)**
- **If we are not considering decimal values, and just a whole number then the answer would be discrete. If we are considering decimal values then the answer would be continuous.**

(o) (1 point) The pH of water that has been measured using the pH scale.

- **Ordinal (you can rank them but not add or multiply them)**
- **Usually, PH values are in decimals. Hence if we consider decimals then the answer would be continuous. If you chose to ignore the decimals and simply work with just the whole numbers then it would be considered discrete.**

(p) (1 point) Day of the week (Sunday, Monday, Tuesday, ...).

- **Ordinal (You can order them in the terms of Monday comes first and Sunday last)**
- **Discrete (there are no decimal days between each whole day)**

(q) (1 point) Calendar dates.

- **Interval (You can add days to it)**
- **Discrete**

(r) (1 point) An angle measured in radians between 0 and 2π.

- **Ratio (you can add and multiply)**
- **Continuous. (Since 1.5(pi) is also valid).**

(s) (1 point) Active Duty Enlisted Basic Military pay grade scale e.g. E-1, E-2, ..., E-9 (for simplicity of the exercise, do not factor in years of service in your answer).

- **Ordinal**
- **Discrete (there is nothing between E-1 and E-2)**

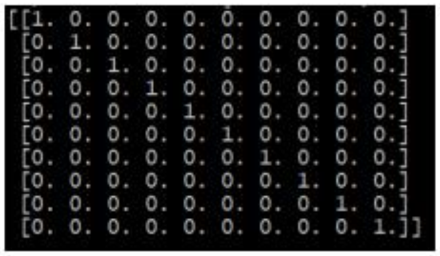
(t) (1 point) The mileage on a vehicle's odometer.

- **Ratio**
- **Continuous.**

2. (15 points) [Matrix Operations] [Ge Gao] Write code in Python to perform the following tasks. Please report your output and relevant code in the document file, and also include your code file (ends with extension .py) in the .zip file.

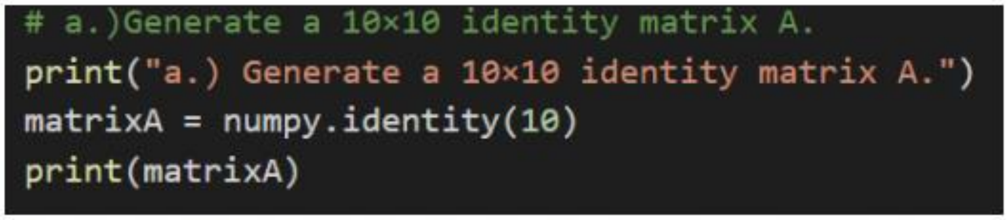
(a) (1 point) Generate a 10×10 identity matrix A.

Ans:



```
[[1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]]
```

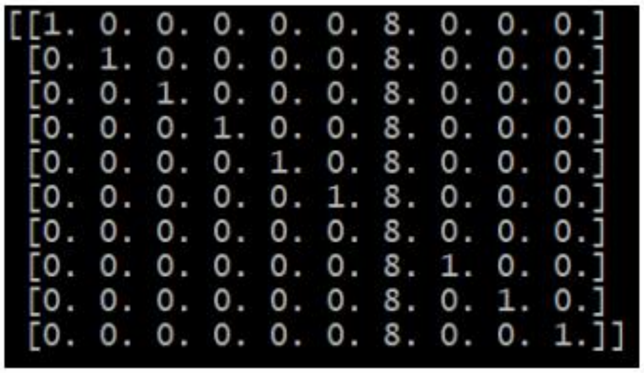
Code:



```python
# a.)Generate a 10×10 identity matrix A.
print("a.) Generate a 10×10 identity matrix A.")
matrixA = numpy.identity(10)
print(matrixA)
```

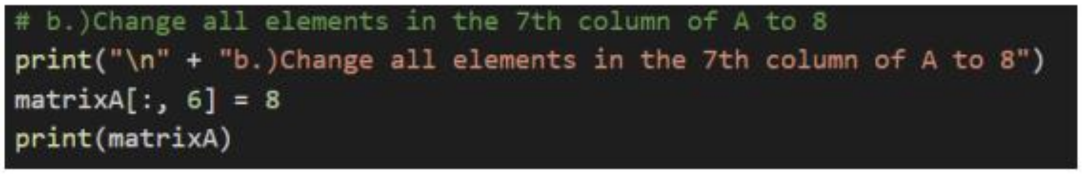
(b) (1 point) Change all elements in the 7th column of A to 8.

Ans:



```
[[1. 0. 0. 0. 0. 0. 8. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 8. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 8. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 8. 0. 0. 0.]
 [0. 0. 0. 0. 1. 0. 8. 0. 0. 0.]
 [0. 0. 0. 0. 0. 1. 8. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 8. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 8. 1. 0. 0.]
 [0. 0. 0. 0. 0. 0. 8. 0. 1. 0.]
 [0. 0. 0. 0. 0. 0. 8. 0. 0. 1.]]
```

Code:

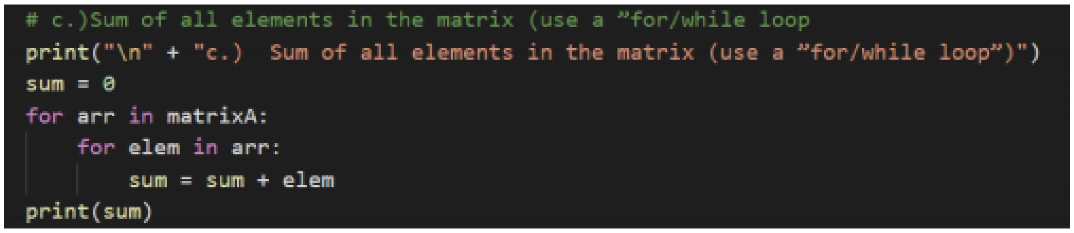


```python
# b.)Change all elements in the 7th column of A to 8
print("\n" + "b.)Change all elements in the 7th column of A to 8")
matrixA[:, 6] = 8
print(matrixA)
```

(c) (1 point) Sum of all elements in the matrix (use a ”for/while loop”).

Ans: 89

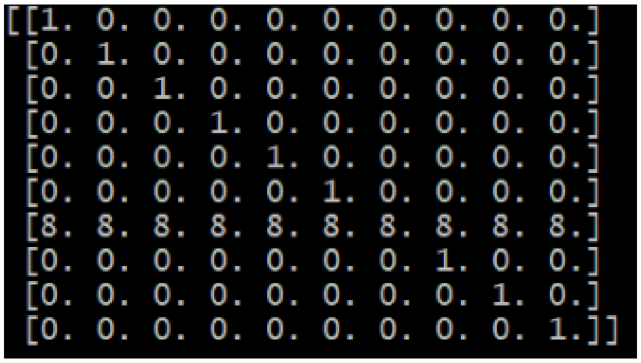
Code:



```
# c.)Sum of all elements in the matrix (use a ”for/while loop
print("\n" + "c.)  Sum of all elements in the matrix (use a ”for/while loop”)")
sum = 0
for arr in matrixA:
    for elem in arr:
        sum = sum + elem
print(sum)
```

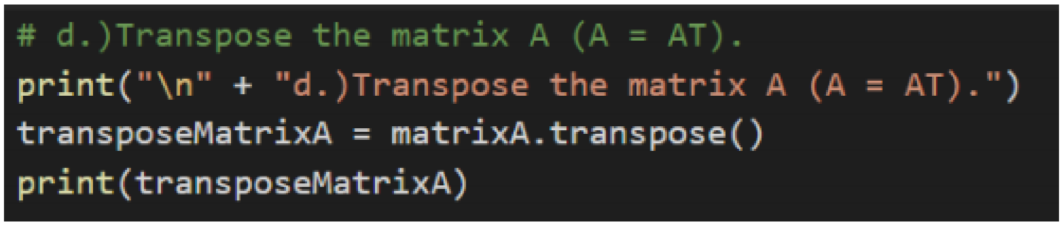
(d) (1 point) Transpose the matrix A (A = AT).

Ans:



```
[[1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
 [8. 8. 8. 8. 8. 8. 8. 8. 8. 8.]
 [0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]]
```

Code:



```
# d.)Transpose the matrix A (A = AT).
print("\n" + "d.)Transpose the matrix A (A = AT).")
transposeMatrixA = matrixA.transpose()
print(transposeMatrixA)
```

(e) (2 points) Calculate sum of the 1st row, the diagonal and the 2nd column in the matrix A.

