- Q1. Assume there are 120 students in our B565 class, and each person has a 2% of chance of carrying coronavirus. We also know that the Omicron variant dominates and let's assume that it accounts for 90% of the new Covid cases. What's the probability that the entire class is free of coronavirus and Omicron, respectively? Show your work. [10 pts]
- i. The probability that the entire class is free of coronavirus:

If each student has a 2% chance of carrying coronavirus, then each student has a 98% chance of not carrying coronavirus.

1-0.02=0.98 is the probability that a student doesn't have the coronavirus.

Hence, if a class of 120 students doesn't have coronavirus, the probability of that happening is **0.98^120=0.08853787272** which is 8.8% approximately. (We multiply the probability of each student not having coronavirus).

ii. The probability that the entire class is free of Omicron:

If each student has a 2% chance of carrying coronavirus, then each student has a 0.02*0.9=0.018 probability of carrying Omicron variant. The probability of a student not having Omicron is 1-0.018=0.982.

Hence, if a class of 120 students doesn't have coronavirus, the probability of that happening is **0.982^120=0.0.11307810828** which is 11.3% approximately. (We multiply the probability of each student not having Omicron).

Q2. Let Ω be the space of possible outcomes of a fair die (with six sides) thrown twice. What is Ω ? Let A be the event that a 4 is observed on either individual throw or the sum of both throws is at least 5. Let B be the event that the difference between the two throws is exactly two. Are A and B independent? What is the probability of B given A? What is the probability of A given B? Show your work. [15 pts]

i. Ω ={(1, 1),(1, 2),(1, 3),(1, 4),(1, 5),(1, 6),(2, 1),(2, 2),(2, 3),(2, 4),(2, 5),(2, 6),(3, 1),(3, 2),(3, 3), (3, 4),(3, 5),(3, 6),(4, 1),(4, 2),(4, 3),(4, 4),(4, 5),(4, 6),(5, 1),(5, 2),(5, 3),(5, 4),(5, 5),(5, 6), (6, 1),(6, 2),(6, 3),(6, 4),(6, 5),(6, 6)}. Size of Ω is 36. $n(\Omega)$ =36

ii. A=Event when a 4 is observed on either individual throw or the sum of both throws is at least 5

 $A = \{(1, 4), (1, 5), (1, 6), (2, 3), (2, 4), (2, 5), (2, 6), (3, 2), (3, 3), (3, 4), (3, 5), (3, 6), (4, 1), (4, 2), (4, 3), (4, 4), (4, 5), (4, 6), (5, 1), (5, 2), (5, 3), (5, 4), (5, 5), (5, 6), (6, 1), (6, 2), (6, 3), (6, 4), (6, 5), (6, 6)\}$ n(A) = 36.

B=The event that the difference between the two throws is exactly two. B= $\{(1,3),(2,4),(3,5),(4,6),(3,1),(4,2),(5,3),(6,4)\}$. n(B)=8

Set $A \cap B = \{(2,4), (3,5), (4,6), (4,2), (5,3), (6,4)\}$. $n(A \cap B) = 6$. A and B are not independent as $A \cap B$ is not empty.

 $P(A)=n(A)/n(\Omega)=30/36=5/6$

 $P(B)=n(B)/n(\Omega)=30/36=8/36=2/9$

 $P(A \cap B) = n(A \cap B)/n(\Omega) = 6/36 = 1/6$

iii. $P(B|A)=(P(A|B)P(B))/P(A)=P(A\cap B)/P(A)=(1/6)/(5/6)=1/5$

iv. $P(A|B)=(P(B|A)P(A))/P(B)=P(A\cap B)/P(B)=(1/6)/(2/9)=3/4$

Q3. Recall that given two vectors x and y of n dimensions, their cosine similarity, Euclidean distance, and correlation can be computed as follows. [15 pts]

Cosine similarity, $cos(x, y) = (x \cdot y)/||x||.||y||$

Euclidean distance, $d(x, y) = (\sum n i=1(xi - yi)^2)^1/2$

Correlation, corr(x, y) = Sxy/(SxSy)

=(
$$\sum n i=1 (xi - x^-)(yi - y^-)$$
)/($\sum n i=1 (xi - x^-)^2$)^1/2.($\sum n i=1 (yi - y^-)^2$)^1/2

Sets can also be represented as vectors of zeros and ones, so for those vectors, Jaccard similarity (intersection over union) can be used. For the following vectors x and y, calculate the indicated similarity or distance measures. Show the steps.

