AI LAB 17

Name: K. Jashuva

Roll No: 2503A51L16

Batch No: 19

TASK 1 - Employee Data Preprocessing

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

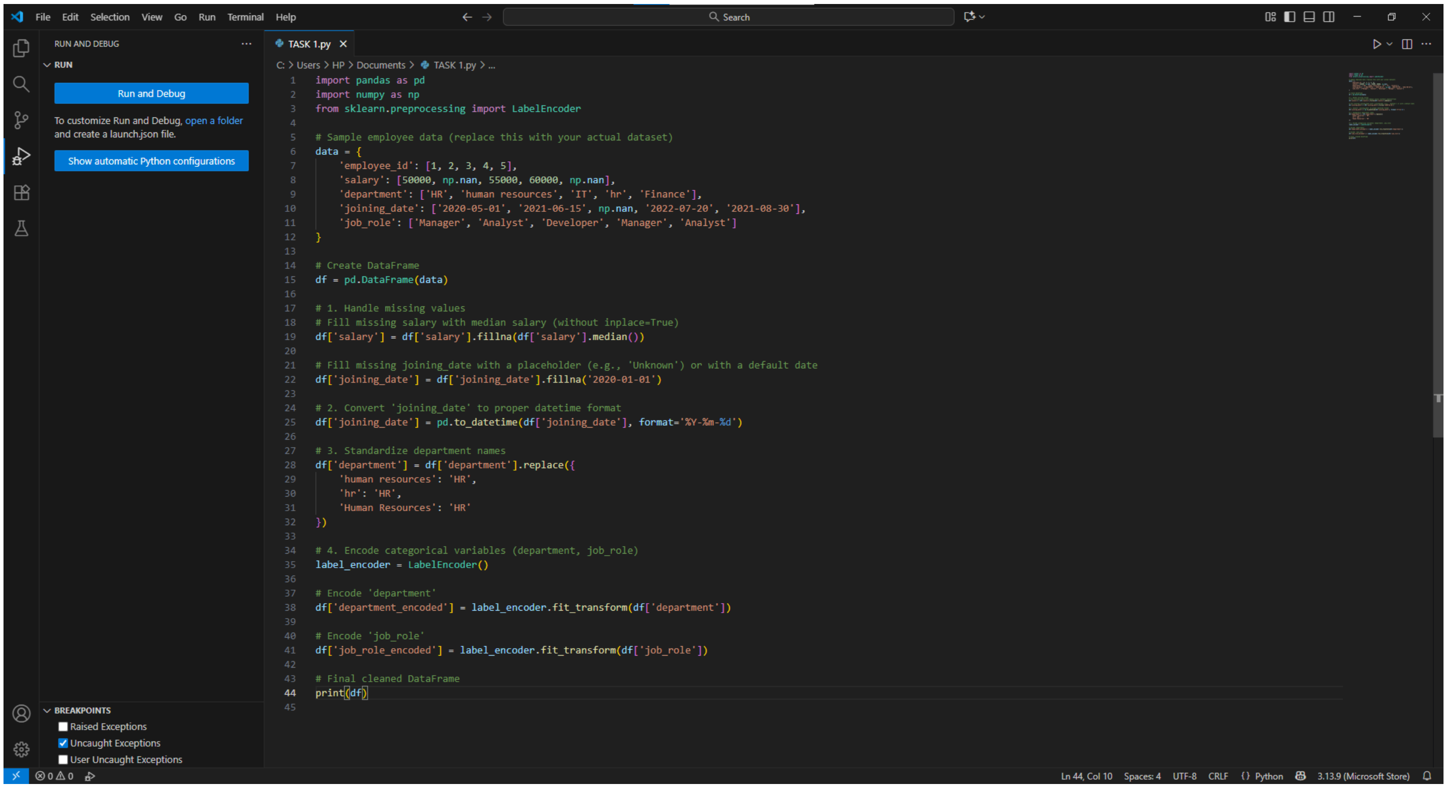
Instructions:

- 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)

