# PharmApp Suite

## 🧠 AI for Drug Discovery and Development 🧪

Copyright 2025 | Nghiên Cứu Thuốc | www.nghiencuuthuoc.com | Zalo: +84888999311 |

## 🧩 Topic: Gộp ChEMBL với dữ liệu nhà cung cấp hóa chất

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 hiểu tầm quan trọng của việc khai thác thông tin từ cơ sở dữ liệu lớn như ChEMBL để tìm kiếm và phát triển các loại thuốc mới. Chúng ta sẽ cùng nhau xây dựng một quy trình phân tích hiệu quả, tuân thủ theo chuẩn AIMLOps template và giải quyết các vấn đề phát sinh.

**Tổng quan dự án:**

Dự án này tập trung vào việc phân tích dữ liệu từ cơ sở dữ liệu ChEMBL 35, sử dụng PostgreSQL (psql) làm hệ quản trị cơ sở dữ liệu và RDKit để xử lý thông tin hóa học. Chúng ta sẽ trích xuất dữ liệu bằng các truy vấn SQL, lưu trữ dưới dạng file CSV và sau đó sử dụng Jupyter Notebook với Python để phân tích và trực quan hóa dữ liệu.

**1. Phân tích mô hình phân tích (Analysis Model):**

Mô hình phân tích của chúng ta sẽ bao gồm các bước chính sau:

* **Data Extraction (Trích xuất dữ liệu):** Sử dụng SQL queries để trích xuất thông tin liên quan từ cơ sở dữ liệu ChEMBL 35. Chúng ta sẽ tập trung vào các bảng như activities, molecule\_dictionary, compound\_structures,…
* **Data Cleaning and Preprocessing (Làm sạch và tiền xử lý dữ liệu):**
  + Loại bỏ các giá trị bị thiếu hoặc không hợp lệ.
  + Chuyển đổi dữ liệu về định dạng phù hợp (ví dụ: chuyển đổi giá trị IC50 từ dạng chuỗi sang số).
  + Chuẩn hóa dữ liệu (ví dụ: chuẩn hóa SMILES strings).
* **Feature Engineering (Xây dựng đặc trưng):** Sử dụng RDKit để tính toán các đặc trưng hóa học từ cấu trúc phân tử (ví dụ: molecular weight, logP, số lượng hydrogen bond donors/acceptors).
* **Exploratory Data Analysis (EDA - Phân tích khám phá dữ liệu):** Sử dụng các kỹ thuật thống kê và trực quan hóa để hiểu rõ hơn về dữ liệu, tìm kiếm các mối quan hệ và xu hướng tiềm năng.
* **Modeling (Xây dựng mô hình):** Xây dựng các mô hình dự đoán (ví dụ: mô hình dự đoán hoạt tính sinh học) bằng cách sử dụng các thuật toán machine learning như linear regression, random forest, support vector machines,…
* **Model Evaluation (Đánh giá mô hình):** Đánh giá hiệu năng của mô hình bằng các metrics phù hợp (ví dụ: R-squared, RMSE, AUC).

**2. Hướng dẫn song ngữ (Bilingual Guide):**

**2.1. Kết nối cơ sở dữ liệu PostgreSQL (Connecting to PostgreSQL Database):**

* **Tiếng Việt:** Sử dụng thư viện psycopg2 trong Python để kết nối đến cơ sở dữ liệu PostgreSQL.
* **English:** Use the psycopg2 library in Python to connect to the PostgreSQL database.

