# PharmApp Suite

## 🧠 AI for Drug Discovery and Development 🧪

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

## 🧩 Topic: Sàng lọc thuốc tác động hệ thần kinh trung ương

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 một chuyên gia R&D dược học, tôi hiểu rõ tầm quan trọng của việc phân tích dữ liệu một cách hiệu quả để khám phá và phát triển thuốc mới.

Dưới đây là phân tích chi tiết, hướng dẫn song ngữ, code mẫu SQL và Python, cùng với 5 ví dụ cụ thể, tập trung vào việc xử lý lỗi bạn gặp phải và tối ưu hóa quy trình làm việc theo chuẩn AIMLOps.

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

Mục tiêu của bạn là phân tích dữ liệu ChEMBL 35 để tìm ra các mối tương quan giữa cấu trúc hóa học của các hợp chất và hoạt tính sinh học của chúng. Đây là một bài toán QSAR (Quantitative Structure-Activity Relationship) điển hình.

**Các bước chính trong mô hình phân tích:**

1. **Thu thập và Chuẩn bị Dữ liệu (Data Acquisition and Preparation):**
   * Trích xuất dữ liệu từ cơ sở dữ liệu ChEMBL 35 (sử dụng SQL).
   * Làm sạch và tiền xử lý dữ liệu (loại bỏ dữ liệu trùng lặp, xử lý giá trị thiếu, chuẩn hóa đơn vị).
2. **Tính toán Đặc trưng Phân tử (Molecular Feature Calculation):**
   * Sử dụng RDKit để tính toán các đặc trưng phân tử (ví dụ: trọng lượng phân tử, logP, số lượng liên kết, diện tích bề mặt).
3. **Lựa chọn Đặc trưng (Feature Selection):**
   * Chọn các đặc trưng phân tử quan trọng nhất có liên quan đến hoạt tính sinh học (sử dụng các phương pháp thống kê hoặc học máy).
4. **Xây dựng Mô hình (Model Building):**
   * Xây dựng mô hình QSAR bằng các thuật toán học máy (ví dụ: hồi quy tuyến tính, SVM, Random Forest).
5. **Đánh giá Mô hình (Model Evaluation):**
   * Đánh giá hiệu suất của mô hình trên tập dữ liệu kiểm tra (sử dụng các chỉ số như R-squared, RMSE, AUC).
6. **Diễn giải Mô hình (Model Interpretation):**
   * Diễn giải mô hình để hiểu các yếu tố cấu trúc nào ảnh hưởng đến hoạt tính sinh học.

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

* **Data Extraction (Trích xuất dữ liệu):** Use SQL queries to extract relevant data from the ChEMBL database.
  + Sử dụng truy vấn SQL để trích xuất dữ liệu liên quan từ cơ sở dữ liệu ChEMBL.
* **Data Cleaning (Làm sạch dữ liệu):** Remove duplicates, handle missing values, and standardize units.
  + Loại bỏ dữ liệu trùng lặp, xử lý giá trị thiếu và chuẩn hóa đơn vị.
* **Feature Calculation (Tính toán đặc trưng):** Use RDKit to calculate molecular descriptors (e.g., molecular weight, logP).
  + Sử dụng RDKit để tính toán các đặc trưng phân tử (ví dụ: trọng lượng phân tử, logP).
* **Model Building (Xây dựng mô hình):** Train machine learning models to predict biological activity.
  + Xây dựng mô hình học máy để dự đoán hoạt tính sinh học.
* **Model Evaluation (Đánh giá mô hình):** Evaluate model performance using appropriate metrics (e.g., R-squared, RMSE).
  + Đánh giá hiệu suất của mô hình bằng các chỉ số phù hợp (ví dụ: R-squared, RMSE).

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

**SQL (để trích xuất dữ liệu):**

-- English  
-- Extract 100 data from chembl\_35  
SELECT  
 act.molregno,  
 act.standard\_value,  
 act.standard\_units,  
 act.assay\_id,  
 md.molfile  
FROM  
 activities act  
JOIN  
 molecule\_dictionary md ON act.molregno = md.molregno  
WHERE  
 act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 AND act.standard\_value IS NOT NULL  
 AND act.standard\_value > 0  
 AND act.standard\_value < 10000 -- Filter values  
 --AND act.standard\_value ~ '^[0-9\.]+$' -- Removed this line because it causes errors  
LIMIT 100;

