# PharmApp Suite

## 🧠 AI for Drug Discovery and Development 🧪

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

## 🧩 Topic: QSAR với Random Forest và đặc trưng từ ChEMBL

Tuyệt vời! Tôi rất vui được tham gia vào dự án phân tích dữ liệu ChEMBL 35 của bạn. Dựa trên thông tin bạn cung cấp, tôi sẽ giúp bạn phân tích, xây dựng mô hình, và cung cấp code mẫu SQL và Python để bạn có thể bắt đầu một cách hiệu quả.

**Tổng quan dự án:**

Dự án của bạn tập trung vào việc khai thác dữ liệu từ cơ sở dữ liệu ChEMBL 35, kết hợp với RDKit để hỗ trợ nghiên cứu và phát triển thuốc. Bạn đang sử dụng một cấu trúc thư mục theo chuẩn AIMLOps Template và sử dụng Jupyter Notebook để thực thi code.

**Mục tiêu:**

* Phân tích dữ liệu ChEMBL 35 để tìm ra các mối quan hệ tiềm năng giữa cấu trúc hóa học và hoạt tính sinh học của các hợp chất.
* Sử dụng RDKit để tính toán các descriptor phân tử từ cấu trúc hóa học.
* Xây dựng mô hình học máy để dự đoán hoạt tính sinh học của các hợp chất mới.

**Phân tích và Hướng dẫn:**

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

Chúng ta sẽ xây dựng một quy trình phân tích dữ liệu và xây dựng mô hình dự đoán, bao gồm các bước sau:

* **Data Extraction (Trích xuất dữ liệu):** Sử dụng SQL để truy vấn dữ liệu từ cơ sở dữ liệu ChEMBL 35.
* **Data Preprocessing (Tiền xử lý dữ liệu):**
  + Làm sạch dữ liệu, xử lý các giá trị thiếu (missing values).
  + Chuyển đổi dữ liệu về định dạng phù hợp cho việc phân tích.
* **Feature Engineering (Xây dựng đặc trưng):**
  + Sử dụng RDKit để tính toán các descriptor phân tử từ cấu trúc SMILES.
  + Chọn lọc các descriptor phù hợp.
* **Model Selection (Lựa chọn mô hình):**
  + Thử nghiệm với nhiều mô hình học máy khác nhau (ví dụ: Linear Regression, Random Forest, Support Vector Machines).
  + Sử dụng cross-validation để đánh giá hiệu suất của các mô hình.
* **Model Evaluation (Đánh giá mô hình):**
  + Sử dụng các metrics phù hợp (ví dụ: RMSE, R-squared) để đánh giá hiệu suất của mô hình trên tập test.
* **Model Deployment (Triển khai mô hình):**
  + Lưu mô hình đã huấn luyện để sử dụng trong tương lai.

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

* **SQL:** Sử dụng SQL để truy vấn và trích xuất dữ liệu từ cơ sở dữ liệu ChEMBL.
* **Python:** Sử dụng Python với các thư viện như pandas, RDKit, scikit-learn để tiền xử lý dữ liệu, tính toán descriptor, xây dựng mô hình và đánh giá mô hình.

**3. Code mẫu SQL và Python (Sample SQL and Python Code):**

**SQL Code (Lấy 100 dòng dữ liệu):**

-- English  
-- Extract 100 rows of data from the chembl\_35 database, including molecule information and activity data.  
  
SELECT  
 md.chembl\_id,  
 md.pref\_name,  
 ms.structure,  
 act.standard\_type,  
 act.standard\_value,  
 act.standard\_units  
FROM  
 molecule\_dictionary md  
JOIN  
 compound\_structures ms ON md.molregno = ms.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  
LIMIT 100;  
  
  
-- Vietnamese  
-- Trích xuất 100 dòng dữ liệu từ cơ sở dữ liệu chembl\_35, bao gồm thông tin về phân tử và dữ liệu hoạt tính.  
  
