PharmApp Suite

🧠 AI for Drug Discovery and Development 💉

🧩 Topic: Trích xuất các hợp chất giống lead

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 bằng RDKit. Chủ đề "Topic_CheMBL_35_3" có vẻ hứa hẹn và tôi sẽ cố gắng đưa ra những hướng dẫn chi tiết nhất.

1. Phân tích Mô hình Phân tích (Analysis Model)

Dựa trên thông tin bạn cung cấp, tôi hình dung mô hình phân tích của bạn như sau:

- **Nguồn Dữ liệu:** ChEMBL 35 (database chứa thông tin về các phân tử, hoạt tính sinh học,...)
- **Công cụ:**
    - PostgreSQL (psql): Hệ quản trị cơ sở dữ liệu quan hệ để truy vấn và trích xuất dữ liệu.
    - RDKit: Thư viện Python chuyên dụng cho tin học hóa học (cheminformatics), cho phép tính toán các đặc tính phân tử (molecular descriptors), tạo fingerprint, và thực hiện các phân tích cấu trúc.
    - Jupyter Notebook: Môi trường lập trình tương tác để viết code Python, hiển thị kết quả và trực quan hóa dữ liệu.
    - Scikit-learn: Thư viện Python cho machine learning, có thể sử dụng để xây dựng các mô hình dự đoán hoạt tính, phân loại, hoặc các mục tiêu khác.
- **Quy trình:**
    1. **Trích xuất dữ liệu từ ChEMBL 35:** Sử dụng SQL để truy vấn database và lấy dữ liệu cần thiết (ví dụ: cấu trúc phân tử, giá trị hoạt tính, thông tin về target). Lưu dữ liệu vào file CSV.
    2. **Tiền xử lý dữ liệu:**
        - Đọc dữ liệu từ file CSV vào Jupyter Notebook.
        - Sử dụng RDKit để chuyển đổi cấu trúc phân tử (ví dụ: SMILES) thành đối tượng RDKit.
        - Tính toán các đặc tính phân tử bằng RDKit (ví dụ: molecular weight, logP, số lượng H-bond donors/acceptors,...).
        - Làm sạch dữ liệu (xử lý giá trị thiếu, outlier,...).
    3. **Phân tích dữ liệu:**
        - Thống kê mô tả (descriptive statistics) để hiểu phân bố dữ liệu.
        - Trực quan hóa dữ liệu (visualization) để tìm kiếm các xu hướng, mối quan hệ.
        - Xây dựng mô hình machine learning (nếu cần) để dự đoán hoạt tính hoặc phân loại các phân tử.
    4. **Diễn giải kết quả:** Đưa ra các kết luận dựa trên phân tích dữ liệu, ví dụ:
        - Các đặc tính phân tử nào liên quan đến hoạt tính?
        - Có thể dự đoán hoạt tính của một phân tử mới dựa trên cấu trúc của nó không?
        - Có thể tìm ra các "scaffold" (khung phân tử) tiềm năng cho việc phát triển thuốc không?

2. Hướng dẫn Song ngữ (Bilingual Instructions)

2.1. Sửa lỗi SQL (Fix SQL Error)

Lỗi `ERROR: operator does not exist: numeric ~ unknown` xảy ra vì bạn đang cố gắng sử dụng toán tử ~ (regular expression match) trên một cột kiểu số (numeric). Bạn cần chuyển đổi cột số thành kiểu chuỗi (text) trước khi so sánh với regular expression.

English:

The error `ERROR: operator does not exist: numeric ~ unknown` occurs because you are trying to use the ~ operator (regular expression match) on a numeric column. You need to cast the numeric column to text before comparing it with a regular expression.

SQL Code (ví dụ):

```sql
SELECT *
FROM activities act
WHERE act.standard_type = 'IC50'
  AND act.standard_relation = '='
  AND CAST(act.standard_value AS TEXT) ~ '^[0-9\.]+$' -- Cast to TEXT
LIMIT 100;
```

Tiếng Việt:

Lỗi `ERROR: operator does not exist: numeric ~ unknown` xảy ra do bạn đang cố gắng sử dụng toán tử ~ (so khớp biểu thức chính quy) trên một cột kiểu số. Bạn cần chuyển đổi cột số thành kiểu chuỗi trước khi so sánh với biểu thức chính quy.

