# PharmApp Suite

## 🧠 AI for Drug Discovery and Development 🧪

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

## 🧩 Topic: Hồ sơ thuốc bị rút khỏi thị trường

Tuyệt vời! Với vai trò là chuyên gia nghiên cứu và phát triển dược học, tôi sẽ giúp bạn phân tích và xây dựng quy trình làm việc cho dự án phân tích dữ liệu ChEMBL 35 sử dụng RDKit, tuân theo cấu trúc AIMLOps và giải quyết các vấn đề bạn đã nêu.

**Tổng quan dự án:**

Dự án này tập trung vào việc khai thác dữ liệu từ cơ sở dữ liệu ChEMBL 35 để hỗ trợ quá trình nghiên cứu và phát triển thuốc. Chúng ta sẽ sử dụng SQL để truy vấn dữ liệu từ PostgreSQL, sau đó sử dụng Python và RDKit để tiền xử lý, phân tích và xây dựng mô hình.

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

Chúng ta sẽ tập trung vào việc xây dựng một quy trình phân tích dữ liệu, bao gồm các bước sau:

* **Data Extraction (Trích xuất dữ liệu):** Sử dụng SQL để truy vấn dữ liệu liên quan đến hoạt tính sinh học của các hợp chất. Chúng ta sẽ tập trung vào các hợp chất có giá trị IC50, Ki, Kd, EC50.
* **Data Cleaning and Preprocessing (Làm sạch và tiền xử lý dữ liệu):** Loại bỏ các giá trị không hợp lệ, chuyển đổi đơn vị, và xử lý các giá trị ngoại lệ.
* **Feature Engineering (Xây dựng đặc trưng):** Sử dụng RDKit để tính toán các đặc trưng hóa học của các hợp chất, chẳng hạn như trọng lượng phân tử, logP, số lượng liên kết quay, và các descriptor khác.
* **Data Analysis and Modeling (Phân tích dữ liệu và xây dựng mô hình):** Sử dụng các kỹ thuật thống kê và học máy để phân tích mối quan hệ giữa các đặc trưng hóa học và hoạt tính sinh học. Chúng ta có thể sử dụng các mô hình như hồi quy tuyến tính, hồi quy logistic, Support Vector Machines, hoặc Random Forests.
* **Visualization (Trực quan hóa):** Sử dụng các biểu đồ và đồ thị để trực quan hóa dữ liệu và kết quả phân tích.

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

**English:**

We will focus on building a data analysis pipeline that includes the following steps:

* **Data Extraction:** Use SQL to query data related to the biological activity of compounds. We will focus on compounds with IC50, Ki, Kd, and EC50 values.
* **Data Cleaning and Preprocessing:** Remove invalid values, convert units, and handle outliers.
* **Feature Engineering:** Use RDKit to calculate chemical features of the compounds, such as molecular weight, logP, number of rotatable bonds, and other descriptors.
* **Data Analysis and Modeling:** Use statistical and machine learning techniques to analyze the relationship between chemical features and biological activity. We can use models such as linear regression, logistic regression, Support Vector Machines, or Random Forests.
* **Visualization:** Use charts and graphs to visualize data and analysis results.

**Tiếng Việt:**

Chúng ta sẽ tập trung vào việc xây dựng một quy trình phân tích dữ liệu bao gồm các bước sau:

* **Trích xuất dữ liệu:** Sử dụng SQL để truy vấn dữ liệu liên quan đến hoạt tính sinh học của các hợp chất. Chúng ta sẽ tập trung vào các hợp chất có giá trị IC50, Ki, Kd và EC50.
* **Làm sạch và tiền xử lý dữ liệu:** Loại bỏ các giá trị không hợp lệ, chuyển đổi đơn vị và xử lý các giá trị ngoại lệ.
* **Xây dựng đặc trưng:** Sử dụng RDKit để tính toán các đặc trưng hóa học của các hợp chất, chẳng hạn như trọng lượng phân tử, logP, số lượng liên kết quay và các descriptor khác.
* **Phân tích dữ liệu và xây dựng mô hình:** Sử dụng các kỹ thuật thống kê và học máy để phân tích mối quan hệ giữa các đặc trưng hóa học và hoạt tính sinh học. Chúng ta có thể sử dụng các mô hình như hồi quy tuyến tính, hồi quy logistic, Support Vector Machines hoặc Random Forests.
* **Trực quan hóa:** Sử dụng các biểu đồ và đồ thị để trực quan hóa dữ liệu và kết quả phân tích.

