Module 1: Digital Health Data Analytics and Visualization

HEALTH DATA MANIPULATION

Ekarat Rattagan, Ph.D

Artificial Intelligence and Metabolomics Research Group
Faculty of Pharmacy, Silpakorn University

Module 1: Digital Health Data Analytics and Visualization

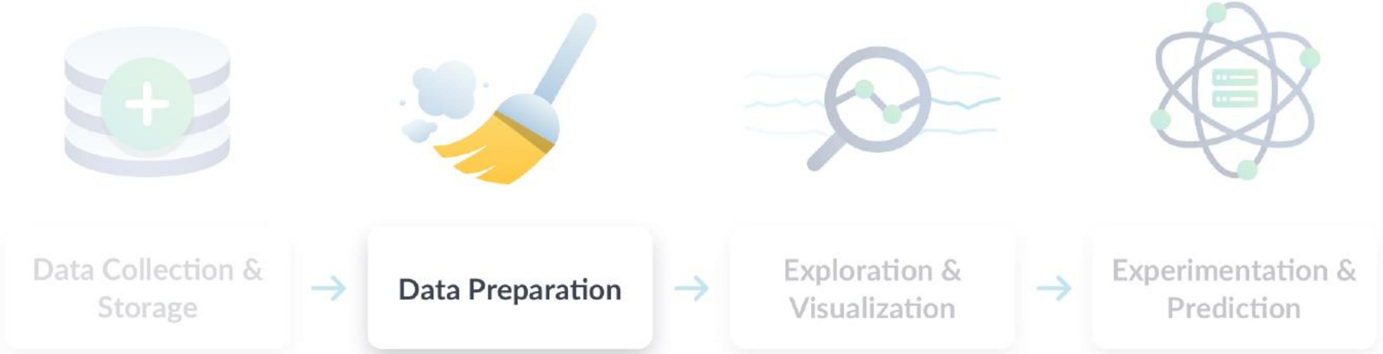
1. Understanding health data analytics ✓
2. Health data collection and storage ✓
3. Database design and query language in healthcare ✓
4. Health data manipulation ✓
5. Data visualizations and actionable dashboard
6. Biostatistical analysis

Ekarat Rattagan, Ph.D

Artificial Intelligence and Metabolomics Research Group
Faculty of Pharmacy, Silpakorn University

Data analysis workflow

- Real-world data is often disorganized and imperfect
- Preparation is necessary to avoid
  - ➢ Mistakes
  - ➢ Ensure accurate results
    Prevent algorithms from being biased



Data Collection & Storage → Data Preparation → Exploration & Visualization → Experimentation & Prediction

Datacamp

Data preparation

Data wrangling refers to the process of cleaning, transforming, and organizing raw data into a format that is suitable for analysis. It typically involves the following steps:

1. **Data Collection:** Gathering raw data from various sources.
2. **Data Cleaning:** Removing errors, inconsistencies, and missing values.
3. **Data Structuring:** Converting the data into a structured format, such as tables or spreadsheets or your requirement.
4. **Data Enrichment:** Adding new data or combining datasets to enhance the value of the original data.
5. **Data Transformation:** Manipulating the data by changing its format, reshaping it, or aggregating values.
6. **Validation:** Ensuring the data is accurate and ready for use in analysis.

1. Data Collection

Data Collection: Gathering raw data from various sources.

1.1 Primary Data: Collected firsthand through surveys, interviews, experiments, or observations.

1.2 Secondary Data: Sourced from existing materials like reports, databases, or academic literature.

1.3 Quantitative Data: Numerical data gathered via surveys, experiments, or sensors for statistical analysis.

1.4 Qualitative Data: Non-numerical data from interviews, open-ended surveys, or observations to understand deeper insights.

1.5 Automated Data: Automatically collected via web scraping, APIs, or machine logs.

1.6 Manual Data: Manually recorded or entered by individuals.

2. Data Cleaning

2. Data Cleaning: Removing errors, inconsistencies, and missing values.

2.1 Removing Duplicates: Eliminating repeated entries.

2.2 Handling Missing Data: Filling in, ignoring, or removing incomplete data.

2.3 Standardizing Formats: Ensuring consistent formats for dates, times, names, etc.

2.4 Outlier Detection: Identifying and addressing extreme or unusual data points.

2.5 Correcting Inaccuracies: Fixing errors in values (e.g., typos, wrong categories).

3. Data Structuring

3. Data Structuring: Converting the data into a structured format, such as tables or spreadsheets or your requirement.

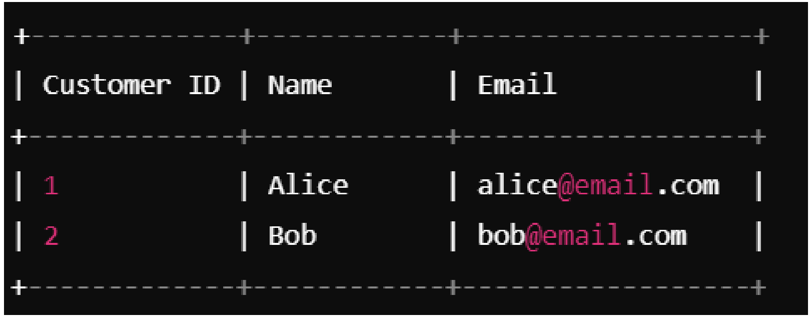
3.1 **Data Parsing:** Breaking down unstructured data (like text) into structured components (e.g., splitting full names into first and last names).

3.2 **Normalization***: Organizing data into tables by eliminating redundancy and dependencies (e.g., separating customer data into separate tables for names and addresses).

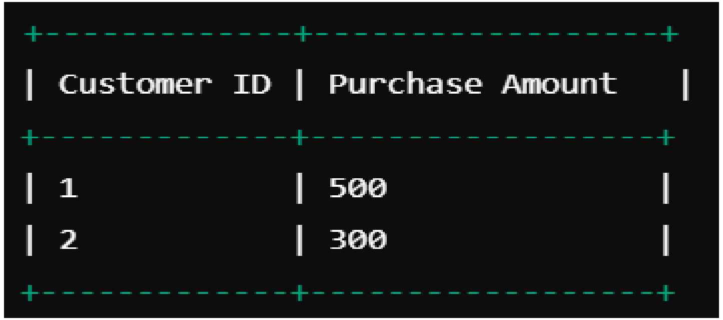
3.3 **Indexing:** Creating indexes or keys to efficiently organize and retrieve data from databases.

4. Data Enrichment

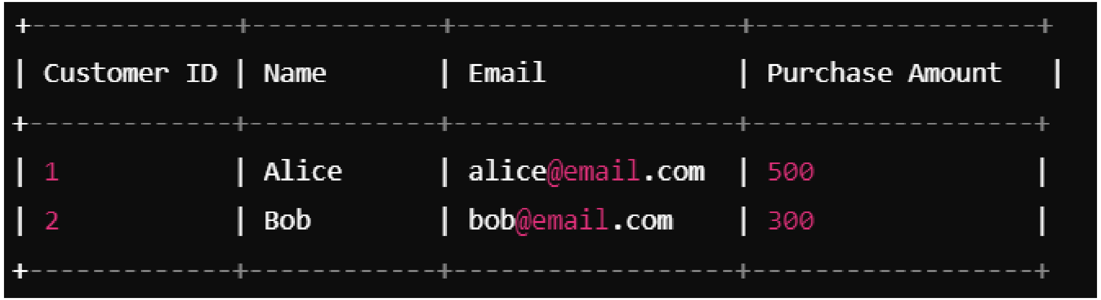
4. Data Enrichment: Adding new data or combining datasets to enhance the value of the original data.



```
+-------------+-------------+------------------+
| Customer ID | Name        | Email            |
+-------------+-------------+------------------+
| 1           | Alice       | alice@email.com  |
| 2           | Bob         | bob@email.com    |
+-------------+-------------+------------------+
```

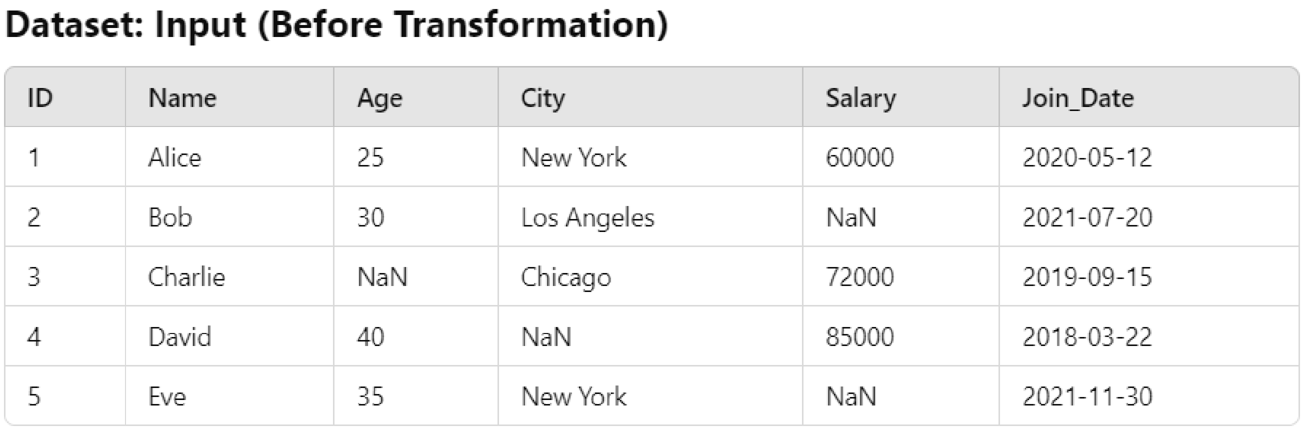


```
+-------------+------------------+
| Customer ID | Purchase Amount  |
+-------------+------------------+
| 1           | 500              |
| 2           | 300              |
+-------------+------------------+
```