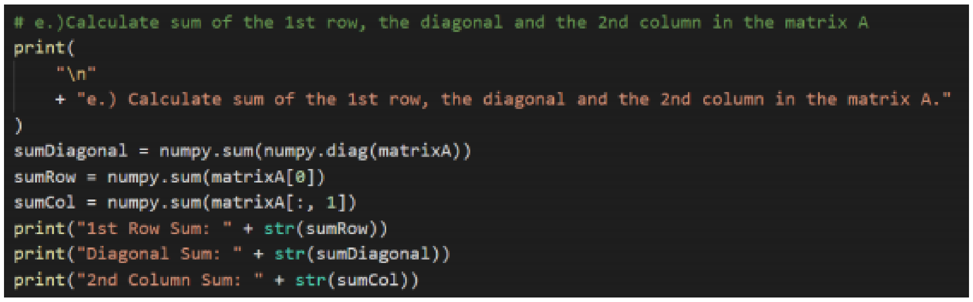
Ans:

1st Row Sum: **9**

Diagonal Sum: **17**

2nd Column Sum : **1**

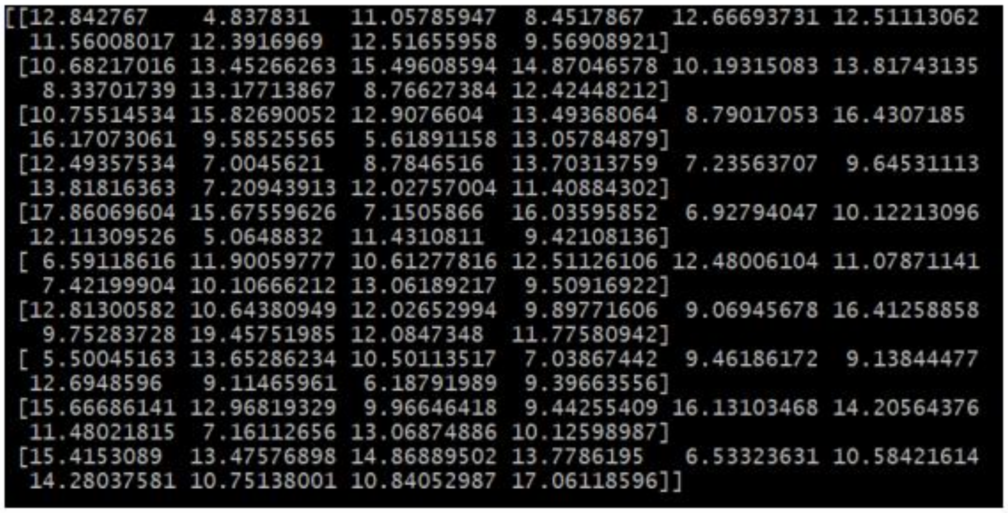
Code:



```
# e.)Calculate sum of the 1st row, the diagonal and the 2nd column in the matrix A
print(
    "\n"
    + "e.) Calculate sum of the 1st row, the diagonal and the 2nd column in the matrix A."
)
sumDiagonal = numpy.sum(numpy.diag(matrixA))
sumRow = numpy.sum(matrixA[0])
sumCol = numpy.sum(matrixA[:, 1])
print("1st Row Sum: " + str(sumRow))
print("Diagonal Sum: " + str(sumDiagonal))
print("2nd Column Sum: " + str(sumCol))
```

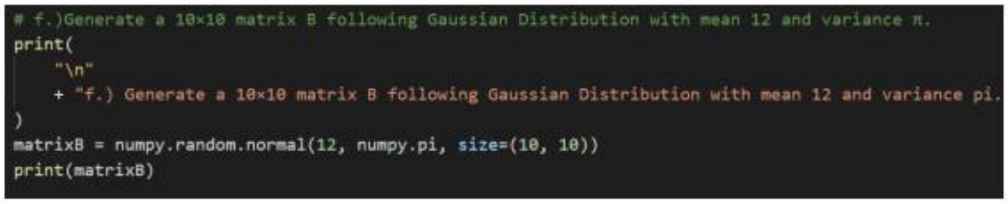
(f) (1 point) Generate a 10×10 matrix B following Gaussian Distribution with mean 12 and variance π.

Ans:



```
[[12.842767    4.837831   11.05785947  8.4517867  12.66693731 12.51113062
  11.56008017 12.3916969  12.51655958  9.56908921]
 [10.68217016 13.45266263 15.49608594 14.87046578 10.19315083 13.81743135
   8.33701739 13.17713867  8.76627384 12.42448212]
 [10.75514534 15.82690052 12.9076604  13.49368064  8.79017053 16.4307185
  16.17073061  9.58525565  5.61891158 13.05784879]
 [12.49357534  7.0045621   8.7846516  13.70313759  7.23563707  9.64531113
  13.81816363  7.20943913 12.02757004 11.40884302]
 [17.86069604 15.67559626  7.1505866  16.03595852  6.92794047 10.12213096
  12.11309526  5.0648832  11.4310811   9.42108136]
 [ 6.59118616 11.90059777 10.61277816 12.51126106 12.48006104 11.07871141
   7.42199904 10.10666212 13.06189217  9.50916922]
 [12.81300582 10.64380949 12.02652994  9.89771606  9.06945678 16.41258858
   9.75283728 19.45751985 12.0847348  11.77580942]
 [ 5.50045163 13.65286234 10.50113517  7.03867442  9.46186172  9.13844477
  12.6948596   9.11465961  6.18791989  9.39663556]
 [15.66686141 12.96819329  9.96646418  9.44255409 16.13103468 14.20564376
  11.48021815  7.16112656 13.06874886 10.12598987]
 [15.4153089  13.47576898 14.86889502 13.7786195   6.53323631 10.58421614
  14.28037581 10.75138001 10.84052987 17.06118596]]
```

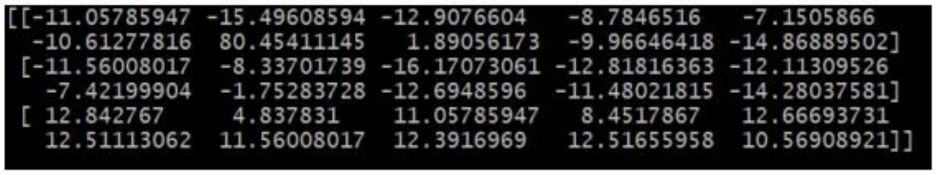
Code:



```
# f.)Generate a 10×10 matrix B following Gaussian Distribution with mean 12 and variance π.
print(
    "\n"
    + "f.) Generate a 10×10 matrix B following Gaussian Distribution with mean 12 and variance pi.
)
matrixB = numpy.random.normal(12, numpy.pi, size=(10, 10))
print(matrixB)
```

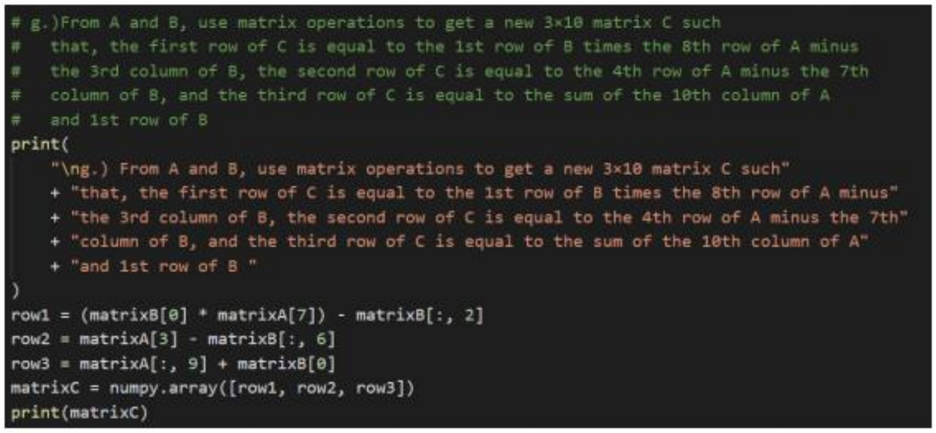
(g) (2 points) From A and B, use matrix operations to get a new 3×10 matrix C such that, the first row of C is equal to the 1st row of B times the 8th row of A minus the 3rd column of B, the second row of C is equal to the 4th row of A minus the 7th column of B, and the third row of C is equal to the sum of the 10th column of A and 1st row of B.