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

**SQL (Extracting Data):**

-- Extracting target, molecule, and activity data for a specific target  
SELECT  
 act.standard\_value,  
 act.standard\_units,  
 act.standard\_type,  
 md.molecule\_structures,  
 td.pref\_name  
FROM  
 activities act  
JOIN  
 molecule\_dictionary md ON act.molregno = md.molregno  
JOIN  
 target\_dictionary td ON act.tid = td.tid  
WHERE  
 td.pref\_name = 'CHEMBL target name' -- Replace with your target of interest  
 AND act.standard\_type = 'IC50' -- Focus on IC50 values  
 AND act.standard\_value IS NOT NULL  
 -- This fix the issue error operator does not exist: numeric ~ unknown  
 AND act.standard\_value::text ~ '^[0-9\.]+$'  
LIMIT 100;

**Python (Data Processing and Feature Engineering):**

import os  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
import numpy as np  
  
# Define base path  
base\_path = "." # Assuming the notebook is in the root directory  
  
# Define data path  
data\_path = os.path.join(base\_path, "data")  
  
# Load data from CSV  
# Assuming the CSV file is named 'chembl\_data.csv'  
try:  
 df = pd.read\_csv(os.path.join(data\_path, "chembl\_data.csv"))  
except FileNotFoundError:  
 print(f"Error: File not found at {os.path.join(data\_path, 'chembl\_data.csv')}")  
 exit()  
  
# Function to calculate molecular weight  
def calculate\_mw(smiles):  
 try:  
 mol = Chem.MolFromSmiles(smiles)  
 if mol:  
 return Descriptors.MolWt(mol)  
 else:  
 return np.nan  
 except:  
 return np.nan  
  
# Apply the function to the 'molecule\_structures' column  
df['molecular\_weight'] = df['molecule\_structures'].apply(calculate\_mw)  
  
# Print the first 5 rows of the dataframe with the new feature  
print(df.head())

**4. Ví dụ code SQL, Python (English):**

**SQL Examples:**

1. **Get all compounds with IC50 values against a specific target:**

SELECT md.chembl\_id, act.standard\_value  
FROM activities act  
JOIN target\_dictionary td ON act.tid = td.tid  
JOIN molecule\_dictionary md ON act.molregno = md.molregno  
WHERE td.pref\_name = 'Acetylcholinesterase' AND act.standard\_type = 'IC50'  
LIMIT 100;

1. **Get compounds with IC50 < 1000 nM:**

SELECT md.chembl\_id, act.standard\_value  
FROM activities act  
JOIN target\_dictionary td ON act.tid = td.tid  
JOIN molecule\_dictionary md ON act.molregno = md.molregno  
WHERE td.pref\_name = 'Acetylcholinesterase' AND act.standard\_type = 'IC50' AND act.standard\_value < 1000  
LIMIT 100;

1. **Count the number of compounds for each target:**

SELECT td.pref\_name, COUNT(DISTINCT md.molregno)  
FROM activities act  
JOIN target\_dictionary td ON act.tid = td.tid  
JOIN molecule\_dictionary md ON act.molregno = md.molregno  
WHERE act.standard\_type = 'IC50'  
GROUP BY td.pref\_name  
ORDER BY COUNT(DISTINCT md.molregno) DESC  
LIMIT 100;

1. **Find the most potent compound (lowest IC50) for a target:**

SELECT md.chembl\_id, act.standard\_value  
FROM activities act  
JOIN target\_dictionary td ON act.tid = td.tid  
JOIN molecule\_dictionary md ON act.molregno = md.molregno  
WHERE td.pref\_name = 'Acetylcholinesterase' AND act.standard\_type = 'IC50'  
ORDER BY act.standard\_value ASC  
LIMIT 100;

1. **Get SMILES and IC50 values for a target:**

SELECT md.chembl\_id, act.standard\_value, ms.molecule\_structures  
FROM activities act  
JOIN target\_dictionary td ON act.tid = td.tid  
JOIN molecule\_dictionary md ON act.molregno = md.molregno  
JOIN molecule\_structures ms ON md.molregno = ms.molregno  
WHERE td.pref\_name = 'Acetylcholinesterase' AND act.standard\_type = 'IC50'  
LIMIT 100;

**Python Examples:**

1. **Calculate LogP for a molecule:**

from rdkit import Chem  
from rdkit.Chem import AllChem  
from rdkit.Chem import Crippen  
  
def calculate\_logp(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol:  
 return Crippen.MolLogP(mol)  
 else:  
 return None  
  
smiles = 'CC(=O)OC1=CC=CC=C1C(=O)O' # Aspirin  
logp = calculate\_logp(smiles)  
print(f"LogP for Aspirin: {logp}")

