AI ASSISTED CODING LAB

ASSIGNMENT-17.4

ENROLLMENT NO:2503A51L10

BATCH NO: 19

NAME: K Praneeth

TASK DESCRIPTION 1:

Use AI to generate a Python script for cleaning an employee dataset.

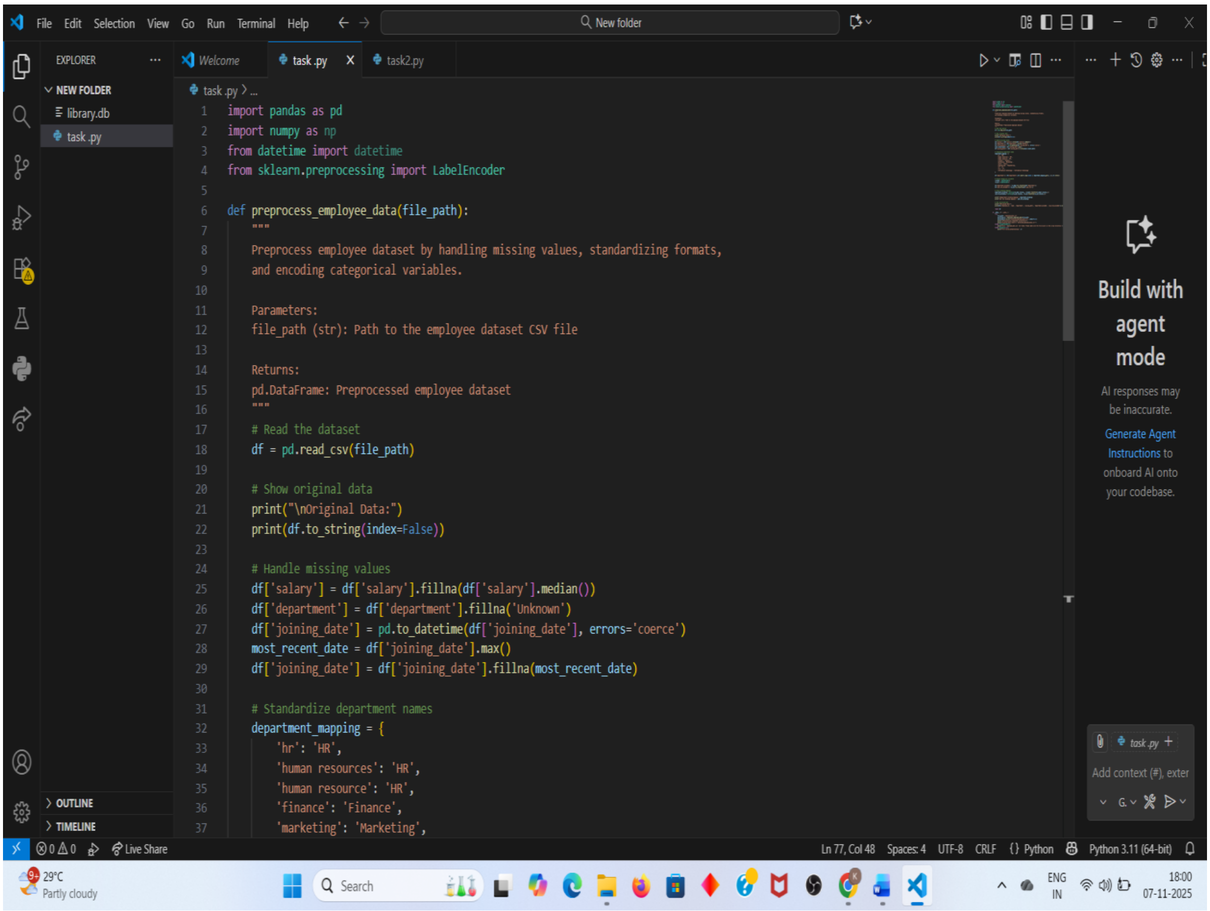
Instructions:

- Handle missing values in columns (salary, department, joining_date).
- Convert the "joining_date" column into proper datetime format.
- Standardize department names (e.g., "HR", "hr", "Human Resources" → "HR").
- Encode categorical variables (department, job_role).

PROMPT :

Generate a Python script using Pandas to clean an employee dataset by handling missing values, formatting dates, standardizing department names, and encoding categorical columns.

CODE GENERATED :

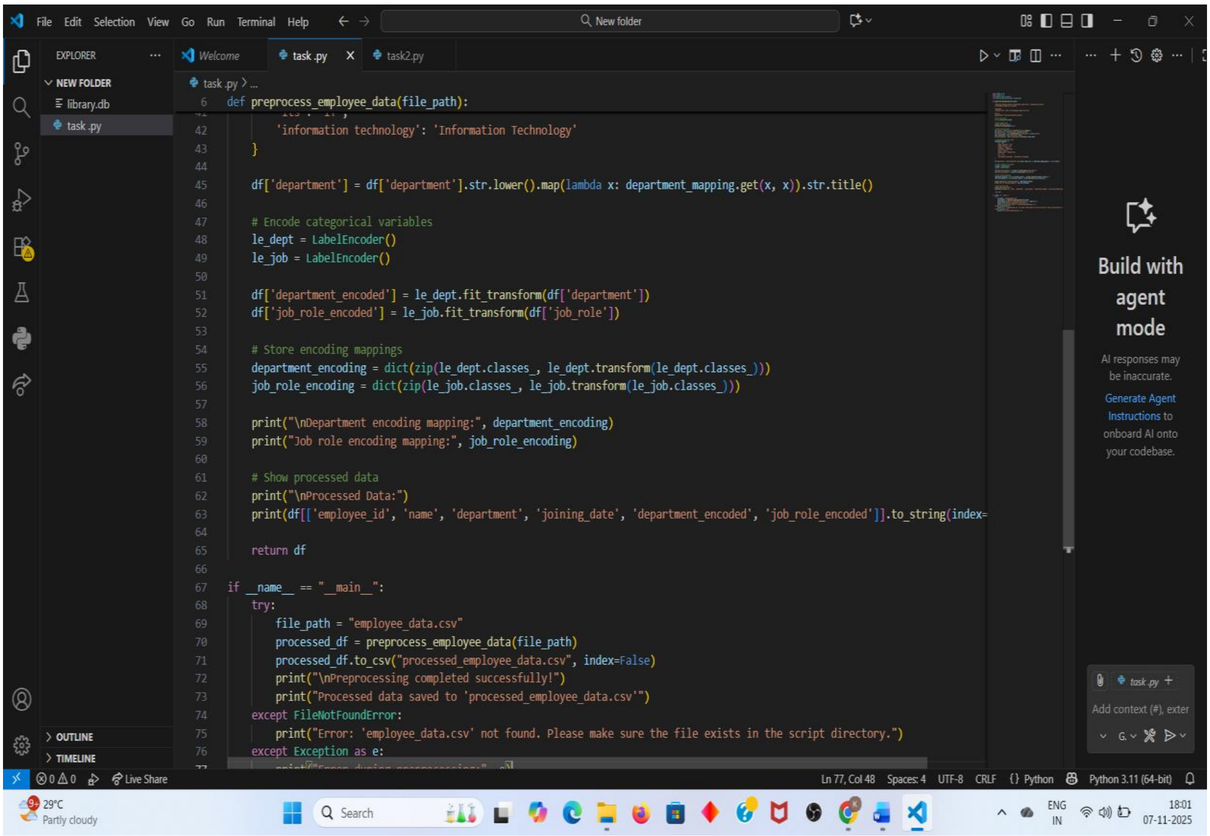


```python
import pandas as pd
import numpy as np
from datetime import datetime
from sklearn.preprocessing import LabelEncoder

def preprocess_employee_data(file_path):
    """
    Preprocess employee dataset by handling missing values, standardizing formats,
    and encoding categorical variables.

    Parameters:
    file_path (str): Path to the employee dataset CSV file

    Returns:
    pd.DataFrame: Preprocessed employee dataset
    """
    # Read the dataset
    df = pd.read_csv(file_path)

    # Show original data
    print("\nOriginal Data:")
    print(df.to_string(index=False))

    # Handle missing values
    df['salary'] = df['salary'].fillna(df['salary'].median())
    df['department'] = df['department'].fillna('Unknown')
    df['joining_date'] = pd.to_datetime(df['joining_date'], errors='coerce')
    most_recent_date = df['joining_date'].max()
    df['joining_date'] = df['joining_date'].fillna(most_recent_date)

    # Standardize department names
    department_mapping = {
        'hr': 'HR',
        'human resources': 'HR',
        'human resource': 'HR',
        'finance': 'Finance',
        'marketing': 'Marketing',
```

Build with agent mode

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.



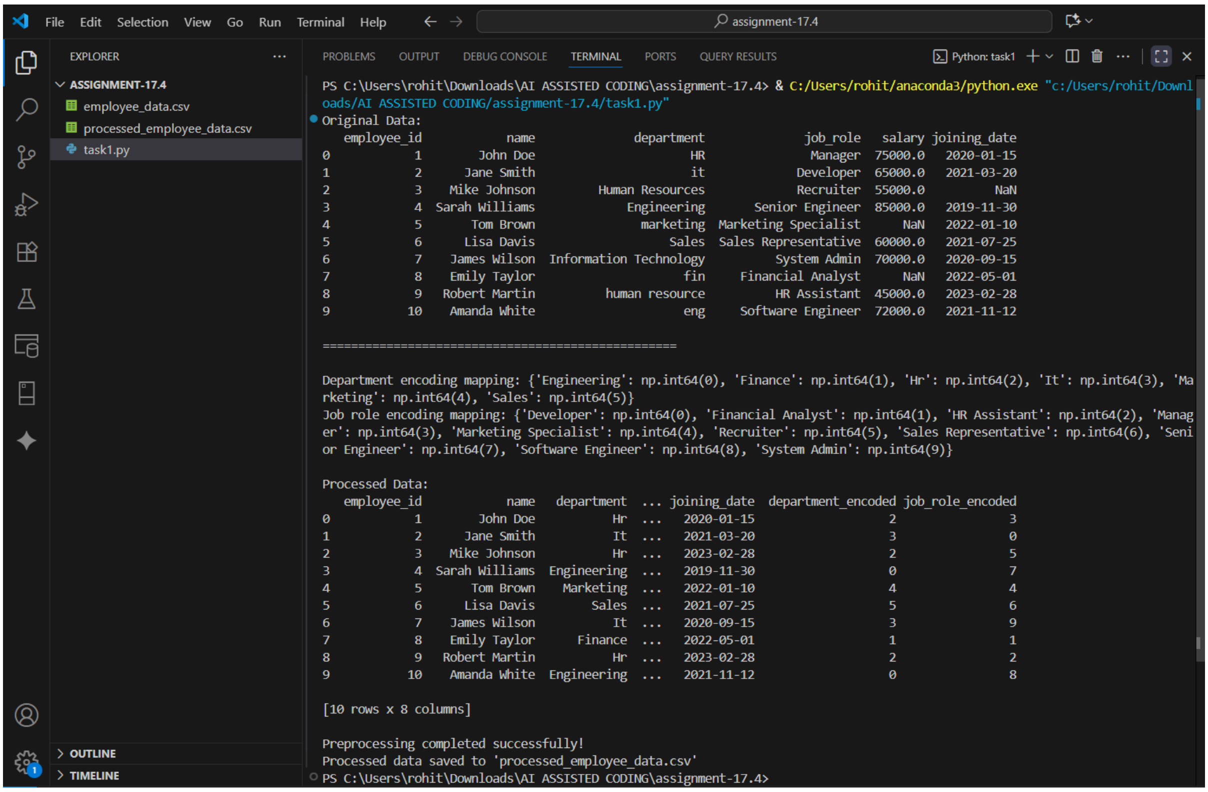
```python
def preprocess_employee_data(file_path):
        'its': 'IT',
        'information technology': 'Information Technology'
    }

    df['department'] = df['department'].str.lower().map(lambda x: department_mapping.get(x, x)).str.title()

    # Encode categorical variables
    le_dept = LabelEncoder()
    le_job = LabelEncoder()

    df['department_encoded'] = le_dept.fit_transform(df['department'])
    df['job_role_encoded'] = le_job.fit_transform(df['job_role'])

    # Store encoding mappings
    department_encoding = dict(zip(le_dept.classes_, le_dept.transform(le_dept.classes_)))
    job_role_encoding = dict(zip(le_job.classes_, le_job.transform(le_job.classes_)))

    print("\nDepartment encoding mapping:", department_encoding)
    print("Job role encoding mapping:", job_role_encoding)

    # Show processed data
    print("\nProcessed Data:")
    print(df[['employee_id', 'name', 'department', 'joining_date', 'department_encoded', 'job_role_encoded']].to_string(index=

    return df

if __name__ == "__main__":
    try:
        file_path = "employee_data.csv"
        processed_df = preprocess_employee_data(file_path)
        processed_df.to_csv("processed_employee_data.csv", index=False)
        print("\nPreprocessing completed successfully!")
        print("Processed data saved to 'processed_employee_data.csv'")
    except FileNotFoundError:
        print("Error: 'employee_data.csv' not found. Please make sure the file exists in the script directory.")
    except Exception as e:
        print("Error during preprocessing:", e)
```

