# PharmApp Suite

## 🧠 AI for Drug Discovery and Development 🧪

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

## 🧩 Topic: Tìm các chất ức chế COX-2 chọn lọc

Tuyệt vời! Tôi rất vui được hỗ trợ bạn trong dự án phân tích dữ liệu ChEMBL 35 này. Dựa trên thông tin bạn cung cấp và mã chủ đề “Topic\_CheMBL\_35\_92”, tôi hiểu rằng bạn muốn thực hiện một phân tích cụ thể trên dữ liệu ChEMBL 35, kết hợp với RDKit để hỗ trợ nghiên cứu và phát triển thuốc.

Dưới đây là phân tích chi tiết, hướng dẫn song ngữ, mã SQL và Python mẫu, cùng với các ví dụ để giúp bạn bắt đầu:

**1. Phân tích và Mô hình Phân tích (Analysis and Analysis Model)**

**Phân tích:**

* **Mục tiêu:** Xác định mối quan hệ giữa cấu trúc hóa học của các hợp chất (được biểu diễn bằng dữ liệu ChEMBL và RDKit) và hoạt tính sinh học của chúng (ví dụ: IC50, Ki).
* **Phương pháp:**
  + **Trích xuất Đặc trưng (Feature Extraction):** Sử dụng RDKit để tính toán các đặc trưng phân tử từ cấu trúc SMILES của các hợp chất. Các đặc trưng này có thể bao gồm trọng lượng phân tử, số lượng liên kết, số lượng vòng, độ tan, v.v.
  + **Lọc và Tiền xử lý Dữ liệu (Data Filtering and Preprocessing):** Lọc dữ liệu ChEMBL để chỉ giữ lại các hợp chất có hoạt tính sinh học đã biết đối với một mục tiêu cụ thể (ví dụ: một protein). Tiền xử lý dữ liệu để xử lý các giá trị bị thiếu, ngoại lệ và chuẩn hóa dữ liệu.
  + **Mô hình hóa (Modeling):** Sử dụng các thuật toán học máy (ví dụ: hồi quy tuyến tính, hồi quy rừng ngẫu nhiên, mạng nơ-ron) để xây dựng mô hình dự đoán hoạt tính sinh học dựa trên các đặc trưng phân tử.
  + **Đánh giá Mô hình (Model Evaluation):** Đánh giá hiệu suất của mô hình bằng cách sử dụng các chỉ số như R-squared, RMSE, MAE, v.v.

**Mô hình Phân tích:**

Mô hình phân tích có thể được biểu diễn như sau:

Cấu trúc Hóa học (SMILES) --> RDKit --> Đặc trưng Phân tử --> Học Máy --> Dự đoán Hoạt tính Sinh học

**2. Hướng dẫn Song ngữ (Bilingual Guide)**

**English:**

This section provides a bilingual guide (English and Vietnamese) to help you understand the code and concepts.

**Vietnamese:**

Phần này cung cấp một hướng dẫn song ngữ (tiếng Anh và tiếng Việt) để giúp bạn hiểu mã và các khái niệm.

**3. Mã SQL và Python (SQL and Python Code)**

**Lưu ý:** Vì bạn chỉ muốn lấy 100 dòng dữ liệu, tôi sẽ thêm mệnh đề LIMIT 100 vào các truy vấn SQL.

**a. SQL (English):**

-- SQL query to extract activity data and compound structures from ChEMBL 35  
-- Extracts 100 rows of data for demonstration purposes.  
  
SELECT  
 md.molregno,  
 cs.canonical\_smiles,  
 act.standard\_type,  
 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  
WHERE  
 act.standard\_type = 'IC50' -- Filter for IC50 values  
 AND act.standard\_units = 'nM' -- Filter for nM units  
 AND act.standard\_value IS NOT NULL -- Ensure standard\_value is not null  
 AND act.standard\_value::text ~ '^[0-9\.]+$' --Check if it is numeric  
LIMIT 100;

**SQL (Tiếng Việt):**

-- Truy vấn SQL để trích xuất dữ liệu hoạt tính và cấu trúc hợp chất từ ChEMBL 35  
-- Trích xuất 100 dòng dữ liệu cho mục đích minh họa.  
  
