PharmApp Suite

🧠 AI for Drug Discovery and Development 💉

🧩 Topic: Học đa nhiệm cho ức chế toàn đích

Tuyệt vời! Tôi rất vui được hỗ trợ bạn trong dự án phân tích dữ liệu ChEMBL 35 này. Với vai trò là một chuyên gia nghiên cứu và phát triển dược học, tôi hiểu rõ tầm quan trọng của việc khai thác dữ liệu ChEMBL để tìm kiếm và phát triển các hợp chất tiềm năng. Dựa trên thông tin bạn cung cấp, tôi sẽ hướng dẫn bạn từng bước, từ phân tích mô hình, viết code SQL và Python, đến xử lý các lỗi thường gặp.

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

Dự án của bạn tập trung vào việc kết hợp dữ liệu từ ChEMBL 35 với các công cụ tin sinh học như RDKit để phục vụ cho nghiên cứu và phát triển thuốc. Dưới đây là một số phân tích và hướng dẫn để bạn có thể triển khai mô hình một cách hiệu quả:

- **Mục tiêu (Objective):** Xác định các hợp chất có hoạt tính sinh học cao đối với một mục tiêu cụ thể (ví dụ: một protein liên quan đến bệnh). Bạn có thể muốn xây dựng một mô hình dự đoán hoạt tính, tìm kiếm các hợp chất tương tự, hoặc phân tích mối quan hệ cấu trúc-hoạt tính (SAR).
- **Nguồn dữ liệu (Data Source):** ChEMBL 35 là một cơ sở dữ liệu lớn chứa thông tin về các hợp chất hóa học và hoạt tính sinh học của chúng. Bạn sẽ sử dụng dữ liệu này làm nguồn chính.
- **Công cụ (Tools):**
    - **PostgreSQL (psql):** Hệ quản trị cơ sở dữ liệu để lưu trữ và truy vấn dữ liệu ChEMBL.
    - **RDKit:** Thư viện Python để phân tích và thao tác với cấu trúc hóa học.
    - **Jupyter Notebook:** Môi trường phát triển tích hợp để viết và chạy code Python.
    - **Scikit-learn:** Thư viện Python cho các thuật toán học máy.
- **Các bước chính (Main Steps):**
    1. **Truy vấn dữ liệu (Data Querying):** Sử dụng SQL để trích xuất dữ liệu từ ChEMBL, bao gồm thông tin về hợp chất (ví dụ: SMILES, InChI), hoạt tính sinh học (ví dụ: IC50, Ki), và mục tiêu.
    2. **Tiền xử lý dữ liệu (Data Preprocessing):** Làm sạch và chuẩn hóa dữ liệu, loại bỏ các giá trị ngoại lệ hoặc không hợp lệ.
    3. **Tính toán descriptor (Descriptor Calculation):** Sử dụng RDKit để tính toán các descriptor phân tử từ cấu trúc hóa học (ví dụ: trọng lượng phân tử, logP, số lượng liên kết).
    4. **Phân tích dữ liệu (Data Analysis):** Sử dụng các kỹ thuật thống kê và học máy để phân tích dữ liệu, ví dụ:
        - **Mô hình hóa QSAR/QSPR (QSAR/QSPR Modeling):** Xây dựng mô hình dự đoán hoạt tính sinh học dựa trên các descriptor phân tử.
        - **Tìm kiếm tương đồng (Similarity Search):** Tìm kiếm các hợp chất tương tự với một hợp chất mục tiêu dựa trên cấu trúc hoặc descriptor.
        - **Phân cụm (Clustering):** Phân nhóm các hợp chất dựa trên tính chất hóa học và hoạt tính sinh học.
    5. **Trực quan hóa dữ liệu (Data Visualization):** Sử dụng các công cụ như Matplotlib hoặc Seaborn để trực quan hóa dữ liệu và kết quả phân tích.

2. Hướng dẫn song ngữ (Bilingual Instructions)

Tôi sẽ cung cấp hướng dẫn bằng cả tiếng Anh và tiếng Việt để bạn dễ dàng theo dõi.