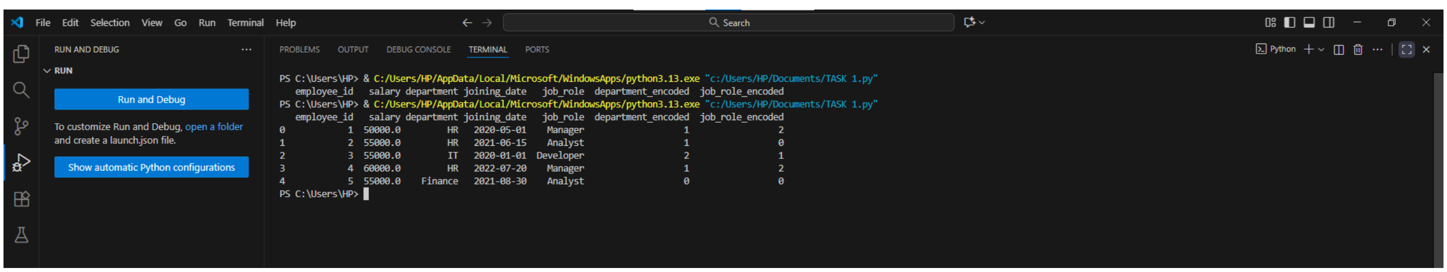
Prompt:

Clean the employee dataset by filling missing values in salary, department, and joining_date, convert joining_date to datetime, standardize department names (e.g., unify “HR” variants), and encode categorical variables (department, job_role) for modeling.

Code :



Output:



Observations:

The dataset contains missing values and inconsistent department names that need cleaning for accuracy. The joining_date column requires conversion to a proper datetime format, and categorical fields like department and job_role should be encoded. After preprocessing, the dataset will be clean, consistent, and ready for analysis or modeling.

Task 2 – Sales Transaction Data Preprocessing

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

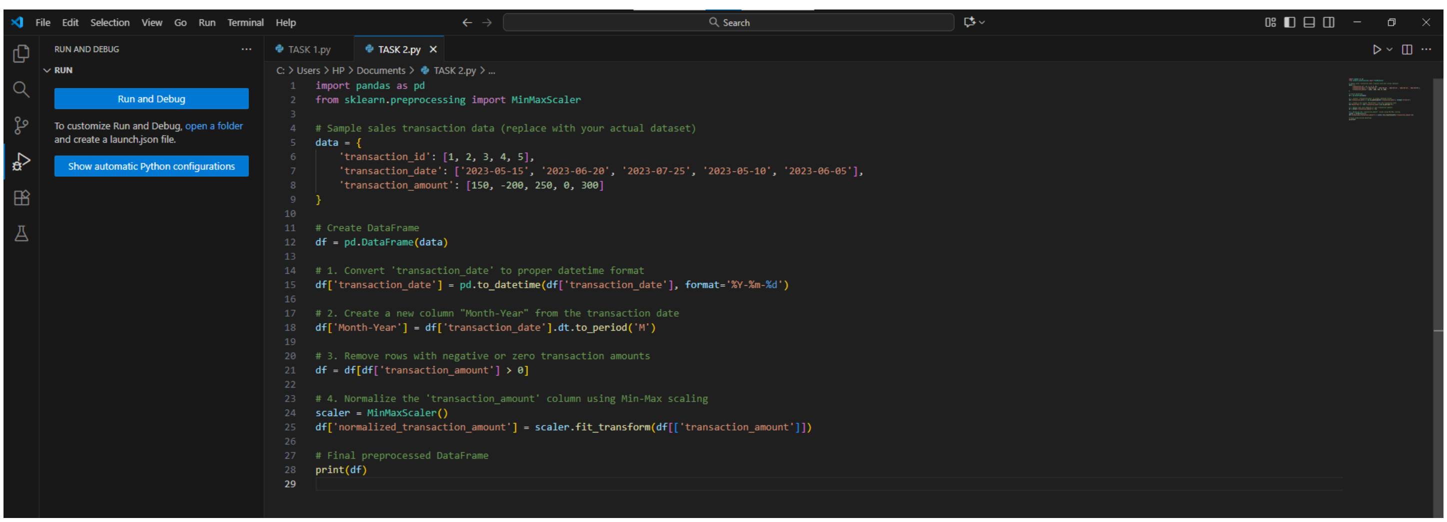
Instructions:

- 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 :

