# PharmApp Suite

## 🧠 AI for Drug Discovery and Development 🧪

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

## 🧩 Topic: Xu hướng phê duyệt theo nhóm target

Tuyệt vời! Tôi rất vui được tham gia vào dự án thú vị này với vai trò chuyên gia R&D dược học. Dựa trên thông tin bạn cung cấp, chúng ta sẽ tập trung vào chủ đề “Topic\_CheMBL\_35\_50” và xây dựng quy trình phân tích dữ liệu từ Chembl 35 kết hợp RDKit, tuân thủ theo template AIMLOps của bạn.

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

Mục tiêu của chủ đề “Topic\_CheMBL\_35\_50” có thể là khám phá mối quan hệ giữa cấu trúc hóa học của các hợp chất (từ Chembl 35) và hoạt tính sinh học của chúng (IC50, Ki, v.v.). Chúng ta sẽ sử dụng RDKit để tạo các descriptor (đặc trưng) cho cấu trúc phân tử, sau đó sử dụng các mô hình học máy để dự đoán hoạt tính.

**Các bước chính:**

1. **Data Extraction (Trích xuất dữ liệu):** Sử dụng SQL để trích xuất thông tin từ database Chembl 35, bao gồm cấu trúc phân tử (ví dụ: SMILES) và dữ liệu hoạt tính (ví dụ: IC50).
2. **Data Preprocessing (Tiền xử lý dữ liệu):**
   * Chuyển đổi SMILES thành đối tượng phân tử RDKit.
   * Tính toán các descriptor phân tử (ví dụ: MW, logP, HBA, HBD, TPSA).
   * Xử lý các giá trị hoạt tính (ví dụ: chuyển đổi IC50 sang pIC50).
   * Loại bỏ các giá trị ngoại lệ hoặc thiếu.
3. **Feature Selection/Engineering (Chọn lọc/Xây dựng đặc trưng):** Chọn các descriptor phù hợp nhất cho mô hình hoặc tạo các đặc trưng mới từ các descriptor hiện có.
4. **Model Building (Xây dựng mô hình):** Sử dụng các thuật toán học máy (ví dụ: Linear Regression, Random Forest, Support Vector Machines) để xây dựng mô hình dự đoán hoạt tính dựa trên các descriptor phân tử.
5. **Model Evaluation (Đánh giá mô hình):** Đánh giá hiệu suất của mô hình bằng cách sử dụng các metric phù hợp (ví dụ: R-squared, RMSE, MAE).
6. **Model Interpretation (Giải thích mô hình):** Tìm hiểu các descriptor nào có ảnh hưởng lớn nhất đến hoạt tính.

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

**2.1. Data Extraction (Trích xuất dữ liệu)**

**English:**

We will use SQL to extract data from the Chembl 35 database. Here’s an example query to retrieve compound structures (SMILES) and activity data (IC50) for a specific target:

-- SQL query to extract data from Chembl  
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  
JOIN  
 target\_dictionary td ON act.tid = td.tid  
WHERE  
 td.pref\_name = 'Target Name' -- Replace with your target of interest  
 AND act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 AND act.standard\_value IS NOT NULL  
 AND act.standard\_value > 0 -- Exclude invalid values  
LIMIT 100;

**Tiếng Việt:**

Chúng ta sẽ sử dụng SQL để trích xuất dữ liệu từ cơ sở dữ liệu Chembl 35. Dưới đây là một truy vấn mẫu để lấy cấu trúc hợp chất (SMILES) và dữ liệu hoạt tính (IC50) cho một mục tiêu cụ thể:

-- Truy vấn SQL để trích xuất dữ liệu từ Chembl  
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  
JOIN  
 target\_dictionary td ON act.tid = td.tid  
WHERE  
 td.pref\_name = 'Tên mục tiêu' -- Thay thế bằng mục tiêu bạn quan tâm  
 AND act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 AND act.standard\_value IS NOT NULL  
 AND act.standard\_value > 0 -- Loại trừ các giá trị không hợp lệ  
LIMIT 100;

