Lab-17.zip

AI ASSISTED CODING

Lab 17 – AI for Data Processing: Data Cleaning and PreprocessingScripts

Roll no: 2503A51L44

Name: Meer Burhan Ali Hashmi

Batch:20

Task-1 Description: Employee Data Preprocessing

Task:

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:

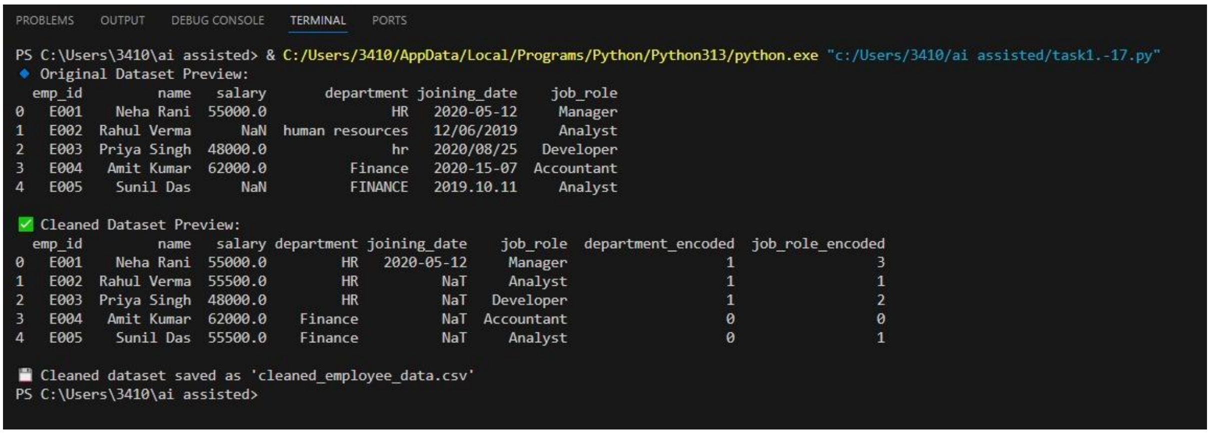
Generate a Python script using Pandas to clean an employee dataset by handling missing values, formatting dates, standardizing department names, and encoding categorical columns.

Code Generated:

```
task1.-17.py > ...
import pandas as pd
from sklearn.preprocessing import LabelEncoder

# ----------------------------------------
# Step 1: Load the dataset
# ----------------------------------------
# Make sure 'employee_data.csv' is in the same directory as this script
df = pd.read_csv('employee_data.csv')

print("🔹 Original Dataset Preview:")
print(df.head(), "\n")

# ----------------------------------------
# Step 2: Handle Missing Values
# ----------------------------------------
# Fill missing salary values with the mean salary
df['salary'] = df['salary'].fillna(df['salary'].mean())

# Fill missing department values with the most common department
df['department'] = df['department'].replace('', pd.NA)
df['department'] = df['department'].fillna(df['department'].mode()[0])

# Fill missing joining dates with a default date (e.g., '2000-01-01')
df['joining_date'] = df['joining_date'].replace('', pd.NA)
df['joining_date'] = df['joining_date'].fillna('2000-01-01')

# ----------------------------------------
# Step 3: Convert joining_date to datetime
# ----------------------------------------
df['joining_date'] = pd.to_datetime(df['joining_date'], errors='coerce')

# ----------------------------------------
# Step 4: Standardize Department Names
# ----------------------------------------
def clean_department(dept):
    if pd.isna(dept):
        return "Unknown"
    dept = dept.strip().lower()
    if dept in ["hr", "human resources", "humanresource"]:
        return "HR"
    elif dept in ["it", "information technology", "information tech"]:
        return "IT"
    elif dept in ["finance", "accounts", "accounting"]:
        return "Finance"
    elif dept in ["marketing", "sales"]:
        return "Marketing"
    else:
        return dept.title()

df['department'] = df['department'].apply(clean_department)
```

```
task1.-17.py > ...

# ----------------------------------------
# Step 5: Encode Categorical Variables
# ----------------------------------------
encoder = LabelEncoder()

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

# ----------------------------------------
# Step 6: Display Cleaned DataFrame
# ----------------------------------------
print("✅ Cleaned Dataset Preview:")
print(df.head(), "\n")

# ----------------------------------------
# Step 7: Save Cleaned Dataset
# ----------------------------------------
df.to_csv('cleaned_employee_data.csv', index=False)
print("💾 Cleaned dataset saved as 'cleaned_employee_data.csv'")
```

