COMPGED Makes Matches Easy To See!

Nefera Croom
NYASUG June 2009

Overview

- What is COMPGED?
- Why is it useful?
- How does it work?
- Customizing COMPGED
- Making COMPGED work for you
- Extra Tips

What is COMPGED?

- SAS function that **comp**utes the **G**eneralized **E**dit **D**istance between two character strings
- COMPGED calculates a number representing how much work it takes to make the second string exactly like the first
- The higher the computed GED, the less likely the two strings match
- Zero = perfect match

Why is it Useful?

- Can be used in conjunction with prints to narrow down good matches in a merged dataset
- Match-Merging on Unique Identifiers not always 100% reliable
- Unique Identifiers not always available to do a Match-Merge
- Allows fuzzy matching when used with PROC SQL

The Secret Life of COMPGED

How does it work?

GED is based on a metric called the Levenshtein distance. The algorithm uses an $(n+1)(m+1)$ matrix where n and m are lengths of two strings to calculate edit distance.

- For every change SAS has to make in order for string2 to be exactly like string1, SAS charges some points
- SAS always uses the most cost-efficient method to turn string2 into string1
- What's the cost of some common edits?

String1	String2	
Adam	Adama	truncate a letter, cost = 10 pts
Jaime	Jiame	swap letters, cost = 20 pts
Andres	Andre	append a letter, cost = 100 pts

Generalized edit distance Based on operation

String1	String2	Operation	GED
baboon	baboon	match	0
baXboon	baboon	insert	100
baoon	baboon	delete	100
baXoon	baboon	replace	100
baboonX	baboon	append	50
baboo	baboon	truncate	10
babboon	baboon	double	20
babon	baboon	single	20
baobon	baboon	swap	20
bab oon	baboon	blank	10
bab,oon	baboon	punctuation	30
bXaoon	baboon	insert+delete	200
bXaYoon	baboon	insert+replace	200
bXoon	baboon	delete+replace	200
Xbaboon	baboon	finsert	200
aboon	baboon	trick question: swap+delete	120
Xaboon	baboon	freplace	200
axoon	baboon	fdelete+replace	300
axoo	baboon	fdelete+replace+truncate	310
axon	baboon	fdelete+replace+single	320
baby	baboon	replace+truncate*2	120
balloon	baboon	replace+insert	200

Activity 1
[5 min]

- Based on the edit values given on the last slide, please calculate how much SAS would charge to turn **string2 into string1**.

	string1	string2	
1)	Nefera Croom	Nepheria Croome	310
2)	092093813	092998313	120
3)	326 E Keshire Blvd.	326 East Keshire Blvd	330
4)	Nepheria Croome	Nefera Croom	350

1) delete (100), replace (100), delete (100), truncate (10)
2) replace (100), swap (20)
3) delete 3x (300), punctuation (30)
4) replace (100), insert (100), insert (100), append (50)

Customizing COMPGED

Making COMPGED Symmetric

- CALL COMPCOST routine
- Set the following operations to equal values
  - INSERT, DELETE
  - FINSERT, FDELETE
  - APPEND, TRUNCATE
  - DOUBLE, SINGLE
- **Syntax:** CALL COMPCOST (insert=, ‘100’, delete=,’100’);

COMPGED CUTOFF

- A cutoff value sets up a budget for how many edits SAS can charge to make string2 into string1.

- Those comparisons with a GED greater than or equal to the cutoff value are assigned the cutoff value.

Syntax:

GEDvalue = COMPGED(string1, string2, cutoff);
= COMPGED(name1, name2, 500);

COMPGED Modifiers

- ‘I’ or ‘i’ ignores case
- ‘L’ or ‘l’ ignores leading blanks
- ‘N’ or ‘n’ ignores quotations and case

Syntax:

GEDvalue = COMPGED(string1, string2, cutoff, ‘modifiers’);

= COMPGED(name1, name2, 500, ‘li’);

Making COMPGED Work For You

Time to take the driver's seat!

Activity 2
[5 min]

- Scenario: You have merged the MDRC file with an outside file on Sample ID. You now want to confirm that you have true matches.
- In SAS Enterprise Guide, click the code node labeled "Activity 2." Examine the code and list. Fill in the COMPGED code to compare the two full names where indicated and re-run the program.
- How many records did your merge yield?
- How many true matches are there?

SAS CODE FOR ACTIVITY 2

```
proc sort data = MDRC out=MDRCs;  by SampleID;
proc sort data = Outside out=Outs; by SampleID2;
data MatchMerge;
merge   MDRCs ( in=md )
        Outs (rename=(SampleID2=SampleID) in=ot);
by SampleID;
inMD = md;
inOt = ot;
run;
```

SAS OUTPUT FOR ACTIVITY 2

Activity 2, Making COMPGED Work For You
PRINT of Merge on SampleID where DOBs Equal, Check to Confirm Merge

Name Comp	fullname	fullname2	dob	dob2	stnum	stnum2	Coverage
0	Kim Basinger	Kim Basinger	11/24/1975	11/24/1975	116	116	Full Medical
10	Angela Bassett	Angela Bassette	05/21/1964	05/21/1964	873	8735	Prenatal Care
200	Brandy Norwood	brandon norwood	01/31/1962	01/31/1962	88	88	Prenatal Care
230	Coretta Scott-King	Coretta Scott	09/18/1978	09/18/1978	504	504	Full Medical
630	B.B. King	rileyB king	05/16/1959	05/16/1959	5608	5608	Full Medical