**Explanation:**

* molecule\_dictionary: Contains general information about molecules.
* compound\_structures: Contains the structure of the compounds (SMILES).
* activities: Contains activity data (IC50, Ki, etc.).
* target\_dictionary: Contains information about the targets (proteins).
* LIMIT 100: giới hạn 100 dòng dữ liệu

**2.2. Data Preprocessing and Feature Calculation (Tiền xử lý dữ liệu và tính toán đặc trưng)**

**English:**

import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
import numpy as np  
import os  
  
# Define base path  
base\_path = "../data" # Adjust this path according to your AIMLOps template  
  
# Load data from CSV file  
file\_path = os.path.join(base\_path, "chembl\_data.csv") # Replace with your actual CSV file name  
df = pd.read\_csv(file\_path)  
  
# Function to calculate molecular descriptors  
def calculate\_descriptors(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is None:  
 return None  
 descriptors = {desc[0]: desc[1](mol) for desc in Descriptors.descList}  
 return descriptors  
  
# Apply the function to each SMILES string  
df['descriptors'] = df['canonical\_smiles'].apply(calculate\_descriptors)  
  
# Convert descriptors to columns  
df = pd.concat([df.drop(['descriptors'], axis=1), df['descriptors'].apply(pd.Series)], axis=1)  
  
# Handle missing values  
df = df.dropna()  
  
# Convert IC50 to pIC50  
df['pIC50'] = -np.log10(df['standard\_value'] \* 1e-9) # Convert nM to M and then to pIC50  
  
# Display the first few rows of the processed data  
print(df.head())

**Tiếng Việt:**

import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
import numpy as np  
import os  
  
# Định nghĩa đường dẫn cơ sở  
base\_path = "../data" # Điều chỉnh đường dẫn này theo template AIMLOps của bạn  
  
# Tải dữ liệu từ file CSV  
file\_path = os.path.join(base\_path, "chembl\_data.csv") # Thay thế bằng tên file CSV thực tế của bạn  
df = pd.read\_csv(file\_path)  
  
# Hàm tính toán các descriptor phân tử  
def calculate\_descriptors(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is None:  
 return None  
 descriptors = {desc[0]: desc[1](mol) for desc in Descriptors.descList}  
 return descriptors  
  
# Áp dụng hàm cho mỗi chuỗi SMILES  
df['descriptors'] = df['canonical\_smiles'].apply(calculate\_descriptors)  
  
# Chuyển đổi descriptor thành các cột  
df = pd.concat([df.drop(['descriptors'], axis=1), df['descriptors'].apply(pd.Series)], axis=1)  
  
# Xử lý các giá trị thiếu  
df = df.dropna()  
  
# Chuyển đổi IC50 thành pIC50  
df['pIC50'] = -np.log10(df['standard\_value'] \* 1e-9) # Chuyển đổi nM sang M và sau đó sang pIC50  
  
# Hiển thị một vài hàng đầu tiên của dữ liệu đã xử lý  
print(df.head())

**Explanation:**

* The code reads the CSV file (obtained from the SQL query) into a Pandas DataFrame.
* It uses RDKit to convert SMILES strings into molecular objects and calculates a set of molecular descriptors.
* It handles missing values and converts IC50 values to pIC50 values.

