Data Preparation

CPE 232: Data Models

Dr. Sansiri Tarnpradab

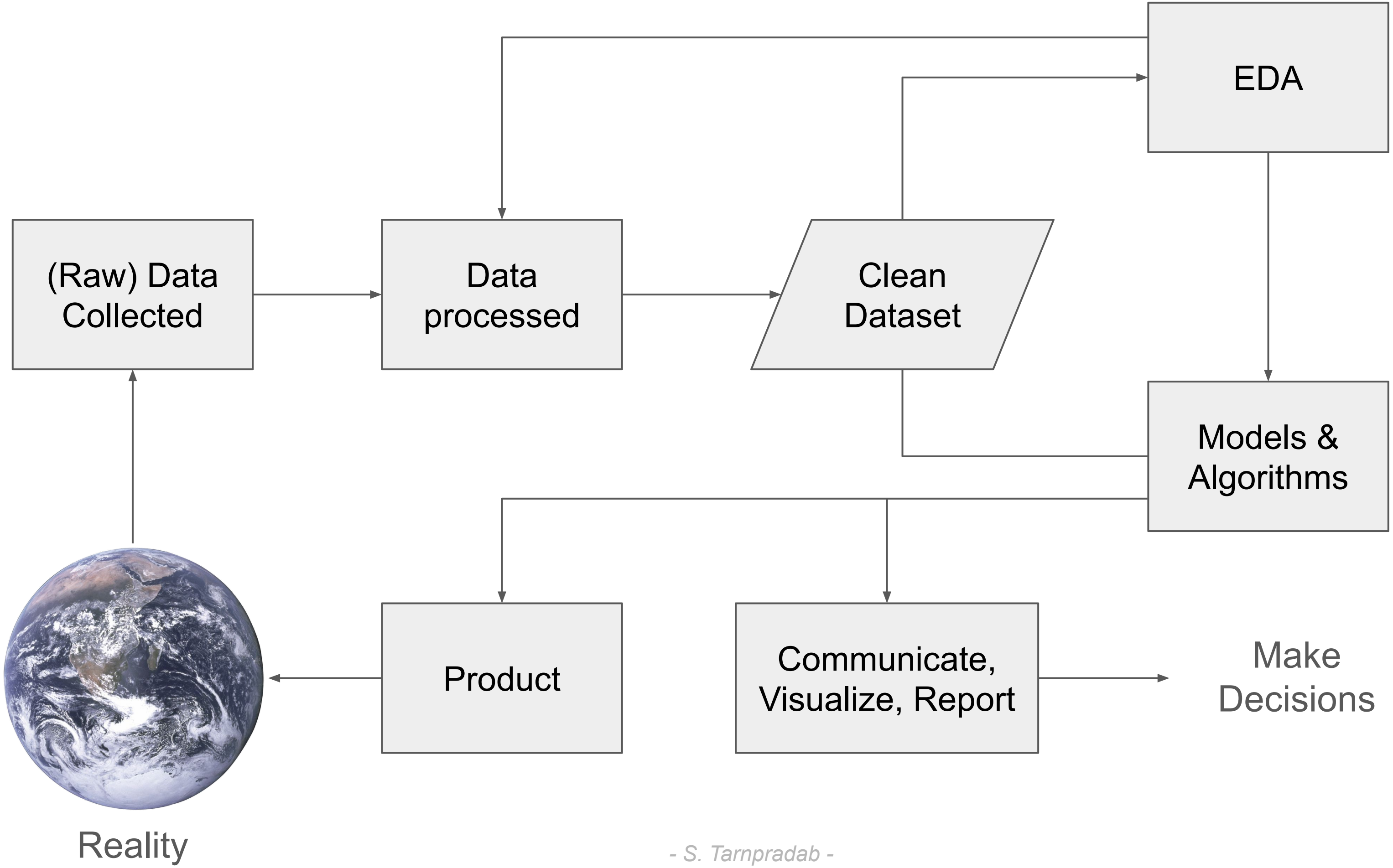
Department of Computer Engineering, KMUTT

Review

Outline

- Data Science Workflow
- Significance of Data Preparation
- Roles in Data Processing
- Data Preprocessing
  - Data Cleaning
  - Data Integration
  - Data Transformation
  - Data Reduction

Data Science Workflow




FOX 35
You know the saying, right?
Garbage In, Garbage Out.
75°
STATION • THE NEWS STATION • THE NE
ORLANDO
THE NEWS STATION
5:46 75°
CLEANING UP AFTER PRESIDENT TRUMP'S VISIT
TWITTER
SPORTS
HICAGO WSOX CHICAGO CUBS (8:05) FOX MLB: CLEVELAND TEXAS (8:05)
MELBOURNE 12am 76° 5am 77° 10am 84°
imgflip.com

Dirty Data: Some Indicating Factors

Incomplete

- Lacking attributes of interest
- Lacking attribute values
- Attributes contain only aggregate data

Noisy

- Contain errors or outliers
- Extreme values can severely affect the dataset's range.

Inconsistent

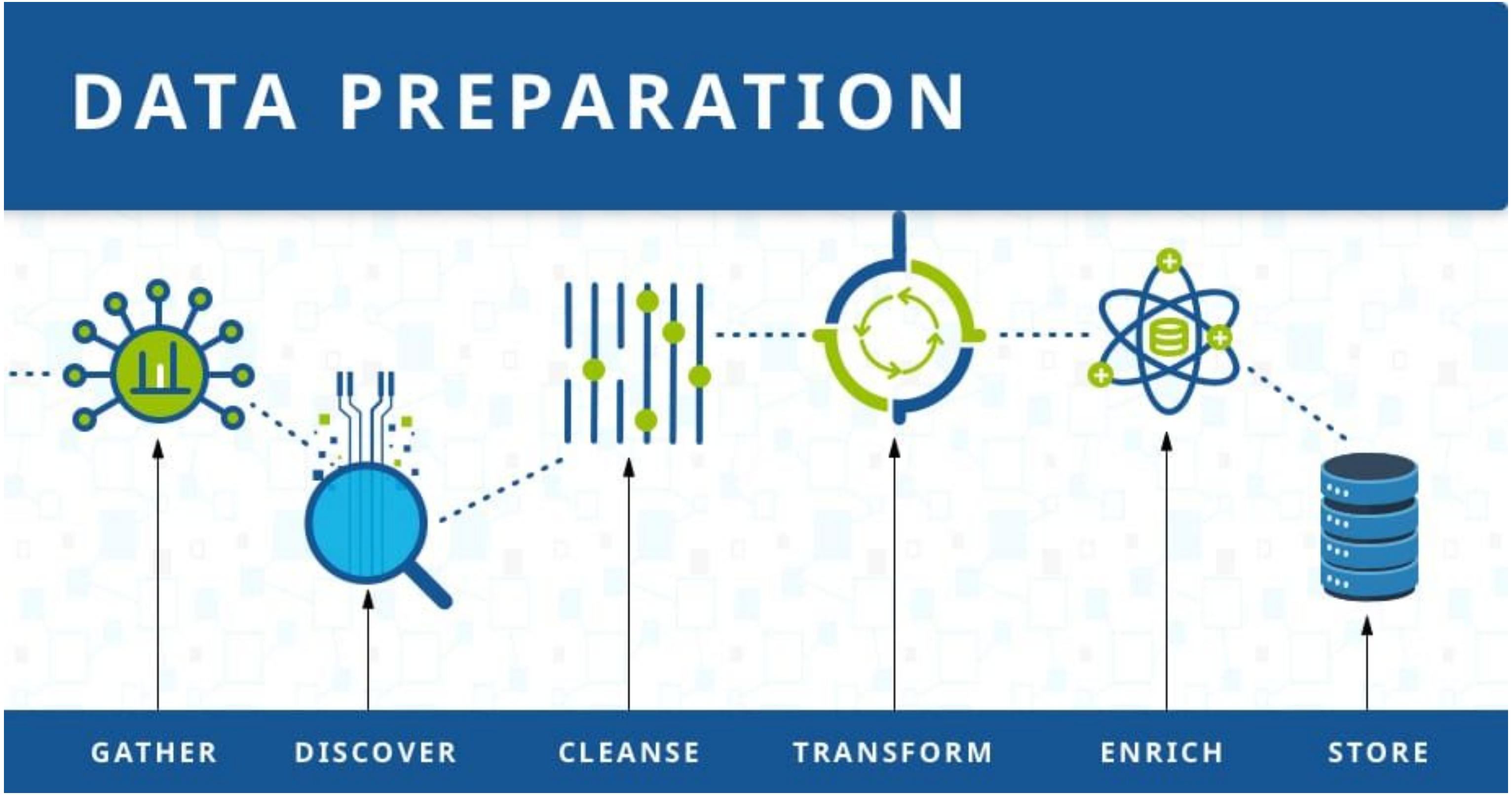
- contain discrepancies in codes or names
- "Name" column contains values other than alphabetical letters.
- Records do not start with a capital letter.