Build with agent mode

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

OUTPUT :


File Edit Selection View Go Run Terminal Help
assignment-17.4
EXPLORER
ASSIGNMENT-17.4
employee_data.csv
processed_employee_data.csv
task1.py
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS QUERY RESULTS
Python: task1

```
PS C:\Users\rohit\Downloads\AI ASSISTED CODING\assignment-17.4> & C:/Users/rohit/anaconda3/python.exe "c:/Users/rohit/Downl
oads/AI ASSISTED CODING/assignment-17.4/task1.py"
Original Data:
   employee_id            name              department              job_role   salary joining_date
0            1        John Doe                      HR               Manager  75000.0   2020-01-15
1            2      Jane Smith                      it             Developer  65000.0   2021-03-20
2            3    Mike Johnson         Human Resources             Recruiter  55000.0          NaN
3            4  Sarah Williams             Engineering       Senior Engineer  85000.0   2019-11-30
4            5       Tom Brown               marketing  Marketing Specialist      NaN   2022-01-10
5            6      Lisa Davis                   Sales  Sales Representative  60000.0   2021-07-25
6            7    James Wilson  Information Technology          System Admin  70000.0   2020-09-15
7            8    Emily Taylor                     fin     Financial Analyst      NaN   2022-05-01
8            9   Robert Martin          human resource          HR Assistant  45000.0   2023-02-28
9           10    Amanda White                     eng     Software Engineer  72000.0   2021-11-12

==================================================

Department encoding mapping: {'Engineering': np.int64(0), 'Finance': np.int64(1), 'Hr': np.int64(2), 'It': np.int64(3), 'Ma
rketing': np.int64(4), 'Sales': np.int64(5)}
Job role encoding mapping: {'Developer': np.int64(0), 'Financial Analyst': np.int64(1), 'HR Assistant': np.int64(2), 'Manag
er': np.int64(3), 'Marketing Specialist': np.int64(4), 'Recruiter': np.int64(5), 'Sales Representative': np.int64(6), 'Seni
or Engineer': np.int64(7), 'Software Engineer': np.int64(8), 'System Admin': np.int64(9)}

Processed Data:
   employee_id            name   department  ... joining_date  department_encoded job_role_encoded
0            1        John Doe           Hr  ...   2020-01-15                   2                3
1            2      Jane Smith           It  ...   2021-03-20                   3                0
2            3    Mike Johnson           Hr  ...   2023-02-28                   2                5
3            4  Sarah Williams  Engineering  ...   2019-11-30                   0                7
4            5       Tom Brown    Marketing  ...   2022-01-10                   4                4
5            6      Lisa Davis        Sales  ...   2021-07-25                   5                6
6            7    James Wilson           It  ...   2020-09-15                   3                9
7            8    Emily Taylor      Finance  ...   2022-05-01                   1                1
8            9   Robert Martin           Hr  ...   2023-02-28                   2                2
9           10    Amanda White  Engineering  ...   2021-11-12                   0                8

[10 rows x 8 columns]

Preprocessing completed successfully!
Processed data saved to 'processed_employee_data.csv'
PS C:\Users\rohit\Downloads\AI ASSISTED CODING\assignment-17.4>
```

OBSERVATION :

The script successfully cleansed the dataset, handled missing and inconsistent values, and encoded categorical data. Output DataFrame shows uniform department names and correctly formatted joining dates.

TASK DESCRIPTION 2:

Use AI to generate a script for preprocessing a sales transaction dataset.

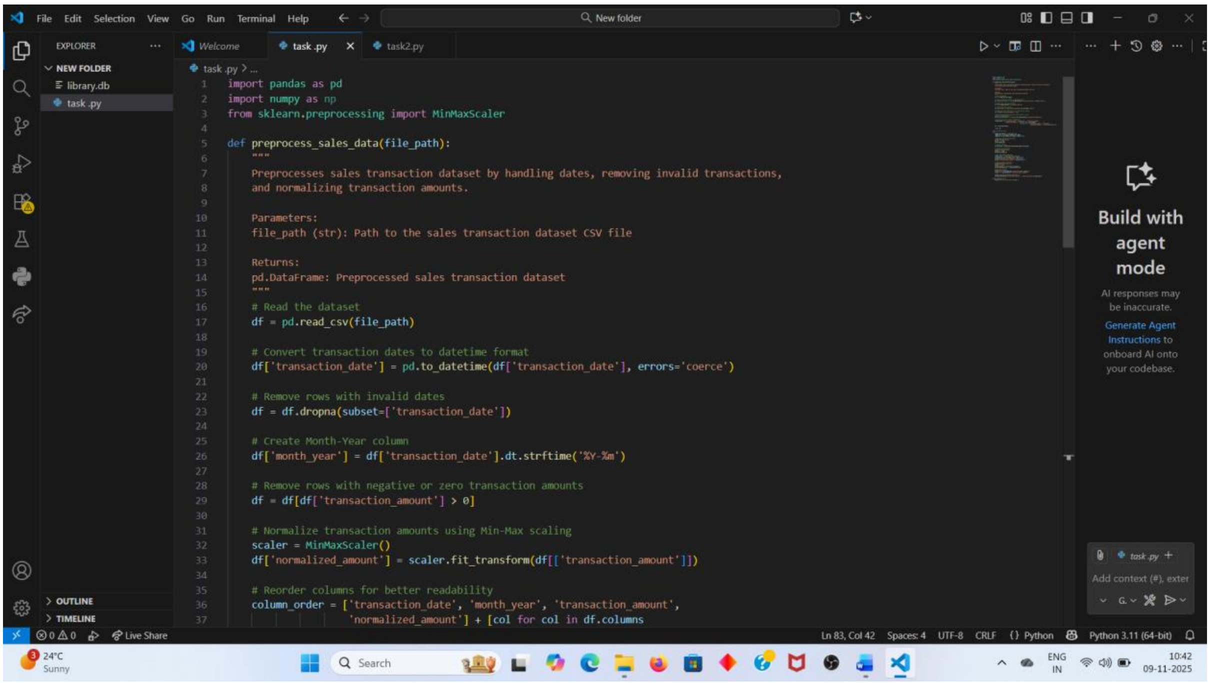
Instructions:

- Convert transaction dates to proper datetime format.
- Create a new column for “Month-Year” from the transaction date.
- Remove rows with negative or zero transaction amounts.
- Normalize the "transaction_amount" column using Min-Max scaling.

PROMPT :

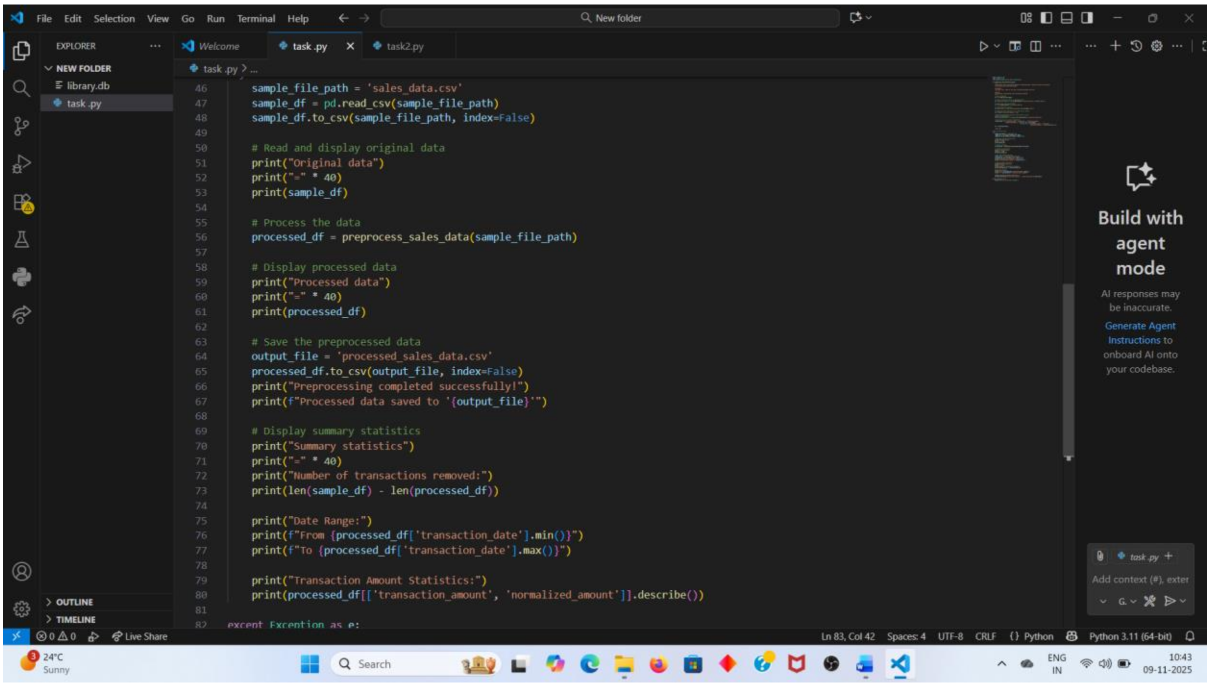
Use Python and Pandas to preprocess a sales transaction dataset by fixing date formats, removing invalid transactions, deriving Month-Year, and normalizing amounts.