- x = (1, 1, 1, 1), y = (2, 2, 2, 2) cosine, correlation and Euclidean.
- x = (0, 1, 0, 1), y = (1, 0, 1, 0), cosine, correlation, Euclidean, Jaccard
- x = (0, -1, 0, 1), y = (1, 0, -1, 0), cosine, correlation, Euclidean
- x = (1, 1, 0, 1, 0, 1), y = (1, 1, 1, 0, 0, 1) cosine, correlation, Jaccard
- x = (2, -1, 0, 2, 0, -3), y = (-1, 1, -1, 0, 0, -1) cosine, correlation

i.
$$x = (1, 1, 1, 1), y = (2, 2, 2, 2)$$

a. Cosine Similarity:

 $||x|| = (1^2 + 1^2 + 1^2 + 1^2)^{(1/2)} = 2$

 $||y||=(2^2+2^2+2^2+2^2)^{(1/2)}=4$

$$cos(x, y) = (x \cdot y)/||x||.||y||=8/(2*4)=1$$

b. Correlation:

$$\begin{array}{l} x^-=(\sum n \ i=1(xi))/n=(1+1+1+1)/4=1 \\ y^-=(\sum n \ i=1(yi))/n=(2+2+2+2)/4=2 \\ (\sum n \ i=1 \ (xi-x^-)(yi-y^-) \)=(1-1)^*(2-2)+(1-1)^*(2-2)+(1-1)^*(2-2)+(1-1)^*(2-2)=0 \\ (\sum n \ i=1 \ (xi-x^-)^2)=(1-1)^2+(1-1)^2+(1-1)^2+(1-1)^2=0 \\ (\sum n \ i=1 \ (xi-x^-)^2)^4/2=0 \\ (\sum n \ i=1 \ (yi-y^-)^2)=(2-2)^2+(2-2)^2+(2-2)^2+(2-2)^2=0 \\ (\sum n \ i=1 \ (yi-y^-)^2)^4/2=0 \end{array}$$

Since, both the numerator and the denominator are 0, the correlation for vectors **x** and **y** are not defined.

c. Euclidean distance:

$$\sum n i=1(xi-yi)^2=(1-2)^2+(1-2)^2+(1-2)^2+(1-2)^2=4*(-1)^2=4$$

($\sum n i=1(xi-yi)^2)^1/2=4^(1/2)=2$

- ii. x = (0, 1, 0, 1), y = (1, 0, 1, 0)
 - a. Cosine Similarity:

$$x.y=0.1+1.0+0.1+1.0=0$$

 $||x||=(0^2+1^2+0^2+1^2)^{(1/2)}=2^{(1/2)}$
 $||y||=(1^2+0^2+1^2+0^2)^{(1/2)}=2^{(1/2)}$
 $\cos(x, y) = (x \cdot y)/||x||.||y||=0$

b. Correlation:

$$x = (\sum n i = 1(xi))/n = (0+1+0+1)/4 = 1/2$$

 $y = (\sum n i = 1(yi))/n = (1+0+1+0)/4 = 1/2$

$$\begin{array}{l} (\sum n \ i=1 \ (xi-x^-)(yi-y^-)) \\ = (0-1/2)(1-1/2)+(1-1/2)(0-1/2)+(0-1/2)(1-1/2)+(1-1/2)(0-1/2)=(-1/4)^*4=-1 \end{array}$$

$$(\sum n \ i=1 \ (xi-x^-)^2) = (0-1/2)^2 + (1-1/2)^2 + (0-1/2)^2 + (1-1/2)^2 = 4*1/4 = 1$$

$$(\sum n \ i=1 \ (xi-x^-)^2)^1/2 = 1$$

$$(\sum n \ i=1 \ (yi-y^-)^2) = (1-1/2)^2 + (0-1/2)^2 + (1-1/2)^2 + (0-1/2)^2 = 4*1/4 = 1$$

$$(\sum n \ i=1 \ (yi-y^-)^2)^1/2 = 1$$
 Hence.

