T0 – owned data

T1 – Shared Data

T\_int

T0 owned

|  |  |  |
| --- | --- | --- |
| Name | Age | Salary |
| xyz | 21 | 40k |

T1 shared

|  |  |  |
| --- | --- | --- |
| Name | Age | Salary |
| xyz1 | 23 | 45k |

T\_int

|  |  |  |
| --- | --- | --- |
| Name | Age | Salary |
| xyz1 | 21-23 | 40-45k |

Method 1: Ignore the T1 (Shared Dataset) as we cannot look at the Shared Dataset and Trust the integration process. So, our assumption is the T\_integrated dataset is true or valid.

So now we have Gain/Loss metric. This can be a ratio. All the same values / old values. For example, in the above tables we have new values like Name (xyz1 + 1), age (23 +1), salary (45k +1)

A = xyz

B = xyz1

We can calculate similarity measure (Jaccard Index) between names. Jaccard Index =

(A integration B) / (A union B).

A integration B = xyz = 3

A union B = xzy1 = 4

JI = ¾ = 0.75

We can take a critical value if JI > 0.70 so if this is true both names are same.

|  |  |
| --- | --- |
| Case when there is only 1 extra character.  A int B = xyz = 3  A U B = xyz1 = 4  JI = ¾ = 0.75 | Case when A = xyz & B = xy1  A int B = 2  A U B = 4  JI = 2/4 = 0.5 |
| Case of xyz & xyz12  A = xyz & B = xyz12  JI = 3/5 = 0.6 | Case of A = Shemon and B = Simon  A int B = 4  A U B = 7  JI = 4/7 = 0.571 |
| SRawat and Shemon Rawat | 4/6 = 2/3 = 0.67 |

Looking at the table above we can see when we have 1 extra character the similarity is 0.75

When we have 1 different character then JI = 0.5

If you have 2 extra characters, then JI = 0.6

If bigger name and 1 different character, we get JI = 0.7143

So we can set critical value to 0.70

If JI is > 0.70 then it’s a similar value and if its less than that we can use clipping and calculate the SED between two names.

Going back to method 1 we can say both names xyz and xyz1 are same. So now we have **3 same** values (xyz, 21, 40k) and we have **8 total values**. And age (23) and salary(45k) **2 different** values.

Same/different ration = 3/2 = 1.5

Different/same ratio = 2/3 = 0.67

Same/total = 3/8 = 0.375

Different/total = 2/6 = 1/3 = 0.333

What do they mean?

Method 2: ED can be calculated for all the integer values and SED can be calculated for String values. But SED and ED are not on the same metric. How to do that? Later we do it with log methods and use clipping on it.

What if we apply ED for int values and SED for int values and then apply the log scaling on both of them so if SED values = 2,4,8,3,5,… after taking log values it scales down. We get = 1,2,3,1.58, 2.32, (some small values.)

If ED values = 200k,400k,800k they would look like log2 200 = 7.64, log2 400 = 8.64, log2 800 = 9.67. The values scale down from 800 to 9.67.

These metrics are much more comparable than before.

Normalizing Method:

1. Min Max Normalizing:

X’ = (x – x(min))/(x(max) – x(min))

In 1. We need the UB and LB. Its good for comparing variables like age & salary.

1. Feature clipping: This method helps with outliers a lot. We can fix a max value and if a value is greater than the critical value then we can just set it to something.
2. Log scaling => x’ = log(x)

This is good when a dataset has very large and very small values. So to make the graph more linear we use log scaling.

1. Z-score values: means the number of SDs, a value is away from mean. Z-score =

(x-mean) /SD

1. Data standardizations

Revising Entropy.

xxxxx xxxxx if all the values are same i.e 10xs then entropy = 0.

If we get some o in that for example xxxxx oooxx then we increase entropy in that case.

Entropy = -sum(p \* log(p))

P = probability of that event happening.

