


PCED

Module 1: Data Acquisition and
Pre-processing

A dark blue diagonal gradient bar that starts from the bottom left and extends towards the top right, covering the lower half of the slide.

Data Acquisition

a. Data Collection Methods:

- **Surveys:** Learn how to design surveys that capture relevant data while minimizing bias. Understand different survey techniques (e.g., online, face-to-face) and their advantages.
- **Web Scraping:** Familiarize yourself with web scraping tools like `BeautifulSoup` or `Selenium` in Python. Know how to extract data from websites and handle challenges like CAPTCHA and dynamic content.

b. Ethical Considerations:

- **Privacy:** Always consider user consent and the legal aspects of collecting data, particularly from websites (e.g., scraping should comply with `robots.txt` rules and GDPR regulations).
- **Bias:** Recognize potential biases in data collection methods and aim for representativeness.
- **Data Security:** Safeguard sensitive information and follow best practices for anonymizing personal data.

Surveys (Manual Entry Example)

```
import pandas as pd
```

```
# Example of survey data collection
```

```
survey_data = {
```

```
    'Name': ['John', 'Doe', 'Jane'],
```

```
    'Age': [28, 34, 29],
```

```
    'Favorite Color': ['Blue', 'Green', 'Red']
```

```
}
```

```
df_survey = pd.DataFrame(survey_data)
```

```
print(df_survey)
```

Understanding the code

1. `Import` is used to get the libraries work for the code we write , it provides data structures like `DataFrames` that are highly efficient for handling and analyzing structured data.
2. `survey_data`: A dictionary containing survey responses. The keys (e.g., 'Name', 'Age', and 'Favorite Color') represent the column names of the dataset. The values for each key are lists of data corresponding to the survey responses.
3. `pd.DataFrame()`: This function creates a `DataFrame` from the dictionary `survey_data`.
A `DataFrame` is like a table in Excel or a SQL database. It has rows and columns, where the rows represent individual records (survey responses), and the columns represent the different variables (Name, Age, Favorite Color).
4. `print(df_survey)` - This prints the `DataFrame` to the console, displaying the survey data in a tabular format.

Auto-scraping

```
import requests

from bs4 import BeautifulSoup

url = 'https://example.com'

response = requests.get(url)

soup = BeautifulSoup(response.text, 'html.parser')

# Extract title of the page

page_title = soup.title.string

print(f"Page Title: {page_title}")

# Extract specific data (e.g., all links on the page)

all_links = soup.find_all('a')

for link in all_links:

    print(link.get('href'))
```

Understanding the code

1. Importing Required Libraries

2. Sending a Request to a Web Page

```
url = 'https://example.com'
```

```
response = requests.get(url)
```

`url = 'https://example.com':` This is the URL of the webpage you want to scrape.

You can change it to any website you wish to access.

`response = requests.get(url):` The `requests.get()` method sends an HTTP GET request to the given URL, retrieving the webpage's content (HTML). The result is stored in the response object.

3. Parsing the HTML Content

```
page_title = soup.title.string
```

```
print(f"Page Title: {page_title}")
```

`soup.title:` This finds the `<title>` tag in the parsed HTML document.

`.string:` This extracts the text inside the `<title>` tag.

`print(f"Page Title: {page_title}"):` This prints the title of the page in a formatted string.

Understanding the code .. II

4. Extracting All Links (Anchor Tags) on the Page

```
all_links = soup.find_all('a')
```

`soup.find_all('a')`: This finds all `<a>` (anchor) tags in the HTML document. Anchor tags typically represent hyperlinks on a webpage. The method returns a list of all the anchor elements.

5. Printing the Links

```
for link in all_links:  
    print(link.get('href'))
```

The for loop iterates through each anchor tag found in `all_links`.

`link.get('href')`: This extracts the value of the href attribute from each `<a>` tag, which contains the URL of the hyperlink. If the tag has no href attribute, it returns `None`.

