Sessionization Methods

Getting Past Big Data Noise

IQPC Big Data Exchange

Berkeley, September 22, 2013

Jim Porzak

What we will cover:

1. Why sessionize?
2. Big data challenges.
3. Sessions and sessionization.
4. Life-stage ideas.
5. Discussion.

My mentor, Tony Gaunt advised:

“Jim, take care of your customers and they will take care of you.”

“Customers” here means folks who have purchased from us, are in processes of purchasing, or should purchase (AKA “prospects”). Or, equivalent in your domain.

Which leads to my corollary:

“Understand your customers so you can serve them better.”

Today, that means use everything we know about our customers
= “*big data*”

So, our job is to understand…

1. the customer,
2. the whole customer, and
3. nothing but the customer.

C ustomer

Initiated

A ctions

CIA’s are:

- Ordering or committing
- Browsing store (physical) or site (on-line)
- Other preference related actions
- Responding to outbound marketing:
  - Email, catalog, other direct mail
- Contacting customer service, etc.
- Social behavior around your brand.
- And more!

Aside: Order Processing in SF

“How do you sell bread?”

1. “You tell me what you want.”
2. “I give you the loaf.”
3. “I tell you how much it costs.”
4. “You give me the money.”

Big gotcha: Most operational order processing systems only track these four steps.

CIA properties

CIA’s are the result of conscious decisions which, we assume, give us some insight into our customer’s motivation.

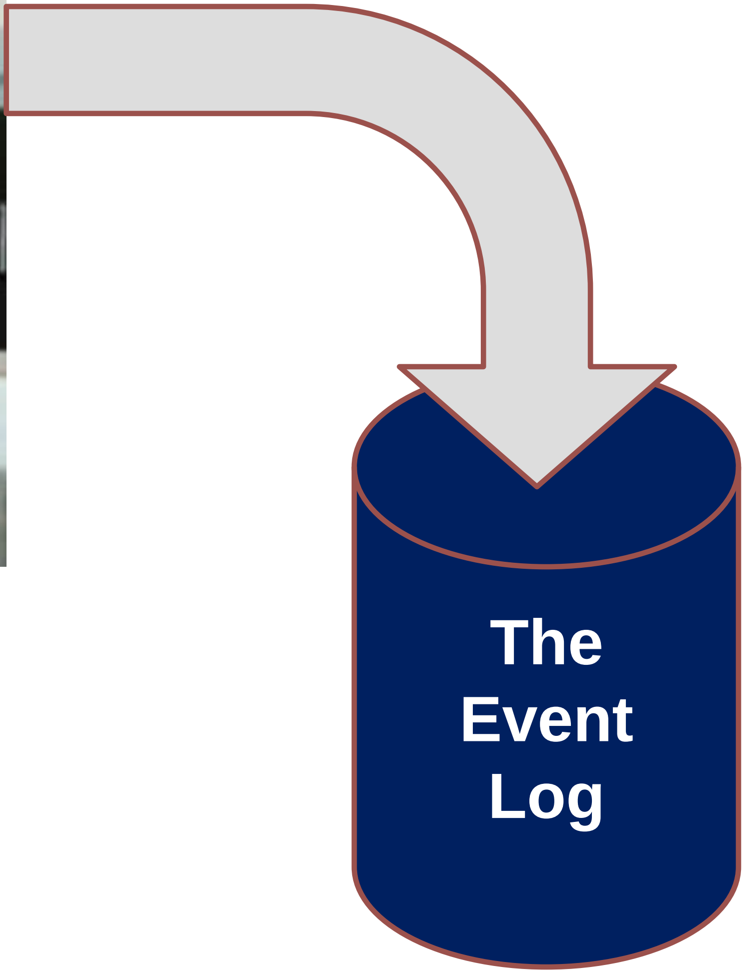
Real-time CIA’s:

- Verbal (& chat)
- Physical (pick-up, click, touch)

Delayed CIA’s:

- Mail (snail & E)