3. Code SQL và Python (SQL and Python Code)

3.1 Code SQL

- **Mục tiêu:** Trích xuất dữ liệu về các hợp chất và hoạt tính sinh học của chúng từ cơ sở dữ liệu ChEMBL.
- **Ví dụ 1:** Lấy 100 hợp chất có hoạt tính IC50 đối với một mục tiêu cụ thể (ví dụ: CHEMBL205).

```sql
-- English
-- Retrieve 100 compounds with IC50 activity against a specific target (e.g.,
CHEMBL205)
SELECT DISTINCT
    md.chembl_id,
    cs.canonical_smiles,
    act.standard_value,
    act.standard_units
FROM
    molecule_dictionary md
JOIN
    compound_structures cs ON md.molregno = cs.molregno
JOIN
    activities act ON md.molregno = act.molregno
JOIN
    target_dictionary td ON act.tid = td.tid
WHERE
    td.chembl_id = 'CHEMBL205'  -- Replace with your target of interest
    AND act.standard_type = 'IC50'
    AND act.standard_units = 'nM'
    AND act.standard_value IS NOT NULL
    AND act.standard_value > 0
ORDER BY
    act.standard_value
LIMIT 100;

-- Vietnamese
-- Lấy 100 hợp chất có hoạt tính IC50 đối với một mục tiêu cụ thể (ví dụ: CHEMBL205)
SELECT DISTINCT
    md.chembl_id,
    cs.canonical_smiles,
    act.standard_value,
    act.standard_units
FROM
    molecule_dictionary md
JOIN
    compound_structures cs ON md.molregno = cs.molregno
JOIN
    activities act ON md.molregno = act.molregno
JOIN
    target_dictionary td ON act.tid = td.tid
WHERE
    td.chembl_id = 'CHEMBL205'  -- Thay thế bằng mục tiêu bạn quan tâm
    AND act.standard_type = 'IC50'
    AND act.standard_units = 'nM'
    AND act.standard_value IS NOT NULL
    AND act.standard_value > 0
ORDER BY
    act.standard_value
LIMIT 100;
```

- **Ví dụ 2:** Lấy 100 hợp chất và thông tin cơ bản của chúng.

```sql
-- English
-- Retrieve 100 compounds and their basic information
SELECT md.chembl_id, cs.canonical_smiles
FROM molecule_dictionary md
JOIN compound_structures cs ON md.molregno = cs.molregno
LIMIT 100;

-- Vietnamese
-- Lấy 100 hợp chất và thông tin cơ bản của chúng
SELECT md.chembl_id, cs.canonical_smiles
FROM molecule_dictionary md
JOIN compound_structures cs ON md.molregno = cs.molregno
LIMIT 100;
```

- **Ví dụ 3:** Lấy 100 hợp chất có hoạt tính Ki.

```sql
-- English
-- Retrieve 100 compounds with Ki activity
SELECT DISTINCT
    md.chembl_id,
    cs.canonical_smiles,
    act.standard_value,
    act.standard_units
FROM
    molecule_dictionary md
JOIN
    compound_structures cs ON md.molregno = cs.molregno
JOIN
    activities act ON md.molregno = act.molregno
WHERE
    act.standard_type = 'Ki'
    AND act.standard_units = 'nM'
    AND act.standard_value IS NOT NULL
    AND act.standard_value > 0
ORDER BY
    act.standard_value
LIMIT 100;

-- Vietnamese
-- Lấy 100 hợp chất có hoạt tính Ki
SELECT DISTINCT
    md.chembl_id,
    cs.canonical_smiles,
    act.standard_value,
    act.standard_units
FROM
    molecule_dictionary md
JOIN
    compound_structures cs ON md.molregno = cs.molregno
JOIN
    activities act ON md.molregno = act.molregno
WHERE
    act.standard_type = 'Ki'
    AND act.standard_units = 'nM'
    AND act.standard_value IS NOT NULL
    AND act.standard_value > 0
ORDER BY
    act.standard_value
LIMIT 100;
```

- **Ví dụ 4:** Lấy các mục tiêu liên quan đến một bệnh cụ thể (ví dụ: ung thư).

```sql
-- English
-- Retrieve targets associated with a specific disease (e.g., cancer)
SELECT DISTINCT
    td.chembl_id,
    td.pref_name
FROM
    target_dictionary td
JOIN
    target_components tc ON td.tid = tc.tid
JOIN
    component_sequences cs ON tc.component_id = cs.component_id
WHERE
    td.target_type = 'SINGLE PROTEIN'
    AND td.organism = 'Homo sapiens'
    AND td.pref_name LIKE '%cancer%';

-- Vietnamese
-- Lấy các mục tiêu liên quan đến một bệnh cụ thể (ví dụ: ung thư)
SELECT DISTINCT
    td.chembl_id,
    td.pref_name
FROM
    target_dictionary td
JOIN
    target_components tc ON td.tid = tc.tid
JOIN
    component_sequences cs ON tc.component_id = cs.component_id
WHERE
    td.target_type = 'SINGLE PROTEIN'
    AND td.organism = 'Homo sapiens'
    AND td.pref_name LIKE '%ung thư%';
```

