SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
Program Name: B. Tech		**Assignment Type: Lab**	**Academic Year:**2025-2026
Course Coordinator Name	Venkataramana Veeramsetty		
Instructor(s) Name	Dr. V. Venkataramana (Co-ordinator) Dr. T. Sampath Kumar Dr. Pramoda Patro Dr. Brij Kishor Tiwari Dr.J.Ravichander Dr. Mohammand Ali Shaik Dr. Anirodh Kumar Mr. S.Naresh Kumar Dr. RAJESH VELPULA Mr. Kundhan Kumar Ms. Ch.Rajitha Mr. M Prakash Mr. B.Raju Intern 1 (Dharma teja) Intern 2 (Sai Prasad) Intern 3 (Sowmya) NS_2 (Mounika)		
Course Code	24CS002PC215	**Course Title**	AI Assisted Coding
Year/Sem	II/I	**Regulation**	R24
Date and Day of Assignment	Week5 - Monday	**Time(s)**	
Duration	2 Hours	**Applicable to Batches**	
AssignmentNumber: 9.1(Present assignment number)/**24**(Total number of assignments)			

Q.No.	Question	*Expected Time to complete*
1	**Lab 17– AI for Data Processing: Data cleaning and preprocessing scripts** The objective of this lab is to enable students to understand and apply **AI-assisted coding tools** for automating and enhancing data preprocessing tasks. Students will: 1. Gain practical experience in **cleaning, transforming, and standardizing real-world datasets** with issues such as missing	Week 9-Monday

values, duplicates, outliers, inconsistent formats, and noisy text.
2. Learn to **leverage AI coding assistants** to generate preprocessing scripts, while critically evaluating and refining the AI-generated code for accuracy, efficiency, and best practices.
3. Develop the ability to design **end-to-end preprocessing pipelines** that prepare raw data for downstream machine learning and analytics applications.
4. Build confidence in **combining human expertise with AI assistance**, ensuring data quality and integrity in diverse domains such as customer feedback, healthcare, and finance.

Lab Question 1: Customer Feedback Dataset

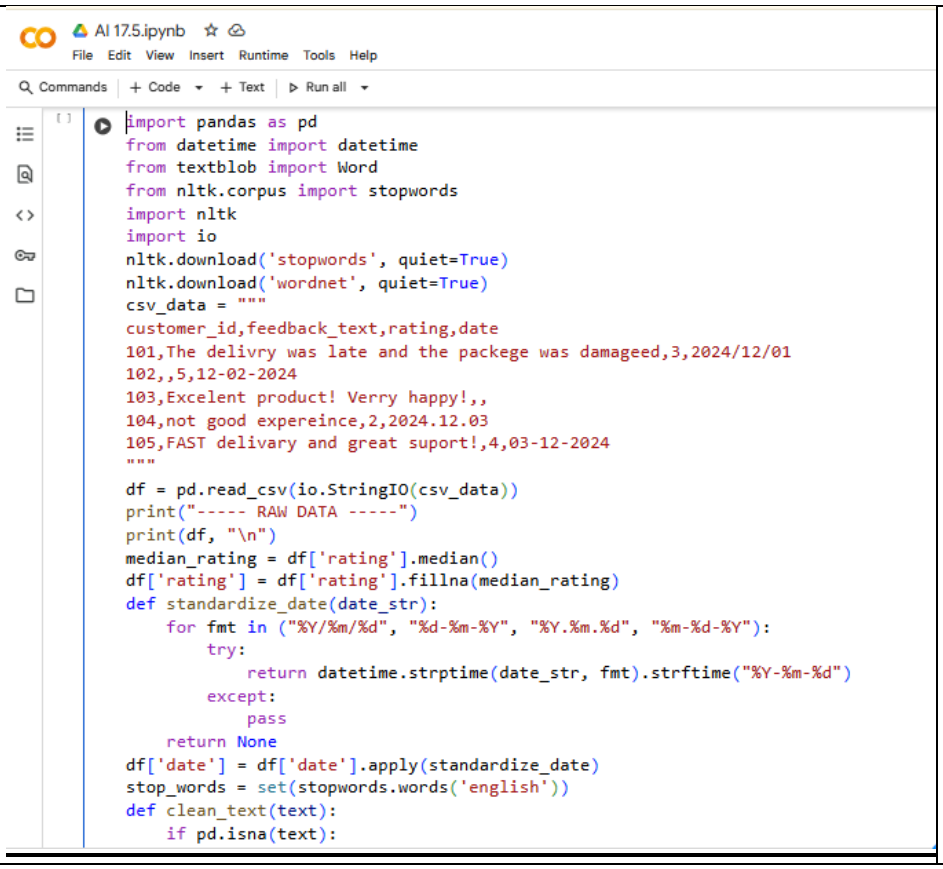
You are given a CSV file containing customer feedback collected from an e-commerce website. The dataset includes columns: customer_id, feedback_text, rating, and date. However, the file has many missing values, typos, and inconsistent date formats.

