Họ tên: Trần Vân Anh

MSV: B20DCCN075

1. Sử dung pandas để chuyển các file excel sang csv

```
# Use pandas to convert Excel files to CSV

import os
import pandas as pd

# Path to the directory containing Excel files
excel_folder = 'DATA_DIEMTHI/'

# Path to the directory containing output CSV files
csv_folder = 'DIEMTHI_CSV/'

# Iterate through all files in the Excel folder
for filename in os.listdir(excel_folder):
    if filename.endswith('.xlsx'):  # Check if the file is in Excel format
        # Read the Excel file
        excel_file_path = os.path.join(excel_folder, filename)
        df = pd.read_excel(excel_file_path)

        # Create the output CSV file name
        csv_filename = filename.replace('.xlsx', '.csv')
        csv_file_path = os.path.join(csv_folder, csv_filename)

        # Save the data to the CSV file
        df.to_csv(csv_file_path, index=False)
    else:
        csv_filename = filename
        csv_file_path = os.path.join(csv_folder, csv_filename)

        # Save the data to the CSV file
        df.to_csv(csv_file_path, index=False)

print("Conversion completed.")
```

```
Conversion completed.
```

- ***Folder DIEMTHI_CSV***

2. Gộp các file csv thành 1 file data_pttk.csv

```
# Merge CSV files into one data_pttk.csv file

import os
import pandas as pd

# Path to the directory containing CSV files
csv_folder = 'DIEMTHI_CSV/'

# List of CSV files in the directory
csv_files = [f for f in os.listdir(csv_folder) if f.endswith('.csv')]

# Initialize an empty DataFrame to hold the combined data
combined_data = pd.DataFrame()

# Iterate through the list of CSV files and read them into the DataFrame
for csv_file in csv_files:
    csv_file_path = os.path.join(csv_folder, csv_file)
    df = pd.read_csv(csv_file_path)
    combined_data = pd.concat([combined_data, df], ignore_index=True)

# Path and name of the output CSV file
output_csv_path = 'data_pttk.csv'

# Write the combined data to the output CSV file
combined_data.to_csv(output_csv_path, index=False)

print("Successfully merged CSV files and saved to", output_csv_path)
```

```
Successfully merged CSV files and saved to data_pttk.csv
```

- ***File data_pttk.csv***

```
mã sv,0.1,0.1.1,Unnamed: 3,0.2,Unnamed: 5,điểm thi,Unnamed: 7
b1,9,7,,6.0,,6.0,sáu điểm
b2,9,7,,6.0,,6.0,sáu điểm
b3,10,9,,7.0,,7.25,bảy hai lăm
b4,9,7,,6.0,,6.0,sáu điểm
b5,10,8,,7.0,,7.5,bảy rưỡi
b6,9,7,,6.5,,6.5,sáu rưỡi
b7,0,0,,0.0,,,
b8,10,8,,6.0,,7.25,bảy hai lăm
b9,10,8,,8.0,,8.0,tám điểm
b10,10,8,,7.5,,8.75,tám bảy lăm
b11,9,7,,6.0,,5.5,năm rưỡi
b12,9,7,,6.0,,6.0,sáu điểm
b13,10,8,,7.0,,7.5,bảy rưỡi
b14,0,0,,0.0,,,
b15,10,8,,7.0,,7.0,bảy điểm
b16,10,8,,6.0,,7.5,bảy rưỡi
b17,10,9,,7.0,,7.25,bảy hai lăm
b18,6,5,,5.0,,5.0,năm diểm
b19,9,7,,6.0,,6.0,sáu điểm
b20,10,9,,7.0,,6.5,sáu rưỡi
b21,9,7,,6.0,,6.0,sáu điểm
b22,9,7,,6.0,,3.5,ba rưỡi
b23,10,8,,7.0,,8.0,tám điểm
b24,9,6,,6.0,,6.5,sáu rưỡi
b25,10,7,,6.0,,6.0,sáu điểm
b26,10,7,,6.0,,7.0,bảy điểm
b27,10,6,,6.0,,6.0,sáu điểm
b28,10,7,,6.0,,7.0,bảy điểm
b29,10,8,,6.5,,7.5,bảy rưỡi
b30,9,8,,6.0,,5.5,năm rưỡi
b30,9,7,,6.0,,6.5,sáu rưỡi
b31,10,7,,6.0,,6.0,sáu điểm
b32,9,7,,6.0,,5.5,năm rưỡi
b34,9,7,,6.0,,4.5,bốn rưỡi
```

3. Thực hiện các bước tiền xử lý data như Chap 12

```
# Perform data preprocessing steps as in Chapter 12

import pandas as pd

# Read data from the merged CSV file
data = pd.read_csv('data_pttk.csv')

# Remove unnecessary columns
data = data.drop(columns=['Unnamed: 3', 'Unnamed: 5'])

# Fill missing values
data['0.1.1'].fillna(0, inplace=True)  # Fill missing values with 0 in the '0.1.1' column

# Convert text data to numerical (if necessary)
data['0.1'] = data['0.1'].astype(float)  # Convert '0.1' column to float type
data['0.2'] = data['0.2'].astype(float)  # Convert '0.2' column to float type

# Save the processed data to a new CSV file (if needed)
data.to_csv('data_pttk_processed.csv', index=False)
```

- ***File data_pttk_processed.csv***

```
mã sv,0.1,0.1.1,0.2,điểm thi,Unnamed: 7
b1,9.0,7,6.0,6.0,sáu điểm
b2,9.0,7,6.0,6.0,sáu điểm
b3,10.0,9,7.0,7.25,bảy hai lăm
b4,9.0,7,6.0,6.0,sáu điểm
b5,10.0,8,7.0,7.5,bảy rưỡi
b6,9.0,7,6.5,6.5,sáu rưỡi
b7,0.0,0,0.0,,
b8,10.0,8,6.0,7.25,bảy hai lăm
b9,10.0,8,8.0,8.0,tám điểm
b10,10.0,8,7.5,8.75,tám bảy lăm
b11,9.0,7,6.0,5.5,năm rưỡi
b12,9.0,7,6.0,6.0,sáu điểm
b13,10.0,8,7.0,7.5,bảy rưỡi
b14,0.0,0,0.0,,
b15,10.0,8,7.0,7.0,bảy điểm
b16,10.0,8,6.0,7.5,bảy rưỡi
b17,10.0,9,7.0,7.25,bảy hai lăm
b18,6.0,5,5.0,5.0,năm diểm
b19,9.0,7,6.0,6.0,sáu điểm
b20,10.0,9,7.0,6.5,sáu rưỡi
b21,9.0,7,6.0,6.0,sáu điểm
b22,9.0,7,6.0,3.5,ba rưỡi
b23,10.0,8,7.0,8.0,tám điểm
b24,9.0,6,6.0,6.5,sáu rưỡi
b25,10.0,7,6.0,6.0,sáu điểm
b26,10.0,7,6.0,7.0,bảy điểm
b27,10.0,6,6.0,6.0,sáu điểm
b28,10.0,7,6.0,7.0,bảy điểm
b29,10.0,8,6.5,7.5,bảy rưỡi
b30,9.0,8,6.0,5.5,năm rưỡi
b30,9.0,7,6.0,6.5,sáu rưỡi
b31,10.0,7,6.0,6.0,sáu điểm
b32,9.0,7,6.0,5.5,năm rưỡi
b34,9.0,7,6.0,4.5,bốn rưỡi
```

4. Sử dụng các kỹ thuật ML cơ bản (Chap 12) để dự đoán điểm thi khi nhập các điểm thành phần

- ***Support Vector Machines***

```python
# Support Vector Machines

import pandas as pd
from sklearn.svm import SVR

# Read preprocessed data
data = pd.read_csv('data_pttk_processed.csv')
data.dropna(subset=['điểm thi'], inplace=True)

# Select features (component scores) and target (exam score)
X = data[['0.1', '0.2', '0.1.1']]
y = data['điểm thi']

# Initialize and train the SVM model
model = SVR(kernel='linear', C=1.0)
model.fit(X, y)

while True:
    try:
        # Input component scores from the keyboard
        score1 = float(input("Enter component score 1: "))
        score2 = float(input("Enter component score 2: "))
        score3 = float(input("Enter component score 3: "))

        # Predict the exam score
        predicted_exam_score = model.predict([[score1, score2, score3]])

        print(f"Predicted exam score: {predicted_exam_score[0]}")
    except ValueError:
        print("Please enter a valid number.")

    continue_prediction = input("Continue prediction (enter 'q' to quit, any other key to continue): ")
    if continue_prediction.lower() == 'q':
        break
```

```
Enter component score 1: 9
Enter component score 2: 8
Enter component score 3: 7

Predicted exam score: 7.090380621675384
Continue prediction (enter 'q' to quit, any other key to continue): q
```

- ***KNN***

```python
# KNN

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsRegressor

# Read data from the preprocessed CSV file
data = pd.read_csv('data_pttk_processed.csv')
data.dropna(subset=['điểm thi'], inplace=True)

# Split the data into features (X) and target (y)
X = data[['0.1', '0.1.1', '0.2']]
y = data['điểm thi']

# Normalize the data
scaler = StandardScaler()
X = scaler.fit_transform(X)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize and train the KNN model
model = KNeighborsRegressor(n_neighbors=5)
model.fit(X_train, y_train)

while True:
    try:
        # Input component scores from the keyboard
        score1 = float(input("Enter component score 1: "))
        score2 = float(input("Enter component score 2: "))
        score3 = float(input("Enter component score 3: "))

        # Predict the exam score
        predicted_exam_score = model.predict([[score1, score2, score3]])

        print(f"Predicted exam score: {predicted_exam_score[0]}")
    except ValueError:
        print("Please enter a valid number.")

    continue_prediction = input("Continue prediction (enter 'q' to quit, any other key to continue): ")
    if continue_prediction.lower() == 'q':
        break
```

```
Enter component score 1: 9
Enter component score 2: 8
Enter component score 3: 7
Predicted exam score: 7.3
Continue prediction (enter 'q' to quit, any other key to continue): q
```