SELECT  
 md.molregno,  
 cs.canonical\_smiles,  
 act.standard\_type,  
 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  
WHERE  
 act.standard\_type = 'IC50' -- Lọc theo giá trị IC50  
 AND act.standard\_units = 'nM' -- Lọc theo đơn vị nM  
 AND act.standard\_value IS NOT NULL -- Đảm bảo standard\_value không null  
 AND act.standard\_value::text ~ '^[0-9\.]+$' --Kiểm tra xem nó có phải là số không  
LIMIT 100;

**b. Python (English):**

import os  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
import psycopg2  
from sklearn.model\_selection import train\_test\_split  
from sklearn.linear\_model import LinearRegression  
from sklearn.metrics import mean\_squared\_error, r2\_score  
  
# Database connection details  
db\_host = "192.168.206.136"  
db\_user = "rd"  
db\_pass = "rd"  
db\_name = "chembl\_35"  
  
# Base path for the project  
base\_path = "." # Assuming the notebook is in the root directory  
  
# SQL query (same as above)  
sql\_query = """  
SELECT  
 md.molregno,  
 cs.canonical\_smiles,  
 act.standard\_type,  
 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  
WHERE  
 act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 AND act.standard\_value IS NOT NULL  
 AND act.standard\_value::text ~ '^[0-9\.]+$'  
LIMIT 100;  
"""  
  
# Function to connect to the database and execute the query  
def fetch\_data\_from\_chembl(host, user, password, database, query):  
 conn = None  
 try:  
 conn = psycopg2.connect(host=host, user=user, password=password, database=database)  
 df = pd.read\_sql\_query(query, conn)  
 return df  
 except psycopg2.Error as e:  
 print(f"Error connecting to the database: {e}")  
 return None  
 finally:  
 if conn:  
 conn.close()  
  
# Fetch data  
df = fetch\_data\_from\_chembl(db\_host, db\_user, db\_pass, db\_name, sql\_query)  
  
if df is not None:  
 print("Data fetched successfully.")  
 print(df.head())  
  
 # RDKit feature extraction function  
 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 pd.Series(descriptors)  
  
 # Apply descriptor calculation  
 df = pd.concat([df, df['canonical\_smiles'].apply(calculate\_descriptors)], axis=1)  
  
 # Data cleaning: remove rows with NaN values resulted from descriptor calculation  
 df = df.dropna()  
  
 # Prepare data for modeling  
 X = df[['MolWt', 'LogP', 'HBD', 'HBA']]  
 y = df['standard\_value'].astype(float) # Ensure y is numeric  
  
 # 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)  
  
 # Model training  
 model = LinearRegression()  
 model.fit(X\_train, y\_train)  
  
 # Model prediction  
 y\_pred = model.predict(X\_test)  
  
 # Model evaluation  
 mse = mean\_squared\_error(y\_test, y\_pred)  
 r2 = r2\_score(y\_test, y\_pred)  
  
 print(f"Mean Squared Error: {mse}")  
 print(f"R-squared: {r2}")  
else:  
 print("Failed to fetch data.")

**Python (Tiếng Việt):**

import os  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
import psycopg2  
from sklearn.model\_selection import train\_test\_split  
from sklearn.linear\_model import LinearRegression  
from sklearn.metrics import mean\_squared\_error, r2\_score  
  
# Thông tin kết nối cơ sở dữ liệu  
db\_host = "192.168.206.136"  
db\_user = "rd"  
db\_pass = "rd"  
db\_name = "chembl\_35"  
  
# Đường dẫn cơ sở cho dự án  
base\_path = "." # Giả sử notebook nằm trong thư mục gốc  
  
# Truy vấn SQL (giống như trên)  
sql\_query = """  
SELECT  
 md.molregno,  
 cs.canonical\_smiles,  
 act.standard\_type,  
 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  
WHERE  
 act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 AND act.standard\_value IS NOT NULL  
 AND act.standard\_value::text ~ '^[0-9\.]+$'  
LIMIT 100;  
"""  
  
# Hàm kết nối đến cơ sở dữ liệu và thực hiện truy vấn  
def fetch\_data\_from\_chembl(host, user, password, database, query):  
 conn = None  
 try:  
 conn = psycopg2.connect(host=host, user=user, password=password, database=database)  
 df = pd.read\_sql\_query(query, conn)  
 return df  
 except psycopg2.Error as e:  
 print(f"Lỗi kết nối đến cơ sở dữ liệu: {e}")  
 return None  
 finally:  
 if conn:  
 conn.close()  
  
