■ Lab – Handling CSV in Pandas (California Housing Dataset)

1. Reading a CSV File

We begin by importing Pandas and loading the California Housing dataset using pd.read_csv(). The dataset is stored in a DataFrame (df). df.head() shows the first 5 rows, giving a quick preview of the data.

```
import pandas as pd

file_path = "/usr/local/lib/python3.10/dist-packages/sklearn/datasets/data/california_housing_train.c
df = pd.read_csv(file_path)
print(df.head())
```

longitude latitude housing_median_age total_rooms total_bedrooms population households median_income median_
-114.31 34.19 15.0 5612.0 1283.0 1015.0 472.0 1.4936 66900.0
-114.47 34.40 19.0 7650.0 1901.0 1129.0 463.0 1.8200 80100.0
-114.56 33.69 17.0 720.0 174.0 333.0 117.0 1.6509 85700.0
-114.57 33.64 14.0 1501.0 337.0 515.0 226.0 3.1917 73400.0
-114.57 33.57 20.0 1454.0 326.0 624.0 262.0 1.9250 65500.0

2. Extracting the Contents of a CSV File

We explore the dataset structure: df.columns gives column names, df.shape gives rows × columns, df.describe() provides summary statistics.

```
df_housing = pd.read_csv(file_path)
print("Columns:", df_housing.columns.tolist())
print("Shape:", df_housing.shape)
print(df_housing.head())
```

Columns: ['longitude','latitude','housing_median_age','total_rooms','total_bedrooms','population','households','median_
Shape: (20640, 9)

3. Appending Data to a CSV

We can add new rows to an existing CSV file by saving first 100 rows and then appending the next 50 rows with to_csv().

```
df_subset = df_housing.head(100)
df_subset.to_csv("housing_subset.csv", index=False)

df_append = df_housing.iloc[100:150]
df_append.to_csv("housing_subset.csv", mode="a", header=False, index=False)

updated_df = pd.read_csv("housing_subset.csv")
print(updated_df.head())
```

After appending → first 5 rows of housing_subset.csv (same as original dataset's start).

4. Reading a CSV Chunk-by-Chunk

Large CSVs can be processed in smaller parts using chunksize. Each chunk loads only a portion of the file into memory.

```
chunk_iter = pd.read_csv(file_path, chunksize=5000)
for i, chunk in enumerate(chunk_iter):
    print(f"Chunk {i+1} → Shape: {chunk.shape}")
```

Chunk 1 → Shape: (5000, 9)
Chunk 2 → Shape: (5000, 9)
Chunk 3 → Shape: (5000, 9)
Chunk 4 → Shape: (2640, 9)

5. Writing Numeric Data into a CSV File

We can extract only numeric attributes such as longitude, latitude, median_income, and median_house_value.

```
numeric_data = df_housing[["longitude","latitude","median_income","median_house_value"]]
numeric_data.to_csv("housing_numeric.csv", index=False)
print(numeric_data.head())
```

longitude latitude median_income median_house_value
-114.31 34.19 1.4936 66900.0
-114.47 34.40 1.8200 80100.0
-114.56 33.69 1.6509 85700.0
-114.57 33.64 3.1917 73400.0
-114.57 33.57 1.9250 65500.0

6. Writing Text Data into a CSV File

We convert median_income into text categories (Low, Medium, High, Very High) and save it along with the numeric column.

```
df_housing["Income_Category"] = pd.cut(
    df_housing["median_income"],
    bins=[0,2,4,6,df_housing["median_income"].max()],
    labels=["Low","Medium","High","Very High"]
)

text_data = df_housing[["median_income","Income_Category"]]
text_data.to_csv("housing_text.csv", index=False)
print(text_data.head())
```

median_income Income_Category
1.4936 Low
1.8200 Low
1.6509 Low
3.1917 Medium
1.9250 Low