-- Vietnamese  
-- Trích xuất 100 dòng dữ liệu từ chembl\_35  
SELECT  
 act.molregno,  
 act.standard\_value,  
 act.standard\_units,  
 act.assay\_id,  
 md.molfile  
FROM  
 activities act  
JOIN  
 molecule\_dictionary md ON act.molregno = md.molregno  
WHERE  
 act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 AND act.standard\_value IS NOT NULL  
 AND act.standard\_value > 0  
 AND act.standard\_value < 10000 -- Lọc giá trị  
 --AND act.standard\_value ~ '^[0-9\.]+$' -- Đã xóa dòng này vì gây ra lỗi  
LIMIT 100;

**Lý do loại bỏ act.standard\_value ~ '^[0-9\.]+$':** Lỗi operator does not exist: numeric ~ unknown xảy ra vì toán tử ~ được sử dụng để so sánh chuỗi với regular expression, nhưng act.standard\_value là kiểu số (numeric). Để khắc phục, bạn có thể bỏ qua điều kiện này (nếu bạn chắc chắn rằng dữ liệu đã được làm sạch) hoặc chuyển đổi act.standard\_value thành kiểu chuỗi trước khi so sánh (nhưng điều này có thể không cần thiết).

**Python (sử dụng RDKit):**

# English  
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, r2\_score  
from sklearn.preprocessing import StandardScaler  
  
# Define base path  
base\_path = "../data"  
  
# Load data from CSV  
csv\_file = "chembl\_35\_data.csv" # Replace with your actual file name  
data = pd.read\_csv(os.path.join(base\_path, csv\_file))  
  
# Handle missing molfile values  
data = data.dropna(subset=['molfile'])  
  
# Function to calculate molecular descriptors  
def calculate\_descriptors(molfile):  
 mol = Chem.MolFromMolBlock(molfile)  
 if mol is None:  
 return None  
 descriptors = {}  
 descriptors['MW'] = Descriptors.MolWt(mol)  
 descriptors['LogP'] = Chem.Crippen.MolLogP(mol)  
 descriptors['HBD'] = Chem.Lipinski.NumHDonors(mol)  
 descriptors['HBA'] = Chem.Lipinski.NumHAcceptors(mol)  
 return descriptors  
  
# Apply descriptor calculation  
data['descriptors'] = data['molfile'].apply(calculate\_descriptors)  
data = data.dropna(subset=['descriptors'])  
  
# Convert descriptors to DataFrame  
data = pd.concat([data.drop(['descriptors'], axis=1), data['descriptors'].apply(pd.Series)], axis=1)  
  
# Convert standard\_value to numeric, handling errors  
data['standard\_value'] = pd.to\_numeric(data['standard\_value'], errors='coerce')  
data = data.dropna(subset=['standard\_value'])  
  
# Prepare data for modeling  
X = data[['MW', 'LogP', 'HBD', 'HBA']] # Use selected descriptors  
y = np.log10(data['standard\_value']) # log transform target variable  
  
# Data scaling  
scaler = StandardScaler()  
X\_scaled = scaler.fit\_transform(X)  
  
# Split 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 a linear regression model  
model = LinearRegression()  
model.fit(X\_train, y\_train)  
  
# Make predictions  
y\_pred = model.predict(X\_test)  
  
# Evaluate the model  
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}")

# Vietnamese  
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, r2\_score  
from sklearn.preprocessing import StandardScaler  
  
# Định nghĩa đường dẫn gốc  
base\_path = "../data"  
  
# Tải dữ liệu từ file CSV  
csv\_file = "chembl\_35\_data.csv" # Thay thế bằng tên file của bạn  
data = pd.read\_csv(os.path.join(base\_path, csv\_file))  
  
# Xử lý giá trị molfile bị thiếu  
data = data.dropna(subset=['molfile'])  
  
# Hàm tính toán các descriptor phân tử  
def calculate\_descriptors(molfile):  
 mol = Chem.MolFromMolBlock(molfile)  
 if mol is None:  
 return None  
 descriptors = {}  
 descriptors['MW'] = Descriptors.MolWt(mol)  
 descriptors['LogP'] = Chem.Crippen.MolLogP(mol)  
 descriptors['HBD'] = Chem.Lipinski.NumHDonors(mol)  
 descriptors['HBA'] = Chem.Lipinski.NumHAcceptors(mol)  
 return descriptors  
  
# Áp dụng tính toán descriptor  
data['descriptors'] = data['molfile'].apply(calculate\_descriptors)  
data = data.dropna(subset=['descriptors'])  
  
# Chuyển đổi descriptor thành DataFrame  
data = pd.concat([data.drop(['descriptors'], axis=1), data['descriptors'].apply(pd.Series)], axis=1)  
  