Dirty Data: Examples

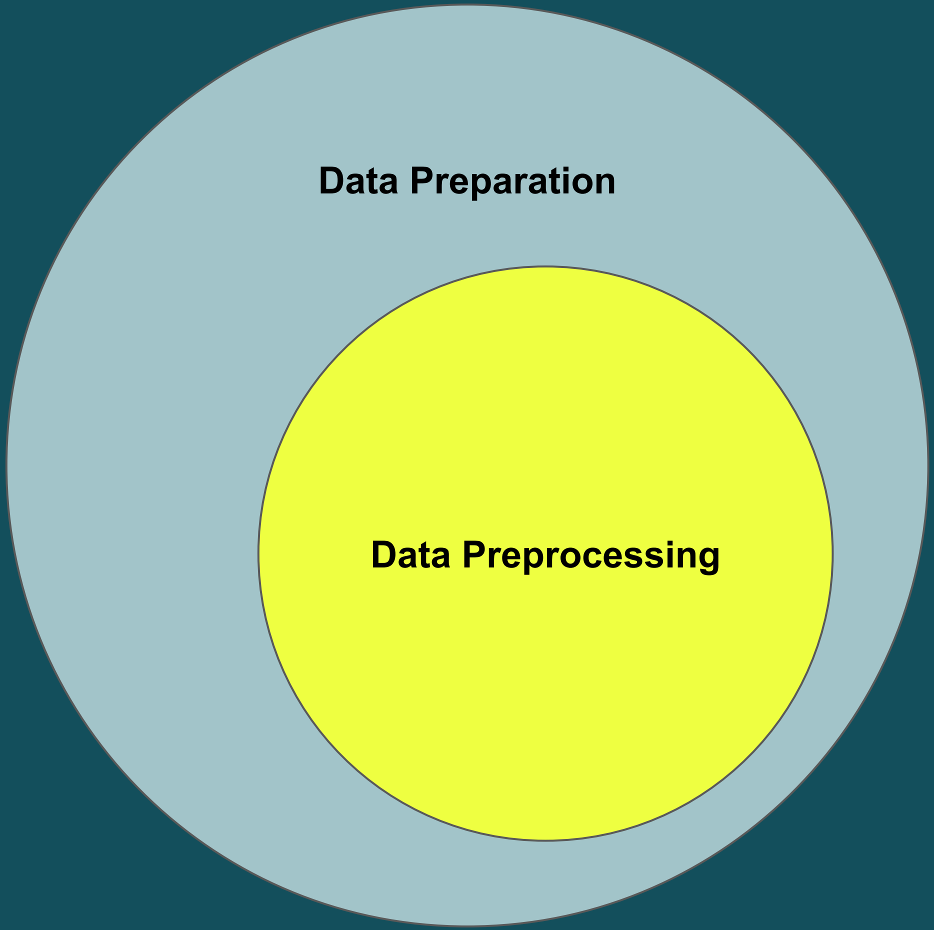
Causes	Results
Data Entry Errors	Typos, misspellings, incorrect values, duplicates
Missing Values	Data fields left blank or not collected (incomplete data)
Inconsistent Formatting	Differences in units, date formats, or other data formats.
Outliers	Errors in measurement/recording due to unusual or extreme values
Data Integration	Mismatched/incompatible data types due to a merge from different sources
Data Storage & Transfer	Loss of data integrity due to corruption during storage/transfer
Data Aging	Data becoming outdated over time
Security Issues	Unauthorized access → compromised data integrity

Preprocessed

Reg images:
https://www.freepik.com/premium-vector/hand-drawn-garbage-cartoon-vector-illustration-clipart-white-background_151608816.htm
https://www.teraflow.ai/3-big-benefits-of-data-cleansing/



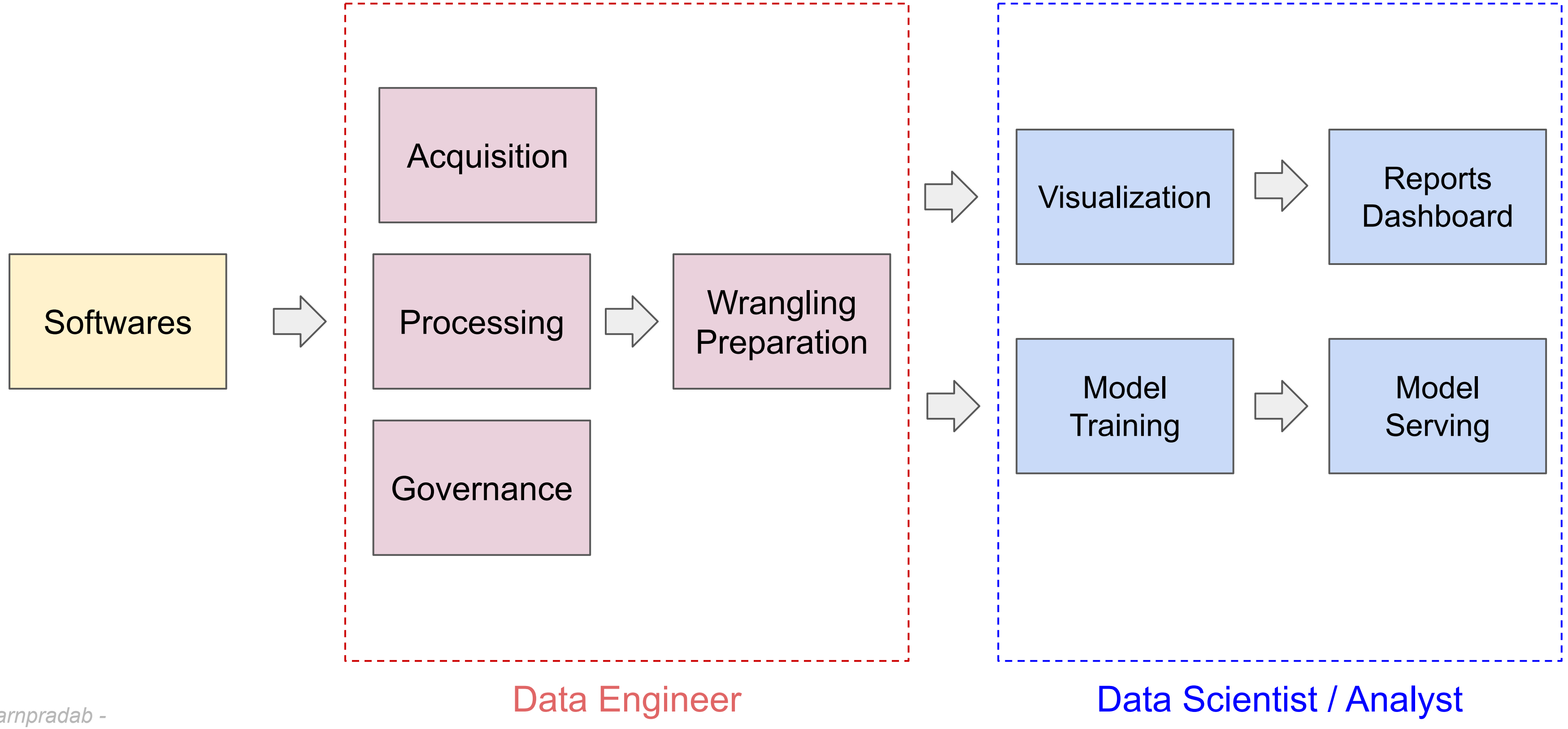
Ref: https://devopedia.org/data-preparation



Data Preparation is *a broader process* that ensures data is clean and structured for analysis.