Example : xxx ooo ----

N = 10

X = 3

O = 3

“-” = 4

Entropy = -3/10 log(3/10) – 3/10log(3/10) – 4/10log(4/10)

If we calculate entropy for all same (xxxx) then entropy = -1\*log(1) = 0

For information gain we split the dataset. We canulate entropy for both the splits we find the entropy for the split data. We find the weighted entropy. Then IG = 1-wt entropy.

Weighted entropy always lower than the entropy of the entire dataset.

Cell level IG? (how? Using Provenance ?)

This means if I look at one cell or the entire data point and use a formulae on both we will should get same IG.

Taking T1, T0 and T\_int from above we have 6 different values, 2 same values, 8 total values.

(xyz,21.40k),(xyz1,21-23,40-45k)

1. 2. 3. 4. 5. 6. 7. 8.

We can come up with a formula : sum (log (x-y)) = log 1 + log 2 + log (5000) = 0+1+12.287

Instead of using 5000 we can use 5 so log 5 = 2.32

Provenance:

R(owned)

|  |  |  |
| --- | --- | --- |
| A | B | Prov symbol |
| 1 | a | Pa Pb |
| 2 | b | Qa Qb |
| 1 | c | Ra Rb |

S(Shared)

|  |  |  |
| --- | --- | --- |
| B | C | Prov symbol |
| a | 10k | Xb Xc |
| c` | 10k | Yb Yc |
| b` | 20k | Zb Zc |

R join S

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C | Symbol |
| 1 | A | 10k | Pa(Pb+Xb)Xc |
| 2 | b+c` | 10k | Qa(Qb+Yb)Yz |
| 1 | c+b` | 20k | Ra(Rb+Zb)Zc |

PX -> Pa.Pb.Xc OR Pa.Xb.Xc

Provenance increases Security and speed

(Pb + Xb) are same in this case => here + means =

(Qb + Yb) are different => so we store the difference so b-c`

(Rb + Zb) are different => so we store the difference so c-b`

If these values like these (Qb + Yb) contain range values and we can directly store the differences in them and display only the symbols for | Qa | Qb + Yb |(b -c`) | Yz | then this will increase the security and the performance and we can use provenance using IG.

\*log of 1 is 0. So if the difference between two strings is only 1 character then they are basically same values as so we get log1 = 0. As log scales the graph down and tries to make it linear this might help us to compare SEDs and EDs.

We might need do a min max normalization together with log scaling in some order.

\*Other thing is what if it’s a completely new value so for SED it will be length of the string OR just 1. For EDs we will calculate the distance from 0.

\*we can use clipping for outliers , log scaling and min max together,

If we use log out IG will range from 0 – 100

As 2^0 = 1

And 2^10 = a very large number

Bloom Filter

2 methods :

1. Hash values as provenance annotation. Eg 1 🡪 0001 🡪 (R,A,1)
2. Apply SED to all and the add ED

GOAL : to scale down bigger values to make graph linear.

https://www.usenix.org/conferences/author-resources/paper-templates

D(a,1)

R(1,b)

(Alb)

(a,b)

D(a,1) -> p

R(1,b) -> q

res(a,b) -> pq

a -> p

1 -> q

1 -> r

b -> s

(a,b) -> p(q=r)s

D(a,1) -> pq

IG -> s

a -> p

1 -> q

b -> s

D(a,1)

R(1,b), R(1,c)

res(alb)

res(a,b), res(a,c)

a -> p

1 -> 1

1 -> q

b -> r

c -> s

res(a,b) -> p(q=q)r

res(a,b) -> p(q=q)r -> pqr

res(a,c) -> p(q=q)s -> pqs

D(a,1) -> pq

R(1,b,c), R(1,c,b)

res(a,b,c), res(a,c,b)

res(a,b,c) -> pqrs

res(a,b,c) -> pqrs -> p(q=q)rs

(d.B = r.B);

D(A,B), R(B,C,D)

JOIN R r ON (d.B = r.B);

res: result table

D(a,1), res(a,b,c)

D(a,1) -> pq

res(a,b,c) -> pqrs

res(a,c,b) -> pqsr

{rs, sr}

{}

A = {pq}

B = {pqrs, pqsr}

A = {}

B = {rs, sr}