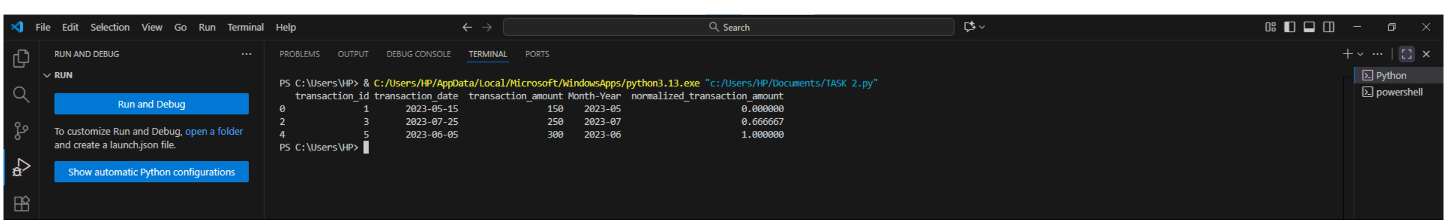
Create a Python script to preprocess a sales transaction dataset by converting transaction dates to datetime format, adding a “Month-Year” column, and removing rows with zero or negative amounts. Normalize the “transaction_amount” column using Min-Max scaling.

Code:



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

# Sample sales transaction data (replace with your actual dataset)
data = {
    'transaction_id': [1, 2, 3, 4, 5],
    'transaction_date': ['2023-05-15', '2023-06-20', '2023-07-25', '2023-05-10', '2023-06-05'],
    'transaction_amount': [150, -200, 250, 0, 300]
}

# Create DataFrame
df = pd.DataFrame(data)

# 1. Convert 'transaction_date' to proper datetime format
df['transaction_date'] = pd.to_datetime(df['transaction_date'], format='%Y-%m-%d')

# 2. Create a new column "Month-Year" from the transaction date
df['Month-Year'] = df['transaction_date'].dt.to_period('M')

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

# 4. Normalize the 'transaction_amount' column using Min-Max scaling
scaler = MinMaxScaler()
df['normalized_transaction_amount'] = scaler.fit_transform(df[['transaction_amount']])