Data Preprocessing is a technical step *within data preparation*, specifically for machine learning tasks.

Different Roles



Data
Cleaning

Data
Integration

Data Preprocessing

Data
Transformation

Data
Reduction

Data Cleaning

Clean(ed) Data

- Accuracy
- Validity
- Reliability
- Timeliness
- Relevance
- Completeness
- Compliance

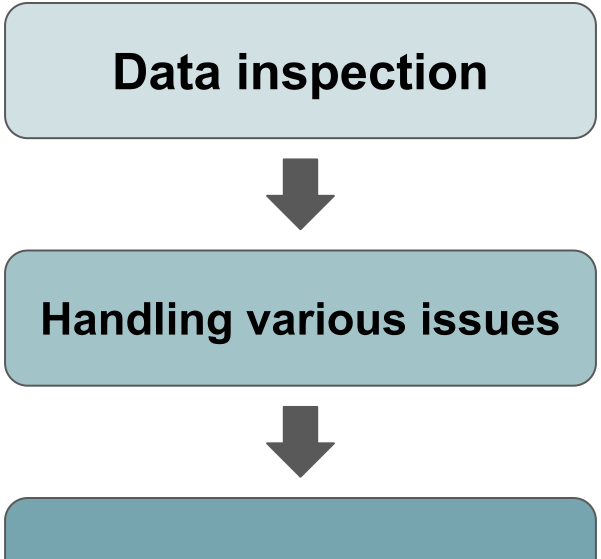
Ref: https://datasciencedojo.com/blog/data-science-memes/

Property	(Bad) Example	Issue
Accuracy *(Incorrect Data)*		
Validity *(Violates Constraints or Formats)*		
Reliability *(Inconsistent Data Across Sources)*		
Timeliness *(Outdated Data)*		

Property	(Bad) Example	Issue
Relevance *(Irrelevant or Unnecessary Data)*		
Completeness *(Missing Critical Data)*		
Compliance *(Violates Regulations or Policies)*	A company stores customers' credit card	Non-compliance can lead to

Data Cleaning

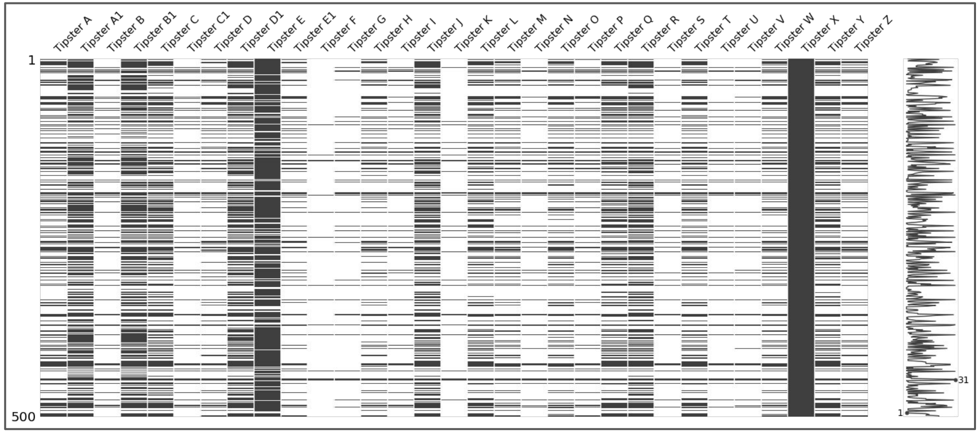
- A fundamental step in the data preparation process
- Contributes to data quality, accuracy, and the overall effectiveness of data-driven decision-making processes within organizations



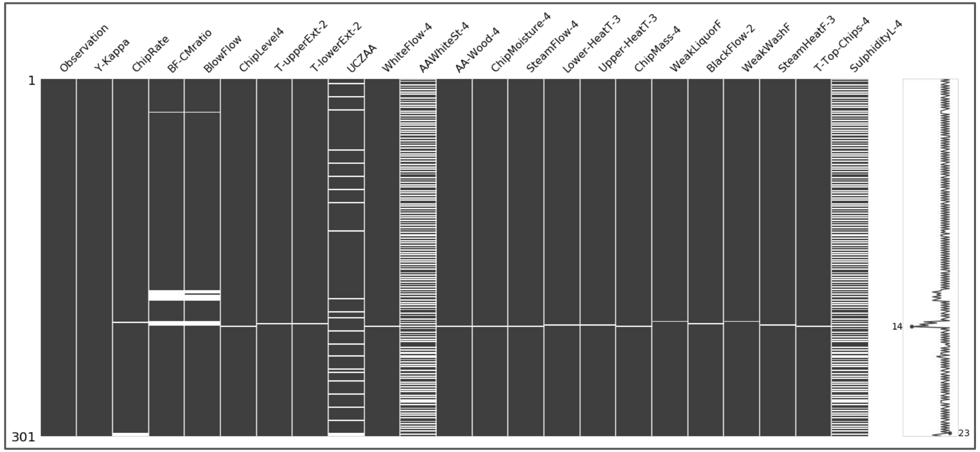
Inspecting Data

- Aka visualizing data
- To develop a comprehensive understanding of the dataset
- To identify missing values, duplicates, and anomalies

E.g. `pip install missingno`



Ref: https://www.kaggle.com/code/residentmario/using-missingno-to-diagnose-data-sparsity



Ref: https://www.geeksforgeeks.org/python-visualize-missing-values-nan-values-using-missingno-library/

Handling Missing Values

- Identify if there is any and for which attribute
- Investigate why they are missing
- Select a proper method to address the issue
- Perform correction

Before cleaning!

Ask questions:

- **What are the features?**
  - Column names → self-explanatory
  - E.g. ZIP_CD, ST_NAME, OWNED, NUM_BEDROOMS
- **What are the expected data types?**
  - int, float, string, boolean, etc.
  - ZIP_CD (int), ST_NAME (string), OWNED (boolean), NUM_BEDROOMS (int)
- **Any missing data detectable?**

Let's see an example ⇨

Shape = (327346, 20)

