CS 412, HW #1

Pouya Akbarzadeh (pa2)

February 24, 2021

Question 1.

1.1 Answers

A) Maximum and minimum.
Mid-Term 37 , 100
Final: 35 , 100

B) First quartile Q1, median, and third quartile Q3
Mid-Term: 68.0 , 77.0 , 87.0
Final: 82.0 , 89.0 , 96.0

C) Mean
Mid-Term: 76.715
Final: 87.084

D) Mode
Mid-Term: 77 83
Final: 97

E) Variance
Mid-Term: 173.10577
Final: 119.11294

1.2 Explanation and Equations

A) Numpy was used to find min and max
B) Numpy's function percentile() was used. 25,50,75 for Q1, Q2, and Q3 relatively.
C) Numpy's function mean() was used.
D) Numpy's function mode()
E) Numpy's function var() was used.

What the code is doing:

Mean: $A = \frac{1}{n}\sum_{i=1}^{n} a_i = \frac{a_1 + a_2 + \cdots + a_n}{n}$

$Median:$

$Q1 : (N+1) * 1/4$

$Q2 : (N+1) * 2/4$

$Q3 : (N+1) * 3/4$

$Mode : Mostrepeatedvalue$

$$Variance : A = \sigma^2 = \frac{\sum_{i=1}^{n}(x_i - \mu)^2}{n}$$

1.3 Code (Not including print statements)

```python
import numpy
import pandas
from scipy import stats

# Reading Data
data_table = pandas.read_table('data.online.scores.txt', names = ['ID#', 'Mid-Term', 'Final'])
finals = data_table['Final']
mid = data_table['Mid-Term']
# https://numpy.org/doc/stable/reference/generated/numpy.array.html
mid_term_score_arr = numpy.array(mid)
final_score_arr = numpy.array(finals)


# Part A
# Mid-Term Data
min_mid = numpy.min(mid_term_score_arr)
max_mid = numpy.max(mid_term_score_arr)
#Final Data
min_val_final = numpy.min(final_score_arr)
max_val_final = numpy.max(final_score_arr)


# Part B
# https://numpy.org/doc/stable/reference/generated/numpy.around.html
#numpy.median(final_score_arr)
mq1 = numpy.percentile(mid_term_score_arr,25) # Q1
mq2 = numpy.percentile(mid_term_score_arr,50) # median
mq3 = numpy.percentile(mid_term_score_arr,75) # Q3
q1 = numpy.percentile(final_score_arr,25) # Q1
q2 = numpy.percentile(final_score_arr,50) # median
q3 = numpy.percentile(final_score_arr,75) # Q3

# Part C
finals_mean = numpy.mean(final_score_arr)
# https://numpy.org/doc/stable/reference/generated/numpy.around.html
finals_mean_rounded = numpy.around(finals_mean, decimals = 3)
mid_mean = numpy.mean(mid_term_score_arr)
mid_mean_rounded = numpy.around(mid_mean, decimals =3)

# Part D
# https://stackoverflow.com/questions/16330831/most-efficient-way-to-find-mode-in-numpy-array
mid_term_mode = mid.mode()
final_mode = finals.mode()

# Part E
# https://www.geeksforgeeks.org/numpy-var-in-python/
mid_var = numpy.var(mid_term_score_arr, dtype = numpy.float32)
var_final = numpy.var(final_score_arr, dtype = numpy.float32)
```

1.4 Terminal Output

```
A) Maximum and minimum.
==========================================================
Mid-Term 37 , 100
Final:  35 , 100

B) First quartile Q1, median, and third quartile Q3
==========================================================
Mid-Term: 68.0 , 77.0 , 87.0
Final:  82.0 , 89.0 , 96.0

C) Mean
==========================================================
Mid-Term:  76.715
Final:  87.084

D) Mode
==========================================================
Mid-Term:  77 83
Final:  97

E) Variance
==========================================================
Mid-Term:  173.10577
Final:  119.11294
```

Question 2.

2.1 Answers

A) Compute and compare the variance of midterm-original and midterm-normalized, i.e., the midterm scores before and after normalization.

Variance MidTerm: 173.10577
Normalized Variance: 1.0

B) Given an original midterm score of 90, what is the corresponding score after normalization?

Mean: 76.715
Std-Dev: 13.156966785699508
Using v' = v - Avg / std-dev
Our score of 90 is normalized to 1.009731210573523

C) Compute the Pearson's correlation coefficient between midterm-original and finals-original?

Pearson's correlation coefficient: 0.544424742312412

D) Compute the Pearson's correlation coefficient between midterm-normalized and finals-original.

Pearson's correlation coefficient: 0.544424742312412

E) Compute the covariance between midterm-original and finals-original.

Covariance: 78.25419419

2.2 Explanation and Equations

A) Numpy var() was used. The function performed the following equation:

$$Variance : A = \sigma^2 = \frac{\sum_{i=1}^{n}(x_i - \mu)^2}{n}$$

Using the standard dev and mean. I used the same equations as I did in question 1 to find those.

