# PharmApp Suite

## 🧠 AI for Drug Discovery and Development 💉



## 🧩 Topic: Phân bố các nhóm target trong ChEMBL

Tuyệt vời! Tôi rất vui được hỗ trợ bạn trong dự án phân tích dữ liệu ChEMBL 35 sử dụng RDKit. Với vai trò là chuyên gia nghiên cứu và phát triển dược học, tôi sẽ cung cấp hướng dẫn chi tiết, song ngữ, code mẫu và phân tích để giúp bạn đạt được mục tiêu nghiên cứu.

**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 RDKit để hỗ trợ quá trình nghiên cứu và phát triển thuốc. Bạn sử dụng PostgreSQL (psql) để quản lý dữ liệu và Jupyter Notebook để phân tích. Việc tuân thủ cấu trúc thư mục AIMLOps Template là một điểm cộng lớn, giúp dự án có tổ chức và dễ quản lý.

**Các bước thực hiện và hướng dẫn chi tiết:**

**1. Chuẩn bị dữ liệu (SQL):**

- **Mục tiêu:** Lọc và trích xuất dữ liệu cần thiết từ ChEMBL 35.
- **Code SQL mẫu (Topic_CheMBL_35_24_1.sql):**

```sql
-- English
-- Extracting 100 data points from ChEMBL 35 for activity analysis.
-- Filters for specific activity types (IC50, Ki) and numeric standard values.

-- Vietnamese
-- Trích xuất 100 điểm dữ liệu từ ChEMBL 35 để phân tích hoạt tính.
-- Lọc theo các loại hoạt tính cụ thể (IC50, Ki) và giá trị chuẩn số.

SELECT
    md.chembl_id,
    act.standard_type,
    act.standard_value,
    act.standard_units,
    cmp.canonical_smiles
FROM
    molecule_dictionary md
JOIN
    activities act ON md.molregno = act.molregno
JOIN
    compound_structures cmp ON md.molregno = cmp.molregno
WHERE
    act.standard_type IN ('IC50', 'Ki')
    AND act.standard_value IS NOT NULL
    AND act.standard_value ~ '^[0-9\.]+$' --Fix cho bug
LIMIT 100;
```

- **Giải thích:**
    - `SELECT`: Chọn các cột dữ liệu quan trọng (ChEMBL ID, loại hoạt tính, giá trị chuẩn, đơn vị, SMILES).
    - `FROM`: Kết hợp các bảng `molecule_dictionary`, `activities`, và `compound_structures` để lấy thông tin đầy đủ.
    - `WHERE`: Lọc dữ liệu theo loại hoạt tính (IC50, Ki) và đảm bảo giá trị chuẩn là số.
    - `LIMIT`: Giới hạn số lượng bản ghi để tránh quá tải máy chủ.

    - **Fix bug**: `act.standard_value ~ '^[0-9\.]+$'` thay thế cho việc so sánh trực tiếp với kiểu `numeric`. Biểu thức chính quy này kiểm tra xem giá trị có phải là một số (có thể chứa dấu chấm) hay không.
- **Lưu ý:**
    - Sử dụng `psql` hoặc pgAdmin để chạy script này trên máy chủ PostgreSQL của bạn.
    - Lưu kết quả vào file CSV (ví dụ: `../data/chembl_35_activity_data.csv`).

**2. Phân tích dữ liệu với RDKit (Python):**

- **Mục tiêu:** Đọc dữ liệu từ file CSV, tiền xử lý, tính toán đặc trưng phân tử và xây dựng mô hình.
- **Code Python mẫu (Topic_CheMBL_35_24_2.ipynb):**

```
# English
# Importing necessary libraries
# Reading the CSV data, calculating molecular descriptors, and building a simple
model.

# Vietnamese
# Nhập các thư viện cần thiết
# Đọc dữ liệu CSV, tính toán các descriptor phân tử và xây dựng một mô hình đơn giản.

import os
import pandas as pd
from rdkit import Chem
from rdkit.Chem import Descriptors
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

# Define the base path
base_path = ".."  # Assuming the notebook is one level deep inside the project

# Construct the data path
data_path = os.path.join(base_path, "data", "chembl_35_activity_data.csv")

# Read the CSV file into a pandas DataFrame
df = pd.read_csv(data_path)

# Function to calculate molecular descriptors using RDKit
def calculate_descriptors(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is None:
        return None
    descriptors = {}
    descriptors['MolWt'] = Descriptors.MolWt(mol)
    descriptors['LogP'] = Descriptors.MolLogP(mol)
    descriptors['HBD'] = Descriptors.NumHDonors(mol)
    descriptors['HBA'] = Descriptors.NumHAcceptors(mol)
    return descriptors

# Apply the descriptor calculation to each SMILES string
df['descriptors'] = df['canonical_smiles'].apply(calculate_descriptors)

# Handle missing descriptors (if any)
df = df.dropna(subset=['descriptors'])

# Convert descriptors to DataFrame
df_descriptors = pd.DataFrame(df['descriptors'].tolist())
```

```
# Merge descriptors with the original DataFrame
df = pd.concat([df, df_descriptors], axis=1)

# Prepare data for modeling
X = df[['MolWt', 'LogP', 'HBD', 'HBA']]  # Feature matrix
y = df['standard_value']  # Target variable

# 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)

# Train a 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)
print(f"Mean Squared Error: {mse}")
```

