**Similarites**

**Proxamity Transformation = s(comma) where s = similarities. S(\*\*comma) = (s - min\_s (min value of s)) / (max\_s – min\_s)**

*For example: data = [10,30]*

*if(real\_similarity == 14) #-----------🡪 where the 14 gets calculated from string to int.*

*value = #s(comma) where s = similarities.*

*S(\*\*comma) = (s - min\_s (min value of s)) / (max\_s – min\_s)*

*return value*

*## (14/ 30) / (30-10)*

*## Should be 0.2*

***Exercise***

*S(comma) = (s-1)/9 where its like computers, the left side is the new variable, and the right side is the original*

**Exercise: if the original similarity between objects is 6.4 on the previously mention scale, what is the similarity when transformed to the range [0,1]?**

**(6.4-1)/9 = 0.6**

**Dissimilarities**

If the proximity measure originally takes values in the interval [0, ∞), one transformation of proximity measure to [0,1] is

D(comma) = d / (1 + d)

The formula is as follows:

**Proxamity Transformation = d(comma) where d = disimilarities. d(\*\*comma) = (d - min\_d (min distance of s)) / (max\_d – min\_d)**

**D = distance**

*Example:*

*Data = [30.00, 40.00]*

*Dissimilarities = 12*

*##(13/30) / (40/30)*

*##0.433333/1.33333*

*##Dissimilarity should equal 0.3249983*

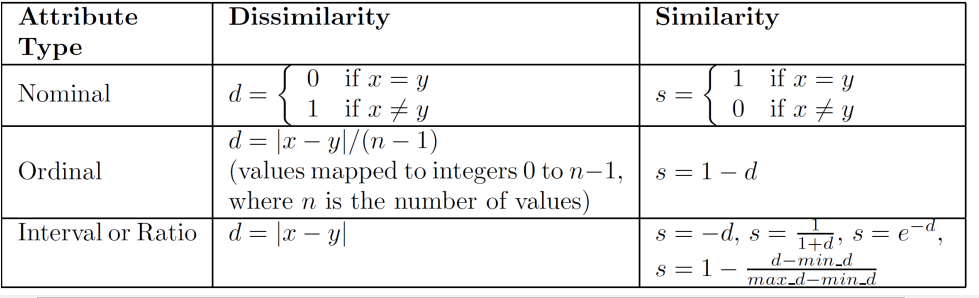
For nominal attributes we let one equal a unique count or modifier, and if not we put it as not equal to, for distinctiveness.

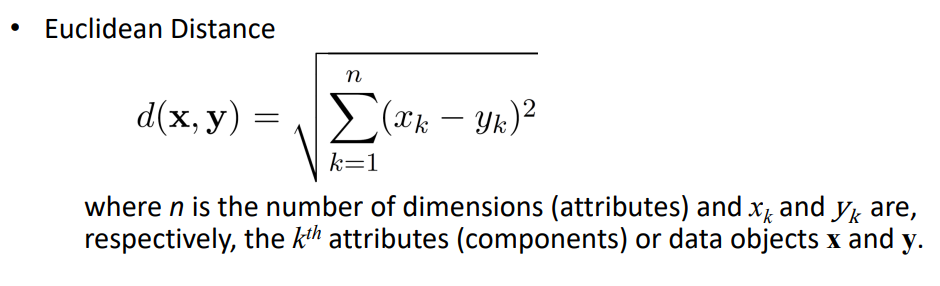
For ordinal attributes, we measure as follows…

|P1 – P2 |(n – 1)

For example.. We measure two different attribute labels that are labelled in returned int form, whereas **P1 = 4 (wonderful on the scale) and P2 = 3 (good in the scale).** We would want it to be (4-3)(number of attributes, in this case, (**5 – 1). This gives us the similarity of ordinal attributes.**

**Intervals and ratios are the absolute difference between P1 and P2, so 1**



**But we can see that similarities are the distance are important. This is where Euclidian Distance becomes important for measuring similarities in interval or ratio elements.** 

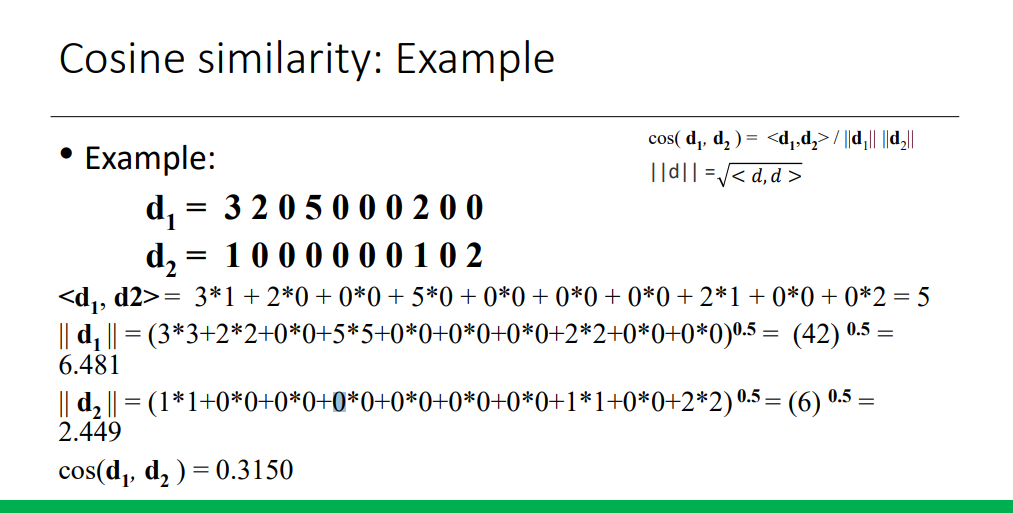
**Similarity in binary vectors (0s and 1s):**

**Simple Matching Coefficent = (f(00) + (f11)) / (f(11) + f(00) + f(10) + f(01))**

**Where number of matches / number of attributes**

**Jicard Matching Method = f(11) / (f(10) + f(11) + f(01))**

***Where number of 11s / number of non-zero attributes***

**Cosine Similarity =** 

**Remember that in cosine similar ||d1|| and ||d2|| must be multipled together. Follow the example**

**Discretization of continuous variables**

Sort all values of a single attribute

• Select how many categories that we want (n)

• Create n groups based on some criteria

• Each value of the original attribute becomes the group label

**Unsupervised discretization**

**Equal width= w = (max – min) / number of bins**

**Equal frequency = number of samples / number of bins you want, and increments by that equal frequency**