**Assignment 1**

By Harsh Sharma

**Question** 1. Classify the attribute type of given attributes with each of four category below:

Category 1: *categorical, numeric*

Category 2: *discrete, continues*

Category 3: *qualitative, quantitative*

Category 4: *nominal, binary, ordinal, interval, or ratio*

**Example:** Age in years. *Answer***:** numeric**,** discrete, quantitative, ratio

Some cases may have more than one interpretation in a category, so briefly indicate your reasoning if you think there may be some ambiguity.

(1) Height above sea level

(2) Bronze, Silver, and Gold medals as awarded at the Olympics.

**Solution:**

1. Height above sea level: Numeric, continuous, quantitative, ratio

**Reasoning**: Height above sea level is a quantifiable, continuous numerical characteristic that can be measured. It adheres to a ratio scale due to its meaningful zero point (sea level) and enables significant ratio comparisons.

1. Bronze, Silver, and Gold medals as awarded at the Olympics: Categorical, discrete, qualitative, nominal

**Reasoning**: Bronze, Silver, and Gold medals represent distinct categories or labels given as awards. They are discrete values and don't have a continuous or quantitative interpretation. The attribute is qualitative as it represents the quality or type of the medal. Within the categorical category, it can be classified as a nominal attribute since there is no inherent order or ranking among the medal categories.

**Question 2.** The (score) values for a subject (*subject1*) tuples are in increasing order.

13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70

(1) Normalize **a value 20** based on ***min-max normalization.***

(2) Normalize **a value 20** based on ***z-score normalization***.

(3) Explain why often normalize attribute values before analysis.

**Solution**

1. To find the normalize value of 20 using min-max normalization we can use this formula,

**normalized\_value = (value - min\_value) / (max\_value - min\_value)**

in the above tuple, the min value is 13 and max value is 70 and so the normalized value for 20 is,

**normalized\_value = (20 - 13) / (70 - 13) = 0.12280**

1. To find the nomarlised v alue of 20 using z-score normalization we need mean and standard deviation of the dataset. And so,

Mean = (13 + 15 + 16 + 16 + 19 + 20 + 20 + 21 + 22 + 22 + 25 + 25 + 25 + 25 + 30 + 33 + 33 + 35 + 35 + 35 + 35 + 36 + 40 + 45 + 46 + 52 + 70) / 27

= 29.96

Standard deviation = square root ([(13 - 29.96) ^2 + (15 - 29.96) ^2 + ... + (70 - 29.96) ^2] / 27)

= 13.475

Therefore, the normalized value is,

normalized\_value = (value - mean) / standard\_deviation

= (20-29.96)/13.475

= -0.743

1. **Attribute normalization is often performed before analysis for several reasons:**
2. Range standardization: Different attributes can have widely varying scales and ranges. By normalizing the values, we can bring them to a standardized range, typically between 0 and 1 or -1 and 1. This ensures that no single attribute dominates the analysis or computations solely based on its larger scale.
3. Comparison across variables: When conducting analyses that involve multiple variables, normalizing the attribute values allows for fair comparisons. It prevents certain variables with larger values from disproportionately influencing the results. Normalization puts the variables on an equal footing and enables meaningful comparisons.

**Question 3.** Answer the following questions.

The (score) values for a subject (*subject2*) tuples are in increasing order.

11, 12, 13, 15, 17, 20, 20, 21, 21, 22, 22, 23, 23, 25, 30, 31, 31, 32, 35, 35, 35, 36, 40, 45, 45, 53, 55

(1) Partition them into three bins by ***equal-width partitioning***. List values in each bin.

(2) Use **s*moothing by bin means*** to smooth the data from Q3 (1) result. List values in each bin.

**Solution**

1. To divide the dataset into three bins we need to find the width of each width. And so,

Bin Width = Range / Number of Bins = 44 / 3 ≈ 14.67

The bins would look like this:

Bin 1: [11, 25]

Bin 2: (25, 39]

Bin 3: (39, 55]

The values in each bin are as follows:

Bin 1: 11, 12, 13, 15, 17, 20, 20, 21, 21, 22, 22, 23, 23, 25

Bin 2: 30, 31, 31, 32, 35, 35, 35, 36

Bin 3: 40, 45, 45, 53, 55

1. To smoothen the bins, we need to find the mean of each bin.

Bin 1 Mean = (11 + 12 + 13 + 15 + 17 + 20 + 20 + 21 + 21 + 22 + 22 + 23 + 23 + 25) / 14 = 18.9

Bin 2 Mean = (30 + 31 + 31 + 32 + 35 + 35 + 35 + 36) / 8 = 33.125

Bin 3 Mean = (40 + 45 + 45 + 53 + 55) / 5 = 47.6

And then we need to replace the values in each with their respective mean values. The values in each bin after smoothing is,

Bin 1 = {18.9, 18.9, 18.9, 18.9, 18.9, 18.9, 18.9, 18.9, 18.9, 18.9, 18.9, 18.9, 18.9, 18.9}

Bin 2 = {33.12, 33.12, 33.12, 33.12, 33.12, 33.12, 33.12, 33.12}

Bin 3 = {47.6, 47.6, 47.6, 47.6, 47.6}

**Question 4.** Consider the time-series (−3,−1, 1, 3, 5, 7, ∗). Here, a missing entry is denoted by ∗. What would be the estimated value of the missing entry using ***linear interpolation,*** *y* = *a · x* + *b,* on a window of size 3 ?