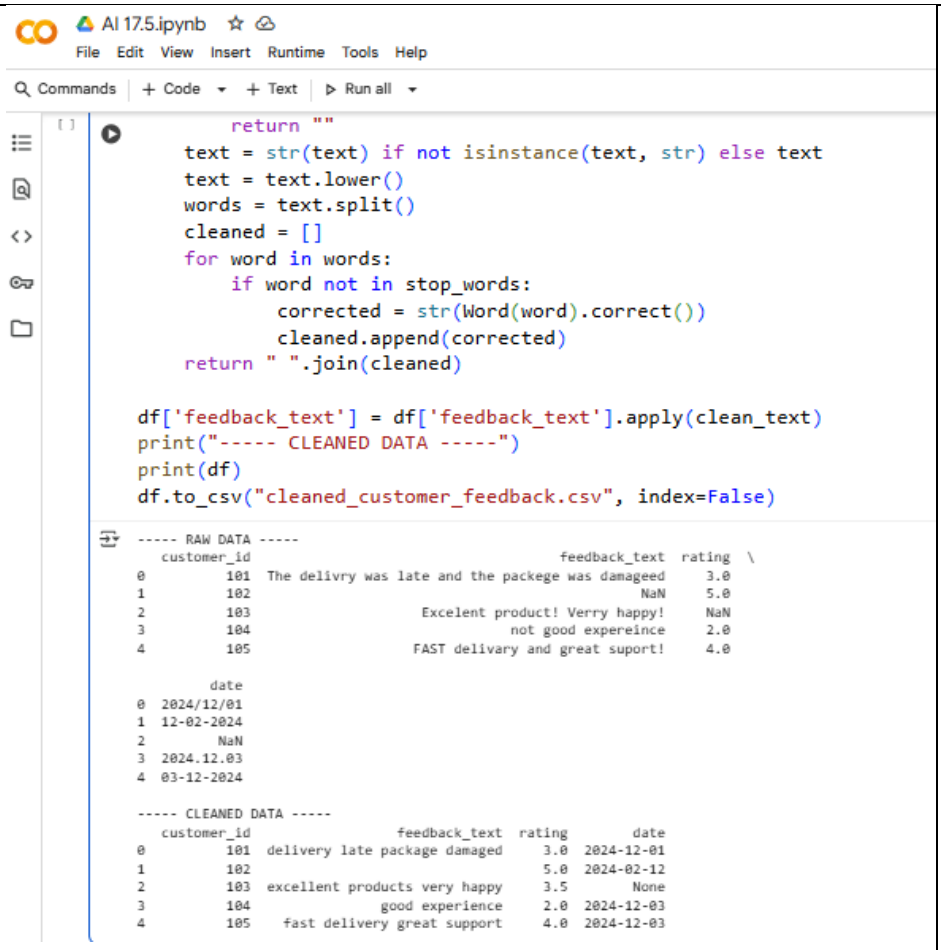
- **Task 1:** Use an AI-assisted coding tool to generate a script that detects and fills missing rating values with the column's median and standardizes the date column into YYYY-MM-DD format.
- **Task 2:** Clean the feedback_text column by removing stopwords, correcting common spelling mistakes, and converting text to lowercase using AI suggestions. Compare the AI-generated preprocessing code with your manually written version.

Prompt:

Create a Python script that tidies up a dataset by doing three things:
- Replace any missing ratings with the median value from that column.
- Standardize all date entries to the YYYY-MM-DD format.
- Clean up the feedback text by converting it to lowercase, removing stopwords, and correcting common spelling errors.

