Week 2
Data Types and Formats

ISTA 322 - Data Engineering

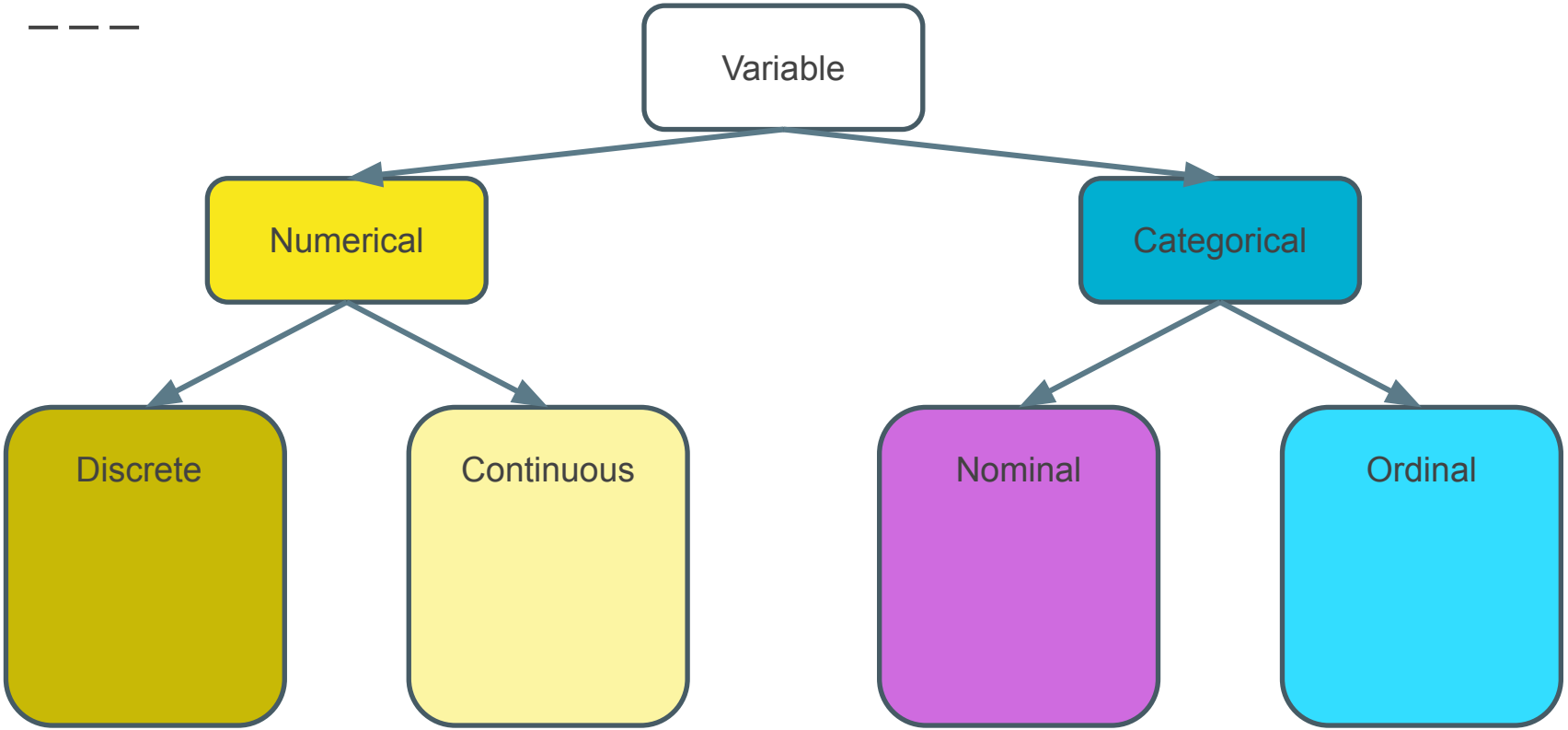
What is data anyway?