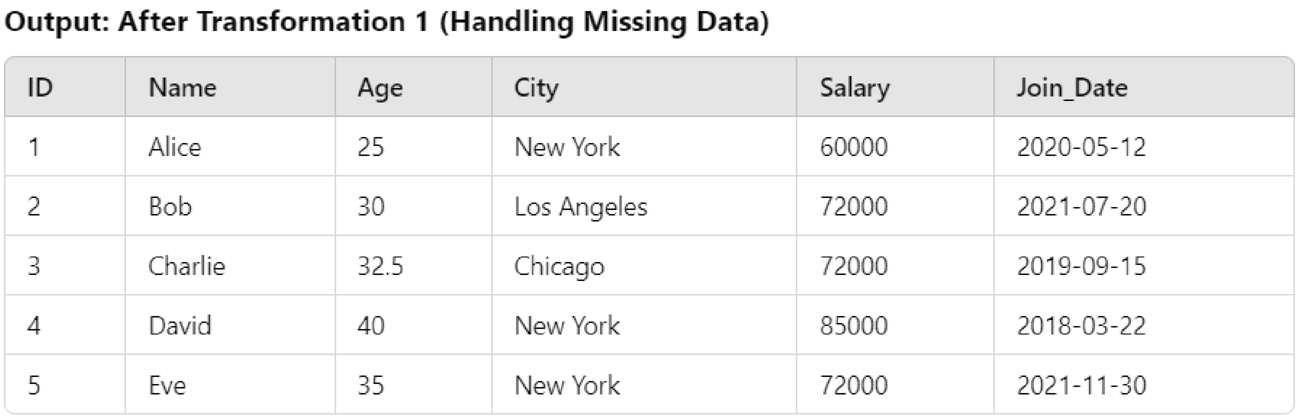
```
+-------------+-------------+------------------+------------------+
| Customer ID | Name        | Email            | Purchase Amount  |
+-------------+-------------+------------------+------------------+
| 1           | Alice       | alice@email.com  | 500              |
| 2           | Bob         | bob@email.com    | 300              |
+-------------+-------------+------------------+------------------+
```

5. Data Transformation



Dataset: Input (Before Transformation)

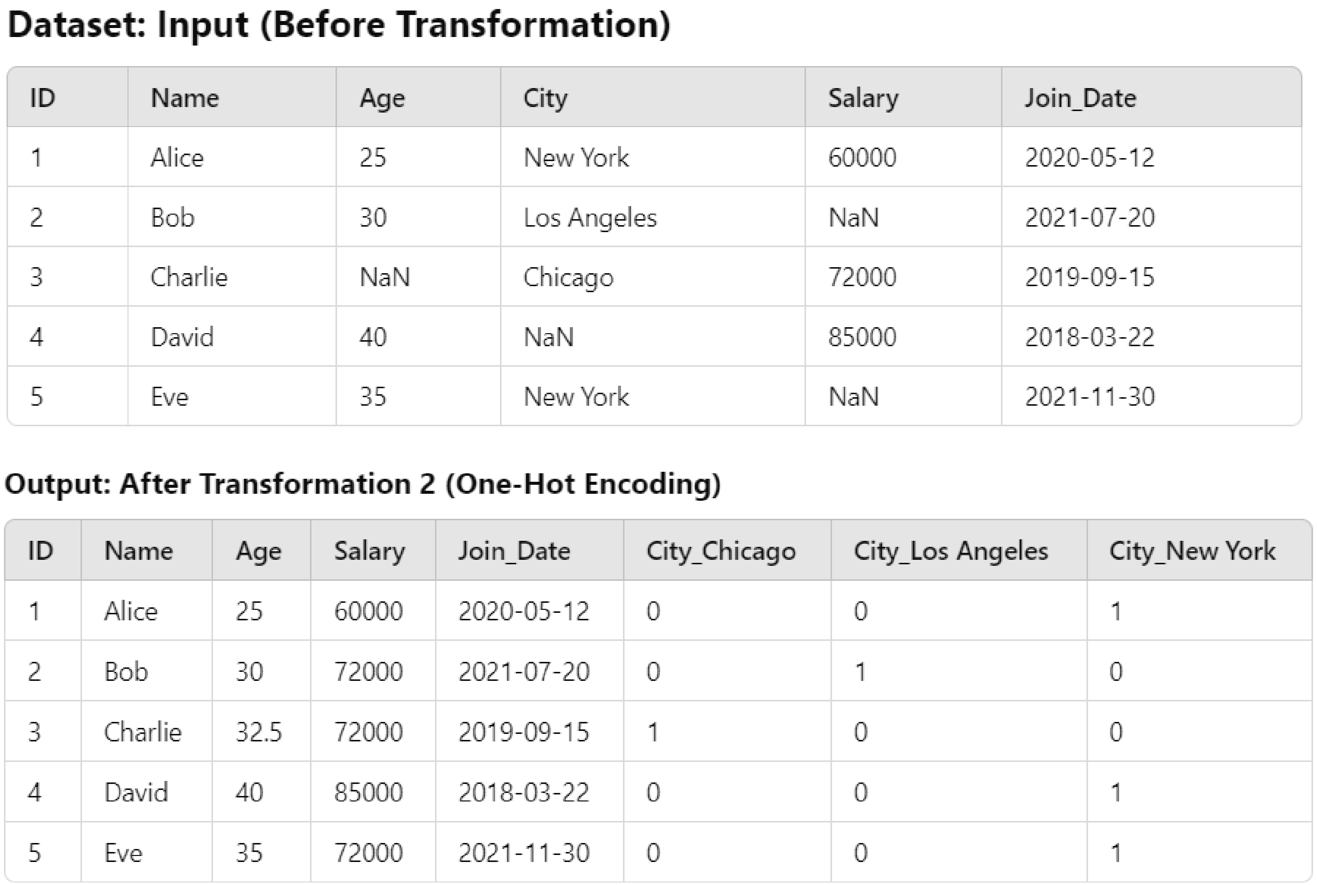
ID	Name	Age	City	Salary	Join_Date
1	Alice	25	New York	60000	2020-05-12
2	Bob	30	Los Angeles	NaN	2021-07-20
3	Charlie	NaN	Chicago	72000	2019-09-15
4	David	40	NaN	85000	2018-03-22
5	Eve	35	New York	NaN	2021-11-30



Output: After Transformation 1 (Handling Missing Data)

ID	Name	Age	City	Salary	Join_Date
1	Alice	25	New York	60000	2020-05-12
2	Bob	30	Los Angeles	72000	2021-07-20
3	Charlie	32.5	Chicago	72000	2019-09-15
4	David	40	New York	85000	2018-03-22
5	Eve	35	New York	72000	2021-11-30

5. Data Transformation



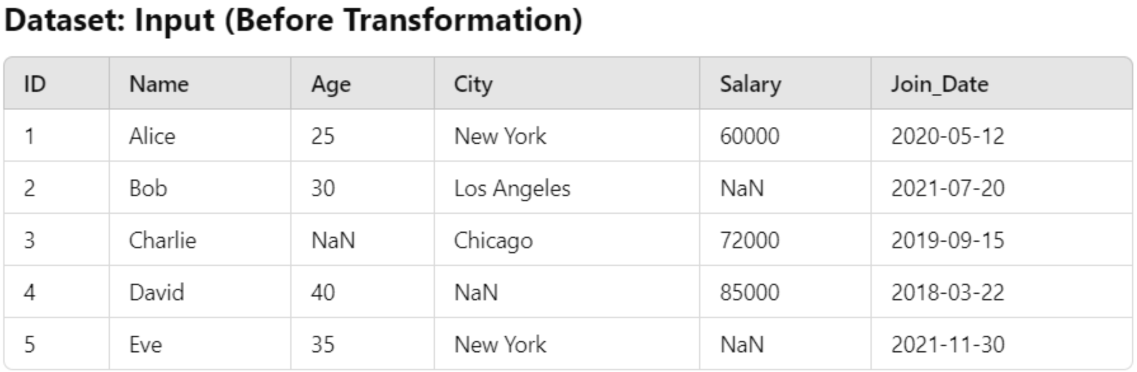
Dataset: Input (Before Transformation)

ID	Name	Age	City	Salary	Join_Date
1	Alice	25	New York	60000	2020-05-12
2	Bob	30	Los Angeles	NaN	2021-07-20
3	Charlie	NaN	Chicago	72000	2019-09-15
4	David	40	NaN	85000	2018-03-22
5	Eve	35	New York	NaN	2021-11-30

