PharmApp Suite

🧠 AI for Drug Discovery and Development 💉

🧩 Topic: Phân tích thay đổi khung hóa học (scaffold hopping)

Tuyệt vời! Với vai trò là một chuyên gia R&D dược học, tôi rất vui được hỗ trợ bạn trong dự án phân tích dữ liệu ChEMBL 35 này. Chúng ta sẽ cùng nhau xây dựng một quy trình làm việc hiệu quả, từ truy vấn SQL đến phân tích dữ liệu bằng Python và RDKit.

Tổng quan dự án

Dự án của bạn tập trung vào việc khai thác 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. Bạn đã thiết lập một môi trường làm việc theo chuẩn MLOps, sử dụng PostgreSQL làm cơ sở dữ liệu và Jupyter Notebook cho phân tích dữ liệu.

Các bước thực hiện

1. **Truy vấn dữ liệu từ ChEMBL 35 bằng SQL:** Sử dụng pgAdmin để chạy các truy vấn SQL và lưu kết quả vào các file CSV.
2. **Tiền xử lý và phân tích dữ liệu bằng Python và RDKit:** Sử dụng Jupyter Notebook để đọc dữ liệu từ file CSV, tiền xử lý dữ liệu (ví dụ: chuẩn hóa SMILES, tính toán descriptors phân tử), và thực hiện các phân tích thống kê và mô hình hóa.
3. **Lưu trữ kết quả và báo cáo:** Lưu trữ kết quả phân tích và các mô hình đã xây dựng, đồng thời tạo báo cáo chi tiết về quá trình thực hiện và kết quả đạt được.

Phân tích và Hướng dẫn chi tiết (song ngữ)

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

Mô hình phân tích của bạn nên tập trung vào việc trả lời các câu hỏi nghiên cứu cụ thể liên quan đến dữ liệu ChEMBL 35. Dưới đây là một số gợi ý:

- **Phân tích mối quan hệ cấu trúc-hoạt tính (SAR/QSAR):** Xác định các đặc điểm cấu trúc phân tử quan trọng ảnh hưởng đến hoạt tính sinh học của các hợp chất.
- **Xác định các hit/lead tiềm năng:** Lọc và đánh giá các hợp chất có hoạt tính cao và phù hợp với các tiêu chí phát triển thuốc.
- **Phân tích đa dạng hóa thư viện hợp chất:** Đánh giá sự đa dạng về cấu trúc và hoạt tính của các hợp chất trong cơ sở dữ liệu.
- **Dự đoán hoạt tính của các hợp chất mới:** Xây dựng các mô hình dự đoán hoạt tính dựa trên cấu trúc phân tử.

Your analysis model should focus on answering specific research questions related to the ChEMBL 35 data. Here are some suggestions:

- ***Structure-Activity Relationship (SAR/QSAR) Analysis:*** *Identify key molecular structural features that influence the biological activity of compounds.*
- ***Identification of Potential Hits/Leads:*** *Filter and evaluate compounds with high activity and suitability for drug development criteria.*
- ***Compound Library Diversification Analysis:*** *Assess the structural and activity diversity of compounds in the database.*
- ***Predicting Activity of New Compounds:*** *Build activity prediction models based on molecular structure.*

2. Hướng dẫn song ngữ (Bilingual Guide)

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

- **SQL:**
  - Sử dụng các câu lệnh `SELECT`, `FROM`, `WHERE`, `JOIN` để truy vấn dữ liệu từ các bảng khác nhau trong cơ sở dữ liệu ChEMBL 35.
  - Sử dụng các hàm SQL như `LIKE`, `UPPER`, `LOWER` để lọc và chuẩn hóa dữ liệu.
  - Sử dụng các hàm thống kê như `AVG`, `MAX`, `MIN`, `COUNT` để tính toán các giá trị thống kê.
  - *Use `SELECT`, `FROM`, `WHERE`, `JOIN` statements to query data from different tables in the ChEMBL 35 database.*
  - *Use SQL functions like `LIKE`, `UPPER`, `LOWER` to filter and normalize data.*
  - *Use statistical functions like `AVG`, `MAX`, `MIN`, `COUNT` to calculate statistical values.*
- **Python:**
  - Sử dụng thư viện `pandas` để đọc dữ liệu từ file CSV và xử lý dữ liệu.
  - Sử dụng thư viện `RDKit` để đọc và xử lý cấu trúc phân tử (ví dụ: chuyển đổi SMILES sang các định dạng khác, tính toán descriptors phân tử).
  - Sử dụng thư viện `scikit-learn` để xây dựng các mô hình học máy (ví dụ: hồi quy tuyến tính, cây quyết định, mạng nơ-ron).
  - Sử dụng thư viện `matplotlib` và `seaborn` để trực quan hóa dữ liệu.
  - *Use the `pandas` library to read data from CSV files and process data.*
  - *Use the `RDKit` library to read and process molecular structures (e.g., convert SMILES to other formats, calculate molecular descriptors).*
  - *Use the `scikit-learn` library to build machine learning models (e.g., linear regression, decision trees, neural networks).*
  - *Use the `matplotlib` and `seaborn` libraries to visualize data.*

