

Data Transformation Using Stata for Survival Analysis

Jae-Woo Kim

1) Goal and Why?

- Stata seems to be picky in terms of the layout of data. For example, it cannot recognize the following type of data, unlike SAS, in survival analysis.

	id	event1	event2	event3	event4	per1	per2	per3	per4	busi
1	1	0	0	1	0	1	2	3	4	1
2	2	0	0	0	1	2	2	3	3	1
3	3	0	0	0	0	0	1	1	1	1
4	4	0	1	0	0	0	3	3	4	2
5	5	0	0	1	0	1	1	2	2	2

How to Transform Top to Down?

	id	event1	event2	event3	event4	per1	per2	per3	per4	busi
1	1	0	0	1	0	1	2	3	4	1
2	2	0	0	0	1	2	2	3	3	1
3	3	0	0	0	0	0	1	1	1	1
4	4	0	1	0	0	0	3	3	4	2
5	5	0	0	1	0	1	1	2	2	2

	id	_t0	_t	event	per	busi
1	1	0	1	0	1	1
2	1	1	2	0	2	1
3	1	2	3	1	3	1
4	2	0	1	0	2	1
5	2	1	2	0	2	1
6	2	2	3	0	3	1
7	2	3	4	1	3	1
8	3	0	1	0	0	1
9	3	1	2	0	1	1
10	3	2	3	0	1	1
11	3	3	4	0	1	1
12	4	0	1	0	0	2
13	4	1	2	1	3	2
14	5	0	1	0	1	2
15	5	1	2	0	1	2
16	5	2	3	1	2	2

Make a Basic Layout

	id	survt	event	
1	1	3	1	
2	2	4	1	
3	3	4	0	
4	4	2	1	
5	5	3	1	

- Id
- Survt: survival time. 4 given to the censored case (id=3)
- Event: 1 for failure; 0 otherwise

Generate starting two time-points

	id	survt	event	_st	_d	_t	_t0
1	1	3	1	1	1	3	0
2	2	4	1	1	1	4	0
3	3	4	0	1	0	4	0
4	4	2	1	1	1	2	0
5	5	3	1	1	1	3	0

- `stset survt, failure(event==1) id(id)`
- Without `id(id)` option, Stata assumes that each observation represents a different subject.
- `_t0` (observation starting); `_t` (time-point when the event occurred, or observation ending for censored cases)

Divide period by using stsplit

	id	survt	event	_st	_d	_t	_t0	d1
1	1	1	.	1	0	1	0	0
2	1	3	1	1	1	3	1	1
3	2	1	.	1	0	1	0	0
4	2	4	1	1	1	4	1	1
5	3	1	.	1	0	1	0	0
6	3	4	0	1	0	4	1	1
7	4	1	.	1	0	1	0	0
8	4	2	1	1	1	2	1	1
9	5	1	.	1	0	1	0	0
10	5	3	1	1	1	3	1	1

- Stsplit option is used when dealing with Heavyside functions.
- Taking advantages of this option, I could generate additional time-points between _t0 and _t.
- First, if stsplit d1, at(1), 5 episodes created.

Observation sequences created

	id	survt	event	_st	_d	_t	_t0	d1	d2	d3
1	1	1	.	1	0	1	0	0	0	0
2	1	2	.	1	0	2	1	1	0	0
3	1	3	1	1	1	3	2	1	2	0
4	2	1	.	1	0	1	0	0	0	0
5	2	2	.	1	0	2	1	1	0	0
6	2	3	.	1	0	3	2	1	2	0
7	2	4	1	1	1	4	3	1	2	3
8	3	1	.	1	0	1	0	0	0	0
9	3	2	.	1	0	2	1	1	0	0
10	3	3	.	1	0	3	2	1	2	0
11	3	4	0	1	0	4	3	1	2	3
12	4	1	.	1	0	1	0	0	0	0
13	4	2	1	1	1	2	1	1	0	0
14	5	1	.	1	0	1	0	0	0	0
15	5	2	.	1	0	2	1	1	0	0
16	5	3	1	1	1	3	2	1	2	0

- If stsplits d2, at(2), then 4 episodes will be generated.
- In the same way, if stsplits d3, at(3), then 2 episodes created. Stop here.
- For example, id 1 has three sequences of time: 0 to 1, 1 to 2, and 2 to 3.

Handling time-varying covariates

	id	event1	event2	event3	event4	per1	per2	per3	per4	busi
1	1	0	0	1	0	1	2	3	4	1
2	2	0	0	0	1	2	2	3	3	1
3	3	0	0	0	0	0	1	1	1	1
4	4	0	1	0	0	0	3	3	4	2
5	5	0	0	1	0	1	1	2	2	2

	A	B	C	D	E	F	G	H	I	J	K	L
1	x1	x2	x3	x4	y1	y2	y3	y4	z1	z2	z3	z4
2	0	0	1	.	1	2	3	4	1	1	1	1
3	0	0	0	1	2	2	3	3	1	1	1	1
4	0	0	0	0	0	1	1	1	1	1	1	1
5	0	1	.	.	0	3	3	4	2	2	2	2
6	0	0	1	.	1	1	2	2	2	2	2	2

- Copy and paste EVENT, PER(tvc) and BUSI (not tvc) into the excel, and save it in the form of (.csv).
- Change the names of two variables to x1... x4, y1... y4, z1... z4 so that Stata can recognize them.
- It is better to replace four 0s defined after the occurrence of event by .