# Final preprocessed DataFrame
print(df)
```

Output:



```
PS C:\Users\HP> & C:/Users/HP/AppData/Local/Microsoft/WindowsApps/python3.13.exe "c:/Users/HP/Documents/TASK 2.py"
   transaction_id transaction_date  transaction_amount Month-Year  normalized_transaction_amount
0               1       2023-05-15                 150    2023-05                       0.000000
2               3       2023-07-25                 250    2023-07                       0.666667
4               5       2023-06-05                 300    2023-06                       1.000000
PS C:\Users\HP>
```

Observations:

The sales transaction dataset requires cleaning and transformation to ensure data accuracy. Transaction dates need to be standardized, and a new Month-Year column will help in time-based analysis. Removing invalid transaction amounts and normalizing values will prepare the data for consistent and reliable analysis or modeling.

Task 3 – Healthcare Patient Records Cleaning

Task:

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).

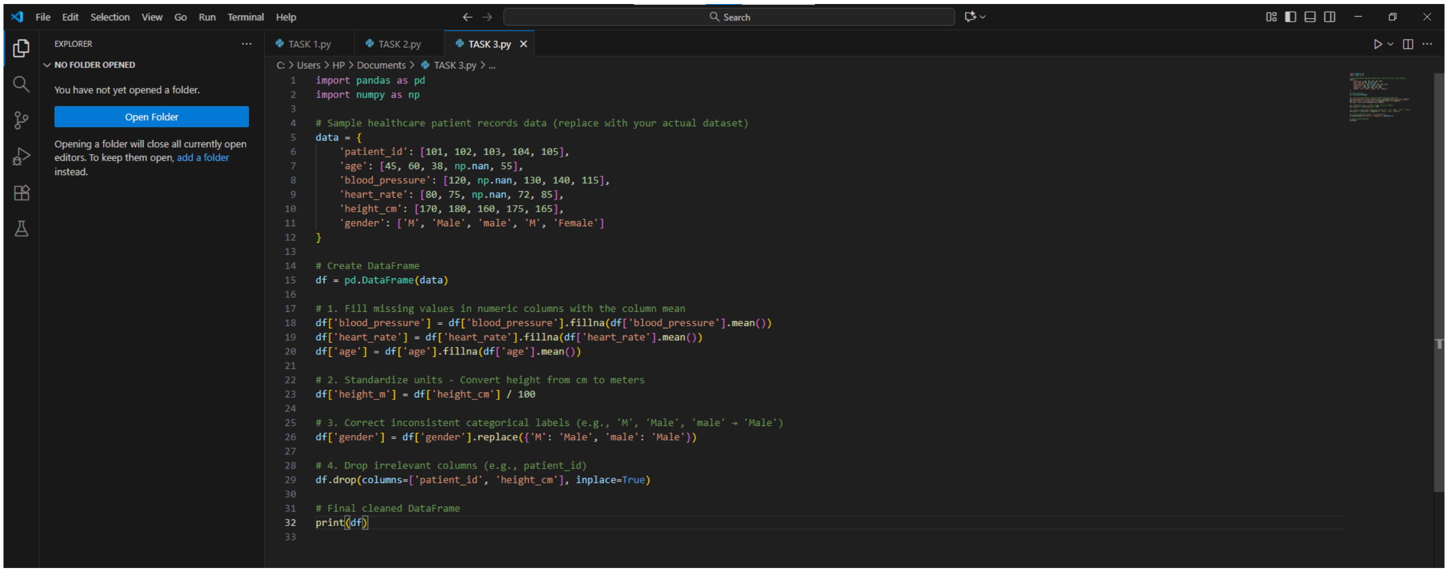
• Correct inconsistent categorical labels (e.g., "M", "Male", "male"→ "Male").

• Drop irrelevant columns such as patient_id after cleaning.

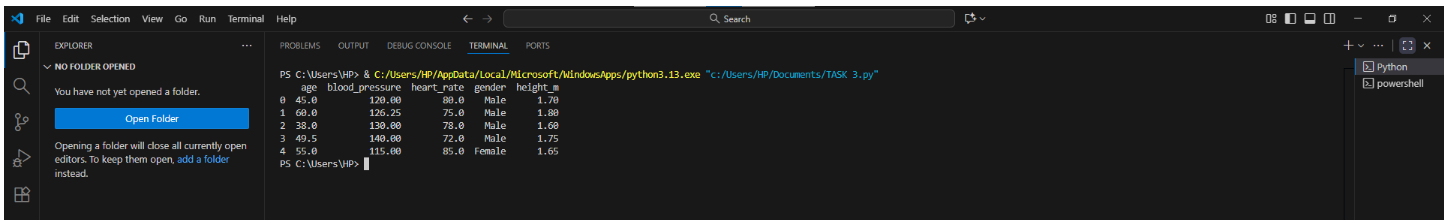
Prompt:

Create a Python script to clean healthcare patient records by filling missing numeric values with the column mean, converting height from centimeters to meters, standardizing gender labels (e.g., "M", "Male", "male" → "Male"), and dropping irrelevant columns like patient_id after cleaning.

Code:



Output:



Observations:

The healthcare dataset contains missing values and inconsistent data formats that need correction for accuracy. Numeric columns require imputation with mean values, and units like height must be standardized for consistency. Cleaning categorical labels and removing irrelevant columns will result in a reliable and well-structured dataset for analysis.

Task 4 – Social Media Sentiment Dataset Preparation

Task:

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

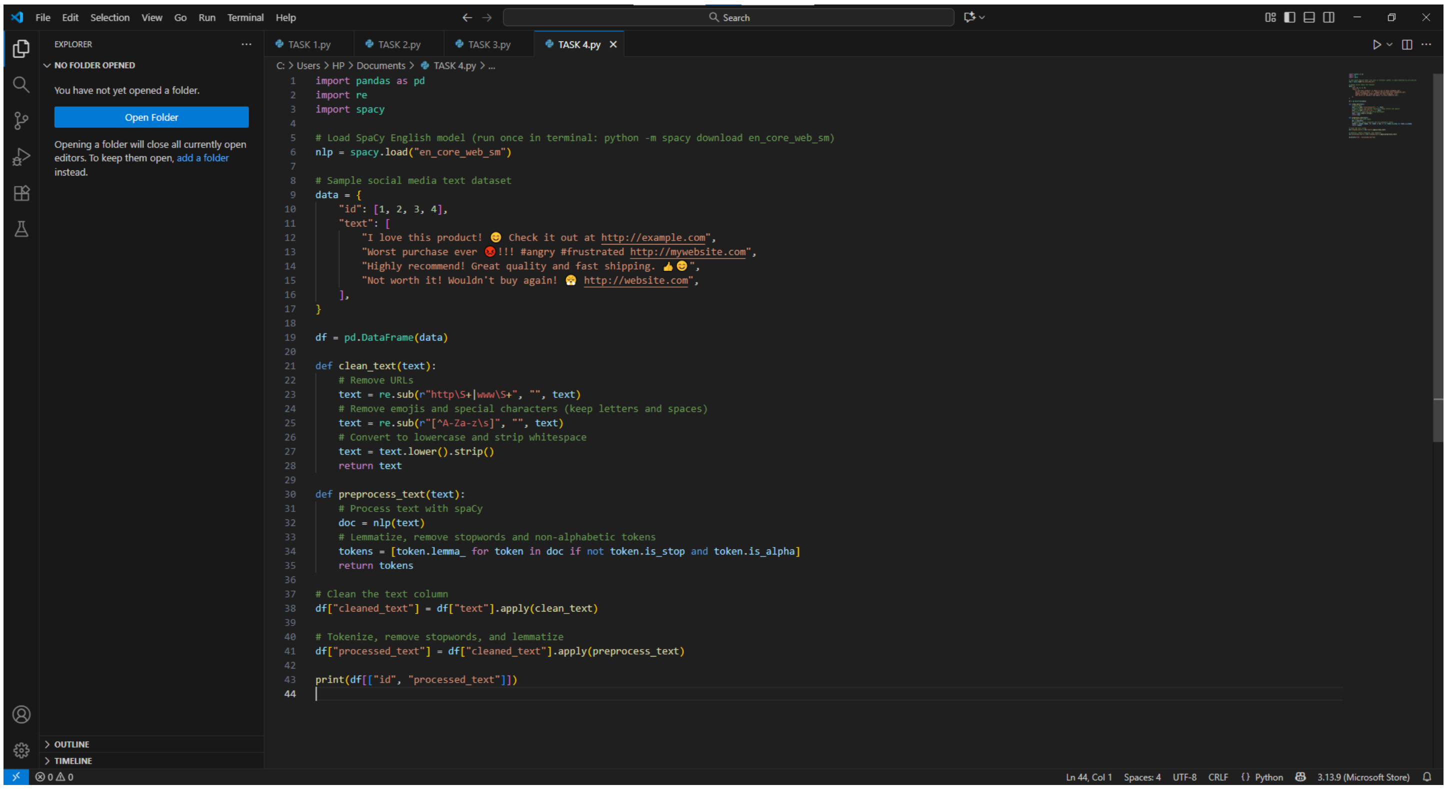
Instructions:

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

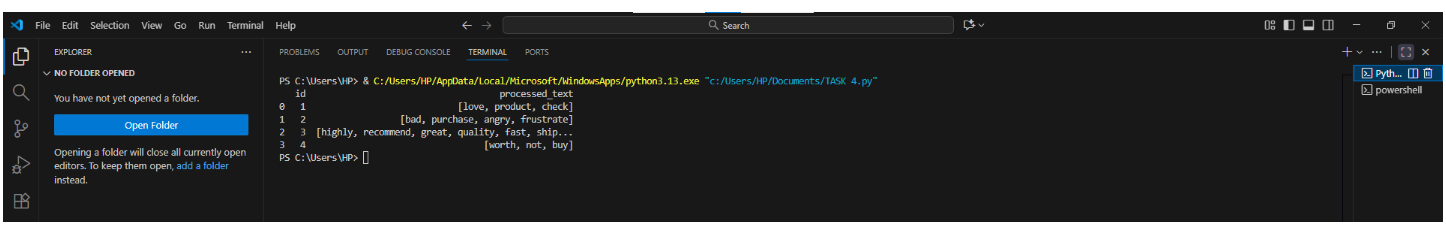
Prompt:

Create a Python script to preprocess a social media sentiment dataset by removing special characters, URLs, and emojis from text. Convert all text to lowercase, tokenize it, remove stopwords, and apply lemmatization to standardize words for accurate sentiment analysis.

Code:



Output:



Observations:

The social media text data contains noise such as emojis, URLs, and special characters that can affect sentiment accuracy. Standardizing text through lowercasing, tokenization, and lemmatization ensures consistency. After preprocessing, the dataset will be cleaner and more suitable for effective sentiment analysis and model training.

Task 5 – Financial Dataset Feature Engineering

Task:

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