1. **Calculate TPSA (Topological Polar Surface Area):**

from rdkit import Chem  
from rdkit.Chem import Descriptors  
  
def calculate\_tpsa(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol:  
 return Descriptors.TPSA(mol)  
 else:  
 return None  
  
smiles = 'CC(=O)OC1=CC=CC=C1C(=O)O' # Aspirin  
tpsa = calculate\_tpsa(smiles)  
print(f"TPSA for Aspirin: {tpsa}")

1. **Calculate number of rotatable bonds:**

from rdkit import Chem  
from rdkit.Chem import Descriptors  
  
def calculate\_rotatable\_bonds(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol:  
 return Descriptors.NumRotatableBonds(mol)  
 else:  
 return None  
  
smiles = 'CC(=O)OC1=CC=CC=C1C(=O)O' # Aspirin  
rotatable\_bonds = calculate\_rotatable\_bonds(smiles)  
print(f"Rotatable bonds for Aspirin: {rotatable\_bonds}")

1. **Convert SMILES to Mol object and visualize:**

from rdkit import Chem  
from rdkit.Chem.Draw import IPythonConsole  
from rdkit.Chem import Draw  
  
smiles = 'CC(=O)OC1=CC=CC=C1C(=O)O'  
mol = Chem.MolFromSmiles(smiles)  
  
if mol:  
 img = Draw.MolToImage(mol)  
 img.save("aspirin.png") # Save the image to a file  
 print("Aspirin image saved to aspirin.png")  
else:  
 print("Could not create Mol object from SMILES string")

1. **Calculate multiple descriptors and store in a dictionary:**

from rdkit import Chem  
from rdkit.Chem import Descriptors  
  
def calculate\_descriptors(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol:  
 descriptors = {  
 'MolecularWeight': Descriptors.MolWt(mol),  
 'LogP': Crippen.MolLogP(mol),  
 'TPSA': Descriptors.TPSA(mol),  
 'RotatableBonds': Descriptors.NumRotatableBonds(mol)  
 }  
 return descriptors  
 else:  
 return None  
  
smiles = 'CC(=O)OC1=CC=CC=C1C(=O)O' # Aspirin  
descriptors = calculate\_descriptors(smiles)  
print(f"Descriptors for Aspirin: {descriptors}")

**Lưu ý về các lỗi đã đề cập:**

* **ERROR: operator does not exist: numeric ~ unknown:**
  + **Nguyên nhân:** Lỗi này xảy ra do bạn đang cố gắng so sánh một trường số (numeric) với một chuỗi (unknown) bằng toán tử ~ (regex match).
  + **Giải pháp:** Ép kiểu cột act.standard\_value sang kiểu text trước khi so sánh bằng cách sử dụng act.standard\_value::text ~ '^[0-9\.]+$'.
* **Phiên bản Scikit-learn cũ:**
  + **Giải pháp:** Nếu bạn không thể nâng cấp Scikit-learn, hãy loại bỏ tham số squared=False khỏi hàm mean\_squared\_error. Hoặc bạn có thể tính căn bậc hai của kết quả để có được RMSE (Root Mean Squared Error).

**Cấu trúc thư mục AIMLOps:**

Đảm bảo rằng bạn tuân thủ cấu trúc thư mục AIMLOps để dễ dàng quản lý và triển khai dự án. Ví dụ:

Topic\_CheMBL\_35\_51/  
├── data/  
│ └── chembl\_data.csv  
├── notebooks/  
│ ├── Topic\_CheMBL\_35\_51\_1\_data\_extraction.ipynb  
│ ├── Topic\_CheMBL\_35\_51\_2\_feature\_engineering.ipynb  
├── models/  
├── scripts/  
└── README.md

**Lời khuyên:**

* Bắt đầu với một tập dữ liệu nhỏ (100 dòng) để thử nghiệm và gỡ lỗi.
* Sử dụng các hàm và lớp để tổ chức code của bạn.
* Viết tài liệu rõ ràng cho code của bạn.
* Sử dụng hệ thống kiểm soát phiên bản (ví dụ: Git) để theo dõi các thay đổi của bạn.
* Chia sẻ kết quả của bạn với đồng nghiệp để nhận phản hồi.

Chúc bạn thành công với dự án của mình! Hãy cho tôi biết nếu bạn có bất kỳ câu hỏi nào khác.