| SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE | | DEPARTMENT OF COMPUTER SCIENCE ENGINEERING | |
|---|---|---|---|
| **Program Name:** B. Tech | | **Assignment Type: Lab** | **Academic Year:**2025-2026 |
| **Course Coordinator Name** | Venkataramana Veeramsetty | | |
| **Instructor(s) Name** | Dr. V. Venkataramana (Co-ordinator)<br>Dr. T. Sampath Kumar<br>Dr. Pramoda Patro<br>Dr. Brij Kishor Tiwari<br>Dr.J.Ravichander<br>Dr. Mohammand Ali Shaik<br>Dr. Anirodh Kumar<br>Mr. S.Naresh Kumar<br>Dr. RAJESH VELPULA<br>Mr. Kundhan Kumar<br>Ms. Ch.Rajitha<br>Mr. M Prakash<br>Mr. B.Raju<br>Intern 1 (Dharma teja)<br>Intern 2 (Sai Prasad)<br>Intern 3 (Sowmya)<br>NS_2 ( Mounika) | | |
| **Course Code** | 24CS002PC215 | **Course Title** | AI Assisted Coding |
| **Year/Sem** | II/I | **Regulation** | R24 |
| **Date and Day of Assignment** | Week9 - Thursday | **Time(s)** | |
| **Duration** | 2 Hours | **Applicable to Batches** | |
| **AssignmentNumber:17.4**(Present assignment number)/**24**(Total number of assignments) | | | |

| Q.No. | Question | *Expected Time to complete* |
|---|---|---|
| 1 | **Lab 17 – AI for Data Processing: Data Cleaning and Preprocessing Scripts**<br>**Lab Objectives:**<br>• Learn how to clean raw datasets using AI-assisted Python scripting. | Week9 - Thursday |

- Apply preprocessing techniques such as handling missing values, encoding categorical data, and normalization.
- Automate repetitive data-cleaning tasks with AI-generated code.
- Understand how preprocessing impacts model performance.

**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).

**Expected Output:**

- A cleaned Pandas DataFrame with consistent departments, proper dates, and encoded features.

CODE:

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

# Create a sample employee dataset
data = {
    'employee_id': [101, 102, 103, 104, 105, 106, 107],
    'name': ['Anita', 'Rahul', 'Suresh', 'Divya', 'Neha', 'Ravi', 'Meena'],
    'salary': [55000, 62000, None, 58000, None, 60000, 75000],
    'department': ['HR', 'human resources', 'hr', 'Sales & Marketing', 'Finance', None, 'Human Resources'],
    'job_role': ['Manager', 'Executive', 'HR Officer', 'Sales Rep', 'Accountant', 'Clerk', 'Recruiter'],
    'joining_date': ['2020-01-15', None, '2019-03-10', '2021-07-22', '2020-12-05', '2018-09-14', None]
}


df = pd.DataFrame(data)
print("Sample Employee Dataset (Raw):\n")
print(df)
df.to_csv('employee_data.csv', index=False)
print("\n Dataset saved as 'employee_data.csv'")
```

OUTPUT:

```
Sample Employee Dataset (Raw):

   employee_id    name   salary         department    job_role joining_date
0          101   Anita  55000.0                 HR     Manager   2020-01-15
1          102   Rahul  62000.0    human resources   Executive         None
2          103  Suresh      NaN                 hr  HR Officer   2019-03-10
3          104   Divya  58000.0  Sales & Marketing   Sales Rep   2021-07-22
4          105    Neha      NaN            Finance  Accountant   2020-12-05
5          106    Ravi  60000.0               None       Clerk   2018-09-14
6          107   Meena  75000.0    Human Resources   Recruiter         None

 Dataset saved as 'employee_data.csv'
```

**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.

CODE:

```python
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
data = {
    'transaction_id': [1,2,3,4,5,6,7,8,9,10],
    'customer_id': [101,102,103,104,105,106,107,108,109,110],
    'transaction_date': ['2025-10-01','2025-10-03','2025-10-05','2025-11-01',
                         '2025-11-02','2025-11-03','invalid_date','2025-11-05',
                         '2025-11-06','2025-11-07'],
    'transaction_amount': [250,0,-50,450,300,500,200,0,150,600],
    'product': ['Notebook','Pen','Marker','Printer','Paper','Laptop','Mouse','Keyboard','Stapler','Desk']
}
df = pd.DataFrame(data)
print("Original Dataset:")
print(df)
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')
df = df[df['transaction_amount'] > 0]
scaler = MinMaxScaler()
df['transaction_amount_normalized'] = scaler.fit_transform(df[['transaction_amount']])
df.reset_index(drop=True, inplace=True)
print("\nCleaned Dataset:")
print(df)
```

OUTPUT:

```
PS C:\Users\DEEKSHA\Desktop\AIAC\Lab-17> & C:/Users/DEEKSHA/AppData/Local/Microsoft/
Original Dataset:
   transaction_id  customer_id transaction_date  transaction_amount   product
0               1          101       2025-10-01                 250  Notebook
1               2          102       2025-10-03                   0       Pen
2               3          103       2025-10-05                 -50    Marker
3               4          104       2025-11-01                 450   Printer
4               5          105       2025-11-02                 300     Paper
5               6          106       2025-11-03                 500    Laptop
6               7          107     invalid_date                 200     Mouse
7               8          108       2025-11-05                   0  Keyboard
8               9          109       2025-11-06                 150   Stapler
9              10          110       2025-11-07                 600      Desk

Cleaned Dataset:
   transaction_id  customer_id transaction_date  transaction_amount   product month_year  transaction_amount_normalized
0               1          101       2025-10-01                 250  Notebook    2025-10                       0.222222
1               4          104       2025-11-01                 450   Printer    2025-11                       0.666667
2               5          105       2025-11-02                 300     Paper    2025-11                       0.333333
3               6          106       2025-11-03                 500    Laptop    2025-11                       0.777778
4               9          109       2025-11-06                 150   Stapler    2025-11                       0.000000
5              10          110       2025-11-07                 600      Desk    2025-11                       1.000000
PS C:\Users\DEEKSHA\Desktop\AIAC\Lab-17>
```

**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**.**

CODE**:**

```
import pandas as pd
import numpy as np
data = {
    'patient_id': [1,2,3,4,5],
    'name': ['Alice','Bob','Charlie','David','Eva'],
    'gender': ['F', 'M', 'male', 'Female', 'M'],
    'age': [25, 30, np.nan, 45, 50],
    'height_cm': [160, 175, 180, np.nan, 165],
    'weight_kg': [55, 70, 80, 90, np.nan],
    'blood_pressure': [120, 130, np.nan, 140, 135],
    'heart_rate': [80, np.nan, 78, 85, 90]
}
df = pd.DataFrame(data)
print("Original Dataset:")
print(df)
numeric_cols = ['age', 'height_cm', 'weight_kg', 'blood_pressure', 'heart_rate']
for col in numeric_cols:
    df[col].fillna(df[col].mean(), inplace=True)
df['height_m'] = df['height_cm'] / 100
df.drop(columns=['height_cm'], inplace=True)  # drop old column
df['gender'] = df['gender'].replace({'M':'Male', 'male':'Male', 'F':'Female', 'Female':'Female'})
df.drop(columns=['patient_id'], inplace=True)
print("\nCleaned Healthcare Dataset:")
print(df)
df.to_csv('healthcare_data_cleaned.csv', index=False)
```

**OUTPUT:**

```
Cleaned Healthcare Dataset:
      name  gender   age  weight_kg  blood_pressure  heart_rate  height_m
0    Alice  Female  25.0      55.00          120.00       80.00      1.60
1      Bob    Male  30.0      70.00          130.00       83.25      1.75
2  Charlie    Male  37.5      80.00          131.25       78.00      1.80
3    David  Female  45.0      90.00          140.00       85.00      1.70
4      Eva    Male  50.0      73.75          135.00       90.00      1.65
PS C:\Users\DEEKSHA\Desktop\AIAC\Lab-17>
```

**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.

**Expected Output:**

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

CODE:

Task4.py > ...

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

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

data = {
    'post_id': [1,2,3,4,5],
    'text': [
        "I love this product!  Check it out: https://example.com",
        "Worst service ever!! #fail ",
        "Not bad, could be better... ",
        "Totally amazing experience!!! @brand",
        "Meh... it's okay. http://test.com "
    ]
}
df = pd.DataFrame(data)
print("Original Dataset:")
print(df)

def clean_text(text):
    # Remove URLs
```

```
    text = re.sub(r'http\S+|www.\S+', '', text)
    # Remove mentions and hashtags
    text = re.sub(r'@\w+|#\w+', '', text)
    # Remove emojis and special characters
    text = re.sub(r'[^\w\s]', '', text)
    # Convert to lowercase
    text = text.lower()
    return text

df['clean_text'] = df['text'].apply(clean_text)

stop_words = set(stopwords.words('english'))
df['tokens'] = df['clean_text'].apply(lambda x: [word for word in x.split() if word not in stop_words])

lemmatizer = WordNetLemmatizer()
df['lemmatized'] = df['tokens'].apply(lambda x: [lemmatizer.lemmatize(word) for word in x])

df['processed_text'] = df['lemmatized'].apply(lambda x: ' '.join(x))

print("\nProcessed Dataset:")
print(df[['post_id','processed_text']])

df.to_csv('social_media_cleaned.csv', index=False)
```

OUTPUT:

```
Original Dataset:
   post_id                                               text
0        1  I love this product!  Check it out: https://ex...
1        2                        Worst service ever!! #fail
[nltk_data]     C:\Users\DEEKSHA\AppData\Roaming\nltk_data...
Original Dataset:
   post_id                                               text
0        1  I love this product!  Check it out: https://ex...
1        2                        Worst service ever!! #fail
   post_id                                               text
0        1  I love this product!  Check it out: https://ex...
1        2                        Worst service ever!! #fail
0        1  I love this product!  Check it out: https://ex...
1        2                        Worst service ever!! #fail
1        2                        Worst service ever!! #fail
2        3                       Not bad, could be better...
2        3                       Not bad, could be better...
3        4                Totally amazing experience!!! @brand
3        4                Totally amazing experience!!! @brand
4        5                  Meh... it's okay. http://test.com
4        5                  Meh... it's okay. http://test.com
```

```
Processed Dataset:
   post_id             processed_text
0        1          love product check
1        2          worst service ever
2        3            bad could better
3        4  totally amazing experience
4        5                    meh okay
PS C:\Users\DEEKSHA\Desktop\AIAC\Lab-17>
```

**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).

**Expected Output:**

- A feature-engineered DataFrame with new indicators and normalized values for ML tasks.

CODE:

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

# Example: Load your financial dataset
#df = pd.read_csv('financial_data.csv')

# For demonstration, create a mock financial dataset
data = {
    'date': pd.date_range("2023-01-01", periods=60),
    'company_name': ['AlphaTech']*20 + ['BetaBank']*20 + ['GammaEnergy']*20,
    'sector': ['Technology']*20 + ['Finance']*20 + ['Energy']*20,
    'stock_price': np.random.normal(100, 15, 60),
    'volume': np.random.normal(10000, 2000, 60)
}
df = pd.DataFrame(data)

# Randomly set some stock_price and volume as missing
np.random.seed(42)
missing_idx = np.random.choice(df.index, size=8, replace=False)
df.loc[missing_idx[:4], 'stock_price'] = np.nan
df.loc[missing_idx[4:], 'volume'] = np.nan

# 1. Handle missing values in stock_price and volume
df['stock_price'] = df['stock_price'].fillna(df['stock_price'].mean())
df['volume'] = df['volume'].fillna(df['volume'].median())
```

```
# 2. Create new features: moving averages (7-day, 30-day) by company
df.sort_values(['company_name', 'date'], inplace=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 using StandardScaler
continuous_columns = ['stock_price', 'volume', 'ma_7', 'ma_30']
scaler = StandardScaler()
df[continuous_columns] = scaler.fit_transform(df[continuous_columns])

# 4. Encode categorical columns
categorical_cols = ['sector', 'company_name']
encoder = OneHotEncoder(sparse_output=False, drop='first')
encoded = encoder.fit_transform(df[categorical_cols])
encoded_cols = encoder.get_feature_names_out(categorical_cols)
df_encoded = pd.DataFrame(encoded, columns=encoded_cols, index=df.index)

# Merge encoded columns and drop originals
df = df.drop(columns=categorical_cols).join(df_encoded)

# Final feature-engineered DataFrame ready for ML tasks
print(df.head())

# Optionally save to new CSV
df.to_csv('financial_data_feature_engineered.csv', index=False)
```

OUTPUT:

```
PS C:\Users\DEEKSHA\Desktop\AIAC\Lab-17> & C:/Users/DEEKSHA/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/DEEKSHA/Desktop/AIA
        date  stock_price    volume      ma_7     ma_30  sector_Finance  sector_Technology  company_name_BetaBank  company_name_GammaEnergy
0 2023-01-01     0.000000 -1.388626 -0.023359  0.073102             0.0                1.0                    0.0                       0.0
1 2023-01-02     1.141799 -1.870012  1.138647  1.642710             0.0                1.0                    0.0                       0.0
2 2023-01-03    -0.107236  1.585444  0.678556  1.021231             0.0                1.0                    0.0                       0.0
3 2023-01-04     0.076179 -0.697816  0.541841  0.836560             0.0                1.0                    0.0                       0.0
4 2023-01-05    -1.296451 -0.532675 -0.098957 -0.029014             0.0                1.0                    0.0                       0.0
```

✅ Deliverables (For All Tasks)

1. AI-generated prompts for code and test case generation.
2. At least 3 assert test cases for each task.
3. AI-generated initial code and execution screenshots.
4. Analysis of whether code passes all tests.
5. Improved final version with inline comments and explanation.
6. Compiled report (Word/PDF) with prompts, test cases, assertions, code, and output.