Reshape it into the panel-type data

	id	year	x	y	z
1	1	1	0	1	1
2	1	2	0	2	1
3	1	3	1	3	1
4	1	4	.	4	1
5	2	1	0	2	1
6	2	2	0	2	1
7	2	3	0	3	1
8	2	4	1	3	1
9	3	1	0	0	1
10	3	2	0	1	1
11	3	3	0	1	1
12	3	4	0	1	1
13	4	1	0	0	2
14	4	2	1	3	2
15	4	3	.	3	2
16	4	4	.	4	2
17	5	1	0	1	2
18	5	2	0	1	2
19	5	3	1	2	2
20	5	4	.	2	2

- insheet using
d:\tvc_exercise.csv
- gen id=_n
- Reshape long x y z,
i(id) j(year)

Consider contributions of id to units

	id	year	per	busi
1	1	1	1	1
2	1	2	2	1
3	1	3	3	1
4	2	1	2	1
5	2	2	2	1
6	2	3	3	1
7	2	4	3	1
8	3	1	0	1
9	3	2	1	1
10	3	3	1	1
11	3	4	1	1
12	4	1	0	2
13	4	2	3	2
14	5	1	1	2
15	5	2	1	2
16	5	3	2	2

- If using `drop if x==`, then a new layout of id exactly corresponds with the original data. (# of rows=16)
- This is why I replaced four 0s by four .s at the beginning.
- `drop x; rename y PER; rename z BUSI`

Merge the two dataset

	id	survt	event	_st	_d	_t	_t0	d1	d2	d3
1	1	1	.	1	0	1	0	0	0	0
2	1	2	.	1	0	2	1	1	0	0
3	1	3	1	1	1	3	2	1	2	0
4	2	1	.	1	0	1	0	0	0	0
5	2	2	.	1	0	2	1	1	0	0
6	2	3	.	1	0	3	2	1	2	0
7	2	4	1	1	1	4	3	1	2	3
8	3	1	.	1	0	1	0	0	0	0
9	3	2	.	1	0	2	1	1	0	0
10	3	3	.	1	0	3	2	1	2	0
11	3	4	0	1	0	4	3	1	2	3
12	4	1	.	1	0	1	0	0	0	0
13	4	2	1	1	1	2	1	1	0	0
14	5	1	.	1	0	1	0	0	0	0
15	5	2	.	1	0	2	1	1	0	0
16	5	3	1	1	1	3	2	1	2	0

	id	year	per	busi
1	1	1	1	1
2	1	2	2	1
3	1	3	3	1
4	2	1	2	1
5	2	2	2	1
6	2	3	3	1
7	2	4	3	1
8	3	1	0	1
9	3	2	1	1
10	3	3	1	1
11	3	4	1	1
12	4	1	0	2
13	4	2	3	2
14	5	1	1	2
15	5	2	1	2
16	5	3	2	2

2) Goal and why?

	d1	d2	d3	d4
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

- How to create a new dataset for discrete-time survival analysis?
- It boils down to making time-dummy variables in Stata.
- Recalling identity matrix, the following commands can be used: `matrix define d=I(4); svmat float d`
- And then, I could insert this time-dummy variables in the original data.

3) Goal and why?

- Some variables might have different values at different time-points. In this situation, one might want to define a new segmented time-dependent covariate.
- I used recid.dat as an example here (obs=432, time period=52).
- First, stset time, failure(arrest==1)
- Second, generate emp= ($_T < 1$)*emp1+($_T \geq 1$ & $_T < 2$)*emp2+($_T \geq 2$ & $_T < 3$)* emp3+ ... + ($_T \geq 50$ & $_T < 51$) * emp51 + ($_T \geq 51$ & $_T < 52$) * emp52
- This step can be used in SPSS (T_ instead of _T)
- In SAS, emp can be defined as follows: if time=1 then emp=emp1; ... if time=52 then emp=emp52;