$$corr(x, y) = Sxy/(SxSy) = -1/1*1=-1$$

c. Euclidean distance:

$$\sum n i=1(xi-yi)^2=(0-1)^2+(1-0)^2+(0-1)^2+(1-0)^2=4$$

($\sum n i=1(xi-yi)^2)^1/2=4^4(1/2)=2$

d. Jaccard Coefficient:

iii.
$$x = (0, -1, 0, 1), y = (1, 0, -1, 0)$$

a. Cosine Similarity:

$$x.y=0.1+-1.0+0.-1+1.0=0$$
 $||x||=(0^2+(-1)^2+0^2+1^2)^(1/2)=2^(1/2)$
 $||y||=(1^2+0^2+(-1)^2+0^2)^(1/2)=2^(1/2)$
 $\cos(x, y) = (x \cdot y)/||x||.||y||=0$

b. Correlation:

$$x=(\sum n i=1(xi))/n=(0+-1+0+1)/4=0$$

 $y=(\sum n i=1(yi))/n=(1+0+-1+0)/4=0$

$$\begin{split} &(\sum n \ i=1 \ (xi-x^-)(yi-y^-)) \\ &= (0)(1)+(-1)(0)+(0)(-1)+(1)(0)=0 \\ &(\sum n \ i=1 \ (xi-x^-)^2)=(0)^2+(-1)^2+(0)^2+(1)^2=2 \\ &(\sum n \ i=1 \ (xi-x^-)^2)^1/2=2^1/2 \\ &(\sum n \ i=1 \ (yi-y^-)^2)=(1)^2+(0)^2+(-1)^2+(0)^2=2 \end{split}$$

 $(\sum n i=1 (yi - y^{-})^{2})^{1/2}=2^{1/2}$

Hence,

$$corr(x, y) = Sxy/(SxSy) = 0$$

c. Euclidean distance:

$$\sum n i=1(xi-yi)^2=(0-1)^2+(-1-0)^2+(0-(-1))^2+(1-0)^2=4$$

($\sum n i=1(xi-yi)^2)^1/2=4^(1/2)=2$

iv.
$$x = (1, 1, 0, 1, 0, 1), y = (1, 1, 1, 0, 0, 1)$$

a. Cosine Similarity:

$$x.y=1.1+1.1+0.1+1.0+0.0+1.1=3$$

 $||x||=(1^2+1^2+0^2+1^2+0^2+1^2)^{(1/2)}=4^{(1/2)}=2$
 $||y||=(1^2+1^2+1^2+0^2+0^2+1^2)^{(1/2)}=4^{(1/2)}=2$
 $\cos(x, y) = (x \cdot y)/||x||.||y||=3/(2*2)=0.75$

b. Correlation:

$$x = (\sum n i=1(xi))/n=(1+1+0+1+0+1)/6=4/6=2/3$$

 $y = (\sum n i=1(yi))/n=(1+1+1+0+0+1)/6=4/6=2/3$

$$(\sum n i=1 (xi - x^-)(yi - y^-))$$

= $(1-2/3)(1-2/3)+(1-2/3)(1-2/3)+(0-2/3)(1-2/3)+(1-2/3)(0-2/3)+(0-2/3)(0-2/3)+(1-2/3)(1-2/3)$
= $1/9+1/9-2/9-2/9+4/9+1/9=3/9=1/3$

 $(\sum n i=1 (xi-$

$$x^{-}$$
)^2)= $(1-2/3)^2+(1-2/3)^2+(0-2/3)^2+(1-2/3)^2+(0-2/3)^2+(1$

$$(\sum n i=1 (xi - x^{-})^{2})^{1/2}=(4/3)^{1/2}$$

 $(\sum n i=1 (yi -$

$$y^{-}$$
)^2)= $(1-2/3)^2+(1-2/3)^2+(1-2/3)^2+(0-1/2)^2+(0-1/2)^2+(1-2/3)^2=1/9+1/9+1/9+4/9+1/9$ =4/3

$$(\sum n i=1 (yi - y^{-})^{2})^{1/2}=(4/3)^{1/2}$$

Hence,

$$corr(x, y) = Sxy/(SxSy) = 1/3/(4/3)^1/2*(4/3)^1/2=0.25$$

c. Jaccard Coefficient:

$$JC = (f11)/(f01+f10+f11)=3/(1+1+3)=0.6$$

v.
$$x = (2, -1, 0, 2, 0, -3), y = (-1, 1, -1, 0, 0, -1)$$

a. Cosine Similarity:

$$\begin{array}{l} x.y=2.-1+-1.1+0.-1+2.0+0.0+-3.-1=-2-1+3=0\\ ||x||=(2^2+-1^2+0^2+2^2+0^2+-3^2)^*(1/2)=18^*(1/2)\\ ||y||=(-1^2+1^2+1^2+0^2+0^2+0^2+-1^2)^*(1/2)=4^*(1/2)=2\\ \cos(x,y)=(x\cdot y)/||x||.||y||=0/(2^*(18)^*1/2)=0.0 \end{array}$$