[*Hint*] When you derive the relationship *y* = *a · x* + *b*, *x* is the time-stamp. So, you can present the dataset with (x, y)s, where y is a value measured at a time point x (any ordered value x), e.g., (1, -3), (2, -1) , (3, 1) , (4, 3), (5, 5) , (6, 7), (7,\*).

In *y* = *a · x* + *b* , the values *a* and *b* are typically solved using methods such as least squares regression, 𝑎=𝑁Σ𝑥𝑦−Σ𝑥Σ𝑦𝑁Σ𝑥2−(Σ𝑥)2, 𝑏=Σ𝑦−𝑎Σ𝑥𝑁.

For linear interpolation of ∗ on a window of size 3, you need to consider its previous three values, i.e., 3, 5, and 7.

**Solution:**

Using the equation y = a · x + b, we can determine the values of a and b by solving the least squares regression equations:

a = (N \* Σxy - Σx \* Σy) / (N \* Σx^2 - (Σx)^2)

b = (Σy - a \* Σx) / N

N = 3 (number of data points in the window)

Σxy = (1 \* 3) + (2 \* 5) + (3 \* 7) = 3 + 10 + 21 = 34

Σx = 1 + 2 + 3 = 6

Σy = 3 + 5 + 7 = 15

Σx^2 = (1^2) + (2^2) + (3^2) = 1 + 4 + 9 = 14

a = (3 \* 34 - 6 \* 15) / (3 \* 14 - 6^2)

a = (102 - 90) / (42 - 36)

a = 12 / 6

a = 2

b = (15 - 2 \* 6) / 3

b = (15 - 12) / 3

b = 3 / 3

b = 1

Now that we have the values of a and b, we can estimate the missing entry (∗) by substituting x = 7 into the linear equation:

Now, y = 2 \* 4 + 1

y = 8 + 1

y = 9

Therefore, the estimated value of the missing entry (∗) using linear interpolation on a window of size 3 is 9.

**Question 5.** Consider the following datasets with 2 attributes A1 and A2 :

|  |  |  |
| --- | --- | --- |
|  | **A1** | **A2** |
| **X1** | **1.5** | **1.7** |
| **X2** | **2** | **1.9** |
| **X3** | **1.6** | **1.8** |
| **X4** | **1.2** | **1.5** |
| **X5** | **1.5** | **1.0** |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | | |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |

**Solution.**

1. **Euclidean Distance:**

To calculate the Euclidean distance, we use the formula:

Distance = sqrt((x2 - x1)^2 + (y2 - y1)^2)

For the given dataset, the Euclidean distances from the query point (1.4, 1.6) to each data point are as follows:

For X1:

Distance = sqrt ((1.5 - 1.4) ^2 + (1.7 - 1.6) ^2) = 0.1414

For X2:

Distance = sqrt ((2 - 1.4) ^2 + (1.9 - 1.6) ^2) = 0.6708

For X3:

Distance = sqrt ((1.6 - 1.4) ^2 + (1.8 - 1.6) ^2) = 0.2828

For X4:

Distance = sqrt ((1.2 - 1.4) ^2 + (1.5 - 1.6) ^2) = 0.2236

For X5:

Distance = sqrt ((1.5 - 1.4) ^2 + (1.0 - 1.6) ^2) = 0.6083

Ranking the data points based on Euclidean distance in ascending order:

X1: 0.1414

X4: 0.2236

X3: 0.2828

X5: 0.6083

X2: 0.6708

1. **Manhattan Distance:**

To calculate the Manhattan distance, we use the formula:

Distance = |x2 - x1| + |y2 - y1|

For the given dataset, the Manhattan distances from the query point (1.4, 1.6) to each data point are as follows:

Manhattan distance = |x2 - x1| + |y2 - y1|

For X1:

Manhattan distance = |1.4 - 1.5| + |1.6 - 1.7| = 0.1 + 0.1 = 0.2

For X2:

Manhattan distance = |1.4 - 2| + |1.6 - 1.9| = 0.6 + 0.3 = 0.9

For X3:

