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:

Write a Python program that generates a customer feedback dataset, fills missing ratings with the median, standardizes date formats to YYYY-MM-DD, and cleans feedback text by lowercasing, correcting spelling mistakes, and removing stopwords.

Code:

```python
import pandas as pd
from dateutil import parser
from textblob import TextBlob
from nltk.corpus import stopwords
import nltk
nltk.download('stopwords', quiet=True)
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

# Step 1: Create sample dataset
data = {
    "customer_id": [101, 102, 103, 104, 105],
    "feedback_text": [
        "I realy love this product! its amazng.",
        "bad quallity and very late delivery",
        "Good prodct but packaging was bad.",
        "Excelent service and fast shipping!",
        "Not worth teh money at all"
    ],
    "rating": [5, None, 4, None, 2],
    "date": ["2025/10/01", "01-10-2025", "10-01-2025", "Oct 1 2025", "2025.10.01"]
}

df = pd.DataFrame(data)

print(" Original Data:")
print(df)

# Task 1: Fill missing ratings & standardize date
df['rating'].fillna(df['rating'].median(), inplace=True)

# Standardize date column
def standardize_date(date_str):
    try:
        return parser.parse(str(date_str)).strftime("%Y-%m-%d")
    except:
        return pd.NaT

df['date'] = df['date'].apply(standardize_date)

# Task 2: Clean feedback text
stop_words = set(stopwords.words('english'))

def clean_feedback(text):
    if pd.isna(text):
        return ""
    text = text.lower()  # lowercase
    corrected = str(TextBlob(text).correct())  # spelling correction
    filtered = " ".join([word for word in corrected.split() if word not in stop_words])
    return filtered

df['feedback_text'] = df['feedback_text'].apply(clean_feedback)

# Final cleaned dataset
print("\n Cleaned Data:")
print(df)
```

```
 Original Data:
   customer_id                        feedback_text  rating        date
0          101  I realy love this product! its amazng.     5.0  2025/10/01
1          102     bad quallity and very late delivery     NaN  01-10-2025
2          103      Good prodct but packaging was bad.     4.0  10-01-2025
3          104     Excelent service and fast shipping!     NaN  Oct 1 2025
4          105              Not worth teh money at all     2.0  2025.10.01

 Cleaned Data:
   customer_id                    feedback_text  rating        date
0          101     really love product! amazing.     5.0  2025-10-01
1          102         bad quality late delivery     4.0  2025-01-10
2          103         good product packing bad.     4.0  2025-10-01
3          104  excellent service fast shipping!     4.0  2025-10-01
4          105                       worth money     2.0  2025-10-01
```

Comparison:

- **AI-generated code** is easier and faster to write. It automatically fixes spelling, formats dates, and fills missing values using smart libraries.

- **manual code** takes more time to write but helps you understand each step clearly. It does the same work in a simpler way but without advanced features like spell correction.

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:

Write a Python program that loads medical record data, detects and caps outliers in the blood_pressure column using the IQR method, standardizes gender labels, and encodes them into numeric form. Display both raw and cleaned datasets.

Code:

```python
import pandas as pd
import numpy as np
import io

csv_data = """
patient_id,age,gender,blood_pressure,cholesterol
P201,30,Male,118,180
P202,47,F,140,220
P203,52,M,135,210
P204,60,Female,260,300
P205,45,m,122,195
P206,50,M,310,250
P207,37,f,600,234
P208,65,FEMALE,290,310
P209,49,male,150,240
P210,41,f,130,200
"""

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)
```

```
----- RAW DATA -----
  patient_id  age  gender  blood_pressure  cholesterol
0       P201   30    Male             118          180
1       P202   47       F             140          220
2       P203   52       M             135          210
3       P204   60  Female             260          300
4       P205   45       m             122          195
5       P206   50       M             310          250
6       P207   37       f             600          234
7       P208   65  FEMALE             290          310
8       P209   49    male             150          240
9       P210   41       f             130          200

Detected Outliers in blood_pressure:
   patient_id  age gender  blood_pressure  cholesterol
6       P207   37      f             600          234

----- CLEANED DATA -----
  patient_id  age  gender  blood_pressure  cholesterol  gender_encoded
0       P201   30    Male         118.000          180               0
1       P202   47  Female         140.000          220               1
2       P203   52    Male         135.000          210               0
3       P204   60  Female         260.000          300               1
4       P205   45    Male         122.000          195               0
5       P206   50    Male         310.000          250               0
6       P207   37  Female         509.375          234               1
7       P208   65  Female         290.000          310               1
8       P209   49    Male         150.000          240               0
9       P210   41  Female         130.000          200               1
```

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:

Write a Python program that removes duplicate transactions, converts all amounts to USD using a conversion dictionary, normalizes timestamps to UTC, and adds a transaction_hour column for analysis.

Code:

```
import pandas as pd
import io
from datetime import datetime
import pytz

csv_data = """
transaction_id,amount,currency,timestamp,merchant
T001,100,USD,2025-10-27 10:30:00-0400,Amazon
T002,8500,INR,2025-10-27 15:00:00+0530,Flipkart
T003,90,EUR,2025-10-27 14:00:00+0100,eBay
T004,100,USD,2025-10-27 10:30:00-0400,Amazon
T005,120,USD,2025-10-27 09:00:00-0400,Target
"""

df = pd.read_csv(io.StringIO(csv_data))
print("----- RAW DATA -----")
print(df, "\n")

# ----- Task 1: Remove Duplicates and Convert Currency -----
df = df.drop_duplicates()

conversion_rates = {
    "USD": 1.0,
    "INR": 0.012,   # 1 INR = 0.012 USD
    "EUR": 1.1      # 1 EUR = 1.1 USD
}
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: x['amount'] * conversion_rates[x['currency']], axis=1)
def normalize_to_utc(ts):
  try:
    dt = pd.to_datetime(ts, utc=True)
    return dt
  except Exception:
    return None
```

```
    return None
df['timestamp'] = df['timestamp'].apply(normalize_to_utc)

# ----- Task 2: Normalize Timestamps and Extract Transaction Hour -----
df['timestamp'] = pd.to_datetime(df['timestamp'], utc=True)
df['transaction_hour'] = df['timestamp'].dt.hour

print("----- CLEANED DATA -----")
print(df)
```

```
----- RAW DATA -----
  transaction_id  amount currency                 timestamp  merchant
0           T001     100      USD  2025-10-27 10:30:00-0400    Amazon
1           T002    8500      INR  2025-10-27 15:00:00+0530  Flipkart
2           T003      90      EUR  2025-10-27 14:00:00+0100       eBay
3           T004     100      USD  2025-10-27 10:30:00-0400    Amazon
4           T005     120      USD  2025-10-27 09:00:00-0400    Target

----- CLEANED DATA -----
  transaction_id  amount currency                 timestamp  merchant  \
0           T001     100      USD 2025-10-27 14:30:00+00:00    Amazon
1           T002    8500      INR 2025-10-27 09:30:00+00:00  Flipkart
2           T003      90      EUR 2025-10-27 13:00:00+00:00       eBay
3           T004     100      USD 2025-10-27 14:30:00+00:00    Amazon
4           T005     120      USD 2025-10-27 13:00:00+00:00    Target

   amount_usd  transaction_hour
0       100.0                14
1       102.0                 9
2        99.0                13
3       100.0                14
4       120.0                13
```