# PharmApp Suite

## 🧠 AI for Drug Discovery and Development 🧪

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

## 🧩 Topic: Phân tích ligand nhắm GPCR

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 và những yêu cầu cụ thể, tôi sẽ giúp bạn xây dựng quy trình phân tích dữ liệu hiệu quả, tập trung vào việc giải quyết các vấn đề bạn gặp phải và tuân thủ theo template AIMLOps.

**Tổng quan về quy trình phân tích**

Chúng ta sẽ thực hiện các bước sau:

1. **Data Extraction (Trích xuất dữ liệu):** Sử dụng SQL để truy vấn dữ liệu từ cơ sở dữ liệu ChEMBL 35 và lưu vào file CSV. Chú ý xử lý lỗi operator does not exist: numeric ~ unknown.
2. **Data Preprocessing (Tiền xử lý dữ liệu):** Đọc file CSV, làm sạch và chuẩn hóa dữ liệu bằng RDKit và các thư viện Python khác.
3. **Feature Engineering (Xây dựng đặc trưng):** Tạo các đặc trưng hóa học (chemical features) từ SMILES strings sử dụng RDKit.
4. **Model Building (Xây dựng mô hình):** Xây dựng mô hình học máy (ví dụ: regression) để dự đoán hoạt tính của các hợp chất.
5. **Model Evaluation (Đánh giá mô hình):** Đánh giá hiệu năng của mô hình và tinh chỉnh nếu cần.

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

Mô hình phân tích sẽ tập trung vào việc xây dựng một mô hình dự đoán hoạt tính (activity) của các hợp chất dựa trên cấu trúc hóa học của chúng. Cụ thể, chúng ta sẽ sử dụng mô hình hồi quy (regression model) để dự đoán giá trị hoạt tính (ví dụ: IC50, Ki).

* **Input (Đầu vào):** SMILES strings (biểu diễn cấu trúc hóa học) của các hợp chất.
* **Process (Quy trình):**
  + Sử dụng RDKit để chuyển đổi SMILES strings thành các đặc trưng số (numerical features) như fingerprints, descriptors.
  + Sử dụng các thuật toán hồi quy (ví dụ: Linear Regression, Random Forest, XGBoost) để xây dựng mô hình dự đoán.
* **Output (Đầu ra):** Mô hình dự đoán giá trị hoạt tính của hợp chất dựa trên cấu trúc hóa học.

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

**2.1. SQL Code (Mã SQL)**

* **Purpose:** Extract data from ChEMBL database and save to CSV file. (Mục đích: Trích xuất dữ liệu từ cơ sở dữ liệu ChEMBL và lưu vào file CSV.)
* **Explanation:** The following SQL code retrieves compound information and activity data, addressing the error related to numeric comparison. (Giải thích: Mã SQL sau truy xuất thông tin hợp chất và dữ liệu hoạt tính, giải quyết lỗi liên quan đến so sánh số.)

-- English  
-- Extracting data from ChEMBL database  
  
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\.]+$' -- Check if the value is numeric  
LIMIT 100;  
  
-- Vietnamese  
-- Trích xuất dữ liệu từ cơ sở dữ liệu 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  
WHERE act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 AND act.standard\_value IS NOT NULL  
 AND act.standard\_value::text ~ '^[0-9\.]+$' -- Kiểm tra xem giá trị có phải là số không  
LIMIT 100;

**Giải thích lỗi và cách khắc phục:**

Lỗi ERROR: operator does not exist: numeric ~ unknown xảy ra do PostgreSQL không thể so sánh trực tiếp kiểu dữ liệu numeric với một biểu thức regular expression (kiểu unknown). Để khắc phục, chúng ta cần ép kiểu dữ liệu standard\_value về kiểu text trước khi so sánh bằng ~ (regular expression matching).

**2.2. Python Code (Mã Python)**

* **Purpose:** Load data from CSV, preprocess data using RDKit, and build a regression model. (Mục đích: Tải dữ liệu từ CSV, tiền xử lý dữ liệu bằng RDKit và xây dựng mô hình hồi quy.)
* **Explanation:** The following Python code loads data, converts SMILES to features, and builds a linear regression model. It also addresses the scikit-learn version issue. (Giải thích: Mã Python sau tải dữ liệu, chuyển đổi SMILES thành đặc trưng và xây dựng mô hình hồi quy tuyến tính. Nó cũng giải quyết vấn đề phiên bản scikit-learn.)

