UNIVERSITY OF WISCONSIN–MADISON

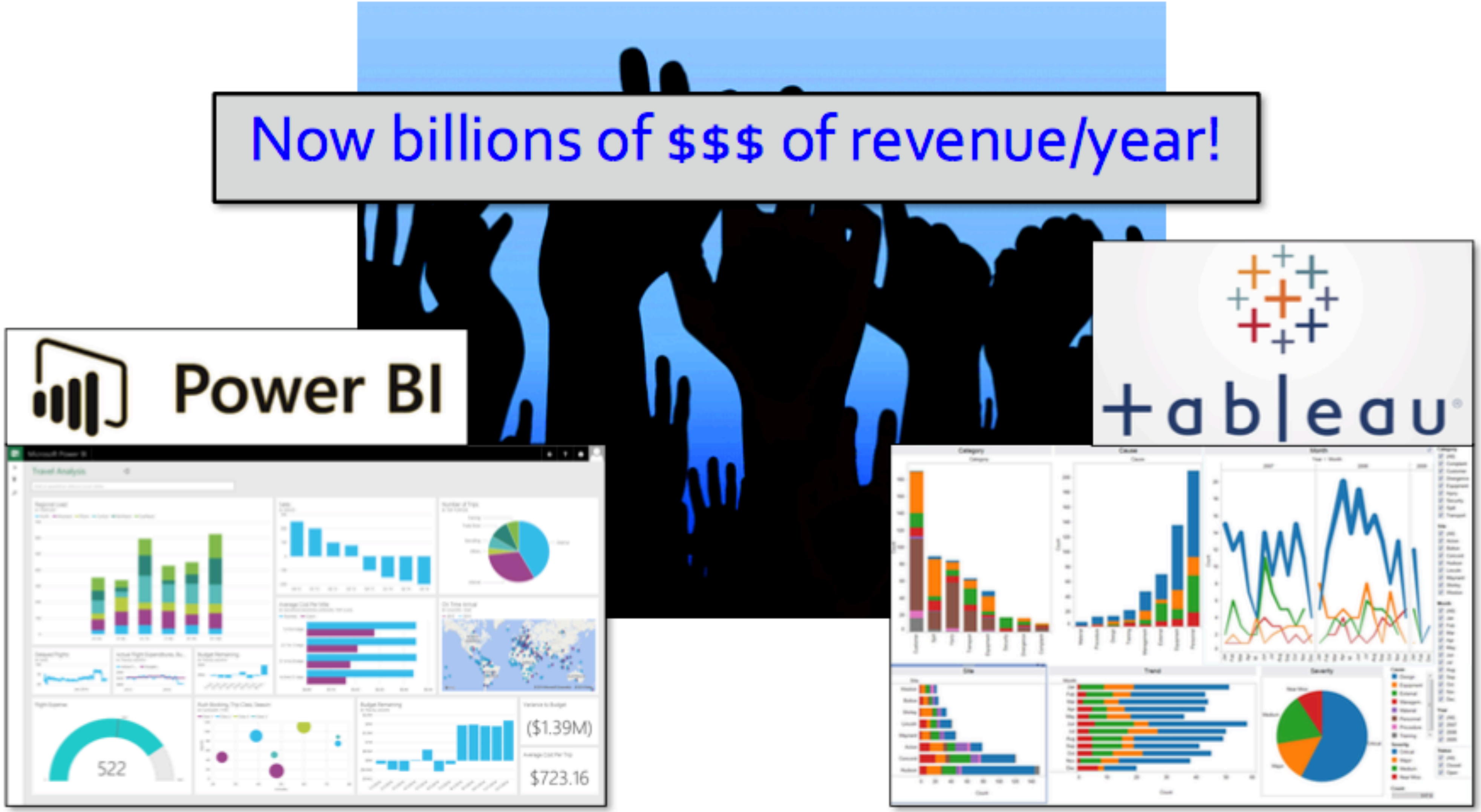
CS639:
Data Management for Data Science

Lecture 25: EDA

Theodoros Rekatsinas

Data Visualizations Today

Now billions of $$$ of revenue/year!



Data Visualizations Today

- → Billions in revenue
- → Huge audience
- → Interactions not code

Data Visualization *is* Data Science for the 99%!

