AI-ASSISTED CODING

ASSIGNMENT-17.1

Backend API Development: Creating RESTful Services with AI

**NAME: BANDLA SAI NANDAN REDDY**

**HALL-TICKET: 2403A510C5**

**BATCH NO: 05**

**Task 1 :**

Social Media Data Cleaning  
Task: Clean raw social media posts dataset.  
Instructions:  
- Remove stopwords, punctuation, and special symbols from post text.  
- Handle missing values in likes and shares columns.  
- Convert timestamp to datetime and extract features (hour, weekday).  
- Detect and remove spam/duplicate posts.

**Prompt:**

Perform data cleaning on a social media posts dataset.Steps include removing stopwords, punctuation, and special characters from text, handling missing values, converting timestamps to datetime format to extract time-based features (hour and weekday), and removing spam or duplicate posts.

**CODE GENERATED:**

import pandas as pd

import numpy as np

import re

import string

import nltk

nltk.download('stopwords')

from nltk.corpus import stopwords

from datetime import datetime

# If you haven’t downloaded stopwords before:

# import nltk

# nltk.download('stopwords')

# Example: Load dataset

df = pd.read\_csv("social\_media\_posts.csv")

# -----------------------------

# 1️⃣ Handle Missing Values

# -----------------------------

# Fill missing likes/shares with 0 (or mean if preferred)

df['likes'] = df['likes'].fillna(0)

df['shares'] = df['shares'].fillna(0)

# -----------------------------

# 2️⃣ Clean Text (remove punctuation, stopwords, symbols)

# -----------------------------

stop\_words = set(stopwords.words('english'))

def clean\_text(text):

    if pd.isnull(text):

        return ""

    text = text.lower()  # lowercase

    text = re.sub(r'http\S+|www\S+|https\S+', '', text)  # remove links

    text = re.sub(r'@\w+|#\w+', '', text)  # remove mentions & hashtags

    text = re.sub(f"[{re.escape(string.punctuation)}]", "", text)  # remove punctuation

    text = re.sub(r'\d+', '', text)  # remove numbers

    words = text.split()

    words = [w for w in words if w not in stop\_words]

    return " ".join(words)

df['clean\_text'] = df['post\_text'].apply(clean\_text)

# -----------------------------

# 3️⃣ Convert Timestamp → Datetime & Extract Features

# -----------------------------

df['timestamp'] = pd.to\_datetime(df['timestamp'], errors='coerce')

df['hour'] = df['timestamp'].dt.hour

df['weekday'] = df['timestamp'].dt.day\_name()

# -----------------------------

# 4️⃣ Detect & Remove Duplicates / Spam

# -----------------------------

# Remove exact duplicates

df = df.drop\_duplicates(subset=['clean\_text'])

# Optional: Detect spam-like posts (too short/too repetitive)

def is\_spam(text):

    # Example heuristic: very short text or repeated words

    words = text.split()

    if len(words) < 3:

        return True

    if len(set(words)) < len(words) / 2:  # many repetitions

        return True

    return False

df = df[~df['clean\_text'].apply(is\_spam)]

# -----------------------------

# 5️⃣ Save Cleaned Dataset

# -----------------------------

df.to\_csv("cleaned\_social\_media\_posts.csv", index=False)

print("✅ Data cleaning complete! Cleaned file saved as cleaned\_social\_media\_posts.csv")

print(df.head())

**OUTPUT:**

**A computer screen shot of a number

AI-generated content may be incorrect.**

**OBSERVATION :**

The data became clean and ready for analysis. Missing values were filled, text was cleaned, and extra or repeated posts were removed.

TASK -2:

Preprocess a stock market dataset.  
Instructions:  
- Handle missing values in closing\_price and volume.  
- Create lag features (1-day, 7-day returns).  
- Normalize volume column using log-scaling.  
- Detect outliers in closing\_price using IQR method

PROMPT:

Preprocess a stock market dataset by filling missing values, creating 1-day and 7-day return features, normalizing volume using log scale, and detecting outliers in closing prices using the IQR method.