Code:


AI 17.5.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
import pandas as pd
from datetime import datetime
from textblob import Word
from nltk.corpus import stopwords
import nltk
import io
nltk.download('stopwords', quiet=True)
nltk.download('wordnet', quiet=True)
csv_data = """
customer_id,feedback_text,rating,date
101,The delivry was late and the packege was damageed,3,2024/12/01
102,,5,12-02-2024
103,Excelent product! Verry happy!,,
104,not good expereince,2,2024.12.03
105,FAST delivary and great suport!,4,03-12-2024
"""
df = pd.read_csv(io.StringIO(csv_data))
print("----- RAW DATA -----")
print(df, "\n")
median_rating = df['rating'].median()
df['rating'] = df['rating'].fillna(median_rating)
def standardize_date(date_str):
for fmt in ("%Y/%m/%d", "%d-%m-%Y", "%Y.%m.%d", "%m-%d-%Y"):
try:
return datetime.strptime(date_str, fmt).strftime("%Y-%m-%d")
except:
pass
return None
df['date'] = df['date'].apply(standardize_date)
stop_words = set(stopwords.words('english'))
def clean_text(text):
if pd.isna(text):



Comparison:

AI-generated version: Quick to produce and works well for straightforward or small-scale data tasks.

Manually refined version: More reliable and easier to understand, with better error handling — perfect for production use or detailed reviews.

Lab Question 2: Medical Records Dataset

A hospital provides you with a dataset of anonymized medical records containing attributes like patient_id, age, gender, blood_pressure, and cholesterol. Some columns include outliers and inconsistent categorical labels (e.g., Male, M, male).

- **Task 1:** Write a script (with AI assistance) to detect and handle outliers in the blood_pressure column using statistical methods (e.g., IQR or z-score).

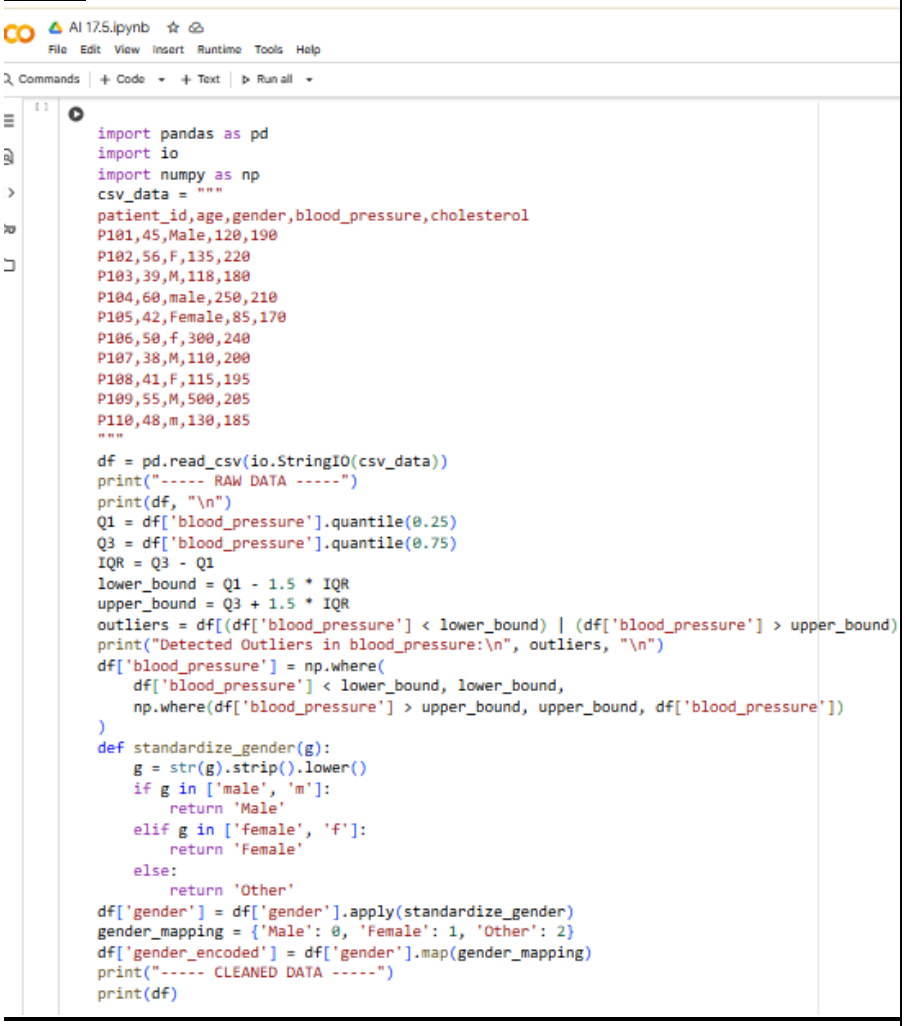
- **Task 2:** Standardize categorical values in the gender column and encode them into numeric form. Let the AI-assisted coding tool propose the preprocessing pipeline, then refine the pipeline manually based on your understanding.