Ans:



```
[[-11.05785947 -15.49608594 -12.9076604   -8.7846516   -7.1505866
  -10.61277816  80.45411145   1.89056173  -9.96646418 -14.86889502]
 [-11.56008017  -8.33701739 -16.17073061 -12.81816363 -12.11309526
   -7.42199904  -1.75283728 -12.6948596  -11.48021815 -14.28037581]
 [ 12.842767     4.837831    11.05785947   8.4517867   12.66693731
   12.51113062  11.56008017  12.3916969   12.51655958  10.56908921]]
```

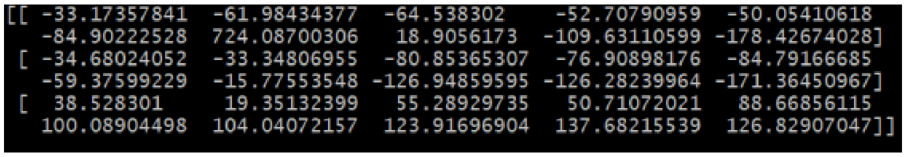
Code:



```
# g.)From A and B, use matrix operations to get a new 3x10 matrix C such
#   that, the first row of C is equal to the 1st row of B times the 8th row of A minus
#   the 3rd column of B, the second row of C is equal to the 4th row of A minus the 7th
#   column of B, and the third row of C is equal to the sum of the 10th column of A
#   and 1st row of B
print(
    "\ng.) From A and B, use matrix operations to get a new 3x10 matrix C such"
    + "that, the first row of C is equal to the 1st row of B times the 8th row of A minus"
    + "the 3rd column of B, the second row of C is equal to the 4th row of A minus the 7th"
    + "column of B, and the third row of C is equal to the sum of the 10th column of A"
    + "and 1st row of B "
)
row1 = (matrixB[0] * matrixA[7]) - matrixB[:, 2]
row2 = matrixA[3] - matrixB[:, 6]
row3 = matrixA[:, 9] + matrixB[0]
matrixC = numpy.array([row1, row2, row3])
print(matrixC)
```

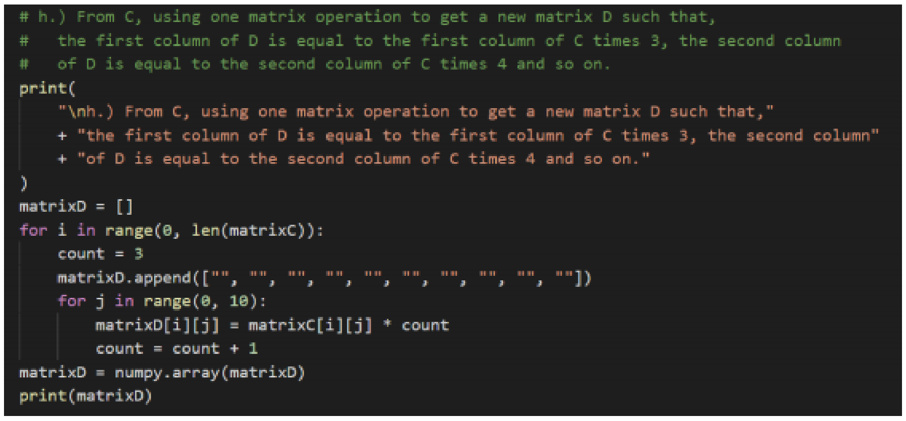
(h) (2 points) From C, using one matrix operation to get a new matrix D such that, the first column of D is equal to the first column of C times 3, the second column of D is equal to the second column of C times 4 and so on.

Ans:



```
[[ -33.17357841  -61.98434377  -64.538302    -52.70790959  -50.05410618
   -84.90222528  724.08700306   18.9056173  -109.63110599 -178.42674028]
 [ -34.68024052  -33.34806955  -80.85365307  -76.90898176  -84.79166685
   -59.37599229  -15.77553548 -126.94859595 -126.28239964 -171.36450967]
 [  38.528301     19.35132399   55.28929735   50.71072021   88.66856115
   100.08904498  104.04072157  123.91696904  137.68215539  126.82907047]]
```

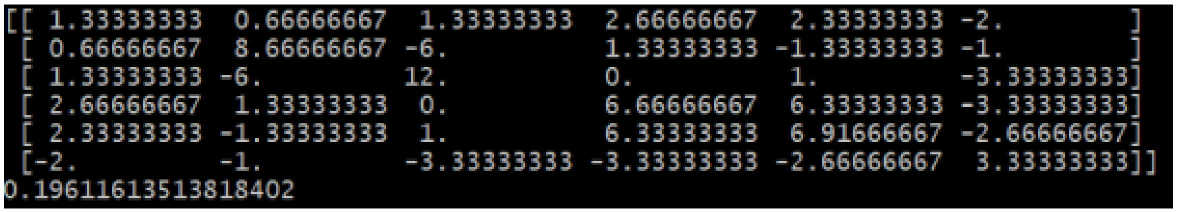
Code:



```
# h.) From C, using one matrix operation to get a new matrix D such that,
#   the first column of D is equal to the first column of C times 3, the second column
#   of D is equal to the second column of C times 4 and so on.
print(
    "\nh.) From C, using one matrix operation to get a new matrix D such that,"
    + "the first column of D is equal to the first column of C times 3, the second column"
    + "of D is equal to the second column of C times 4 and so on."
)
matrixD = []
for i in range(0, len(matrixC)):
    count = 3
    matrixD.append(["", "", "", "", "", "", "", "", "", ""])
    for j in range(0, 10):
        matrixD[i][j] = matrixC[i][j] * count
        count = count + 1
matrixD = numpy.array(matrixD)
print(matrixD)
```

(i) (2 points) X = [2, 0, 2, 0]T , Y = [9, 8, 4, 3]T , Z = [7, 1, 7, 9]T , M = [6, 4, 8, 2]T , N = [5, 4, 9, 3]T , P = [1, 5, 2, 4]T . Compute the covariance matrix of X, Y ,Z, M, N and P. Then compute the Pearson correlation coefficients between X and Y.

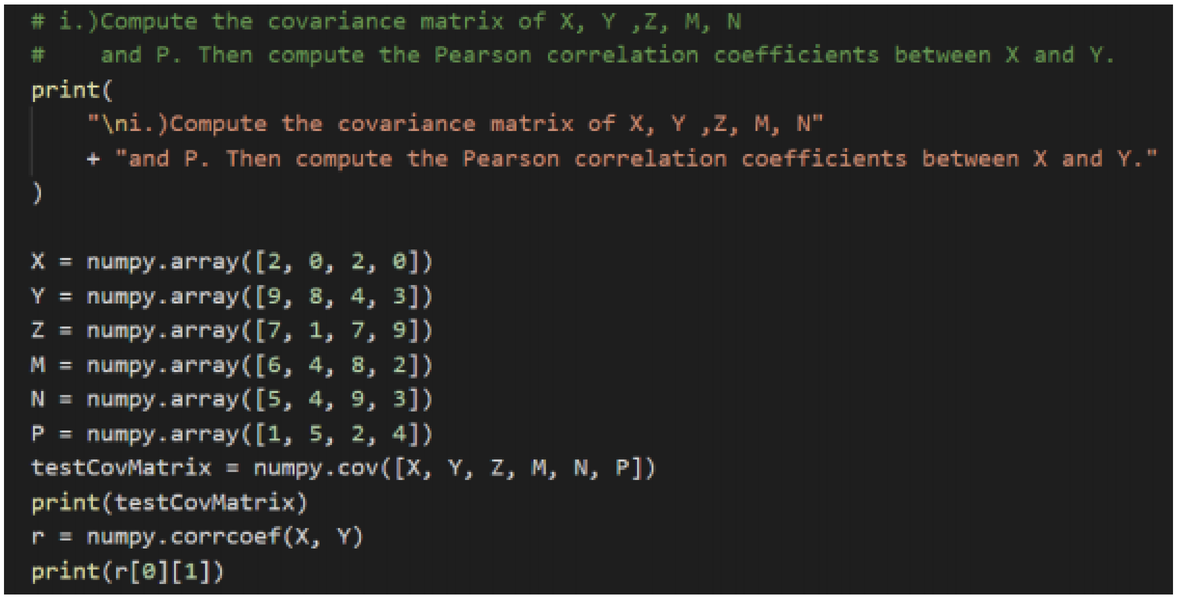
Ans:



```
[[ 1.33333333  0.66666667  1.33333333  2.66666667  2.33333333 -2.        ]
 [ 0.66666667  8.66666667 -6.          1.33333333 -1.33333333 -1.        ]
 [ 1.33333333 -6.         12.          0.          1.         -3.33333333]
 [ 2.66666667  1.33333333  0.          6.66666667  6.33333333 -3.33333333]
 [ 2.33333333 -1.33333333  1.          6.33333333  6.91666667 -2.66666667]
 [-2.         -1.         -3.33333333 -3.33333333 -2.66666667  3.33333333]]
0.19611613513818402
```

Correlation Coefficient: .19611613513818402

Code:

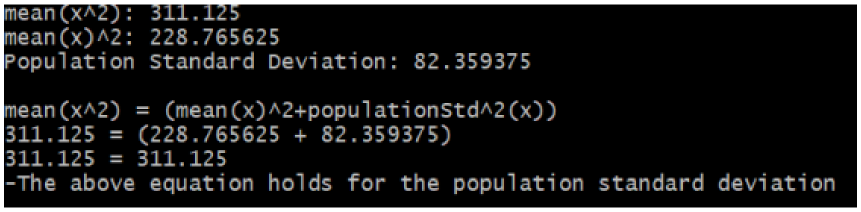


```python
# i.)Compute the covariance matrix of X, Y ,Z, M, N
#    and P. Then compute the Pearson correlation coefficients between X and Y.
print(
    "\ni.)Compute the covariance matrix of X, Y ,Z, M, N"
    + "and P. Then compute the Pearson correlation coefficients between X and Y."
)

X = numpy.array([2, 0, 2, 0])
Y = numpy.array([9, 8, 4, 3])
Z = numpy.array([7, 1, 7, 9])
M = numpy.array([6, 4, 8, 2])
N = numpy.array([5, 4, 9, 3])
P = numpy.array([1, 5, 2, 4])
testCovMatrix = numpy.cov([X, Y, Z, M, N, P])
print(testCovMatrix)
r = numpy.corrcoef(X, Y)
print(r[0][1])
```

(j) (2 points) Given the equation: mean(x^2) = (mean(x)^2 2+sigma^2(x)) where x = [19, 12, 16, 6, 11, 27, 1, 29]T . Please determine whether the equation holds when:

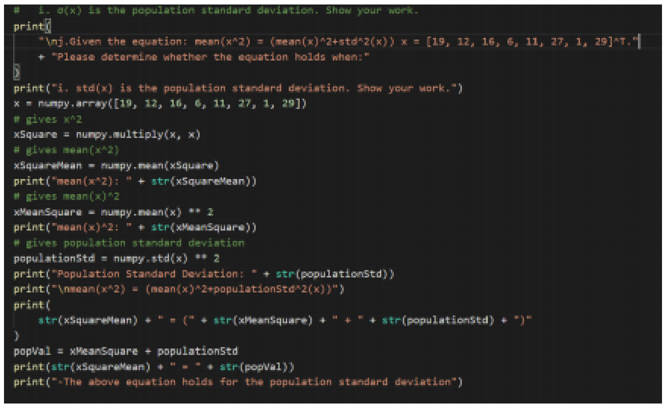
i. σ(x) is the population standard deviation. Show your work.

Ans:



```
mean(x^2): 311.125
mean(x)^2: 228.765625
Population Standard Deviation: 82.359375

mean(x^2) = (mean(x)^2+populationStd^2(x))
311.125 = (228.765625 + 82.359375)
311.125 = 311.125
-The above equation holds for the population standard deviation
```

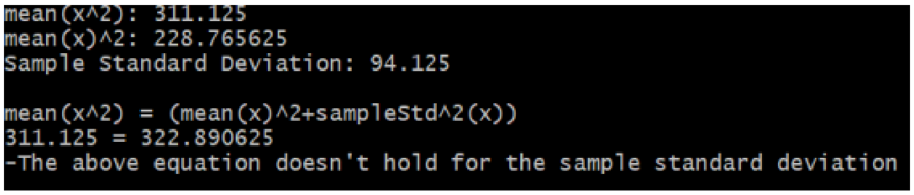
Code:



```python
#   i. σ(x) is the population standard deviation. Show your work.
print(
    "\nj.Given the equation: mean(x^2) = (mean(x)^2+std^2(x)) x = [19, 12, 16, 6, 11, 27, 1, 29]^T."
    + "Please determine whether the equation holds when:"
)
print("i. std(x) is the population standard deviation. Show your work.")
x = numpy.array([19, 12, 16, 6, 11, 27, 1, 29])
# gives x^2
xSquare = numpy.multiply(x, x)
# gives mean(x^2)
xSquareMean = numpy.mean(xSquare)
print("mean(x^2): " + str(xSquareMean))
# gives mean(x)^2
xMeanSquare = numpy.mean(x) ** 2
print("mean(x)^2: " + str(xMeanSquare))
# gives population standard deviation
populationStd = numpy.std(x) ** 2
print("Population Standard Deviation: " + str(populationStd))
print("\nmean(x^2) = (mean(x)^2+populationStd^2(x))")
print(
    str(xSquareMean) + " = (" + str(xMeanSquare) + " + " + str(populationStd) + ")"
)
popVal = xMeanSquare + populationStd
print(str(xSquareMean) + " = " + str(popVal))
print("-The above equation holds for the population standard deviation")
```

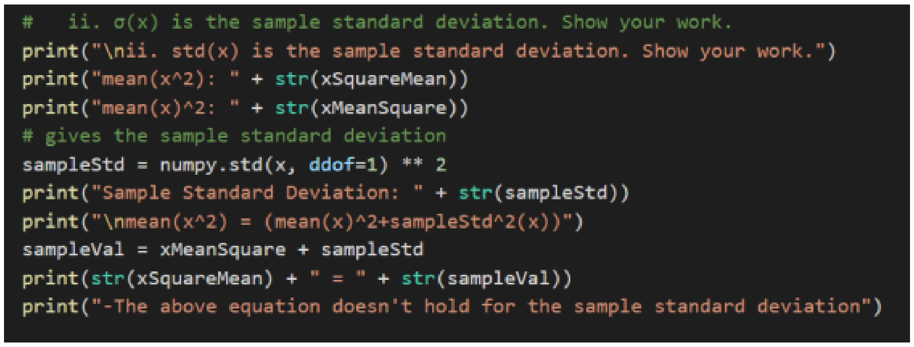
ii. σ(x) is the sample standard deviation. Show your work.

Ans:



```
mean(x^2): 311.125
mean(x)^2: 228.765625
Sample Standard Deviation: 94.125

mean(x^2) = (mean(x)^2+sampleStd^2(x))
311.125 = 322.890625
-The above equation doesn't hold for the sample standard deviation
```

Code:



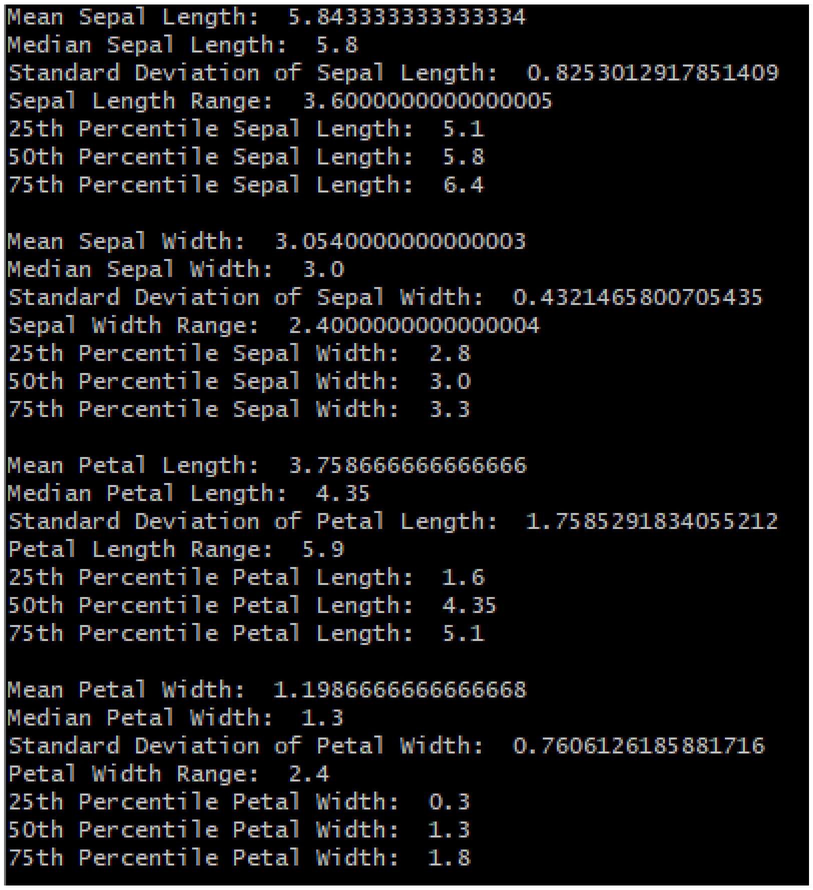
```python
#   ii. σ(x) is the sample standard deviation. Show your work.
print("\nii. std(x) is the sample standard deviation. Show your work.")
print("mean(x^2): " + str(xSquareMean))
print("mean(x)^2: " + str(xMeanSquare))
# gives the sample standard deviation
sampleStd = numpy.std(x, ddof=1) ** 2
print("Sample Standard Deviation: " + str(sampleStd))
print("\nmean(x^2) = (mean(x)^2+sampleStd^2(x))")
sampleVal = xMeanSquare + sampleStd
print(str(xSquareMean) + " = " + str(sampleVal))
print("-The above equation doesn't hold for the sample standard deviation")
```