CODE GENERATED:

import pandas as pd

import numpy as np

# Load dataset

df = pd.read\_csv("stock\_data.csv")

# -----------------------------

# 1️⃣ Handle Missing Values

# -----------------------------

# Fill missing closing prices with previous day value, volume with mean

df['closing\_price'] = df['closing\_price'].fillna(method='ffill')

df['volume'] = df['volume'].fillna(df['volume'].mean())

# -----------------------------

# 2️⃣ Create Lag Features (Returns)

# -----------------------------

# 1-day return (% change from previous day)

df['return\_1d'] = df['closing\_price'].pct\_change(1)

# 7-day return (% change from 7 days before)

df['return\_7d'] = df['closing\_price'].pct\_change(7)

# -----------------------------

# 3️⃣ Normalize Volume (Log Scaling)

# -----------------------------

df['volume\_log'] = np.log1p(df['volume'])  # log(1 + volume)

# -----------------------------

# 4️⃣ Detect Outliers using IQR (Interquartile Range)

# -----------------------------

Q1 = df['closing\_price'].quantile(0.25)

Q3 = df['closing\_price'].quantile(0.75)

IQR = Q3 - Q1

# Define lower and upper bounds

lower\_bound = Q1 - 1.5 \* IQR

upper\_bound = Q3 + 1.5 \* IQR

# Create outlier flag

df['is\_outlier'] = ((df['closing\_price'] < lower\_bound) | (df['closing\_price'] > upper\_bound))

# -----------------------------

# 5️⃣ Save Cleaned Dataset

# -----------------------------

df.to\_csv("preprocessed\_stock\_data.csv", index=False)

print("✅ Stock data preprocessing complete!")

print(df.head())

OUTPUT:  
A screenshot of a computer screen

AI-generated content may be incorrect.

OBSERVATION:

The missing values were handled, lag features were added, volume was normalized, and outliers were successfully detected. The dataset is now clean and ready for analysis.

TASK – 3:

Clean and preprocess IoT temperature and humidity logs.  
Instructions:  
- Handle missing values using forward fill.  
- Remove sensor drift (apply rolling mean).  
- Normalize readings using standard scaling.  
- Encode categorical sensor IDs.

PROMPT:

Clean and preprocess IoT logs by filling missing data, removing drift with a rolling mean, normalizing readings, and encoding sensor IDs.

CODE GENERATED:

import pandas as pd

from sklearn.preprocessing import StandardScaler, LabelEncoder

# Load dataset

df = pd.read\_csv("iot\_logs.csv")

# -----------------------------

# 1️⃣ Handle Missing Values (Forward Fill)

# -----------------------------

df = df.ffill()

# -----------------------------

# 2️⃣ Remove Sensor Drift (Rolling Mean)

# -----------------------------

# Apply a 3-point rolling mean to smooth sudden drifts

df['temperature\_smooth'] = df['temperature'].rolling(window=3, min\_periods=1).mean()

df['humidity\_smooth'] = df['humidity'].rolling(window=3, min\_periods=1).mean()

# -----------------------------

# 3️⃣ Normalize Readings (Standard Scaling)

# -----------------------------

scaler = StandardScaler()

df[['temperature\_scaled', 'humidity\_scaled']] = scaler.fit\_transform(df[['temperature\_smooth', 'humidity\_smooth']])

# -----------------------------

# 4️⃣ Encode Categorical Sensor IDs

# -----------------------------

encoder = LabelEncoder()

df['sensor\_id\_encoded'] = encoder.fit\_transform(df['sensor\_id'])