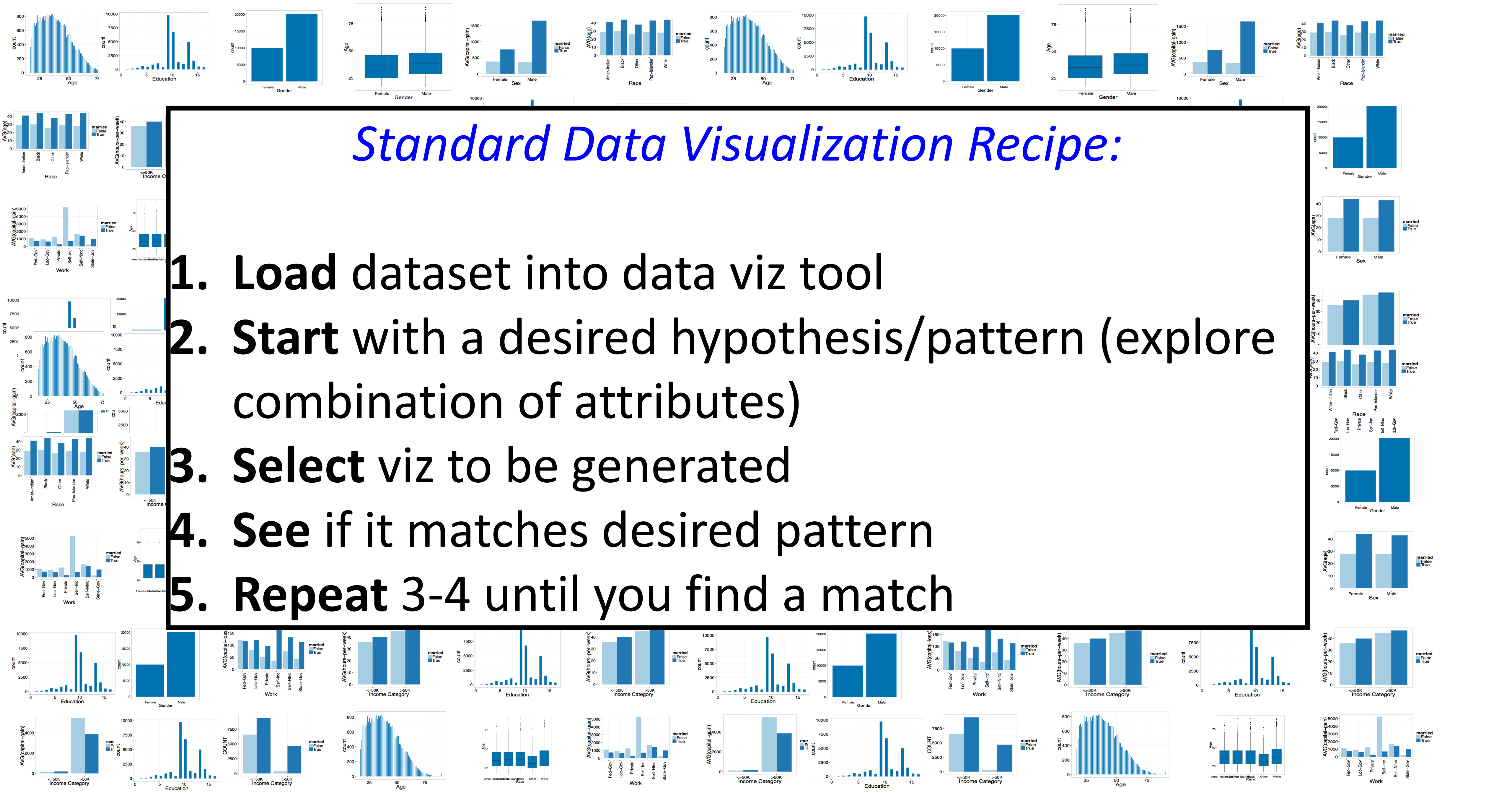
However, these tools are SERIOUSLY limited in their power…

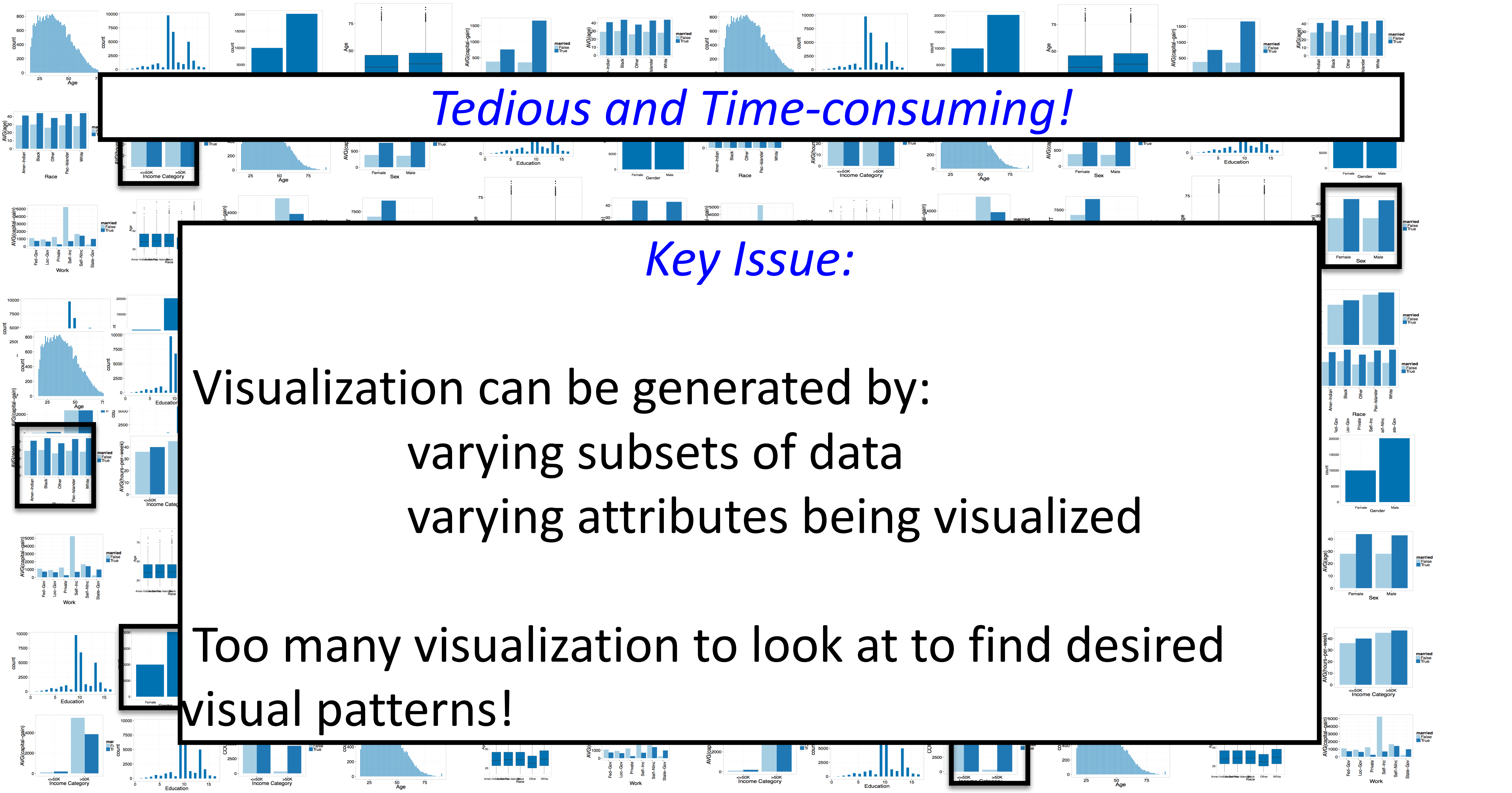
Deriving insights is laborious and time-consuming!