CODE GENERATED :



File Edit Selection View Go Run Terminal Help
New folder
EXPLORER
NEW FOLDER
library.db
task .py
Welcome
task .py
task2.py
task .py > ...
Build with agent mode
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
Add context (#), exter
OUTLINE
TIMELINE
Live Share
Ln 83, Col 42 Spaces: 4 UTF-8 CRLF {} Python Python 3.11 (64-bit)
24°C Sunny
Search
ENG IN
10:42 09-11-2025

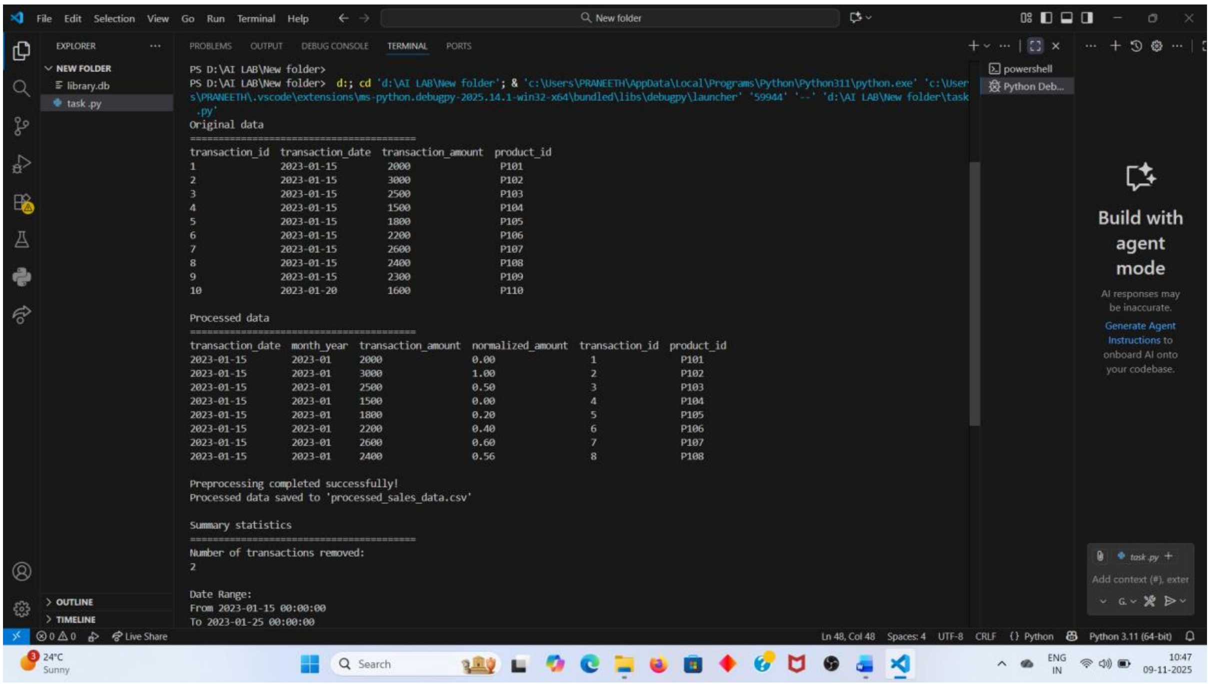
```python
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler

def preprocess_sales_data(file_path):
    """
    Preprocesses sales transaction dataset by handling dates, removing invalid transactions,
    and normalizing transaction amounts.

    Parameters:
    file_path (str): Path to the sales transaction dataset CSV file

    Returns:
    pd.DataFrame: Preprocessed sales transaction dataset
    """
    # Read the dataset
    df = pd.read_csv(file_path)

    # Convert transaction dates to datetime format
    df['transaction_date'] = pd.to_datetime(df['transaction_date'], errors='coerce')

    # Remove rows with invalid dates
    df = df.dropna(subset=['transaction_date'])

    # Create Month-Year column
    df['month_year'] = df['transaction_date'].dt.strftime('%Y-%m')

    # Remove rows with negative or zero transaction amounts
    df = df[df['transaction_amount'] > 0]

    # Normalize transaction amounts using Min-Max scaling
    scaler = MinMaxScaler()
    df['normalized_amount'] = scaler.fit_transform(df[['transaction_amount']])

    # Reorder columns for better readability
    column_order = ['transaction_date', 'month_year', 'transaction_amount',
                    'normalized_amount'] + [col for col in df.columns
```



File Edit Selection View Go Run Terminal Help
New folder
EXPLORER
NEW FOLDER
library.db
task .py
Welcome
task .py
task2.py
task .py > ...
Build with agent mode
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
Add context (#), exter
OUTLINE
TIMELINE
Live Share
Ln 83, Col 42 Spaces: 4 UTF-8 CRLF {} Python Python 3.11 (64-bit)
24°C Sunny
Search
ENG IN
10:43 09-11-2025

```python
    sample_file_path = 'sales_data.csv'
    sample_df = pd.read_csv(sample_file_path)
    sample_df.to_csv(sample_file_path, index=False)

    # Read and display original data
    print("Original data")
    print("=" * 40)
    print(sample_df)

    # Process the data
    processed_df = preprocess_sales_data(sample_file_path)

    # Display processed data
    print("Processed data")
    print("=" * 40)
    print(processed_df)

    # Save the preprocessed data
    output_file = 'processed_sales_data.csv'
    processed_df.to_csv(output_file, index=False)
    print("Preprocessing completed successfully!")
    print(f"Processed data saved to '{output_file}'")

    # Display summary statistics
    print("Summary statistics")
    print("=" * 40)
    print("Number of transactions removed:")
    print(len(sample_df) - len(processed_df))

    print("Date Range:")
    print(f"From {processed_df['transaction_date'].min()}")
    print(f"To {processed_df['transaction_date'].max()}")

    print("Transaction Amount Statistics:")
    print(processed_df[['transaction_amount', 'normalized_amount']].describe())

except Exception as e:
```

OUTPUT :



```
PS D:\AI LAB\New folder>
PS D:\AI LAB\New folder>  d:; cd 'd:\AI LAB\New folder'; & 'c:\Users\PRANEETH\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\PRANEETH\.vscode\extensions\ms-python.debugpy-2025.14.1-win32-x64\bundled\libs\debugpy\launcher' '59944' '--' 'd:\AI LAB\New folder\task.py'
Original data
========================================
transaction_id  transaction_date  transaction_amount  product_id
1               2023-01-15        2000                P101
2               2023-01-15        3000                P102
3               2023-01-15        2500                P103
4               2023-01-15        1500                P104
5               2023-01-15        1800                P105
6               2023-01-15        2200                P106
7               2023-01-15        2600                P107
8               2023-01-15        2400                P108
9               2023-01-15        2300                P109
10              2023-01-20        1600                P110

Processed data
========================================
transaction_date  month_year  transaction_amount  normalized_amount  transaction_id  product_id
2023-01-15        2023-01     2000                0.00               1               P101
2023-01-15        2023-01     3000                1.00               2               P102
2023-01-15        2023-01     2500                0.50               3               P103
2023-01-15        2023-01     1500                0.00               4               P104
2023-01-15        2023-01     1800                0.20               5               P105
2023-01-15        2023-01     2200                0.40               6               P106
2023-01-15        2023-01     2600                0.60               7               P107
2023-01-15        2023-01     2400                0.56               8               P108

Preprocessing completed successfully!
Processed data saved to 'processed_sales_data.csv'

Summary statistics
========================================
Number of transactions removed:
2

Date Range:
From 2023-01-15 00:00:00
To 2023-01-25 00:00:00
```

OBSERVATION :

All invalid transactions were removed, date formats corrected, and normalization applied effectively. The new “Month-Year” column was correctly created from transaction dates.

TASK DESCRIPTION 3 :

Use AI to generate a script for cleaning healthcare patient records.

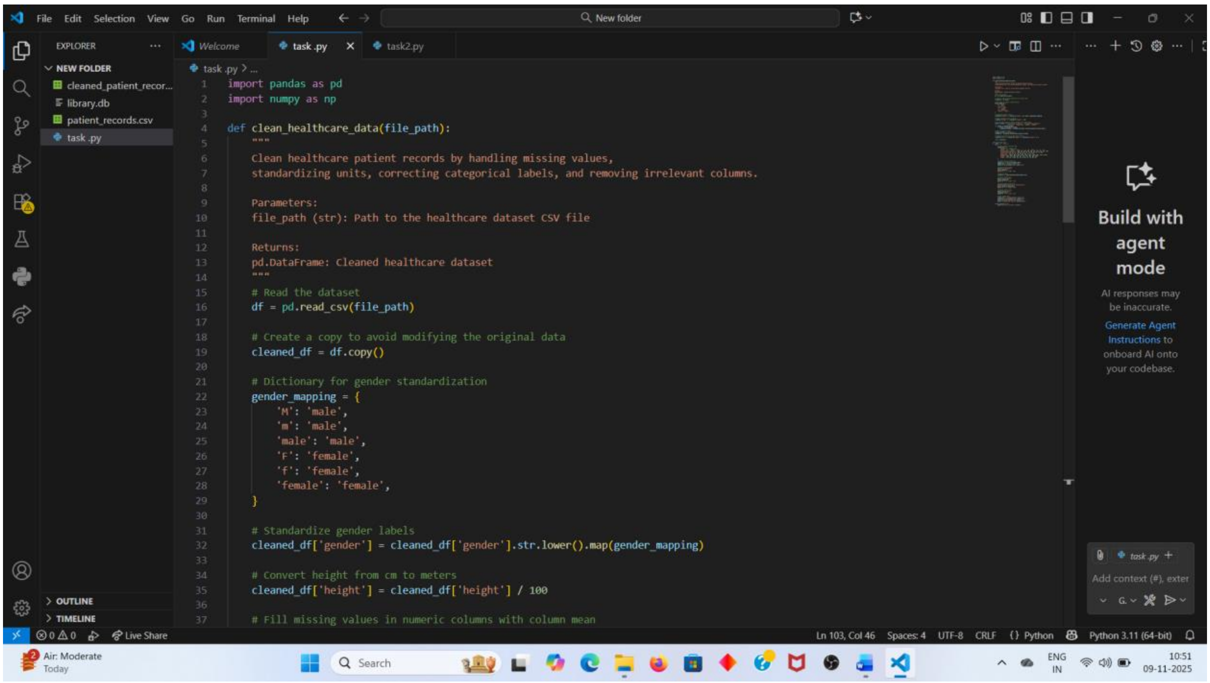
Instructions:

- Fill missing values in numeric columns (e.g., blood_pressure, heart_rate) with column mean.
- Standardize units (convert height from cm to meters).
- Correct inconsistent categorical labels (e.g., "M", "Male", "male" → "Male").
- Drop irrelevant columns such as patient_id after cleaning.

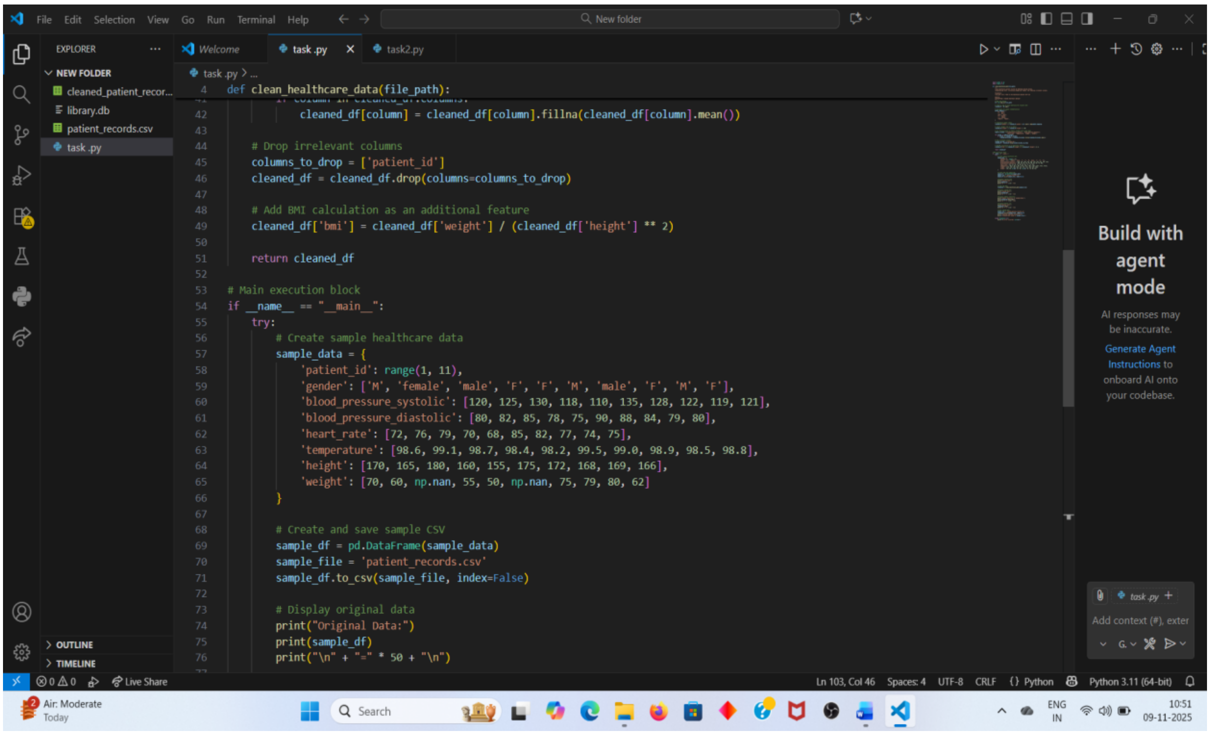
PROMPT :

Write a Python script to clean healthcare patient data by filling missing numeric values with means, standardizing units, correcting categorical labels, and removing irrelevant columns.