recid.dat

	time	arrest	finaid	age	race	workexp	marital	parole	priors	educat	empl
1	20	1	0	27	1	0	0	1	3	3	0
2	17	1	0	18	1	0	0	1	8	4	0
3	25	1	0	19	0	1	0	1	13	3	0
4	52	0	1	23	1	1	1	1	1	5	0
5	52	0	0	19	0	1	0	1	3	3	0
6	52	0	0	24	1	1	0	0	2	4	0
7	23	1	0	25	1	1	1	1	0	4	1
8	52	0	1	21	1	1	0	1	4	3	0
9	52	0	0	22	1	0	0	0	6	3	0
10	52	0	0	20	1	1	0	0	0	5	0
11	52	0	1	26	1	0	0	1	3	3	0
12	52	0	0	40	1	1	0	0	2	5	0
13	37	1	0	17	1	1	0	1	2	3	0
14	52	0	0	37	1	1	0	0	2	3	0
15	25	1	0	20	1	0	0	1	3	4	0
16	46	1	1	22	1	1	0	1	2	3	0
17	28	1	0	19	1	0	0	0	7	3	0
18	52	0	0	20	1	0	0	0	2	3	0

Logical expression, but too long?

- $(_t < 1) * emp1 + (_t \geq 1 \ \& \ _t < 2) * emp2 + (_t \geq 2 \ \& \ _t < 3) * emp3 + (_t \geq 3 \ \& \ _t < 4) * emp4 + (_t \geq 4 \ \& \ _t < 5) * emp5 + (_t \geq 5 \ \& \ _t < 6) * emp6 + (_t \geq 6 \ \& \ _t < 7) * emp7 + (_t \geq 7 \ \& \ _t < 8) * emp8 + (_t \geq 8 \ \& \ _t < 9) * emp9 + (_t \geq 9 \ \& \ _t < 10) * emp10 + (_t \geq 10 \ \& \ _t < 11) * emp11 + (_t \geq 11 \ \& \ _t < 12) * emp12 + (_t \geq 12 \ \& \ _t < 13) * emp13 + (_t \geq 13 \ \& \ _t < 14) * emp14 + (_t \geq 14 \ \& \ _t < 15) * emp15 + (_t \geq 15 \ \& \ _t < 16) * emp16 + (_t \geq 16 \ \& \ _t < 17) * emp17 + (_t \geq 17 \ \& \ _t < 18) * emp18 + (_t \geq 18 \ \& \ _t < 19) * emp19 + (_t \geq 19 \ \& \ _t < 20) * emp20 + (_t \geq 20 \ \& \ _t < 21) * emp21 + (_t \geq 21 \ \& \ _t < 22) * emp22 + (_t \geq 22 \ \& \ _t < 23) * emp23 + (_t \geq 23 \ \& \ _t < 24) * emp24 + (_t \geq 24 \ \& \ _t < 25) * emp25 + (_t \geq 25 \ \& \ _t < 26) * emp26 + (_t \geq 26 \ \& \ _t < 27) * emp27 + (_t \geq 27 \ \& \ _t < 28) * emp28 + (_t \geq 28 \ \& \ _t < 29) * emp29 + (_t \geq 29 \ \& \ _t < 30) * emp30 + (_t \geq 30 \ \& \ _t < 31) * emp31 + (_t \geq 31 \ \& \ _t < 32) * emp32 + (_t \geq 32 \ \& \ _t < 33) * emp33 + (_t \geq 33 \ \& \ _t < 34) * emp34 + (_t \geq 34 \ \& \ _t < 35) * emp35 + (_t \geq 35 \ \& \ _t < 36) * emp36 + (_t \geq 36 \ \& \ _t < 37) * emp37 + (_t \geq 37 \ \& \ _t < 38) * emp38 + (_t \geq 38 \ \& \ _t < 39) * emp39 + (_t \geq 39 \ \& \ _t < 40) * emp40 + (_t \geq 40 \ \& \ _t < 41) * emp41 + (_t \geq 41 \ \& \ _t < 42) * emp42 + (_t \geq 42 \ \& \ _t < 43) * emp43 + (_t \geq 43 \ \& \ _t < 44) * emp44 + (_t \geq 44 \ \& \ _t < 45) * emp45 + (_t \geq 45 \ \& \ _t < 46) * emp46 + (_t \geq 46 \ \& \ _t < 47) * emp47 + (_t \geq 47 \ \& \ _t < 48) * emp48 + (_t \geq 48 \ \& \ _t < 49) * emp49 + (_t \geq 49 \ \& \ _t < 50) * emp50 + (_t \geq 50 \ \& \ _t < 51) * emp51 + (_t \geq 51 \ \& \ _t < 52) * emp52$

Type mismatch r(109):

if not numerical, then doublecheck

- **confirm numeric variable emp1 emp2 emp3
emp4 emp5 emp6 emp7 emp8 emp9 emp10
emp11 emp12 emp13 emp14 emp15 emp16
emp17 emp18 emp19 emp20 emp21 emp22
emp23 emp24 emp25 emp26 emp27 emp28
emp29 emp30 emp31 emp32 emp33 emp34
emp35 emp36 emp37 emp38 emp39 emp40
emp41 emp42 emp43 emp44 emp45 emp46
emp47 emp48 emp49 emp50 emp51 emp52**