Logging Customer Behavior



CIA's vs Log Data

CIA's

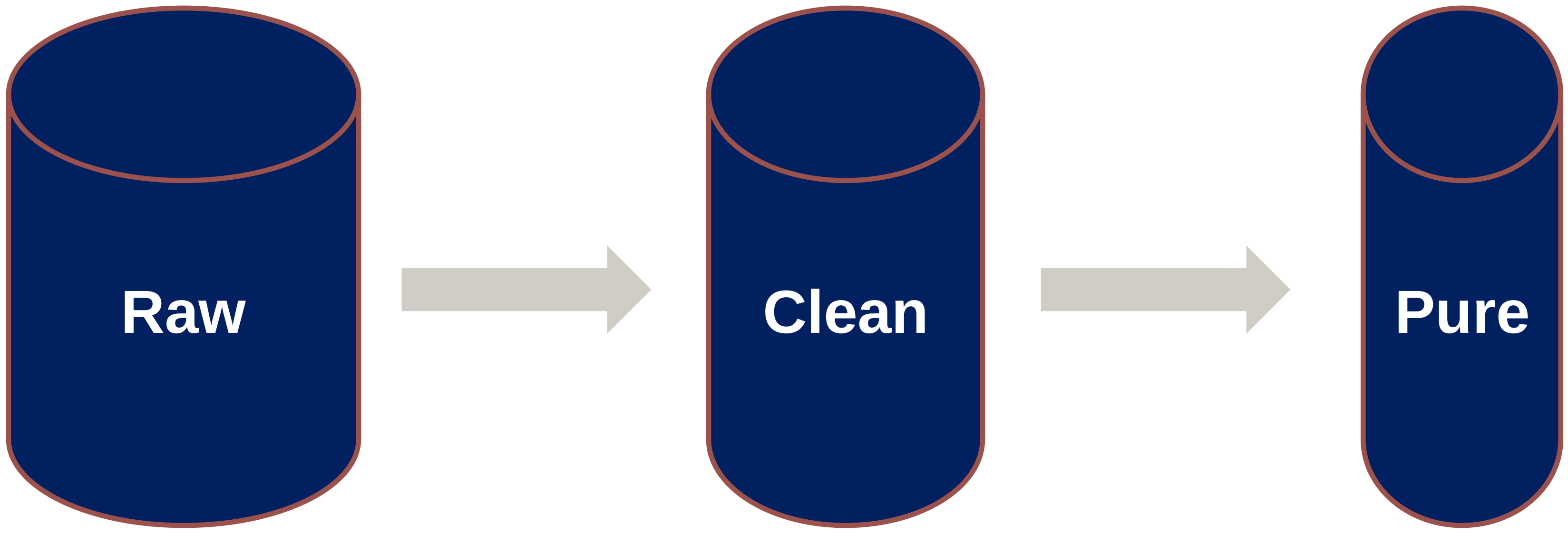
- Slow: Couple per minute.
- Few: One per conscious action.
- Specific: to individuals

Log Data

- Fast: ~100's per second.
- Many: one CIA spawns a lot of log events
- Ambiguous: non-human actors initiate events

From Log to CIA Event

(3. nothing but the customer)



Remove all machine generated events not directly a result of CIA

Remove non-human events – bots, etc.

Typical CIA Event Streams

Physical

- Enter Store
- Walk past shoes
- Pick-up yellow Polo's
- Put them down
- Walk to ties
- Pick up cardinal red
- Put it down
- Pick up gold & blue
- Put it down
- Leave store

Web Site

- Click on Polo banner.
- Enter Polo page
- Click yellow color
- Zoom & rotate photo
- Click on "ties" menu item
- Click on "school" menu item
- Click on "Cal Berkeley"

Event Stream Data (Pure)

1. Timestamp
2. Customer Identifier
3. Event Properties
   - Location
   - Action
   - Item
   - Amount ($’s & #’s)
   - Other parameters

Aside: Identifying Customers

The Customer ID (AKA User ID) is *<u>your</u>* primary unique identifier for your customers.

USE IT EVERYWHERE!

Especially when exchanging data.

Some secondary, fuzzy, identifiers:

- Cookie ID
- Email address
- Phone Number
- Street Address
- Name

Sessionize Event Streams (1 of 2)

Data: Cust ID, Timestamp, <event details>

1. Sort by Cust ID, Timestamp
2. Add time-interval to prior event
3. Apply end-of-session rule
   – “leaving” store or interval > some # minutes
4. Sequence #’s for
   1. Sessions for customer
   2. Events within session

Sessionize Event Streams (2 of 2)

Summarize sessions to single rows:

- Customer ID
- Session Sequence #
- Timestamp for start of session
- Duration in minutes
- Interval to prior session in hours (?)
- # CIA’s in session
- <the “What” descriptors>

Finding Customers in Anonymous Visitors

Visitor ID	Customer ID	TimeStamp	Notes
123	NULL	4/1/13	Anonymous
123	NULL	4/3/13	Anonymous
123	ABC	4/21/13	Register
123	NULL	4/23/13	Not logged in
456	ABC	4/24/13	New cookie
123	ABC	5/15/13	Returning

Hint: Keep a table of [visitor ID, Customer ID, IP Address, …] associations

Sequences of Sessions

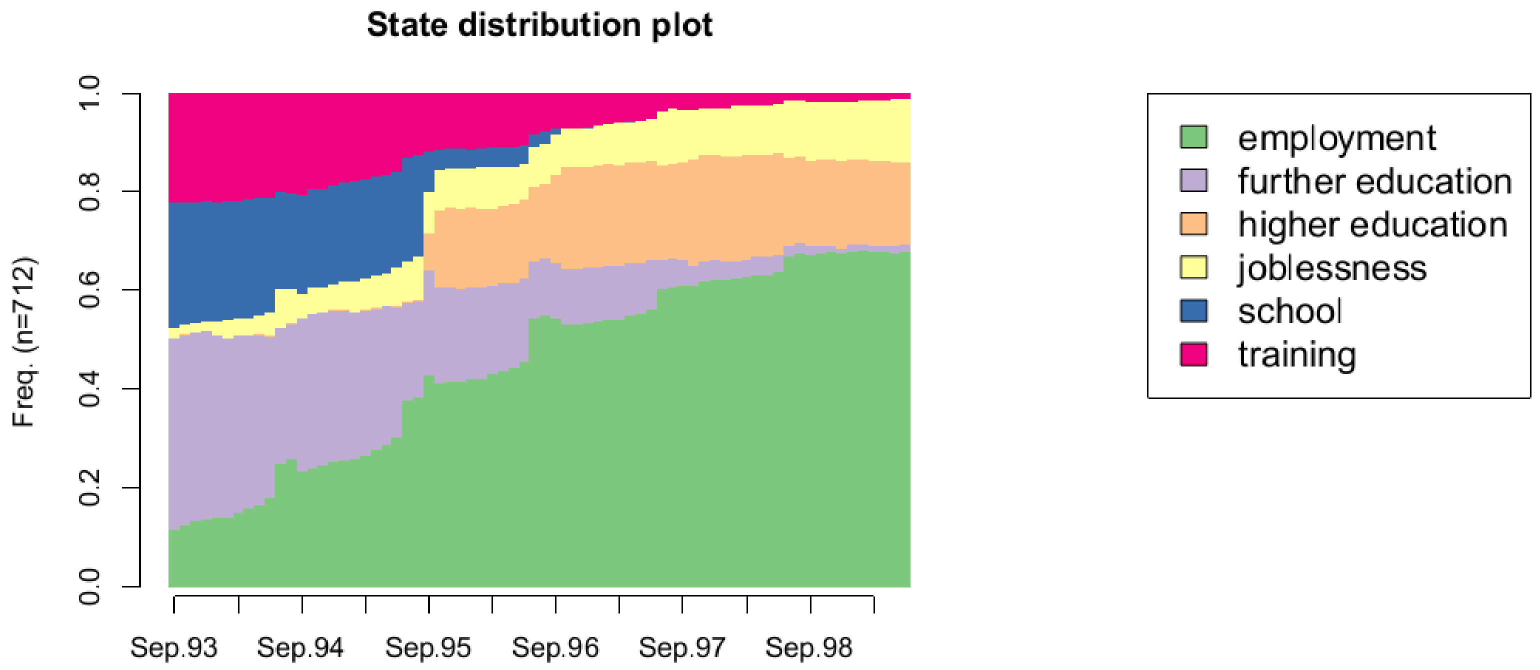
From hypothetical shopping site:

Cust ID	Timestamp	Summary
ABC	4/1/12	Browse bike
ABC	6/5/12	Buy two backpacks
ABC	9/15/12	Browse dishes
ABC	12/6/12	Buy food processor
ABC	4/1/13	Browse baby toys

Understanding Life Stages

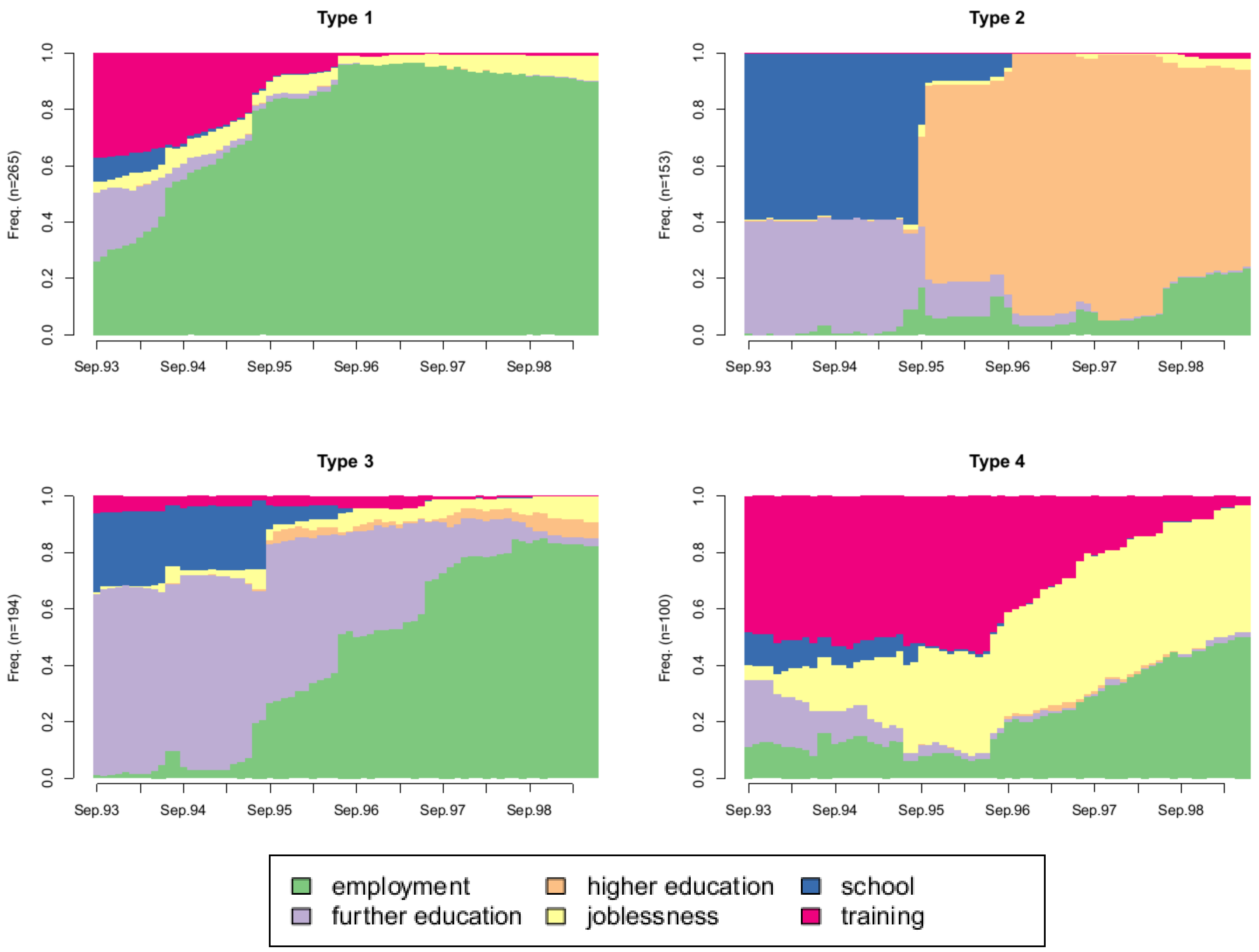
Sociologists do this. Use their tools!