Image scraping from google

```
import os

import requests

from bs4 import BeautifulSoup

# Function to download images from the URLs

def download_images(image_urls, folder_name):

    if not os.path.exists(folder_name):

        os.makedirs(folder_name)
```


Image from google .. II

```
for i, url in enumerate(image_urls):
    try:
        image_data = requests.get(url).content
        file_path = os.path.join(folder_name, f'image_{i+1}.jpg')
        with open(file_path, 'wb') as handler:
            handler.write(image_data)
        print(f"Image {i+1} downloaded: {file_path}")
    except Exception as e:
        print(f"Failed to download image {i+1}: {e}")
```

Image from google ..III

```
def scrape_images(query, num_images=10):
    # Prepare the Google search URL
    search_url = f"https://www.google.com/search?q={query}&tbm=isch"

    # Send an HTTP request to get the content of the search results page
    headers = {
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/58.0.3029.110 Safari/537.3"
    }
    response = requests.get(search_url, headers=headers)

    # Parse the HTML content using BeautifulSoup
    soup = BeautifulSoup(response.text, 'html.parser')

    # Find all image elements ('img' tags)
    img_elements = soup.find_all('img')
```

Image from google .. IV

```
image_urls = []
    for img in img_elements:
        src = img.get('src')
        if src and src.startswith('http'):
            image_urls.append(src)
        if len(image_urls) >= num_images:
            break

    return image_urls

if __name__ == "__main__":
    # User inputs the search query
    search_title = input("Enter the title of images you want to download: ")
    # Scrape image URLs from Google
    image_links = scrape_images(search_title, num_images=10)
    # Download images to a folder
    download_images(image_links, folder_name=search_title.replace(' ', '_'))
```

Data Pre-Processing

- Data Preprocessing: An Overview
- Data Quality
- Major Tasks in Data Preprocessing
- Data Cleaning
- Data Integration
- Data Reduction
- Data Transformation and Data Discretization
- Summary

Data Quality: Why Preprocess the Data?

- Measures for data quality: A multidimensional view
- Accuracy: correct or wrong, accurate or not
- Completeness: not recorded, unavailable, ...
- Consistency: some modified but some not, dangling, ...
- Timeliness: timely update?
- Believability: how trustable the data are correct?
- Interpretability: how easily the data can be understood?

Major Tasks in Data Preprocessing

Data cleaning

- Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies

Data integration

- Integration of multiple databases, data cubes, or files

Data reduction

- Dimensionality reduction
- Numerosity reduction
- Data compression

Data transformation and data discretization

- Normalization
- Concept hierarchy generation

Data Cleaning

Data in the Real World Is Dirty: Lots of potentially incorrect data, e.g., instrument faulty, human or computer error, transmission error

incomplete: lacking attribute values, lacking certain attributes of interest, or containing only aggregate data
e.g., Occupation=" " (missing data)

noisy: containing noise, errors, or outliers
e.g., Salary="-10" (an error)

inconsistent: containing discrepancies in codes or names, e.g.,
Age="42", Birthday="03/07/2010"

Was rating "1, 2, 3", now rating "A, B, C"

discrepancy between duplicate records

Intentional (e.g., disguised missing data)

Jan. 1 as everyone's birthday?

Incomplete (Missing) Data

Data is not always available

- E.g., many tuples have no recorded value for several attributes, such as customer income in sales data

Missing data may be due to

- equipment malfunction
- inconsistent with other recorded data and thus deleted
- data not entered due to misunderstanding
- certain data may not be considered important at the time of entry
- not register history or changes of the data

Missing data may need to be inferred

How to Handle Missing Data?

Ignore the tuple: usually done when class label is missing (when doing classification)—not effective when the % of missing values per attribute varies considerably

Fill in the missing value manually: tedious + infeasible?

Fill in it automatically with

- a global constant : e.g., “unknown”, a new class?!
- the attribute mean
- the attribute mean for all samples belonging to the same class: smarter
- the most probable value: inference-based such as Bayesian formula or decision tree

Noisy Data

Noise: random error or variance in a measured variable

Incorrect attribute values may be due to

- faulty data collection instruments
- data entry problems
- data transmission problems
- technology limitation
- inconsistency in naming convention

Other data problems which require data cleaning

- duplicate records
- incomplete data
- inconsistent data

How to Handle Noisy Data?

Binning

- first sort data and partition into (equal-frequency) bins
- then one can smooth by bin means, smooth by bin median, smooth by bin boundaries, etc.

Regression

- smooth by fitting the data into regression functions

Clustering

- detect and remove outliers

Combined computer and human inspection

- detect suspicious values and check by human (e.g., deal with possible outliers)

Data Cleaning as a Process

Data discrepancy detection

- Use metadata (e.g., domain, range, dependency, distribution)
- Check field overloading
- Check uniqueness rule, consecutive rule and null rule
- Use commercial tools
 - Data scrubbing: use simple domain knowledge (e.g., postal code, spell-check) to detect errors and make corrections
 - Data auditing: by analyzing data to discover rules and relationship to detect violators (e.g., correlation and clustering to find outliers)

Data migration and integration

- Data migration tools: allow transformations to be specified
- ETL (Extraction/Transformation/Loading) tools: allow users to specify transformations through a graphical user interface

Integration of the two processes

- Iterative and interactive (e.g., Potter's Wheels)

Data Integration

Data integration:

- Combines data from multiple sources into a coherent store

Schema integration: e.g., A.cust-id B.cust-#

- Integrate metadata from different sources

Entity identification problem:

- Identify real world entities from multiple data sources, e.g., Bill Clinton = William Clinton

Detecting and resolving data value conflicts

- For the same real world entity, attribute values from different sources are different
- Possible reasons: different representations, different scales, e.g., metric vs. British units

Handling Redundancy in Data Integration

Redundant data occur often when integration of multiple databases

- Object identification: The same attribute or object may have different names in different databases
- Derivable data: One attribute may be a “derived” attribute in another table, e.g., annual revenue

Redundant attributes may be able to be detected by correlation analysis and covariance analysis

Careful integration of the data from multiple sources may help reduce/avoid redundancies and inconsistencies and improve mining speed and quality

Correlation Analysis (Nominal Data)

X² (chi-square) test

$$\chi^2 = \sum \frac{(\textit{Observed} - \textit{Expected})^2}{\textit{Expected}}$$

The larger the X² value, the more likely the variables are related

The cells that contribute the most to the X² value are those whose actual count is very different from the expected count

Correlation does not imply causality

- # of hospitals and # of car-theft in a city are correlated
- Both are causally linked to the third variable: population

Chi-Square Calculation: An Example

	Play chess	Not play chess	Sum (row)
Like science fiction	250(90)	200(360)	450
Not like science fiction	50(210)	1000(840)	1050
Sum(col.)	300	1200	1500

Calculation

X² (chi-square) calculation (numbers in parenthesis are expected counts calculated based on the data distribution in the two categories)

$$\chi^2 = \frac{(250 - 90)^2}{90} + \frac{(50 - 210)^2}{210} + \frac{(200 - 360)^2}{360} + \frac{(1000 - 840)^2}{840} = 507.93$$

It shows that like_science_fiction and play_chess are correlated in the group

Correlation Analysis (Numeric Data)

Correlation coefficient (also called Pearson's product moment coefficient)

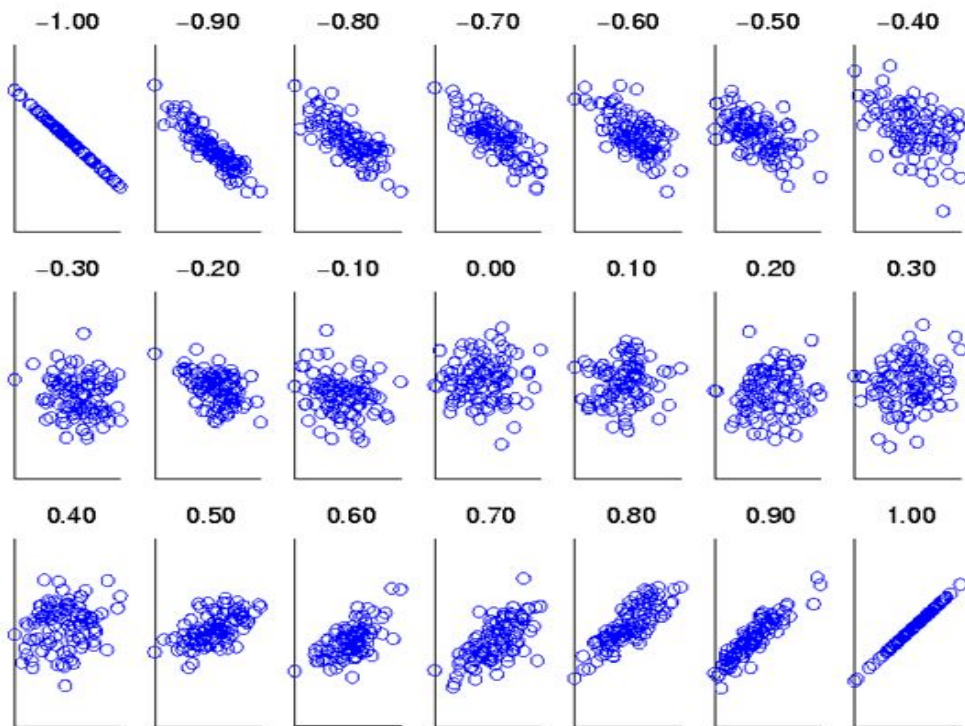
$$r_{A,B} = \frac{\sum_{i=1}^n (a_i - \bar{A})(b_i - \bar{B})}{(n-1)\sigma_A\sigma_B} = \frac{\sum_{i=1}^n (a_i b_i) - n\bar{A}\bar{B}}{(n-1)\sigma_A\sigma_B}$$