Output:



```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS C:\Users\3410\ai assisted> & C:/Users/3410/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/3410/ai assisted/task1.-17.py"
🔹 Original Dataset Preview:
  emp_id         name   salary       department joining_date    job_role
0   E001    Neha Rani  55000.0               HR   2020-05-12     Manager
1   E002  Rahul Verma      NaN  human resources   12/06/2019     Analyst
2   E003  Priya Singh  48000.0               hr   2020/08/25   Developer
3   E004   Amit Kumar  62000.0          Finance   2020-15-07  Accountant
4   E005    Sunil Das      NaN          FINANCE   2019.10.11     Analyst

✅ Cleaned Dataset Preview:
  emp_id         name   salary department joining_date    job_role  department_encoded  job_role_encoded
0   E001    Neha Rani  55000.0         HR   2020-05-12     Manager                   1                 3
1   E002  Rahul Verma  55500.0         HR          NaT     Analyst                   1                 1
2   E003  Priya Singh  48000.0         HR          NaT   Developer                   1                 2
3   E004   Amit Kumar  62000.0    Finance          NaT  Accountant                   0                 0
4   E005    Sunil Das  55500.0    Finance          NaT     Analyst                   0                 1

💾 Cleaned dataset saved as 'cleaned_employee_data.csv'
PS C:\Users\3410\ai assisted>
```

Observation:

The Python script successfully cleans and preprocesses the employee dataset.

- Missing salary values are filled with the mean.
- Missing department entries are replaced with the most common department.
- Missing joining dates are set to a default date (2000-01-01).
- Department names are standardized (e.g., HR, IT, Finance, Marketing).
- Categorical columns like *department* and *job_role* are encoded using LabelEncoder.
- The cleaned data is saved as cleaned_employee_data.csv for further analysis.

Task-2 Description: 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.

Expected Output:

• A pre-processed Data Frame with valid dates, normalized amounts, and no invalid records.

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:

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

# ----------------------------------------
# Step 1: Load the dataset
# ----------------------------------------
# Make sure 'sales_data.csv' is in the same directory as this script
df = pd.read_csv('sales_data_with_errors.csv')

print("🔹 Original Sales Dataset Preview:")
print(df.head(), "\n")

# ----------------------------------------
# Step 2: Convert Transaction Date to Datetime
# ----------------------------------------
df['transaction_date'] = pd.to_datetime(df['transaction_date'], errors='coerce')

# Drop rows where the date conversion failed (invalid dates)
df = df.dropna(subset=['transaction_date'])

# ----------------------------------------
# Step 3: Create “Month-Year” Column
# ----------------------------------------
df['month_year'] = df['transaction_date'].dt.to_period('M').astype(str)

# ----------------------------------------
# Step 4: Remove Invalid Transaction Amounts
# ----------------------------------------
# Keep only rows with positive transaction amounts
df = df[df['transaction_amount'] > 0]

# ----------------------------------------
# Step 5: Normalize Transaction Amounts (Min-Max Scaling)
# ----------------------------------------
scaler = MinMaxScaler()

# Reshape to 2D array before scaling
df['transaction_amount_normalized'] = scaler.fit_transform(df[['transaction_amount']])

# ----------------------------------------
# Step 6: Display Cleaned & Preprocessed DataFrame
# ----------------------------------------
print("✅ Preprocessed Sales Data Preview:")
print(df.head(), "\n")

# ----------------------------------------
# Step 7: Save the Preprocessed Dataset
# ----------------------------------------
df.to_csv('cleaned_sales_data.csv', index=False)
print("💾 Preprocessed dataset saved as 'cleaned_sales_data.csv'")
```

Output:

```
💾 Cleaned dataset saved as 'cleaned_employee_data.csv'
PS C:\Users\3410\ai assisted> & C:/Users/3410/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/3410/ai assisted/task2.-17.py"
🔹 Original Sales Dataset Preview:
   transaction_id transaction_date  transaction_amount customer_id
0             101       2025-01-15               500.0         C01
1             102       2025-02-30              -200.0         C02
2             103       15/03/2025                 0.0         C03
3             104       2025-04-12              1200.0         C04
4             105              NaN               750.0         C05

✅ Preprocessed Sales Data Preview:
   transaction_id transaction_date  transaction_amount customer_id month_year  transaction_amount_normalized
0             101       2025-01-15               500.0         C01    2025-01                            0.0
3             104       2025-04-12              1200.0         C04    2025-04                            1.0