# Lấy dữ liệu  
df = fetch\_data\_from\_chembl(db\_host, db\_user, db\_pass, db\_name, sql\_query)  
  
if df is not None:  
 print("Lấy dữ liệu thành công.")  
 print(df.head())  
  
 # Hàm trích xuất đặc trưng 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 pd.Series(descriptors)  
  
 # Áp dụng tính toán descriptor  
 df = pd.concat([df, df['canonical\_smiles'].apply(calculate\_descriptors)], axis=1)  
  
 # Làm sạch dữ liệu: loại bỏ các hàng có giá trị NaN do tính toán descriptor  
 df = df.dropna()  
  
 # Chuẩn bị dữ liệu cho mô hình hóa  
 X = df[['MolWt', 'LogP', 'HBD', 'HBA']]  
 y = df['standard\_value'].astype(float) # Đảm bảo y là số  
  
 # Chia dữ liệu thành tập huấn luyện và tập kiểm tra  
 X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)  
  
 # Huấn luyện mô hình  
 model = LinearRegression()  
 model.fit(X\_train, y\_train)  
  
 # Dự đoán mô hình  
 y\_pred = model.predict(X\_test)  
  
 # Đánh giá mô hình  
 mse = mean\_squared\_error(y\_test, y\_pred)  
 r2 = r2\_score(y\_test, y\_pred)  
  
 print(f"Lỗi bình phương trung bình: {mse}")  
 print(f"R-squared: {r2}")  
else:  
 print("Không thể lấy dữ liệu.")

**Giải thích Code:**

1. **Kết nối CSDL (Database Connection):** Sử dụng psycopg2 để kết nối đến CSDL PostgreSQL ChEMBL 35.
2. **Truy vấn SQL (SQL Query):** Truy vấn dữ liệu từ các bảng molecule\_dictionary, compound\_structures, và activities. Lọc dữ liệu theo standard\_type = 'IC50' và standard\_units = 'nM'.
3. **Trích xuất Đặc trưng (Feature Extraction):** Sử dụng RDKit để tính toán các đặc trưng phân tử như MolWt, LogP, HBD, và HBA từ cấu trúc SMILES.
4. **Mô hình hóa (Modeling):** Sử dụng LinearRegression từ sklearn để xây dựng mô hình dự đoán hoạt tính sinh học.
5. **Đánh giá (Evaluation):** Đánh giá mô hình bằng cách sử dụng mean\_squared\_error và r2\_score.

**4. Ví dụ Mã SQL và Python Mẫu (Example SQL and Python Code)**

Dưới đây là 5 ví dụ về các truy vấn SQL và mã Python khác nhau mà bạn có thể sử dụng để phân tích dữ liệu ChEMBL 35:

**Ví dụ 1:**

**SQL (English):**

-- Get compounds with specific substructure  
SELECT md.molregno, cs.canonical\_smiles  
FROM molecule\_dictionary md  
JOIN compound\_structures cs ON md.molregno = cs.molregno  
WHERE cs.canonical\_smiles LIKE '%[cO][cO]%'  
LIMIT 100;

**SQL (Tiếng Việt):**

-- Lấy các hợp chất có cấu trúc con cụ thể  
SELECT md.molregno, cs.canonical\_smiles  
FROM molecule\_dictionary md  
JOIN compound\_structures cs ON md.molregno = cs.molregno  
WHERE cs.canonical\_smiles LIKE '%[cO][cO]%'  
LIMIT 100;

**Python (English):**

# Calculate more descriptors  
def calculate\_more\_descriptors(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is None:  
 return None  
 descriptors = {}  
 descriptors['NumRotatableBonds'] = Descriptors.NumRotatableBonds(mol)  
 descriptors['TPSA'] = Descriptors.TPSA(mol)  
 return pd.Series(descriptors)

**Python (Tiếng Việt):**

# Tính toán thêm các descriptor  
def calculate\_more\_descriptors(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is None:  
 return None  
 descriptors = {}  
 descriptors['NumRotatableBonds'] = Descriptors.NumRotatableBonds(mol)  
 descriptors['TPSA'] = Descriptors.TPSA(mol)  
 return pd.Series(descriptors)