flights

	year	month	day	dep_time	sched_dep_time	dep_delay	arr_time	sched_arr_time	arr_delay	carrier	flight	tailnum	origin	dest	air_time	distance	hour	minute	time_hour
0	2013	1	1	517	515		830	819	11	UA	1545	N14228	EWR	IAH	227	1400	5	15	2013-01-01T05:00:00Z
1	2013	1	1	533	529	4.0	850	830	20	UA	1714	N24211	LGA	IAH	227	1416	5	29	2013-01-01T05:00:00Z
2	2013	1	1	542	540		923	850	33	AA	1141	N619AA	JFK	MIA	160	1089	5	40	2013-01-01T05:00:00Z
3	2013	1	1	544	545	-1.0	1004	1022	-18	B6	725	N804JB	JFK	BQN	183	1576	5	45	2013-01-01T05:00:00Z
4	2013	1	1	554	600	-6.0	812	837	-25	DL	461	N668DN	LGA	ATL	116	762	6	0	2013-01-01T06:00:00Z
5	2013	1	1	554	558	-4.0	740	728	12	UA	1696	N39463	EWR	ORD	150	719	5	58	2013-01-01T05:00:00Z
6	2013	1	1	555	600	-5.0	913	854	19	B6	507	N516JB	EWR	FLL	158	1065	6	0	2013-01-01T06:00:00Z
7	2013	1	1	557	600	-3.0	709	723	-14	EV	5708	N829AS	LGA	IAD	53	229	6	0	2013-01-01T06:00:00Z
8	2013	1	1	557	600	-3.0	838	846	-8	B6	79	N593JB	JFK	MCO	140	944	6	0	2013-01-01T06:00:00Z
9	2013	1	1	558	600	-2.0	753	745	8	AA	301	N3ALAA	LGA	ORD	138	733	6	0	2013-01-01T06:00:00Z
10	2013	1	1	558	600	-2.0	849	851	-2	B6	49	N793JB	JFK	PBI	149	1028	6	0	2013-01-01T06:00:00Z
11	2013	1	1	558	600	-2.0	853	856	-3	B6	71	N657JB	JFK	TPA	158	1005	6	0	2013-01-01T06:00:00Z
12	2013	1	1	558	600	-2.0	924	917	7	UA	194	N29129	JFK	LAX	345	2475	6	0	2013-01-01T06:00:00Z
13	2013	1	1	558	600	-2.0	923	937	-14	UA	1124	N53441	EWR	SFO	361	2565	6	0	2013-01-01T06:00:00Z
14	2013	1	1	559	600	-1.0	941	910	31	AA	707	N3DUAA	LGA	DFW	257	1389	6	0	2013-01-01T06:00:00Z
15	2013	1	1	559	559	0.0	702	706	-4	B6	1806	N708JB	JFK	BOS	44	187	5	59	2013-01-01T05:00:00Z
16	2013	1	1	559	600	-1.0	854	902	-8	UA	1187	N76515	EWR	LAS	337	2227	6	0	2013-01-01T06:00:00Z
17	2013	1	1	600	600	0.0	851	858	-7	B6	371	N595JB	LGA	FLL	152	1076	6	0	2013-01-01T06:00:00Z
18	2013	1	1	600	600	0.0	837	825	12	MQ	4650	N542MQ	LGA	ATL	134	762	6	0	2013-01-01T06:00:00Z
19	2013	1	1	601	600	1.0	844	850	-6	B6	343	N644JB	EWR	PBI	147	1023	6	0	2013-01-01T06:00:00Z
20	2013	1	1	602	610	-8.0	812	820	-8	DL	1919	N971DL	LGA	MSP	170	1020	6	10	2013-01-01T06:00:00Z
21	2013	1	1	602	605	-3.0	821	805	16	MQ	4401	N730MQ	LGA	DTW	105	502	6	5	2013-01-01T06:00:00Z
22	2013	1	1	606	610	-4.0	858	910	-12	AA	1895	N633AA	EWR	MIA	152	1085	6	10	2013-01-01T06:00:00Z
23	2013	1	1	606	610	-4.0	837	845	-8	DL	1743	N3739P	JFK	ATL	128	760	6	10	2013-01-01T06:00:00Z

```
[ ] # Read flight data
    flightData = pd.read_csv('[redacted]flights.csv')
    flightData.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 327346 entries, 0 to 327345
Data columns (total 20 columns):
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   Unnamed: 0      327346 non-null  int64  
 1   year            327346 non-null  int64  
 2   month           327346 non-null  int64  
 3   day             327346 non-null  int64  
 4   dep_time        327346 non-null  int64  
 5   sched_dep_time  327346 non-null  int64  
 6   dep_delay       321132 non-null  float64
 7   arr_time        327346 non-null  int64  
 8   sched_arr_time  327346 non-null  int64  
 9   arr_delay       327346 non-null  int64  
 10  carrier         327346 non-null  object 
 11  flight          327346 non-null  int64  
 12  tailnum         327346 non-null  object 
 13  origin          327346 non-null  object 
 14  dest            327346 non-null  object 
 15  air_time        327346 non-null  int64  
 16  distance        327346 non-null  int64  
 17  hour            327346 non-null  int64  
 18  minute          327346 non-null  int64  
 19  time_hour       327346 non-null  object 
dtypes: float64(1), int64(14), object(5)
memory usage: 49.9+ MB
```

Before cleaning!

Ask questions:

- What are the features?
- What are the expected data types?

```python
# Check if there's any missing data
flightData.isnull().any()
```

```
Unnamed: 0        False
year              False
month             False
day               False
dep_time          False
sched_dep_time    False
dep_delay          True
arr_time          False
sched_arr_time    False
arr_delay         False
carrier           False
flight            False
tailnum           False
origin            False
dest              False
air_time          False
distance          False
hour              False
minute            False
time_hour         False
dtype: bool
```

Next question:

Any missing data detectable?

```
[ ] # Show rows with missing data
    print (flightData[flightData.dep_delay.isnull()])
```

```
        Unnamed: 0  year  month  day  dep_time  sched_dep_time  dep_delay  \
0                0  2013      1    1       517             515        NaN
2                2  2013      1    1       542             540        NaN
69              69  2013      1    1       702             700        NaN
73              73  2013      1    1       715             713        NaN
98              98  2013      1    1       752             750        NaN
...            ...   ...    ...  ...       ...             ...        ...
326863      326863  2013      9   30      1357            1355        NaN
327005      327005  2013      9   30      1610            1608        NaN
327020      327020  2013      9   30      1621            1619        NaN
327065      327065  2013      9   30      1702            1700        NaN
327102      327102  2013      9   30      1731            1729        NaN

        arr_time  sched_arr_time  arr_delay carrier  flight tailnum origin  \
0            830             819         11      UA    1545  N14228    EWR
2            923             850         33      AA    1141  N619AA    JFK
69          1058            1014         44      B6     671  N779JB    JFK
73           911             850         21      UA     544  N841UA    EWR
98          1025            1029         -4      UA     477  N511UA    LGA
...          ...             ...        ...     ...     ...     ...    ...
326863      1547            1615        -28      WN     246  N430WN    EWR
327005      1729            1752        -23      B6    1105  N306JB    JFK
327020      1856            1919        -23      B6     283  N632JB    JFK
327065      1940            1921         19      DL    2042  N346NB    EWR
327102      2008            2030        -22      UA    1692  N36472    EWR

       dest  air_time  distance  hour  minute             time_hour
0       IAH       227      1400     5      15  2013-01-01T05:00:00Z
2       MIA       160      1089     5      40  2013-01-01T05:00:00Z
69      LAX       381      2475     7       0  2013-01-01T07:00:00Z
73      ORD       156       719     7      13  2013-01-01T07:00:00Z
98      DEN       249      1620     7      50  2013-01-01T07:00:00Z
...     ...       ...       ...   ...     ...                   ...
326863  PHX       267      2133    13      55  2013-09-30T13:00:00Z
327005  ORD       111       740    16       8  2013-09-30T16:00:00Z
327020  MCO       129       944    16      19  2013-09-30T16:00:00Z
327065  ATL        99       746    17       0  2013-09-30T17:00:00Z
327102  SAN       302      2425    17      29  2013-09-30T17:00:00Z

[6214 rows x 20 columns]
```

Shape of all records of which dep_delay is null = (6214, 20)

Now what do we do?

- $\boxed{\text{Remove rows}}$
- Remove columns
- Fill with zeros
- Fill with some values
  (What values then?)

```python
# index of missing data
index_nan = flightData.dep_delay.index[flightData.dep_delay.isnull()]
print (index_nan)
```

```
Int64Index([     0,      2,     69,     73,     98,    185,    200,    236,
               245,    325,
            ...
            326581, 326651, 326660, 326741, 326840, 326863, 327005, 327020,
            327065, 327102],
           dtype='int64', length=6214)
```

Get index of missing data

Shape = (327346, 20) - (6214, 20) → (321132, 20)

```python
flightData_1 = flightData.dropna(how ='any')
flightData_1.isnull().any()
```

```
Unnamed: 0        False
year              False
month             False
day               False
dep_time          False
sched_dep_time    False
dep_delay         False
arr_time          False
sched_arr_time    False
arr_delay         False
carrier           False
flight            False
tailnum           False
origin            False
dest              False
air_time          False
distance          False
hour              False
minute            False
time_hour         False
dtype: bool
```

Drop them

```python
# Compute mean
x = np.mean(flightData.dep_delay)
print("%1.1f"%x)
```

```
12.8
```

Now what do we do?