# Tiếng Việt: Ví dụ kết nối cơ sở dữ liệu  
# English: Example of connecting to the database  
import psycopg2  
  
try:  
 conn = psycopg2.connect(  
 host="192.168.206.136",  
 database="chembl\_35",  
 user="rd",  
 password="rd")  
  
 cur = conn.cursor()  
 print("Kết nối thành công đến PostgreSQL")  
 print("Successfully connected to PostgreSQL")  
  
except psycopg2.Error as e:  
 print(f"Lỗi kết nối: {e}")  
 print(f"Connection error: {e}")  
finally:  
 if conn:  
 cur.close()  
 conn.close()  
 print("Ngắt kết nối PostgreSQL")  
 print("PostgreSQL connection closed")

**2.2. Trích xuất dữ liệu bằng SQL (Extracting Data using SQL):**

* **Tiếng Việt:** Sử dụng các câu lệnh SQL SELECT, JOIN, WHERE để trích xuất dữ liệu từ các bảng khác nhau trong cơ sở dữ liệu ChEMBL.
* **English:** Use SQL statements like SELECT, JOIN, WHERE to extract data from different tables in the ChEMBL database.

-- Tiếng Việt: Ví dụ truy vấn SQL  
-- English: Example SQL query  
SELECT md.chembl\_id,  
 cs.canonical\_smiles,  
 act.standard\_type,  
 act.standard\_value,  
 act.standard\_units  
FROM activities act  
JOIN molecule\_dictionary md ON act.molregno = md.molregno  
JOIN compound\_structures cs ON md.molregno = cs.molregno  
WHERE act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
LIMIT 100;

**2.3. Sử dụng RDKit để tính toán đặc trưng (Using RDKit for Feature Calculation):**

* **Tiếng Việt:** Sử dụng thư viện RDKit để chuyển đổi SMILES strings thành các đối tượng phân tử và tính toán các đặc trưng hóa học.
* **English:** Use the RDKit library to convert SMILES strings into molecule objects and calculate chemical features.

# Tiếng Việt: Ví dụ sử dụng RDKit  
# English: Example of using RDKit  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
  
def calculate\_features(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 mw = Descriptors.MolWt(mol)  
 logp = Descriptors.MolLogP(mol)  
 # Tính toán thêm các đặc trưng khác  
 return mw, logp  
 else:  
 return None, None  
  
# Ví dụ sử dụng hàm  
# Example usage  
smiles = 'CC(=O)Oc1ccccc1C(=O)O'  
mw, logp = calculate\_features(smiles)  
print(f"Molecular Weight: {mw}, LogP: {logp}")

**3. Code SQL, Python (English):**

**3.1. SQL (for data extraction):**

-- SQL query to extract data for analysis  
SELECT md.chembl\_id,  
 cs.canonical\_smiles,  
 act.standard\_type,  
 act.standard\_value,  
 act.standard\_units  
FROM activities act  
JOIN molecule\_dictionary md ON act.molregno = md.molregno  
JOIN compound\_structures cs ON md.molregno = cs.molregno  
WHERE act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 AND act.standard\_value IS NOT NULL -- Add this line to filter out NULL values  
LIMIT 100;

**Lưu ý sửa lỗi SQL:** Lỗi ERROR: operator does not exist: numeric ~ unknown xảy ra do bạn đang cố gắng so sánh một cột kiểu số (act.standard\_value) với một chuỗi ('^[0-9\.]+$'). Trong PostgreSQL, bạn không cần sử dụng regular expression để kiểm tra xem một cột kiểu số có phải là số hay không. Thay vào đó, hãy đảm bảo rằng cột standard\_value không chứa giá trị NULL và có thể chuyển đổi thành kiểu số.

**3.2. Python (for data analysis and feature engineering):**

import pandas as pd  
import psycopg2  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
import numpy as np  
  
# Database connection details  
db\_host = "192.168.206.136"  
db\_name = "chembl\_35"  
db\_user = "rd"  
db\_pass = "rd"  
  
# Function to connect to PostgreSQL  
def connect\_to\_db():  
 try:  
 conn = psycopg2.connect(  
 host=db\_host,  
 database=db\_name,  
 user=db\_user,  
 password=db\_pass)  
 return conn  
 except psycopg2.Error as e:  
 print(f"Error connecting to PostgreSQL: {e}")  
 return None  
  