SELECT  
 md.chembl\_id,  
 md.pref\_name,  
 ms.structure,  
 act.standard\_type,  
 act.standard\_value,  
 act.standard\_units  
FROM  
 molecule\_dictionary md  
JOIN  
 compound\_structures ms ON md.molregno = ms.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  
LIMIT 100;

**Python Code:**

# English  
import os  
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  
  
# Define base path  
base\_path = "../data" # Adjust if necessary  
  
# 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))  
  
# Data Preprocessing  
data = data.dropna(subset=['structure', 'standard\_value'])  
data['standard\_value'] = pd.to\_numeric(data['standard\_value'], errors='coerce')  
data = data.dropna(subset=['standard\_value'])  
  
# RDKit Feature Calculation  
def calculate\_descriptors(smiles):  
 try:  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is None:  
 return None  
 descriptors = {}  
 for name, func in Descriptors.descList:  
 try:  
 descriptors[name] = func(mol)  
 except:  
 descriptors[name] = np.nan  
 return pd.Series(descriptors)  
 except:  
 return None  
  
data['descriptors'] = data['structure'].apply(calculate\_descriptors)  
data = data.dropna(subset=['descriptors'])  
  
# Convert descriptors to DataFrame  
descriptors\_df = pd.DataFrame(data['descriptors'].tolist())  
data = pd.concat([data, descriptors\_df], axis=1)  
  
# Drop rows with NaN values in descriptors  
data = data.dropna(subset=descriptors\_df.columns)  
  
# Prepare data for modeling  
X = data[descriptors\_df.columns]  
y = data['standard\_value']  
  
# 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)  
  
# 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  
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  
  
# Định nghĩa đường dẫn gốc  
base\_path = "../data" # Điều chỉnh nếu cần thiết  
  
# Tải dữ liệu từ file CSV  
csv\_file = "chembl\_35\_data.csv" # Thay thế bằng tên file thực tế của bạn  
data = pd.read\_csv(os.path.join(base\_path, csv\_file))  
  
# Tiền xử lý dữ liệu  
data = data.dropna(subset=['structure', 'standard\_value'])  
data['standard\_value'] = pd.to\_numeric(data['standard\_value'], errors='coerce')  
data = data.dropna(subset=['standard\_value'])  
  
# Tính toán các descriptor phân tử sử dụng RDKit  
def calculate\_descriptors(smiles):  
 try:  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is None:  
 return None  
 descriptors = {}  
 for name, func in Descriptors.descList:  
 try:  
 descriptors[name] = func(mol)  
 except:  
 descriptors[name] = np.nan  
 return pd.Series(descriptors)  
 except:  
 return None  
  
data['descriptors'] = data['structure'].apply(calculate\_descriptors)  
data = data.dropna(subset=['descriptors'])  
  
# Chuyển đổi descriptors thành DataFrame  
descriptors\_df = pd.DataFrame(data['descriptors'].tolist())  
data = pd.concat([data, descriptors\_df], axis=1)  
  
# Loại bỏ các hàng có giá trị NaN trong descriptors  
data = data.dropna(subset=descriptors\_df.columns)  
  
# Chuẩn bị dữ liệu cho mô hình hóa  
X = data[descriptors\_df.columns]  
y = data['standard\_value']  
  
# 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 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 .sql và python mẫu (Sample SQL and Python Code Examples):**

**SQL Examples:**

1. **Lọc các hợp chất có trọng lượng phân tử (MW) dưới 500:**

-- English  
SELECT md.chembl\_id, ms.structure FROM molecule\_dictionary md JOIN compound\_structures ms ON md.molregno = ms.molregno WHERE md.mw\_freebase < 500 LIMIT 100;  
  
-- Vietnamese  
SELECT md.chembl\_id, ms.structure FROM molecule\_dictionary md JOIN compound\_structures ms ON md.molregno = ms.molregno WHERE md.mw\_freebase < 500 LIMIT 100;