- **Ví dụ 5:** Lấy các hợp chất có hoạt tính trên một mục tiêu cụ thể và có trọng lượng phân tử nằm trong một khoảng nhất định.

```sql
-- English
-- Retrieve compounds with activity on a specific target and molecular weight within a
certain range
SELECT DISTINCT
    md.chembl_id,
    cs.canonical_smiles,
    act.standard_value,
    act.standard_units
FROM
    molecule_dictionary md
JOIN
    compound_structures cs ON md.molregno = cs.molregno
JOIN
    activities act ON md.molregno = act.molregno
JOIN
    target_dictionary td ON act.tid = td.tid
WHERE
    td.chembl_id = 'CHEMBL205'  -- Replace with your target of interest
    AND act.standard_type = 'IC50'
    AND act.standard_units = 'nM'
    AND act.standard_value IS NOT NULL
    AND act.standard_value > 0
    AND md.mw_freebase BETWEEN 200 AND 500; -- Molecular weight range

-- Vietnamese
-- Lấy các hợp chất có hoạt tính trên một mục tiêu cụ thể và có trọng lượng phân tử
```

```
nằm trong một khoảng nhất định
SELECT DISTINCT
    md.chembl_id,
    cs.canonical_smiles,
    act.standard_value,
    act.standard_units
FROM
    molecule_dictionary md
JOIN
    compound_structures cs ON md.molregno = cs.molregno
JOIN
    activities act ON md.molregno = act.molregno
JOIN
    target_dictionary td ON act.tid = td.tid
WHERE
    td.chembl_id = 'CHEMBL205'  -- Thay thế bằng mục tiêu bạn quan tâm
    AND act.standard_type = 'IC50'
    AND act.standard_units = 'nM'
    AND act.standard_value IS NOT NULL
    AND act.standard_value > 0
    AND md.mw_freebase BETWEEN 200 AND 500; -- Khoảng trọng lượng phân tử
```

Lưu ý:

- Thay thế `'CHEMBL205'` bằng ID mục tiêu thực tế bạn quan tâm.
- Bạn có thể tùy chỉnh các điều kiện `WHERE` để lọc dữ liệu theo nhu cầu của bạn.
- Chạy các truy vấn này trên pgAdmin và lưu kết quả vào file CSV.

3.2 Code Python

- **Mục tiêu:** Đọc dữ liệu từ file CSV, tính toán descriptor phân tử bằng RDKit, và thực hiện phân tích cơ bản.

```
# English
import os
import pandas as pd
from rdkit import Chem
from rdkit.Chem import Descriptors
import numpy as np

# Define the base path
base_path = ".."  # Adjust if needed

# Define the path to the CSV file
csv_file_path = os.path.join(base_path, "data", "your_data.csv")  # Replace
"your_data.csv"

# Read the CSV file into a pandas DataFrame
try:
    df = pd.read_csv(csv_file_path)
    print("Data loaded successfully!")
except FileNotFoundError:
    print(f"Error: File not found at {csv_file_path}")
    exit()

# Function to calculate molecular weight using RDKit
def calculate_molecular_weight(smiles):
    try:
        mol = Chem.MolFromSmiles(smiles)
        if mol:
            return Descriptors.MolWt(mol)
```

```
        else:
            return np.nan  # Return NaN if the SMILES is invalid
    except:
        return np.nan

# Apply the function to create a new column 'molecular_weight'
df['molecular_weight'] = df['canonical_smiles'].apply(calculate_molecular_weight)

# Display the first few rows with the new column
print(df.head())

# Basic analysis: Calculate the mean molecular weight
mean_molecular_weight = df['molecular_weight'].mean()
print(f"Mean Molecular Weight: {mean_molecular_weight}")


# Vietnamese
import os
import pandas as pd
from rdkit import Chem
from rdkit.Chem import Descriptors
import numpy as np

# Định nghĩa đường dẫn gốc
base_path = ".."  # Điều chỉnh nếu cần

# Định nghĩa đường dẫn đến file CSV
csv_file_path = os.path.join(base_path, "data", "your_data.csv")  # Thay thế
"your_data.csv"

# Đọc file CSV vào một DataFrame của pandas
try:
    df = pd.read_csv(csv_file_path)
    print("Dữ liệu đã được tải thành công!")
except FileNotFoundError:
    print(f"Lỗi: Không tìm thấy file tại {csv_file_path}")
    exit()

# Hàm tính toán trọng lượng phân tử sử dụng RDKit
def calculate_molecular_weight(smiles):
    try:
        mol = Chem.MolFromSmiles(smiles)
        if mol:
            return Descriptors.MolWt(mol)
        else:
            return np.nan  # Trả về NaN nếu SMILES không hợp lệ
    except:
        return np.nan

# Áp dụng hàm để tạo một cột mới 'molecular_weight'
df['molecular_weight'] = df['canonical_smiles'].apply(calculate_molecular_weight)

# Hiển thị một vài hàng đầu tiên với cột mới
print(df.head())

# Phân tích cơ bản: Tính trọng lượng phân tử trung bình
mean_molecular_weight = df['molecular_weight'].mean()
print(f"Trọng lượng phân tử trung bình: {mean_molecular_weight}")
```