- **Giải thích:**
    - **Import thư viện:** Nhập các thư viện cần thiết (pandas, RDKit, scikit-learn).
    - **Đọc dữ liệu:** Đọc file CSV chứa dữ liệu đã trích xuất từ ChEMBL.
    - **Tính toán đặc trưng phân tử:** Sử dụng RDKit để tính toán các đặc trưng phân tử (ví dụ: MolWt, LogP, HBD, HBA) từ chuỗi SMILES.
    - **Xử lý dữ liệu thiếu:** Loại bỏ các dòng có giá trị đặc trưng bị thiếu.
    - **Chuẩn bị dữ liệu cho mô hình:** Chọn các đặc trưng làm biến độc lập (X) và giá trị hoạt tính làm biến phụ thuộc (y).
    - **Chia dữ liệu:** Chia dữ liệu thành tập huấn luyện và tập kiểm tra.
    - **Huấn luyện mô hình:** Sử dụng mô hình hồi quy tuyến tính để huấn luyện trên tập huấn luyện.
    - **Đánh giá mô hình:** Đánh giá hiệu suất của mô hình trên tập kiểm tra bằng cách tính Mean Squared Error (MSE).

**3. Sửa lỗi:**

- **Lỗi SQL:** `ERROR: operator does not exist: numeric ~ unknown`
    - **Nguyên nhân:** PostgreSQL không hỗ trợ toán tử ~ (regular expression match) trực tiếp trên kiểu dữ liệu `numeric`.
    - **Giải pháp:** Chuyển đổi giá trị `standard_value` sang kiểu `text` trước khi so sánh bằng cách sử dụng `CAST(act.standard_value AS TEXT) ~ '^[0-9\.]+$'`.
- **Lỗi Python:** `TypeError: mean_squared_error() got an unexpected keyword argument 'squared'`
    - **Nguyên nhân:** Phiên bản scikit-learn cũ không hỗ trợ tham số `squared=False` trong hàm `mean_squared_error`.
    - **Giải pháp:** Nâng cấp scikit-learn lên phiên bản mới hơn (`pip install --upgrade scikit-learn`) hoặc tính căn bậc hai của MSE để có RMSE (Root Mean Squared Error).

**4. Ví dụ code SQL và Python:**

**Ví dụ 1: Lọc theo khoảng giá trị hoạt tính:**

```sql
-- English
-- Select compounds with IC50 values between 100 and 1000.

-- Vietnamese
-- Chọn các hợp chất có giá trị IC50 nằm trong khoảng từ 100 đến 1000.

SELECT chembl_id, standard_value
FROM activities act
JOIN molecule_dictionary md ON act.molregno = md.molregno
WHERE standard_type = 'IC50' AND standard_value BETWEEN 100 AND 1000
LIMIT 100;
```

```python
# English
# Filter DataFrame for compounds with MolWt between 200 and 400.

# Vietnamese
# Lọc DataFrame để tìm các hợp chất có MolWt nằm trong khoảng từ 200 đến 400.

df_filtered = df[(df['MolWt'] >= 200) & (df['MolWt'] <= 400)]
print(df_filtered.head())
```

**Ví dụ 2: Tính số lượng hợp chất theo loại hoạt tính:**

```sql
-- English
-- Count the number of compounds for each standard type.

-- Vietnamese
-- Đếm số lượng hợp chất cho mỗi loại hoạt tính.

SELECT standard_type, COUNT(*)
FROM activities
GROUP BY standard_type;
```

```python
# English
# Calculate and print the average LogP.

# Vietnamese
# Tính toán và in ra giá trị LogP trung bình.

print(f"Average LogP: {df['LogP'].mean()}")
```