**2.3. Model Building and Evaluation (Xây dựng và đánh giá mô hình)**

**English:**

from sklearn.model\_selection import train\_test\_split  
from sklearn.linear\_model import LinearRegression  
from sklearn.metrics import mean\_squared\_error, r2\_score  
from sklearn.preprocessing import StandardScaler  
  
# Prepare the data  
X = df.drop(['molregno', 'canonical\_smiles', 'standard\_type', 'standard\_value', 'standard\_units', 'pIC50'], axis=1) # Drop non-descriptor columns  
y = df['pIC50']  
  
# Scale the features  
scaler = StandardScaler()  
X\_scaled = scaler.fit\_transform(X)  
  
# Split the data into training and testing sets  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42)  
  
# Train the model (Linear Regression)  
model = LinearRegression()  
model.fit(X\_train, y\_train)  
  
# Make predictions  
y\_pred = model.predict(X\_test)  
  
# Evaluate the model  
r2 = r2\_score(y\_test, y\_pred)  
rmse = np.sqrt(mean\_squared\_error(y\_test, y\_pred))  
  
print(f"R-squared: {r2}")  
print(f"RMSE: {rmse}")

**Tiếng Việt:**

from sklearn.model\_selection import train\_test\_split  
from sklearn.linear\_model import Ridge  
from sklearn.metrics import mean\_squared\_error, r2\_score  
from sklearn.preprocessing import StandardScaler  
  
# Chuẩn bị dữ liệu  
X = df.drop(['molregno', 'canonical\_smiles', 'standard\_type', 'standard\_value', 'standard\_units', 'pIC50'], axis=1) # Loại bỏ các cột không phải descriptor  
y = df['pIC50']  
  
# Scale các đặc trưng  
scaler = StandardScaler()  
X\_scaled = scaler.fit\_transform(X)  
  
# Chia dữ liệu thành tập huấn luyện và tập kiểm tra  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42)  
  
# Huấn luyện mô hình (Ridge Regression)  
model = Ridge(alpha=1.0) # You can adjust the alpha parameter  
model.fit(X\_train, y\_train)  
  
# Dự đoán  
y\_pred = model.predict(X\_test)  
  
# Đánh giá mô hình  
r2 = r2\_score(y\_test, y\_pred)  
rmse = np.sqrt(mean\_squared\_error(y\_test, y\_pred))  
  
print(f"R-squared: {r2}")  
print(f"RMSE: {rmse}")

**Explanation:**

* The code prepares the data by separating features (descriptors) from the target variable (pIC50).
* It splits the data into training and testing sets.
* It trains a Linear Regression model on the training data.
* It makes predictions on the testing data and evaluates the model’s performance using R-squared and RMSE.

**3. Sửa lỗi (Error Handling)**

**a. ERROR: operator does not exist: numeric ~ unknown LINE 12: AND act.standard\_value ~ '^[0-9\.]+$'**

**English:**

This error occurs because the ~ operator is used for regular expression matching, but the standard\_value column is likely of numeric type. To fix this, you can cast the standard\_value to text before using the ~ operator or use a different approach to filter numeric values.

**Fixed 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  
JOIN  
 target\_dictionary td ON act.tid = td.tid  
WHERE  
 td.pref\_name = 'Your Target Name'  
 AND act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 AND act.standard\_value IS NOT NULL  
 AND act.standard\_value > 0 -- Corrected line: Directly compare as numeric  
LIMIT 100;

**Tiếng Việt:**

Lỗi này xảy ra vì toán tử ~ được sử dụng để so khớp biểu thức chính quy, nhưng cột standard\_value có khả năng là kiểu số. Để khắc phục điều này, bạn có thể chuyển đổi standard\_value thành văn bản trước khi sử dụng toán tử ~ hoặc sử dụng một cách tiếp cận khác để lọc các giá trị số.

**Truy vấn SQL đã sửa:**

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  
JOIN  
 target\_dictionary td ON act.tid = td.tid  
WHERE  
 td.pref\_name = 'Tên mục tiêu của bạn'  
 AND act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 AND act.standard\_value IS NOT NULL  
 AND act.standard\_value > 0 -- Dòng đã sửa: So sánh trực tiếp dưới dạng số  
LIMIT 100;

**b. sklearn version is too old**

**English:**

If you encounter an error related to squared=False in mean\_squared\_error, it means your scikit-learn version is outdated. You should update it using pip:

pip install --upgrade scikit-learn

If upgrading is not feasible, you can calculate RMSE manually:

mse = mean\_squared\_error(y\_test, y\_pred)  
rmse = np.sqrt(mse)