↑ errors ↑ frustration ↑ wasted time ↓ insights ↓ exploration

Standard Data Visualization Recipe:

1. **Load** dataset into data viz tool
2. **Start** with a desired hypothesis/pattern (explore combination of attributes)
3. **Select** viz to be generated
4. **See** if it matches desired pattern
5. **Repeat** 3-4 until you find a match





Tedious and Time-consuming!

Key Issue:

Visualization can be generated by:

varying subsets of data

varying attributes being visualized

Too many visualization to look at to find desired visual patterns!

1. Visualization recommendations

What you will learn about in this section

1. Space of Visualizations

2. Recommendation Metrics

Goal

Given a dataset and a task, automatically produce a set of visualizations that are the most "interesting" given the task

Particularly vague

Goal

Given a dataset and a task, automatically produce a set of visualizations that are the most "interesting" given the task

Example

- Data analyst studying census data
- age, education, marital-status, sex, race, income, hours-worked etc.
  - A = # attributes in table

- Task: Compare on various socioeconomic indicators, unmarried adults vs. all adults

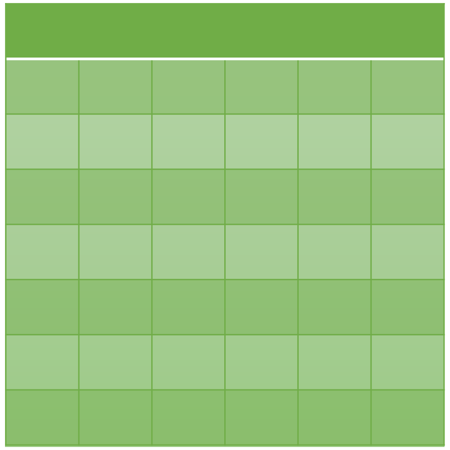
Space of visualizations

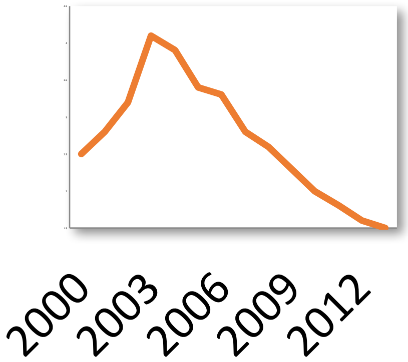
For simplicity, assume a single table
(star schema)

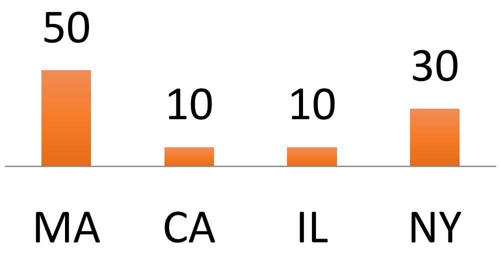
Visualizations = agg. + grp. by queries

Vi = SELECT d, f(m)
FROM table
WHERE ___
GROUP BY d

(d, m, f):
dimension, measure, aggregate







Space of visualizations

Vi = SELECT d, f(m)
FROM table
WHERE ___
GROUP BY d

(d, m, f):
dimension, measure, aggregate
{d} : race, work-type, sex etc.
{m} : capital-gain, capital-loss, hours-per-week
{f} : COUNT, SUM, AVG

Goal

Given a dataset and a task, automatically produce a set of visualizations that are the most "**interesting**" given the task