- **Data** - Collection of facts consisting of numbers/words that are taken though measurements with the goal of describing things
- Let’s create data on me
  - Get weight - 230lbs
  - Get height - 72”
  - Hair color - brown
- These are measurements that describe me
  - Level of precision is determined by tool used
  - Also at data entry
- Obviously numbers and words

Describing data

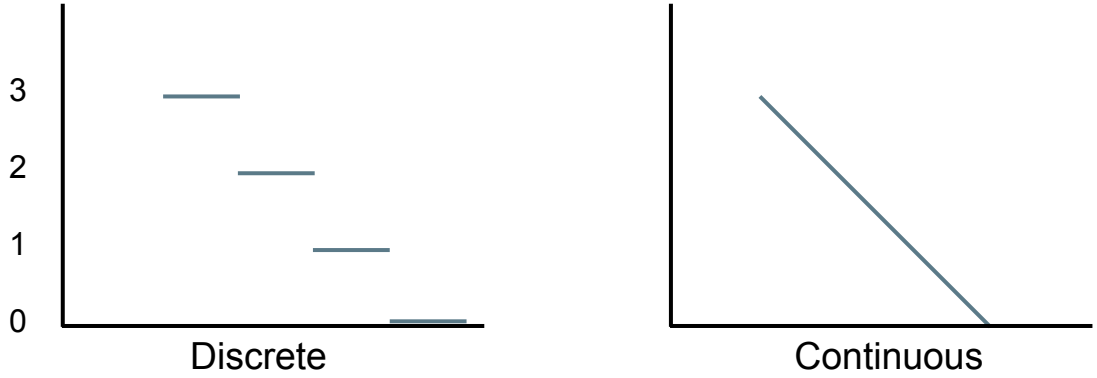
- Should be simple: numbers and words
- But there’s more nuance than that
- Numbers
  - Continuous vs discrete, binary vs non-binary
- Words
  - Nominal vs ordinal, numbers stored as text, text coded as numbers, T/F
- Nuance is important to accurately describe reality
- Practical reasons too (e.g. storage, modeling)

Describing data - Overview



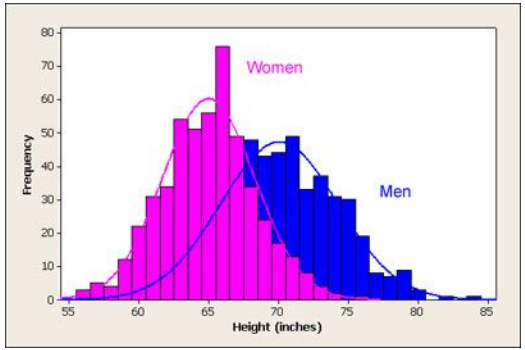
Continuous vs. discrete

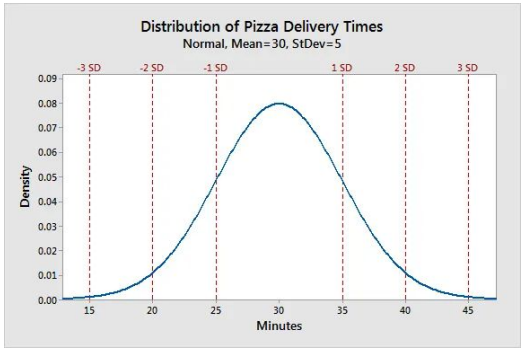
- Useful to think whether data is continuous or discrete
- Discrete - data that can only have certain values
  - Limited amount of ‘in-between’ numbers
- Continuous - data that can take any value (continuum)
  - Any ‘in-between’ number is possible

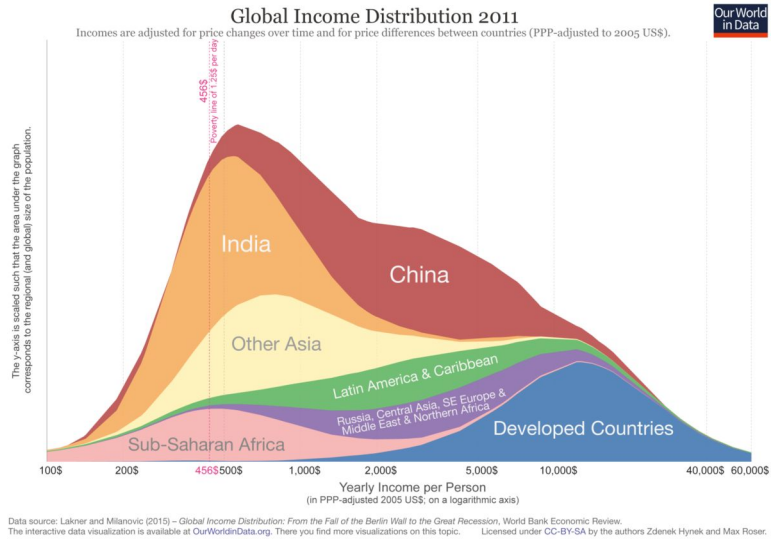


Datatypes - Numeric continuous - Float

- Continuous numeric datatypes you're familiar with
- Height, weight, sales per store, etc
  - Can have any value in a range
  - Have a decimal
- Float datatype