- Remove rows
- Remove columns
- Fill with zeros
- Fill with some values

(What values then?)

```python
index_nan = flightData.dep_delay.index[flightData.dep_delay.isnull()]
print(flightData.fillna(value={'dep_delay':x}))
```

```
        Unnamed: 0  year  month  day  dep_time  sched_dep_time  dep_delay  \
0                0  2013      1    1       517             515  12.759401
1                1  2013      1    1       533             529   4.000000
2                2  2013      1    1       542             540  12.759401
3                3  2013      1    1       544             545  -1.000000
4                4  2013      1    1       554             600  -6.000000
...            ...   ...    ...  ...       ...             ...        ...
327341      327341  2013      9   30      2240            2245  -5.000000
327342      327342  2013      9   30      2240            2250 -10.000000
327343      327343  2013      9   30      2241            2246  -5.000000
327344      327344  2013      9   30      2307            2255  12.000000
327345      327345  2013      9   30      2349            2359 -10.000000

        arr_time  sched_arr_time  arr_delay carrier  flight tailnum origin  \
0            830             819         11      UA    1545  N14228    EWR
1            850             830         20      UA    1714  N24211    LGA
2            923             850         33      AA    1141  N619AA    JFK
3           1004            1022        -18      B6     725  N804JB    JFK
4            812             837        -25      DL     461  N668DN    LGA
...          ...             ...        ...     ...     ...     ...    ...
327341      2334            2351        -17      B6    1816  N354JB    JFK
327342      2347               7        -20      B6    2002  N281JB    JFK
327343      2345               1        -16      B6     486  N346JB    JFK
327344      2359            2358          1      B6     718  N565JB    JFK
327345       325             350        -25      B6     745  N516JB    JFK
```

Data Imputation:
the process of replacing missing data with substituted values

Handling Duplicates

- First, identify duplicates

- Remove them
  - **Row-based**: if the entire record is duplicated (straightforward)

```
df_no_duplicates = df.drop_duplicates()
```

  - **Column-based**: specify column to remove

```
df_no_duplicates = df.drop_duplicates(subset=['column1', 'column2'])

df['is_duplicate'] = df.duplicated(subset=['column1', 'column2']) //flag
```

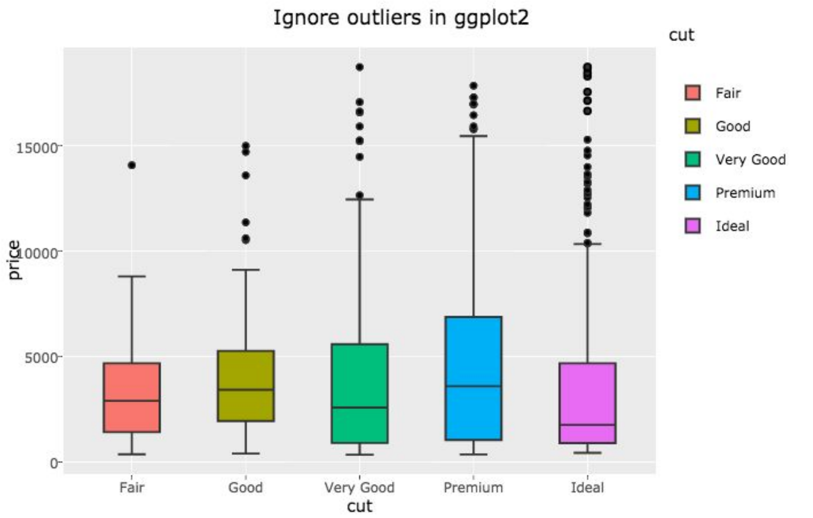
Handling Anomalies

What is Anomaly?

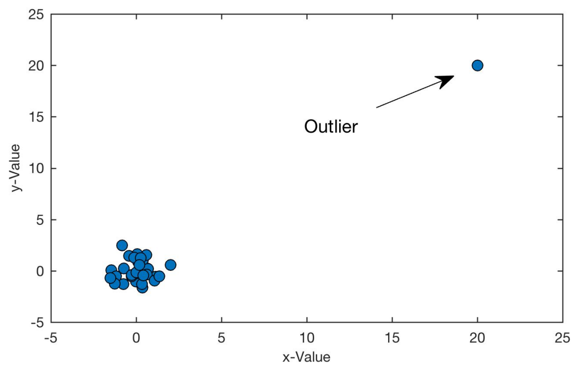
- An occurrence where a data point is exceptionally different from the main distribution

- May cause problems in visualization and modeling

- Can be detected by distribution plots



Ref: https://stackoverflow.com/questions/43499229/removing-outliers-from-boxplot-and-plotly



Ref: https://medium.com/analytics-vidhya/outlier-detection-in-machine-learning-382557c775aa

Things could be more challenging

The data is not in a format that is easy to work with…

- Stored or presented in a way that is hard to process
- Need to convert it to computer-friendly format

> **“Add two diced tomatoes, three cloves of garlic, and a pinch of salt in the mix.”**

Data Munging

Convert into computer-friendly format (manually, automatically, semi-automatically)

- Munging
- Manipulating
- Wrangling

> “Add two diced tomatoes, three cloves of garlic, and a pinch of salt in the mix.”

Data Integration

Data Integration

- Gathering data from various sources
- **Steps:** *(roughly)*
  - Combine data from multiple sources into a coherent storage place (e.g. a single file/database)
  - Engage in schema integration
  - Detect and resolve data value conflicts
  - Address redundant data
- **Common Challenges:**
  - Inconsistencies
  - Duplication
  - Schema mismatches
  - Scalability

Ref: https://www.cloverdx.com/explore/data-integration

Types of Data Integration

Manual

Collected and merged manually

E.g. Copying and pasting data from multiple spreadsheets into one.

Uniform Data Access (UDA)

Data remains in original locations but is accessed through a unified interface.

E.g. A BI dashboard that queries multiple databases without merging them.

Application-Based

Software applications connect and exchange data automatically.

E.g. CRM software syncing with an e-commerce platform.

ETL & Data Warehousing

Data is extracted, transformed, and loaded (ETL) into a central repository.

E.g. A data warehouse like Amazon Redshift consolidating customer data.