3. Code SQL, Python (English)

SQL:

```sql
-- Select 100 compounds with IC50 values against a specific target
SELECT
    md.chembl_id,
    act.standard_value,
    act.standard_units
FROM
    molecule_dictionary md
JOIN
    activities act ON md.molregno = act.molregno
WHERE
    act.standard_type = 'IC50'
    AND act.target_chembl_id = 'CHEMBL205' -- Replace with your target of interest
    AND act.standard_value IS NOT NULL
    AND act.standard_units = 'nM'
    AND act.standard_value ~ '^[0-9\.]+$' --Ensure that standard_value contains only numbers
LIMIT 100;
```

Python:

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

# Define base path
base_path = "../data"

# Read data from CSV file
csv_file = "chembl_ic50_data.csv"  # Replace with your CSV file name
data = pd.read_csv(os.path.join(base_path, csv_file))

# 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 the first 5 rows of the DataFrame
print(data.head())
```

Giải thích lỗi và cách khắc phục:

- **Lỗi SQL:** `ERROR: operator does not exist: numeric ~ unknown`
    - Nguyên nhân: Lỗi này xảy ra do bạn đang cố gắng sử dụng toán tử ~ (regular expression match) trên một cột kiểu số (numeric).
    - Khắc phục:
        1. Đảm bảo cột `act.standard_value` có kiểu dữ liệu phù hợp (ví dụ: text hoặc varchar).
        2. Nếu cột `act.standard_value` là kiểu số, bạn có thể sử dụng hàm `CAST` để chuyển đổi nó sang kiểu text trước khi sử dụng toán tử ~.

```sql
--Corrected SQL code
SELECT
    md.chembl_id,
    act.standard_value,
    act.standard_units
FROM
    molecule_dictionary md
JOIN
    activities act ON md.molregno = act.molregno
WHERE
    act.standard_type = 'IC50'
    AND act.target_chembl_id = 'CHEMBL205' -- Replace with your target of interest
    AND act.standard_value IS NOT NULL
    AND act.standard_units = 'nM'
    AND CAST(act.standard_value AS TEXT) ~ '^[0-9\.]+$' --Ensure that standard_value
contains only numbers
LIMIT 100;
```

- **Lỗi Python:** `TypeError: mean_squared_error() got an unexpected keyword argument 'squared'`
    - Nguyên nhân: Phiên bản scikit-learn bạn đang sử dụng không hỗ trợ tham số `squared` trong hàm `mean_squared_error`.

- Khắc phục: Nâng cấp phiên bản scikit-learn lên phiên bản mới hơn (ví dụ: 1.0 trở lên) hoặc bỏ tham số `squared=False` (khi đó hàm sẽ trả về giá trị MSE thay vì RMSE).

```
from sklearn.metrics import mean_squared_error

# Tính MSE
mse = mean_squared_error(y_true, y_pred)

# Nếu cần tính RMSE
rmse = mse**0.5
```

4. Ví dụ code SQL và Python mẫu

Ví dụ 1: Truy vấn thông tin cơ bản về hợp chất (SQL)

```
-- Get basic information about compounds
SELECT
    md.chembl_id,
    md.pref_name,
    cs.canonical_smiles
FROM
    molecule_dictionary md
JOIN
    compound_structures cs ON md.molregno = cs.molregno
LIMIT 100;
```

Ví dụ 1: Tính toán logP (Python)