 $corr(x, y) = Sxy/(SxSy) = 0/((18^{1/2})*((10/3)^{1/2})) = 0$

b. Correlation:

$$\begin{array}{l} x^-=(\sum n \ i=1(xi))/n=(2-1+0+2+0-3)/6=0 \\ y^-=(\sum n \ i=1(yi))/n=(-1+1-1+0+0-1)/6=-2/6 \\ \\ (\sum n \ i=1 \ (xi-x^-)(yi-y^-)) \\ =(2)(-1+2/6)+(-1)(1+2/6)+(0)(1+2/6)+(2)(-2/6)+(0)(+2/6)+(-3)(-1+2/6)=0 \\ \\ (\sum n \ i=1 \ (xi-x^-)^2)=(2)^2+(-1)^2+(0)^2+(2)^2+0^2+(-3)^2=18 \\ \\ (\sum n \ i=1 \ (xi-x^-)^2)^4/2=18^4/2 \\ \\ (\sum n \ i=1 \ (yi-y^-)^2)^2=(-1+2/6)^2+(1+2/6)^2+(1+2/6)^2+(-2/6)^2+(+2/6)^2+(-1+2/6)^2=10/3 \\ \\ (\sum n \ i=1 \ (yi-y^-)^2)^4/2=(10/3)^4/2 \\ \\ \text{Hence,} \end{array}$$