Output: After Transformation 2 (One-Hot Encoding)

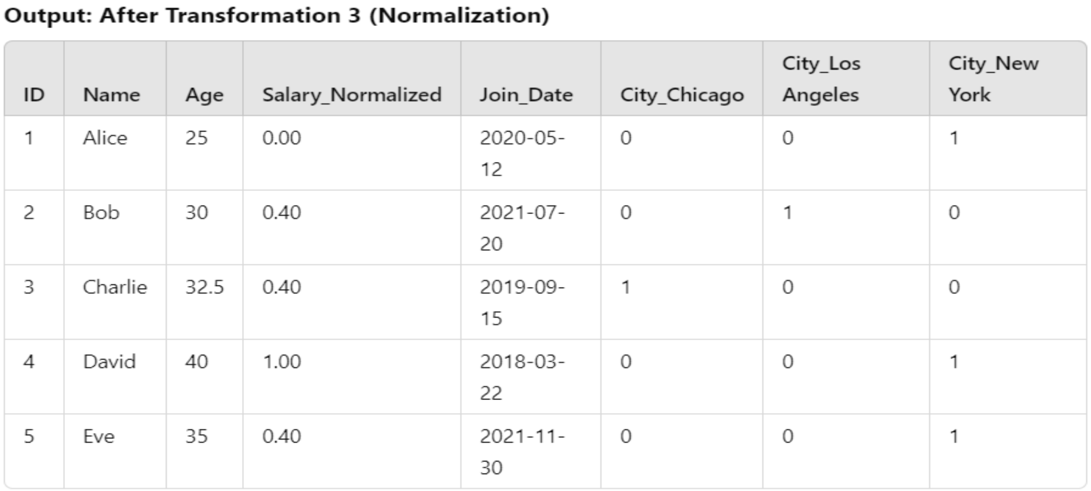
ID	Name	Age	Salary	Join_Date	City_Chicago	City_Los Angeles	City_New York
1	Alice	25	60000	2020-05-12	0	0	1
2	Bob	30	72000	2021-07-20	0	1	0
3	Charlie	32.5	72000	2019-09-15	1	0	0
4	David	40	85000	2018-03-22	0	0	1
5	Eve	35	72000	2021-11-30	0	0	1

5. Data Transformation



Dataset: Input (Before Transformation)

ID	Name	Age	City	Salary	Join_Date
1	Alice	25	New York	60000	2020-05-12
2	Bob	30	Los Angeles	NaN	2021-07-20
3	Charlie	NaN	Chicago	72000	2019-09-15
4	David	40	NaN	85000	2018-03-22
5	Eve	35	New York	NaN	2021-11-30



Output: After Transformation 3 (Normalization)

ID	Name	Age	Salary_Normalized	Join_Date	City_Chicago	City_Los Angeles	City_New York
1	Alice	25	0.00	2020-05-12	0	0	1
2	Bob	30	0.40	2021-07-20	0	1	0
3	Charlie	32.5	0.40	2019-09-15	1	0	0
4	David	40	1.00	2018-03-22	0	0	1
5	Eve	35	0.40	2021-11-30	0	0	1

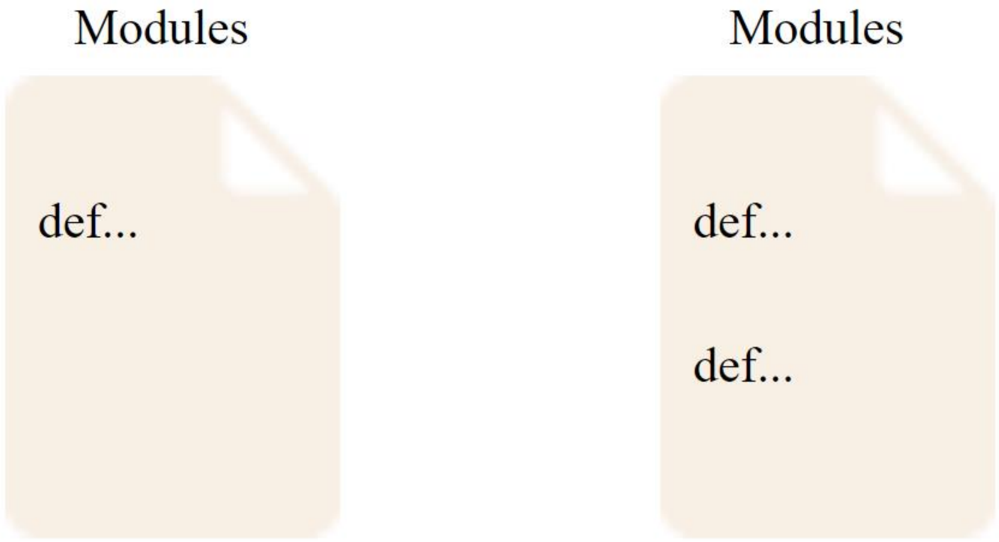
6. Validation

1. **Type Validation**: Ensures correct data types.
2. **Range Validation**: Checks numeric data within ranges.
3. **Format Validation**: Ensures data follows a format (dates, emails).
4. **Uniqueness Validation**: Ensures uniqueness of key fields.
5. **Consistency Validation**: Verifies logical consistency across fields.
6. **Null/Not Null Validation**: Ensures mandatory fields are not empty.
7. **Cross-Field Validation**: Ensures compatibility between fields.
8. **Domain Validation**: Checks data against predefined sets.
9. **Length Validation**: Ensures string/numeric length matches requirements.
10. **Reference Validation**: Ensures valid references between tables.
11. **Statistical Validation**: Verifies data against statistical norms.
12. **Business Rule Validation**: Validates data against custom business logic.

Data preparation: Python

- High level programming language
  - Web applications
  - Graphics
  - Games
  - Data Analytics
  - Data Visualization
  - Artificial Intelligence

Modules and Packages



Modules

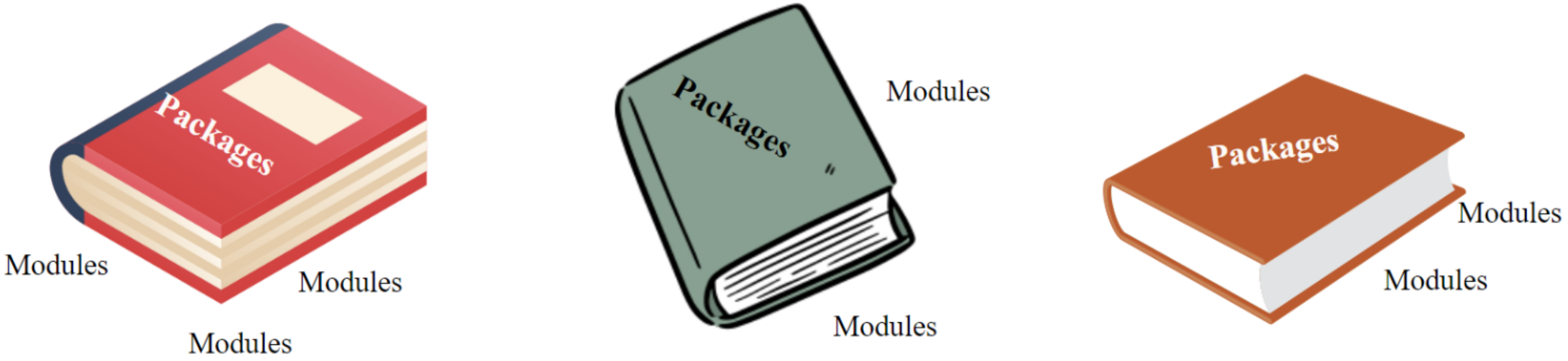
def...