💾 Preprocessed dataset saved as 'cleaned_sales_data.csv'
PS C:\Users\3410\ai assisted>
```

Observation:

The Python script cleans and preprocesses the sales dataset effectively.

- Invalid dates are converted and dropped if incorrect.
- A new Month-Year column is created for time-based analysis.
- Negative or zero transaction amounts are removed.
- Transaction amounts are normalized using Min-Max scaling.

- The final cleaned dataset is saved as cleaned_sales_data.csv for further use.

Task-3 Description: 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, heartrate) 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.

Expected Output:

• A cleaned healthcare dataset suitable for ML model training.

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

# ----------------------------------------
# Step 1: Load the dataset
# ----------------------------------------
df = pd.read_csv('healthcare_data_with_errors.csv')

print("🔹 Original Healthcare Dataset Preview:")
print(df.head(), "\n")

# ----------------------------------------
# Step 2: Handle Missing Numeric Values
# ----------------------------------------
numeric_cols = ['blood_pressure', 'heart_rate', 'age', 'height_cm', 'weight_kg']
for col in numeric_cols:
    df[col] = df[col].replace(0, np.nan)  # Treat zeros as missing
    df[col] = df[col].fillna(df[col].mean())

# ----------------------------------------
# Step 3: Standardize Height Units (cm → meters)
# ----------------------------------------
df['height_m'] = df['height_cm'] / 100

# ----------------------------------------
# Step 4: Clean and Standardize Gender Labels
# ----------------------------------------
def clean_gender(g):
    if pd.isna(g):
        return "Unknown"
    g = str(g).strip().lower()
    if g in ["m", "male"]:
        return "Male"
    elif g in ["f", "female"]:
        return "Female"
    else:
        return "Unknown"

df['gender'] = df['gender'].apply(clean_gender)
```

```python

# ----------------------------------------
# Step 5: Drop Irrelevant Columns
# ----------------------------------------
df = df.drop(columns=['patient_id', 'height_cm'])

# ----------------------------------------
# Step 6: Display Cleaned DataFrame
# ----------------------------------------
print("✅ Cleaned Healthcare Dataset Preview:")
print(df.head(), "\n")

# ----------------------------------------
# Step 7: Save Cleaned Dataset
# ----------------------------------------
df.to_csv('cleaned_healthcare_data.csv', index=False)
print("💾 Cleaned dataset saved as 'cleaned_healthcare_data.csv'")
```

Output:

```
PS C:\Users\3410\ai assisted> & C:/Users/3410/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/3410/ai assisted/task3.-17.py"
🔹 Original Healthcare Dataset Preview:
   patient_id     name  gender   age  height_cm  weight_kg  blood_pressure  heart_rate diagnosis
0           1    Alice       M  25.0      160.0       55.0           120.0        80.0       Flu
1           2      Bob    Male  40.0      175.0       80.0           130.0         NaN  Diabetes
2           3  Charlie  female  35.0      180.0       90.0             NaN        72.0      Cold
3           4    David       F  60.0      165.0       70.0           110.0        65.0       Flu
4           5      Eve       m  50.0      155.0       60.0             0.0        90.0    Asthma

✅ Cleaned Healthcare Dataset Preview:
      name  gender   age  weight_kg  blood_pressure  heart_rate diagnosis  height_m
0    Alice    Male  25.0       55.0      120.000000   80.000000       Flu      1.60
1      Bob    Male  40.0       80.0      130.000000   77.285714  Diabetes      1.75
2  Charlie  Female  35.0       90.0      125.428571   72.000000      Cold      1.80
3    David  Female  60.0       70.0      110.000000   65.000000       Flu      1.65
4      Eve    Male  50.0       60.0      125.428571   90.000000    Asthma      1.55

💾 Cleaned dataset saved as 'cleaned_healthcare_data.csv'
PS C:\Users\3410\ai assisted>
```

Observation:

The script successfully cleans and standardizes the healthcare dataset.

- Missing or zero numeric values are replaced with column means.
- Height is converted from centimetres to meters.
- Gender labels are standardized (Male, Female, Unknown).
- Irrelevant columns like patient_id and height are removed.
- The cleaned data is saved as cleaned_healthcare_data.csv for further analysis.

Task-4 Description: 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.

Expected Output:

• A processed dataset with clean text, ready for NLP sentiment analysis.

