PharmApp Suite

🧠 AI for Drug Discovery and Development 💉

🧩 Topic: Liên kết ChEMBL với dữ liệu thử nghiệm lâm sàng

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 R&D dược học, tôi sẽ tập trung vào việc xây dựng một quy trình phân tích rõ ràng, hiệu quả và có giá trị khoa học.

Tổng quan dự án:

Dự án của bạn nhằm mục đích tận dụng dữ liệu từ cơ sở dữ liệu ChEMBL 35, kết hợp với các công cụ tin sinh học như RDKit, để hỗ trợ quá trình nghiên cứu và phát triển thuốc. Việc sử dụng PostgreSQL (psql) để quản lý dữ liệu và Jupyter Notebook để phân tích là một lựa chọn hợp lý.

Phân tích và Hướng dẫn (Song ngữ):

1. Phân tích Mô hình (Analysis Model):

Mục tiêu chính của chúng ta là khai thác thông tin có giá trị từ dữ liệu ChEMBL 35 để:

- **Xác định các hợp chất có hoạt tính sinh học tiềm năng:** Tìm kiếm các hợp chất có hoạt tính cao đối với các mục tiêu (targets) cụ thể.
- **Xây dựng mô hình dự đoán hoạt tính (SAR/QSAR):** Phát triển các mô hình có thể dự đoán hoạt tính của các hợp chất mới dựa trên cấu trúc hóa học của chúng.
- **Phân tích mối quan hệ cấu trúc-hoạt tính (SAR):** Hiểu rõ các đặc điểm cấu trúc nào đóng vai trò quan trọng trong việc quyết định hoạt tính của một hợp chất.
- **Đánh giá tính chất dược động học (ADMET):** Dự đoán khả năng hấp thụ, phân phối, chuyển hóa, thải trừ và độc tính của các hợp chất.

Our main goals are:

- **Identify potential bioactive compounds:** Search for compounds with high activity against specific targets.
- **Build activity prediction models (SAR/QSAR):** Develop models that can predict the activity of new compounds based on their chemical structure.
- **Analyze structure-activity relationships (SAR):** Understand which structural features are important in determining the activity of a compound.
- **Assess pharmacokinetic properties (ADMET):** Predict the absorption, distribution, metabolism, excretion, and toxicity of compounds.

Quy trình phân tích đề xuất (Suggested workflow):

1. **Truy vấn và trích xuất dữ liệu từ ChEMBL 35 (Query and extract data from ChEMBL 35):** Sử dụng SQL để truy vấn dữ liệu liên quan đến mục tiêu nghiên cứu của bạn (ví dụ: một protein cụ thể, một loại bệnh). Lọc dữ liệu dựa trên các tiêu chí như loại hoạt tính (IC50, Ki, EC50), giá trị hoạt tính và đảm bảo chất lượng dữ liệu.
2. **Tiền xử lý dữ liệu (Data preprocessing):** Làm sạch dữ liệu, xử lý các giá trị thiếu, chuẩn hóa các đơn vị hoạt tính và loại bỏ các bản ghi trùng lặp.
3. **Tính toán đặc trưng phân tử (Molecular feature calculation):** Sử dụng RDKit để tính toán các đặc trưng phân tử (descriptors) từ cấu trúc hóa học của các hợp chất. Các đặc trưng này có thể bao gồm các thuộc tính vật lý hóa học, topo, và vân tay (fingerprints).

4. **Phân tích khám phá dữ liệu (Exploratory data analysis - EDA):** Sử dụng các kỹ thuật trực quan hóa dữ liệu (ví dụ: biểu đồ phân tán, biểu đồ hộp) để khám phá các mối quan hệ giữa các đặc trưng phân tử và hoạt tính sinh học.
5. **Xây dựng mô hình học máy (Machine learning model building):** Sử dụng các thuật toán học máy (ví dụ: hồi quy tuyến tính, cây quyết định, mạng nơ-ron) để xây dựng các mô hình dự đoán hoạt tính.
6. **Đánh giá mô hình (Model evaluation):** Đánh giá hiệu suất của mô hình bằng cách sử dụng các tập dữ liệu kiểm tra và các chỉ số đánh giá phù hợp (ví dụ: R-squared, RMSE, AUC).
7. **Giải thích mô hình (Model interpretation):** Tìm hiểu các đặc trưng phân tử nào đóng vai trò quan trọng nhất trong việc dự đoán hoạt tính.
8. **Ứng dụng kết quả (Application of results):** Sử dụng các mô hình đã xây dựng để sàng lọc ảo các hợp chất mới, thiết kế các hợp chất có hoạt tính tốt hơn và dự đoán tính chất ADMET.