Prompt:

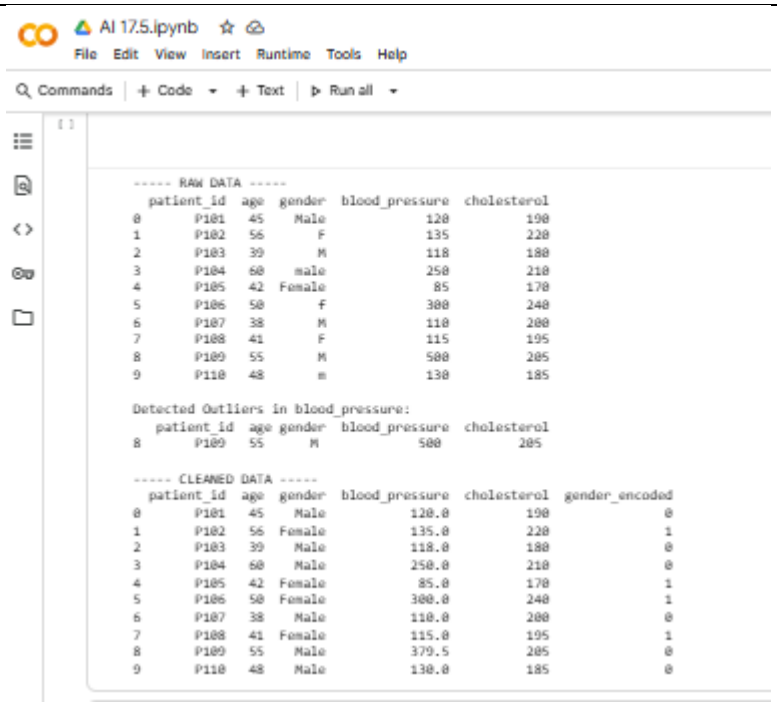
Create a Python script to clean up a medical dataset by doing two things:

- Identify and handle outliers using the Interquartile Range (IQR) method.
- Fix inconsistent gender labels (like "Male", "M", "male") by standardizing them and then converting them into numeric values for analysis.

Code:




```
import pandas as pd
import io
import numpy as np
csv_data = """
patient_id,age,gender,blood_pressure,cholesterol
P101,45,Male,120,190
P102,56,F,135,220
P103,39,M,118,180
P104,60,male,250,210
P105,42,Female,85,170
P106,50,f,300,240
P107,38,M,110,200
P108,41,F,115,195
P109,55,M,500,205
P110,48,m,130,185
"""
df = pd.read_csv(io.StringIO(csv_data))
print("----- RAW DATA -----")
print(df, "\n")
Q1 = df['blood_pressure'].quantile(0.25)
Q3 = df['blood_pressure'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
outliers = df[(df['blood_pressure'] < lower_bound) | (df['blood_pressure'] > upper_bound)]
print("Detected Outliers in blood_pressure:\n", outliers, "\n")
df['blood_pressure'] = np.where(
    df['blood_pressure'] < lower_bound, lower_bound,
    np.where(df['blood_pressure'] > upper_bound, upper_bound, df['blood_pressure'])
)
def standardize_gender(g):
    g = str(g).strip().lower()
    if g in ['male', 'm']:
        return 'Male'
    elif g in ['female', 'f']:
        return 'Female'
    else:
        return 'Other'
df['gender'] = df['gender'].apply(standardize_gender)
gender_mapping = {'Male': 0, 'Female': 1, 'Other': 2}
df['gender_encoded'] = df['gender'].map(gender_mapping)
print("----- CLEANED DATA -----")
print(df)
```



Lab Question 3: Financial Transactions Dataset

A bank gives you transaction data with columns: transaction_id, amount, currency, timestamp, and merchant. The dataset contains multiple issues: different currency units (USD, INR, EUR), timestamps in various time zones, and duplicated rows.