Lưu ý:

- Thay thế "your_data.csv" bằng tên file CSV thực tế của bạn.
- Đảm bảo rằng bạn đã cài đặt RDKit (`pip install rdkit`).
- Đoạn code trên chỉ là một ví dụ đơn giản. Bạn có thể mở rộng nó để tính toán các descriptor khác, thực hiện phân tích thống kê, hoặc xây dựng mô hình học máy.

Ví dụ 1: Tính LogP

```
# English
from rdkit.Chem import Crippen

def calculate_logp(smiles):
    try:
        mol = Chem.MolFromSmiles(smiles)
        if mol:
            return Crippen.MolLogP(mol)
        else:
            return np.nan
    except:
        return np.nan

df['logp'] = df['canonical_smiles'].apply(calculate_logp)
print(df.head())

# Vietnamese
from rdkit.Chem import Crippen

def tinh_logp(smiles):
    try:
        mol = Chem.MolFromSmiles(smiles)
        if mol:
            return Crippen.MolLogP(mol)
        else:
            return np.nan
    except:
        return np.nan

df['logp'] = df['canonical_smiles'].apply(tinh_logp)
print(df.head())
```

Ví dụ 2: Tính số lượng vòng

```
# English
from rdkit.Chem import Rings

def calculate_num_rings(smiles):
    try:
        mol = Chem.MolFromSmiles(smiles)
        if mol:
            return Rings.NumRings(mol)
        else:
            return np.nan
    except:
        return np.nan

df['num_rings'] = df['canonical_smiles'].apply(calculate_num_rings)
print(df.head())

# Vietnamese
```

```
from rdkit.Chem import Rings

def tinh_so_vong(smiles):
    try:
        mol = Chem.MolFromSmiles(smiles)
        if mol:
            return Rings.NumRings(mol)
        else:
            return np.nan
    except:
        return np.nan

df['num_rings'] = df['canonical_smiles'].apply(tinh_so_vong)
print(df.head())
```

Ví dụ 3: Vẽ cấu trúc phân tử

```
# English
from rdkit.Chem.Draw import MolToImage

def draw_molecule(smiles):
    try:
        mol = Chem.MolFromSmiles(smiles)
        if mol:
            return MolToImage(mol)
        else:
            return None
    except:
        return None

# Draw the first molecule
image = draw_molecule(df['canonical_smiles'][0])
if image:
    image.show()

# Vietnamese
from rdkit.Chem.Draw import MolToImage

def ve_phan_tu(smiles):
    try:
        mol = Chem.MolFromSmiles(smiles)
        if mol:
            return MolToImage(mol)
        else:
            return None
    except:
        return None

# Vẽ phân tử đầu tiên
image = ve_phan_tu(df['canonical_smiles'][0])
if image:
    image.show()
```

Ví dụ 4: Lọc các hợp chất tuân theo quy tắc Lipinski

```
# English
from rdkit.Chem import Lipinski

def lipinski_rules(smiles):
    try:
        mol = Chem.MolFromSmiles(smiles)
```

```
        if mol:
            mw = Descriptors.MolWt(mol)
            logp = Crippen.MolLogP(mol)
            hbd = Lipinski.NumHDonors(mol)
            hba = Lipinski.NumHAcceptors(mol)
            if mw <= 500 and logp <= 5 and hbd <= 5 and hba <= 10:
                return True
            else:
                return False
        else:
            return False
    except:
        return False

df['lipinski'] = df['canonical_smiles'].apply(lipinski_rules)
print(df[df['lipinski'] == True].head())

# Vietnamese
from rdkit.Chem import Lipinski

def quy_tac_lipinski(smiles):
    try:
        mol = Chem.MolFromSmiles(smiles)
        if mol:
            mw = Descriptors.MolWt(mol)
            logp = Crippen.MolLogP(mol)
            hbd = Lipinski.NumHDonors(mol)
            hba = Lipinski.NumHAcceptors(mol)
            if mw <= 500 and logp <= 5 and hbd <= 5 and hba <= 10:
                return True
            else:
                return False
        else:
            return False
    except:
        return False

df['lipinski'] = df['canonical_smiles'].apply(quy_tac_lipinski)
print(df[df['lipinski'] == True].head())
```