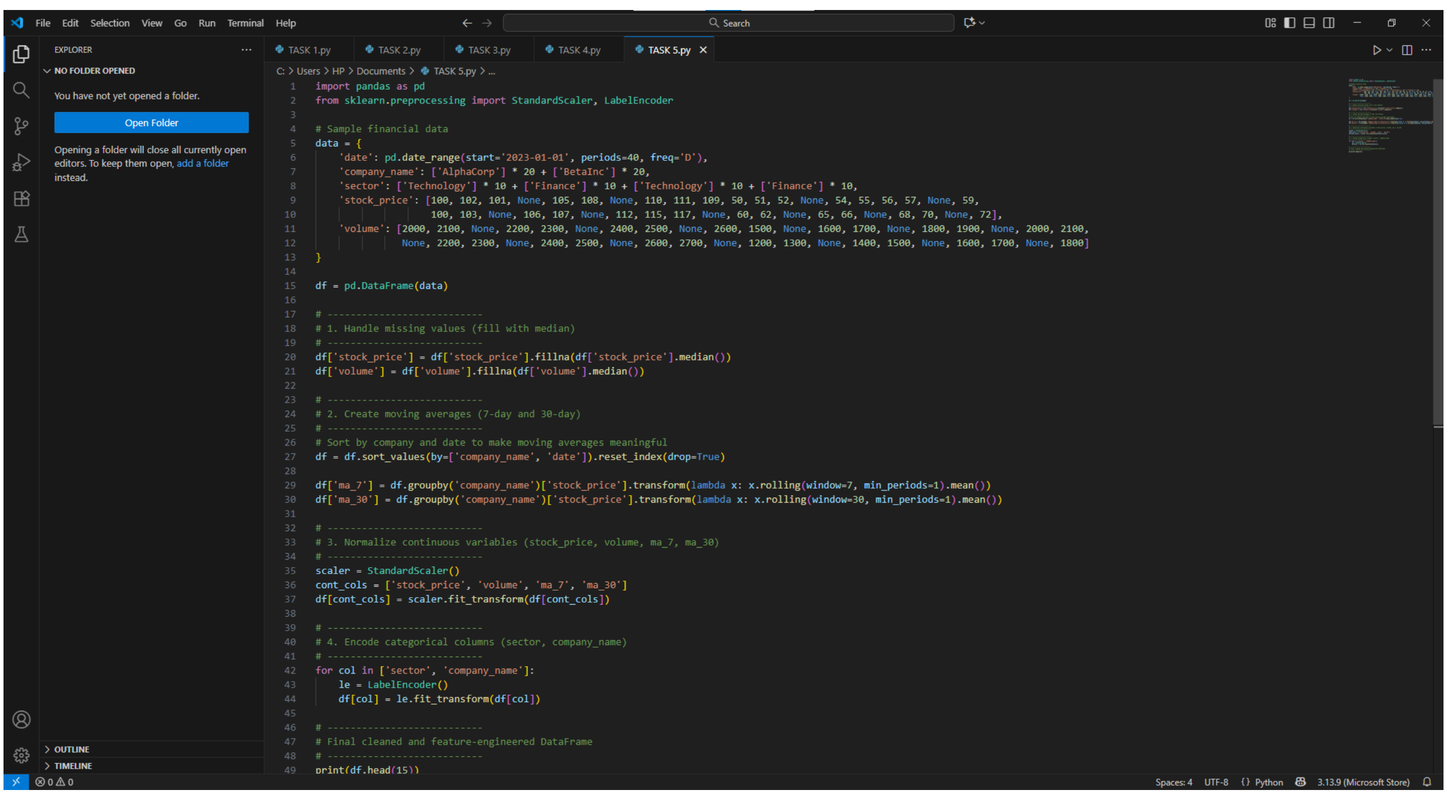
Instructions:

- 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:

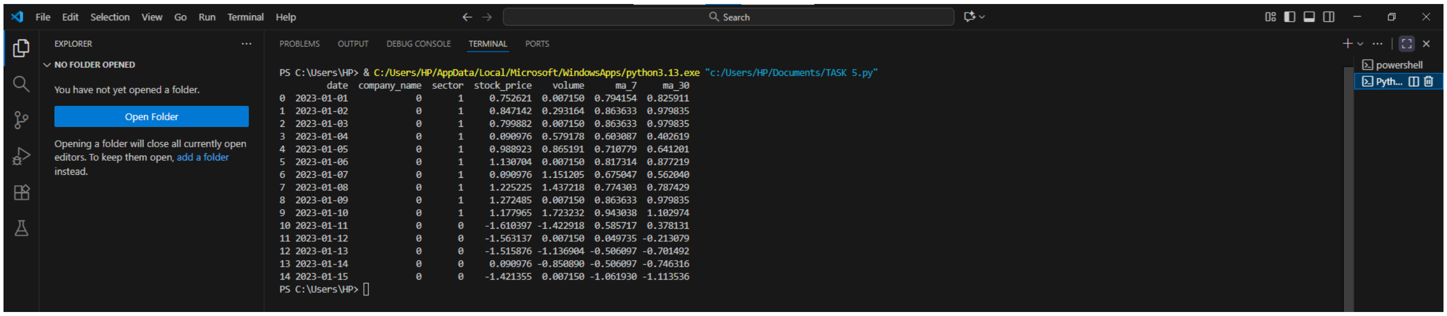
Create a Python script to preprocess a financial dataset by handling missing values in stock price and volume, generating new features like 7-day and 30-day moving averages, normalizing continuous variables using StandardScaler, and encoding categorical columns such as sector and company_name.

Code:



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

# Sample financial data
data = {
    'date': pd.date_range(start='2023-01-01', periods=40, freq='D'),
    'company_name': ['AlphaCorp'] * 20 + ['BetaInc'] * 20,
    'sector': ['Technology'] * 10 + ['Finance'] * 10 + ['Technology'] * 10 + ['Finance'] * 10,
    'stock_price': [100, 102, 101, None, 105, 108, None, 110, 111, 109, 50, 51, 52, None, 54, 55, 56, 57, None, 59,
                    100, 103, None, 106, 107, None, 112, 115, 117, None, 60, 62, None, 65, 66, None, 68, 70, None, 72],
    'volume': [2000, 2100, None, 2200, 2300, None, 2400, 2500, None, 2600, 1500, None, 1600, 1700, None, 1800, 1900, None, 2000, 2100,
               None, 2200, 2300, None, 2400, 2500, None, 2600, 2700, None, 1200, 1300, None, 1400, 1500, None, 1600, 1700, None, 1800]
}

df = pd.DataFrame(data)

# --------------------------
# 1. Handle missing values (fill with median)
# --------------------------
df['stock_price'] = df['stock_price'].fillna(df['stock_price'].median())