2. Hướng dẫn (Song ngữ):

Bước 1: Kết nối đến cơ sở dữ liệu ChEMBL 35 (Connect to the ChEMBL 35 database)

```
import psycopg2

# Thông tin kết nối (Connection details)
host = "192.168.206.136"
database = "chembl_35"
user = "rd"
password = "rd"

try:
    # Kết nối đến PostgreSQL (Connect to PostgreSQL)
    conn = psycopg2.connect(host=host, database=database, user=user,
password=password)
    cur = conn.cursor()
    print("Kết nối thành công đến cơ sở dữ liệu ChEMBL 35!")
    print("Successfully connected to the ChEMBL 35 database!")

except psycopg2.Error as e:
    print(f"Lỗi kết nối: {e}")
    print(f"Connection error: {e}")
```

Bước 2: Khắc phục lỗi SQL (Fixing the SQL error)

Lỗi `ERROR: operator does not exist: numeric ~ unknown` xảy ra do bạn đang cố gắng sử dụng toán tử `~` (regular expression match) trên một cột kiểu `numeric`. Để khắc phục, bạn có thể ép kiểu cột `standard_value` sang kiểu `text` trước khi so sánh:

```
SELECT act.molregno, act.standard_value
FROM activities act
WHERE act.standard_type = 'IC50'
  AND act.standard_relation = '='
  AND act.standard_value::text ~ '^[0-9\.]+$' -- Ép kiểu sang text (Cast to text)
LIMIT 100;
```

Explanation:

- `act.standard_value::text` : This part explicitly casts the numeric `standard_value` column to text.
- `~ '^[0-9\.]+$'` : This uses regular expressions to check if the value consists of only numbers and decimal points.

Bước 3: Khắc phục lỗi Scikit-learn (Fixing the Scikit-learn error)

Nếu bạn đang sử dụng phiên bản Scikit-learn cũ, hãy nâng cấp lên phiên bản mới nhất hoặc loại bỏ tham số `squared=False` nếu không cần thiết. Nếu bạn cần tính toán RMSE (Root Mean Squared Error), bạn có thể làm như sau:

```
from sklearn.metrics import mean_squared_error
import numpy as np

# Ví dụ (Example)
y_true = [3, -0.5, 2, 7]
y_predicted = [2.5, 0.0, 2, 8]

# Tính MSE (Calculate MSE)
mse = mean_squared_error(y_true, y_predicted)

# Tính RMSE (Calculate RMSE)
rmse = np.sqrt(mse)

print(f"MSE: {mse}")
print(f"RMSE: {rmse}")
```

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

Ví dụ 1: Lấy 100 hợp chất có IC50 cho một mục tiêu cụ thể (Get 100 compounds with IC50 for a specific target)

```
-- SQL
SELECT act.molregno, act.standard_value, act.standard_units
FROM activities act
JOIN target_dictionary td ON act.tid = td.tid
WHERE td.chembl_id = 'CHEMBL205'  -- Thay đổi thành CHEMBL ID của mục tiêu bạn quan
tâm (Change to your target CHEMBL ID)
  AND act.standard_type = 'IC50'
  AND act.standard_relation = '='
  AND act.standard_value::text ~ '^[0-9\.]+$'
LIMIT 100;

# Python
import pandas as pd

# Thực hiện truy vấn SQL (Execute the SQL query)
sql_query = """
SELECT act.molregno, act.standard_value, act.standard_units
FROM activities act
JOIN target_dictionary td ON act.tid = td.tid
WHERE td.chembl_id = 'CHEMBL205'
  AND act.standard_type = 'IC50'
  AND act.standard_relation = '='
  AND act.standard_value::text ~ '^[0-9\.]+$'
LIMIT 100;
"""

df = pd.read_sql_query(sql_query, conn)

print(df.head())
```

Ví dụ 2: Tính toán đặc trưng phân tử sử dụng RDKit (Calculate molecular features using RDKit)

```
# Python
from rdkit import Chem
from rdkit.Chem import Descriptors
```

```python
def calculate_descriptors(smiles):
    """Calculates molecular descriptors for a given SMILES string."""
    mol = Chem.MolFromSmiles(smiles)
    if mol is None:
        return None
    descriptors = {}
    descriptors['MW'] = Descriptors.MolWt(mol)
    descriptors['LogP'] = Descriptors.MolLogP(mol)
    descriptors['HBD'] = Descriptors.NumHDonors(mol)
    descriptors['HBA'] = Descriptors.NumHAcceptors(mol)
    return descriptors

# Ví dụ sử dụng (Example usage)
smiles = 'CC(=O)Oc1ccccc1C(=O)O'  # Aspirin
descriptors = calculate_descriptors(smiles)
print(descriptors)
```