**Ví dụ 3: Kết hợp dữ liệu từ nhiều bảng:**

```sql
-- English
-- Retrieve ChEMBL ID, SMILES, and target name for specific activities.

-- Vietnamese
-- Lấy ChEMBL ID, SMILES và tên target cho các hoạt tính cụ thể.

SELECT md.chembl_id, cmp.canonical_smiles, td.pref_name
FROM molecule_dictionary md
JOIN activities act ON md.molregno = act.molregno
JOIN compound_structures cmp ON md.molregno = cmp.molregno
JOIN target_dictionary td ON act.tid = td.tid
WHERE act.standard_type = 'IC50'
LIMIT 100;
```

```python
# English
# Create a scatter plot of MolWt vs. LogP.

# Vietnamese
```

```
# Vẽ biểu đồ phân tán của MolWt so với LogP.

import matplotlib.pyplot as plt
plt.scatter(df['MolWt'], df['LogP'])
plt.xlabel('MolWt')
plt.ylabel('LogP')
plt.title('MolWt vs. LogP')
plt.show()
```

**Ví dụ 4: Sắp xếp dữ liệu theo giá trị hoạt tính:**

```
-- English
-- Retrieve ChEMBL ID and standard value, ordered by standard value in ascending
order.

-- Vietnamese
-- Lấy ChEMBL ID và giá trị chuẩn, sắp xếp theo giá trị chuẩn theo thứ tự tăng dần.

SELECT md.chembl_id, act.standard_value
FROM molecule_dictionary md
JOIN activities act ON md.molregno = act.molregno
WHERE act.standard_type = 'IC50'
ORDER BY act.standard_value ASC
LIMIT 100;
```

```
# English
# Calculate the correlation matrix of the descriptors.

# Vietnamese
# Tính toán ma trận tương quan của các descriptor.

correlation_matrix = df[['MolWt', 'LogP', 'HBD', 'HBA']].corr()
print(correlation_matrix)
```

**Ví dụ 5: Lọc dữ liệu theo SMILES:**

```
-- English
-- Retrieve molecules with SMILES containing a specific substructure.

-- Vietnamese
-- Lấy các phân tử có SMILES chứa một cấu trúc con cụ thể.

SELECT md.chembl_id, cmp.canonical_smiles
FROM molecule_dictionary md
JOIN compound_structures cmp ON md.molregno = cmp.molregno
WHERE cmp.canonical_smiles LIKE '%c1ccccc1%'
LIMIT 100;
```

```
# English
# Save the filtered DataFrame to a new CSV file.

# Vietnamese
# Lưu DataFrame đã lọc vào một tệp CSV mới.

df_filtered.to_csv(os.path.join(base_path, "data", "filtered_data.csv"), index=False)
```

**Phân tích mô hình:**

Mô hình hồi quy tuyến tính là một khởi đầu tốt để khám phá mối quan hệ giữa các đặc trưng phân tử và hoạt tính. Tuy nhiên, cần lưu ý rằng:

- **Tính tuyến tính:** Mô hình này giả định mối quan hệ tuyến tính giữa các biến, điều này có thể không đúng trong thực tế.
- **Đa cộng tuyến:** Các đặc trưng phân tử có thể tương quan với nhau, ảnh hưởng đến độ ổn định của mô hình.
- **Phạm vi áp dụng:** Mô hình có thể không dự đoán chính xác cho các hợp chất nằm ngoài phạm vi dữ liệu huấn luyện.

**Các bước tiếp theo:**

1. **Thử nghiệm các mô hình khác:** Sử dụng các mô hình phức tạp hơn như Random Forest, SVM, hoặc Neural Networks.
2. **Lựa chọn đặc trưng:** Sử dụng các phương pháp lựa chọn đặc trưng để chọn ra các đặc trưng quan trọng nhất.
3. **Kiểm tra chéo:** Sử dụng kiểm tra chéo để đánh giá độ ổn định của mô hình.
4. **Giải thích mô hình:** Sử dụng các phương pháp giải thích mô hình (ví dụ: SHAP values) để hiểu rõ hơn về mối quan hệ giữa các đặc trưng và hoạt tính.

Tôi hy vọng những hướng dẫn và code mẫu này sẽ giúp bạn tiến xa trong dự án. Nếu bạn có bất kỳ câu hỏi nào, đừng ngần ngại hỏi!