3. (24 points) [Data Visualization] [John Wesley Hostetter] In this question, please summarize and explore data in the provided file “data\iris.csv”. This data is the “Iris Data Set” and is a famous database created by R.A. Fisher in 1936. It was later donated by Michael Marshall to the UCI Machine Learning Repository (https://archive.ics. uci.edu/ml/datasets/Iris). In this data file, each row represents an instance and has 5 attributes:

1. sepal length in centimeters
2. sepal width in centimeters
3. petal length in centimeters
4. petal width in centimeters
5. class (Iris Setosa, Iris Versicolour, Iris Virginica)

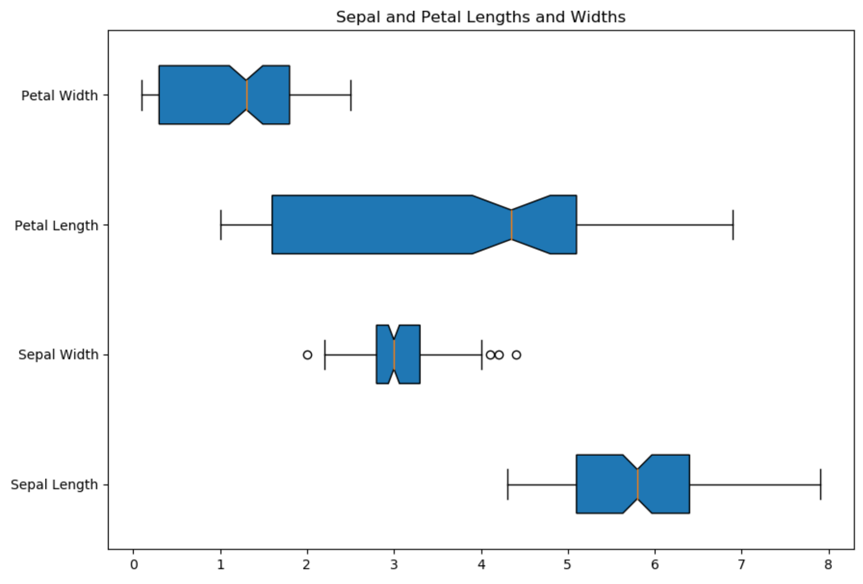
Write code in Python to perform the following tasks. Please report your output and relevant code in the document file, and also include your code file (ends with extension .py) in the .zip file.

(a) (3 points) Compute the mean, median, standard deviation, range, 25th percentiles, 50th percentiles, 75th percentiles for each feature (except the class attribute).

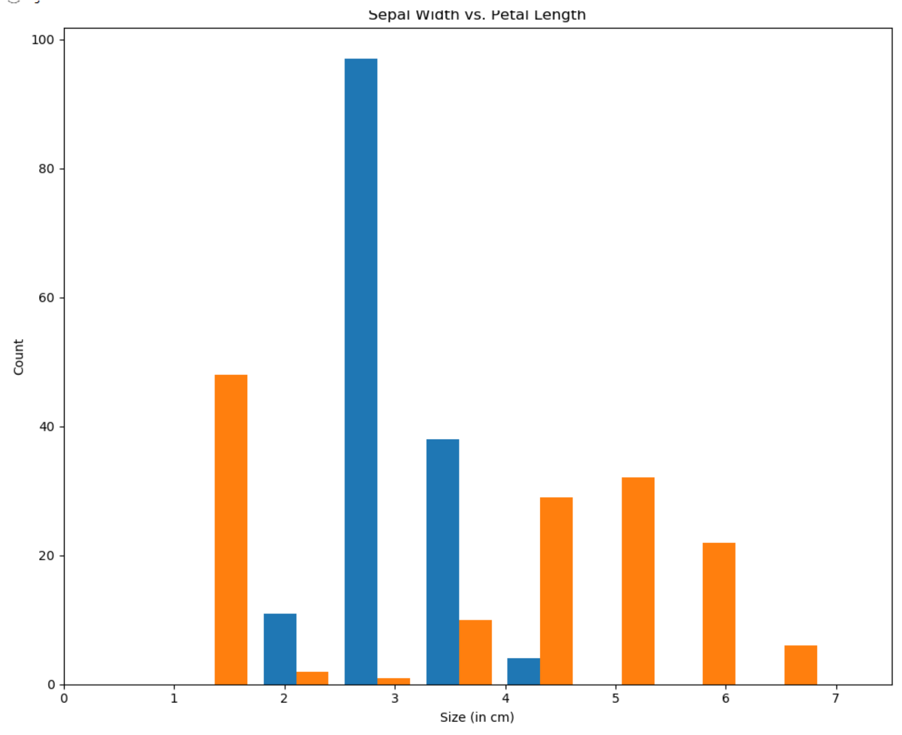


```
Mean Sepal Length:  5.843333333333334
Median Sepal Length:  5.8
Standard Deviation of Sepal Length:  0.8253012917851409
Sepal Length Range:  3.6000000000000005
25th Percentile Sepal Length:  5.1
50th Percentile Sepal Length:  5.8
75th Percentile Sepal Length:  6.4

Mean Sepal Width:  3.0540000000000003
Median Sepal Width:  3.0
Standard Deviation of Sepal Width:  0.4321465800705435
Sepal Width Range:  2.4000000000000004
25th Percentile Sepal Width:  2.8
50th Percentile Sepal Width:  3.0
75th Percentile Sepal Width:  3.3

Mean Petal Length:  3.758666666666666
Median Petal Length:  4.35
Standard Deviation of Petal Length:  1.7585291834055212
Petal Length Range:  5.9
25th Percentile Petal Length:  1.6
50th Percentile Petal Length:  4.35
75th Percentile Petal Length:  5.1

Mean Petal Width:  1.1986666666666668
Median Petal Width:  1.3
Standard Deviation of Petal Width:  0.7606126185881716
Petal Width Range:  2.4
25th Percentile Petal Width:  0.3
50th Percentile Petal Width:  1.3
75th Percentile Petal Width:  1.8
```

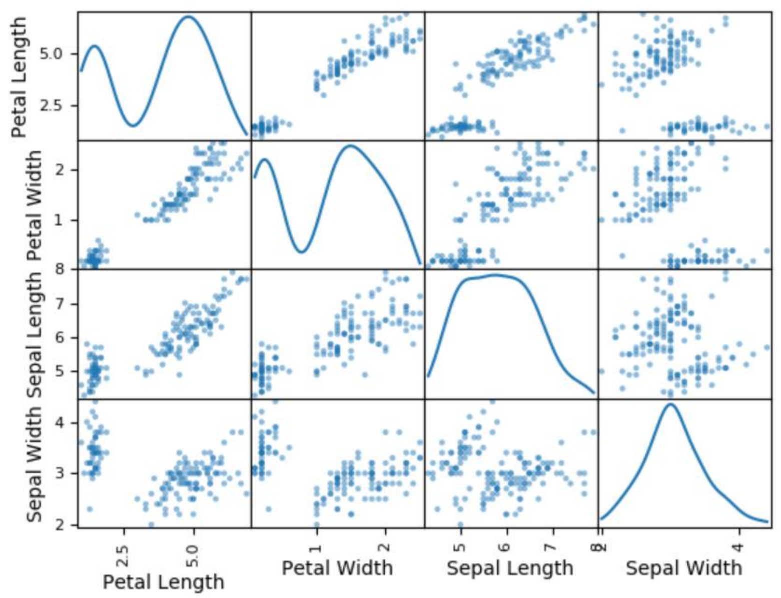
(b) (3 points) Make a box-and-whisker plot for each feature (except the class attribute). Be sure to include a title for each plot of what feature is being described.



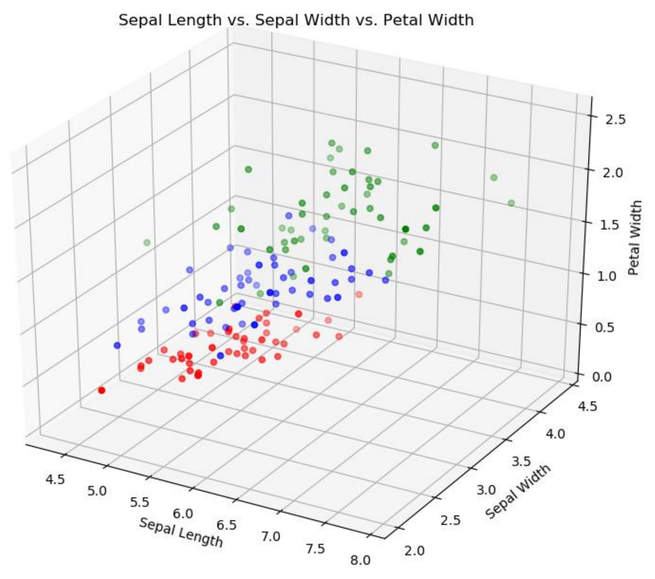
(c) (4 points) Create histogram plot using 8 bins for the two features sepal width and petal length, respectively.



(d) (4 points) Create a scatter matrix of the data. Include all features, but use the class attribute to change the color of the data points (for convenience, you may use a library for this). For the diagonal of the scatter matrix, plot the kernel density estimation (KDE).



(e) (5 points) Now, write code to produce a three-dimensional scatter plot using the sepal length, sepal width and petal width as dimensions, and color the data points according to the class attribute.



(f) (5 points) The quantile-quantile plot can be used for comparing the distribution of data against the normal distribution. Create a quantile-quantile plot for the two features sepal length and petal length, respectively. Give a brief analysis for the two plots.

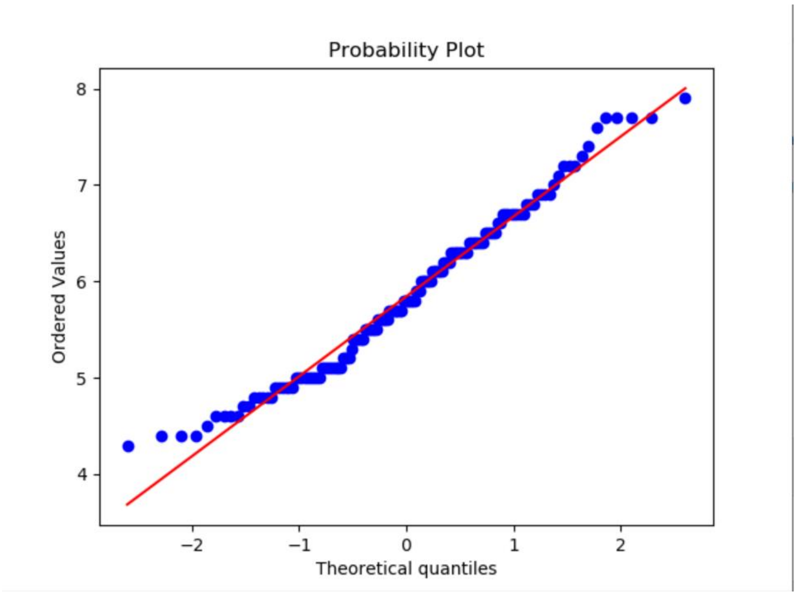


Figure 1: Sepal Length

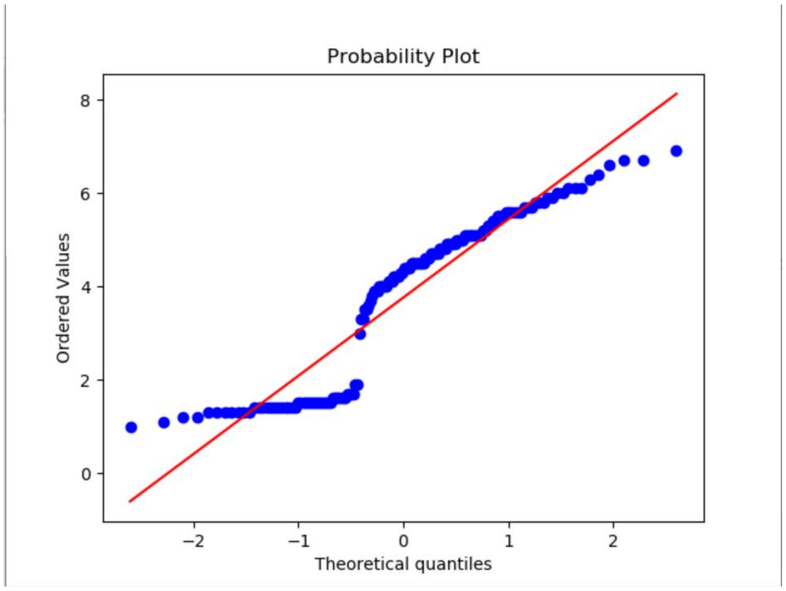


Figure 2: Petal Length

- As seen the Sepal Length is fairly linear indicating that it follows a symmetric normal distribution.

-In the Petal Length QQ plot it is seen that the values displayed have a gap indicating the petal length distribution is bimodal

4. (16 points) [Short Answer Questions] [Angela Zhang] Please read Chapters 2.2 and 2.3 in Tan et. al, textbook and answer the following questions:

(a) (8 points) Data quality.

i. (1 point) How is precision often measured?
-Precision is often measured by the standard deviation of a set of values.

ii. (2 points) When can we reliably determine if there truly exists a bias?
-Bias can only be truly determined when a measured quantity is made known by a means which is external from the current situation. If you were to use internal measures, they would have the bias applied and would not determine anything

iii. (2 points) Explain how accuracy and precision are different from one another.
-I think of accuracy and precision as different if you imagine a target with a bullseye. Precise shots are repeatedly hitting very close to each other but not necessarily on the bullseye. Accurate shots are close to the bullseye but not necessarily close to each other.

iv. (3 points) List and explain at least 3 ways on how to handle missing values.
-The first way you can handle missing data is simply by eliminating either the object with the missing data or the entire attribute with missing data from all objects. With this technique you lose information, information which may be vital to the analysis of the data making your analysis unreliable. This happens often when there is much data missing.

(b) (8 points) Data preprocessing.

i. (2 points) Compare the advantages and disadvantages of data aggregation.
-**Advantages for data aggregation include smaller data sets which reduces memory consumption and processing time. This advantage can be very helpful when needing to use an expensive data mining algorithm. Other advantages are that you can get a higher scoped view of the data and that the data is typically more stable than non-aggregated data.**

ii. (1 point) What is the key principle for effective sampling?
-If the sample has about the same properties as the original set of data then the sample will work almost as well as the original data set.

iii. (2 points) When trying to do effective sampling, how do we determine the proper sampling size? (specifically, name a technique and explain it)
-Adaptive/progressive sampling is a technique to effectively sample. These techniques start with a small sample size and then incrementally increase the sample size until the sample is sufficient.

iv. (2 points) In your own words, explain the curse of dimensionality.
-The curse of dimensionality is when the more attributes/dimensions that the data to be analyzed contains the more difficult it is to pull meaningful data out of the set. For example, if your data set is a glass of water and in that glass of water you dissolve some sugar, when you add more water to the glass the more difficult it will be to tell that the water has sugar in it.

v. (1 point) What is often required for effective feature extraction?

-Having knowledge of the domain for which you are developing and extracting features as they are typically highly domain specific.

5. (11 points) [Sampling] [John Wesley Hostetter] Answer the following questions:

(a) (5 points) A chess dataset contains records for 10,000 unique games, where 5,000 games result in a win for player white and 5,000 games result in a win for player black. Among the 10,000 unique games, 3% witness an en passant move and 11% have a castling move occur at least once on either the King's side or Queen's side. For simplicity, assume that the two events - en passant or castling - are mutually exclusive. Suppose we are developing a classifier in hopes of predicting the outcome of a chess game as it is being played. However, we are unable to use the entire data set due to computational limitations, and thus can only use a sample of the entire data set. Which sampling method would be appropriate and why? If we are sampling 2,000 games from the provided dataset, how many games should be selected from each group using your choice of sampling methods. Briefly justify your answer.

1. 10000 Chess unique games have 5000 wins for the player using whites and 5000 wins for the player using blacks, the probability of whites and blacks are the same. So in a reduced sample color does not affect the result of the game.

2. There are 3% (i.e. 300 out of 10000) en passant move and 11% (i.e. 1100 out of 10000) castling move events which affect the result of the game so reduced data set must contain the same percentage of these events.

3. I will choose stratified sampling to have the same representation of these two events as the original datasets.

4. If we are sampling 2000 samples that is 20% of original samples than we will choose 20% of 300 i.e. 60 games from en passant move games and 20% of 1100 i.e. 220 games from castling move games. Rest 1720 games from other than these events games.

(b) (6 points) Consider the following scenario: The data set originally had 10,127 games, but the 127 games that were omitted from consideration in the previous section did not result in a win for either player, but rather, a draw instead. Are the games that resulted in a draw, noise or outliers? Briefly justify your answer and describe the differences between noise and outliers.

-Noise is a modification or distortion of original values. Whereas outliers are the objects with characteristics that are considerably different than most of the other data objects in the data set.

-A draw in a game is not considerably different than most other outcomes. It is a mere modification hence a draw would be categorized as noise and not as an outlier.

6(a).When $tf_{ij} > 0$, what are the maximum and minimum values of the corresponding tf'_{ij} values based on the above transformation? Please specify what cases the max and min value achieves.