Code SQL (ví dụ):

```sql
SELECT *
FROM activities act
WHERE act.standard_type = 'IC50'
  AND act.standard_relation = '='
  AND CAST(act.standard_value AS TEXT) ~ '^[0-9\.]+$' -- Chuyển đổi sang TEXT
LIMIT 100;
```

2.2. Sửa lỗi Scikit-learn (Fix Scikit-learn Error)

Nếu bạn đang sử dụng phiên bản scikit-learn cũ (trước 0.22), tham số `squared=False` không được hỗ trợ trong hàm `mean_squared_error`. Bạn có thể nâng cấp scikit-learn hoặc tính căn bậc hai của kết quả `mean_squared_error` để có được RMSE (Root Mean Squared Error).

English:

If you are using an old version of scikit-learn (before 0.22), the `squared=False` parameter is not supported in the `mean_squared_error` function. You can upgrade scikit-learn or calculate the square root of the `mean_squared_error` result to get RMSE (Root Mean Squared Error).

Python Code (ví dụ):

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

# Giả sử y_true và y_pred là các mảng chứa giá trị thực tế và giá trị dự đoán
mse = mean_squared_error(y_true, y_pred)
rmse = np.sqrt(mse)  # Tính RMSE
print(f"RMSE: {rmse}")
```

Tiếng Việt:

Nếu bạn đang sử dụng phiên bản scikit-learn cũ (trước 0.22), tham số `squared=False` không được hỗ trợ trong hàm `mean_squared_error`. Bạn có thể nâng cấp scikit-learn hoặc tính căn bậc hai của kết quả `mean_squared_error` để có được RMSE (Root Mean Squared Error).

Code Python (ví dụ):

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

# Giả sử y_true và y_pred là các mảng chứa giá trị thực tế và giá trị dự đoán
mse = mean_squared_error(y_true, y_pred)
rmse = np.sqrt(mse)  # Tính RMSE
print(f"RMSE: {rmse}")
```

3. Code SQL, Python (English)

3.1. SQL (Example)

```sql
-- Select molecule details and IC50 values for a specific target
SELECT
    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'  -- Example target CHEMBL ID
    AND act.standard_type = 'IC50'
    AND act.standard_relation = '='
    AND CAST(act.standard_value AS TEXT) ~ '^[0-9\.]+$'
LIMIT 100;
```

3.2. Python (Example)

```python
import pandas as pd
from rdkit import Chem
from rdkit.Chem import Descriptors

# Load data from CSV
data = pd.read_csv('../data/your_data.csv')  # Replace 'your_data.csv'

# Function to calculate molecular weight
def calculate_mw(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol:
        return Descriptors.MolWt(mol)
    else:
        return None

# Apply the function to the SMILES column
data['molecular_weight'] = data['canonical_smiles'].apply(calculate_mw)

print(data.head())
```

4. Ví dụ Code SQL và Python mẫu (5 Examples)

4.1. Ví dụ 1: Lấy dữ liệu cơ bản và tính Molecular Weight (Fetch Basic Data and Calculate Molecular Weight)

SQL:

```sql
SELECT md.chembl_id, cs.canonical_smiles
FROM molecule_dictionary md
JOIN compound_structures cs ON md.molregno = cs.molregno
LIMIT 100;
```

Python:

```python
import pandas as pd
from rdkit import Chem
from rdkit.Chem import Descriptors

data = pd.read_csv('../data/example1.csv') # Replace 'example1.csv'

def calculate_mw(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol:
        return Descriptors.MolWt(mol)
    else:
        return None

data['molecular_weight'] = data['canonical_smiles'].apply(calculate_mw)
print(data.head())
```