Ví dụ 5: Tính TPSA (Diện tích bề mặt cực)

```
# English
from rdkit.Chem import TPSA

def calculate_tpsa(smiles):
    try:
        mol = Chem.MolFromSmiles(smiles)
        if mol:
            return TPSA.TPSA(mol)
        else:
            return np.nan
    except:
        return np.nan

df['tpsa'] = df['canonical_smiles'].apply(calculate_tpsa)
print(df.head())

# Vietnamese
from rdkit.Chem import TPSA
```

```python
def tinh_tpsa(smiles):
    try:
        mol = Chem.MolFromSmiles(smiles)
        if mol:
            return TPSA.TPSA(mol)
        else:
            return np.nan
    except:
        return np.nan

df['tpsa'] = df['canonical_smiles'].apply(tinh_tpsa)
print(df.head())
```

4. Xử lý lỗi (Error Handling)

- **Lỗi a: `ERROR: operator does not exist: numeric ~ unknown`**

  Lỗi này xảy ra khi bạn sử dụng toán tử ~ (regex match) trên một cột kiểu số (numeric). Để khắc phục, bạn cần đảm bảo rằng cột `act.standard_value` là kiểu text trước khi sử dụng regex.

```sql
-- English
-- Corrected SQL query
SELECT DISTINCT
    md.chembl_id,
    cs.canonical_smiles,
    act.standard_value,
    act.standard_units
FROM
    molecule_dictionary md
JOIN
    compound_structures cs ON md.molregno = cs.molregno
JOIN
    activities act ON md.molregno = act.molregno
WHERE
    act.standard_type = 'IC50'
    AND act.standard_units = 'nM'
    AND act.standard_value IS NOT NULL
    AND act.standard_value::text ~ '^[0-9\.]+$'  -- Cast to text before regex
match
LIMIT 100;

-- Vietnamese
-- Truy vấn SQL đã sửa
SELECT DISTINCT
    md.chembl_id,
    cs.canonical_smiles,
    act.standard_value,
    act.standard_units
FROM
    molecule_dictionary md
JOIN
    compound_structures cs ON md.molregno = cs.molregno
JOIN
    activities act ON md.molregno = act.molregno
WHERE
    act.standard_type = 'IC50'
    AND act.standard_units = 'nM'
    AND act.standard_value IS NOT NULL
    AND act.standard_value::text ~ '^[0-9\.]+$'  -- Chuyển đổi sang text trước
```

```
khi so khớp regex
LIMIT 100;
```

Ở đây, `act.standard_value::text` chuyển đổi cột `act.standard_value` sang kiểu text trước khi thực hiện so khớp regex.

- **Lỗi b: Phiên bản scikit-learn cũ không hỗ trợ `squared=False`**

  Nếu bạn đang sử dụng phiên bản scikit-learn cũ, hãy bỏ tham số `squared=False` hoặc nâng cấp lên phiên bản mới hơn.

```python
# English
# For older versions of scikit-learn
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(y_true, y_pred)
rmse = mse**0.5  # Calculate RMSE manually

# Vietnamese
# Đối với các phiên bản scikit-learn cũ
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(y_true, y_pred)
rmse = mse**0.5  # Tính RMSE thủ công
```

5. Cấu trúc thư mục (Folder Structure)

Bạn đã đề cập đến cấu trúc thư mục theo chuẩn AIMLOps Template. Hãy đảm bảo rằng bạn tuân thủ cấu trúc này để dễ dàng quản lý và tái sử dụng code.

Lời khuyên (Advice)

- **Bắt đầu từ những bước nhỏ:** Đừng cố gắng làm mọi thứ cùng một lúc. Bắt đầu với việc trích xuất dữ liệu cơ bản, tính toán một vài descriptor, và thực hiện một phân tích đơn giản.
- **Tài liệu hóa code:** Viết comment rõ ràng để giải thích code của bạn. Điều này sẽ giúp bạn và người khác hiểu và bảo trì code dễ dàng hơn.
- **Sử dụng hệ thống quản lý phiên bản:** Sử dụng Git để quản lý code của bạn.
- **Tham khảo tài liệu:** Đọc kỹ tài liệu của ChEMBL, RDKit, và scikit-learn để hiểu rõ hơn về các công cụ này.

Tôi hy vọng hướng dẫn này sẽ giúp bạn bắt đầu dự án của mình một cách suôn sẻ. Nếu bạn có bất kỳ câu hỏi nào, đừng ngần ngại hỏi. Chúc bạn thành công!