Interesting visualizations

A visualization is interesting if it displays

a large deviation from some reference

Deviation-based Utility

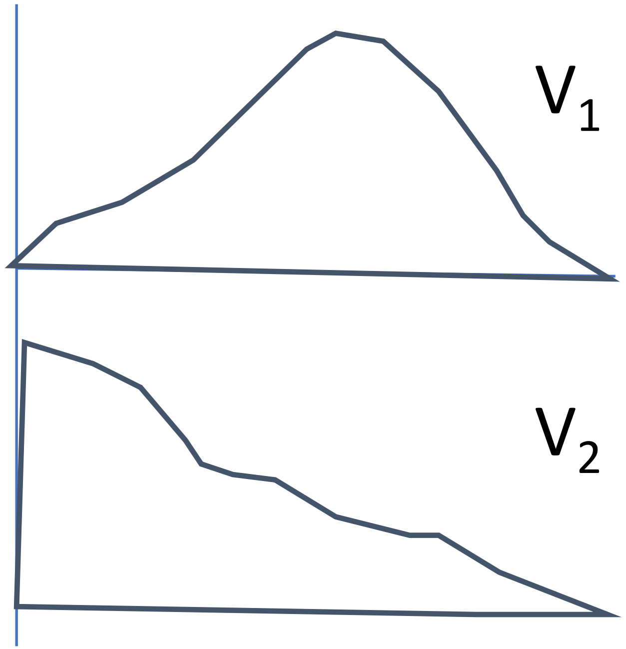
Target Reference

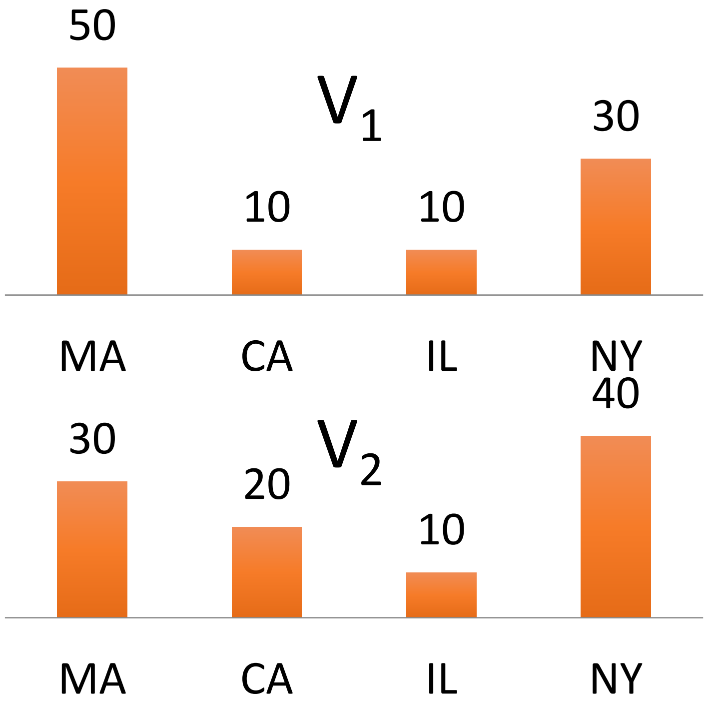
Task: compare unmarried adults with all adults

V1 = SELECT d, f(m) FROM table WHERE target GROUP BY d

V2 = SELECT d, f(m) FROM table WHERE reference GROUP BY d

Compare induced probability distributions!

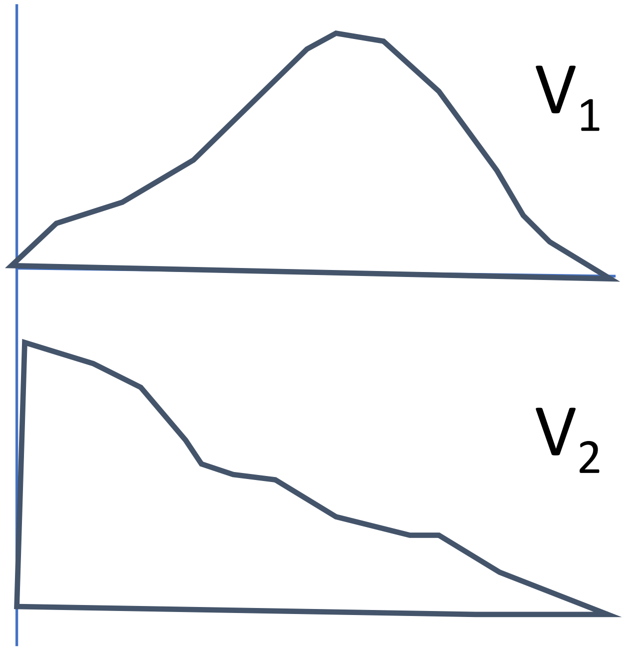




Deviation-based Utility Metric

A visualization is interesting if it displays
a large deviation from some reference

Many metrics for computing distance between distributions



D [P(V1), P(V2)]

Earth mover's distance
L1, L2 distance
K-L divergence

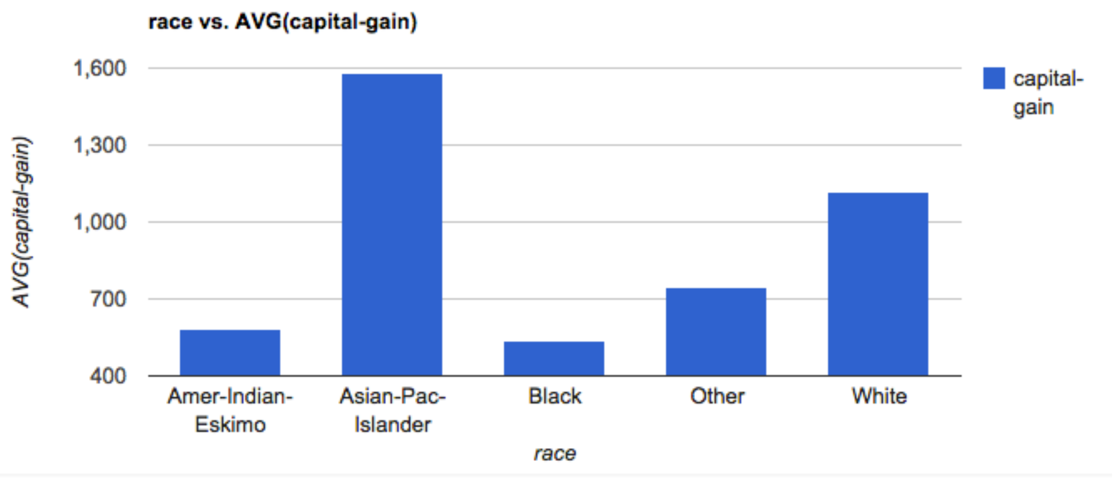
Any distance metric b/n distributions is OK!

Computing Expected Trend

Race vs. AVG(capital-gain)

