Fundamental Concepts in Data Insight:

Introduction to Big Data

Fundamentals for a General Audience

Introduction to Big Data

- Why now?
    - Why is Big Data important?
    - WB. Who cares about Big Data?
- What is Big Data?
    - How do we decide if a problem is Big Data?
    - WB. When is a problem Big Data?
    - WB. Are all Big Data Systems for Big Data?
    - WB. What is a relational database?
- The 3 Vs
    - Velocity
- Vairety
    - WB. Variety: Relational & Non-Relational
    - WB. Variety: K-V, Documents
    - WB. Variety: Graphs
    - WB. Variety: Columnar
    - WB. Variety: Images & Audio
- Volume
- Review: What is a Data Model?
- Review: What types of Big Data Systems are there?
- WB. Data Archicture & Components
    - What does a modern data system need?
    - Case Study: NoSQL vs. Relational
- Individual Project (10 min)
- Group Project (20 min)

Why now?

- traditional data needs
  - retail
  - transactions
  - data essential to online needs
- origins of big data
  - user behaviour
  - internet
  - public agencies
  - sensor data
  - images, audio, text
    - OCR, ...
- hardware to easily stored
  - HDDs, SSDs
- networks to easily transfer
  - 1Gb/s to 100Gb/s

Why is Big Data important?

- Effective data insight and decision-making requires
  - high quality data
  - large amounts of data
  - live and updated data
- Digital transformation or pivots require data
  - must begin by instituing big data systems

WB. Who cares about Big Data?

- executives, analysts, engineers

What is Big Data?

- Big Data is a term without a definition.
- It is a term set against something else.
- The other: traditional data systems.

> *Big data is a field that treats ways to analyze, systematically extract information from, or otherwise deal with data sets that are too large or complex to be dealt with by* ***traditional data-processing application software****.*
>
> *Big data challenges include capturing data, data storage, data analysis, search, sharing, transfer, visualization, querying, updating, information privacy.*

- wikipedia, Big Data

How do we decide if a problem is Big Data?

WB. When is a problem Big Data?

WB. Are all Big Data Systems for Big Data?

WB. What is a relational database?

The 3 Vs

- Velocity
- Variety
- Volume

Velocity

- real-time
- problems "quickly" become non-traditional
    - traditional solutions *do not scale well*
- response required < 1s
- ratio = traditional-query-time / response-time
- is this close-to or more-than 1?
    - eg., 60 seconds / 120 secods = 0.5
    - eg., 60 seconds / 30 seconds = 2

WB. Variety: Relational & Non-Relational

WB. Variety: K-V, Documents

WB. Variety: Graphs

WB. Variety: Columnar

WB. Variety: Images & Audio

Vairety

- data complexity
- problems "quickly" become non-traditional
    - traditional solutions *do not scale well*
- independent dimensions in the dataset
    - independent attributes of the data
- repeated information:
    - (lat, long), (address), (geotag)
- large amounts of *relevant* non-repeated information
- image: 1MP is 1,000,000
    - even a small image is 1-million dimensions
- if you're considering >20 *relevant*, *indepedent* attributes, may have big-data problem

Volume

- traditional solutions *scale well*
- RAM
    - how much memory does a query need?
    - ratio = query-need / machine-amount
        - of the best plausible single machine
    - eg., 100 GB / 512 GB
        - not a big data problem
    - eg., 1 TB / 512 GB
        - could be a big data problem
        - may also be possible to optimize the query
- Storage
    - how big is the entire dataset?
    - ratio = disk-size / machine-amount
    - eg., 10 TB / 16TB
        - not a big data problem

Review: What is a Data Model?

- The data model describes how the data is structured.
    - The query brings together datasets, reshapes and summaries
- There's a tradeoff
    - either structure data *specific* to a need,
        - query simpler & faster
    - or make the structure general,
        - querying takes longer.
- Relational databases have a universal data structure
    - ie., a table
    - this *can* have negative performance implications
        - if you're problem requires non-tabular structure
            - order,
            - sparcity,
            - heirachy,
            - extensive re-structuring.

Review: What types of Big Data Systems are there?

- non-relational (ie., non-tabular): NoSQL
    - documents
    - graphs
    - columnar stores
        - dataframes, bigtables
    - text, audio, images
- fast small systems
    - event systems
- slow big systems
    - distributied file systems

WB. Data Archicture & Components

What does a modern data system need?