Modules

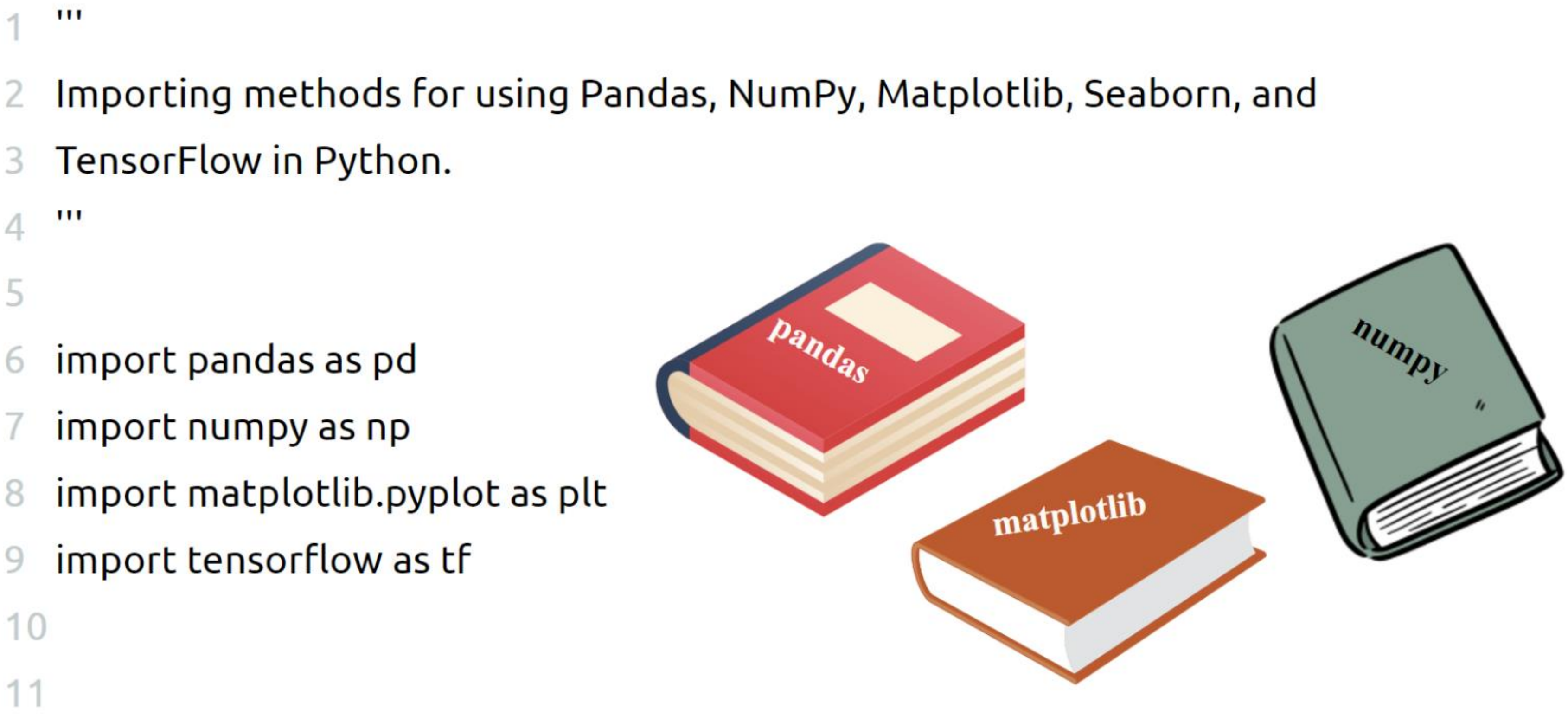
def...

def...

Modules and Packages



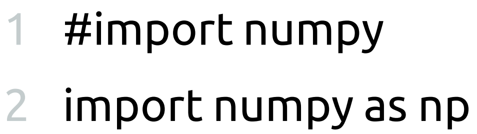
Modules and Packages: how to use



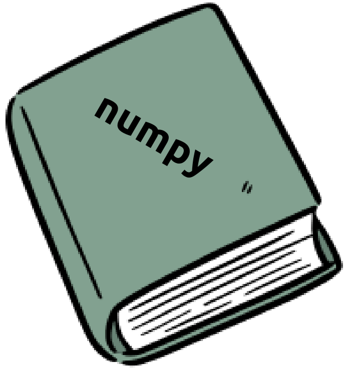
```
'''
Importing methods for using Pandas, NumPy, Matplotlib, Seaborn, and
TensorFlow in Python.
'''


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf


```

NumPy

This package is fundamental for scientific computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.



```
#import numpy
import numpy as np
```



NumPy: array introduction

```
#create list
alist = [1, 2, 3, 4, 5]
twoDlist = [[1, 2, 3],[4, 5, 6],[7, 8, 9]]

#get the first element
print(alist[0])
print(twoDlist [1])
print(twoDlist [2][2])
```

NumPy: array introduction

```
#create dictionary
twoDdict = {
    'rx1': {'A': 1, 'B': 2, 'C': 3},
    'rx2': {'A': 4, 'B': 5, 'C': 6},
    'rx3': {'A': 7, 'B': 8, 'C': 9}
}

#get the first element
print(twoDdict [rx1])
print(twoDdict [rx2][B])

```

NumPy: array introduction

```
a = [-1, 2, 4]
b = [3, 1, -5]

# c = a+ b
# c ?
c=a+b
c
```

$$\begin{array}{cc} a & b \\ \begin{bmatrix}-1\\2\\4\end{bmatrix} & \begin{bmatrix}3\\1\\-5\end{bmatrix}\end{array}$$

NumPy: array introduction

```
import numpy as np
a = [-1, 2, 4]
b = [3, 1, -5]


a_array = np.array(a)
b_array = np.array(b)
c = a_array + b_array
c


```

$$\begin{array}{cc} a & b \\ \begin{bmatrix}-1\\2\\4\end{bmatrix} & \begin{bmatrix}3\\1\\-5\end{bmatrix}\end{array}$$

NumPy: shape, size, and type

```
import numpy as np

d = [
  ["SU", 19, 87, 63],
  ["CU", 20, 71, 54 ]
]

d_array = np.array(d)
print(d_array)
d_array.shape
d_array.size
d_array.dtype
```

d

$$\begin{bmatrix} \text{SU} & 19 & 87 & 63 \\ \text{CU} & 20 & 71 & 54 \end{bmatrix}$$

NumPy: shape, size, and type

```
import numpy as np

d = [
  ["SU", 19, 87, 63],
  ["CU", 20, 71, 54 ]
]

d_array = np.array(d)
element = d_array[1, 1]
element


```

d

$$\begin{bmatrix} \mathrm{SU} & 19 & 87 & 63 \\ \mathrm{CU} & 20 & 71 & 54 \end{bmatrix}$$

NumPy: array operations

```
import numpy as np

d = [
  [19, 87, 63],
  [20, 71, 54 ]
]
d_array = np.array(d)
add_scalar = d_array + 5
add_scalar = d_array + d_array
```

NumPy: array operations

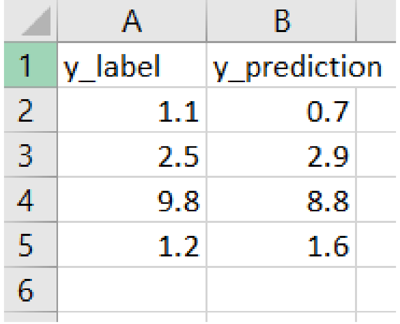
```
import numpy as np

d = [
  [19, 87, 63],
  [20, 71, 54 ]
]
d_array = np.array(d)
added_array = np.add(d_array, 10)
sqrt_array = np.sqrt(d_array)
total_sum = np.sum(d_array)
mean_value = np.mean(d_array)
transpose_d = d_array.T
```

NumPy: array operations

```python
import numpy as np

h = np.array([ [4, 2], [3, 5] ])

matrix_mult = h @ h
determinant = np.linalg.det(h)
inverse = np.linalg.inv(h)
eigenvalues, eigenvectors = np.linalg.eig(h)



```

NumPy: array operations

```
import numpy as np

data = np.genfromtxt('mse.csv', delimiter=',', skip_header=1)

y_label = data[:, 0]
y_prediction = data[:, 1]

mse = np.mean((y_label - y_prediction) ** 2)
```



	A	B
1	y_label	y_prediction
2	1.1	0.7
3	2.5	2.9
4	9.8	8.8
5	1.2	1.6
6		

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

MSE = mean squared error

n = number of data points

Y_i = observed values

$\hat{Y}_i$ = predicted values

NumPy: shape and size

```
import numpy as np

d = [
 ["SU", 19, 87, 63],
 ["CU", 20, 71, 54 ]
]

d_array = np.array(d)
reshaped_array = np.reshape(d_array, (4, 2))
flat_array = d_array.flatten()


```

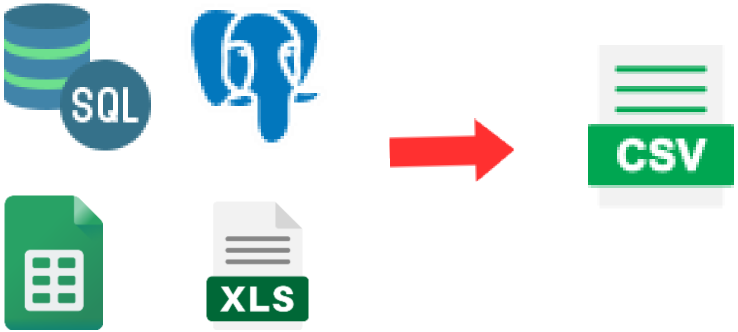
NumPy: shape and size

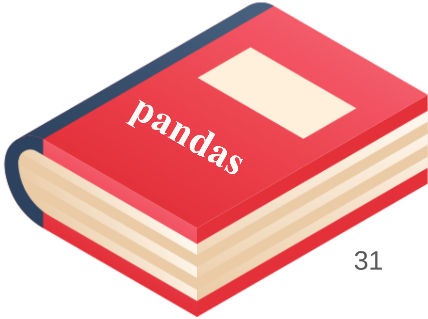
```
import numpy as np

d = [[1, 2, 3, 4], [5, 6, 7, 8 ]]
g = [[10, 20, 30, 40],[50, 60, 70, 80]]

d_array = np.array(d)
g_array = np.array(g)
concatenated_v = np.vstack((d_array, g_array))
concatenated_h = np.hstack((d_array, g))
split_h = np.hsplit(d_array, 2)
split_v = np.vsplit(d_array, 2)

```

Pandas

A package that provides data structures and data analysis tools. It's particularly well-suited for working with structured data, like tables, and is commonly used for data manipulation, cleaning, and analysis.

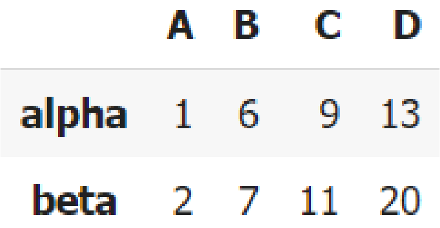
```
#import pandas
import pandas as pd
```





Pandas: dataframe

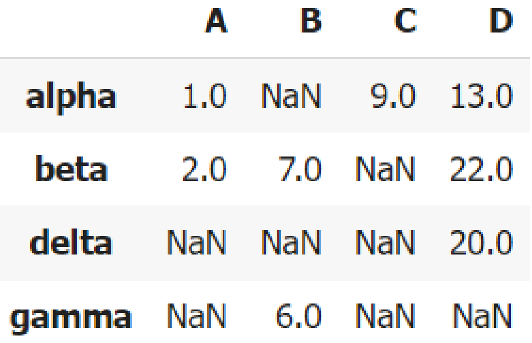
```
import pandas as pd

#DataFrame with 4 Series (columns) and 4 indices (rows)
data = {
  'A': [1, 2],
  'B': [6, 7],
  'C': [9, 11],
  'D': [13,20]
}
df = pd.DataFrame(data, index=['alpha', 'beta'])
df

```



	A	B	C	D
alpha	1	6	9	13
beta	2	7	11	20

Pandas: dataframe

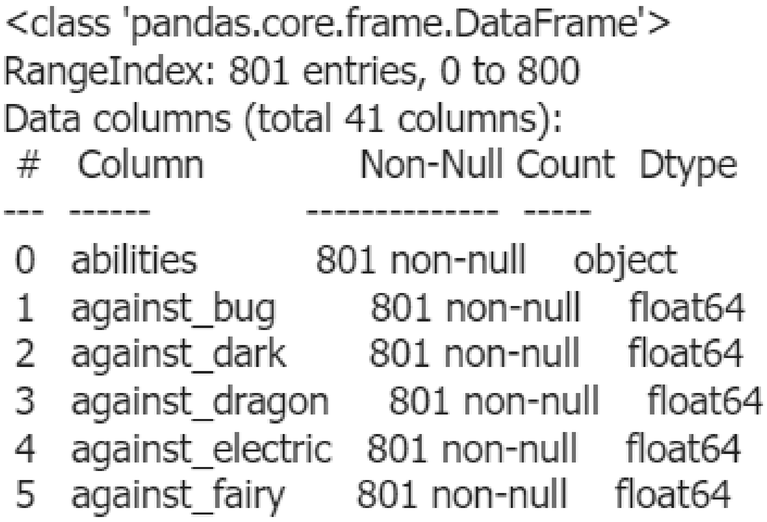
```
import pandas as pd

#DataFrame with 4 Series (columns) and 4 indices (rows)
data = {
  'A': pd.Series([1,2], index=['alpha', 'beta']),
  'B': pd.Series([6,7], index=['gamma', 'beta']),
  'C': pd.Series([9], index=['alpha']),
  'D': pd.Series([13,20,22], index=['alpha', 'delta','beta'])
}
df = pd.DataFrame(data)
df

```



	A	B	C	D
alpha	1.0	NaN	9.0	13.0
beta	2.0	7.0	NaN	22.0
delta	NaN	NaN	NaN	20.0
gamma	NaN	6.0	NaN	NaN

Pandas: csv importing

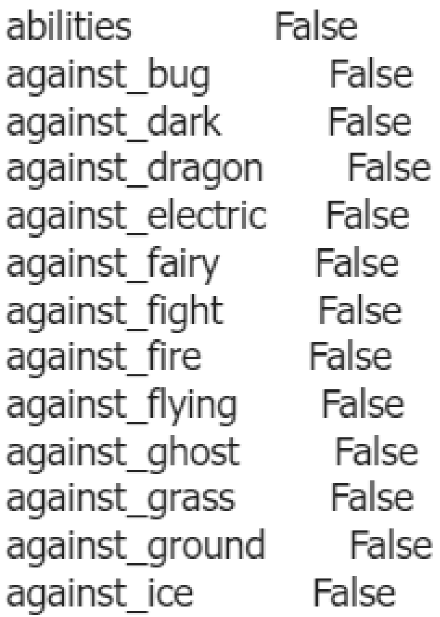
```python
import pandas as pd

df = pd.read_csv('pokemon.csv')
df
df.head()
df.tail()
df.info()
```



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 801 entries, 0 to 800
Data columns (total 41 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   abilities         801 non-null    object
 1   against_bug       801 non-null    float64
 2   against_dark      801 non-null    float64
 3   against_dragon    801 non-null    float64
 4   against_electric  801 non-null    float64
 5   against_fairy     801 non-null    float64
```

Pandas: shape and null value

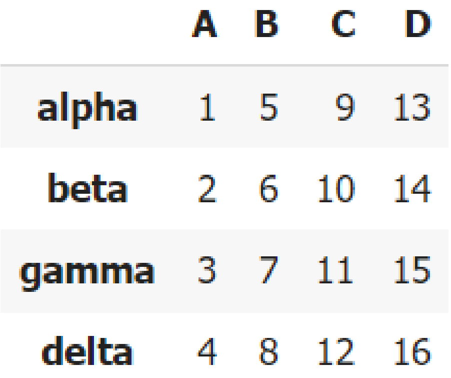
```
import pandas as pd

df = pd.read_csv('pokemon.csv')

df.shape
df.isnull().any()
```



```
abilities            False
against_bug          False
against_dark         False
against_dragon       False
against_electric     False
against_fairy        False
against_fight        False
against_fire         False
against_flying       False
against_ghost        False
against_grass        False
against_ground       False
against_ice          False
```

Pandas: dataframe operations

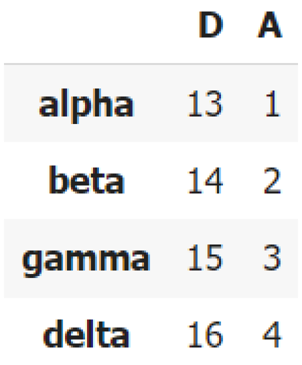
```python
import pandas as pd
data = {
    'A': [1, 2, 3, 4],
    'B': [5, 6, 7, 8],
    'C': [9, 10, 11, 12],
    'D': [13, 14, 15, 16]
}
df = pd.DataFrame(data, index=['alpha', 'beta', 'gamma', 'delta'])

df


```



	A	B	C	D
alpha	1	5	9	13
beta	2	6	10	14
gamma	3	7	11	15
delta	4	8	12	16

Pandas: dataframe operations

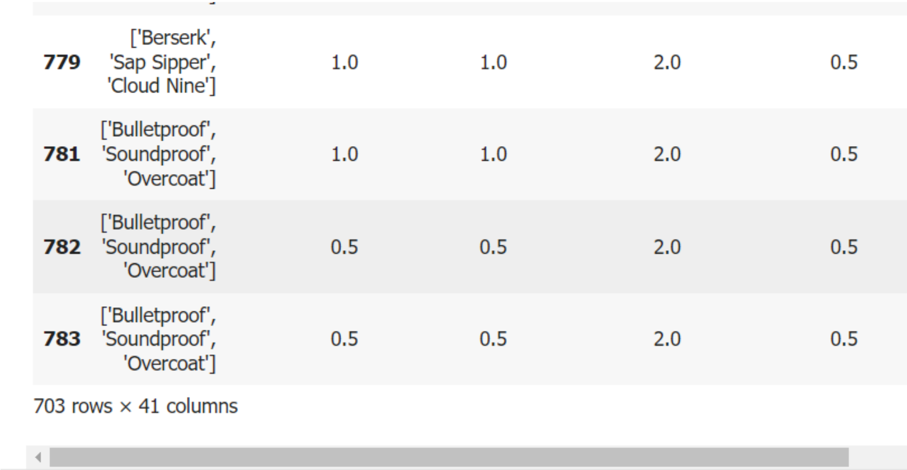
```python
import pandas as pd
data = {
    'A': [1, 2, 3, 4],
    'B': [5, 6, 7, 8],
    'C': [9, 10, 11, 12],
    'D': [13, 14, 15, 16]
}
df = pd.DataFrame(data, index=['alpha', 'beta', 'gamma', 'delta'])

df[['D', 'A']]


```



	D	A
alpha	13	1
beta	14	2
gamma	15	3
delta	16	4

Pandas: dataframe operations

```
import pandas as pd

df = pd.read_csv('pokemon.csv')

df = df[df['percentage_male'].notna()]
df
```

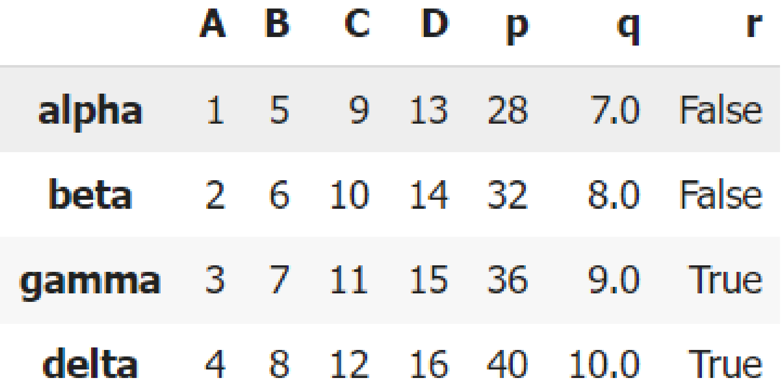


779	['Berserk', 'Sap Sipper', 'Cloud Nine']	1.0	1.0	2.0	0.5
781	['Bulletproof', 'Soundproof', 'Overcoat']	1.0	1.0	2.0	0.5
782	['Bulletproof', 'Soundproof', 'Overcoat']	0.5	0.5	2.0	0.5
783	['Bulletproof', 'Soundproof', 'Overcoat']	0.5	0.5	2.0	0.5

703 rows × 41 columns

Pandas: dataframe operations

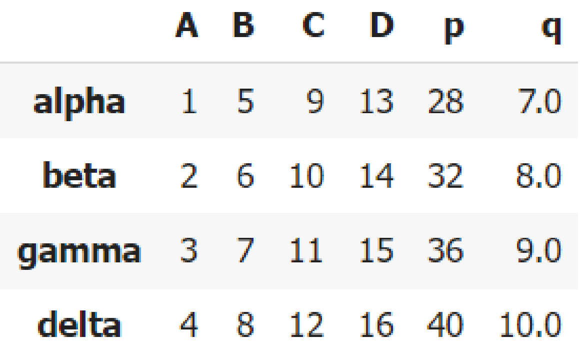
```
import pandas as pd
data = {
 'A': [1, 2, 3, 4],
 'B': [5, 6, 7, 8],
 'C': [9, 10, 11, 12],
 'D': [13, 14, 15, 16]
}
df = pd.DataFrame(data, index=['alpha', 'beta', 'gamma', 'delta'])
df['p'] = df['A'] + df['B']+ df['C']+ df['D']
df['q'] = df['p'] /4
df['r'] = df['q'] >=8.5
df
```



	A	B	C	D	p	q	r
alpha	1	5	9	13	28	7.0	False
beta	2	6	10	14	32	8.0	False
gamma	3	7	11	15	36	9.0	True
delta	4	8	12	16	40	10.0	True

Pandas: dataframe operations

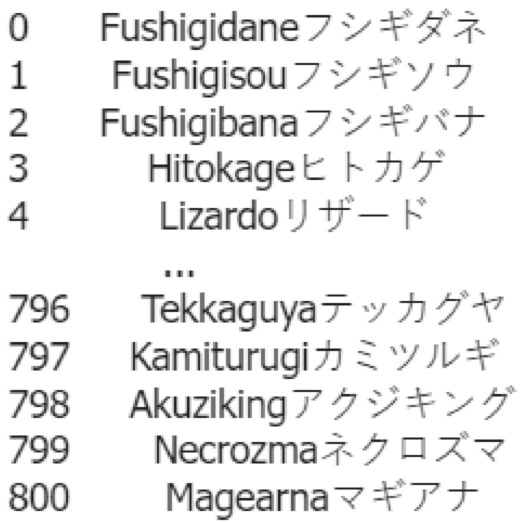
```
import pandas as pd
data = {
  'A': [1, 2, 3, 4],
  'B': [5, 6, 7, 8],
  'C': [9, 10, 11, 12],
  'D': [13, 14, 15, 16]
}
df = pd.DataFrame(data, index=['alpha', 'beta', 'gamma', 'delta'])
df['p'] = df['A'] + df['B']+ df['C']+ df['D']
df['q'] = df['p'] /4
df['r'] = df['q'] >=8.5
df = df.drop('r', axis=1)
df
```



	A	B	C	D	p	q
alpha	1	5	9	13	28	7.0
beta	2	6	10	14	32	8.0
gamma	3	7	11	15	36	9.0
delta	4	8	12	16	40	10.0

Pandas: dataframe operations

```
import pandas as pd

df = pd.read_csv('pokemon.csv')

df.columns
df.japanese_name
df['japanese_name']
print(df['japanese_name'])
```

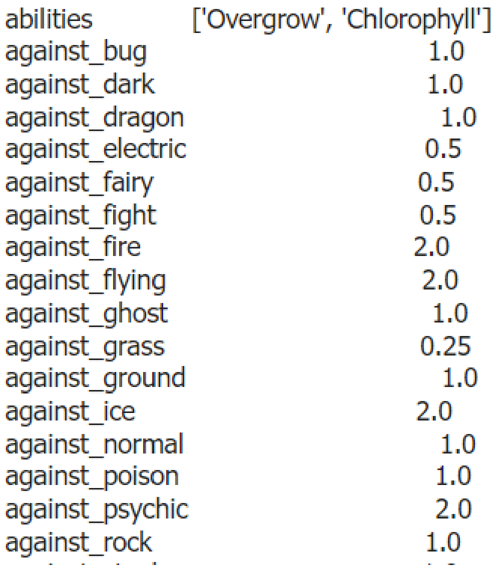


```
0      Fushigidaneフシギダネ
1       Fushigisouフシギソウ
2      Fushigibanaフシギバナ
3          Hitokageヒトカゲ
4          Lizardoリザード
                ...
796     Tekkaguyaテッカグヤ
797   Kamiturugiカミツルギ
798  Akuzikingアクジキング
799     Necrozmaネクロズマ
800       Magearnaマギアナ
```

Dataset

Pandas: dataframe operations

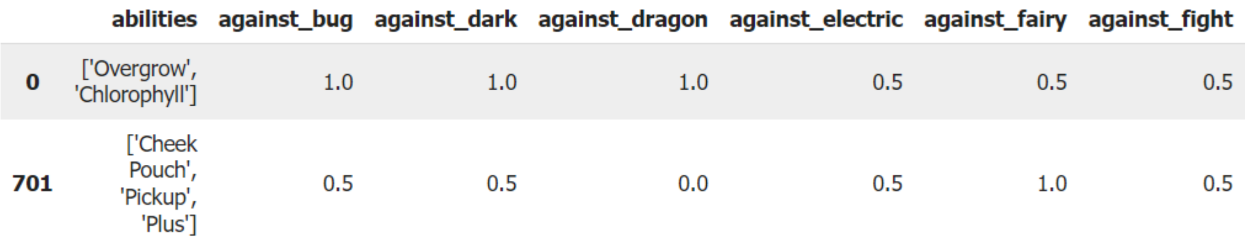
```
import pandas as pd

df = pd.read_csv('pokemon.csv')

p1=df.iloc[0]
p1
```



```
abilities          ['Overgrow', 'Chlorophyll']
against_bug                            1.0
against_dark                           1.0
against_dragon                         1.0
against_electric                       0.5
against_fairy                          0.5
against_fight                          0.5
against_fire                           2.0
against_flying                         2.0
against_ghost                          1.0
against_grass                          0.25
against_ground                         1.0
against_ice                            2.0
against_normal                         1.0
against_poison                         1.0
against_psychic                        2.0
against_rock                           1.0
```

Pandas: dataframe operations

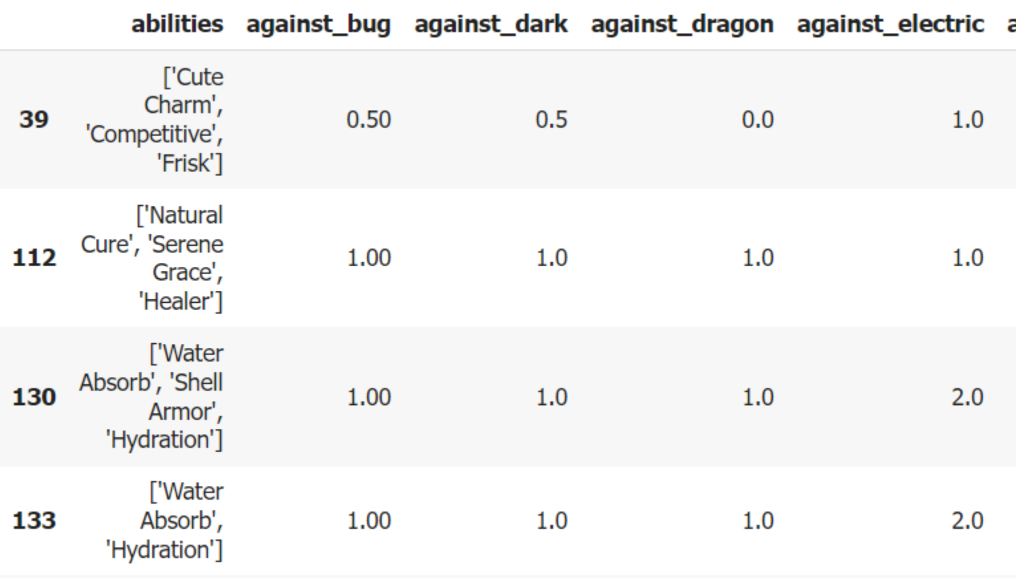
```
import pandas as pd

df = pd.read_csv('pokemon.csv')

p1=df.iloc[0]
p2=df.iloc[701]
```



	abilities	against_bug	against_dark	against_dragon	against_electric	against_fairy	against_fight
0	['Overgrow', 'Chlorophyll']	1.0	1.0	1.0	0.5	0.5	0.5
701	['Cheek Pouch', 'Pickup', 'Plus']	0.5	0.5	0.0	0.5	1.0	0.5

Pandas: dataframe operations

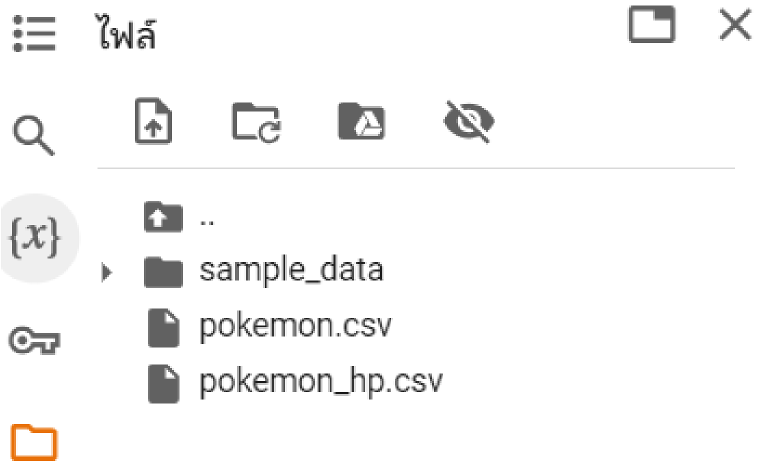
```
import pandas as pd

df = pd.read_csv('pokemon.csv')

df[df.hp>=125]
```



	abilities	against_bug	against_dark	against_dragon	against_electric	a
39	['Cute Charm', 'Competitive', 'Frisk']	0.50	0.5	0.0	1.0	
112	['Natural Cure', 'Serene Grace', 'Healer']	1.00	1.0	1.0	1.0	
130	['Water Absorb', 'Shell Armor', 'Hydration']	1.00	1.0	1.0	2.0	
133	['Water Absorb', 'Hydration']	1.00	1.0	1.0	2.0	

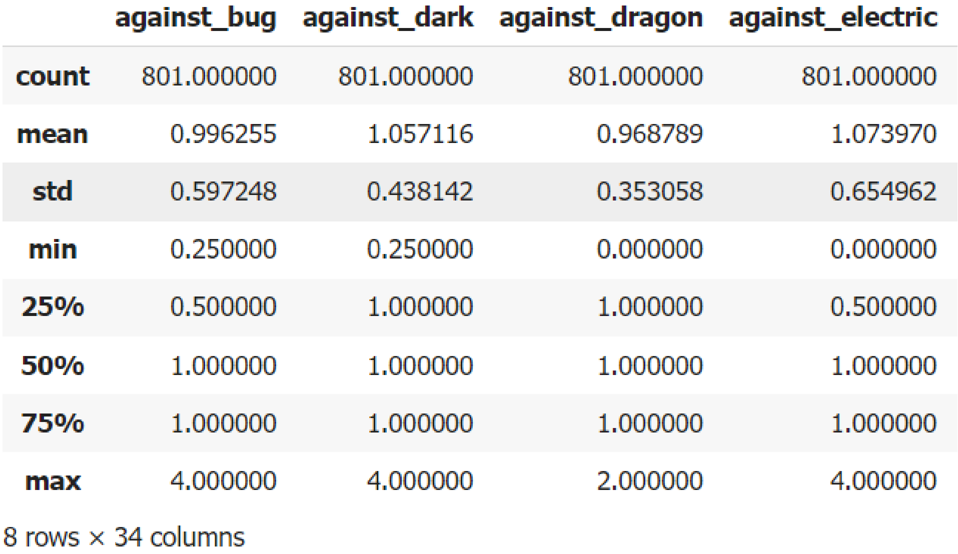
Pandas: dataframe exporting

```
import pandas as pd

df = pd.read_csv('pokemon.csv')

df_hp=df[df.hp>=125]
df_hp.to_csv('pokemon_hp.csv', index=False)
```



Pandas: basic statistics

```
import pandas as pd

df = pd.read_csv('pokemon.csv')

df.describe()
```

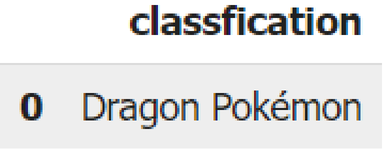


	against_bug	against_dark	against_dragon	against_electric
count	801.000000	801.000000	801.000000	801.000000
mean	0.996255	1.057116	0.968789	1.073970
std	0.597248	0.438142	0.353058	0.654962
min	0.250000	0.250000	0.000000	0.000000
25%	0.500000	1.000000	1.000000	0.500000
50%	1.000000	1.000000	1.000000	1.000000
75%	1.000000	1.000000	1.000000	1.000000
max	4.000000	4.000000	2.000000	4.000000

8 rows × 34 columns

Pandas: basic statistics

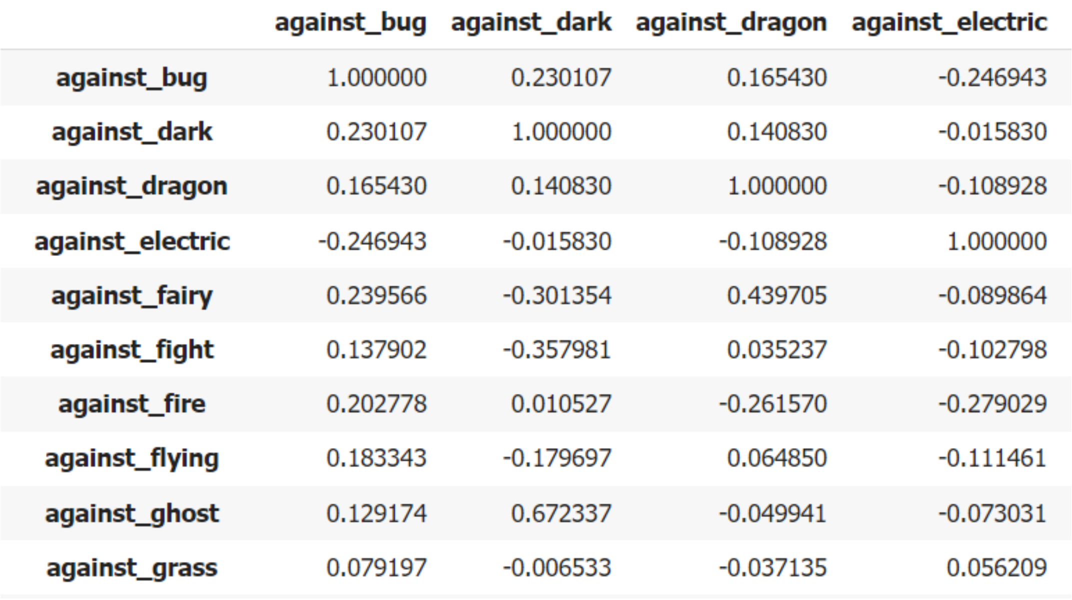
```
import pandas as pd

df = pd.read_csv('pokemon.csv')

df_class= df[['classfication']]
df_class.mode()
```



	classfication
0	Dragon Pokémon

Pandas: basic statistics

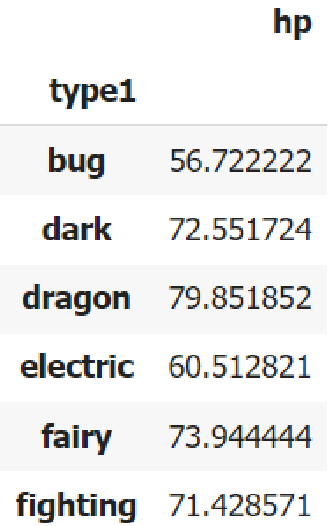
```
import pandas as pd

df = pd.read_csv('pokemon.csv')