# Function to execute SQL query and return data as a Pandas DataFrame  
def execute\_sql\_query(conn, sql\_query):  
 try:  
 df = pd.read\_sql\_query(sql\_query, conn)  
 return df  
 except Exception as e:  
 print(f"Error executing SQL query: {e}")  
 return None  
  
# Function to calculate RDKit features  
def calculate\_rdkit\_features(smiles):  
 try:  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 mw = Descriptors.MolWt(mol)  
 logp = Descriptors.MolLogP(mol)  
 hbd = Descriptors.NumHDonors(mol)  
 hba = Descriptors.NumHAcceptors(mol)  
 # You can add more descriptors here  
 return mw, logp, hbd, hba  
 else:  
 return None, None, None, None  
 except:  
 return None, None, None, None  
  
# Main analysis script  
def main():  
 # SQL query to extract data  
 sql\_query = """  
 SELECT md.chembl\_id,  
 cs.canonical\_smiles,  
 act.standard\_type,  
 act.standard\_value,  
 act.standard\_units  
 FROM activities act  
 JOIN molecule\_dictionary md ON act.molregno = md.molregno  
 JOIN compound\_structures cs ON md.molregno = cs.molregno  
 WHERE act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 AND act.standard\_value IS NOT NULL  
 LIMIT 100;  
 """  
  
 # Connect to the database  
 conn = connect\_to\_db()  
 if conn is None:  
 return  
  
 # Execute the SQL query  
 df = execute\_sql\_query(conn, sql\_query)  
 conn.close()  
  
 if df is None:  
 return  
  
 # Print the first few rows of the DataFrame  
 print("Original Data:")  
 print(df.head())  
  
 # Calculate RDKit features  
 df[['mw', 'logp', 'hbd', 'hba']] = df['canonical\_smiles'].apply(lambda x: pd.Series(calculate\_rdkit\_features(x)))  
  
 # Handle missing values after RDKit calculation  
 df.replace([np.inf, -np.inf], np.nan, inplace=True)  
 df.dropna(inplace=True)  
  
 # Convert standard\_value to numeric  
 df['standard\_value'] = pd.to\_numeric(df['standard\_value'], errors='coerce')  
 df.dropna(subset=['standard\_value'], inplace=True)  
  
  
 # Print the DataFrame with calculated features  
 print("\nData with RDKit Features:")  
 print(df.head())  
  
 # Basic EDA (Exploratory Data Analysis)  
 print("\nDescriptive Statistics:")  
 print(df.describe())  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

**Lưu ý về phiên bản scikit-learn:** Nếu bạn gặp lỗi TypeError: mean\_squared\_error() got an unexpected keyword argument 'squared', hãy đảm bảo rằng bạn đang sử dụng phiên bản scikit-learn mới hơn. Nếu không, bạn có thể loại bỏ tham số squared=False (tham số này chỉ có trong các phiên bản mới hơn).

**4. Ví dụ code SQL và Python (Code Examples):**

Dưới đây là 5 ví dụ code SQL và Python để bạn tham khảo:

**Ví dụ 1: Trích xuất thông tin về các hợp chất có hoạt tính trên một target cụ thể (Extracting compound information with activity on a specific target):**

* **SQL:**

SELECT md.chembl\_id,  
 cs.canonical\_smiles,  
 act.standard\_value  
FROM activities act  
JOIN molecule\_dictionary md ON act.molregno = md.molregno  
JOIN compound\_structures cs ON md.molregno = cs.molregno  
WHERE act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 AND act.target\_chembl\_id = 'CHEMBL205' -- Replace with your target  
LIMIT 100;