```
from rdkit import Chem
from rdkit.Chem import AllChem
from rdkit.Chem import Descriptors

def calculate_logp(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        logp = Descriptors.MolLogP(mol)
        return logp
    else:
        return None

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

Ví dụ 2: Truy vấn hoạt tính của hợp chất trên một target cụ thể (SQL)

```
-- Get activity data for a specific target
SELECT
    md.chembl_id,
    act.standard_type,
    act.standard_value,
    act.standard_units
FROM
    molecule_dictionary md
JOIN
    activities act ON md.molregno = act.molregno
WHERE
    act.target_chembl_id = 'CHEMBL205'
    AND act.standard_type IN ('IC50', 'Ki', 'Kd')
LIMIT 100;
```

Ví dụ 2: Vẽ biểu đồ phân bố logP (Python)

```python
import matplotlib.pyplot as plt
import seaborn as sns

sns.histplot(data['logp'].dropna())
plt.title('Distribution of logP')
plt.xlabel('logP')
plt.ylabel('Frequency')
plt.show()
```

Ví dụ 3: Truy vấn các hợp chất có hoạt tính IC50 dưới 100 nM (SQL)

```sql
-- Get compounds with IC50 < 100 nM
SELECT
    md.chembl_id,
    act.standard_value
FROM
    molecule_dictionary md
JOIN
    activities act ON md.molregno = act.molregno
WHERE
    act.standard_type = 'IC50'
    AND act.standard_value < 100
    AND act.standard_units = 'nM'
LIMIT 100;
```

Ví dụ 3: Tính toán và thêm số liên kết hydro chấp nhận (HBA) và cho (HBD) (Python)

```python
from rdkit.Chem import Lipinski

def calculate_hba(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        hba = Lipinski.NumHAcceptors(mol)
        return hba
    else:
        return None

def calculate_hbd(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        hbd = Lipinski.NumHDonors(mol)
        return hbd
    else:
        return None

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

print(data[['canonical_smiles', 'hba', 'hbd']].head())
```

Ví dụ 4: Truy vấn các target liên quan đến một bệnh cụ thể (SQL)

```sql
-- Get targets associated with a specific disease
SELECT
    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'
LIMIT 100;
```

Ví dụ 4: Trực quan hóa HBA và HBD (Python)

```
sns.scatterplot(x='hba', y='hbd', data=data)
plt.title('HBA vs HBD')
plt.xlabel('Number of Hydrogen Bond Acceptors')
plt.ylabel('Number of Hydrogen Bond Donors')
plt.show()
```

Ví dụ 5: Tính TPSA (SQL)

```
--this function does not exist in SQL, this is only an example of a query
--that needs to be implemented elsewhere
SELECT smiles, TPSA(smiles) FROM compounds LIMIT 100;
```

Ví dụ 5: Tính TPSA (Python)

```
from rdkit.Chem import rdMolDescriptors

def calculate_tpsa(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        tpsa = rdMolDescriptors.CalcTPSA(mol)
        return tpsa
    else:
        return None

data['tpsa'] = data['canonical_smiles'].apply(calculate_tpsa)
print(data[['canonical_smiles', 'tpsa']].head())
```

Lưu ý quan trọng:

- **Thay đổi tên file và target:** Đảm bảo bạn thay đổi tên file CSV (`chembl_ic50_data.csv`) và target ID (`CHEMBL205`) trong các ví dụ trên cho phù hợp với dữ liệu và mục tiêu nghiên cứu của bạn.
- **Cài đặt thư viện:** Đảm bảo bạn đã cài đặt đầy đủ các thư viện cần thiết (pandas, RDKit, scikit-learn, matplotlib, seaborn). Bạn có thể sử dụng pip để cài đặt: `pip install pandas rdkit scikit-learn matplotlib seaborn`
- **Xử lý lỗi:** Luôn kiểm tra và xử lý các lỗi có thể xảy ra trong quá trình truy vấn và phân tích dữ liệu.
- **Tối ưu hóa truy vấn SQL:** Để tăng tốc độ truy vấn, bạn có thể sử dụng các chỉ mục (indexes) trên các cột thường xuyên được sử dụng trong mệnh đề `WHERE`.

Hy vọng những hướng dẫn và ví dụ trên sẽ giúp bạn triển khai dự án phân tích dữ liệu ChEMBL 35 một cách hiệu quả. Chúc bạn thành công!