Middleware-Based

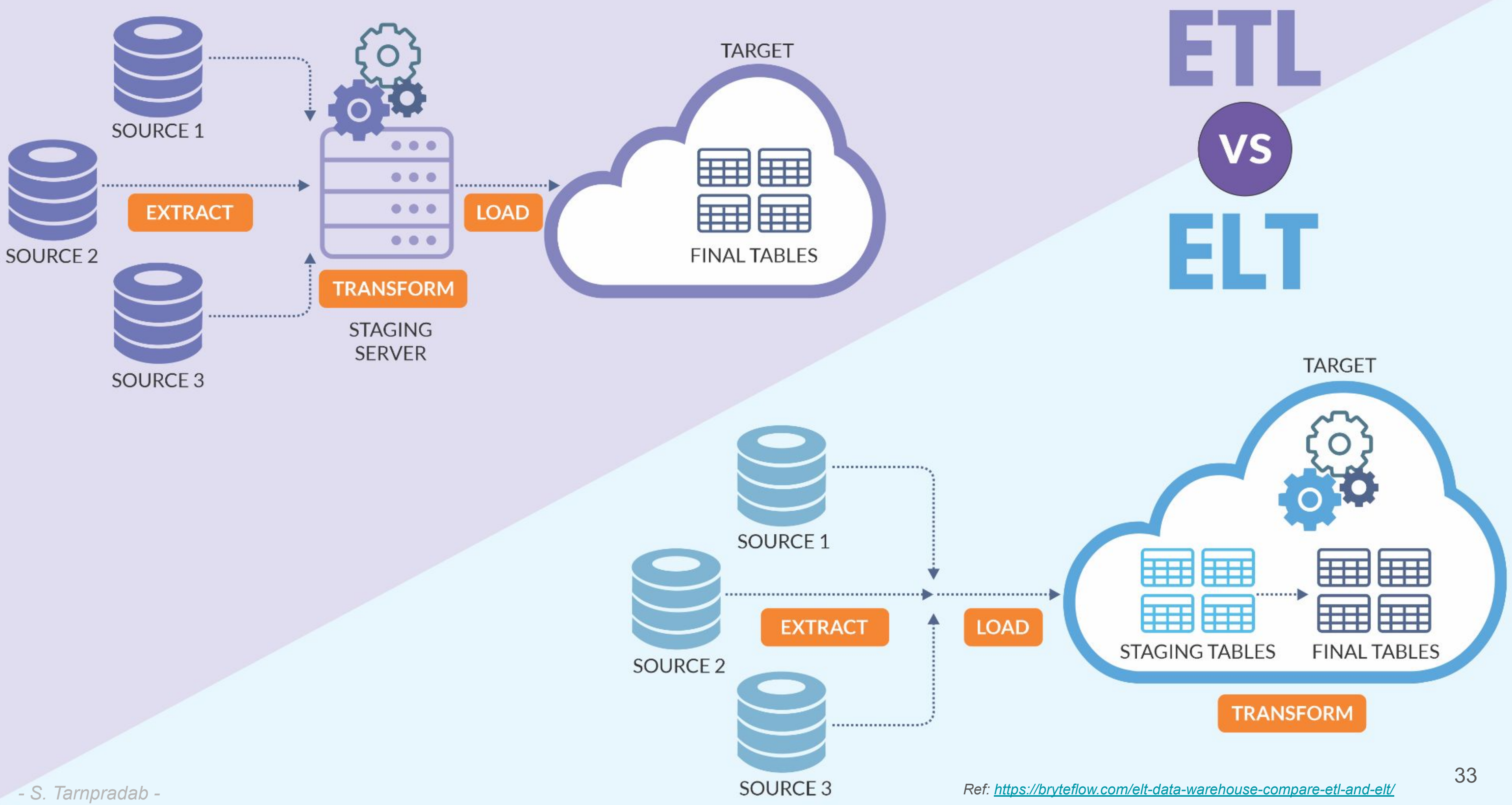
A middleware system acts as a bridge between databases and applications.

E.g. Apache Kafka facilitating real-time data exchange between different platforms.

API-Based

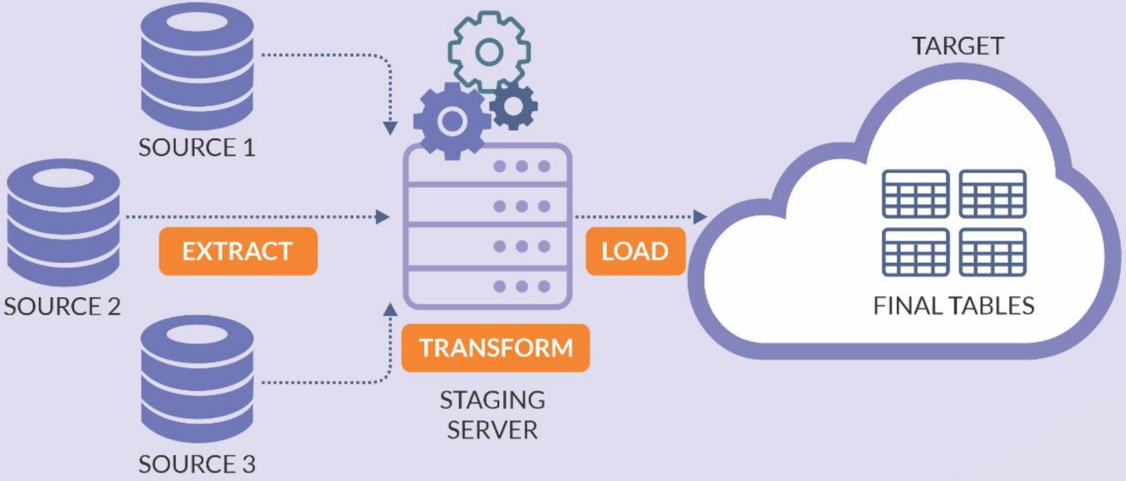
APIs enable seamless data exchange between systems.

E.g. Google Maps API integrating location data into a travel app.



Ref: https://bryteflow.com/elt-data-warehouse-compare-etl-and-elt/

ETL



Process Order:
Extract → Transform → Load

Transformation Location:
In a staging area before loading into the target system

Best for:
Traditional data warehouses where structured data is required

Flexibility:
Less flexible; predefined transformations are required

Ref: https://bryteflow.com/elt-data-warehouse-compare-etl-and-elt/

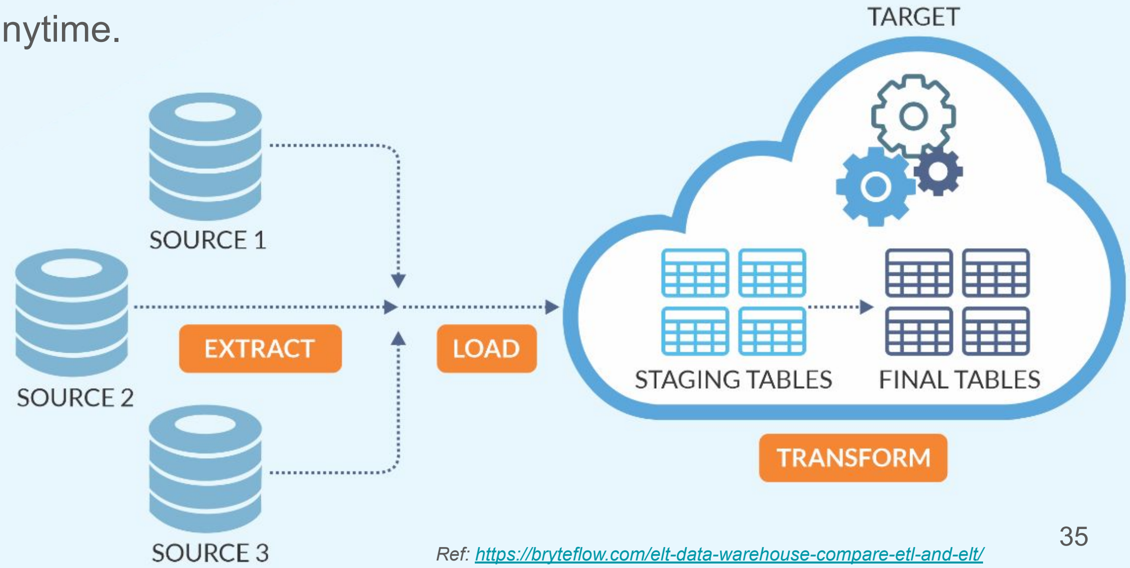
Process Order:
Extract → Load → Transform

Transformation Location:
Data is loaded into a data warehouse or data lake first, then transformed as needed.

Best for:
Big data and cloud-based data lakes where large amounts of raw data are stored.

Flexibility:
More flexible; raw data can be transformed anytime.

ELT



Ref: https://bryteflow.com/elt-data-warehouse-compare-etl-and-elt/

Data Transformation

Data Transformation

Transformation → modifying or converting the structure of data to be more suitable for analysis or modeling.

