AI ASSINGMENT 17.4

ROLL NO: 2403A52397.

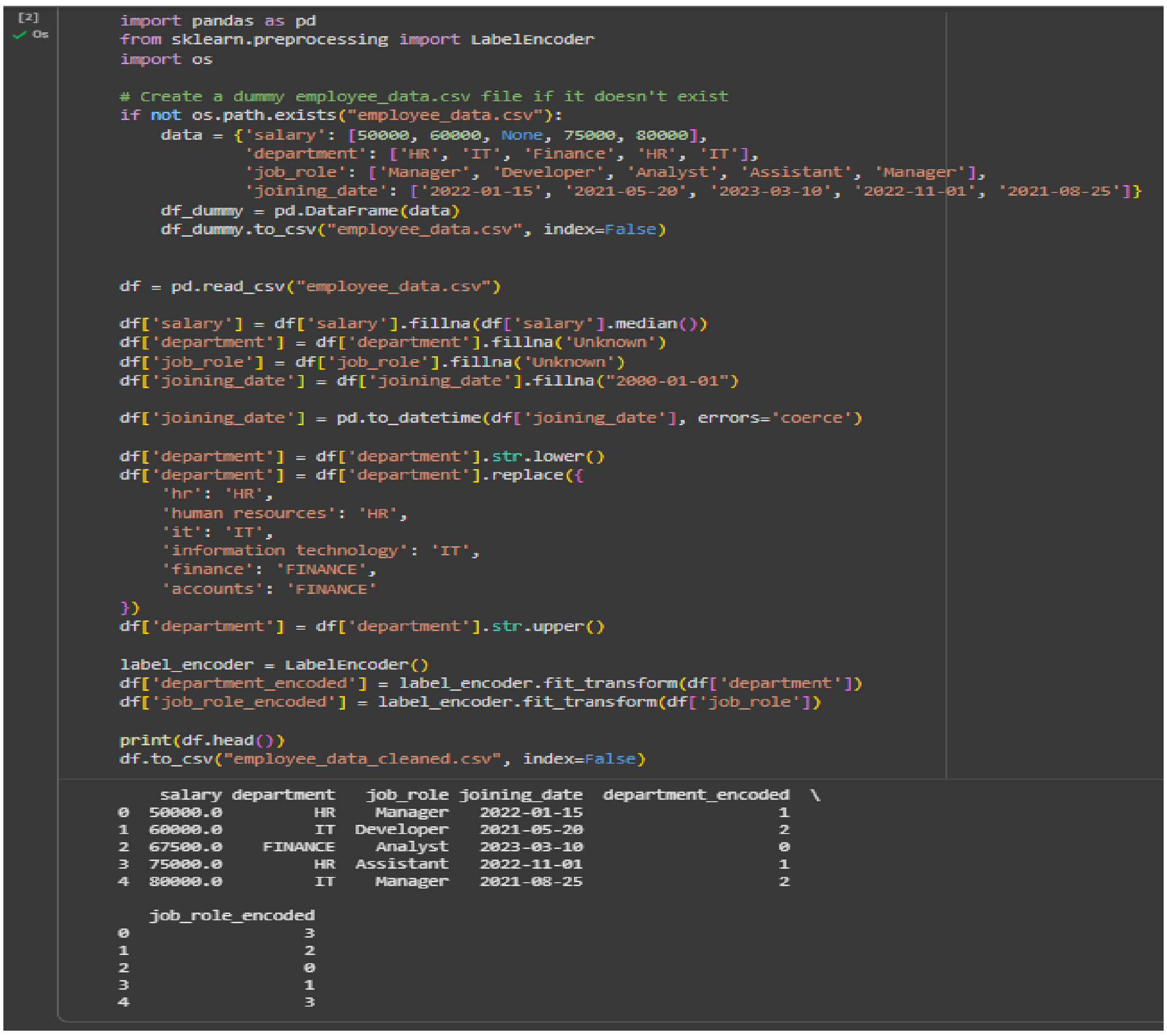
NAME:K.VARSHITHA.