B) Scipy zscore() was used
then using v' = v - Avg / std-dev, we normalize the score

C) Scipy pearsonr() was used

D) Scipy pearsonr() was used
For both C and D the pearsonr() function utilized the following equation.

Formula

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

r = correlation coefficient
x_i = values of the x-variable in a sample
$\bar{x}$ = mean of the values of the x-variable
y_i = values of the y-variable in a sample
$\bar{y}$ = mean of the values of the y-variable

E) Numpy's stack() was used

This allowed me to find covariance. I could have use the cov() function in Panda, but this worked just as fine. It utilized this equation.

$$cov_{x,y} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{N-1}$$

$cov_{x,y}$ = covariance between variable a and y
x_i = data value of x
y_i = data value of y
$\bar{x}$ = mean of x
$\bar{y}$ = mean of y
N = number of data values

2.3 Code (Not including print statements)

```python
import numpy
import pandas
from scipy import stats

# Reading Data
data_table = pandas.read_table('data.online.scores.txt', names = ['ID#', 'Mid-Term', 'Final'])
finals = data_table['Final']
mid = data_table['Mid-Term']
# https://numpy.org/doc/stable/reference/generated/numpy.array.html
mid_term_score_arr = numpy.array(mid)
final_score_arr = numpy.array(finals)

# Part A
var_final = numpy.var(final_score_arr, dtype = numpy.float32)
var_mid = numpy.var(mid_term_score_arr, dtype = numpy.float32)

mid_norm = stats.zscore(mid_term_score_arr)
final_norm = stats.zscore(final_score_arr)
var_mid_norm = numpy.var(mid_norm, dtype = numpy.float32)
var_final_norm = numpy.var(final_norm, dtype = numpy.float32)

# Part B
given_score = 90
mid_std = numpy.std(mid_term_score_arr)
mid_mean = numpy.mean(mid_term_score_arr)
b_norm = (given_score - mid_mean) / mid_std

# Part C and D
corr_o_mid_o_fin = stats.pearsonr(mid_term_score_arr, final_score_arr)

# Part E
stack = numpy.stack((mid_term_score_arr, final_score_arr), axis = 0)
my_cov = numpy.cov(stack)
```

2.4 Terminal Output

```
A) Compute and compare the variance of midterm-original andmidterm-normalized, i.e., the midterm scores before and after nor
malization.
=====================================================================
Variance MidTerml:  173.10577
Normalized Variance:  1.0


B) Given an original midterm score of 90, what is the correspondingscore after normalization?
=====================================================================
Mean:  76.715
Std-Dev:  13.156966785699508
Using v' = v - Avg / std-dev
Our score of 90 is normalized to  1.009731210573523


C) Compute the Pearson's correlation coefficient between midterm-originaland finals-original.?
=====================================================================
Pearson's correlation coefficient:  (0.544424742312412, 3.0298169609726267e-78)


D) Compute the Pearson's correlation coefficient between midterm-normalizedand finals-original.
=====================================================================
Pearson's correlation coefficient:  (0.544424742312412, 3.0298169609726267e-78)


E) Compute the covariance between midterm-original and finals-original.
=====================================================================
Covariance:  [[173.27905405  78.25419419]
 [ 78.25419419 119.23217618]]
```

Question 3.

3.1 Answers

a) Each library has multiple copies of each book. Based on all the books (treat the counts of the 100 books as a feature vector for each of the libraries), compute the Minkowski distance of the vectors for CML and CBL with regard to different h values:

i) 6152.0
ii) 715.3278968417211
iii) 170.0

b) Compute the cosine similarity between the feature vectors for CML and CBL.

Cosine similarity: 0.8414040256623079

c) Compute the Kullback-Leibler (KL) divergence between CML and CBL

KL(CML || CBL): 0.149377 -> 14.94%
KL(CBL || CML): 0.16565 -> 16.57%

3.2 Explanation and Equations

a) Using scipy library and used minkowski() function.

$$(\sum_{i=1}^{n} |X_i - Y_i|^p)^{1/p}$$

The case where $p = 1$ is equivalent to the Manhattan distance and the case where $p = 2$ is equivalent to the Euclidean distance.

Based on equation above, this would have been calculated without any specific function. Rather by manual calculation. We could have found difference of CBL and CML data, taken their absolute value and summed these differences together. The p * 1/p cancels out. And thus, we are left with the original equation showed above. The function had the following parameters, (vector 1, vector 2, h value). Thus, for parts i, ii, and iii I simply changed the h value that was given. For part iii, I used float('inf')

b) For this part, I again, used scipy, however, this time I used the cosine distance function.

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

I could have simply used this equation, but using the predefined function allowed for a much cleaner code base.

c) KL divergence could have been found using SK-Learns lib, however, I used entropy in numpy. When scipy.stats.entropy is used, it will normalize the probabilities to one. Scipy's entropy function will calculate KL divergence if feed two vectors p and q, each representing a probability distribution. If the two vectors are not pdfs, it will normalize them first.

$$D(p||q) = \sum_{x \in X} p(x) \log \frac{p(x)}{q(x)}.$$

Using the equation shown above, we could 'manually' calculate this.