<u>Given:</u>

- $tf'_{ij} = tf'_{ij} * (1 + \sum_0^j \frac{p_{ij} log p_{ij}}{log m})$ where $p_{ij} = \frac{tf_{ij}}{gf_i}$
- $0 < tf_{ij} <= k$
- $gf_i = \sum_1^m tf_{ij}$
  -above represents the number of times ith term appears in all documents
- $1 < m$
  -above represents max number of documents

<u>Proof:</u>
Min:
-Occurs when $tf_{ij} = 1$ (aka only 1 of ith term found in each document)

<u>Step 1</u>: Assume $tf_{ij} = 1$ and solve for p_{ij}
$p_{ij} = \frac{tf_{ij}}{\sum_1^m tf_{ij}} = \frac{1}{\sum_1^m 1} = \frac{1}{m}$

<u>Step 2</u>: plug in the values for p_{ij} and tf_{ij} into the tf'_{ij} formula
$tf'_{ij} = tf_{ij} * (1 + \sum_0^j \frac{p_{ij} log p_{ij}}{log m})$
$= 1 * (1 + \sum_0^j \frac{\frac{1}{m} log \frac{1}{m}}{log m})$
$= 1 + j[\frac{\frac{1}{m} log \frac{1}{m}}{log m}]$
$= 1 + \frac{-j}{m}$
$= \frac{m-j}{m}$

Ans: $= \frac{m-j}{m}$

Max:
-Occurs when $tf_{ij} = k$ (aka only k(max frequency) of ith term is found in each document)

<u>Step 1</u>: Assume $tf_{ij} = k$ and solve for p_{ij}
$p_{ij} = \frac{tf_{ij}}{\sum_1^m tf_{ij}} = \frac{k}{\sum_1^m k} = \frac{k}{km} = \frac{1}{m}$

Step 2: plug in the values for p_{ij} and tf_{ij} into the tf'_{ij} formula

$$tf'_{ij} = tf_{ij} * (1 + \sum_0^j \frac{p_{ij} log p_{ij}}{log m})$$
$$= k * (1 + \sum_0^j \frac{\frac{1}{m} log \frac{1}{m}}{log m})$$
$$= k * (1 + \frac{-j}{m})$$
$$= \frac{k(m-j)}{m}$$

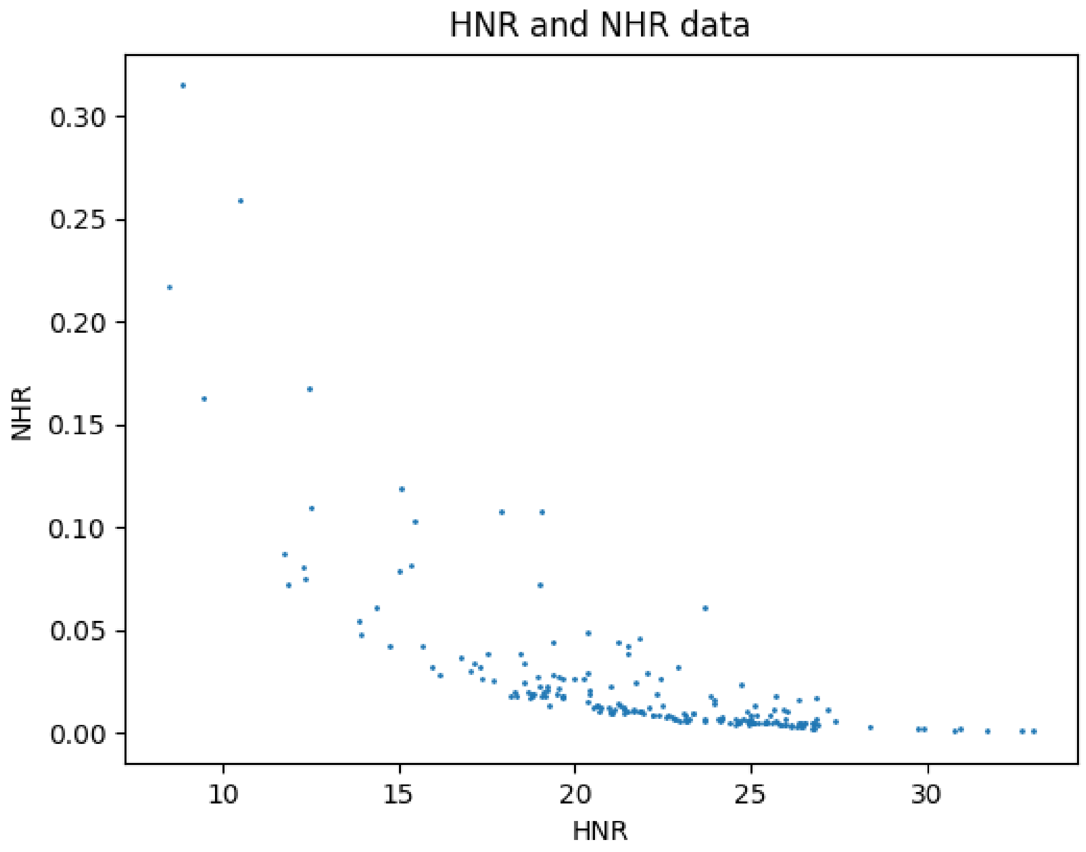
Ans: $= \frac{k(m-j)}{m}$

b.)Briefly explain the purpose for this transformation in the context of natural language processing.

Ans: The purpose of this transformation is to get the average occurrence of the ith term found from the first to the jth document.

7. (30 points) [Distance] [John Wesley Hostetter] For this exercise, use the provided file "data\parkinsons.csv" (https://archive.ics.uci.edu/ml/datasets/Parkinsons), which contains a list of 195 data instances. Each of these instances represents a recording. There are 23 attributes; please refer to the included link for documentation. For this exercise, we will only be concerned with a select few - namely, the HNR and NHR attributes. Write code in Python to perform the following tasks, please report your output and relevant code in the document file, and also include your code file (ends with .py) in the .zip file.

(a) (4 points) Load the file and read all the columns. Make a plot between the HNR and the NHR of the observations. Label the axes (NHR should be y-axis and HNR should be x-axis). Call this plot "HNR and NHR data". What general interpretation can you make from this plot?

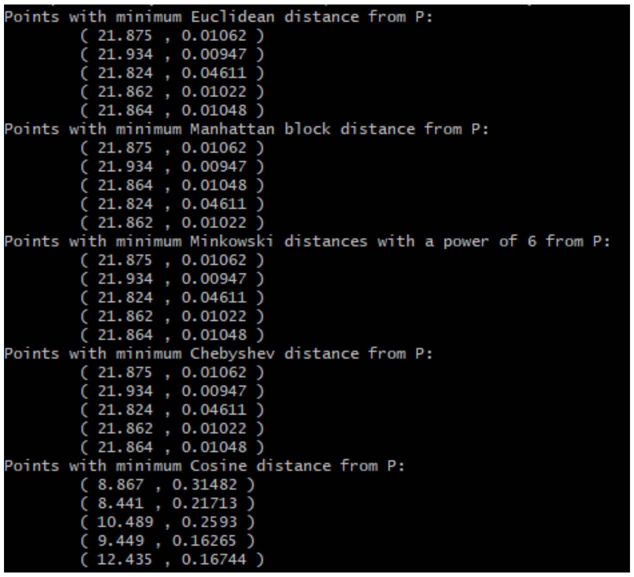


-From the HNR and NHR data plot you can see that the relationship between these two attributes is logarithmic. As HNR increases, NHR decreases logarithmically coming to a lower limit of zero and vice versa. Also, the typical subject seems to have an HNR of around 17-27 and a NHR of around 0 to 0.05.

(b) (2 points) Compute the mean of the attributes HNR and NHR. Define a data point called P such that P = (mean(HNR), mean(NHR)).

-Data point P = (21.885974358974366, 0.024847076923076923)

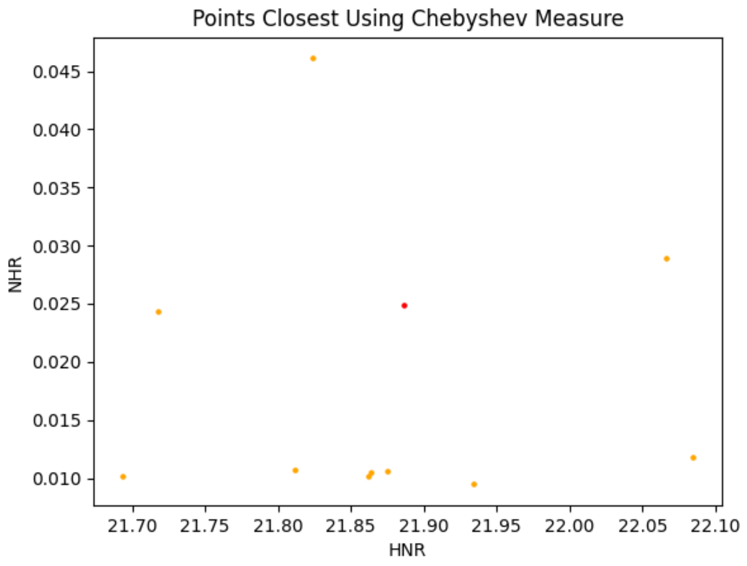
(c) (10 points) Compute the distance between P and the 195 data points using the following distance measures: 1) Euclidean distance, 2) Manhattan block metric, 3) Minkowski metric (for power=6), 4) Chebyshev distance, and 5) Cosine distance. List the closest 5 points for each distance. (d) For each distance measure, identify the 10 points from the dataset that are the closest to the point P from

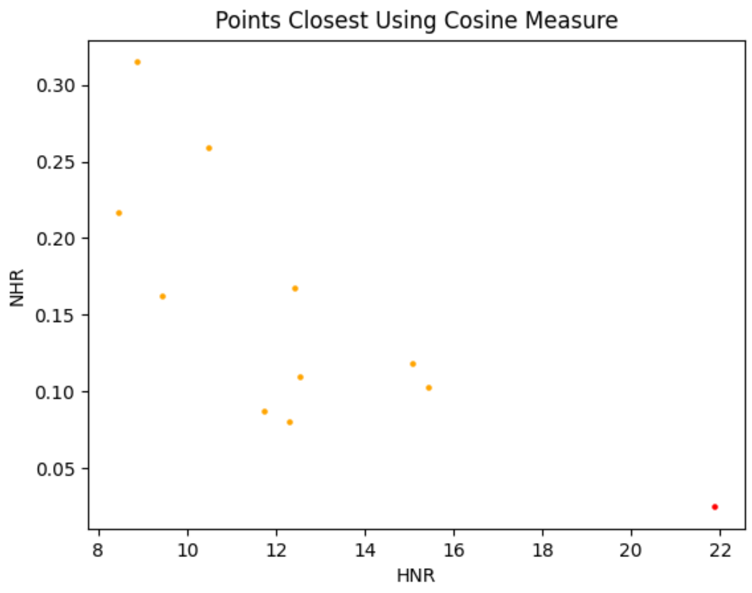


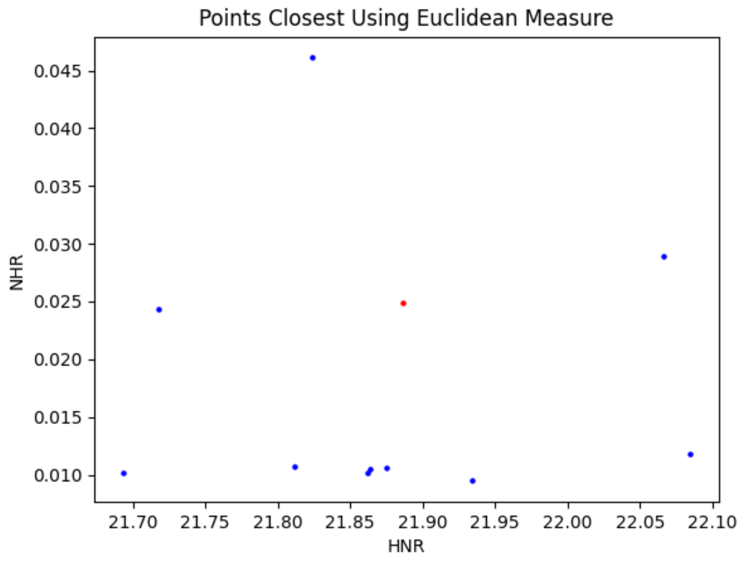
```
Points with minimum Euclidean distance from P:
        ( 21.875 , 0.01062 )
        ( 21.934 , 0.00947 )
        ( 21.824 , 0.04611 )
        ( 21.862 , 0.01022 )
        ( 21.864 , 0.01048 )
Points with minimum Manhattan block distance from P:
        ( 21.875 , 0.01062 )
        ( 21.934 , 0.00947 )
        ( 21.864 , 0.01048 )
        ( 21.824 , 0.04611 )
        ( 21.862 , 0.01022 )
Points with minimum Minkowski distances with a power of 6 from P:
        ( 21.875 , 0.01062 )
        ( 21.934 , 0.00947 )
        ( 21.824 , 0.04611 )
        ( 21.862 , 0.01022 )
        ( 21.864 , 0.01048 )
Points with minimum Chebyshev distance from P:
        ( 21.875 , 0.01062 )
        ( 21.934 , 0.00947 )
        ( 21.824 , 0.04611 )
        ( 21.862 , 0.01022 )
        ( 21.864 , 0.01048 )
Points with minimum Cosine distance from P:
        ( 8.867 , 0.31482 )
        ( 8.441 , 0.21713 )
        ( 10.489 , 0.2593 )
        ( 9.449 , 0.16265 )
        ( 12.435 , 0.16744 )
```

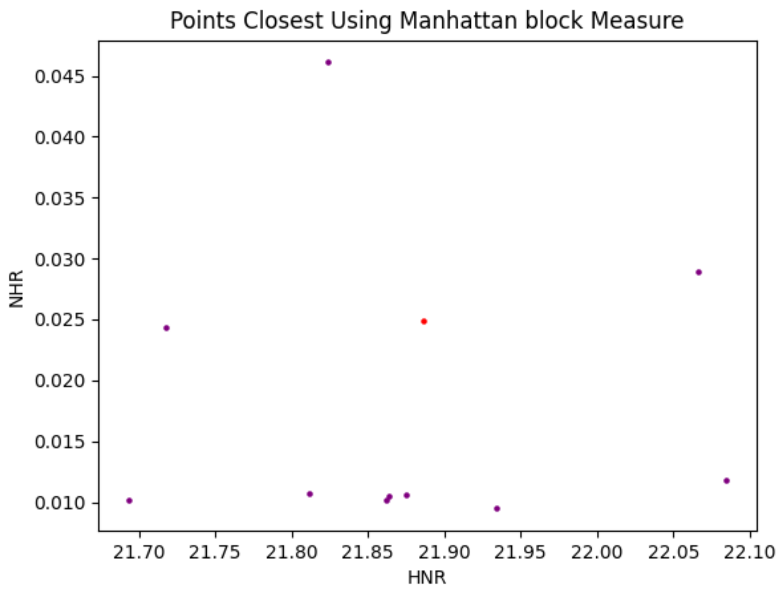
(d). (You are allowed to use any package functions to calculate the distances.)

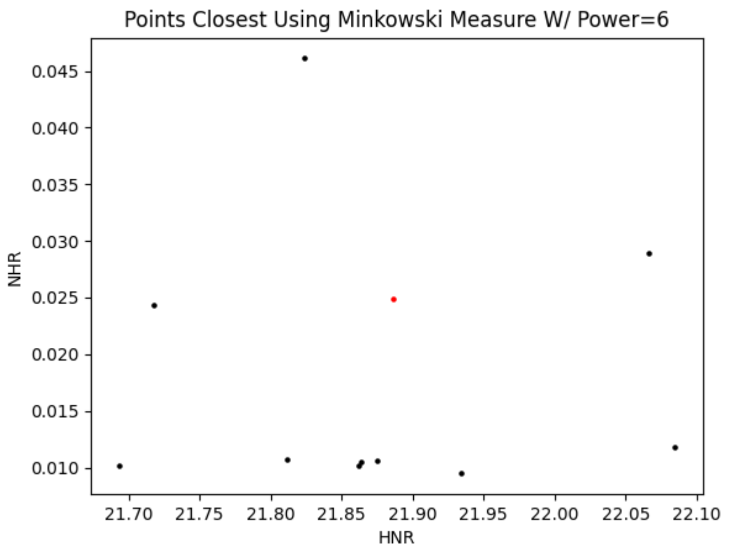
i. (10 points) Create plots, one for each distance measure. Place P on the plot and mark the 10 closest points. To mark them, you could use different colors or shapes. Make sure the points can be uniquely identified.




Points Closest Using Cosine Measure
NHR
HNR
0.30
0.25
0.20
0.15
0.10
0.05
8
10
12
14
16
18
20
22


Points Closest Using Euclidean Measure
NHR
HNR
0.045
0.040
0.035
0.030
0.025
0.020
0.015
0.010
21.70
21.75
21.80
21.85
21.90
21.95
22.00
22.05
22.10





ii. (4 points) Verify if the set of points is the same across all the distance measures. If there is any big difference, briefly explain why it is.

-All sets of points are the same across distance measures with the exception of the cosine distance scatterplot. This plot is wildly different from the pattern. I believe that this is due to the cosine distance being based on angular distance while the others don't take this into account as greatly.