# English  
import os  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import AllChem  
from sklearn.model\_selection import train\_test\_split  
from sklearn.linear\_model import LinearRegression  
from sklearn.metrics import mean\_squared\_error  
  
# Define base path  
base\_path = "../data"  
csv\_file = "chembl\_data.csv" # Replace with your actual file name  
csv\_path = os.path.join(base\_path, csv\_file)  
  
# Load data from CSV  
try:  
 df = pd.read\_csv(csv\_path)  
except FileNotFoundError:  
 print(f"Error: The file {csv\_path} was not found.")  
 exit()  
  
# Data Preprocessing  
df = df.dropna(subset=['canonical\_smiles', 'standard\_value'])  
  
# Convert SMILES to features using RDKit (Morgan Fingerprints)  
def smiles\_to\_features(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 fp = AllChem.GetMorganFingerprintAsBitVect(mol, 2, nBits=1024)  
 return list(fp)  
 else:  
 return None  
  
df['features'] = df['canonical\_smiles'].apply(smiles\_to\_features)  
df = df.dropna(subset=['features'])  
  
# Prepare data for modeling  
X = list(df['features'])  
y = df['standard\_value'].astype(float)  
  
# 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)  
  
# Build and train the Linear Regression model  
model = LinearRegression()  
model.fit(X\_train, y\_train)  
  
# Make predictions  
y\_pred = model.predict(X\_test)  
  
# Evaluate the model  
try:  
 mse = mean\_squared\_error(y\_test, y\_pred, squared=False) # Try with squared=False first  
except TypeError:  
 mse = mean\_squared\_error(y\_test, y\_pred) # If squared=False is not supported  
  
print(f"Mean Squared Error: {mse}")  
  
# Vietnamese  
import os  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import AllChem  
from sklearn.model\_selection import train\_test\_split  
from sklearn.linear\_model import LinearRegression  
from sklearn.metrics import mean\_squared\_error  
  
# Định nghĩa đường dẫn cơ sở  
base\_path = "../data"  
csv\_file = "chembl\_data.csv" # Thay thế bằng tên file thực tế của bạn  
csv\_path = os.path.join(base\_path, csv\_file)  
  
# Tải dữ liệu từ CSV  
try:  
 df = pd.read\_csv(csv\_path)  
except FileNotFoundError:  
 print(f"Lỗi: Không tìm thấy file {csv\_path}.")  
 exit()  
  
# Tiền xử lý dữ liệu  
df = df.dropna(subset=['canonical\_smiles', 'standard\_value'])  
  
# Chuyển đổi SMILES thành đặc trưng sử dụng RDKit (Morgan Fingerprints)  
def smiles\_to\_features(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 fp = AllChem.GetMorganFingerprintAsBitVect(mol, 2, nBits=1024)  
 return list(fp)  
 else:  
 return None  
  
df['features'] = df['canonical\_smiles'].apply(smiles\_to\_features)  
df = df.dropna(subset=['features'])  
  
# Chuẩn bị dữ liệu cho mô hình  
X = list(df['features'])  
y = df['standard\_value'].astype(float)  
  
# 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)  
  
# Xây dựng và 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  
try:  
 mse = mean\_squared\_error(y\_test, y\_pred, squared=False) # Thử với squared=False trước  
except TypeError:  
 mse = mean\_squared\_error(y\_test, y\_pred) # Nếu squared=False không được hỗ trợ  
  
print(f"Sai số bình phương trung bình: {mse}")

**Giải thích lỗi và cách khắc phục:**

* **squared=False in mean\_squared\_error:** Phiên bản cũ của scikit-learn không hỗ trợ tham số squared=False trong hàm mean\_squared\_error. Để khắc phục, chúng ta sử dụng một khối try...except để thử sử dụng squared=False trước. Nếu lỗi TypeError xảy ra, chúng ta sẽ sử dụng mean\_squared\_error mà không có tham số squared=False.
* **Xử lý FileNotFoundError:** Đoạn code đã được bổ sung khối try...except để bắt lỗi FileNotFoundError khi file CSV không tồn tại.

**3. Ví dụ Code (Code Examples)**

Dưới đây là 5 ví dụ code SQL và Python mẫu để bạn tham khảo:

**3.1. SQL Examples**

1. **Extract data for a specific target (Lấy dữ liệu cho một mục tiêu cụ thể):**

-- English  
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.target\_name = 'CHEMBL205' -- Replace with your target name  
AND 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;  
  
-- Vietnamese  
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.target\_name = 'CHEMBL205' -- Thay thế bằng 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::text ~ '^[0-9\.]+$'  
LIMIT 100;

1. **Extract data for a specific assay (Lấy dữ liệu cho một assay cụ thể):**

-- English  
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 assays a ON act.assay\_id = a.assay\_id  
WHERE a.description LIKE '%acetylcholinesterase%' -- Replace with your assay description  
AND 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;  
  
-- Vietnamese  
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 assays a ON act.assay\_id = a.assay\_id  
WHERE a.description LIKE '%acetylcholinesterase%' -- Thay thế bằng mô tả assay của bạn  
AND 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;

1. **Extract data with a specific activity range (Lấy dữ liệu với một khoảng giá trị hoạt tính cụ thể):**

-- English  
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  
WHERE act.standard\_type = 'IC50'  
AND act.standard\_units = 'nM'  
AND act.standard\_value BETWEEN 10 AND 100 -- Replace with your activity range  
AND act.standard\_value IS NOT NULL  
AND act.standard\_value::text ~ '^[0-9\.]+$'  
LIMIT 100;  
  