# Chuyển đổi standard\_value sang kiểu số, xử lý lỗi  
data['standard\_value'] = pd.to\_numeric(data['standard\_value'], errors='coerce')  
data = data.dropna(subset=['standard\_value'])  
  
# Chuẩn bị dữ liệu cho mô hình  
X = data[['MW', 'LogP', 'HBD', 'HBA']] # Sử dụng các descriptor đã chọn  
y = np.log10(data['standard\_value']) # log transform biến mục tiêu  
  
# Chuẩn hóa dữ liệu  
scaler = StandardScaler()  
X\_scaled = scaler.fit\_transform(X)  
  
# Chia dữ liệu thành tập huấn luyện và 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 hồi quy tuyến tính  
model = LinearRegression()  
model.fit(X\_train, y\_train)  
  
# Dự đoán  
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"Mean Squared Error: {mse}")  
print(f"R-squared: {r2}")

**4. Ví dụ Code (Code Examples)**

**Ví dụ 1: Tính toán nhiều descriptor hơn (Calculating More Descriptors)**

**SQL:** (Không thay đổi)

**Python:**

# English  
from rdkit.Chem import AllChem  
  
def calculate\_descriptors(molfile):  
 mol = Chem.MolFromMolBlock(molfile)  
 if mol is None:  
 return None  
 descriptors = {}  
 descriptors['MW'] = Descriptors.MolWt(mol)  
 descriptors['LogP'] = Chem.Crippen.MolLogP(mol)  
 descriptors['HBD'] = Chem.Lipinski.NumHDonors(mol)  
 descriptors['HBA'] = Chem.Lipinski.NumHAcceptors(mol)  
 descriptors['TPSA'] = Chem.QED.properties(mol).PSA  
 descriptors['RotatableBonds'] = Chem.Lipinski.NumRotatableBonds(mol)  
 return descriptors

# Vietnamese  
from rdkit.Chem import AllChem  
  
def calculate\_descriptors(molfile):  
 mol = Chem.MolFromMolBlock(molfile)  
 if mol is None:  
 return None  
 descriptors = {}  
 descriptors['MW'] = Descriptors.MolWt(mol)  
 descriptors['LogP'] = Chem.Crippen.MolLogP(mol)  
 descriptors['HBD'] = Chem.Lipinski.NumHDonors(mol)  
 descriptors['HBA'] = Chem.Lipinski.NumHAcceptors(mol)  
 descriptors['TPSA'] = Chem.QED.properties(mol).PSA  
 descriptors['RotatableBonds'] = Chem.Lipinski.NumRotatableBonds(mol)  
 return descriptors

**Ví dụ 2: Sử dụng Random Forest (Using Random Forest)**

**SQL:** (Không thay đổi)

**Python:**

# English  
from sklearn.ensemble import RandomForestRegressor  
  
# Train a Random Forest model  
model = RandomForestRegressor(n\_estimators=100, random\_state=42) # you can adjust the hyperparameters of the model, such as n\_estimators.  
model.fit(X\_train, y\_train)  
  
# Make predictions  
y\_pred = model.predict(X\_test)  
  
# Evaluate the model  
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}")

# Vietnamese  
from sklearn.ensemble import RandomForestRegressor  
  
# Huấn luyện mô hình Random Forest  
model = RandomForestRegressor(n\_estimators=100, random\_state=42) # Bạn có thể điều chỉnh các siêu tham số của mô hình, chẳng hạn như n\_estimators.  
model.fit(X\_train, y\_train)  
  
# Dự đoán  
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"Mean Squared Error: {mse}")  
print(f"R-squared: {r2}")

**Ví dụ 3: Feature Selection (Lựa chọn Đặc trưng)**

**SQL:** (Không thay đổi)

**Python:**

# English  
from sklearn.feature\_selection import SelectKBest, f\_regression  
  
# Feature selection using SelectKBest  
selector = SelectKBest(score\_func=f\_regression, k=3)  
X\_new = selector.fit\_transform(X\_scaled, y)  
  
# Get the selected features  
selected\_features = X.columns[selector.get\_support()]  
print("Selected Features:", selected\_features)  
  
# Split data into training and testing sets using selected features  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_new, y, test\_size=0.2, random\_state=42)  
  
# Train a linear regression model  
model = LinearRegression()  
model.fit(X\_train, y\_train)  
  
# Make predictions  
y\_pred = model.predict(X\_test)  
  
# Evaluate the model  
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}")

# Vietnamese  
from sklearn.feature\_selection import SelectKBest, f\_regression  
  
# Lựa chọn đặc trưng sử dụng SelectKBest  
selector = SelectKBest(score\_func=f\_regression, k=3)  
X\_new = selector.fit\_transform(X\_scaled, y)  
  