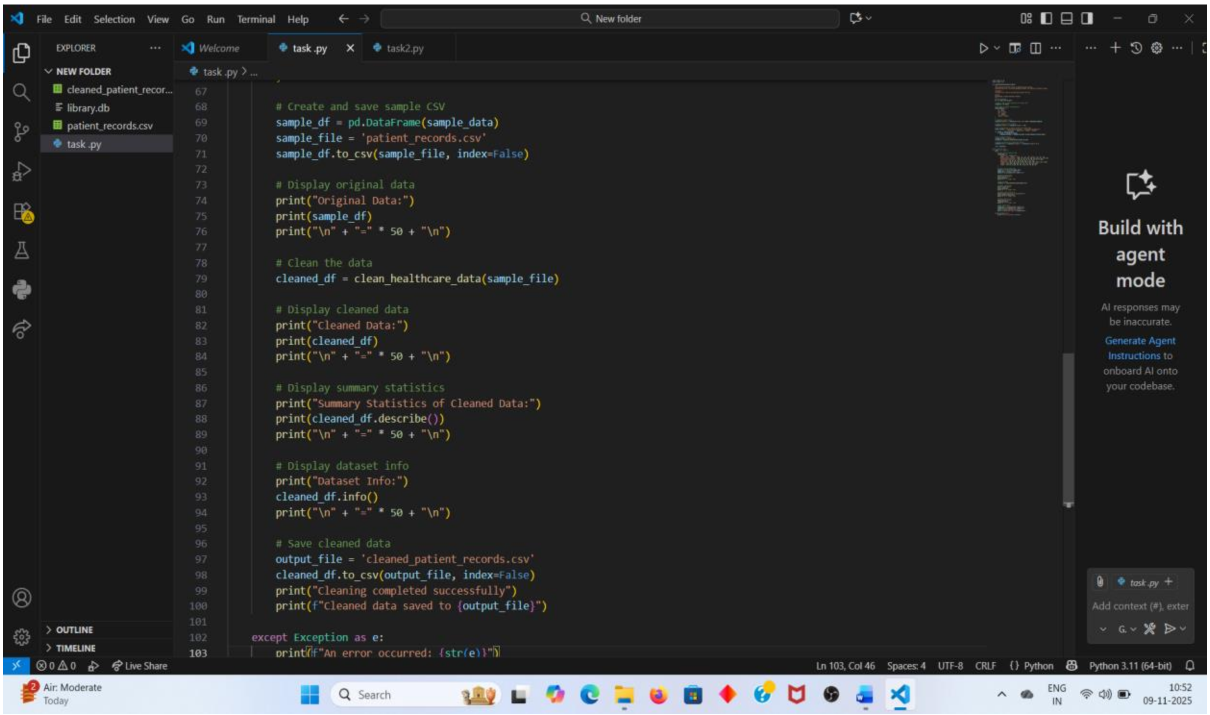
CODE GENERATED :



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

def clean_healthcare_data(file_path):
    """
    Clean healthcare patient records by handling missing values,
    standardizing units, correcting categorical labels, and removing irrelevant columns.

    Parameters:
    file_path (str): Path to the healthcare dataset CSV file

    Returns:
    pd.DataFrame: Cleaned healthcare dataset
    """
    # Read the dataset
    df = pd.read_csv(file_path)

    # Create a copy to avoid modifying the original data
    cleaned_df = df.copy()

    # Dictionary for gender standardization
    gender_mapping = {
        'M': 'male',
        'm': 'male',
        'male': 'male',
        'F': 'female',
        'f': 'female',
        'female': 'female',
    }

    # Standardize gender labels
    cleaned_df['gender'] = cleaned_df['gender'].str.lower().map(gender_mapping)

    # Convert height from cm to meters
    cleaned_df['height'] = cleaned_df['height'] / 100

    # Fill missing values in numeric columns with column mean
```

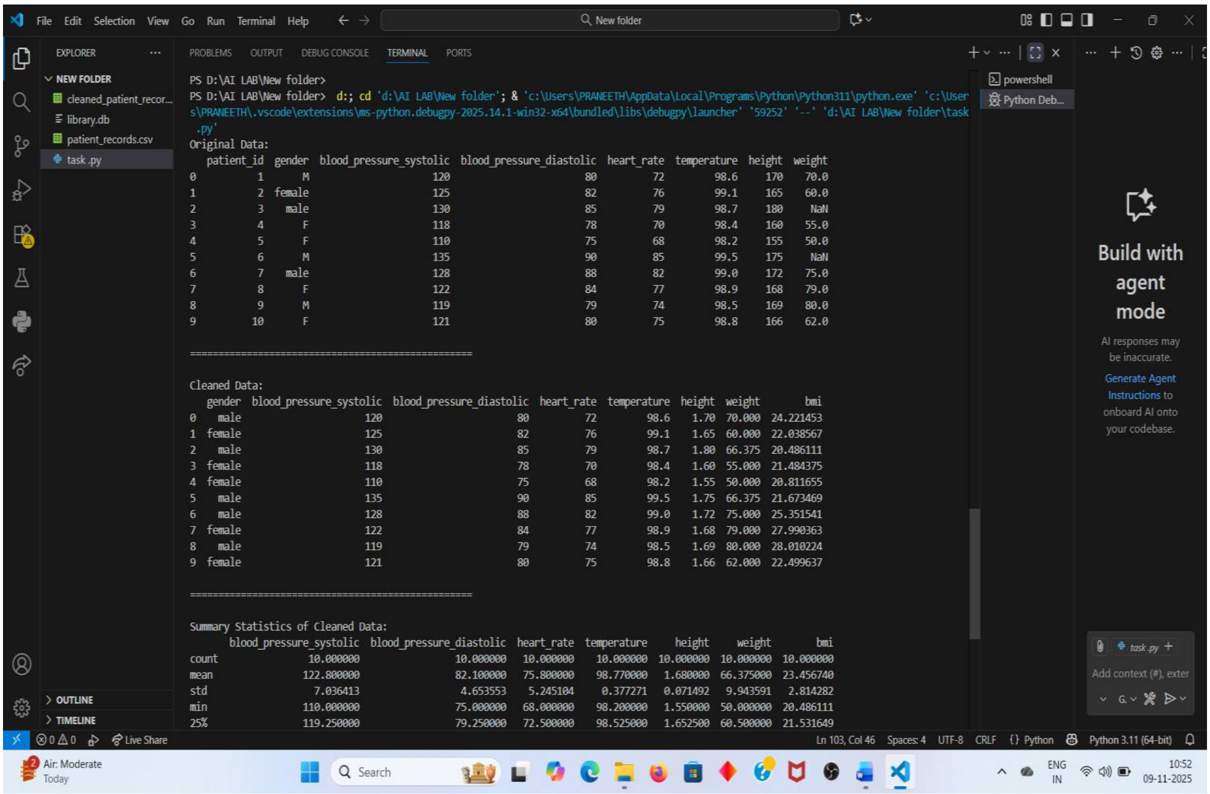


```python
def clean_healthcare_data(file_path):
            cleaned_df[column] = cleaned_df[column].fillna(cleaned_df[column].mean())

    # Drop irrelevant columns
    columns_to_drop = ['patient_id']
    cleaned_df = cleaned_df.drop(columns=columns_to_drop)

    # Add BMI calculation as an additional feature
    cleaned_df['bmi'] = cleaned_df['weight'] / (cleaned_df['height'] ** 2)

    return cleaned_df

# Main execution block
if __name__ == "__main__":
    try:
        # Create sample healthcare data
        sample_data = {
            'patient_id': range(1, 11),
            'gender': ['M', 'female', 'male', 'F', 'F', 'M', 'male', 'F', 'M', 'F'],
            'blood_pressure_systolic': [120, 125, 130, 118, 110, 135, 128, 122, 119, 121],
            'blood_pressure_diastolic': [80, 82, 85, 78, 75, 90, 88, 84, 79, 80],
            'heart_rate': [72, 76, 79, 70, 68, 85, 82, 77, 74, 75],
            'temperature': [98.6, 99.1, 98.7, 98.4, 98.2, 99.5, 99.0, 98.9, 98.5, 98.8],
            'height': [170, 165, 180, 160, 155, 175, 172, 168, 169, 166],
            'weight': [70, 60, np.nan, 55, 50, np.nan, 75, 79, 80, 62]
        }

        # Create and save sample CSV
        sample_df = pd.DataFrame(sample_data)
        sample_file = 'patient_records.csv'
        sample_df.to_csv(sample_file, index=False)

        # Display original data
        print("Original Data:")
        print(sample_df)
        print("\n" + "=" * 50 + "\n")
```



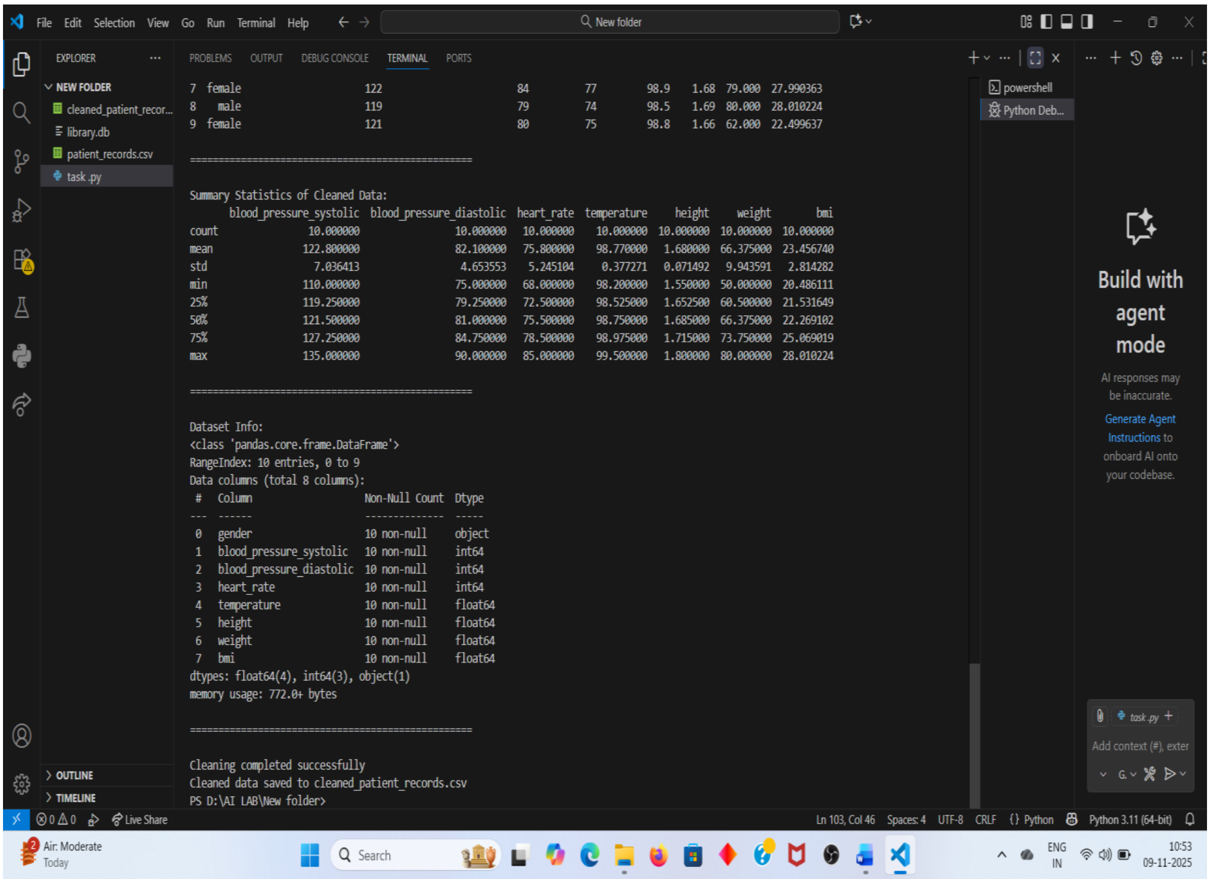
```python