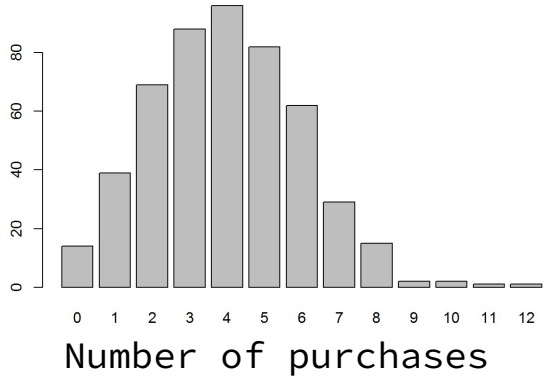






Datatypes - Numeric discrete - Integer

- Number of clicks, number of sales, passengers on a plane
- These are numeric
  - Can have any whole value in a range
- But they’re not continuous
  - Can’t have fractional values
  - No half clicks, half sales, half passengers, etc
- Integer datatype



Datatypes - Numeric discrete - Binary

- Binary datatypes are very common to indicate yes/no or present/absent
- yes = 1, no = 0

purchased	purch_b
yes	1
no	0
no	0
no	0
yes	0
no	0

Datatypes - Numeric discrete - Binary

- Binary datatypes are very common to indicate yes/no or present/absent
- yes = 1, no = 0
- Many ML models need numeric
- Can encode levels of strings
  - One hot encoding
- Integer datatype

OS			
windows			
android			
mac			
mac			
windows			
android			

Datatypes - Numeric discrete - Binary

- Binary datatypes are very common to indicate yes/no or present/absent
- yes = 1, no = 0
- Many ML models need numeric
- Can encode levels of strings
  - One hot encoding
- Integer datatype

OS	win_b	and_b	mac_b
windows	1	0	0
android	0	1	0
mac	0	0	1
mac	0	0	1
windows	1	0	0
android	0	1	0

Nominal vs. ordinal

- Ordinal
  - Values have a hierarchy/relative value to each other
- Nominal
  - Values are independent of each other

Datatypes – Categorical - Strings

- Strings are any length of alphanumeric characters
- Could be nominal – aka text!
  - ‘black’, ‘brown’, ‘blonde’, ‘red’
  - ‘Pineapple on pizza is good, change my mind’
  - ‘Y'all shouldn’t be having parties with 100 people’
  - ‘2013-01-12 07:54:22 __RequestToken_Lw__=2B3CC Channie Tununak’
- Could be ordinal
  - ‘small’, ‘medium’, ‘large’
  - ‘extremely unhappy’, ‘unhappy’, ‘neutral’, ‘happy’, ‘extremely happy’
- String datatype

Datatypes - Categorical - Boolean

- Booleans are a lot like binary
  - Two values, True and False
  - Not strings, though
- True = 1
- False = 0
- Frequently used in logical statements
  - If x is TRUE, then do y
- Boolean datatype

Datatypes - Datetimes

- Time data is really important for rolling up data
  - Sales in a hour/day/month
  - Average monthly temperature
  - Number of requests per minute
- Frequently imported as strings
  - ‘02/18/2020 12:12:47’
- But can’t do critical operations with them as strings
- Casting to datetime allows program to understand these as continuous values
- Datetime datatype
  - Not native in python - need ‘datetime’ module
  - SQL does have them

Datatypes

- The exact syntax and methods you'll use to manipulate these will vary across tools
- And there's more nuance within
- But it's important to think about the data type and if the values make sense for its type
- Again, probably review for many of you
  - But still can't be said enough :)

Moving on to Data Structures

- Data structures are ‘how’ data types are organized
  - Numeric data stored in a list: [67, 49, 88, 95, 77]
  - Numeric data is a data type
  - List is the way it’s stored
- Lists, data frames, and dictionaries
- Most have seen, but we’ll run through them quick