df['volume'] = df['volume'].fillna(df['volume'].median())

# --------------------------
# 2. Create moving averages (7-day and 30-day)
# --------------------------
# Sort by company and date to make moving averages meaningful
df = df.sort_values(by=['company_name', 'date']).reset_index(drop=True)

df['ma_7'] = df.groupby('company_name')['stock_price'].transform(lambda x: x.rolling(window=7, min_periods=1).mean())
df['ma_30'] = df.groupby('company_name')['stock_price'].transform(lambda x: x.rolling(window=30, min_periods=1).mean())

# --------------------------
# 3. Normalize continuous variables (stock_price, volume, ma_7, ma_30)
# --------------------------
scaler = StandardScaler()
cont_cols = ['stock_price', 'volume', 'ma_7', 'ma_30']
df[cont_cols] = scaler.fit_transform(df[cont_cols])

# --------------------------
# 4. Encode categorical columns (sector, company_name)
# --------------------------
for col in ['sector', 'company_name']:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])

# --------------------------
# Final cleaned and feature-engineered DataFrame
# --------------------------
print(df.head(15))
```

Output:



Observations:

The financial dataset requires handling missing values to maintain data reliability. Creating moving average features will help capture market trends, while normalization ensures balanced scaling for modeling. Encoding categorical variables makes the dataset suitable for machine learning and predictive analytics.