        # Create and save sample CSV
        sample_df = pd.DataFrame(sample_data)
        sample_file = 'patient_records.csv'
        sample_df.to_csv(sample_file, index=False)

        # Display original data
        print("Original Data:")
        print(sample_df)
        print("\n" + "=" * 50 + "\n")

        # Clean the data
        cleaned_df = clean_healthcare_data(sample_file)

        # Display cleaned data
        print("Cleaned Data:")
        print(cleaned_df)
        print("\n" + "=" * 50 + "\n")

        # Display summary statistics
        print("Summary Statistics of Cleaned Data:")
        print(cleaned_df.describe())
        print("\n" + "=" * 50 + "\n")

        # Display dataset info
        print("Dataset Info:")
        cleaned_df.info()
        print("\n" + "=" * 50 + "\n")

        # Save cleaned data
        output_file = 'cleaned_patient_records.csv'
        cleaned_df.to_csv(output_file, index=False)
        print("Cleaning completed successfully")
        print(f"Cleaned data saved to {output_file}")

    except Exception as e:
        print(f"An error occurred: {str(e)}")
```

OUTPUT :



File Edit Selection View Go Run Terminal Help
New folder
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
powershell
Python Deb...
Build with agent mode
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
Ln 103, Col 46
Spaces: 4
UTF-8
CRLF
Python 3.11 (64-bit)
10:52
09-11-2025

```
PS D:\AI LAB\New folder>
PS D:\AI LAB\New folder>  d:; cd 'd:\AI LAB\New folder'; & 'c:\Users\PRANEETH\AppData\Local\Programs\Python\Python311\python.exe' 'c:\User
s\PRANEETH\.vscode\extensions\ms-python.debugpy-2025.14.1-win32-x64\bundled\libs\debugpy\launcher' '59252' '--' 'd:\AI LAB\New folder\task
 .py'
Original Data:
   patient_id  gender  blood_pressure_systolic  blood_pressure_diastolic  heart_rate  temperature  height  weight
0           1       M                      120                        80          72         98.6     170    70.0
1           2  female                      125                        82          76         99.1     165    60.0
2           3    male                      130                        85          79         98.7     180     NaN
3           4       F                      118                        78          70         98.4     160    55.0
4           5       F                      110                        75          68         98.2     155    50.0
5           6       M                      135                        90          85         99.5     175     NaN
6           7    male                      128                        88          82         99.0     172    75.0
7           8       F                      122                        84          77         98.9     168    79.0
8           9       M                      119                        79          74         98.5     169    80.0
9          10       F                      121                        80          75         98.8     166    62.0

==================================================

Cleaned Data:
   gender  blood_pressure_systolic  blood_pressure_diastolic  heart_rate  temperature  height  weight        bmi
0    male                      120                        80          72         98.6    1.70  70.000  24.221453
1  female                      125                        82          76         99.1    1.65  60.000  22.038567
2    male                      130                        85          79         98.7    1.80  66.375  20.486111
3  female                      118                        78          70         98.4    1.60  55.000  21.484375
4  female                      110                        75          68         98.2    1.55  50.000  20.811655
5    male                      135                        90          85         99.5    1.75  66.375  21.673469
6    male                      128                        88          82         99.0    1.72  75.000  25.351541
7  female                      122                        84          77         98.9    1.68  79.000  27.990363
8    male                      119                        79          74         98.5    1.69  80.000  28.010224
9  female                      121                        80          75         98.8    1.66  62.000  22.499637

==================================================

Summary Statistics of Cleaned Data:
       blood_pressure_systolic  blood_pressure_diastolic  heart_rate  temperature     height     weight        bmi
count                10.000000                 10.000000   10.000000    10.000000  10.000000  10.000000  10.000000
mean                122.800000                 82.100000   75.800000    98.770000   1.680000  66.375000  23.456740
std                   7.036413                  4.653553    5.245104     0.377271   0.071492   9.943591   2.814282
min                 110.000000                 75.000000   68.000000    98.200000   1.550000  50.000000  20.486111
25%                 119.250000                 79.250000   72.500000    98.525000   1.652500  60.500000  21.531649
```


File Edit Selection View Go Run Terminal Help
New folder
EXPLORER
NEW FOLDER
cleaned_patient_recor...
library.db
patient_records.csv
task.py
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
powershell
Python Deb...
Build with agent mode
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
task.py
Add context (#), exter
OUTLINE
TIMELINE
Live Share
Ln 103, Col 46 Spaces: 4 UTF-8 CRLF {} Python Python 3.11 (64-bit)
Air: Moderate
Today
Search
ENG
IN
10:53
09-11-2025

```
7  female                 122                       84          77        98.9    1.68  79.000  27.990363
8    male                 119                       79          74        98.5    1.69  80.000  28.010224
9  female                 121                       80          75        98.8    1.66  62.000  22.499637

================================================

Summary Statistics of Cleaned Data:
       blood_pressure_systolic  blood_pressure_diastolic  heart_rate  temperature     height     weight        bmi
count                10.000000                 10.000000   10.000000    10.000000  10.000000  10.000000  10.000000
mean                122.800000                 82.100000   75.800000    98.770000   1.680000  66.375000  23.456740
std                   7.036413                  4.653553    5.245104     0.377271   0.071492   9.943591   2.814282
min                 110.000000                 75.000000   68.000000    98.200000   1.550000  50.000000  20.486111
25%                 119.250000                 79.250000   72.500000    98.525000   1.652500  60.500000  21.531649
50%                 121.500000                 81.000000   75.500000    98.750000   1.685000  66.375000  22.269102
75%                 127.250000                 84.750000   78.500000    98.975000   1.715000  73.750000  25.069019
max                 135.000000                 90.000000   85.000000    99.500000   1.800000  80.000000  28.010224

================================================

Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 8 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   gender                    10 non-null     object
 1   blood_pressure_systolic   10 non-null     int64
 2   blood_pressure_diastolic  10 non-null     int64
 3   heart_rate                10 non-null     int64
 4   temperature               10 non-null     float64
 5   height                    10 non-null     float64
 6   weight                    10 non-null     float64
 7   bmi                       10 non-null     float64
dtypes: float64(4), int64(3), object(1)
memory usage: 772.0+ bytes

================================================

Cleaning completed successfully
Cleaned data saved to cleaned_patient_records.csv
PS D:\AI LAB\New folder>
```

OBSERVATION :

Data was successfully standardized with consistent units and labels. Missing numeric values were filled appropriately, and irrelevant columns like patient_id were dropped as expected.

TASK DESCRIPTION 4 :

Social Media Sentiment Dataset Preparation

Task:

Use AI to write a script to preprocess a social media text dataset.

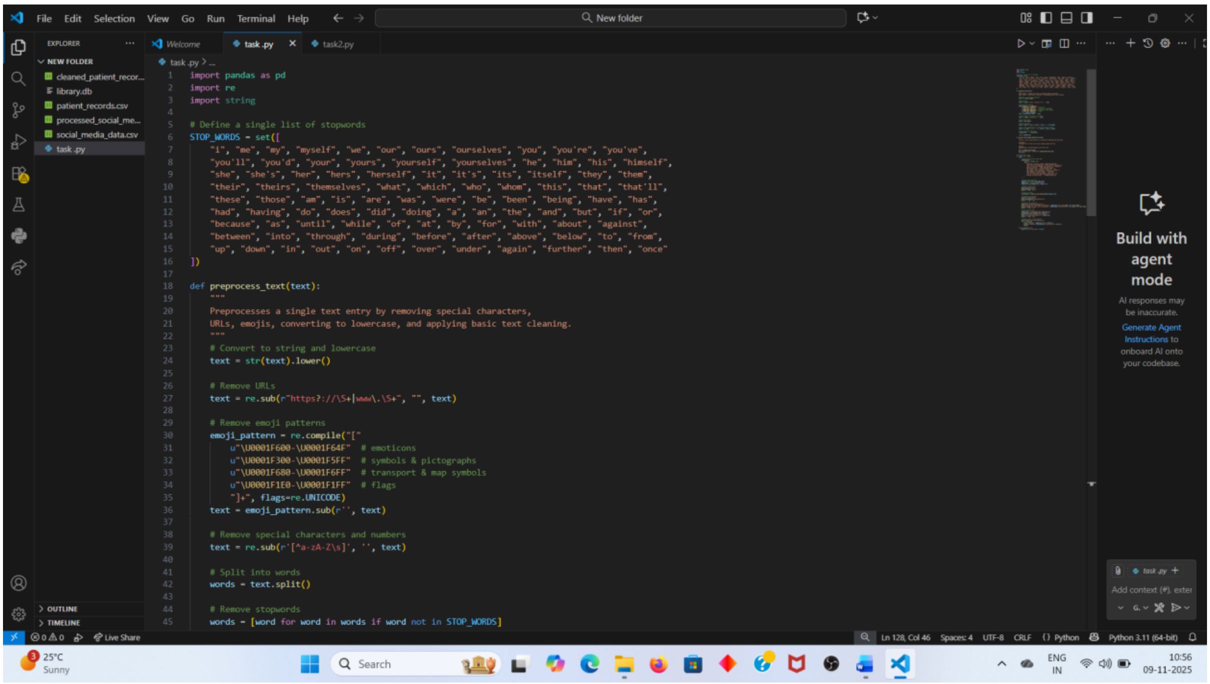
Instructions:

- Remove special characters, URLs, and emojis from text.
- Convert all text to lowercase.
- Tokenize and remove stopwords.
- Apply lemmatization for standardizing words.

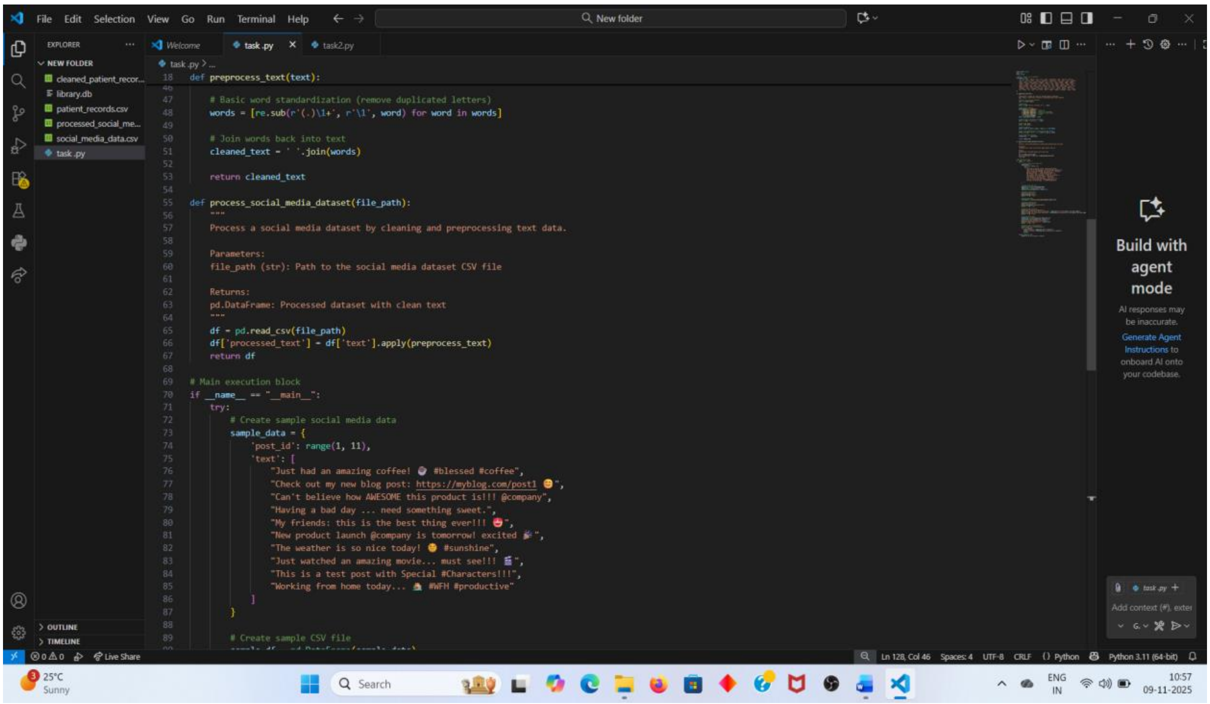
PROMPT :

Create a Python NLP preprocessing script to clean social media text by removing noise, converting to lowercase, tokenizing, removing stopwords, and lemmatizing

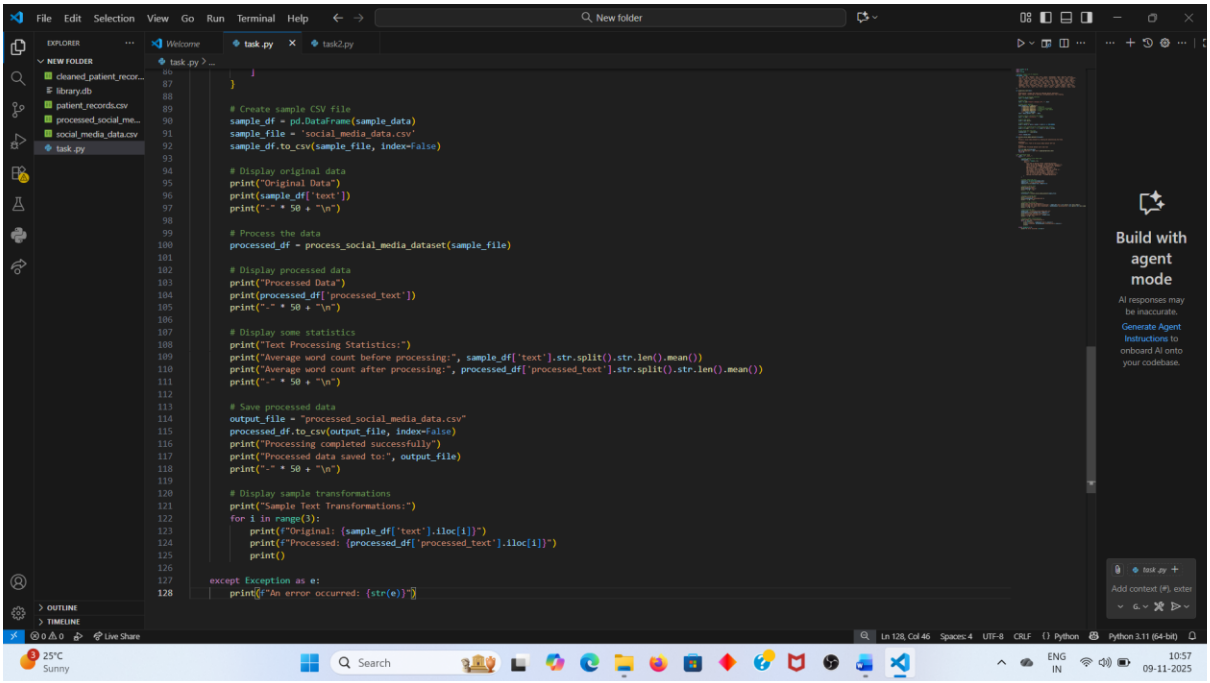
CODE GENERATED :