1. **Tìm các hợp chất có chứa một khung cấu trúc cụ thể (ví dụ: benzene ring):**

-- English  
SELECT md.chembl\_id, ms.structure FROM molecule\_dictionary md JOIN compound\_structures ms ON md.molregno = ms.molregno WHERE ms.structure LIKE '%c1ccccc1%' LIMIT 100;  
  
-- Vietnamese  
SELECT md.chembl\_id, ms.structure FROM molecule\_dictionary md JOIN compound\_structures ms ON md.molregno = ms.molregno WHERE ms.structure LIKE '%c1ccccc1%' LIMIT 100;

1. **Lấy thông tin về các mục tiêu (targets) liên quan đến một hợp chất cụ thể:**

-- English  
SELECT td.chembl\_id, td.pref\_name FROM target\_dictionary td JOIN target\_components tc ON td.tid = tc.tid JOIN component\_sequences cs ON tc.component\_id = cs.component\_id WHERE cs.sequence LIKE '%amino acid sequence%' LIMIT 100;  
  
-- Vietnamese  
SELECT td.chembl\_id, td.pref\_name FROM target\_dictionary td JOIN target\_components tc ON td.tid = tc.tid JOIN component\_sequences cs ON tc.component\_id = cs.component\_id WHERE cs.sequence LIKE '%amino acid sequence%' LIMIT 100;

1. **Tìm các hoạt động (activities) có giá trị IC50 nhỏ hơn 100 nM:**

-- English  
SELECT md.chembl\_id, act.standard\_value FROM molecule\_dictionary md JOIN activities act ON md.molregno = act.molregno WHERE act.standard\_type = 'IC50' AND act.standard\_value < 100 LIMIT 100;  
  
-- Vietnamese  
SELECT md.chembl\_id, act.standard\_value FROM molecule\_dictionary md JOIN activities act ON md.molregno = act.molregno WHERE act.standard\_type = 'IC50' AND act.standard\_value < 100 LIMIT 100;

1. **Thống kê số lượng hợp chất cho mỗi loại hoạt động (activity type):**

-- English  
SELECT act.standard\_type, COUNT(\*) FROM activities act GROUP BY act.standard\_type LIMIT 100;  
  
-- Vietnamese  
SELECT act.standard\_type, COUNT(\*) FROM activities act GROUP BY act.standard\_type LIMIT 100;

**Python Examples:**

1. **Tính toán logP cho các hợp chất:**

# English  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
  
def calculate\_logp(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 return Descriptors.MolLogP(mol)  
 else:  
 return None  
  
# Apply the function to a DataFrame column  
data['logP'] = data['structure'].apply(calculate\_logp)  
  
# Vietnamese  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
  
def calculate\_logp(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 return Descriptors.MolLogP(mol)  
 else:  
 return None  
  
# Áp dụng hàm cho một cột DataFrame  
data['logP'] = data['structure'].apply(calculate\_logp)

1. **Tính toán số lượng vòng (ring count) trong mỗi hợp chất:**

# English  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
  
def calculate\_ring\_count(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 return Descriptors.RingCount(mol)  
 else:  
 return None  
  
# Apply the function to a DataFrame column  
data['ring\_count'] = data['structure'].apply(calculate\_ring\_count)  
  
# Vietnamese  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
  
def calculate\_ring\_count(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 return Descriptors.RingCount(mol)  
 else:  
 return None  
  
# Áp dụng hàm cho một cột DataFrame  
data['ring\_count'] = data['structure'].apply(calculate\_ring\_count)

1. **Chuẩn hóa dữ liệu hoạt tính (ví dụ: chuyển đổi IC50 sang pIC50):**

# English  
import numpy as np  
  
def convert\_ic50\_to\_pic50(ic50):  
 if ic50 is not None and ic50 > 0:  
 return -np.log10(ic50 / 1e9) # Convert nM to M  
 else:  
 return None  
  