Reference Trend

SELECT race, AVG(capital-gain) FROM census GROUP BY race



$P(V_1)$

Expected

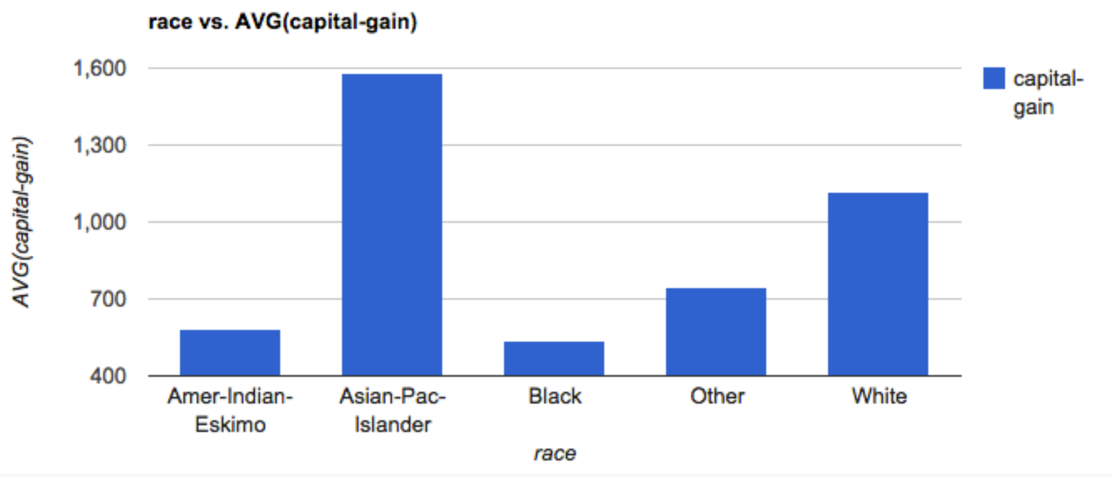
Distribution

Computing Actual Trend

Race vs. AVG(capital-gain)

TargetTrend

SELECT race, AVG(capital-gain) FROM census GROUP BY race WHERE <u>marital-status='unmarried'</u>