Ví dụ 3: Phân tích tương quan giữa LogP và IC50 (Correlation analysis between LogP and IC50)

```python
# Python
import pandas as pd
import numpy as np
from rdkit import Chem
from rdkit.Chem import Descriptors

# Hàm tính toán LogP từ SMILES
def calculate_logp(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol:
        return Descriptors.MolLogP(mol)
    return np.nan

# Giả sử bạn đã có DataFrame 'df' từ truy vấn SQL, có chứa molregno và standard_value
# Cần có thêm cột SMILES để tính LogP

# Lấy SMILES từ bảng molecule_dictionary dựa vào molregno
def get_smiles(molregno, conn):
    sql_query = f"""
    SELECT smiles FROM molecule_dictionary WHERE molregno = {molregno};
    """
    smiles = pd.read_sql_query(sql_query, conn).iloc[0]['smiles']
    return smiles

# Áp dụng hàm get_smiles để lấy SMILES cho mỗi molregno
df['smiles'] = df['molregno'].apply(lambda x: get_smiles(x, conn))

# Tính LogP cho mỗi SMILES
df['logp'] = df['smiles'].apply(calculate_logp)

# Chuyển đổi IC50 sang pIC50 (nếu cần)
df['pIC50'] = -np.log10(df['standard_value'] / 1e9)  # Giả sử standard_value ở đơn vị nM

# Tính tương quan
correlation = df['logp'].corr(df['pIC50'])
print(f"Tương quan giữa LogP và pIC50: {correlation}")
```

Ví dụ 4: Sàng lọc ảo dựa trên LogP (Virtual screening based on LogP)

```python
# Python
# Giả sử bạn có một danh sách SMILES 'candidate_smiles'
```

```
candidate_smiles = ['Cc1ccccc1', 'Clc1ccccc1', 'Oc1ccccc1']

# Tính LogP cho các hợp chất ứng viên
candidate_logp = [calculate_logp(smiles) for smiles in candidate_smiles]

# Lọc các hợp chất có LogP nằm trong khoảng mong muốn (ví dụ: 1 đến 3)
desirable_logp_range = (1, 3)
selected_candidates = [smiles for smiles, logp in zip(candidate_smiles,
candidate_logp) if desirable_logp_range[0] <= logp <= desirable_logp_range[1]]

print(f"Các hợp chất ứng viên được chọn: {selected_candidates}")
```

Ví dụ 5: Xây dựng mô hình hồi quy tuyến tính đơn giản (Build a simple linear regression model)

```
# Python
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
import numpy as np

# Chuẩn bị dữ liệu (Prepare the data)
X = df['logp'].values.reshape(-1, 1)  # LogP là biến độc lập (LogP is the independent
variable)
y = df['pIC50'].values  # pIC50 là biến phụ thuộc (pIC50 is the dependent variable)

# Chia dữ liệu thành tập huấn luyện và tập kiểm tra (Split the data into training and
test sets)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Tạo mô hình hồi quy tuyến tính (Create a linear regression model)
model = LinearRegression()

# Huấn luyện mô hình (Train the model)
model.fit(X_train, y_train)

# Dự đoán trên tập kiểm tra (Make predictions on the test set)
y_pred = model.predict(X_test)

# Đánh giá mô hình (Evaluate the model)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)

print(f"MSE: {mse}")
print(f"RMSE: {rmse}")
```

4. Lưu ý quan trọng:

- **Chất lượng dữ liệu:** Đảm bảo dữ liệu ChEMBL 35 bạn sử dụng đã được làm sạch và chuẩn hóa.
- **Lựa chọn đặc trưng:** Lựa chọn các đặc trưng phân tử phù hợp với mục tiêu nghiên cứu của bạn.
- **Đánh giá mô hình:** Đánh giá mô hình một cách khách quan bằng cách sử dụng các tập dữ liệu kiểm tra độc lập.
- **Giải thích kết quả:** Luôn cố gắng giải thích ý nghĩa sinh học của các kết quả bạn thu được.

Tôi hy vọng những hướng dẫn và ví dụ này sẽ giúp bạn bắt đầu dự án của mình một cách hiệu quả. Hãy cho tôi biết nếu bạn có bất kỳ câu hỏi nào khác. Chúc bạn thành công!