- **Task 1:** Use AI-assisted coding to write a script that removes duplicate transactions and converts all amount values into a single currency (e.g., USD) using a provided conversion dictionary.
- **Task 2:** Normalize the timestamp column into UTC format and create a new column transaction_hour for downstream time-series analysis. Compare the AI's preprocessing code against your own optimized version.

Prompt:

Build a Python script to clean and standardize a financial dataset by doing the following:

1. Eliminate any duplicate transaction records.
2. Convert all transaction amounts to USD using the provided exchange rates.
3. Normalize all timestamps to UTC format and add a new column that captures just the hour of each transaction.

Code:

AI 17.5.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

```
import pandas as pd
import io
from datetime import datetime
import pytz
csv_data = """
transaction_id,amount,currency,timestamp,merchant
T001,100,USD,2024-12-01 10:00:00-05:00,Amazon
T002,8300,INR,2024/12/01 20:30:00+05:30,Flipkart
T003,90,EUR,01-12-2024 15:00:00+01:00,eBay
T004,100,USD,2024-12-01 10:00:00-05:00,Amazon
T005,5000,INR,2024.12.01 22:00:00+05:30,BigBasket
T006,85,EUR,2024-12-01T14:00:00+01:00,Zalando
T007,100,USD,2024-12-01 10:00:00-05:00,Amazon
"""
df = pd.read_csv(io.StringIO(csv_data))
print("----- RAW DATA -----")
print(df, "\n")
df = df.drop_duplicates(subset=['transaction_id', 'amount', 'currency', 'timestamp', 'merchant'])
conversion_rates = {
    'USD': 1.0,
    'INR': 0.012,
    'EUR': 1.1
}
def convert_to_usd(amount, currency):
    rate = conversion_rates.get(currency.upper(), 1)
    return round(amount * rate, 2)
df['amount_usd'] = df.apply(lambda x: convert_to_usd(x['amount'], x['currency']), axis=1)
def normalize_to_utc(ts):
    try:
        dt = pd.to_datetime(ts, utc=True)
        return dt
    except Exception:
```

AI 17.5.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

```
        return None
df['timestamp_utc'] = df['timestamp'].apply(normalize_to_utc)
df['transaction_hour'] = df['timestamp_utc'].dt.hour
print("----- CLEANED DATA -----")
print(df[['transaction_id', 'amount', 'currency', 'amount_usd', 'timestamp_utc', 'transaction_hour', 'merchant'
```

```
----- RAW DATA -----
  transaction_id  amount currency                  timestamp   merchant
0           T001     100      USD  2024-12-01 10:00:00-05:00     Amazon
1           T002    8300      INR  2024/12/01 20:30:00+05:30   Flipkart
2           T003      90      EUR  01-12-2024 15:00:00+01:00      eBay
3           T004     100      USD  2024-12-01 10:00:00-05:00     Amazon
4           T005    5000      INR  2024.12.01 22:00:00+05:30  BigBasket
5           T006      85      EUR  2024-12-01T14:00:00+01:00    Zalando
6           T007     100      USD  2024-12-01 10:00:00-05:00     Amazon

----- CLEANED DATA -----
  transaction_id  amount currency  amount_usd             timestamp_utc  \
0           T001     100      USD       100.0 2024-12-01 15:00:00+00:00
1           T002    8300      INR        99.6 2024-12-01 15:00:00+00:00
2           T003      90      EUR        99.0 2024-01-12 14:00:00+00:00
3           T004     100      USD       100.0 2024-12-01 15:00:00+00:00
4           T005    5000      INR        60.0 2024-12-01 16:30:00+00:00
5           T006      85      EUR        93.5 2024-12-01 13:00:00+00:00
6           T007     100      USD       100.0 2024-12-01 15:00:00+00:00

   transaction_hour   merchant
0                15     Amazon
1                15    Flipkart
2                14       eBay
3                15     Amazon
4                16   BigBasket
5                13    Zalando
6                15     Amazon
```