**Ví dụ 2:**

**SQL (English):**

-- Find compounds with high molecular weight  
SELECT md.molregno, cs.canonical\_smiles  
FROM molecule\_dictionary md  
JOIN compound\_structures cs ON md.molregno = cs.molregno  
WHERE md.molregno IN (SELECT molregno FROM compound\_properties WHERE mw >= 500)  
LIMIT 100;

**SQL (Tiếng Việt):**

-- Tìm các hợp chất có trọng lượng phân tử cao  
SELECT md.molregno, cs.canonical\_smiles  
FROM molecule\_dictionary md  
JOIN compound\_structures cs ON md.molregno = cs.molregno  
WHERE md.molregno IN (SELECT molregno FROM compound\_properties WHERE mw >= 500)  
LIMIT 100;

**Python (English):**

# Use a different model  
from sklearn.ensemble import RandomForestRegressor  
model = RandomForestRegressor(n\_estimators=100, random\_state=42)  
model.fit(X\_train, y\_train)

**Python (Tiếng Việt):**

# Sử dụng một mô hình khác  
from sklearn.ensemble import RandomForestRegressor  
model = RandomForestRegressor(n\_estimators=100, random\_state=42)  
model.fit(X\_train, y\_train)

**Ví dụ 3:**

**SQL (English):**

-- Extract compounds targeting a specific protein  
SELECT md.molregno, 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  
JOIN target\_dictionary td ON act.tid = td.tid  
WHERE td.pref\_name = 'Acetylcholinesterase'  
LIMIT 100;

**SQL (Tiếng Việt):**

-- Trích xuất các hợp chất nhắm mục tiêu một protein cụ thể  
SELECT md.molregno, 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  
JOIN target\_dictionary td ON act.tid = td.tid  
WHERE td.pref\_name = 'Acetylcholinesterase'  
LIMIT 100;

**Python (English):**

# Visualize the results  
import matplotlib.pyplot as plt  
plt.scatter(y\_test, y\_pred)  
plt.xlabel("Actual Values")  
plt.ylabel("Predicted Values")  
plt.show()

**Python (Tiếng Việt):**

# Trực quan hóa kết quả  
import matplotlib.pyplot as plt  
plt.scatter(y\_test, y\_pred)  
plt.xlabel("Giá trị thực tế")  
plt.ylabel("Giá trị dự đoán")  
plt.show()

**Ví dụ 4:**

**SQL (English):**

-- Find compounds with activity against a specific target family  
SELECT md.molregno, 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  
JOIN target\_dictionary td ON act.tid = td.tid  
JOIN target\_components tc ON td.tid = tc.tid  
JOIN component\_sequences cseq ON tc.component\_id = cseq.component\_id  
WHERE cseq.sequence LIKE '%GPLGS%'  
LIMIT 100;

**SQL (Tiếng Việt):**

-- Tìm các hợp chất có hoạt tính chống lại một họ mục tiêu cụ thể  
SELECT md.molregno, 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  
JOIN target\_dictionary td ON act.tid = td.tid  
JOIN target\_components tc ON td.tid = tc.tid  
JOIN component\_sequences cseq ON tc.component\_id = cseq.component\_id  
WHERE cseq.sequence LIKE '%GPLGS%'  
LIMIT 100;

**Python (English):**

# Feature Scaling  
from sklearn.preprocessing import StandardScaler  
scaler = StandardScaler()  
X\_train = scaler.fit\_transform(X\_train)  
X\_test = scaler.transform(X\_test)

**Python (Tiếng Việt):**

# Tỉ lệ hóa đặc trưng  
from sklearn.preprocessing import StandardScaler  
scaler = StandardScaler()  
X\_train = scaler.fit\_transform(X\_train)  
X\_test = scaler.transform(X\_test)

**Ví dụ 5:**

**SQL (English):**

-- Extract compounds with specific physicochemical properties  
SELECT md.molregno, cs.canonical\_smiles, cp.logp, cp.psa  
FROM molecule\_dictionary md  
JOIN compound\_structures cs ON md.molregno = cs.molregno  
JOIN compound\_properties cp ON md.molregno = cp.molregno  
WHERE cp.logp BETWEEN 2 AND 4 AND cp.psa < 100  
LIMIT 100;