mean std min 25% **50**% **75**% max 8 rows × 572 columns In [4]: class_column = microbiome['Class'].tolist() In [5]: microbiome = microbiome.drop('Class', 1) In [6]: **from sklearn.decomposition import** PCA pca = PCA(n_components=2) pca.fit(microbiome) #the results will show that PC1 dominates for the original data print("variance explained", pca.explained_variance_ratio_, "singular_val ues", pca.singular_values_) print(pca.explained_variance_ratio_.sum()) #transform the data according to the PCA results microbiome_transformed = pca.transform(microbiome) variance explained [0.29922075 0.06936047] singular_values [337.41686876 162.45270424] 0.3685812134598305 Using PCA, the data matrix is projected to its first two principal components. The projected values of the original data are stored in a pandas DataFrame object named projected. In [8]: projected = pd.DataFrame(microbiome_transformed,columns=['pc1','pc2'],in dex=range(1,345))projected['Class'] =class_column projected Out[8]: pc2 Class pc1 **1** -2.943049 -2.248375 **2** -7.215752 1.835718 **3** 24.051681 -4.850785 n **4** -8.684207 6.431731 n **5** 2.565706 -3.969398 **340** -7.294597 1.556875 **341** -2.864927 18.011306 n **342** -8.629940 -7.938661 **343** -6.603719 -2.611676 **344** -8.060710 -9.187330 344 rows × 3 columns In [9]: import matplotlib.pyplot as plt colors = {'n':'b', 't2d':'r'} markerTypes = {'n':'+', 't2d':'x'} for classType in markerTypes: d = projected[projected['Class']==classType] plt.scatter(d['pc1'],d['pc2'],c=colors[classType],s=60,marker=marker Types[classType]) plt.xlabel("PC1") plt.ylabel("PC2") Out[9]: Text(0, 0.5, 'PC2') 40 30 20 2 10 -10-20 -30ó 60 20 In [10]: # #apply standardscaler to scale the data from sklearn.preprocessing import StandardScaler microbiome_scaled = StandardScaler().fit_transform(microbiome) first_column = np.array(microbiome_scaled[:, 0]) print(first_column[:10], "...") print("mean %.2f" % first_column.mean()) print("var %.2f" % first_column.var()) $\hbox{-0.17202823 } \hbox{-0.17202823 } \hbox{-0.17202823 } \hbox{-0.17202823] } \dots$ mean -0.00 var 1.00 In [11]: pca_scaled = PCA(n_components=2) pca_scaled.fit(microbiome_scaled) print("variance explained", pca_scaled.explained_variance_ratio_, "singu lar_values", pca_scaled.singular_values_) #print(microbiome_scaled) microbiome_scaled_transformed = pca_scaled.transform(microbiome_scaled) variance explained [0.05774328 0.0388078] singular_values [106.59282118 87.38497417] In [12]: projected_scaled = pd.DataFrame(microbiome_scaled_transformed,columns=['pc1', 'pc2'], index=range(1, 345)) projected_scaled['Class'] =class_column projected_scaled Out[12]: pc2 Class pc1 **1** -1.141012 -0.439461 **2** -0.968041 -0.449722 n **3** 3.600214 -0.127970 **4** -0.564952 -0.295627 **5** -0.814761 -0.414652 **340** 1.395982 0.923275 n **341** -0.391177 -0.200669 **342** -0.622712 -0.350434 n **343** 4.654344 0.850055 **344** 2.848240 0.739558 344 rows × 3 columns In [13]: import matplotlib.pyplot as plt colors = {'n':'b', 't2d':'r'} markerTypes = {'n':'+', 't2d':'x'} **for** classType **in** markerTypes: d = projected_scaled[projected_scaled['Class']==classType] plt.scatter(d['pc1'],d['pc2'],c=colors[classType],s=60,marker=marker Types[classType]) plt.xlabel("PC1") plt.ylabel("PC2") Out[13]: Text(0, 0.5, 'PC2') 80 60 20 100 PC1 Finally, we draw a scatter plot to display the projected values. We created 2 diagrams: one with normal data and one with scaled data. We can see that the class 't2d' values overlap with the 'n' class values, in many cases. Hence, there is not much variability captured using PCA with 2 components. From the given output, the variance is very low. Hence, PCA is a good approach to reduce this dataset. Now, we will perform t-SNE on this dataset. In [14]: #t-sne: from sklearn.manifold import TSNE from numpy import reshape import seaborn as sns microbiome1 = pd.read_csv("T2D_abundance.csv", delim_whitespace=True) microbiome1.head() Out[14]: $k_Archaea|p_Euryarchaeota|c_Methanobacteria|o_Methanobacteriales|f_Methanobacteriaceae$ con-001 con-002 con-003 con-004 con-005 5 rows × 573 columns In [15]: class_column1 = microbiome1['Class'].tolist() class_column1 Out[15]: ['n', 'n' 'n', 't2d', 'n', 'n' 'n', 't2d', 't2d' 't2d', 't2d' 't2d' 't2d' 't2d' 't2d' 't2d' 't2d', 't2d', 't2d', 't2d' 't2d' 't2d' 't2d' 't2d' 't2d', 't2d', 'n', 'n', 'n', 'n', 'n' 'n', 'n', 'n', 'n', 'n', 'n', 'n', 'n' 'n', 'n', 'n', 'n', 'n', 'n', 'n' 'n' 'n', 'n', 'n', 'n', 'n', 'n', 'n' 'n' 'n' 'n', 'n', 'n', 'n', 'n', 'n' 'n', 'n'] In [16]: microbiome1 = microbiome1.drop('Class', 1) In [17]: x = microbiome1y = class_column1 Then, we'll define the model by using the TSNE class, here the n_components parameter defines the number of target dimensions. In [18]: tsne = TSNE(n_components=2, verbose=1, random_state=123) $z = tsne.fit_transform(x)$ [t-SNE] Computing 91 nearest neighbors... [t-SNE] Indexed 344 samples in 0.124s... [t-SNE] Computed neighbors for 344 samples in 0.291s... [t-SNE] Computed conditional probabilities for sample 344 / 344 [t-SNE] Mean sigma: 9.567743 [t-SNE] KL divergence after 250 iterations with early exaggeration: 67.8 94165 [t-SNE] KL divergence after 1000 iterations: 0.865274 Now, we will visualize the data. In [19]: df = pd.DataFrame() df["y"] = ydf["comp-1"] = z[:,0]df["comp-2"] = z[:,1]sns.scatterplot(x="comp-1", y="comp-2", hue=df.y.tolist(), palette=sns.color_palette("hls", 2), data=df).set(title="Microbiome Data T-SNE projecti on") Out[19]: [Text(0.5, 1.0, 'Microbiome Data T-SNE projection')] Microbiome Data T-SNE projection 20 10 -10-20 n t2d -10 10 15 20 comp-1 t-SNE offers slightly better separation of the classes and there is not much overlap between the data points. It offers good dimensionality reduction too. Q5. The plot below was used to demonstrate the Curse of Dimensionality. Implement a code to simulate your own data, and generate your special plot of curse of dimensionality. Try dimensions from 2 to 50 with a step size of 1. And for each dimension, randomly generate 500 data points. Use Euclidean distance. [30 pts] We are assuming A is the set of points given and B is the reference point, if you compute the distance from B to each of the points in B, the difference between the maximum distance and the minimum distance goes to 0. In [23]: import numpy as np import matplotlib.pyplot as plt import math log_values = [] for step in range(2,51): # Generate 500 random points in N dimensions. $A = [np.random.randint(-50, 50, step) for _ in range(500)]$ B = np.random.randint(-50,50,step) # Computing distances as per the given problem #https://www.delftstack.com/howto/numpy/calculate-euclidean-distanc diffs = [np.linalg.norm(a-B) for a in A] mxd = max(diffs)mnd = min(diffs) #taking log of whole equation log_value = math.log10((mxd-mnd)/mnd) log_values.append(log_value) # Plotting the curve for simulated data using matplotlib plt.plot(range(2,51),log_values) plt.xlabel('Number of dimensions') plt.ylabel('log10((MAX_DIST - MIN_DIST)/MIN_DIST)') plt.show()