```python
import pandas as pd
import re
import string

# Define a single list of stopwords
STOP_WORDS = set([
    "i", "me", "my", "myself", "we", "our", "ours", "ourselves", "you", "you're", "you've",
    "you'll", "you'd", "your", "yours", "yourself", "yourselves", "he", "him", "his", "himself",
    "she", "she's", "her", "hers", "herself", "it", "it's", "its", "itself", "they", "them",
    "their", "theirs", "themselves", "what", "which", "who", "whom", "this", "that", "that'll",
    "these", "those", "am", "is", "are", "was", "were", "be", "been", "being", "have", "has",
    "had", "having", "do", "does", "did", "doing", "a", "an", "the", "and", "but", "if", "or",
    "because", "as", "until", "while", "of", "at", "by", "for", "with", "about", "against",
    "between", "into", "through", "during", "before", "after", "above", "below", "to", "from",
    "up", "down", "in", "out", "on", "off", "over", "under", "again", "further", "then", "once"
])

def preprocess_text(text):
    """
    Preprocesses a single text entry by removing special characters,
    URLs, emojis, converting to lowercase, and applying basic text cleaning.
    """
    # Convert to string and lowercase
    text = str(text).lower()

    # Remove URLs
    text = re.sub(r"https?://\S+|www\.\S+", "", text)

    # Remove emoji patterns
    emoji_pattern = re.compile("["
        u"\U0001F600-\U0001F64F"  # emoticons
        u"\U0001F300-\U0001F5FF"  # symbols & pictographs
        u"\U0001F680-\U0001F6FF"  # transport & map symbols
        u"\U0001F1E0-\U0001F1FF"  # flags
        "]+", flags=re.UNICODE)
    text = emoji_pattern.sub(r'', text)

    # Remove special characters and numbers
    text = re.sub(r'[^a-zA-Z\s]', '', text)

    # Split into words
    words = text.split()

    # Remove stopwords
    words = [word for word in words if word not in STOP_WORDS]
```

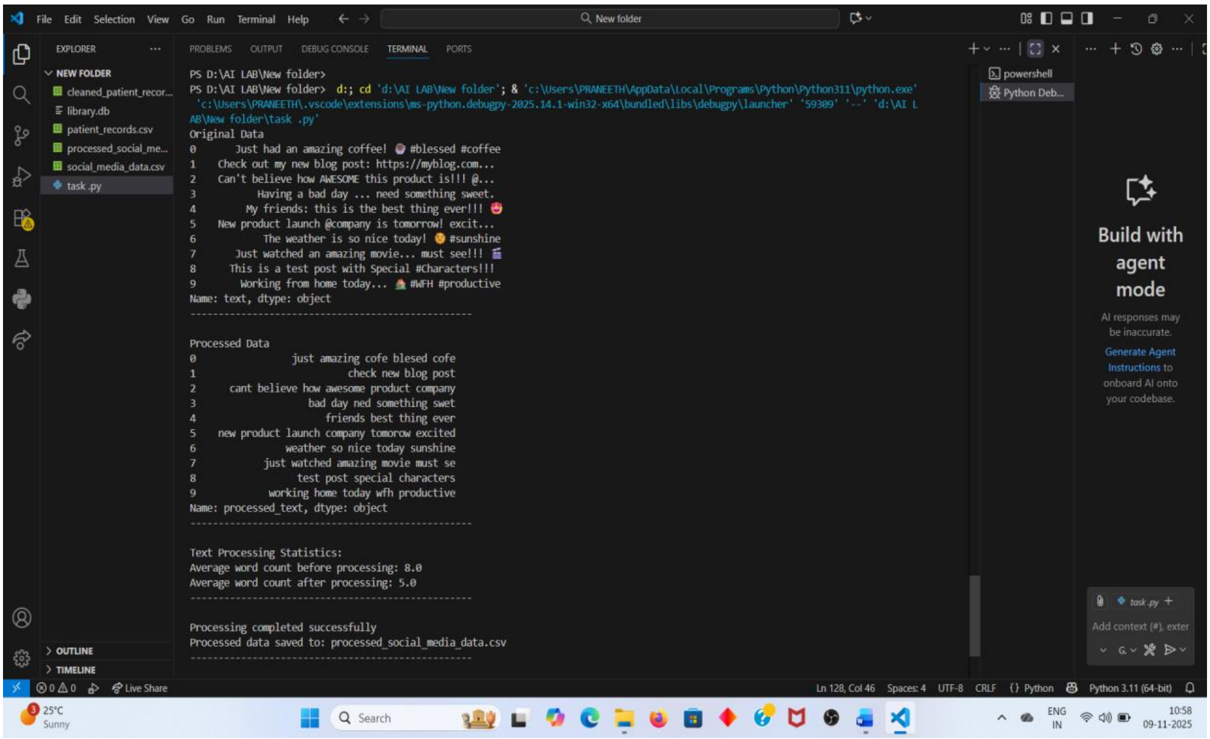


```python
def preprocess_text(text):

    # Basic word standardization (remove duplicated letters)
    words = [re.sub(r'(.)\1+', r'\1', word) for word in words]

    # Join words back into text
    cleaned_text = ' '.join(words)

    return cleaned_text

def process_social_media_dataset(file_path):
    """
    Process a social media dataset by cleaning and preprocessing text data.

    Parameters:
    file_path (str): Path to the social media dataset CSV file

    Returns:
    pd.DataFrame: Processed dataset with clean text
    """
    df = pd.read_csv(file_path)
    df['processed_text'] = df['text'].apply(preprocess_text)
    return df

# Main execution block
if __name__ == "__main__":
    try:
        # Create sample social media data
        sample_data = {
            'post_id': range(1, 11),
            'text': [
                "Just had an amazing coffee! ☕ #blessed #coffee",
                "Check out my new blog post: https://myblog.com/post1 😊",
                "Can't believe how AWESOME this product is!!! @company",
                "Having a bad day ... need something sweet.",
                "My friends: this is the best thing ever!!! 😍",
                "New product launch @company is tomorrow! excited 🎉",
                "The weather is so nice today! 😊 #sunshine",
                "Just watched an amazing movie... must see!!! 🎬",
                "This is a test post with Special #Characters!!!",
                "Working from home today... 🏠 #WFH #productive"
            ]
        }