**SQL (Tiếng Việt):**

-- Trích xuất các hợp chất có tính chất lý hóa cụ thể  
SELECT md.molregno, cs.canonical\_smiles, cp.logp, cp.psa  
FROM molecule\_dictionary md  
JOIN compound\_structures cs ON md.molregno = cs.molregno  
JOIN compound\_properties cp ON md.molregno = cp.molregno  
WHERE cp.logp BETWEEN 2 AND 4 AND cp.psa < 100  
LIMIT 100;

**Python (English):**

# Hyperparameter Tuning  
from sklearn.model\_selection import GridSearchCV  
param\_grid = {'n\_estimators': [50, 100, 200], 'max\_depth': [4, 6, 8]}  
grid\_search = GridSearchCV(RandomForestRegressor(random\_state=42), param\_grid, cv=3, scoring='neg\_mean\_squared\_error')  
grid\_search.fit(X\_train, y\_train)  
best\_model = grid\_search.best\_estimator\_

**Python (Tiếng Việt):**

# Điều chỉnh siêu tham số  
from sklearn.model\_selection import GridSearchCV  
param\_grid = {'n\_estimators': [50, 100, 200], 'max\_depth': [4, 6, 8]}  
grid\_search = GridSearchCV(RandomForestRegressor(random\_state=42), param\_grid, cv=3, scoring='neg\_mean\_squared\_error')  
grid\_search.fit(X\_train, y\_train)  
best\_model = grid\_search.best\_estimator\_

**5. Giải quyết Lỗi (Error Resolution)**

* **Lỗi SQL (SQL Error):**
* ERROR: operator does not exist: numeric ~ unknown, LINE 12: AND act.standard\_value ~ '^[0-9\.]+$'
* **Giải thích (Explanation):** Lỗi này xảy ra vì PostgreSQL không thể sử dụng toán tử ~ (regular expression match) trực tiếp trên cột kiểu numeric. **Cách sửa (Solution):** Chuyển đổi cột standard\_value sang kiểu text trước khi sử dụng toán tử ~. Tôi đã sửa trong code SQL ở trên bằng cách sử dụng act.standard\_value::text ~ '^[0-9\.]+$'
* **Lỗi Python (Python Error):**
* phiên bản scikit-learn cũ không hỗ trợ tham số squared=False trong hàm mean\_squared\_error
* **Giải thích (Explanation):** Tham số squared=False trong hàm mean\_squared\_error chỉ được hỗ trợ trong các phiên bản scikit-learn mới hơn. **Cách sửa (Solution):** Loại bỏ tham số squared=False nếu bạn đang sử dụng phiên bản scikit-learn cũ. Nếu bạn muốn tính Root Mean Squared Error (RMSE), bạn có thể lấy căn bậc hai của MSE:
* mse = mean\_squared\_error(y\_test, y\_pred)  
  rmse = mse\*\*0.5 # Calculate RMSE  
  print(f"Root Mean Squared Error: {rmse}")

**Lưu ý Quan trọng:**

* **Cấu trúc thư mục:** Đảm bảo bạn tuân thủ cấu trúc thư mục AIMLOps Template. Sử dụng os.path.join(base\_path, ...) để xây dựng đường dẫn một cách chính xác.
* **Phiên bản thư viện:** Kiểm tra và cập nhật các thư viện Python (RDKit, scikit-learn, pandas, psycopg2) lên phiên bản mới nhất để tránh các lỗi không tương thích.
* **Hiệu suất:** Vì bạn chỉ lấy 100 dòng dữ liệu, hiệu suất không phải là vấn đề lớn. Tuy nhiên, khi bạn làm việc với toàn bộ tập dữ liệu ChEMBL, hãy cân nhắc sử dụng các kỹ thuật tối ưu hóa như lập chỉ mục CSDL, xử lý song song, v.v.

Hy vọng những hướng dẫn và ví dụ này sẽ giúp bạn bắt đầu dự án của mình một cách suôn sẻ! Nếu bạn có bất kỳ câu hỏi nào khác, đừng ngần ngại hỏi. Chúc bạn thành công!