10

import matplotlib.pyplot as plt

smoother curve.

import math

plt.show()

0.7 0.6 0.6 0.5 0.4 0.3 0.2 0.1 0.0 0.0 0.0

References:

In []:

log_values = []

for step in range(2,51):

mxd = max(diffs)
mnd = min(diffs)

In [24]: import numpy as np

20

30

If we take log of only the numerator in the function for the y axis, we get a better and much

 $A = [np.random.randint(-50, 50, step) for _ in range(500)]$

#https://www.delftstack.com/howto/numpy/calculate-euclidean-distanc

Number of dimensions

Generate 500 random points in N dimensions.

Computing distances as per the given problem

Plotting the curve for simulated data using matplotlib

plt.ylabel('log10((MAX_DIST - MIN_DIST)/MIN_DIST)')

diffs = [np.linalg.norm(a-B) for a in A]

B = np.random.randint(-50,50,step)

#taking log of whole equation

log_values.append(log_value)

plt.plot(range(2,51),log_values)
plt.xlabel('Number of dimensions')

log_value = math.log10(mxd-mnd)/mnd

20

[3]. https://www.kaggle.com/morecoding/scalingpca-demo/

[4]. https://www.kaggle.com/morecoding/datapreprocessing

[5]. https://www.delftstack.com/howto/numpy/calculate-euclidean-distance/

30

[1]. https://stackoverflow.com/questions/69247773/generating-a-curse-of-dimensionality-chart

[2]. https://www.datatechnotes.com/2020/11/tsne-visualization-example-in-python.html

Number of dimensions

50

50

Q4. Analyze a microbiome dataset. [30 pts]

First, we will perform PCA on this dataset

• This dataset includes the microbiome profiles of 344 people, some with type 2 diabetes, and others without. The microbiome profile for a person stores the relative abundance of different bacterial specie found in the stool sample collected from that person. The last column shows the

• Perform PCA and t-SNE on the dataset and visualize the data in 2D space. In the plots, each

• Report what you learn from the PCA analyses. How much variability of the data is captured by using only two dimensions? Is PCA a good approach for dimensionality reduction for this dataset?

• Does t-SNE result in a good dimensionality reduction of this dataset? Why or why not

microbiome = pd.read_csv("T2D_abundance.csv", delim_whitespace=True)

k__Archaea|p__Euryarchaeota|c__Methanobacteria|o__Methanobacteriales|f__Methanobacteriaceae

k__Archaea|p__Euryarchaeota|c__Methanobacteria|o__Methanobacteriales|f__Methanobacteriacea

class (with diabetes or not), and the other columns are for the relative abundances.

Do you see clusters of people according to their disease status?

• The dataset is available here.

data point is a user.

In [1]: import pandas as pd

con-001 con-002 con-003 con-004

Out[1]:

import numpy as np

microbiome.head()

5 rows × 573 columns

microbiome.describe()

In [2]: microbiome.shape

Out[2]: (344, 573)

count

In [3]:

Out[3]: