PharmApp Suite

🧠 AI for Drug Discovery and Development 💉

🧩 Topic: So sánh ChEMBL với danh mục thuốc thiết yếu WHO

Tuyệt vời! Với vai trò là chuyên gia R&D dược học, tôi rất vui được hỗ trợ bạn trong dự án phân tích dữ liệu ChEMBL 35 này. Mã chủ đề Topic_CheMBL_35_56 cho thấy bạn đang tập trung vào một chủ đề cụ thể, điều này rất tốt để có một hướng đi rõ ràng.

1. Phân tích mô hình phân tích (Analysis of the Analysis Model):

Dựa trên thông tin bạn cung cấp, chúng ta sẽ xây dựng một quy trình phân tích dữ liệu theo hướng sau:

- **Chuẩn bị dữ liệu (Data Preparation):**
  - Sử dụng SQL để truy vấn và lọc dữ liệu từ cơ sở dữ liệu ChEMBL 35.
  - Lưu dữ liệu đã lọc vào các file CSV.
  - Sử dụng RDKit để tính toán các descriptor phân tử từ SMILES strings.
- **Phân tích khám phá (Exploratory Data Analysis - EDA):**
  - Phân tích thống kê mô tả (descriptive statistics) để hiểu phân phối dữ liệu.
  - Trực quan hóa dữ liệu (data visualization) để tìm kiếm các xu hướng và mối quan hệ tiềm năng.
- **Xây dựng mô hình (Model Building):**
  - Tùy thuộc vào mục tiêu của Topic_CheMBL_35_56, chúng ta có thể xây dựng các mô hình khác nhau, ví dụ:
    - **Mô hình hồi quy (Regression models):** Dự đoán hoạt tính của hợp chất dựa trên các descriptor phân tử.
    - **Mô hình phân loại (Classification models):** Phân loại hợp chất thành chất ức chế hoặc không ức chế dựa trên các descriptor phân tử.
- **Đánh giá mô hình (Model Evaluation):**
  - Sử dụng các chỉ số phù hợp (ví dụ: R-squared, RMSE cho hồi quy; Accuracy, Precision, Recall, F1-score cho phân loại) để đánh giá hiệu suất của mô hình.

2. Hướng dẫn song ngữ (Bilingual Guidance):

Dưới đây là hướng dẫn chi tiết bằng cả tiếng Anh và tiếng Việt:

English:

1. **Data Extraction using SQL:** We'll use SQL queries to extract relevant data from the ChEMBL 35 database, focusing on the specific criteria defined by Topic_CheMBL_35_56. We'll limit the initial extraction to 100 rows for faster processing and to avoid overloading the system.
2. **Data Loading and Preprocessing (Python):** The extracted data (CSV files) will be loaded into a Pandas DataFrame. We'll handle missing values, convert data types, and prepare the data for RDKit processing.
3. **Molecular Descriptor Calculation (RDKit):** RDKit will be used to calculate various molecular descriptors from the SMILES strings. These descriptors will serve as features for our machine learning models.
4. **Model Building and Evaluation (Python):** We'll use scikit-learn to build and evaluate machine learning models (regression or classification) based on the calculated descriptors.
5. **Addressing the Error:** The SQL error "ERROR: operator does not exist: numeric ~ unknown" indicates an issue with regular expression matching on numeric data. We'll modify the SQL

query to handle numeric values correctly (see example below). For the scikit-learn error, ensure you have a compatible version or remove `squared=False` if necessary.

Tiếng Việt:

1. **Trích xuất dữ liệu bằng SQL:** Chúng ta sẽ sử dụng các truy vấn SQL để trích xuất dữ liệu liên quan từ cơ sở dữ liệu ChEMBL 35, tập trung vào các tiêu chí cụ thể được xác định bởi Topic_CheMBL_35_56. Chúng ta sẽ giới hạn việc trích xuất ban đầu ở 100 dòng để xử lý nhanh hơn và tránh làm quá tải hệ thống.
2. **Tải và tiền xử lý dữ liệu (Python):** Dữ liệu đã trích xuất (các file CSV) sẽ được tải vào Pandas DataFrame. Chúng ta sẽ xử lý các giá trị bị thiếu, chuyển đổi kiểu dữ liệu và chuẩn bị dữ liệu cho quá trình xử lý RDKit.
3. **Tính toán descriptor phân tử (RDKit):** RDKit sẽ được sử dụng để tính toán các descriptor phân tử khác nhau từ chuỗi SMILES. Các descriptor này sẽ đóng vai trò là các đặc trưng cho các mô hình học máy của chúng ta.
4. **Xây dựng và đánh giá mô hình (Python):** Chúng ta sẽ sử dụng scikit-learn để xây dựng và đánh giá các mô hình học máy (hồi quy hoặc phân loại) dựa trên các descriptor đã tính toán.
5. **Giải quyết lỗi:** Lỗi SQL "ERROR: operator does not exist: numeric ~ unknown" cho thấy có vấn đề với việc khớp biểu thức chính quy trên dữ liệu số. Chúng ta sẽ sửa đổi truy vấn SQL để xử lý các giá trị số một cách chính xác (xem ví dụ bên dưới). Đối với lỗi scikit-learn, hãy đảm bảo bạn có phiên bản tương thích hoặc loại bỏ `squared=False` nếu cần.

3. Ví dụ code SQL và Python (SQL and Python Code Examples):

SQL Examples:

```sql
-- Example 1: Extracting data for a specific target (replace 'CHEMBL205' with a relevant target ID)
-- Ví dụ 1: Trích xuất dữ liệu cho một mục tiêu cụ thể (thay thế 'CHEMBL205' bằng ID mục tiêu thích hợp)
SELECT act.molregno, act.standard_value, act.standard_units, act.standard_type, md.smiles
FROM activities act
JOIN molecule_dictionary md ON act.molregno = md.molregno
WHERE act.tid = (SELECT tid FROM target_dictionary WHERE chembl_id = 'CHEMBL205')
AND act.standard_relation = '='
AND act.standard_value IS NOT NULL
AND act.standard_units = 'nM'
AND act.standard_type = 'IC50'
AND act.standard_value::TEXT ~ '^[0-9\.]+$' -- Corrected line: explicit cast to TEXT for regex
LIMIT 100;

-- Example 2: Extracting data based on activity values within a range
-- Ví dụ 2: Trích xuất dữ liệu dựa trên các giá trị hoạt động trong một phạm vi
SELECT act.molregno, act.standard_value, act.standard_units, act.standard_type, md.smiles
FROM activities act
JOIN molecule_dictionary md ON act.molregno = md.molregno
WHERE act.standard_relation = '='
AND act.standard_value BETWEEN 100 AND 1000 -- Example range
AND act.standard_units = 'nM'
AND act.standard_type = 'IC50'
LIMIT 100;

-- Example 3:  Extracting data and filtering by molecule properties (e.g., molecular weight) - requires joining with compound_structures
-- Ví dụ 3: Trích xuất dữ liệu và lọc theo thuộc tính phân tử (ví dụ: trọng lượng phân tử) - yêu cầu kết hợp với compound_structures
```

```sql
-- NOTE: This example assumes there's a way to link activities to compound_structures (e.g., through molregno or another ID).  Adapt the join condition as needed.
-- LƯU Ý: Ví dụ này giả định có một cách để liên kết các hoạt động với compound_structures (ví dụ: thông qua molregno hoặc một ID khác). Điều chỉnh điều kiện kết hợp cho phù hợp.
SELECT act.molregno, act.standard_value, act.standard_units, act.standard_type, md.smiles --, cs.mol_weight  (Assuming mol_weight is in compound_structures)
FROM activities act
JOIN molecule_dictionary md ON act.molregno = md.molregno
--LEFT JOIN compound_structures cs ON act.molregno = cs.molregno  -- Adapt the join
WHERE act.standard_relation = '='
AND act.standard_value IS NOT NULL
AND act.standard_units = 'nM'
AND act.standard_type = 'IC50'
--AND cs.mol_weight < 500  -- Example molecular weight filter
LIMIT 100;

-- Example 4: Extracting data for a specific assay (replace 'CHEMBL194' with a relevant assay ID)
-- Ví dụ 4: Trích xuất dữ liệu cho một assay cụ thể (thay thế 'CHEMBL194' bằng ID assay thích hợp)
SELECT act.molregno, act.standard_value, act.standard_units, act.standard_type, md.smiles
FROM activities act
JOIN molecule_dictionary md ON act.molregno = md.molregno
WHERE act.assay_id = (SELECT assay_id FROM assays WHERE chembl_id = 'CHEMBL194')
AND act.standard_relation = '='
AND act.standard_value IS NOT NULL
AND act.standard_units = 'nM'
AND act.standard_type = 'IC50'
LIMIT 100;

-- Example 5: Extracting data with pChEMBL values (if available)
-- Ví dụ 5: Trích xuất dữ liệu với các giá trị pChEMBL (nếu có)
SELECT act.molregno, act.standard_value, act.standard_units, act.standard_type, act.pchembl_value, md.smiles
FROM activities act
JOIN molecule_dictionary md ON act.molregno = md.molregno
WHERE act.pchembl_value IS NOT NULL
AND act.standard_relation = '='
AND act.standard_value IS NOT NULL
AND act.standard_units = 'nM'
AND act.standard_type = 'IC50'
LIMIT 100;
```

Python Examples:

```python
import os
import pandas as pd
from rdkit import Chem
from rdkit.Chem import Descriptors
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
#from sklearn.metrics import mean_squared_error, r2_score # Import necessary modules
import numpy as np  # Import numpy

base_path = "../data"  # Adjust the base path as needed

# Example 1: Load data from CSV, calculate molecular weight, and build a simple
```

```
regression model
# Ví dụ 1: Tải dữ liệu từ CSV, tính trọng lượng phân tử và xây dựng một mô hình hồi
quy đơn giản
def calculate_molecular_weight(smiles):
    """Calculates the molecular weight of a molecule given its SMILES string."""
    """Tính trọng lượng phân tử của một phân tử dựa trên chuỗi SMILES của nó."""
    mol = Chem.MolFromSmiles(smiles)
    if mol:
        return Descriptors.MolWt(mol)
    else:
        return None

def example_regression_model(csv_filename="chembl_data.csv",
target_column="standard_value", smiles_column="smiles"):
    """Loads data, calculates molecular weight, builds a linear regression model, and
evaluates it."""
    """Tải dữ liệu, tính trọng lượng phân tử, xây dựng mô hình hồi quy tuyến tính và
đánh giá nó."""
    try:
        # Load data
        file_path = os.path.join(base_path, csv_filename)
        df = pd.read_csv(file_path)
        df = df.dropna(subset=[smiles_column, target_column])  # Drop rows with
missing SMILES or target values

        # Calculate molecular weight
        df['mol_weight'] = df[smiles_column].apply(calculate_molecular_weight)
        df = df.dropna(subset=['mol_weight']) # Drop rows where molecular weight
calculation failed

        # Prepare data for modeling
        X = df[['mol_weight']]
        y = df[target_column].astype(float) # Ensure target variable is numeric

        # Split 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)

        # Build and train the linear regression model
        model = LinearRegression()
        model.fit(X_train, y_train)

        # Make predictions on the test set
        y_pred = model.predict(X_test)

        # Evaluate the model
        mse = mean_squared_error(y_test, y_pred)
        r2 = r2_score(y_test, y_pred)
        print(f"Mean Squared Error: {mse}")
        print(f"R-squared: {r2}")
        return model, mse, r2

    except FileNotFoundError:
        print(f"Error: File not found at {file_path}")
        return None, None, None
    except Exception as e:
        print(f"An error occurred: {e}")
        return None, None, None

#Example 2: Load data and calculate QED
```

```
#Ví dụ 2: Tải dữ liệu và tính toán QED

def calculate_QED(smiles):
    """Calculates the QED (Quantitative Estimate of Drug-likeness) of a molecule."""
    """Tính toán QED (Định lượng ước tính về độ giống thuốc) của một phân tử."""
    mol = Chem.MolFromSmiles(smiles)
    if mol:
        return Descriptors.qed(mol)
    else:
        return None
def example_calculate_qed(csv_filename="chembl_data.csv", smiles_column="smiles"):
     """Loads data, calculates QED ,and evaluates it."""
     """Tải dữ liệu, tính toán QED."""
     try:
        # Load data
        file_path = os.path.join(base_path, csv_filename)
        df = pd.read_csv(file_path)
        df = df.dropna(subset=[smiles_column])  # Drop rows with missing SMILES
        # Calculate QED
        df['QED'] = df[smiles_column].apply(calculate_QED)
        print(df[['smiles', 'QED']].head()) # Print the first few rows with SMILES and
QED
     except FileNotFoundError:
        print(f"Error: File not found at {file_path}")
     except Exception as e:
        print(f"An error occurred: {e}")

#Example 3: Load data and filter by Lipinski's Rule of Five
#Ví dụ 3: Tải dữ liệu và lọc theo quy tắc 5 của Lipinski

def lipinski_filter(smiles):
    """Applies Lipinski's Rule of Five to filter drug-like molecules."""
    """Áp dụng quy tắc 5 của Lipinski để lọc các phân tử giống thuốc."""
    mol = Chem.MolFromSmiles(smiles)
    if mol:
        mw = Descriptors.MolWt(mol)
        logp = Chem.Crippen.MolLogP(mol)
        hbd = Chem.Lipinski.NumHDonors(mol)
        hba = Chem.Lipinski.NumHAcceptors(mol)
        return mw <= 500 and logp <= 5 and hbd <= 5 and hba <= 10
    else:
        return False

def example_lipinski_filter(csv_filename="chembl_data.csv", smiles_column="smiles"):
    """Loads data, applies Lipinski's filter, and prints the filtered data."""
    """Tải dữ liệu, áp dụng bộ lọc Lipinski và in dữ liệu đã lọc."""
    try:
        # Load data
        file_path = os.path.join(base_path, csv_filename)
        df = pd.read_csv(file_path)
        df = df.dropna(subset=[smiles_column])  # Drop rows with missing SMILES

        # Apply Lipinski's filter
        df['lipinski_compliant'] = df[smiles_column].apply(lipinski_filter)
        lipinski_df = df[df['lipinski_compliant']]

        print(f"Number of molecules before filtering: {len(df)}")
        print(f"Number of molecules after filtering: {len(lipinski_df)}")
        print(lipinski_df[[smiles_column, 'lipinski_compliant']].head())
    except FileNotFoundError:
```

```
        print(f"Error: File not found at {file_path}")
    except Exception as e:
        print(f"An error occurred: {e}")

#Example 4: Calculate and visualize molecular weight distribution
#Ví dụ 4: Tính toán và trực quan hóa phân phối trọng lượng phân tử

import matplotlib.pyplot as plt
import seaborn as sns

def example_visualize_mw(csv_filename="chembl_data.csv", smiles_column="smiles"):
    """Loads data, calculates molecular weight, and visualizes the distribution."""
    """Tải dữ liệu, tính trọng lượng phân tử và trực quan hóa phân phối."""
    try:
        # Load data
        file_path = os.path.join(base_path, csv_filename)
        df = pd.read_csv(file_path)
        df = df.dropna(subset=[smiles_column])  # Drop rows with missing SMILES

        # Calculate molecular weight
        df['mol_weight'] = df[smiles_column].apply(calculate_molecular_weight)
        df = df.dropna(subset=['mol_weight'])

        # Visualize the distribution
        sns.histplot(df['mol_weight'], kde=True)
        plt.title('Distribution of Molecular Weight')
        plt.xlabel('Molecular Weight (Daltons)')
        plt.ylabel('Frequency')
        plt.show()
    except FileNotFoundError:
        print(f"Error: File not found at {file_path}")
    except Exception as e:
        print(f"An error occurred: {e}")

#Example 5: Building a basic classification model with Morgan fingerprints
#Ví dụ 5: Xây dựng mô hình phân loại cơ bản với dấu vân tay Morgan
from rdkit.Chem import AllChem
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report

def calculate_morgan_fingerprint(smiles, radius=2, nBits=2048):
    """Calculates the Morgan fingerprint (ECFP) for a molecule."""
    """Tính toán dấu vân tay Morgan (ECFP) cho một phân tử."""
    mol = Chem.MolFromSmiles(smiles)
    if mol:
        return AllChem.GetMorganFingerprintAsBitVect(mol, radius, nBits=nBits)
    else:
        return None

def example_classification_model(csv_filename="chembl_data.csv",
smiles_column="smiles", target_column="active"): #active is 0 or 1
    """Loads data, calculates Morgan fingerprints, builds a random forest
classification model, and evaluates it."""
    """Tải dữ liệu, tính toán dấu vân tay Morgan, xây dựng mô hình phân loại rừng ngẫu
nhiên và đánh giá nó."""
    try:
        # Load data
        file_path = os.path.join(base_path, csv_filename)
        df = pd.read_csv(file_path)
        df = df.dropna(subset=[smiles_column, target_column])
```

```
        # Calculate Morgan fingerprints
        df['fingerprint'] = df[smiles_column].apply(calculate_morgan_fingerprint)
        df = df.dropna(subset=['fingerprint'])

        # Prepare data for modeling
        X = list(df['fingerprint'])  # Convert fingerprints to a list
        y = df[target_column].astype(int)  # Ensure target variable is integer

        # Split 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)

        # Build and train the random forest classification model
        model = RandomForestClassifier(random_state=42)
        model.fit(X_train, X_train)

        # Make predictions on the test set
        y_pred = model.predict(X_test)

        # Evaluate the model
        accuracy = accuracy_score(y_test, y_pred)
        report = classification_report(y_test, y_pred)

        print(f"Accuracy: {accuracy}")
        print(f"Classification Report:\n{report}")
        return model, accuracy, report
    except FileNotFoundError:
        print(f"Error: File not found at {file_path}")
        return None, None, None
    except Exception as e:
        print(f"An error occurred: {e}")
        return None, None, None
# To run the examples, uncomment the relevant lines:
# Ví dụ chạy các ví dụ, hãy bỏ comment các dòng liên quan:

#example_regression_model()
#example_calculate_qed()
#example_lipinski_filter()
#example_visualize_mw()
#example_classification_model()
```

Quan trọng (Important Notes):

- **Đường dẫn (Paths):** Luôn sử dụng `os.path.join(base_path, filename)` để đảm bảo code của bạn hoạt động trên các hệ điều hành khác nhau.
- **Xử lý lỗi (Error Handling):** Các ví dụ trên đã bao gồm một số xử lý lỗi cơ bản (ví dụ: `try...except`), nhưng bạn nên mở rộng chúng để xử lý các tình huống khác nhau có thể xảy ra trong quá trình phân tích dữ liệu.
- **Chọn descriptor phân tử (Choosing Molecular Descriptors):** Việc chọn descriptor phân tử phù hợp là rất quan trọng để xây dựng các mô hình hiệu quả. Hãy xem xét các descriptor liên quan đến mục tiêu nghiên cứu của bạn (Topic_CheMBL_35_56).
- **Tối ưu hóa mô hình (Model Optimization):** Sau khi xây dựng mô hình ban đầu, bạn có thể thử các kỹ thuật tối ưu hóa khác nhau (ví dụ: điều chỉnh siêu tham số, lựa chọn đặc trưng) để cải thiện hiệu suất của mô hình.

Chúc bạn thành công với dự án của mình! Nếu bạn có bất kỳ câu hỏi nào khác, đừng ngần ngại hỏi.