Lists

- Ordered sequence of elements
- Test_scores = [67, 49, 88, 95, 77]
  - print(test_scores)
    - Output: [67, 49, 88, 95, 77]
- Can be updated, added to, sliced, etc using different methods in python
- Can also be strings or mixed
  - Dan_info = [230, 72, ‘brown’]
- []

Dataframe

- Not native to python - Need Pandas
- Just a 2-D object with labeled columns and rows
- Multiple lists bound together
  - Test_scores = [67, 49, 88, 95, 77]
  - Study_time = [35, 14, 75, 89, 68]
  - School_year = [‘fr’, ‘jr’, ‘sr’, ‘sr’, ‘fr’]
- Pandas is very powerful for data wrangling

test_scores	study_time	school_year
67	35	fr
49	14	jr
88	75	sr
95	89	sr
77	68	fr

Dictionary

- Dictionaries are key-value pairs
  - {key:value , key:value}
- Dan_dict = {name: ‘dan’, height: 72, weight: 230}
- Dictionaries can be searched quickly if you know they key
- Can be converted to data frames
- JSON files are organized sets of key-value pairs

Exciting huh?

- OK, not the most thrilling topic, but we need to make sure we’re all on the same page
- Other part of this week will be coding
- But just need to cover a bit on several key data formats
  - Flat files, relational database, json

Flat files

- Flat files refer to 2 dimensional data
  - csv, tsv
- Each row represents an observation
- Only one file containing all info
- Often great for analysis
  - It's a data frame!
- Slower to search
- Inefficient storage

transact_id	store_id	store_state	country	UPC	price
x88943	az_23	AZ	USA	49914	2.57
x88943	az_23	AZ	USA	99371	1.99
a85921	to_39	Ontario	Canada	95831	8.99
a85921	to_39	Ontario	Canada	99492	5.49
a85921	to_39	Ontario	Canada	27482	4.49
z88930	az_45	USA	USA	33491	0.99

Relational database

- Goal will be to put transformed data into relational database
  - The ‘L’ in ETL
- Tables ‘relate’ to one another based on keys
- Efficient
- Can be queried in many ways (e.g. SQL)

TABLE ID: STORE		
store_id	**store_state**	**country**
az_23	AZ	USA
az_45	AZ	USA
ca_12	CA	USA
to_39	Ontario	Canada

TABLE ID: TRANSACTIONS			
transact_id	**store_id**	**UPC**	**price**
x88943	az_23	49914	2.57
x88943	az_23	99371	1.99
a85921	to_39	95831	8.99
a85921	to_39	99492	5.49
a85921	to_39	27482	4.49
z88930	az_45	33491	0.99

JSON

- Semi-structured
- Need to parse and load into DB
- Keys = column names
- Values = observations
- Can then query