where n is the number of tuples, \bar{A} and \bar{B} are the respective means of A and B , σ_A and σ_B are the respective standard deviation of A and B , and $\sum(a_i b_i)$ is the sum of the AB cross-product.

If $r_{A,B} > 0$, A and B are positively correlated (A 's values increase as B 's). The higher, the stronger correlation.

$r_{A,B} = 0$: independent; $r_{AB} < 0$: negatively correlated

Visually Evaluating Correlation



Scatter plots showing the similarity from -1 to 1 .

Correlation (viewed as linear relationship)

Correlation measures the linear relationship between objects

To compute correlation, we standardize data objects, A and B, and then take their dot product

$$a'_k = (a_k - \text{mean}(A)) / \text{std}(A)$$

$$b'_k = (b_k - \text{mean}(B)) / \text{std}(B)$$

$$\text{correlation}(A, B) = A' \bullet B'$$

Covariance (Numeric Data)

Covariance is similar to correlation

$$Cov(A, B) = E((A - \bar{A})(B - \bar{B})) = \frac{\sum_{i=1}^n (a_i - \bar{A})(b_i - \bar{B})}{n}$$

Correlation coefficient: $r_{A,B} = \frac{Cov(A, B)}{\sigma_A \sigma_B}$

where n is the number of tuples, \bar{A} and \bar{B} are the respective mean or expected values of A and B , σ_A and σ_B are the respective standard deviation of A and B .

Positive covariance: If $Cov_{A,B} > 0$, then A and B both tend to be larger than their expected values.

Negative covariance: If $Cov_{A,B} < 0$ then if A is larger than its expected value, B is likely to be smaller than its expected value.

Independence: $Cov_{A,B} = 0$ but the converse is not true:

Some pairs of random variables may have a covariance of 0 but are not independent. Only under some additional assumptions (e.g., the data follow multivariate normal distributions) does a covariance of 0 imply independence

Co-Variance: An Example

$$Cov(A, B) = E((A - \bar{A})(B - \bar{B})) = \frac{\sum_{i=1}^n (a_i - \bar{A})(b_i - \bar{B})}{n}$$

- It can be simplified in computation as

$$Cov(A, B) = E(A \cdot B) - \bar{A}\bar{B}$$

- Suppose two stocks A and B have the following values in one week:
(2, 5), (3, 8), (5, 10), (4, 11), (6, 14).
- Question: If the stocks are affected by the same industry trends, will their prices rise or fall together?
 - $E(A) = (2 + 3 + 5 + 4 + 6) / 5 = 20/5 = 4$
 - $E(B) = (5 + 8 + 10 + 11 + 14) / 5 = 48/5 = 9.6$
 - $Cov(A, B) = (2 \times 5 + 3 \times 8 + 5 \times 10 + 4 \times 11 + 6 \times 14) / 5 - 4 \times 9.6 = 4$
- Thus, A and B rise together since $Cov(A, B) > 0$.