PRINT of Merge on SampleID where DOBs Not Equal, Check to Confirm Merge

Name Comp	fullname	fullname2	dob	dob2	stnum	stnum2	Coverage
0	Russell Simmons	russell simmons	05/04/1972	06/30/1966	62985	745	Emergency Only
100	Thomas Edison	Tomas Edison	08/11/1964	11/08/1964	5567	5567	Full Medical
600	Reese Witherspoon	John Witherspoon	03/15/1972	08/01/1955	565	565	Emergency Only

Fuzzy Merging with Proc SQL and COMPGED

```
Proc SQL;
Create table
    AdultMerge as
Select *,
    CompGED(mdrc_name, outside_name,400,'LN')
     as NameComp
From
      MDRC, Outside
Where
      calculated NameComp lt 400 and (ssn1 = ssn2
      or stnum = stnum2 or dob = dob2
      or calculated NameComp lt 100)
Order by NameComp;
Quit;
```

Activity 3
[10 min]

- Scenario: You have merged an outside dataset with your cross reference file on Sample ID and are not satisfied with the low match rate. You want to merge based on COMPGED of name as well as some other variables.
- In SAS Enterprise Guide click on the code node labeled "Activity 3." Using the syntax from slide 12, fill in the missing code where indicated and run the program.
- Do a join based on COMPGED using PROC SQL including the variables that will help distinguish true matches.
- Remember: Sure Fire Way to Get Hot Output!

Activity 3 Cont'd

- How man records did the SQL join yield?
- How many were true matches?
- What could have been done differently to yield more or fewer matches?

SAS CODE FOR ACTIVITY 3

```
PROC SQL;
create table GEDSQL as
SELECT  * ,   compged(fullname,fullname2,'i')
              as NameComp2
FROM          mdrc, outside
WHERE         sampleID = sampleID2 or
              dob=dob2 or stnum=stnum2 or
              calculated namecomp2 lt 300
ORDER BY      calculated NameComp2;
quit;
```

SAS OUPUT FOR ACTIVITY 3

Activity 3, PROC SQL and COMPGED

PRINT of Join on COMPGED using Proc SQL Where SampleIDs Equal

Comp2	Name fullname	fullname2	sample ID	sample ID2	dob	dob2	stnum	stnum2
0	Kim Basinger	Kim Basinger	106563	106563	11/24/1975	11/24/1975	116	116
0	Russell Simmons	russell simmons	122585	122585	05/04/1972	06/30/1966	62985	745
10	Angela Bassett	Angela Bassette	056501	056501	05/21/1964	05/21/1964	873	8735
100	Thomas Edison	Tomas Edison	054587	054587	08/11/1964	11/08/1964	5567	5567
200	Brandy Norwood	brandon norwood	118701	118701	01/31/1962	01/31/1962	88	88
230	Coretta Scott-King	Coretta Scott	116486	116486	09/18/1978	09/18/1978	504	504
600	Reese Witherspoon	John Witherspoon	063925	063925	03/15/1972	08/01/1955	565	565
630	B.B. King	rileyB king	072801	072801	05/16/1959	05/16/1959	5608	5608

PRINT of Join on COMPGED using Proc SQL Where SampleIDs Not Equal

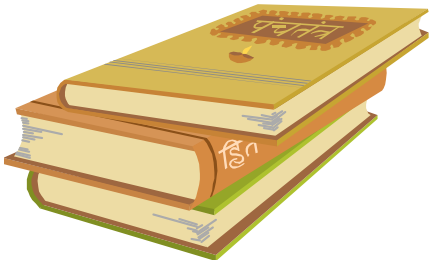
Comp2	Name fullname	fullname2	sample ID	sample ID2	dob	dob2	stnum	stnum2
40	Forest Whitaker	Forrest Whittaker	083826	083026	02/09/1969	02/09/1969	177	177
100	Cory Booker	Corey Booker	582831	582837	04/30/1954	04/30/1954	5313	5373
120	Bob Dylan	Bobby Dylan	583971	583977	11/16/1966	11/16/1966	936	936
200	Jack LaLanne	jackie Lalanne	582777	246851	04/18/1971	04/18/1971	517	1012
1100	Fiona Apple	Apple fiona	092605	092695	05/21/1977	05/21/1977	804	804

Extra Tips

- Use a WHERE statement to make the Cartesian product created with PROC SQL execute more efficiently
- Create a permanent dataset and comment out code to help program run more quickly
- Run a COMPGED once without a cutoff to get a sense of a good cutoff value

References

- SAS(R) 9.2 Language Reference Dictionary on COMPGED and CALL COMPCOST functions
- Staum, Paulette. 2007. “Fuzzy Matching Using the COMPGED function”. Proceedings from the Northeast SAS Users Group 2007.
- "Levenshtein distance." Wikipedia, the free encyclopedia. 18 Mar. 2009



THANK YOU!

- To all of you for interaction and feedback!
- SPECIAL THANK YOU to
- **Jared Smith** for teaching me what a great tool COMPGED is
- **Paulette Staum** for her help with Proc SQL and COMPGED
- **Christopher Bost** for supporting me in developing this presentation