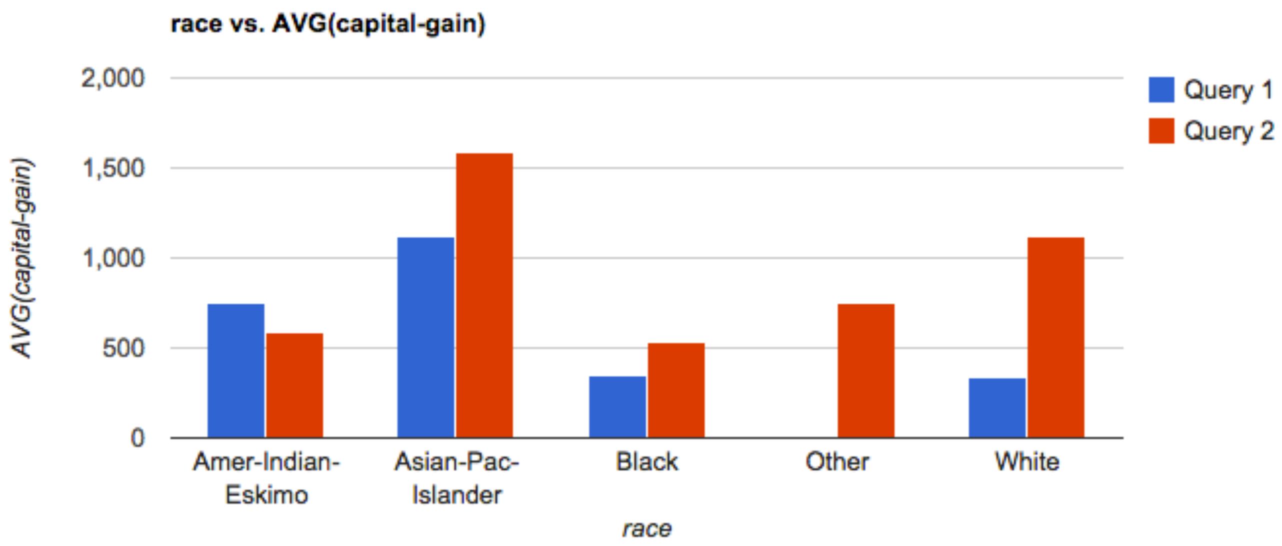


$P(V_2)$

Actual

Distribution

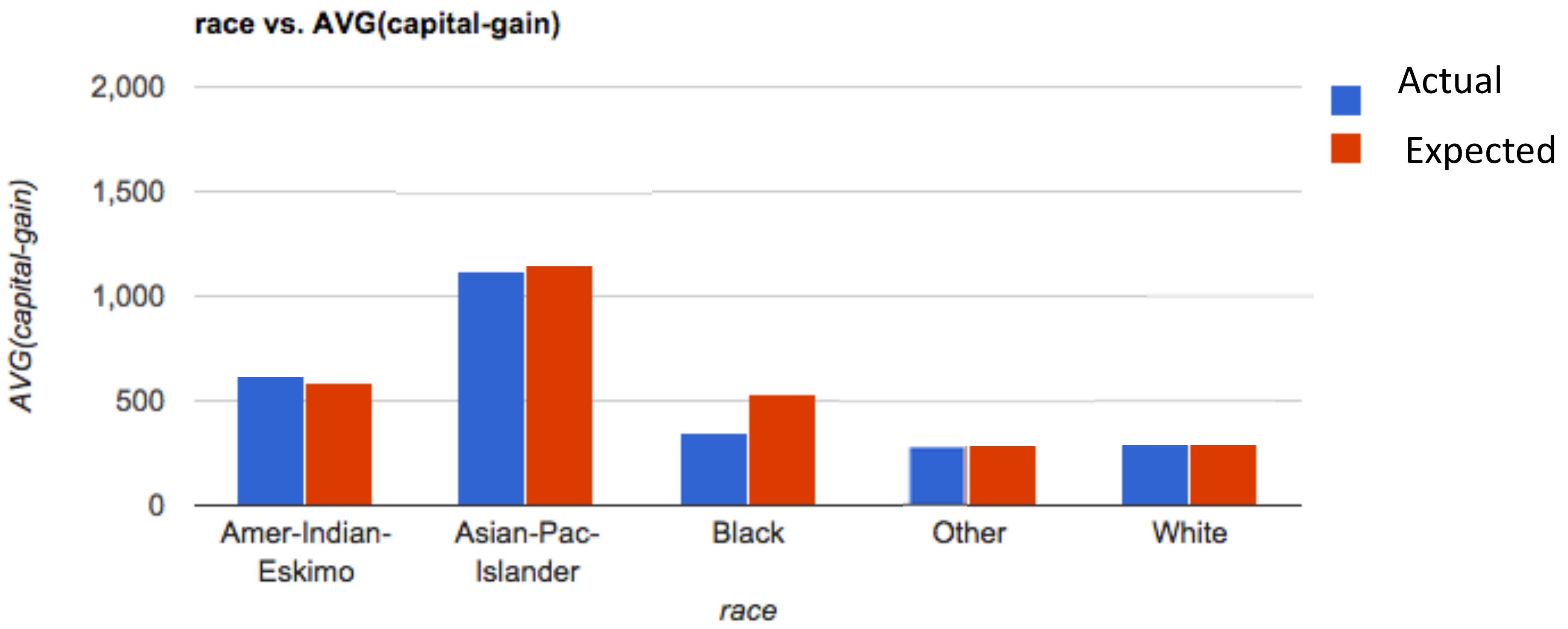
Computing Utility



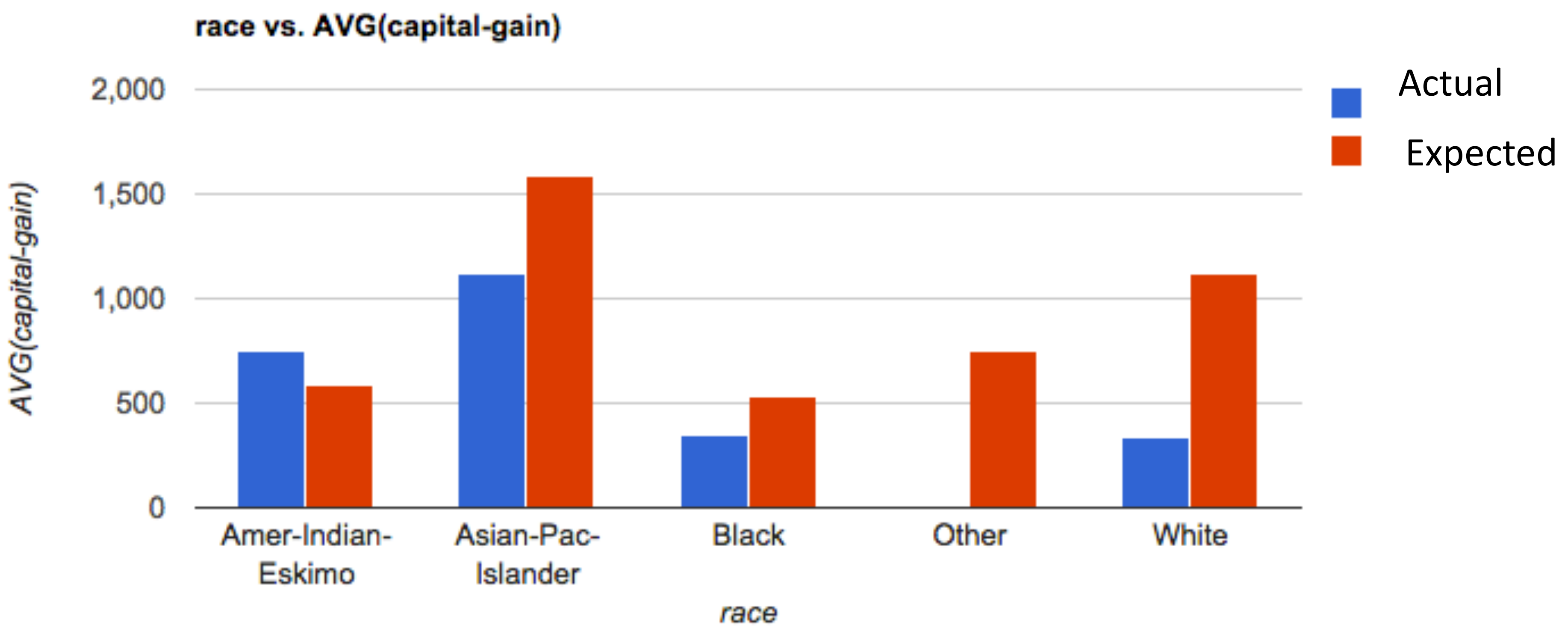
$$U = D[P(V_1) , P(V_2)]$$

D = EMD, L2 etc.

Low Utility Visualization



High Utility Visualization



Other metrics

- Data characteristics
- Task or Insight
- Semantics and Domain Knowledge
- Visual Ease of Understanding
- User Preference

2. DB-inspired Optimizations

What you will learn about in this section

1. Ranking Visualizations

2. Optimizations

Ranking

Across all (d, m, f), where

V1 = SELECT d, f(m) FROM table WHERE target GROUP BY d

V2 = SELECT d, f(m) FROM table WHERE reference GROUP BY d

Goal: return *k* best utility visualizations (d, m, f),
(those with largest D[V1, V2])

Vi = (d: dimension, m: measure, f: aggregate)

10s of dimensions, 10s of measures, handful of aggregates

2* d * m * f

➔100s of queries for a single user task!

➔Can be even larger. How?