* **Python:**

import pandas as pd  
import psycopg2  
  
def get\_compounds\_by\_target(target\_id, limit=100):  
 conn = psycopg2.connect(host=db\_host, database=db\_name, user=db\_user, password=db\_pass)  
 sql\_query = f"""  
 SELECT md.chembl\_id,  
 cs.canonical\_smiles,  
 act.standard\_value  
 FROM activities act  
 JOIN molecule\_dictionary md ON act.molregno = md.molregno  
 JOIN compound\_structures cs ON md.molregno = cs.molregno  
 WHERE act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 AND act.target\_chembl\_id = '{target\_id}'  
 LIMIT {limit};  
 """  
 df = pd.read\_sql\_query(sql\_query, conn)  
 conn.close()  
 return df  
  
target\_chembl\_id = 'CHEMBL205'  
compounds\_df = get\_compounds\_by\_target(target\_chembl\_id)  
print(compounds\_df.head())

**Ví dụ 2: Tính toán số lượng hợp chất cho mỗi target (Calculate the number of compounds for each target):**

* **SQL:**

SELECT target\_chembl\_id, COUNT(DISTINCT molregno) AS num\_compounds  
FROM activities  
WHERE standard\_type = 'IC50'  
GROUP BY target\_chembl\_id  
ORDER BY num\_compounds DESC  
LIMIT 10;

* **Python:**

import pandas as pd  
import psycopg2  
  
def count\_compounds\_per\_target(limit=10):  
 conn = psycopg2.connect(host=db\_host, database=db\_name, user=db\_user, password=db\_pass)  
 sql\_query = f"""  
 SELECT target\_chembl\_id, COUNT(DISTINCT molregno) AS num\_compounds  
 FROM activities  
 WHERE standard\_type = 'IC50'  
 GROUP BY target\_chembl\_id  
 ORDER BY num\_compounds DESC  
 LIMIT {limit};  
 """  
 df = pd.read\_sql\_query(sql\_query, conn)  
 conn.close()  
 return df  
  
target\_counts\_df = count\_compounds\_per\_target()  
print(target\_counts\_df.head())

**Ví dụ 3: Phân tích phân bố giá trị IC50 (Analyze IC50 value distribution):**

* **SQL:** (Không cần thiết, có thể thực hiện trực tiếp trong Python sau khi trích xuất dữ liệu)
* **Python:**

import pandas as pd  
import psycopg2  
import matplotlib.pyplot as plt  
import numpy as np  
  
def analyze\_ic50\_distribution(limit=100):  
 conn = psycopg2.connect(host=db\_host, database=db\_name, user=db\_user, password=db\_pass)  
 sql\_query = f"""  
 SELECT standard\_value  
 FROM activities  
 WHERE standard\_type = 'IC50'  
 AND standard\_units = 'nM'  
 AND standard\_value IS NOT NULL  
 LIMIT {limit};  
 """  
 df = pd.read\_sql\_query(sql\_query, conn)  
 conn.close()  
  
 # Convert to numeric and drop NaN values  
 df['standard\_value'] = pd.to\_numeric(df['standard\_value'], errors='coerce')  
 df.dropna(subset=['standard\_value'], inplace=True)  
  
 # Take the logarithm of IC50 values  
 df['pIC50'] = -np.log10(df['standard\_value'] / 1e9) # Convert nM to M  
  
 plt.hist(df['pIC50'], bins=30)  
 plt.xlabel('pIC50')  
 plt.ylabel('Frequency')  
 plt.title('Distribution of pIC50 Values')  
 plt.show()  
  
analyze\_ic50\_distribution()

**Ví dụ 4: Tính toán các descriptor vật lý hóa học và xem xét tương quan (Calculate physicochemical descriptors and explore correlations):**