Manhattan distance = |1.4 - 1.6| + |1.6 - 1.8| = 0.2 + 0.2 = 0.4

For X4:

Manhattan distance = |1.4 - 1.2| + |1.6 - 1.5| = 0.2 + 0.1 = 0.3

For X5:

Manhattan distance = |1.4 - 1.5| + |1.6 - 1| = 0.1 + 0.6 = 0.7

Ranking the data points (in ascending order):

X1: Manhattan distance = 0.2

X4: Manhattan distance = 0.3

X3: Manhattan distance = 0.4

X5: Manhattan distance = 0.7

X2: Manhattan distance = 0.9

Therefore, based on Euclidean distance, the ranking from most similar to least similar to the query (1.4, 1.6) would be: X1, X4, X5, X3, X2. Based on Manhattan distance, the ranking from most similar to least similar to the query (1.4, 1.6) would be: X1 < X4 < X3 < X5 < X2

**Question 6.** Compute the (1) ***match-based similarity*** and (2) ***Jaccard coefficient*** between the two sets {A, B, C} and {A, C, D, E}.

**Solution**.

1) Match-based Similarity:

The two sets {A, B, C} and {A, C, D, E} have two common elements (A and C). The total number of unique elements across both sets is 5 (A, B, C, D, E). Therefore, the match-based similarity is:

Match-based Similarity = Number of common elements / Total number of unique elements

= 2 / 5

= 0.4

2) Jaccard Coefficient:

The two sets {A, B, C} and {A, C, D, E} have an intersection of {A, C} (2 elements) and a union of {A, B, C, D, E} (5 elements). Therefore, the Jaccard coefficient is:

Jaccard Coefficient = Size of intersection / Size of union

= 2 / 5

= 0.4

**Question 7.** Consider the two sentences below.

S1: “The sly fox jumped over the lazy dog.”

S2: “The dog jumped at the intruder.”

(1) Convert S1 and S2 to ***frequency term vectors***.

Assume the lexicon here is {the sly fox jumped over lazy dog at intruder}.

(2) Compute the ***cosine similarity*** measure of S1 and S2:

**Solution.**

1. Conversion to Frequency Term Vectors:

Using the lexicon {the, sly, fox, jumped, over, lazy, dog, at, intruder}, we can represent S1 and S2 as frequency term vectors.

S1: “The sly fox jumped over the lazy dog.”

Frequency = [2, 1, 1, 1, 1, 1, 1, 0, 0]

S2: “The dog jumped at the intruder.”

Frequency = [2, 0, 0, 1, 0, 0, 1, 1, 1]

Computing Cosine Similarity:

To compute the cosine similarity between S1 and S2, we can use the formula:

Cosine Similarity = (S1 · S2) / (||S1|| \* ||S2||)

Using the frequency term vectors from above, we have:

S1 · S2 = (11) + (10) + (10) + (11) + (10) + (10) + (11) + (01) + (0\*1) = 3

||S1|| = sqrt(1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 0^2 + 0^2) = sqrt(7) ≈ 2.646

||S2|| = sqrt(1^2 + 0^2 + 0^2 + 1^2 + 0^2 + 0^2 + 1^2 + 1^2 + 1^2) = sqrt(5) ≈ 2.236

Cosine Similarity = (S1 · S2) / (||S1|| \* ||S2||) = 3 / (2.646 \* 2.236) ≈ 0.674

Therefore, the cosine similarity between S1 and S2 is approximately 0.674.

**Question 8**: The columns of the matrix correspond to the sets, and the rows correspond to elements of the universal set from which elements of the sets are drawn. There is a 1 in row *r* and column *c* if the element for row *r* is a member of the set for column *c*, otherwise 0.

(1) Compute the ***minhash signature*** for each column when we use a hash function: *h*(*x*) = 2*x*+1 mod 6. Show the final **signature matrix**.

(2) Estimate the ***Jaccard similarities*** of the underlying sets ***S*2** and ***S*4** from the signature matrix from (1).

**Solution.**

Minhash Signature Matrix: Using the hash function h(x) = (2x + 1) mod 6, we apply it to each element in the columns and select the minimum value as the signature.

Signature Matrix:

For S1:

h(0) = 1, h(0) = 1, h(1) = 3, h(0) = 1, h(0) = 1, h(1) = 3

Minimum hash value = 1

For S2:

h(1) = 3, h(1) = 3, h(0) = 1, h(0) = 1, h(0) = 1, h(0) = 1

Minimum hash value = 1

For S3:

h(0) = 1, h(0) = 1, h(0) = 1, h(1) = 3, h(1) = 3, h(0) = 1

Minimum hash value = 1

For S4:

h(1) = 3, h(0) = 1, h(1) = 3, h(0) = 1, h(1) = 3, h(0) = 1

Minimum hash value = 1

final signature matrix is

|  |  |  |  |
| --- | --- | --- | --- |
| S1 | S2 | S3 | S4 |
| 1 | 1 | 1 | 1 |

b) Estimate the Jaccard similarities of the underlying sets S2 and S4 from the signature matrix from (a):

Jaccard similarities = 1/4

So, jaccard similarities for S2 S4 is 0.25.