Even larger space of queries

- Binning
- 3 dimensional or 4 dimensional visualizations
- Scatterplot or map visualizations
- …

Back to ranking

Across all (d, m, f), where

V1 = SELECT d, f(m) FROM table WHERE target GROUP BY d

V2 = SELECT d, f(m) FROM table WHERE reference GROUP BY d

Goal: return *k* best utility visualizations (d, m, f),
(those with largest D[V1, V2])

Naïve Approach

For each (d, m, f) in sequence

evaluate queries for V1 (target), V2 (reference)

compute D[V1, V2]

Return the k (d, m, f) with largest D values

Issues with Naïve Approach

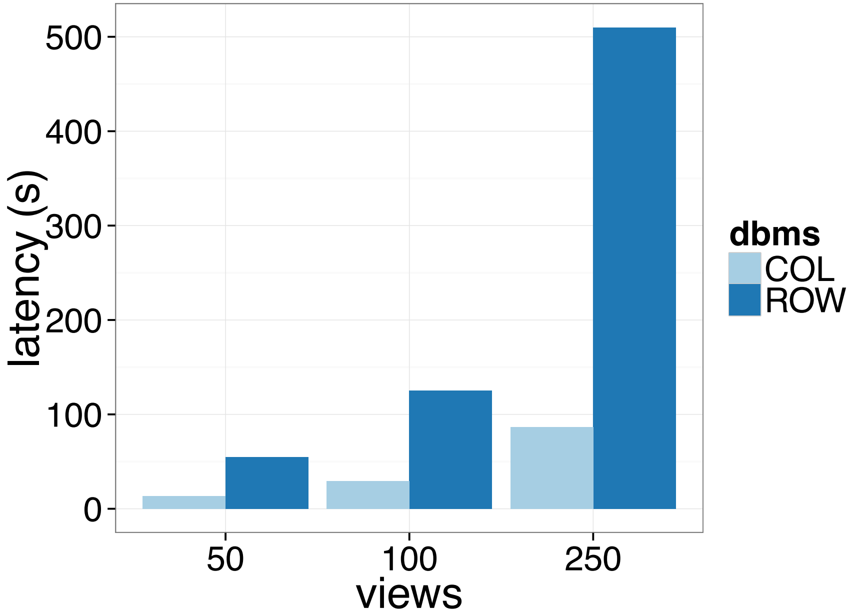
- Repeated processing of same data in sequence across queries
- Computation wasted on low-utility visualizations

Sharing

Pruning

Optimizations

- Each visualization = 2 SQL queries



- Latency > 100s
- Minimize number of queries and scans

Optimizations

- Combine aggregate queries on target and ref

- Combine multiple aggregates

  (d1, m1, f1), (d1, m2, f1) → (d1, [m1, m2], f1)

- Combine multiple group-bys*

  (d1, m1, f1), (d2, m1, f1) → ([d1, d2], m1, f1)

  Could be problematic…

- Parallel Query Execution

Combining Multiple Group-by's

- Too few group-bys leads to many table scans

- Too many group-bys hurt performance
  - # groups = Π (# distinct values per attributes)

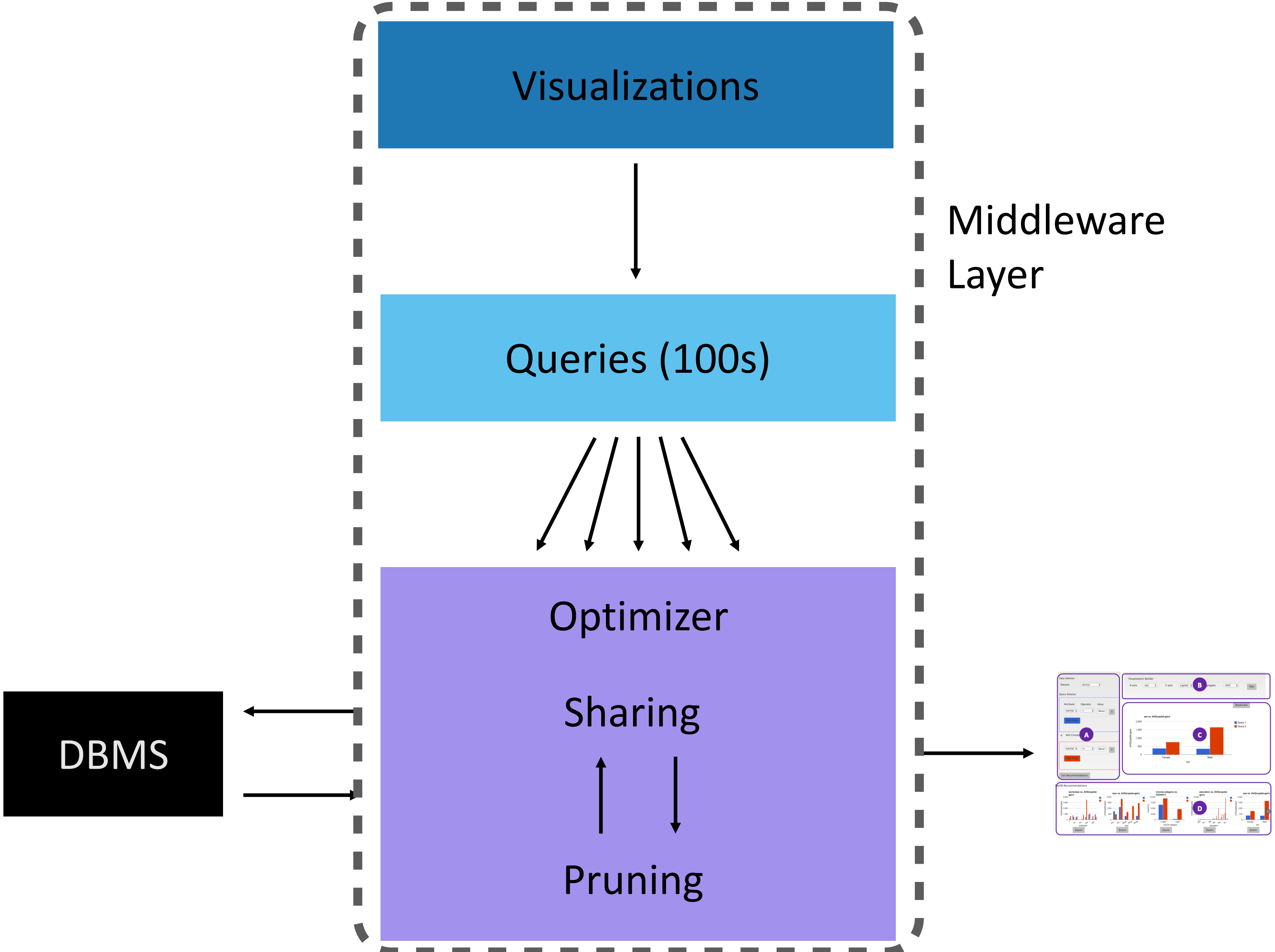
- Optimal group-by combination ≈ bin-packing
  - Bin volume = log S (max number of groups)
  - Volume of items (attributes) = log ($|a_i|$)
  - Minimize # bins s.t.