df.corr()
```



	against_bug	against_dark	against_dragon	against_electric
against_bug	1.000000	0.230107	0.165430	-0.246943
against_dark	0.230107	1.000000	0.140830	-0.015830
against_dragon	0.165430	0.140830	1.000000	-0.108928
against_electric	-0.246943	-0.015830	-0.108928	1.000000
against_fairy	0.239566	-0.301354	0.439705	-0.089864
against_fight	0.137902	-0.357981	0.035237	-0.102798
against_fire	0.202778	0.010527	-0.261570	-0.279029
against_flying	0.183343	-0.179697	0.064850	-0.111461
against_ghost	0.129174	0.672337	-0.049941	-0.073031
against_grass	0.079197	-0.006533	-0.037135	0.056209

Pandas: basic statistics

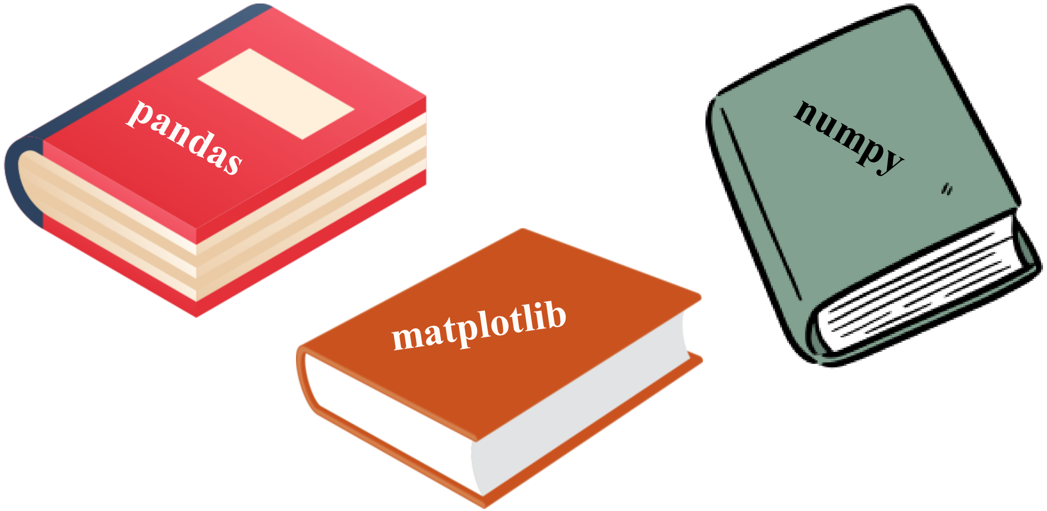
```
import pandas as pd

df = pd.read_csv('pokemon.csv')

df2 = df[[ 'type1', 'hp']]
df2.groupby("type1").mean()
```



```
                 hp
type1
bug       56.722222
dark      72.551724
dragon    79.851852
electric  60.512821
fairy     73.944444
fighting  71.428571
```

Modules and Packages: importing

```
'''
Importing methods for using Pandas, NumPy, Matplotlib, Seaborn, and
TensorFlow in Python.
'''

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf


```

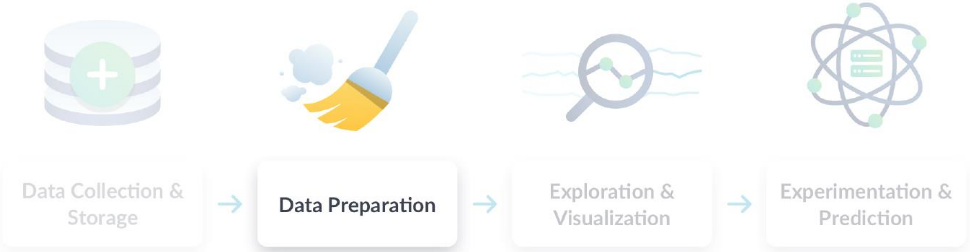


Data preparation: techniques

1. Data Integration
2. Data Manipulation
3. Data Cleaning
4. Data Transformation
5. Feature Engineering
6. Data Normalization
7. Data Augmentation
8. Data Reduction
9. Data Imputation
10. Data Validation
11. Data Enrichment
12. Data Partitioning

Tools

- Spreadsheet
- Statistical tools
- RDMS
- BigQuery
- Python

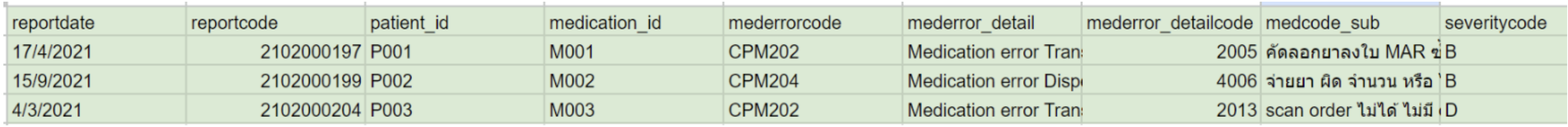

Data Collection & Storage
Data Preparation
Exploration & Visualization
Experimentation & Prediction

Data integration

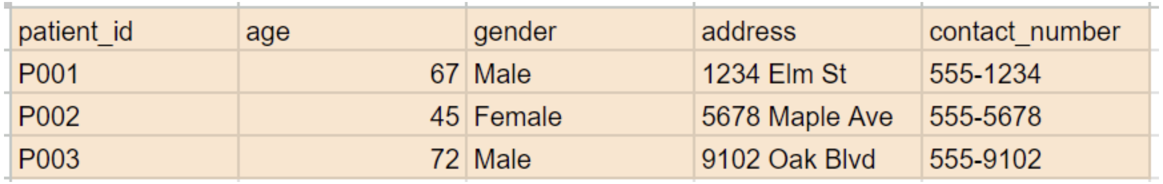
- Combine data from various sources into a single, cohesive dataset
- Merge datasets from different databases, files, or APIs.
- Resolve conflicts between data sources (e.g., different formats, naming conventions).
- Ensure that integrated data aligns correctly (e.g., matching IDs or timestamps).

Data integration: case study

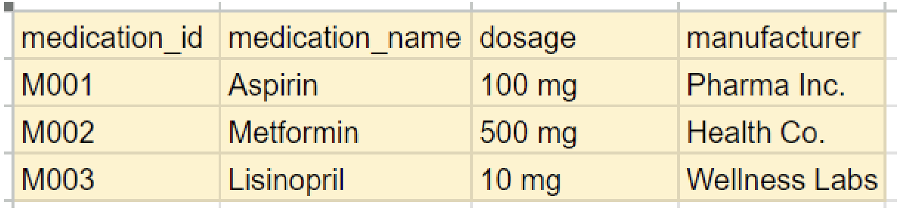
Before dataset



reportdate	reportcode	patient_id	medication_id	mederrorcode	mederror_detail	mederror_detailcode	medcode_sub	severitycode
17/4/2021	2102000197	P001	M001	CPM202	Medication error Tran	2005	คัดลอกยาลงใบ MAR ข	B
15/9/2021	2102000199	P002	M002	CPM204	Medication error Disp	4006	จ่ายยา ผิด จำนวน หรือ	B
4/3/2021	2102000204	P003	M003	CPM202	Medication error Tran	2013	scan order ไม่ได้ ไม่มี	D



patient_id	age	gender	address	contact_number
P001	67	Male	1234 Elm St	555-1234
P002	45	Female	5678 Maple Ave	555-5678
P003	72	Male	9102 Oak Blvd	555-9102



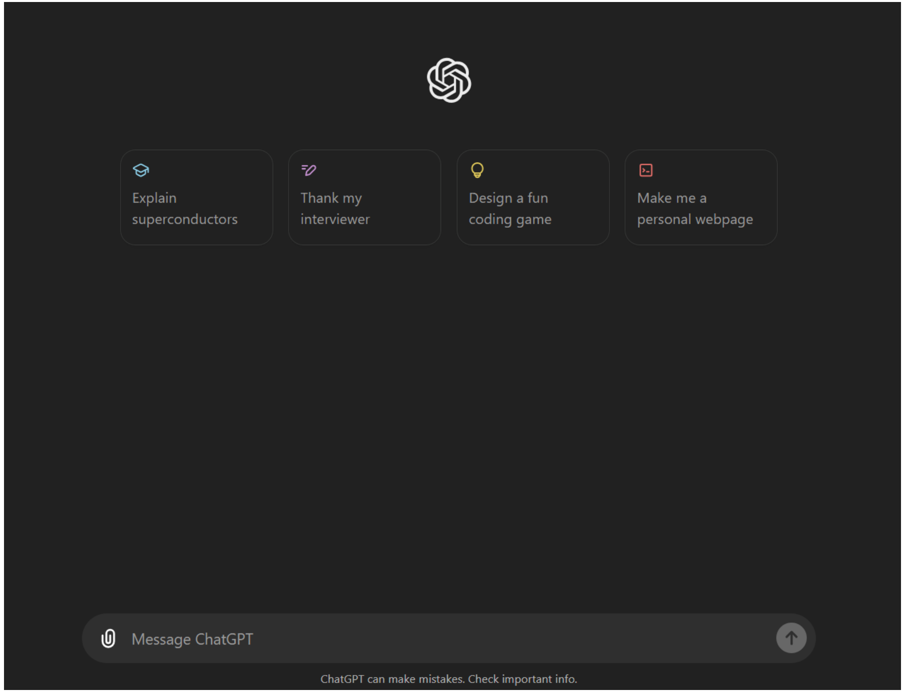
medication_id	medication_name	dosage	manufacturer
M001	Aspirin	100 mg	Pharma Inc.
M002	Metformin	500 mg	Health Co.
M003	Lisinopril	10 mg	Wellness Labs

Data manipulation

- Modify and adjust the data to make it more suitable for analysis
- Filter, sort, and organize data
- Compute new variables or derive new columns from existing data
- Reformat data types and structures (e.g., converting strings to dates)

Data preparation with GenAI

1. Data manipulation
2. Data cleaning



Code exercise

1. Example 1: https://github.com/ekaratnida/Applied-machine-learning/blob/master/Example1_EDA.ipynb
2. Example 2: https://github.com/ekaratnida/Applied-machine-learning/blob/master/Example2_EDA.ipynb