3.3 Code (Not including print statements)

```python
import numpy
import pandas
from scipy import stats
from scipy.spatial import distance
from scipy.stats import entropy

# Reading Data
# I was changing text file before, I do not think I should do that
data_table = "data.libraries.inventories.txt"
parse_data = pandas.read_csv(data_table, "\t")
cml = numpy.array(parse_data.loc[parse_data['library'] == 'CML'])[0]
cbl = numpy.array(parse_data.loc[parse_data['library'] == 'CBL'])[0]
#get rid of the book stuff
cml = numpy.delete(cml, 0)
cbl = numpy.delete(cbl, 0)
#Part A
q3p1 = distance.minkowski(cml, cbl, 1)
q3p2 = distance.minkowski(cml, cbl, 2)
q3p3 = distance.minkowski(cml, cbl, float('inf'))

#Part B
q3p4 = 1 - distance.cosine(list(cml), list(cbl))

#Part C
kl1 = entropy(list(cml), list(cbl))
kl2 = entropy(list(cbl), list(cml))
kl1perc = kl1 * 100
kl2perc = kl2 * 100
```

3.4 Terminal Output

```
a) Each library has multiple copies of each book. Based on all the books (treat
the counts of the 100 books as a feature vector for each of the libraries), compute the
Minkowski distance of the vectors for CML and CBL with regard to different h values:
==========================================================================================
i)      6152
ii)     715.3278968417211
iii)    170


b) Compute the cosine similarity between the feature vectors for CML and CBL.
==========================================================================================
Cosine similarity:  0.8414040256623079


c) Compute the Kullback-Leibler (KL) divergence between CML and CBL
==========================================================================================
KL(CML || CBL):  0.20708093733159472 in '%' it will be  20.7081 %
KL(CBL || CML:  0.22964703120204427 in '%' it will be  22.96 %
```

Question 4.

4.1 Answers

a) Calculate the distance between the binary attributes Buy Beer and Buy Diaper by assuming they are symmetric binary variables.

Distance: 0.015691868758915834

b) Calculate the distance between the binary attributes Buy Beer and Buy Diaper by assuming they are symmetric binary variables.

Jaccard Coefficient : 0.7317073170731707

c) Compute the χ2 statistic for the contingency table.

χ2: 2450.716326822006

d) Consider a hypothesis test based on the χ2 statistic where the null hypothesis is that Buy Beer and Buy Diaper are independent? Can you reject the null hypothesis? ***at a significance level of α = 0.05? Explain your answer, and mention the degrees of freedom used for the hypothesis test.***

If the value was more than 0.05 we could not reject the null hypothesis
The value based on info given was: 0.0
We were able to reject
P value: 0.0
Deg of freedom: 1

4.2 Explanation and Equations

A) This was done without using any library. First, I added the items then I divided by the sum of the entire array.
B) The Jaccard Coefficient between buying beer and diaper was done with this equation.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

Which translates to

The Jaccard similarity coefficient, J, is given as

$$J = \frac{M_{11}}{M_{01} + M_{10} + M_{11}}.$$

C) Done with the use of scipy chi2_contigency()
D) Done with the use of scipy chi2_contigency()

4.3 Code (Not including print statements)

```python
import numpy
from scipy import stats


table_q4 = numpy.array([[150,40],[15,3300]])
# Part A
sum = numpy.sum(table_q4)
beer = table_q4[0][1]
diaper = table_q4 [1][0]
dis = (beer + diaper)/sum;

# Part B
set1 = table_q4[0][0]
set2 = table_q4[0][1]
set3 = table_q4[1][0]
jac = set1 / (set1+set2+set3)
# Part C
chi =  stats.chi2_contingency(table_q4)[0]
# Part D
chi_2 =  stats.chi2_contingency(table_q4)[1]
freedom = stats.chi2_contingency(table_q4)[2]
```

4.4 Terminal Output

```
a) Calculate the distance between the binary attributes Buy Beer and Buy Diaper
by assuming they are symmetric binary variables.
==========================================================================================
Distance:  0.015691868758915834


b) Calculate the distance between the binary attributes Buy Beer and Buy Diaper
by assuming they are symmetric binary variables.
==========================================================================================
Jaccard Coefficient :  0.7317073170731707


c) Compute the χ2 statistic for the contingency table.
==========================================================================================
χ2:  2450.716326822006


d) Consider a hypothesis test based on the χ2 statistic where the null hypothesis
is that Buy Beer and Buy Diaper are independent. Can you reject the null hypothesis
at a significance level of α = 0.05? Explain your answer, and also mention the degrees
of freedom used for the hypothesis test.
==========================================================================================
If the value was more than 0.05 we could not reject the null hypothesis
The value based on info given was:  0.0
We were able to reject
P value:  0.0
Deg of freedom:  1
```