$$\Sigma_i \log(|a_i|) <= \log S$$

Pruning optimizations

Discard low-utility views early to avoid wasted computation

- Keep running estimates of utility
- Prune visualizations based on estimates
  - Two flavors
    - Vanilla Confidence Interval based Pruning
    - Multi-armed Bandit Pruning

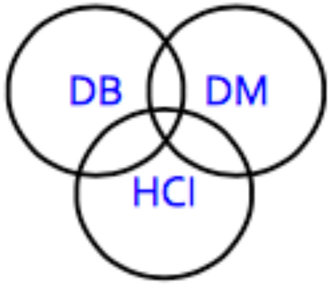

Visualizations
Queries (100s)
Middleware
Layer
Optimizer
Sharing
Pruning
DBMS

More on automated visualizations

Desiderata for automation:

- Expressive – specify what you want
- Interactive – interact with results, cater to non-programmers
- Scalable – get interesting results quickly

Drawing from



Enter Zenvisage:
(zen + envisage: to effortlessly visualize)

zenvisage

8

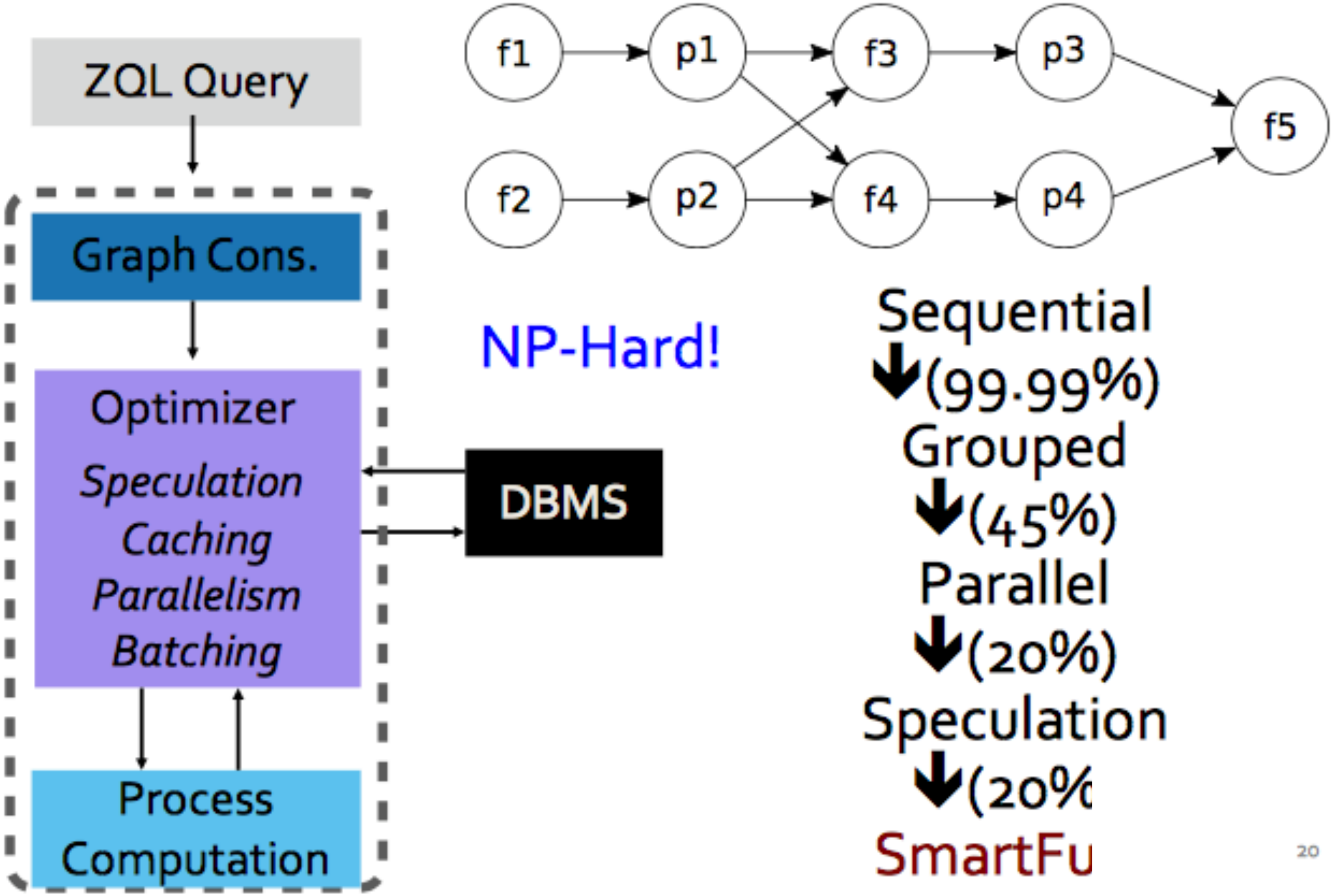
ZQL: a viz exploration language

- Inspired from QBE & VizQL / Grammar of Graphics
- Captures four key operations on viz collections

  Compose *Filter* *Compare* *Sort*

- Incorporates data mining primitives
- Powerful; formally demonstrated "completeness"

Intelligent query optimizer



Summary

Human in the loop analytics are here to stay!