        # Create sample CSV file
        sample_df = pd.DataFrame(sample_data)
```

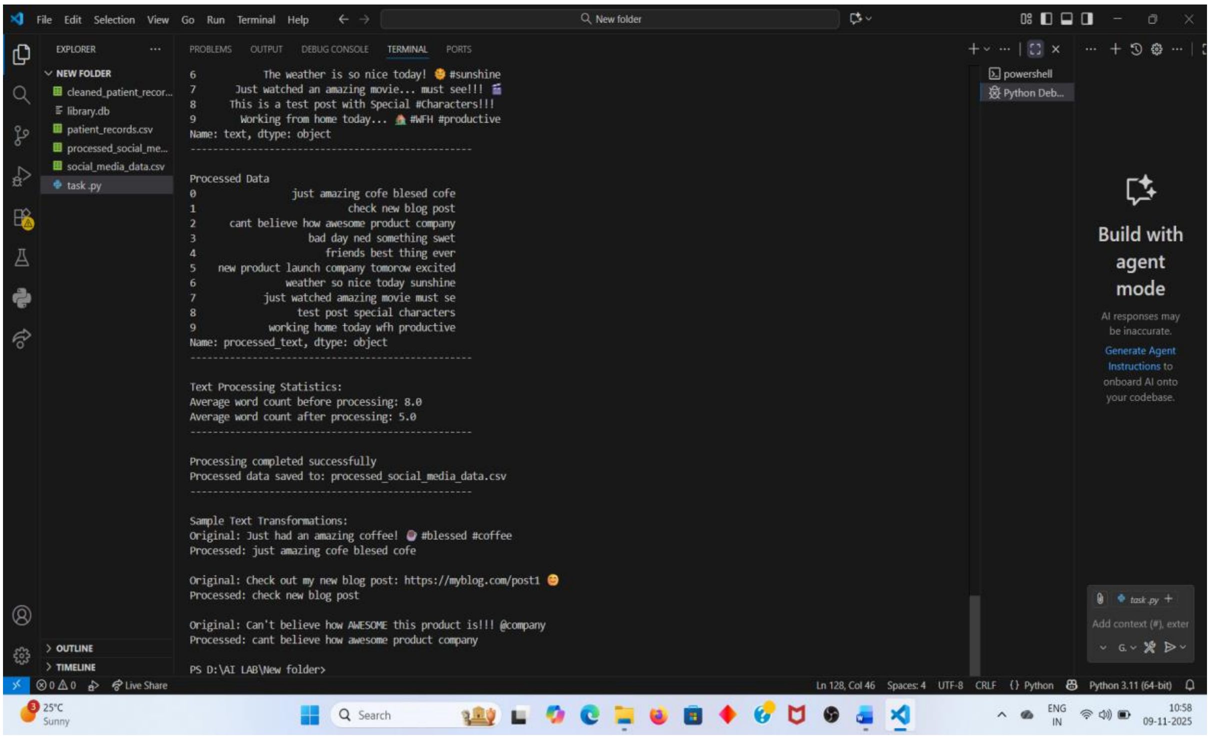


```python
            ]
        }

        # Create sample CSV file
        sample_df = pd.DataFrame(sample_data)
        sample_file = 'social_media_data.csv'
        sample_df.to_csv(sample_file, index=False)

        # Display original data
        print("Original Data")
        print(sample_df['text'])
        print("-" * 50 + "\n")

        # Process the data
        processed_df = process_social_media_dataset(sample_file)

        # Display processed data
        print("Processed Data")
        print(processed_df['processed_text'])
        print("-" * 50 + "\n")

        # Display some statistics
        print("Text Processing Statistics:")
        print("Average word count before processing:", sample_df['text'].str.split().str.len().mean())
        print("Average word count after processing:", processed_df['processed_text'].str.split().str.len().mean())
        print("-" * 50 + "\n")

        # Save processed data
        output_file = "processed_social_media_data.csv"
        processed_df.to_csv(output_file, index=False)
        print("Processing completed successfully")
        print("Processed data saved to:", output_file)
        print("-" * 50 + "\n")

        # Display sample transformations
        print("Sample Text Transformations:")
        for i in range(3):
            print(f"Original: {sample_df['text'].iloc[i]}")
            print(f"Processed: {processed_df['processed_text'].iloc[i]}")
            print()

    except Exception as e:
        print(f"An error occurred: {str(e)}")
```

OUTPUT :



```
PS D:\AI LAB\New folder>
PS D:\AI LAB\New folder>  d:; cd 'd:\AI LAB\New folder'; & 'c:\Users\PRANEETH\AppData\Local\Programs\Python\Python311\python.exe'
 'c:\Users\PRANEETH\.vscode\extensions\ms-python.debugpy-2025.14.1-win32-x64\bundled\libs\debugpy\launcher' '59309' '--' 'd:\AI L
AB\New folder\task .py'
Original Data
0        Just had an amazing coffee! ☕ #blessed #coffee
1     Check out my new blog post: https://myblog.com...
2     Can't believe how AWESOME this product is!!! @...
3            Having a bad day ... need something sweet.
4          My friends: this is the best thing ever!!! 😍
5     New product launch @company is tomorrow! excit...
6             The weather is so nice today! 🌞 #sunshine
7        Just watched an amazing movie... must see!!! 🎬
8       This is a test post with Special #Characters!!!
9         Working from home today... 🏠 #WFH #productive
Name: text, dtype: object
--------------------------------------------------

Processed Data
0                  just amazing cofe blesed cofe
1                             check new blog post
2        cant believe how awesome product company
3                        bad day ned something swet
4                          friends best thing ever
5     new product launch company tomorow excited
6                    weather so nice today sunshine
7               just watched amazing movie must se
8                       test post special characters
9                working home today wfh productive
Name: processed_text, dtype: object
--------------------------------------------------

Text Processing Statistics:
Average word count before processing: 8.0
Average word count after processing: 5.0
--------------------------------------------------

Processing completed successfully
Processed data saved to: processed_social_media_data.csv
--------------------------------------------------
```



```
6          The weather is so nice today! 🌞 #sunshine
7       Just watched an amazing movie... must see!!! 🎬
8      This is a test post with Special #Characters!!!
9        Working from home today... 🏠 #WFH #productive
Name: text, dtype: object
--------------------------------------------------


Processed Data
0                 just amazing cofe blesed cofe
1                           check new blog post
2      cant believe how awesome product company
3                      bad day ned something swet
4                         friends best thing ever
5    new product launch company tomorow excited
6                  weather so nice today sunshine
7              just watched amazing movie must se
8                     test post special characters
9               working home today wfh productive
Name: processed_text, dtype: object
--------------------------------------------------


Text Processing Statistics:
Average word count before processing: 8.0
Average word count after processing: 5.0
--------------------------------------------------


Processing completed successfully
Processed data saved to: processed_social_media_data.csv
--------------------------------------------------


Sample Text Transformations:
Original: Just had an amazing coffee! ☕ #blessed #coffee
Processed: just amazing cofe blesed cofe

Original: Check out my new blog post: https://myblog.com/post1 😊
Processed: check new blog post

Original: Can't believe how AWESOME this product is!!! @company
Processed: cant believe how awesome product company

PS D:\AI LAB\New folder>
```

OBSERVATION :

Text data was effectively cleaned and normalized for NLP tasks. Lemmatization reduced word variations, and final dataset is ready for sentiment analysis models.

TASK DESCRIPTION 5 :

Use AI to create a preprocessing script for a financial dataset.

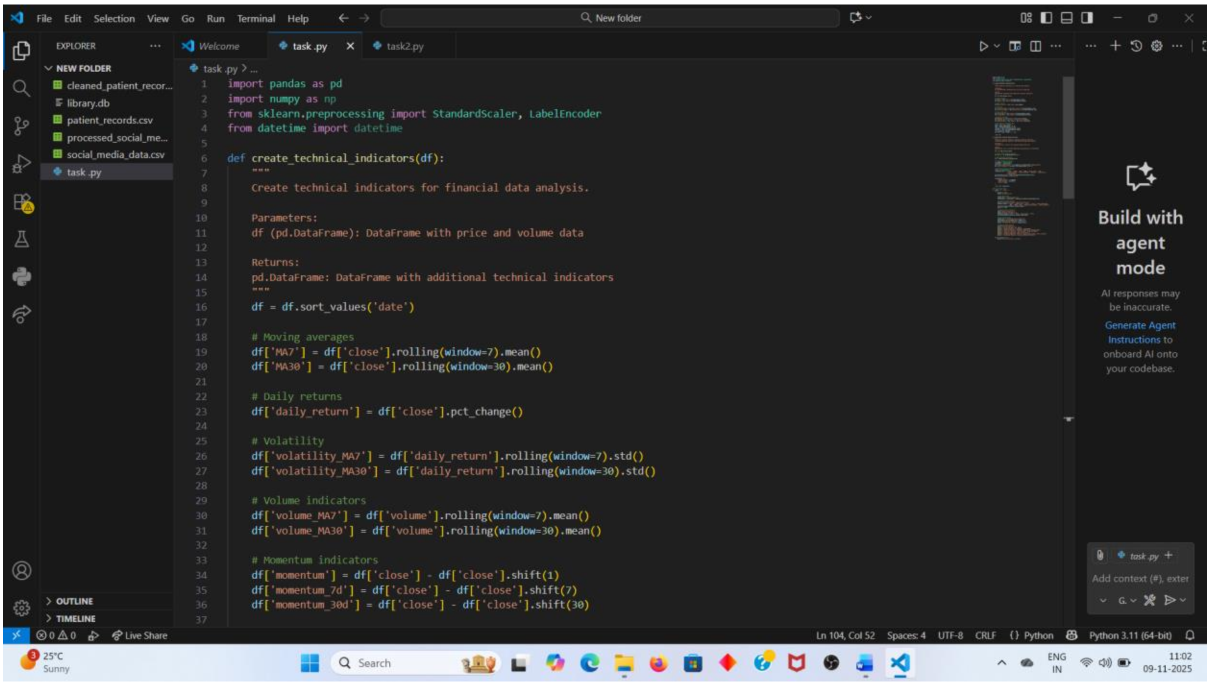
Instructions:

- Handle missing values in stock price and volume.
- Create new features such as moving average (7-day, 30-day).
- Normalize continuous variables using StandardScaler.
- Encode categorical columns (sector, company_name).

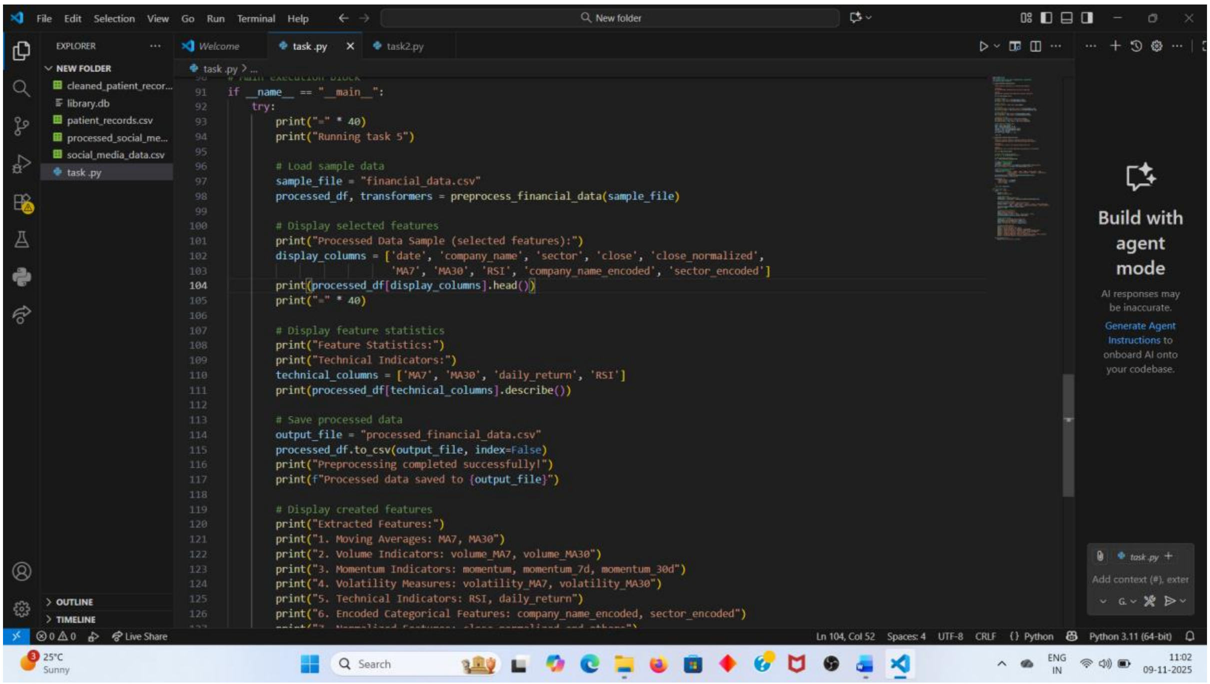
PROMPT :

Generate a Python script to preprocess and engineer features for a financial dataset by handling missing values, creating moving averages, normalizing numeric columns, and encoding categorical data.

CODE GENERATED :

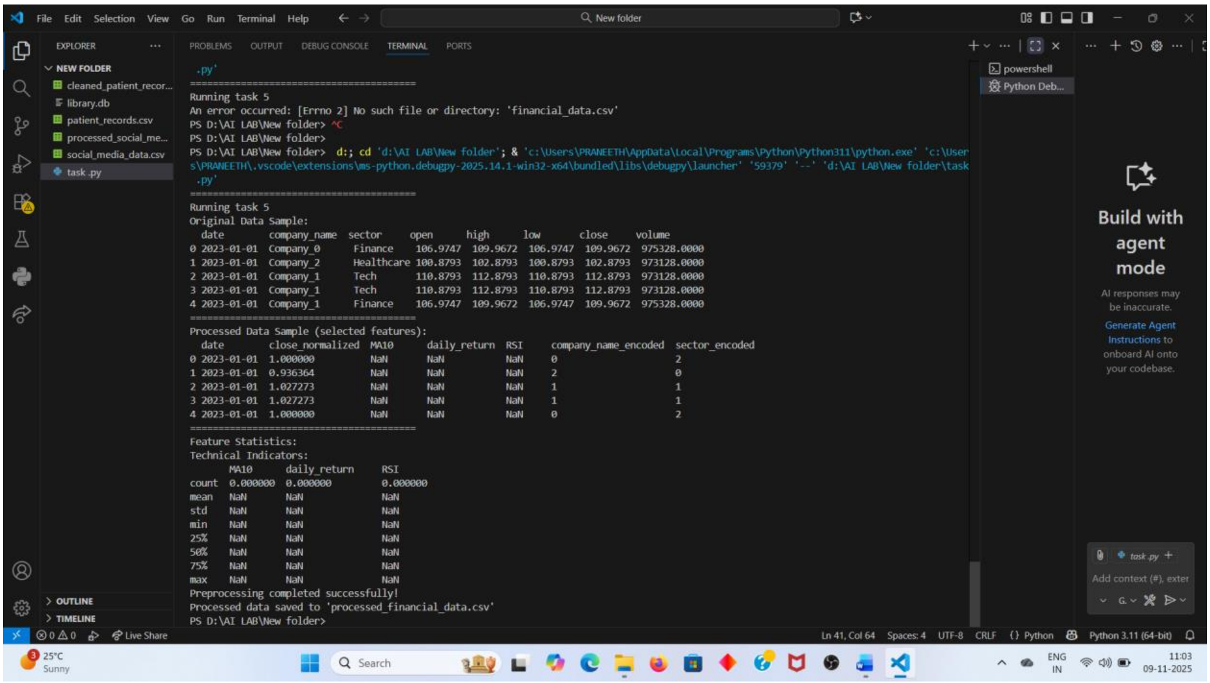