**Tiếng Việt:**

Nếu bạn gặp lỗi liên quan đến squared=False trong mean\_squared\_error, điều đó có nghĩa là phiên bản scikit-learn của bạn đã lỗi thời. Bạn nên cập nhật nó bằng pip:

pip install --upgrade scikit-learn

Nếu việc nâng cấp không khả thi, bạn có thể tính RMSE thủ công:

mse = mean\_squared\_error(y\_test, y\_pred)  
rmse = np.sqrt(mse)

**4. Ví dụ code (Code Examples)**

**4.1. SQL (Data Extraction)**

**Example 1: Select compounds targeting a specific protein and their IC50 values.**

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 = 'CHEMBL205' AND act.standard\_type = 'IC50' AND act.standard\_units = 'nM'  
LIMIT 100;

**Example 2: Select compounds with Ki values for a particular target.**

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 = 'CHEMBL205' AND act.standard\_type = 'Ki' AND act.standard\_units = 'nM'  
LIMIT 100;

**Example 3: Filter compounds based on molecular weight (requires a function or table with MW data).** This example assumes you have a table with molecular weights.

-- This is a hypothetical example. Replace 'mw\_table' with your actual table.  
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 mw\_table mw ON md.molregno = mw.molregno -- Hypothetical table  
WHERE td.pref\_name = 'CHEMBL205' AND act.standard\_type = 'IC50' AND act.standard\_units = 'nM' AND mw.mw < 500  
LIMIT 100;

**Example 4: Select compounds and their activities within a specific range.**

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 = 'CHEMBL205' AND act.standard\_type = 'IC50' AND act.standard\_units = 'nM' AND act.standard\_value BETWEEN 10 AND 100  
LIMIT 100;

**Example 5: Retrieve compounds and their activities for multiple targets.**

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 IN ('CHEMBL205', 'CHEMBL206', 'CHEMBL207') AND act.standard\_type = 'IC50' AND act.standard\_units = 'nM'  
LIMIT 100;

**4.2. Python (Data Preprocessing and Modeling)**

import pandas as pd  
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, r2\_score  
import numpy as np  
import os  
  
# Example 1: Calculate LogP and Molecular Weight and Handle Errors  
def calculate\_logp\_mw(smiles):  
 try:  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 logp = Descriptors.MolLogP(mol)  
 mw = Descriptors.MolWt(mol)  
 return logp, mw  
 else:  
 return None, None  
 except:  
 return None, None  
  
  
# Example 2: Feature Scaling (StandardScaler)  
from sklearn.preprocessing import StandardScaler  
  
def scale\_features(df, columns\_to\_scale):  
 scaler = StandardScaler()  
 df[columns\_to\_scale] = scaler.fit\_transform(df[columns\_to\_scale])  
 return df  
  
# Example 3: Train-Test Split with a Fixed Random State  
def split\_data(X, y, test\_size=0.2, random\_state=42):  
 X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=test\_size, random\_state=random\_state)  
 return X\_train, X\_test, y\_train, y\_test  
  
  
# Example 4: Train and Evaluate a Linear Regression Model  
def train\_evaluate\_model(X\_train, X\_test, y\_train, y\_test):  
 model = LinearRegression()  
 model.fit(X\_train, y\_train)  
 y\_pred = model.predict(X\_test)  
 r2 = r2\_score(y\_test, y\_pred)  
 rmse = np.sqrt(mean\_squared\_error(y\_test, y\_pred))  
 return r2, rmse, model  
  
# Example 5: Save Model  
import joblib  
def save\_model(model, filepath):  
 joblib.dump(model, filepath)

This comprehensive guide should give you a solid foundation for analyzing Chembl 35 data. Let me know if you have any specific questions or need further assistance!