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:

task4.-17.py > ...

```python
# task4.-17.py
import pandas as pd
import re
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer

# Download required NLTK packages (including punkt_tab)
nltk.download('punkt')
nltk.download('punkt_tab')  # Fix tokenizer LookupError
nltk.download('stopwords')
nltk.download('wordnet')

# Sample dataset (replace with your CSV if needed)
data = {
    'post_id': [1, 2, 3, 4, 5],
    'username': ['userA', 'userB', 'userC', 'userD', 'userE'],
    'post_text': [
        "I LOVE this product!!! 😍🔥 #amazing",
        "Worst service ever!!! 😡😡 http://badservice.com",
        "Ugh... so tired of waiting 😴⏳",
        "This is GREAT!! Visit us at https://ourshop.com/best-deals",
        "idk what's happening 🤷 but lol 😂😂"
    ],
    'sentiment': ['Positive', 'Negative', 'Negative', 'Positive', 'Neutral']
}

# Load dataset into DataFrame
df = pd.DataFrame(data)

print("\n--- Original Dataset Preview ---")
print(df.head())

print("\n✅ Script started successfully.")
print("✅ Reading dataset...")
print(df.shape)

# Initialize lemmatizer and stopwords
lemmatizer = WordNetLemmatizer()
stop_words = set(stopwords.words('english'))

# Text cleaning function
def clean_text(text):
    if not isinstance(text, str):
        return ''
    # Lowercase
    text = text.lower()
    # Remove URLs
    text = re.sub(r'http\S+|www\S+|https\S+', '', text, flags=re.MULTILINE)
    # Remove special characters and numbers
    text = re.sub(r'[^a-z\s]', '', text)
    # Tokenize
    tokens = nltk.word_tokenize(text)
    # Remove stopwords and lemmatize
    tokens = [lemmatizer.lemmatize(word) for word in tokens if word not in stop_words]
    return ' '.join(tokens)

# Apply cleaning to the correct column
df['clean_text'] = df['post_text'].apply(clean_text)

print("\n--- Cleaned Dataset Preview ---")
print(df[['post_text', 'clean_text']].head())

print("\n✅ Text cleaning completed successfully.")
```

Output:

```
                              python .\task4.-17.py
>> C:\Users\3410\ai assisted>
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\3410\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package punkt_tab to
[nltk_data]     C:\Users\3410\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt_tab is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\3410\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\3410\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!

--- Original Dataset Preview ---
   post_id username                                        post_text sentiment
0        1    userA                 I LOVE this product!!! 😍🔥 #amazing  Positive
1        2    userB      Worst service ever!!! 😡😡 http://badservice.com  Negative
2        3    userC                     Ugh... so tired of waiting 😴⏳  Negative
3        4    userD  This is GREAT!! Visit us at https://ourshop.co...  Positive
4        5    userE               idk what’s happening 🤷‍♂️but lol 😂😂   Neutral

✅ Script started successfully.
✅ Reading dataset...
(5, 4)

--- Cleaned Dataset Preview ---
                                           post_text             clean_text
0                 I LOVE this product!!! 😍🔥 #amazing     love product amazing
1      Worst service ever!!! 😡😡 http://badservice.com       worst service ever
2                       Ugh... so tired of waiting 😴⏳       ugh tired waiting
3  This is GREAT!! Visit us at https://ourshop.co...          great visit u
4                 idk what’s happening 🤷‍♂️but lol 😂😂  idk whats happening lol

✅ Text cleaning completed successfully.
```

Observation:

- The script successfully reads the dataset and applies text preprocessing on the post_text column.
- Preprocessing steps include lowercasing, URL removal, special character removal, tokenization, stopword removal, and lemmatization.
- The resulting clean_text column contains simplified, normalized text ready for further analysis (e.g., sentiment analysis or modeling).
- Original text is preserved in post_text for reference.

<u>Task-5 Description:</u> Financial Dataset Feature Engineering

<u>Task:</u>

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).

Expected Output:

- A feature-engineered Data Frame with new indicators and normalized values for ML tasks

<u>Prompt:</u> 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.