# Lấy các đặc trưng đã chọn  
selected\_features = X.columns[selector.get\_support()]  
print("Selected Features:", selected\_features)  
  
# Chia dữ liệu thành tập huấn luyện và kiểm tra sử dụng các đặc trưng đã chọn  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_new, y, test\_size=0.2, random\_state=42)  
  
# Huấn luyện mô hình hồi quy tuyến tính  
model = LinearRegression()  
model.fit(X\_train, y\_train)  
  
# Dự đoán  
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"Mean Squared Error: {mse}")  
print(f"R-squared: {r2}")

**Ví dụ 4: Lưu Mô hình (Saving the Model)**

**SQL:** (Không thay đổi)

**Python:**

# English  
import joblib  
  
# Train your model (e.g., Linear Regression)  
model = LinearRegression()  
model.fit(X\_train, y\_train)  
  
# Save the model to a file  
model\_filename = "linear\_regression\_model.pkl"  
joblib.dump(model, os.path.join(base\_path, model\_filename))  
  
print(f"Model saved to {os.path.join(base\_path, model\_filename)}")  
  
# Load the model from the file  
loaded\_model = joblib.load(os.path.join(base\_path, model\_filename))  
  
# Use the loaded model to make predictions  
y\_pred = loaded\_model.predict(X\_test)

# Vietnamese  
import joblib  
  
# Huấn luyện mô hình của bạn (ví dụ: Hồi quy tuyến tính)  
model = LinearRegression()  
model.fit(X\_train, y\_train)  
  
# Lưu mô hình vào một file  
model\_filename = "linear\_regression\_model.pkl"  
joblib.dump(model, os.path.join(base\_path, model\_filename))  
  
print(f"Mô hình đã được lưu vào {os.path.join(base\_path, model\_filename)}")  
  
# Tải mô hình từ file  
loaded\_model = joblib.load(os.path.join(base\_path, model\_filename))  
  
# Sử dụng mô hình đã tải để dự đoán  
y\_pred = loaded\_model.predict(X\_test)

**Ví dụ 5: Sử dụng mô hình SVM (Support Vector Machine)**

**SQL:** (Không thay đổi)

**Python:**

#English  
from sklearn.svm import SVR  
  
# Train the SVM model  
model = SVR(kernel='linear') # You can also use other kernels like 'rbf', 'poly', etc.  
model.fit(X\_train, y\_train)  
  
# Make predictions  
y\_pred = model.predict(X\_test)  
  
# Evaluate the model  
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}")

#Vietnamese  
from sklearn.svm import SVR  
  
# Huấn luyện mô hình SVM  
model = SVR(kernel='linear') # Bạn cũng có thể sử dụng các kernel khác như 'rbf', 'poly', vv.  
model.fit(X\_train, y\_train)  
  
# Dự đoán  
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"Mean Squared Error: {mse}")  
print(f"R-squared: {r2}")

**Lưu ý quan trọng:**

* **Phiên bản Scikit-learn:** Đảm bảo bạn đang sử dụng phiên bản Scikit-learn mới nhất để hỗ trợ tham số squared=False trong hàm mean\_squared\_error. Nếu không, hãy loại bỏ tham số này. Trong các phiên bản mới, squared=False không còn cần thiết và mặc định trả về RMSE (Root Mean Squared Error).
* **Đường dẫn:** Sử dụng os.path.join(base\_path, filename) để đảm bảo tính di động của code.
* **Xử lý lỗi:** Luôn kiểm tra và xử lý các lỗi có thể xảy ra trong quá trình tính toán đặc trưng và xây dựng mô hình.
* **Chuẩn hóa dữ liệu:** Chuẩn hóa dữ liệu trước khi đưa vào mô hình học máy để cải thiện hiệu suất.
* **Lựa chọn mô hình:** Thử nghiệm với nhiều mô hình học máy khác nhau để tìm ra mô hình tốt nhất cho bài toán của bạn.
* **Lưu trữ dữ liệu:** Lưu trữ dữ liệu đã xử lý và mô hình đã huấn luyện để tái sử dụng và so sánh.

Hy vọng những hướng dẫn và ví dụ này sẽ giúp bạn hoàn thành dự án phân tích dữ liệu ChEMBL 35 một cách hiệu quả! Nếu bạn có bất kỳ câu hỏi nào khác, đừng ngần ngại hỏi.