Task 1 – 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).



```python
import pandas as pd
from sklearn.preprocessing import LabelEncoder
import os

# Create a dummy employee_data.csv file if it doesn't exist
if not os.path.exists("employee_data.csv"):
    data = {'salary': [50000, 60000, None, 75000, 80000],
            'department': ['HR', 'IT', 'Finance', 'HR', 'IT'],
            'job_role': ['Manager', 'Developer', 'Analyst', 'Assistant', 'Manager'],
            'joining_date': ['2022-01-15', '2021-05-20', '2023-03-10', '2022-11-01', '2021-08-25']}
    df_dummy = pd.DataFrame(data)
    df_dummy.to_csv("employee_data.csv", index=False)


df = pd.read_csv("employee_data.csv")

df['salary'] = df['salary'].fillna(df['salary'].median())
df['department'] = df['department'].fillna('Unknown')
df['job_role'] = df['job_role'].fillna('Unknown')
df['joining_date'] = df['joining_date'].fillna("2000-01-01")

df['joining_date'] = pd.to_datetime(df['joining_date'], errors='coerce')

df['department'] = df['department'].str.lower()
df['department'] = df['department'].replace({
    'hr': 'HR',
    'human resources': 'HR',
    'it': 'IT',
    'information technology': 'IT',
    'finance': 'FINANCE',
    'accounts': 'FINANCE'
})
df['department'] = df['department'].str.upper()

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

print(df.head())
df.to_csv("employee_data_cleaned.csv", index=False)
```

```
    salary department   job_role joining_date  department_encoded  \
0  50000.0         HR    Manager   2022-01-15                   1
1  60000.0         IT  Developer   2021-05-20                   2
2  67500.0    FINANCE    Analyst   2023-03-10                   0
3  75000.0         HR  Assistant   2022-11-01                   1
4  80000.0         IT    Manager   2021-08-25                   2

   job_role_encoded
0                 3
1                 2
2                 0
3                 1
4                 3
```

Task 2 – Sales Transaction Data Preprocessing

Task:

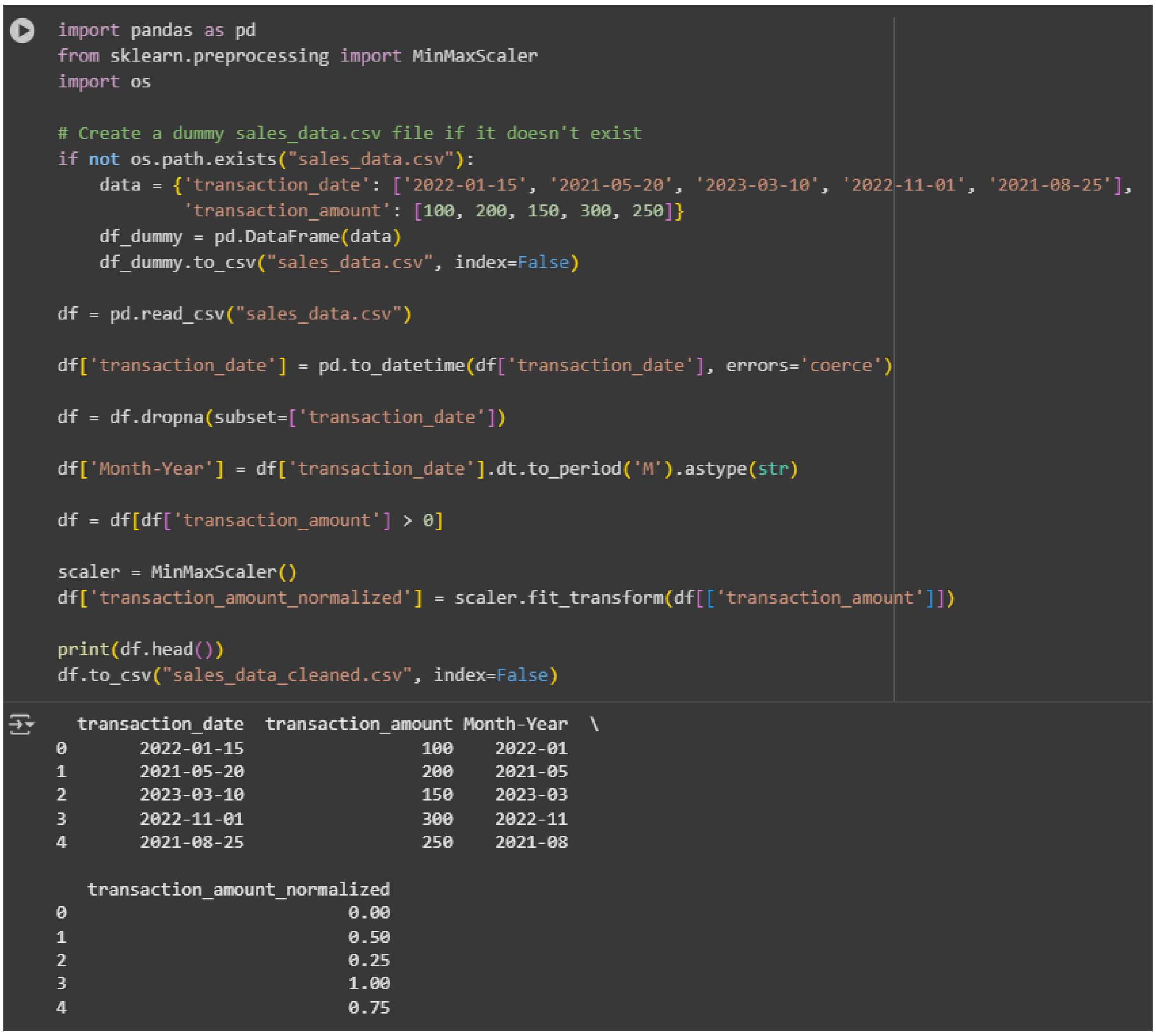
Use AI to generate a script for preprocessing a sales transaction dataset.

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.

Expected Output:

A preprocessed DataFrame with valid dates, normalized amounts, and no invalid records



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

# Create a dummy sales_data.csv file if it doesn't exist
if not os.path.exists("sales_data.csv"):
    data = {'transaction_date': ['2022-01-15', '2021-05-20', '2023-03-10', '2022-11-01', '2021-08-25'],
            'transaction_amount': [100, 200, 150, 300, 250]}
    df_dummy = pd.DataFrame(data)
    df_dummy.to_csv("sales_data.csv", index=False)

df = pd.read_csv("sales_data.csv")

df['transaction_date'] = pd.to_datetime(df['transaction_date'], errors='coerce')

df = df.dropna(subset=['transaction_date'])

df['Month-Year'] = df['transaction_date'].dt.to_period('M').astype(str)

df = df[df['transaction_amount'] > 0]

scaler = MinMaxScaler()
df['transaction_amount_normalized'] = scaler.fit_transform(df[['transaction_amount']])

print(df.head())
df.to_csv("sales_data_cleaned.csv", index=False)
```

```
  transaction_date  transaction_amount Month-Year  \
0       2022-01-15                 100    2022-01
1       2021-05-20                 200    2021-05
2       2023-03-10                 150    2023-03
3       2022-11-01                 300    2022-11
4       2021-08-25                 250    2021-08

   transaction_amount_normalized
0                           0.00
1                           0.50
2                           0.25
3                           1.00
4                           0.75
```

Task 3 – Healthcare Patient Records Cleaning

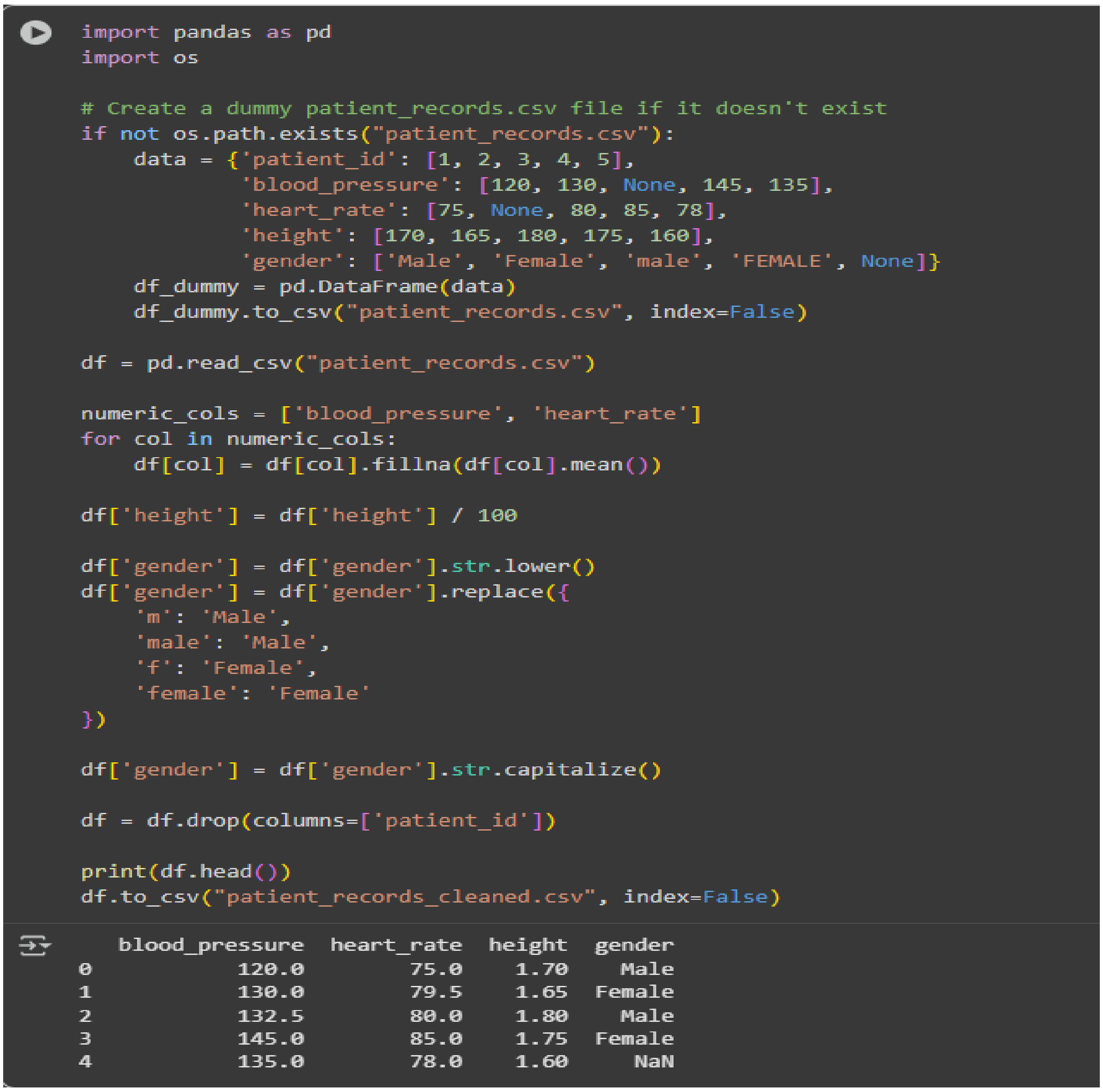
Task:
Use AI to generate a script for cleaning healthcare patient records**.**

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.

Expected Output:

- A cleaned healthcare dataset suitable for ML model training**.**



```python
import pandas as pd
import os

# Create a dummy patient_records.csv file if it doesn't exist
if not os.path.exists("patient_records.csv"):
    data = {'patient_id': [1, 2, 3, 4, 5],
            'blood_pressure': [120, 130, None, 145, 135],
            'heart_rate': [75, None, 80, 85, 78],
            'height': [170, 165, 180, 175, 160],
            'gender': ['Male', 'Female', 'male', 'FEMALE', None]}
    df_dummy = pd.DataFrame(data)
    df_dummy.to_csv("patient_records.csv", index=False)

df = pd.read_csv("patient_records.csv")

numeric_cols = ['blood_pressure', 'heart_rate']
for col in numeric_cols:
    df[col] = df[col].fillna(df[col].mean())

df['height'] = df['height'] / 100

df['gender'] = df['gender'].str.lower()
df['gender'] = df['gender'].replace({
    'm': 'Male',
    'male': 'Male',
    'f': 'Female',
    'female': 'Female'
})

df['gender'] = df['gender'].str.capitalize()

df = df.drop(columns=['patient_id'])

print(df.head())
df.to_csv("patient_records_cleaned.csv", index=False)
```

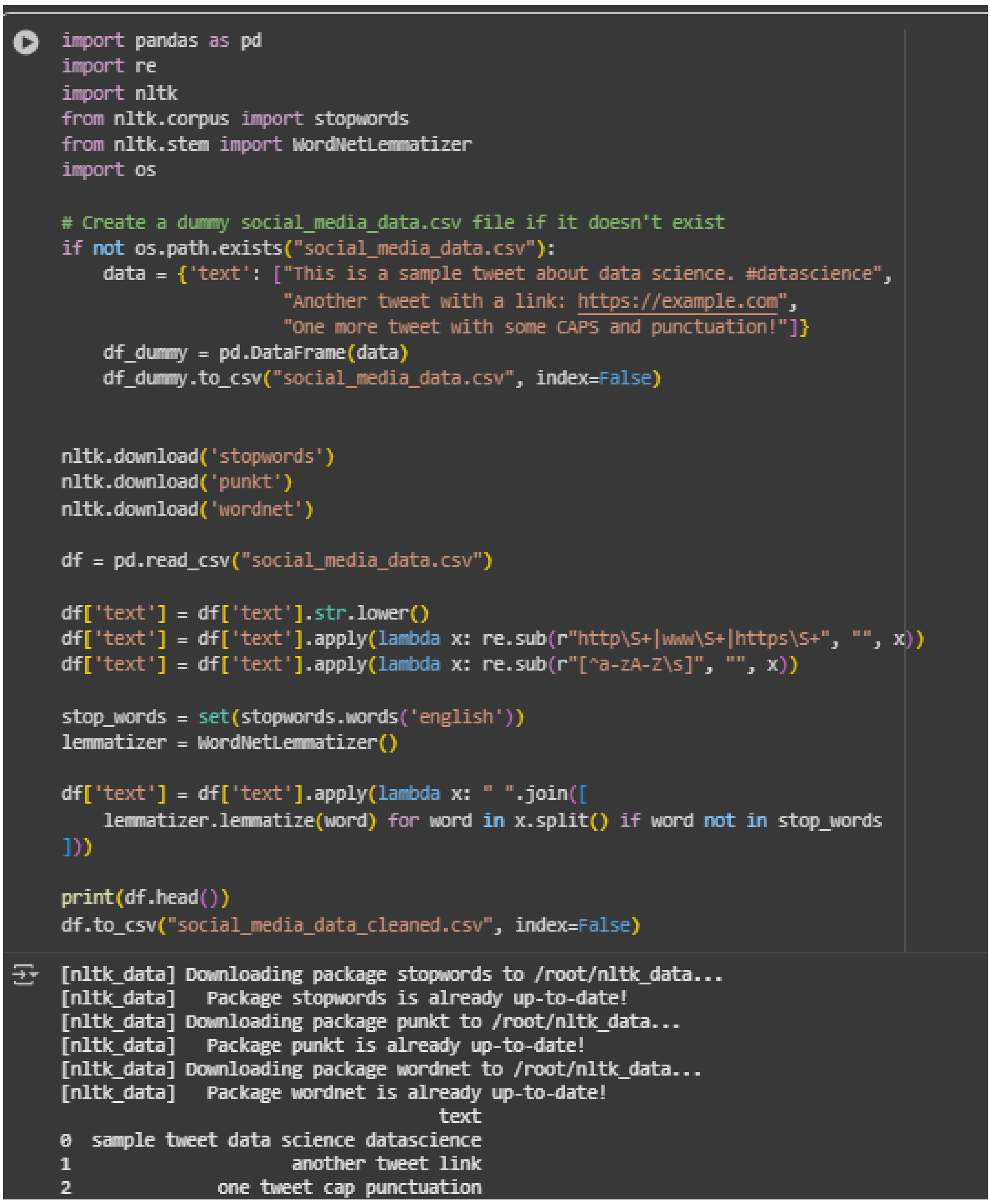
```
   blood_pressure  heart_rate  height  gender
0           120.0        75.0    1.70    Male
1           130.0        79.5    1.65  Female
2           132.5        80.0    1.80    Male
3           145.0        85.0    1.75  Female
4           135.0        78.0    1.60     NaN
```

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.



```python
import pandas as pd
import re
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
import os

# Create a dummy social_media_data.csv file if it doesn't exist
if not os.path.exists("social_media_data.csv"):
    data = {'text': ["This is a sample tweet about data science. #datascience",
                     "Another tweet with a link: https://example.com",
                     "One more tweet with some CAPS and punctuation!"]}
    df_dummy = pd.DataFrame(data)
    df_dummy.to_csv("social_media_data.csv", index=False)


nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')

df = pd.read_csv("social_media_data.csv")

df['text'] = df['text'].str.lower()
df['text'] = df['text'].apply(lambda x: re.sub(r"http\S+|www\S+|https\S+", "", x))
df['text'] = df['text'].apply(lambda x: re.sub(r"[^a-zA-Z\s]", "", x))

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

df['text'] = df['text'].apply(lambda x: " ".join([
    lemmatizer.lemmatize(word) for word in x.split() if word not in stop_words
]))

print(df.head())
df.to_csv("social_media_data_cleaned.csv", index=False)
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
                                   text
0  sample tweet data science datascience
1                   another tweet link
2             one tweet cap punctuation
```

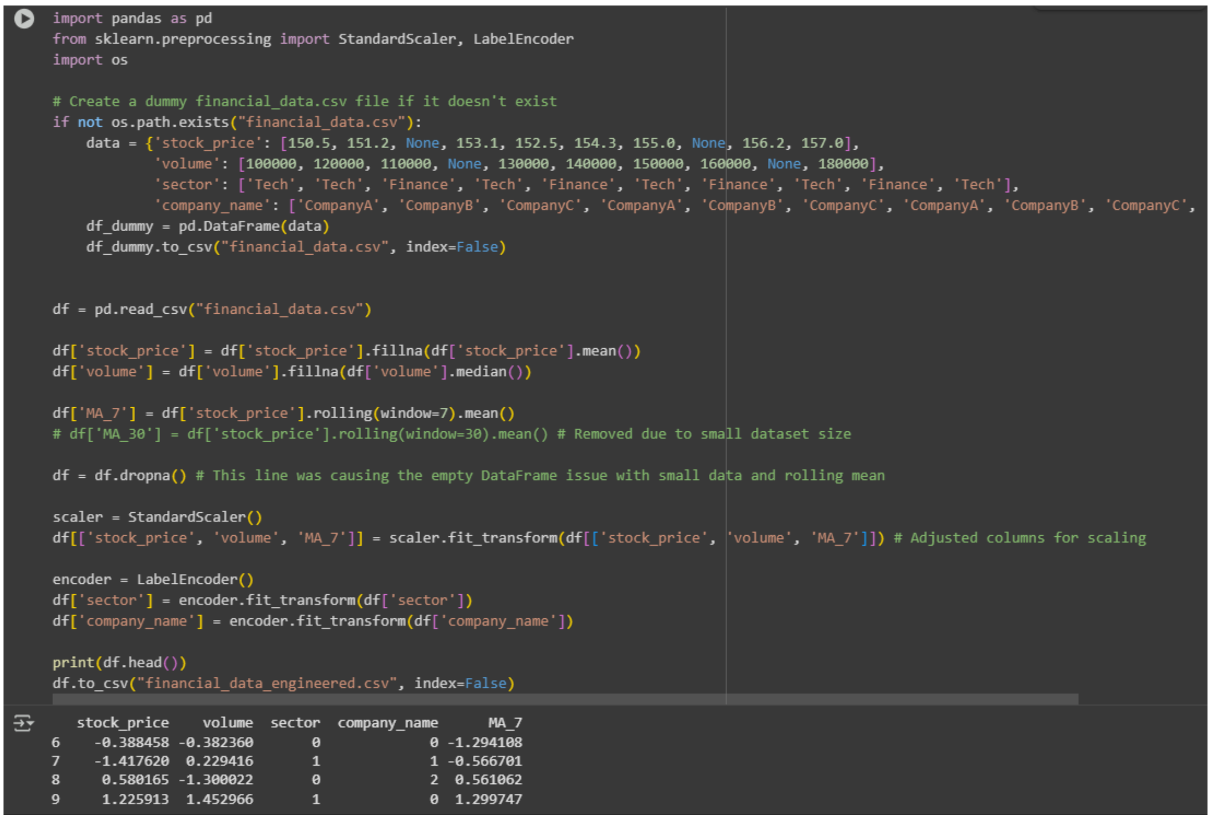
Task 5 – Financial Dataset Feature Engineering

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).



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

# Create a dummy financial_data.csv file if it doesn't exist
if not os.path.exists("financial_data.csv"):