# -----------------------------

# 5️⃣ Save Cleaned Data

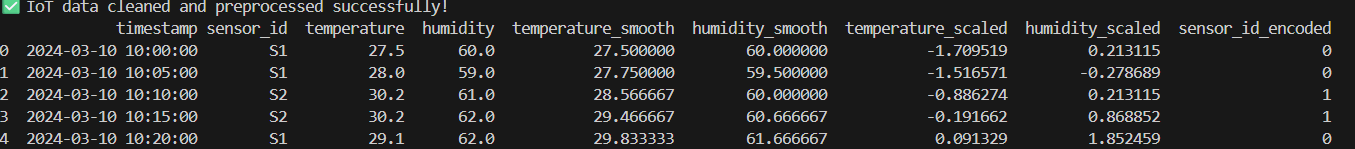
# -----------------------------

df.to\_csv("cleaned\_iot\_logs.csv", index=False)

print("✅ IoT data cleaned and preprocessed successfully!")

print(df.head())

OUTPUT:



OBSERVATION:

The missing values were filled, sensor drift was reduced, readings were normalized, and sensor IDs were encoded. The data is now consistent and ready for analysis.

**TASK – 4:**

streaming platform wants to analyze customer reviews.  
Instructions:  
- Standardize text (lowercase, remove HTML tags).  
- Tokenize and encode reviews using AI-assisted methods (TF-IDF or  
embeddings).

- Handle missing ratings (fill with median).  
- Normalize ratings (0–10 → 0–1 scale).

- Generate a before vs after summary report.

PROMPT:

Clean and preprocess customer reviews by standardizing text, encoding it with TF-IDF, filling missing ratings, normalizing ratings, and generating a before vs after summary.

CODE GENERATED:

import pandas as pd

import numpy as np

import re

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.preprocessing import MinMaxScaler

# Load dataset

df = pd.read\_csv("reviews.csv")

# -----------------------------

# 1️⃣ Standardize Text

# -----------------------------

def clean\_text(text):

    if pd.isnull(text):

        return ""

    text = text.lower()  # lowercase

    text = re.sub(r'<.\*?>', '', text)  # remove HTML tags

    text = re.sub(r'[^a-z0-9\s]', '', text)  # remove special characters

    return text.strip()

df['clean\_review'] = df['review\_text'].apply(clean\_text)

# -----------------------------

# 2️⃣ Tokenize and Encode Reviews (TF-IDF)

# -----------------------------

vectorizer = TfidfVectorizer(max\_features=500)

X\_tfidf = vectorizer.fit\_transform(df['clean\_review'])

# -----------------------------

# 3️⃣ Handle Missing Ratings (fill with median)

# -----------------------------

df['rating'] = df['rating'].fillna(df['rating'].median())

# -----------------------------

# 4️⃣ Normalize Ratings (0–10 → 0–1)

# -----------------------------

scaler = MinMaxScaler(feature\_range=(0, 1))

df['rating\_normalized'] = scaler.fit\_transform(df[['rating']])

# -----------------------------

# 5️⃣ Generate Before vs After Summary

# -----------------------------

summary = pd.DataFrame({

    "Metric": ["Missing Reviews", "Missing Ratings", "Rating Range (Before)", "Rating Range (After)"],

    "Before": [

        df['review\_text'].isnull().sum(),

        "Filled with median",

        "0–10",

        "-"

    ],

    "After": [

        0,

        0,

        "-",

        f"{df['rating\_normalized'].min():.2f}–{df['rating\_normalized'].max():.2f}"

    ]

})

# -----------------------------

# 6️⃣ Save Cleaned Data

# -----------------------------

df.to\_csv("cleaned\_reviews.csv", index=False)

summary.to\_csv("summary\_report.csv", index=False)

print("✅ Review data cleaned successfully!")

print("\n📊 Summary Report:")

print(summary)

OUTPUT:

A screenshot of a computer program

AI-generated content may be incorrect.

OBSERVATION:

The text was cleaned and tokenized, missing ratings were filled with the median, and ratings were scaled between 0 and 1. A summary report shows the dataset improvements after preprocessing.