4.2. Ví dụ 2: Lọc theo IC50 và tính LogP (Filter by IC50 and Calculate LogP)

SQL:

```sql
SELECT md.chembl_id, cs.canonical_smiles, act.standard_value
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 CAST(act.standard_value AS TEXT) ~ '^[0-9\.]+$'
LIMIT 100;
```

Python:

```python
import pandas as pd
from rdkit import Chem
from rdkit.Chem import Descriptors
from rdkit.Chem import AllChem

data = pd.read_csv('../data/example2.csv') # Replace 'example2.csv'

def calculate_logp(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol:
        return Descriptors.MolLogP(mol)
    else:
        return None

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

4.3. Ví dụ 3: Lấy dữ liệu liên quan đến một target cụ thể (Fetch Data Related to a Specific Target)

SQL:

```sql
SELECT md.chembl_id, cs.canonical_smiles, act.standard_value
FROM molecule_dictionary md
JOIN compound_structures cs ON md.molregno = cs.molregno
```

```sql
JOIN activities act ON md.molregno = act.molregno
JOIN target_dictionary td ON act.tid = td.tid
WHERE td.chembl_id = 'CHEMBL182' -- Ví dụ: EGFR
AND act.standard_type = 'IC50'
AND CAST(act.standard_value AS TEXT) ~ '^[0-9\.]+$'
LIMIT 100;
```

Python:

```python
import pandas as pd

data = pd.read_csv('../data/example3.csv') # Replace 'example3.csv'
print(data.head())
```

4.4. Ví dụ 4: Tính số lượng H-bond donors và acceptors (Calculate Number of H-bond Donors and Acceptors)

SQL:

```sql
SELECT md.chembl_id, cs.canonical_smiles
FROM molecule_dictionary md
JOIN compound_structures cs ON md.molregno = cs.molregno
LIMIT 100;
```

Python:

```python
import pandas as pd
from rdkit import Chem
from rdkit.Chem import Descriptors

data = pd.read_csv('../data/example4.csv') # Replace 'example4.csv'

def calculate_hba(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol:
        return Descriptors.NumHAcceptors(mol)
    else:
        return None

def calculate_hbd(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol:
        return Descriptors.NumHDonors(mol)
    else:
        return None


data['hba'] = data['canonical_smiles'].apply(calculate_hba)
data['hbd'] = data['canonical_smiles'].apply(calculate_hbd)
print(data.head())
```

4.5. Ví dụ 5: Tạo Fingerprint và đơn giản thống kê (Generate Fingerprint and Simple Statistics)

SQL:

```sql
SELECT md.chembl_id, cs.canonical_smiles
FROM molecule_dictionary md
JOIN compound_structures cs ON md.molregno = cs.molregno
LIMIT 100;
```

Python:

```python
import pandas as pd
from rdkit import Chem
from rdkit.Chem import AllChem
from rdkit import DataStructs

data = pd.read_csv('../data/example5.csv') # Replace 'example5.csv'

def generate_fingerprint(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol:
        fp = AllChem.GetMorganFingerprintAsBitVect(mol, 2, nBits=1024)
        return fp
    else:
        return None

data['fingerprint'] = data['canonical_smiles'].apply(generate_fingerprint)
print(data.head())

# Simple statistics on fingerprints (example: count valid fingerprints)
valid_fingerprints = data['fingerprint'].count()
print(f"Number of valid fingerprints: {valid_fingerprints}")
```

Lưu ý:

- Nhớ thay đổi tên file CSV trong code Python cho phù hợp với file bạn đã tạo.
- Các ví dụ trên chỉ là bước khởi đầu. Bạn có thể tùy chỉnh và mở rộng chúng để phù hợp với mục tiêu nghiên cứu cụ thể của mình.
- Hãy nhớ sử dụng `os.path.join(base_path, ...)` để xây dựng đường dẫn một cách chính xác và dễ quản lý.
- Đối với các phân tích phức tạp hơn, bạn có thể cần sử dụng các mô hình machine learning từ scikit-learn.

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