For example:

Log Transformation
Feature Scaling
Binning
Pivoting
Data Splitting
Data Validation
Feature Engineering

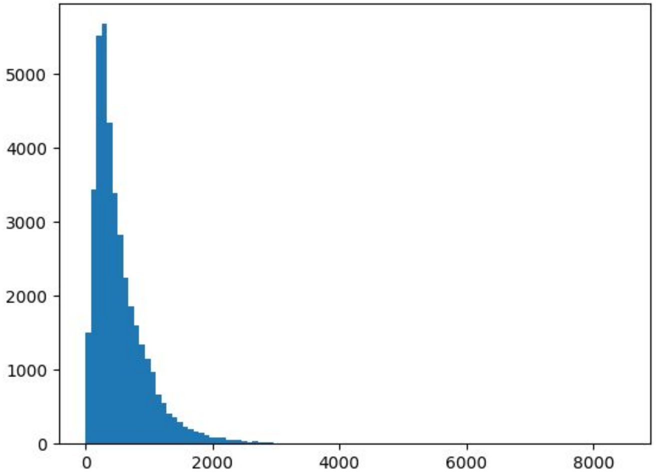
Machine Learning

Log Transformation

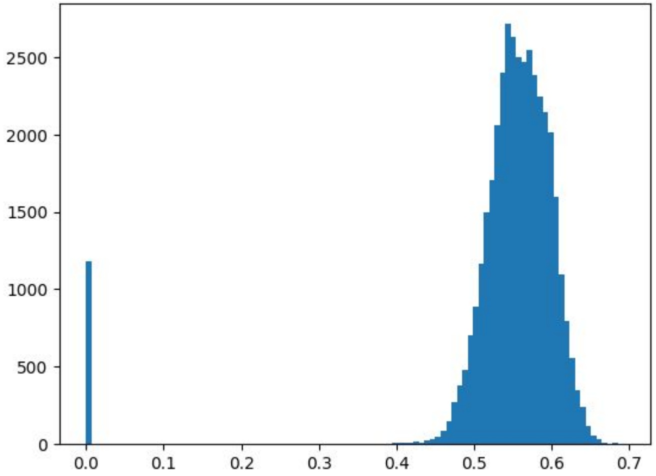
```python
import matplotlib.pyplot as plt

# plotting a histogram
plt.hist(df['n_tokens_content'], bins=100)
plt.show()
```

```python
df['n_tokens_content'] = np.log10(df['n_tokens_content']+1)
plt.hist(df['n_tokens_content'], bins=100)
plt.show()
```



Before Log Transformation



After Log Transformation

Feature Scaling

Normalization

- Min-Max Scaling
- Ensuring that all variables have the same scale (e.g., [0, 1])

Standardization

$$z = \frac{x - \mu}{\sigma}$$

- Z-score Scaling
- Standardize the data to have a mean of 0 and a standard deviation of 1

Binning

Converting numerical variables into discrete bins or categories

```
small = np.random.randint(0, 100, 20) # 20 random integers generated in range 0 to 99
small
```

✓ 0.2s

```
array([79, 12, 67,  0, 60, 92, 28, 48, 97, 18,  0, 32, 28, 92,  6, 60, 46,
       79, 84, 79])
```

```
np.floor_divide(small, 10)
```

✓ 0.5s

```
array([7, 1, 6, 0, 6, 9, 2, 4, 9, 1, 0, 3, 2, 9, 0, 6, 4, 7, 8, 7])
```

Feature Engineering

Generate new features through mathematical operations, aggregations, or combinations of existing features.

Detect Rumors Using Time Series of Social Context Information on Microblogging Websites

Jing Ma[1,3] Wei Gao[2] Zhongyu Wei[4] Yueming Lu[1] Kam-Fai Wong[3,5]

[1]Beijing University of Posts and Telecommunications, Beijing, China
[2]Qatar Computing Research Institute, Hamad Bin Khalifa University, Doha, Qatar
[3]Dept. of SEEM, The Chinese University of Hong Kong, Hong Kong
[4]Computer Science Department, The University of Texas at Dallas, Texas 75080, USA
[5]MoE Key Laboratory of High Confidence Software Technologies, China
{majing,kfwong}@se.cuhk.edu.hk, wgao@qf.org.qa, zywei@hlt.utdallas.edu, ymlu@bupt.edu.cn

Average sentiment score: Similar feature but not the same was used in [2, 10]. Given a sentiment lexicon and an emoticon lexicon, the average sentiment score of microblogs in a time span of event E_i is calculated as:

$$\frac{1}{|m_i|}\sum_{j=1}^{|m_i|}(|w_{pos}|_{ij} - |w_{neg}|_{ij} + |e_{pos}|_{ij} - |e_{neg}|_{ij})$$

Table 1: Description of features $f_{t,k}$ on microblogs from time 0 to time interval t of an event

Content-based features
LDA-based topic distribution of microblogs with 18 topics [10]
Average length of microblogs [2]
# of positive (negative) words in microblogs [2]
Average sentiment score of microblogs [2, 10]
% of microblogs with URL [2, 10, 11]
% of microblogs with smiling (frowning) emoticons [2]
% of positive (negative) microblogs [2]
% of microblogs with the first-person pronouns [2]
% of microblogs with hashtags [2, 11]
% of microblogs with @ mentions [2]
% of microblogs with question marks [2]
% of microblogs with exclamation marks [2]
% of microblogs with multiple question/exclamation marks [2]
User-based features
% of users that provide personal description [2, 10, 11]
% of users that provide personal picture in profile
% of verified users [2, 10, 11]
% of verified users of each type, e.g., celebrities [10, 11]
% of male (female) users [10, 11]
% of users located in large (small) cities
Average # of friends of users [2, 10, 11]
Average # of followers of users [2, 10, 11]
Average # of posts of users [2, 10, 11]
Average days users' accounts exist since registration [2, 10, 11]
Average reputation score of users (i.e., followers/followees ratio)
Diffusion-based features
Average # of retweets [2, 10, 11]
Average # of comments for Weibo posts [10, 11]
# of microblogs [2]

Ref: Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai Wong. Detect rumors using time series of social context information on microblogging websites. In Proceedings of CIKM, 2015

Data Reduction

Data Reduction

A process in which a reduced representation of a dataset that produces the same or similar analytical results is obtained.

Two of the most common techniques:

1. Data Cube Aggregation
2. Dimensionality Reduction

Data Cube Aggregation

Organizing data in a multidimensional structure for faster querying and analysis

Year	Quarter	Month	Region	City	Category	Brand	Item	Sales Revenue
2024	Q1	Jan	West	LA	Laptop	Dell	XPS	50,000
2024	Q1	Jan	West	SF	Laptop	HP	Envy	40,000
2024	Q1	Feb	East	NY	Phone	Apple	iPhone	70,000
2024	Q1	Mar	East	DC	Phone	Samsung	Galaxy	60,000

Year	Quarter	Month	Region	City	Category	Brand	Item	Sales Revenue
2024	Q1	Jan	West	LA	Laptop	Dell	XPS	50,000
2024	Q1	Jan	West	SF	Laptop	HP	Envy	40,000
2024	Q1	Feb	East	NY	Phone	Apple	iPhone	70,000
2024	Q1	Mar	East	DC	Phone	Samsung	Galaxy	60,000

Aggregated by Quarter:

- Summed all sales revenue within Q1 2024 across all locations and products
- 220,000

Aggregated by Region:

- Summed sales for each region in Q1 2024
- West: 90,000
- East: 13,000

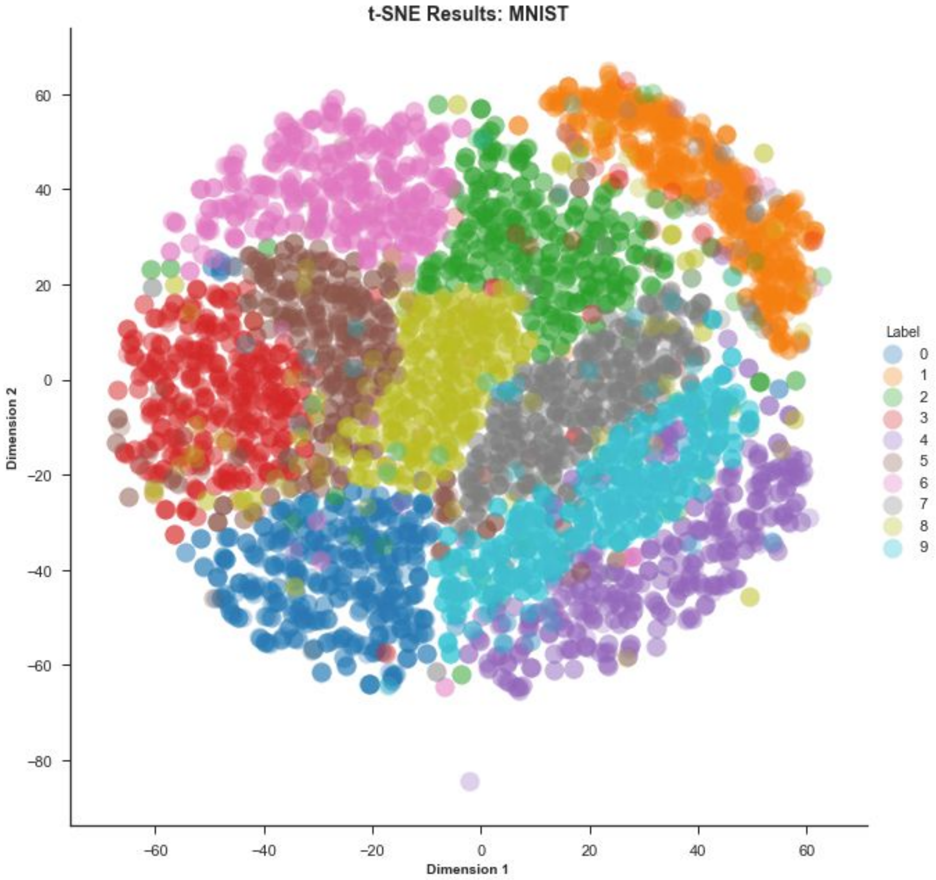
Aggregated by Category:

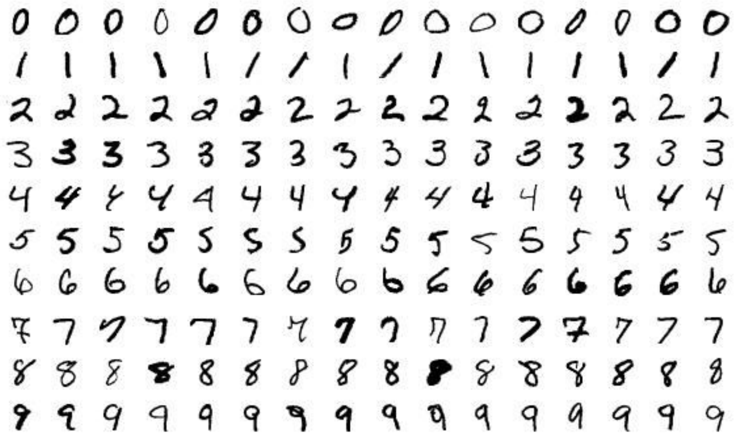
- Summed sales revenue for each product category in Q1 2024
- Laptop: 90,000
- Phone: 13,000

Dimensionality Reduction

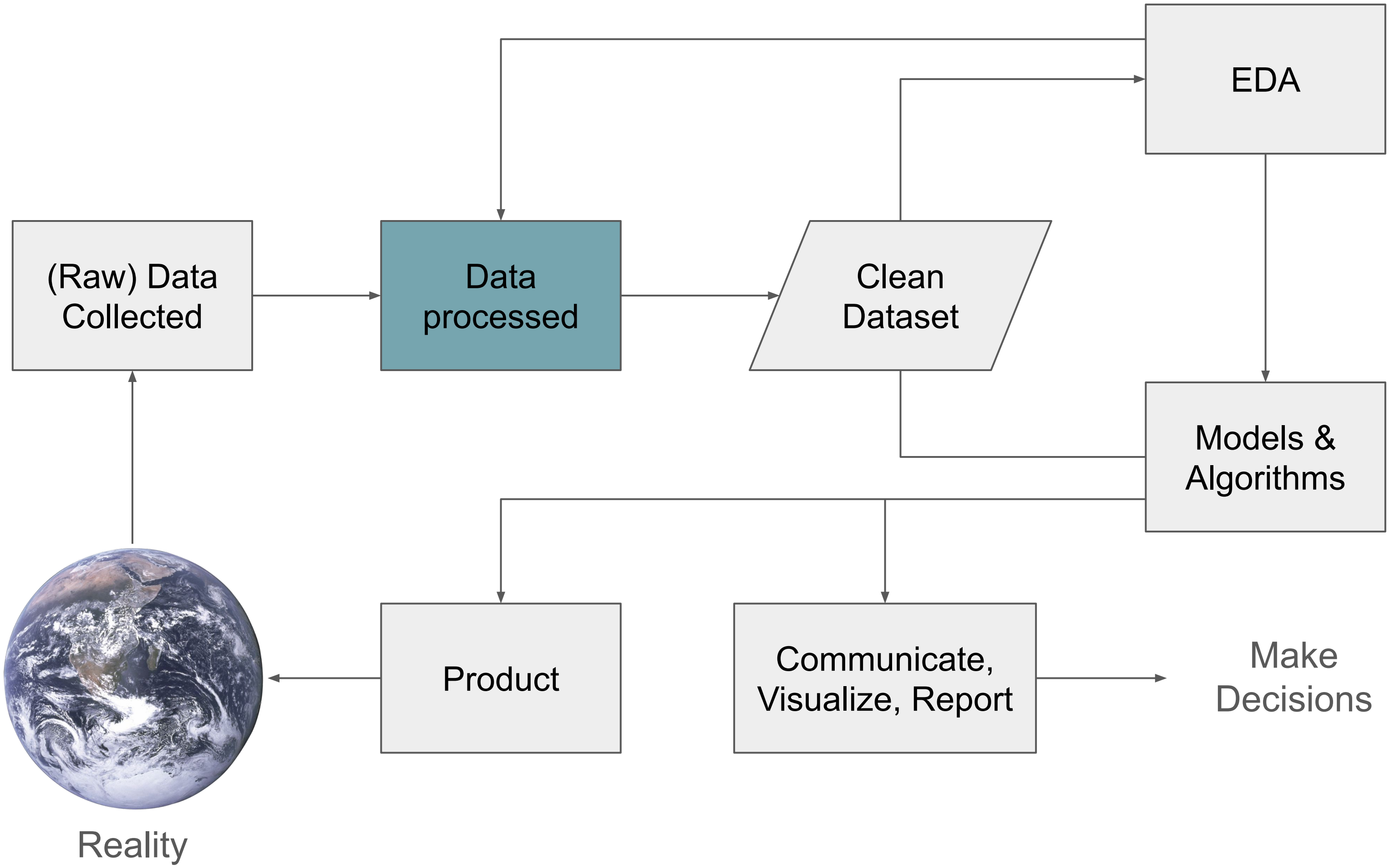
- Reducing the number of features (dimensions) in a dataset while preserving essential information.
- Helps avoid the **"curse of dimensionality"**
- **Techniques:**
    - Principal Component Analysis (PCA)
        - Identifies the most significant dimensions.
    - t-SNE & UMAP
        - Used for visualization of high-dimensional data.
    - Feature Selection
        - Removes irrelevant or redundant features.

Ref: https://agdal1125.github.io/2019/09/22/Curse-of-High-Dimensionality.html





Ref images:
https://medium.com/towards-data-science/an-introduction-to-t-sne-with-python-example-5a3a293108d1
https://paperswithcode.com/dataset/mnist


EDA
(Raw) Data Collected
Data processed
Clean Dataset
Models & Algorithms
Product
Communicate, Visualize, Report
Make Decisions
Reality

In Summary

- Data Science Workflow
- Significance of Data Preparation
- Roles in Data Processing
- Data Preprocessing
  - Data Cleaning
  - Data Integration
  - Data Transformation
  - Data Reduction

Ref: https://datasciencedojo.com/blog/data-science-memes/

Q & A