    data = {'stock_price': [150.5, 151.2, None, 153.1, 152.5, 154.3, 155.0, None, 156.2, 157.0],
            'volume': [100000, 120000, 110000, None, 130000, 140000, 150000, 160000, None, 180000],
            'sector': ['Tech', 'Tech', 'Finance', 'Tech', 'Finance', 'Tech', 'Finance', 'Tech', 'Finance', 'Tech'],
            'company_name': ['CompanyA', 'CompanyB', 'CompanyC', 'CompanyA', 'CompanyB', 'CompanyC', 'CompanyA', 'CompanyB', 'CompanyC',
    df_dummy = pd.DataFrame(data)
    df_dummy.to_csv("financial_data.csv", index=False)


df = pd.read_csv("financial_data.csv")

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

df['MA_7'] = df['stock_price'].rolling(window=7).mean()
# df['MA_30'] = df['stock_price'].rolling(window=30).mean() # Removed due to small dataset size

df = df.dropna() # This line was causing the empty DataFrame issue with small data and rolling mean

scaler = StandardScaler()
df[['stock_price', 'volume', 'MA_7']] = scaler.fit_transform(df[['stock_price', 'volume', 'MA_7']]) # Adjusted columns for scaling

encoder = LabelEncoder()
df['sector'] = encoder.fit_transform(df['sector'])
df['company_name'] = encoder.fit_transform(df['company_name'])

print(df.head())
df.to_csv("financial_data_engineered.csv", index=False)
```

```
   stock_price    volume  sector  company_name      MA_7
6    -0.388458 -0.382360       0             0 -1.294108
7    -1.417620  0.229416       1             1 -0.566701
8     0.580165 -1.300022       0             2  0.561062
9     1.225913  1.452966       1             0  1.299747
```