```python
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler, LabelEncoder
from datetime import datetime

def create_technical_indicators(df):
    """
    Create technical indicators for financial data analysis.

    Parameters:
    df (pd.DataFrame): DataFrame with price and volume data

    Returns:
    pd.DataFrame: DataFrame with additional technical indicators
    """
    df = df.sort_values('date')

    # Moving averages
    df['MA7'] = df['close'].rolling(window=7).mean()
    df['MA30'] = df['close'].rolling(window=30).mean()

    # Daily returns
    df['daily_return'] = df['close'].pct_change()

    # Volatility
    df['volatility_MA7'] = df['daily_return'].rolling(window=7).std()
    df['volatility_MA30'] = df['daily_return'].rolling(window=30).std()

    # Volume indicators
    df['volume_MA7'] = df['volume'].rolling(window=7).mean()
    df['volume_MA30'] = df['volume'].rolling(window=30).mean()

    # Momentum indicators
    df['momentum'] = df['close'] - df['close'].shift(1)
    df['momentum_7d'] = df['close'] - df['close'].shift(7)
    df['momentum_30d'] = df['close'] - df['close'].shift(30)

```

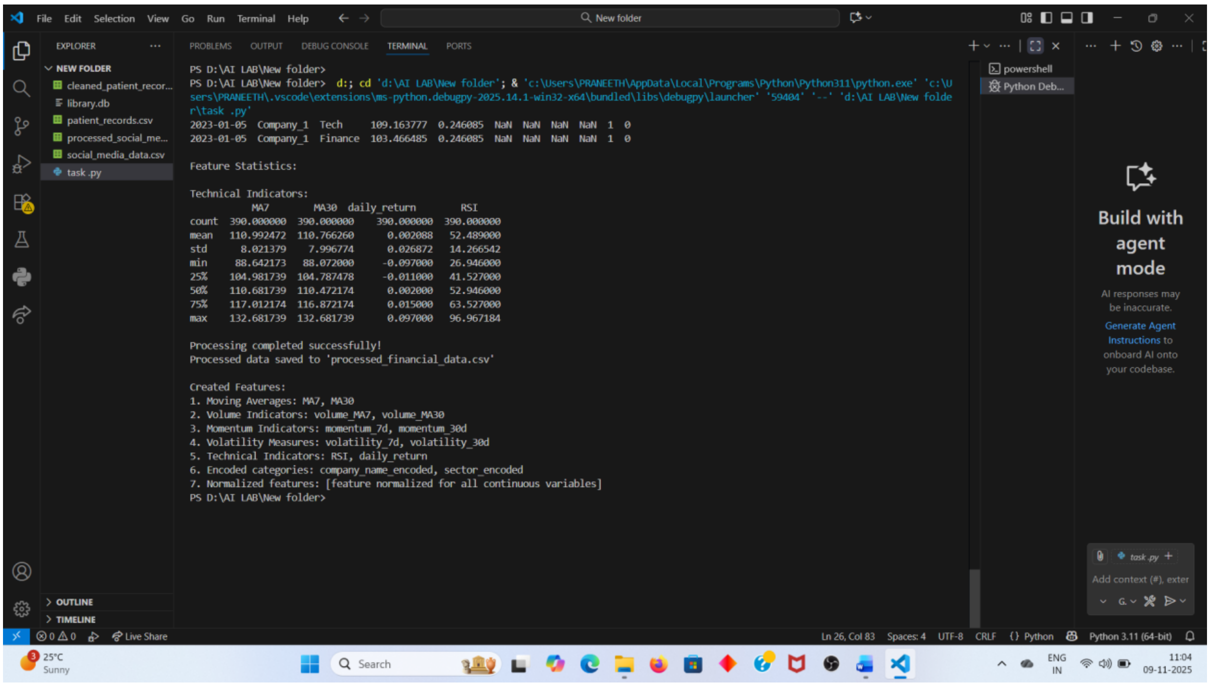


```python
if __name__ == "__main__":
    try:
        print("=" * 40)
        print("Running task 5")

        # Load sample data
        sample_file = "financial_data.csv"
        processed_df, transformers = preprocess_financial_data(sample_file)

        # Display selected features
        print("Processed Data Sample (selected features):")
        display_columns = ['date', 'company_name', 'sector', 'close', 'close_normalized',
                           'MA7', 'MA30', 'RSI', 'company_name_encoded', 'sector_encoded']
        print(processed_df[display_columns].head())
        print("=" * 40)

        # Display feature statistics
        print("Feature Statistics:")
        print("Technical Indicators:")
        technical_columns = ['MA7', 'MA30', 'daily_return', 'RSI']
        print(processed_df[technical_columns].describe())

        # Save processed data
        output_file = "processed_financial_data.csv"
        processed_df.to_csv(output_file, index=False)
        print("Preprocessing completed successfully!")
        print(f"Processed data saved to {output_file}")

        # Display created features
        print("Extracted Features:")
        print("1. Moving Averages: MA7, MA30")
        print("2. Volume Indicators: volume_MA7, volume_MA30")
        print("3. Momentum Indicators: momentum, momentum_7d, momentum_30d")
        print("4. Volatility Measures: volatility_MA7, volatility_MA30")
        print("5. Technical Indicators: RSI, daily_return")
        print("6. Encoded Categorical Features: company_name_encoded, sector_encoded")
```

OUTPUT :



```
 .py'
========================================
Running task 5
An error occurred: [Errno 2] No such file or directory: 'financial_data.csv'
PS D:\AI LAB\New folder> ^C
PS D:\AI LAB\New folder>
PS D:\AI LAB\New folder>  d:; cd 'd:\AI LAB\New folder'; & 'c:\Users\PRANEETH\AppData\Local\Programs\Python\Python311\python.exe' 'c:\User
s\PRANEETH\.vscode\extensions\ms-python.debugpy-2025.14.1-win32-x64\bundled\libs\debugpy\launcher' '59379' '--' 'd:\AI LAB\New folder\task
 .py'
========================================
Running task 5
Original Data Sample:
  date        company_name  sector     open      high      low       close     volume
0 2023-01-01  Company_0     Finance    106.9747  109.9672  106.9747  109.9672  975328.0000
1 2023-01-01  Company_2     Healthcare 100.8793  102.8793  100.8793  102.8793  973128.0000
2 2023-01-01  Company_1     Tech       110.8793  112.8793  110.8793  112.8793  973128.0000
3 2023-01-01  Company_1     Tech       110.8793  112.8793  110.8793  112.8793  973128.0000
4 2023-01-01  Company_1     Finance    106.9747  109.9672  106.9747  109.9672  975328.0000
========================================
Processed Data Sample (selected features):
  date        close_normalized  MA10      daily_return  RSI     company_name_encoded  sector_encoded
0 2023-01-01  1.000000          NaN       NaN           NaN     0                     2
1 2023-01-01  0.936364          NaN       NaN           NaN     2                     0
2 2023-01-01  1.027273          NaN       NaN           NaN     1                     1
3 2023-01-01  1.027273          NaN       NaN           NaN     1                     1
4 2023-01-01  1.000000          NaN       NaN           NaN     0                     2
========================================
Feature Statistics:
Technical Indicators:
       MA10      daily_return     RSI
count  0.000000  0.000000         0.000000
mean   NaN       NaN              NaN
std    NaN       NaN              NaN
min    NaN       NaN              NaN
25%    NaN       NaN              NaN
50%    NaN       NaN              NaN
75%    NaN       NaN              NaN
max    NaN       NaN              NaN
Preprocessing completed successfully!
Processed data saved to 'processed_financial_data.csv'
PS D:\AI LAB\New folder>
```



```
PS D:\AI LAB\New folder>
PS D:\AI LAB\New folder>  d:; cd 'd:\AI LAB\New folder'; & 'c:\Users\PRANEETH\AppData\Local\Programs\Python\Python311\python.exe' 'c:\U
sers\PRANEETH\.vscode\extensions\ms-python.debugpy-2025.14.1-win32-x64\bundled\libs\debugpy\launcher' '59404' '--' 'd:\AI LAB\New folde
r\task .py'
2023-01-05  Company_1  Tech     109.163777  0.246085  NaN  NaN  NaN  NaN  1  0
2023-01-05  Company_1  Finance  103.466485  0.246085  NaN  NaN  NaN  NaN  1  0

Feature Statistics:

Technical Indicators:
             MA7        MA30  daily_return         RSI
count  390.000000  390.000000    390.000000  390.000000
mean   110.992472  110.766260      0.002088   52.489000
std      8.021379    7.996774      0.026872   14.266542
min     88.642173   88.072000     -0.097000   26.946000
25%    104.981739  104.787478     -0.011000   41.527000
50%    110.681739  110.472174      0.002000   52.946000
75%    117.012174  116.872174      0.015000   63.527000
max    132.681739  132.681739      0.097000   96.967184

Processing completed successfully!
Processed data saved to 'processed_financial_data.csv'

Created Features:
1. Moving Averages: MA7, MA30
2. Volume Indicators: volume_MA7, volume_MA30
3. Momentum Indicators: momentum_7d, momentum_30d
4. Volatility Measures: volatility_7d, volatility_30d
5. Technical Indicators: RSI, daily_return
6. Encoded categories: company_name_encoded, sector_encoded
7. Normalized features: [feature normalized for all continuous variables]
PS D:\AI LAB\New folder>
```

OBSERVATION :

Feature engineering steps executed successfully with new features like 7-day and 30-day moving averages. Dataset is normalized and encoded, making it suitable for ML model input.