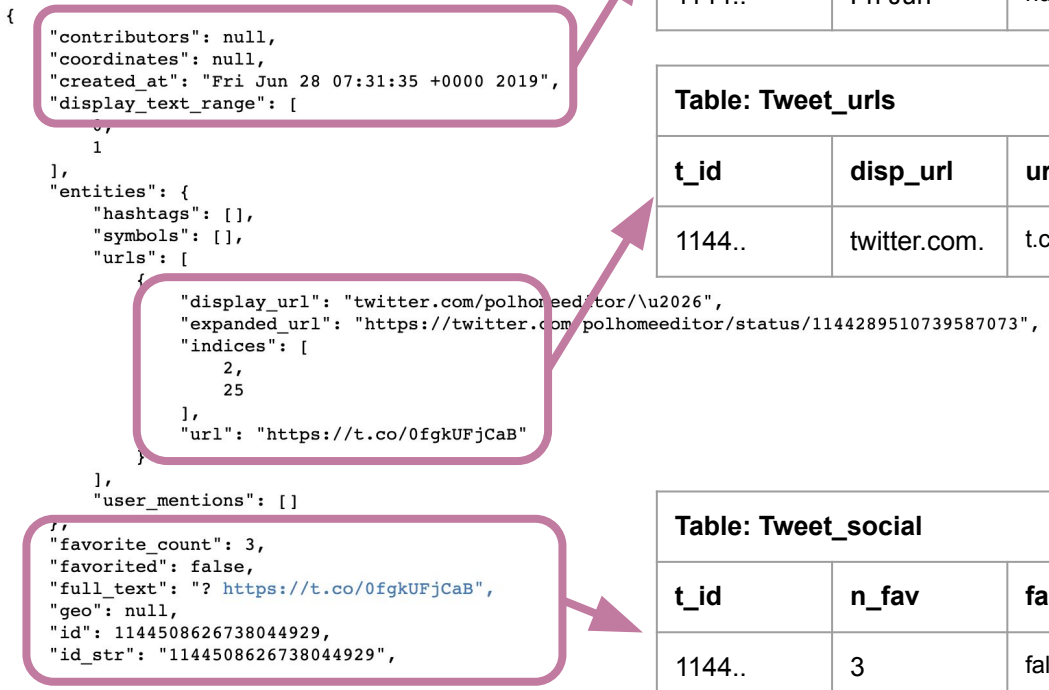


Table: Tweet_info

t_id	t_time	t_coord
1144..	Fri Jun	null

Table: Tweet_urls

t_id	disp_url	url
1144..	twitter.com.	t.co/0f...

Table: Tweet_social

t_id	n_fav	favorited
1144..	3	false

Recap

Need to know data types, put into structures, manipulate

Move into database in organized framework so can be queried and analyzed

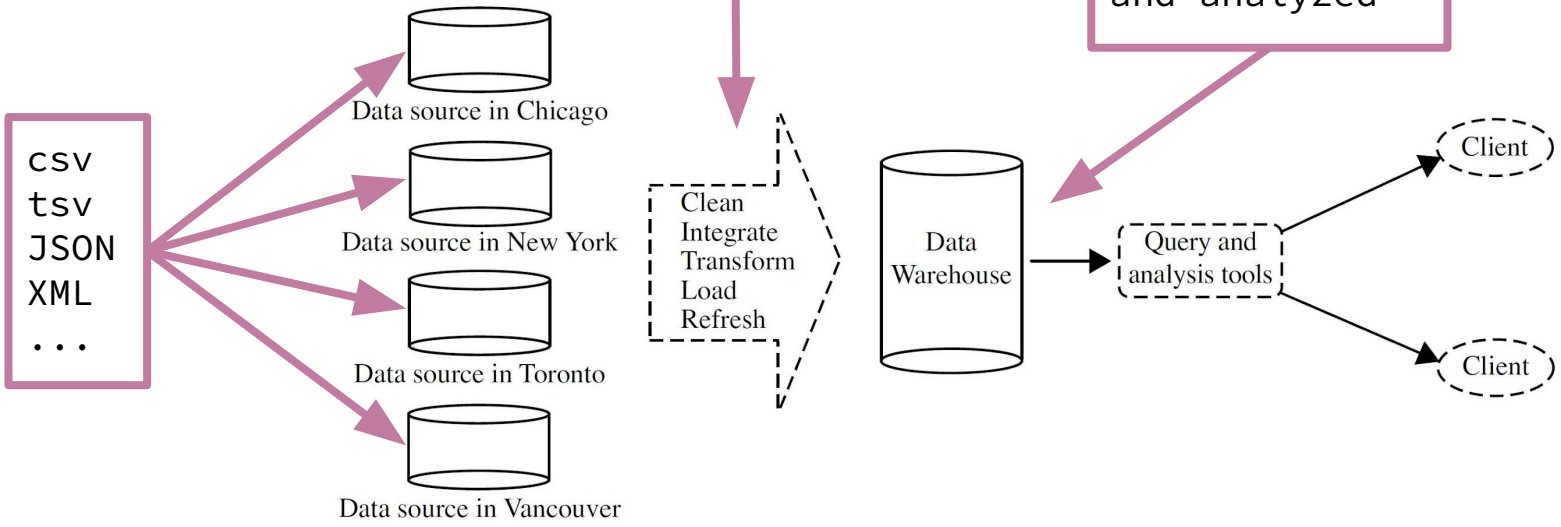


Figure 1.6 Typical framework of a data warehouse for *AllElectronics*.

Recap

- It’s ok if some bits are not clear
  - Just to make sure everybody is on the same general page
- These are more concepts that we will be applying across languages
- Other part of this week (and rest of the class) we’ll be applying these ideas and more.