* **SQL:** (Như ví dụ 1 để lấy SMILES)
* **Python:**

import pandas as pd  
import psycopg2  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
import matplotlib.pyplot as plt  
  
def analyze\_descriptor\_correlations(limit=100):  
 conn = psycopg2.connect(host=db\_host, database=db\_name, user=db\_user, password=db\_pass)  
 sql\_query = f"""  
 SELECT cs.canonical\_smiles  
 FROM activities act  
 JOIN molecule\_dictionary md ON act.molregno = md.molregno  
 JOIN compound\_structures cs ON md.molregno = cs.molregno  
 WHERE act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 LIMIT {limit};  
 """  
 df = pd.read\_sql\_query(sql\_query, conn)  
 conn.close()  
  
 def calculate\_descriptors(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol:  
 return Descriptors.MolWt(mol), Descriptors.MolLogP(mol)  
 return None, None  
  
 df['MW'], df['LogP'] = zip(\*df['canonical\_smiles'].apply(calculate\_descriptors))  
 df.dropna(inplace=True)  
  
 plt.scatter(df['MW'], df['LogP'])  
 plt.xlabel('Molecular Weight')  
 plt.ylabel('LogP')  
 plt.title('Correlation between MW and LogP')  
 plt.show()  
  
analyze\_descriptor\_correlations()

**Ví dụ 5: Xây dựng mô hình dự đoán hoạt tính (Build a simple activity prediction model):**

* **SQL:** (Như ví dụ 1 để lấy SMILES và IC50)
* **Python:**

import pandas as pd  
import psycopg2  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
from sklearn.model\_selection import train\_test\_split  
from sklearn.linear\_model import LinearRegression  
from sklearn.metrics import mean\_squared\_error  
import numpy as np  
  
def build\_activity\_prediction\_model(limit=100):  
 conn = psycopg2.connect(host=db\_host, database=db\_name, user=db\_user, password=db\_pass)  
 sql\_query = f"""  
 SELECT cs.canonical\_smiles, act.standard\_value  
 FROM activities act  
 JOIN molecule\_dictionary md ON act.molregno = md.molregno  
 JOIN compound\_structures cs ON md.molregno = cs.molregno  
 WHERE act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 AND act.standard\_value IS NOT NULL  
 LIMIT {limit};  
 """  
 df = pd.read\_sql\_query(sql\_query, conn)  
 conn.close()  
  
 def calculate\_descriptors(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol:  
 return Descriptors.MolWt(mol), Descriptors.MolLogP(mol)  
 return None, None  
  
 df['MW'], df['LogP'] = zip(\*df['canonical\_smiles'].apply(calculate\_descriptors))  
 df.dropna(inplace=True)  
 df['standard\_value'] = pd.to\_numeric(df['standard\_value'], errors='coerce')  
 df.dropna(subset=['standard\_value'], inplace=True)  
  
 X = df[['MW', 'LogP']]  
 y = -np.log10(df['standard\_value'] / 1e9) # pIC50  
  
 X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)  
  
 model = LinearRegression()  
 model.fit(X\_train, y\_train)  
  
 y\_pred = model.predict(X\_test)  
 mse = mean\_squared\_error(y\_test, y\_pred)  
 print(f'Mean Squared Error: {mse}')  
  
build\_activity\_prediction\_model()

**Lưu ý quan trọng:**

* **Đường dẫn (Paths):** Luôn sử dụng os.path.join(base\_path, ...) để đảm bảo tính tương thích của code trên các hệ điều hành khác nhau.
* **Xử lý lỗi (Error Handling):** Bổ sung các khối try...except để xử lý các lỗi có thể xảy ra trong quá trình kết nối cơ sở dữ liệu, tính toán đặc trưng, hoặc xây dựng mô hình.
* **Logging:** Sử dụng thư viện logging để ghi lại các thông tin quan trọng trong quá trình chạy code.
* **AIMLOps Template:** Đảm bảo rằng code của bạn tuân thủ theo cấu trúc thư mục và quy ước đặt tên file của AIMLOps template.

Tôi hy vọng những hướng dẫn và ví dụ này sẽ giúp bạn bắt đầu dự án của mình một cách thuận lợi. Nếu bạn có bất kỳ câu hỏi nào khác, đừng ngần ngại hỏi!