**PART II**

**1. The purpose of this practice is to get familiar with R scripts used in data exploration. Follow “Ch 3. Data Exploration” (from Zhao, “R and Data Mining”) provided. Submit your practice on all scripts from Ch 3.1 to Ch 3.5 with showing the results.**

**Alternatively, you can show the same results using Python or any programming language you choose.**

**Deliverables: (1) Your program codes, e.g., the script codes in Ch 3.1 – Ch 3.5 provided.  
(2) A proof to show the successful execution of your program, e.g., screen shots on running**

**and the program output.**

**Solutions.**

Here I’m sharing the output of my console along with the graph screenshots: (Also, I’ll be sharing the code)

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Plot ofplot(density(iris$Sepal.Length)):

A graph of a function

Description automatically generated

Plot ofhist(iris$Sepal.Length):

A graph of a bar graph

Description automatically generated

A computer screen shot of a black screen

Description automatically generated

Plot of pie(table(iris$Species)):

A pie chart with different colors

Description automatically generated

Plot of barplot(table(iris$Species)):

A group of rectangular objects

Description automatically generated

A computer screen shot of a program

Description automatically generated

Plot ofboxplot(Sepal.Length~Species, data=iris):

**A picture containing diagram, text, technical drawing, rectangle

Description automatically generated**

Plot of with(iris, plot(Sepal.Length, Sepal.Width, col=Species, pch=as.numeric(Species)))

A picture containing text, diagram, line, plot

Description automatically generated

Plot of plot(jitter(iris$Sepal.Length), jitter(iris$Sepal.Width)):

A graph with dots and numbers

Description automatically generated

plot of pairs(iris):

A picture containing text, diagram

Description automatically generated

Plot of scatterplot3d(iris$Petal.Width, iris$Sepal.Length, iris$Sepal.Width):

A picture containing diagram, line, pattern

Description automatically generated

A computer screen shot of a program

Description automatically generated

Plot of heatmap(distMatrix):

A picture containing diagram, rectangle, square, plan

Description automatically generated

Plot of levelplot(Petal.Width ~ Sepal.Length \* Sepal.Width, iris, cuts = 9,

col.regions = grey.colors(10)[10:1]):

A graph of a number of gray squares

Description automatically generated

Plot of filled.contour(volcano, color=terrain.colors, asp=1, plot.axes=contour(volcano, add=T)):

A picture containing screenshot, colorfulness, diagram, graphics

Description automatically generated

Plot of persp(volcano, theta=25, phi=30, expand=0.5, col="lightblue"):

A picture containing sketch, design, art

Description automatically generated

Plot of parcoord(iris[1:4], col=iris$Species):

A picture containing sketch, drawing, art, design

Description automatically generated

Plot of parallelplot(~iris[1:4] | Species, data=iris):

A picture containing diagram, line, plot, plan

Description automatically generated

Plot for qplot(Sepal.Length, Sepal.Width, data=iris, facets=Species ~.):

A picture containing text, screenshot, number, diagram

Description automatically generated

**To save the graph as a PDF.**

A picture containing text, screenshot, font

Description automatically generated

**2. Download the Arrythmia data set from https://archive.ics.uci.edu/ml/datasets/Arrhythmia. Using a programming language or data mining tool you chose, normalize all the numeric attributes in the dataset with a mean of 0 and a standard deviation of 1.**

**Deliverables: (1) A list of first 10 records transformed with normalized values, i.e., z-scores. (2) Your program codes, if any**

**(3) A proof to show the program’s successful execution,**

**Code and execution are below:** Code is successfully executed, and first 10 records are shown.

Code file is also attached in the zip file with label as zscore.py

**Output:**

A screenshot of a computer

Description automatically generated with medium confidence

**3. Again download the Arrythmia data set. Discretize each numerical attribute into 10 equiwidth ranges (bins), and then transform each attribute’s value to the corresponding bin name. Assume the bin names of each attribute are given as *attribute name*\_Bin1, *attribute name*\_Bin2, ..., *attribute name\_*Bin10.**

**Deliverables: (1) A list of first 10 records transformed using the discretization. (2) Your program codes, if any**

**(3) A proof to show the program’s successful execution,**

**Code and execution are below:** Code is successfully executed, and first 10 records are shown.

Code file is also attached in the zip file with label as discretize.py

**Output:**

**A screenshot of a computer

Description automatically generated with medium confidence**