5. Sử dụng Deep learning với Linear regression để dự đoán điểm thi

```python
# Use Deep Learning with Linear Regression to predict exam scores

import pandas as pd
from sklearn.linear_model import LinearRegression

# Read preprocessed data
data = pd.read_csv('data_pttk_processed.csv')

# Remove rows with NaN values in the data
data = data.dropna()

# Prepare training data
X = data[['0.1', '0.2', '0.1.1']]
y = data['điểm thi']

# Initialize and train the linear regression model
model = LinearRegression()
model.fit(X, y)
```

```
▾ LinearRegression
LinearRegression()
```

```
▸ LinearRegression
```

```python
while True:
    try:
        # Nhập điểm thành phần từ người dùng
        diem1 = float(input("Nhập điểm thành phần 1: "))
        diem2 = float(input("Nhập điểm thành phần 2: "))
        diem3 = float(input("Nhập điểm thành phần 3: "))

        # Dự đoán điểm thi
        diem_thi_du_doan = model.predict([[diem1, diem2, diem3]])

        print(f"Điểm thi dự đoán: {diem_thi_du_doan[0]}")
    except ValueError:
        print("Vui lòng nhập số hợp lệ.")

    tiep_tuc = input("Tiếp tục dự đoán (nhập 'q' để thoát, bất kỳ phím nào để tiếp tục): ")
    if tiep_tuc.lower() == 'q':
        break
```

- ***Input***

```
Enter component score 1: 9
Enter component score 2: 8

Enter component score 3: 7
```

- ***Output***

```
Enter component score 1: 9
Enter component score 2: 8
Enter component score 3: 7

Predicted exam score: 7.460648656563437
Continue prediction (enter 'q' to quit, any other key to continue): q
```

6. Đánh giá kết quả

```python
# Evaluate the results
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVR
from sklearn.neural_network import MLPRegressor
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_absolute_error, mean_squared_error

# Load the preprocessed data
data = pd.read_csv('data_pttk_processed.csv')
data.dropna(subset=['điểm thi'], inplace=True)
X = data[['0.1', '0.2', '0.1.1']]
y = data['điểm thi']

# Normalize the data for KNN
scaler = StandardScaler()
X_normalized = scaler.fit_transform(X)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_normalized, y, test_size=0.2, random_state=42)

# Initialize and train models
linear_regression_model = LinearRegression()
knn_model = KNeighborsRegressor(n_neighbors=5)
svm_model = SVR(kernel='linear', C=1.0)
deep_learning_model = MLPRegressor(hidden_layer_sizes=(100, 50), activation='relu', solver='adam', random_state=42)

linear_regression_model.fit(X_train, y_train)
knn_model.fit(X_train, y_train)
svm_model.fit(X_train, y_train)
deep_learning_model.fit(X_train, y_train)
```

```python
# Predictions
linear_regression_predictions = linear_regression_model.predict(X_test)
knn_predictions = knn_model.predict(X_test)
svm_predictions = svm_model.predict(X_test)
deep_learning_predictions = deep_learning_model.predict(X_test)

# Evaluate models
print("Mean Absolute Error:")
print("Linear Regression:", mean_absolute_error(y_test, linear_regression_predictions))
print("KNN:", mean_absolute_error(y_test, knn_predictions))
print("SVM:", mean_absolute_error(y_test, svm_predictions))
print("Deep Learning:", mean_absolute_error(y_test, deep_learning_predictions))

print("\nMean Squared Error:")
print("Linear Regression:", mean_squared_error(y_test, linear_regression_predictions))
print("KNN:", mean_squared_error(y_test, knn_predictions))
print("SVM:", mean_squared_error(y_test, svm_predictions))
print("Deep Learning:", mean_squared_error(y_test, deep_learning_predictions))
```

```
Mean Absolute Error:
Linear Regression: 0.4937000278444146
KNN: 0.5357142857142856
SVM: 0.4986023763805584
Deep Learning: 0.60533005718507

Mean Squared Error:
Linear Regression: 0.5463303746126383
KNN: 0.601547619047619
SVM: 0.5321159466192745
Deep Learning: 0.6486968552133002
```

- MAE tính giá trị trung bình của sự sai lệch tuyệt đối giữa giá trị dự đoán và giá trị thực tế (thực tế).
- MAE cho biết mức độ sai lệch trung bình giữa dự đoán và giá trị thực tế mà mô hình đã tạo ra.
- MAE càng thấp, mô hình càng chính xác. MAE thích hợp khi bạn quan tâm đến việc đánh giá sai số tuyệt đối.
- Mean Squared Error (MSE):
- MSE tính giá trị trung bình của bình phương sai số giữa giá trị dự đoán và giá trị thực tế.
- MSE là một phép đo thường được sử dụng và nhạy cảm hơn đối với các giá trị sai lệch lớn vì nó bình phương các sai số.
- MSE càng thấp, mô hình càng chính xác. Tuy nhiên, nó có xu hướng tăng cường tác động của các giá trị sai lệch lớn.