- data lake
    - raw unstructured data
    - images, archives, ...
- data warehouse
    - data structured for analysis
- data applications
    - customer-facing software
    - internal analytical software
- data engineering
    - systems to pipeline and ETL data
- data science
    - systems of data insight and analysis

Case Study: NoSQL vs. Relational

WHITEBOARD

Individual Project (10 min)

- open the demo files for this module
- run, explore and modify

Group Project (20 min)

- consider these types of data, brain storm what detials they have:
  - suspect network
  - case files
  - history of credit card transactions
  - realtime observations of criminal behaviour on honeytrap darkweb sites

- choose from:
  - graph database, document database, relational, event systems
  - not all answers "wrong"

Reflection

- The Strengths & Weaknesses of Relational Databases
- Forensic Case Studies in Big Data
    - NoSQL Systems
    - Event Systems
    - Image & Audio Analysis
- Duplicating Data & Polyglot Persistance
- Realtime vs. Batch Analysis

Appendix

What is SQL?

- structured query language
- increasinly, lingua franca of data

```sql
CREATE TABLE evidence (
    id int, category text, report text, department text
);

INSERT INTO evidence VALUES (1, "Money", "...", "Finance" );

SELECT * FROM evidence WHERE department = "finance";
```

How do big data systems work together?

- events capture maximal analytical info
- data duplicated
    - polyglot persistance
    - including, esp. traditional systems
- realtime and batch analysis
- events are achived on distributed file systems
    - old event streams can be processed whenever
    - "as-if live"

What is a Traditional Data System?

A traditional data system is relational (ie., tabular) and designed to store live transactional data and to provide this data for historical reporting (fact-finding).

Traditional systems are excellent at this task, and handily beat any “big data” system which could be used to accomplish it (often, by a very long way).

When do traditional systems win?

Organizations are in a difficult place when they have to leave the traditional world; always only a problem cannot be solved using those techniques. The big-data world requires complex skills, timelines, project scope and costs.

Traditional systems provide global consistency (the data looks up-to-date from where-ever and when-ever you are); and tabular datasets (rows, columns). Consistency comes with a performance penalty: if everyone has to be up to date, you have to slow parts (or disable them) to keep the whole system accurate. A tabular structure is also not efficient for storing non-tabular datasets: graphs; sparse documents; images and highly complex data (1000s+ columns).

When do traditional systems fail?

There is no fixed point when a traditional system fails and a big data system is needed. Typically, metrics such as the volume (size), velocity (eg., query rate), and variety (data complexity) are used. However these are moving goal posts. In 2000 when “Big Data” became a term of art, volume was the greatest challenge: 1 TB of data would require a very large number of machines. Today, an iPad comes with 1TB, and purchasing several 12 TB hard drives is cheap and easy.

When is a big data system needed?

Today medium-size mainstream businesses are unlikely to depart from traditional systems due to volume constraints alone. Rather, if big data systems are needed it is for velocity (eg., real-time analytics) or variety (eg., working with images). However, of course, increasingly some businesses really do have large volume requirements (multi-petabyte).

Modern big data systems are driven by real-time event capture: a highly efficient means of storing information in its most complete form (who, what, where, when). This enables high-velocity querying. From these event systems non-traditional data systems receive data (graphs for social network analysis, file systems for archives and images, and so on). This data is duplicated in many places, including within traditional systems, to enable a highly flexible and isolated approach to analytics.

In such modern big data systems data is highly available for all analytical tasks, and captured in a maximally insight-driven way (capturing who/what/where/when enables insight).

What roles enable big data?

Building these systems requires, in order: software engineers, data engineers, data analysis, and data scientists.

Software engineers build the automated systems of data collection (“ingestion”) which provide the raw data from the outside; either from apps in the case of user data, or eg., IoT devices which provide sensor data.

Data Engineers build the pipelines which accept data from these applications, transform it into a usable structure, and determine how and where it will be stored. Their role is to bridge the user-facing world of software to the internal world of analytics by automating and building data systems to enrich and structure data ready for analytics.

Data Analysts are the first fact-finding layer of data recipients. They work with analytical tools (eg., excel) and dashboards (typically built by software engineers). Their job is to summarize and aggregate the relevant datasets they are presented with to answer fact-finding strategic business questions.

Data Scientists are the final hire. They are accelerators across all these prior functions: software engineering, data engineering, analytics & insight. They are able to derive the last-mile high-value from these practice areas. They bring to software engineering, intelligent applications; to data engineering, systems which enrich and structure data for greater insight; to analytics, deep explanatory and predictive projects.

However, a data scientist can, typically, only operate to advance existing systems (or, otherwise, is a very expensive resource to build them with).

A medium business requires perhaps three software teams, one data engineering team, two analytical teams and perhaps two or three data scientists. A large business may want to double or treble these numbers. National organizations may have dedicated data science functions.