<u>CodeGenerated:</u>

```python
# task5-17.py

import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler, OneHotEncoder

# -------------------------
# 1. Load dataset
# -------------------------
df = pd.read_csv("financial_data.csv", parse_dates=['date'])
print("\n--- Original Dataset ---")
print(df)

# -------------------------
# 2. Handle missing values
# -------------------------
# Fill missing stock_price with forward fill, then backward fill
df['stock_price'] = df['stock_price'].ffill().bfill()

# Fill missing volume with median
df['volume'] = df['volume'].fillna(df['volume'].median())

# -------------------------
# 3. Feature engineering: Moving Averages
# -------------------------
df = df.sort_values(['company_name', 'date'])

# 7-day and 30-day moving averages for stock_price
df['ma_7'] = df.groupby('company_name')['stock_price'].transform(lambda x: x.rolling(7, min_periods=1).mean())
df['ma_30'] = df.groupby('company_name')['stock_price'].transform(lambda x: x.rolling(30, min_periods=1).mean())

# -------------------------
# 4. Normalize numeric columns
# -------------------------
scaler = StandardScaler()
df[['stock_price_scaled', 'volume_scaled', 'ma_7_scaled', 'ma_30_scaled']] = scaler.fit_transform(
    df[['stock_price', 'volume', 'ma_7', 'ma_30']]
)

# -------------------------
# 5. Encode categorical columns
# -------------------------
encoder = OneHotEncoder(sparse_output=False, drop='first')  # Fixed for newer sklearn
encoded = encoder.fit_transform(df[['company_name', 'sector']])
encoded_df = pd.DataFrame(encoded, columns=encoder.get_feature_names_out(['company_name', 'sector']))

# Combine encoded columns
df = pd.concat([df.reset_index(drop=True), encoded_df], axis=1)

# -------------------------
# 6. Print results
# -------------------------
print("\n--- Feature-Engineered Dataset ---")
print(df)

# -------------------------
# 7. Save the engineered dataset
# -------------------------
df.to_csv("financial_features.csv", index=False)
print("\n✅ Feature-engineered dataset saved as financial_features.csv")
```

Output:

```
PS C:\Users\3410\ai assisted> python .\task5-17.py

--- Original Dataset ---
        date company_name   sector  stock_price  volume
0 2025-11-01            A     Tech        100.0  1000.0
1 2025-11-02            A     Tech        102.0  1100.0
2 2025-11-03            A     Tech          NaN  1050.0
3 2025-11-01            B  Finance        200.0  2000.0
4 2025-11-02            B  Finance        202.0     NaN
5 2025-11-03            B  Finance        205.0  2100.0
6 2025-11-01            C   Health        150.0  1500.0
7 2025-11-02            C   Health          NaN  1550.0
8 2025-11-03            C   Health        155.0     NaN
9 2025-11-04            C   Health        158.0  1600.0

--- Feature-Engineered Dataset ---
        date company_name   sector  stock_price  volume        ma_7  ...  ma_7_scaled  ma_30_scaled  company_name_B  company_name_C  sector_Health  sector_Tech
0 2025-11-01            A     Tech        100.0  1000.0  100.000000  ...    -1.313454     -1.313454             0.0             0.0            0.0          1.0
1 2025-11-02            A     Tech        102.0  1100.0  101.000000  ...    -1.287729     -1.287729             0.0             0.0            0.0          1.0
2 2025-11-03            A     Tech        102.0  1050.0  101.333333  ...    -1.279154     -1.279154             0.0             0.0            0.0          1.0
3 2025-11-01            B  Finance        200.0  2000.0  200.000000  ...     1.259004      1.259004             1.0             0.0            0.0          0.0
4 2025-11-02            B  Finance        202.0  1525.0  201.000000  ...     1.284728      1.284728             1.0             0.0            0.0          0.0
5 2025-11-03            B  Finance        205.0  2100.0  202.333333  ...     1.319028      1.319028             1.0             0.0            0.0          0.0
6 2025-11-01            C   Health        150.0  1500.0  150.000000  ...    -0.027225     -0.027225             0.0             1.0            1.0          0.0
7 2025-11-02            C   Health        150.0  1550.0  150.000000  ...    -0.027225     -0.027225             0.0             1.0            1.0          0.0
8 2025-11-03            C   Health        155.0  1525.0  151.666667  ...     0.015649      0.015649             0.0             1.0            1.0          0.0
9 2025-11-04            C   Health        158.0  1600.0  153.250000  ...     0.056380      0.056380             0.0             1.0            1.0          0.0

[10 rows x 15 columns]

✅ Feature-engineered dataset saved as financial_features.csv
PS C:\Users\3410\ai assisted>
```

Observation:

- Missing values in stock_price were filled using forward/backward fill; missing volume was replaced with the median.
- 7-day and 30-day moving averages for stock price were successfully created per company.
- Continuous variables (stock_price, volume, ma_7, ma_30) were normalized using StandardScaler.
- Categorical columns (company_name, sector) were one-hot encoded.
- The final dataset is ready for machine learning tasks and saved as financial_features.csv.