# Apply the function to a DataFrame column  
data['pIC50'] = data['standard\_value'].apply(convert\_ic50\_to\_pic50)  
  
# Vietnamese  
import numpy as np  
  
def convert\_ic50\_to\_pic50(ic50):  
 if ic50 is not None and ic50 > 0:  
 return -np.log10(ic50 / 1e9) # Chuyển đổi nM sang M  
 else:  
 return None  
  
# Áp dụng hàm cho một cột DataFrame  
data['pIC50'] = data['standard\_value'].apply(convert\_ic50\_to\_pic50)

1. **Xây dựng mô hình Random Forest để dự đoán hoạt tính:**

# English  
from sklearn.ensemble import RandomForestRegressor  
from sklearn.model\_selection import train\_test\_split  
from sklearn.metrics import mean\_squared\_error  
  
# Prepare data (assuming X and y are already defined)  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)  
  
# Create and train the model  
model = RandomForestRegressor(n\_estimators=100, random\_state=42)  
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)  
print(f"Mean Squared Error: {mse}")  
  
# Vietnamese  
from sklearn.ensemble import RandomForestRegressor  
from sklearn.model\_selection import train\_test\_split  
from sklearn.metrics import mean\_squared\_error  
  
# Chuẩn bị dữ liệu (giả sử X và y đã được định nghĩa)  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)  
  
# Tạo và huấn luyện mô hình  
model = RandomForestRegressor(n\_estimators=100, random\_state=42)  
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)  
print(f"Mean Squared Error: {mse}")

1. **Sử dụng PCA để giảm số chiều của dữ liệu:**

# English  
from sklearn.decomposition import PCA  
from sklearn.preprocessing import StandardScaler  
  
# Scale the data  
scaler = StandardScaler()  
X\_scaled = scaler.fit\_transform(X)  
  
# Apply PCA  
pca = PCA(n\_components=10) # Reduce to 10 components  
X\_pca = pca.fit\_transform(X\_scaled)  
  
# Vietnamese  
from sklearn.decomposition import PCA  
from sklearn.preprocessing import StandardScaler  
  
# Chuẩn hóa dữ liệu  
scaler = StandardScaler()  
X\_scaled = scaler.fit\_transform(X)  
  
# Áp dụng PCA  
pca = PCA(n\_components=10) # Giảm xuống 10 thành phần  
X\_pca = pca.fit\_transform(X\_scaled)

**Lưu ý về lỗi:**

* **Lỗi SQL “operator does not exist: numeric ~ unknown”:** Lỗi này xảy ra do bạn đang cố gắng sử dụng toán tử ~ (regular expression match) trên một cột kiểu số (numeric). Để khắc phục, bạn có thể cast cột standard\_value sang kiểu text trước khi so sánh:

WHERE CAST(act.standard\_value AS TEXT) ~ '^[0-9\.]+$'

* **Lỗi “squared=False” trong mean\_squared\_error:** Nếu bạn đang sử dụng phiên bản scikit-learn cũ, hãy loại bỏ tham số squared=False hoặc nâng cấp lên phiên bản mới hơn.

**Lưu ý quan trọng:**

* Hãy nhớ thay thế các tên file và đường dẫn trong code mẫu bằng các giá trị phù hợp với cấu trúc thư mục và tên file của bạn.
* Đây chỉ là những ví dụ cơ bản. Bạn có thể cần điều chỉnh code để phù hợp với yêu cầu cụ thể của dự án.
* Việc lựa chọn mô hình phù hợp phụ thuộc vào đặc điểm của dữ liệu và mục tiêu của bạn. Hãy thử nghiệm với nhiều mô hình khác nhau và đánh giá hiệu suất của chúng một cách cẩn thận.

Chúc bạn thành công với dự án của mình! Nếu bạn có bất kỳ câu hỏi nào, đừng ngần ngại hỏi.