How a cohort of Swiss youth transition from learning to working.



Source: TraMineR User's Guide. See appendix for link.

Life Stage Segments



Source: TraMineR User's Guide. See appendix for link.

Summary

- It’s all about customers & CIA’s
- Raw logs to CIA’s
- Sessionization
- Integration along customer identifiers
- Life-stage ideas

Questions? Comments?

Now would be the time!

Email: JPorzak@gmail.com
LinkedIn: http://www.linkedin.com/in/jimporzak/

APPENDIX

- CIA Data Sources
- Sessionization in SQL
- Useful Links

CIA Data Sources

- Brick & mortar experience
- Web experience – site & external
- Back-room/back-end processing
- Customer service & other in-bound
- Email, direct mail, & other out-bound

Integration is hard, but yields much better view of customer.

Sessionization in PostgreSQL Part 1

```
-- get event sequence #s & seconds after prior event
CREATE TABLE v_sessions1 AS
SELECT *,
       ROW_NUMBER() OVER(Members) AS event_seq_number,
       event_at - LAG(event_at) OVER(Members)
           AS interval_to_prior
  FROM v_events
WINDOW Members
    AS (PARTITION BY member_id   -- unique member ID
            ORDER BY event_at    -- timestamp of event
       )
;
```

Sessionization in PostgreSQL Part 2

```
-- update with session sequence #
CREATE TABLE v_sessions2 AS
SELECT *,
       COUNT(CASE WHEN interval_to_prior IS NULL OR
                       interval_to_prior > '30 minutes'
                  THEN 1 ELSE NULL END) OVER(Members)
           AS session_seq_number
  FROM v_sessions1
WINDOW Members
    AS (PARTITION BY member_id      -- unique member ID
            ORDER BY event_seq_number  -- Session #
       )
;
```

Sessionization in PostgreSQL Part 3

```
-- now roll up into sessions getting session start, total time in session,
-- site areas explored, other site specific rollups

CREATE TABLE v_sessions AS
SELECT member_id,
       session_seq_number,
       MIN(event_at) AS session_start_at,
       COUNT(*) AS num_events_in_session,
       SUM(CASE WHEN interval_to_prior > '30 minutes'
                THEN NULL ELSE interval_to_prior END) AS session_duration,
       STRING_AGG(DISCRETE site_area, ', ') AS site_areas_visited,
       <other site specific aggregations>
  FROM v_sessions2
 GROUP BY member_id,
          session_seq_number
 ORDER BY member_id,
          session_seq_number
;
```

Useful Links

- PostgreSQL: http://www.postgresql.org/
  - Window functions: http://www.postgresql.org/docs/current/static/tutorial-window.html
- R: http://www.r-project.org/
  - Rstudio: http://www.rstudio.com/
  - TraMineR: http://mephisto.unige.ch/traminer/
- Vertica’s Sessionize with Style
  - Part 1: http://www.vertica.com/2010/10/04/sessionize-with-style-part-1/
  - Part 2: http://www.vertica.com/2010/10/28/sessionize-with-style-part-2/