-- Vietnamese  
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  
WHERE act.standard\_type = 'IC50'  
AND act.standard\_units = 'nM'  
AND act.standard\_value BETWEEN 10 AND 100 -- Thay thế bằng khoảng giá trị hoạt tính của bạn  
AND act.standard\_value IS NOT NULL  
AND act.standard\_value::text ~ '^[0-9\.]+$'  
LIMIT 100;

1. **Extract data and order by activity (Lấy dữ liệu và sắp xếp theo hoạt tính):**

-- English  
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  
WHERE act.standard\_type = 'IC50'  
AND act.standard\_units = 'nM'  
AND act.standard\_value IS NOT NULL  
AND act.standard\_value::text ~ '^[0-9\.]+$'  
ORDER BY act.standard\_value ASC -- or DESC for descending order  
LIMIT 100;  
  
-- Vietnamese  
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  
WHERE act.standard\_type = 'IC50'  
AND act.standard\_units = 'nM'  
AND act.standard\_value IS NOT NULL  
AND act.standard\_value::text ~ '^[0-9\.]+$'  
ORDER BY act.standard\_value ASC -- hoặc DESC để sắp xếp giảm dần  
LIMIT 100;

1. **Extract distinct compounds (Lấy các hợp chất khác nhau):**

-- English  
SELECT DISTINCT cs.canonical\_smiles  
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;  
  
-- Vietnamese  
SELECT DISTINCT cs.canonical\_smiles  
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;

**3.2. Python Examples**

1. **Calculate and visualize molecular weight (Tính toán và trực quan hóa khối lượng phân tử):**

# English  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
import matplotlib.pyplot as plt  
import os  
  
base\_path = "../data"  
csv\_file = "chembl\_data.csv"  
csv\_path = os.path.join(base\_path, csv\_file)  
  
df = pd.read\_csv(csv\_path)  
df = df.dropna(subset=['canonical\_smiles'])  
  
def calculate\_mw(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 return Descriptors.MolWt(mol)  
 else:  
 return None  
  
df['mol\_weight'] = df['canonical\_smiles'].apply(calculate\_mw)  
df = df.dropna(subset=['mol\_weight'])  
  
plt.hist(df['mol\_weight'], bins=50)  
plt.xlabel('Molecular Weight')  
plt.ylabel('Frequency')  
plt.title('Distribution of Molecular Weight')  
plt.show()  
  
# Vietnamese  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
import matplotlib.pyplot as plt  
import os  
  
base\_path = "../data"  
csv\_file = "chembl\_data.csv"  
csv\_path = os.path.join(base\_path, csv\_file)  
  
df = pd.read\_csv(csv\_path)  
df = df.dropna(subset=['canonical\_smiles'])  
  
def calculate\_mw(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 return Descriptors.MolWt(mol)  
 else:  
 return None  
  
df['mol\_weight'] = df['canonical\_smiles'].apply(calculate\_mw)  
df = df.dropna(subset=['mol\_weight'])  
  
plt.hist(df['mol\_weight'], bins=50)  
plt.xlabel('Khối lượng phân tử')  
plt.ylabel('Tần số')  
plt.title('Phân bố khối lượng phân tử')  
plt.show()

1. **Calculate LogP (Tính toán LogP):**

# English  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
import os  
  
base\_path = "../data"  
csv\_file = "chembl\_data.csv"  
csv\_path = os.path.join(base\_path, csv\_file)  
  
df = pd.read\_csv(csv\_path)  
df = df.dropna(subset=['canonical\_smiles'])  
  
def calculate\_logp(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 return Descriptors.MolLogP(mol)  
 else:  
 return None  
  
df['logp'] = df['canonical\_smiles'].apply(calculate\_logp)  
df = df.dropna(subset=['logp'])  
print(df[['canonical\_smiles', 'logp']].head())  
  
# Vietnamese  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
import os  
  
base\_path = "../data"  
csv\_file = "chembl\_data.csv"  
csv\_path = os.path.join(base\_path, csv\_file)  
  
df = pd.read\_csv(csv\_path)  
df = df.dropna(subset=['canonical\_smiles'])  
  
def calculate\_logp(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 return Descriptors.MolLogP(mol)  
 else:  
 return None  
  
df['logp'] = df['canonical\_smiles'].apply(calculate\_logp)  
df = df.dropna(subset=['logp'])  
print(df[['canonical\_smiles', 'logp']].head())

1. **Calculate TPSA (Tính toán TPSA):**

# English  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
import os  
  
base\_path = "../data"  
csv\_file = "chembl\_data.csv"  
csv\_path = os.path.join(base\_path, csv\_file)  
  
df = pd.read\_csv(csv\_path)  
df = df.dropna(subset=['canonical\_smiles'])  
  
def calculate\_tpsa(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 return Descriptors.TPSA(mol)  
 else:  
 return None  
  
df['tpsa'] = df['canonical\_smiles'].apply(calculate\_tpsa)  
df = df.dropna(subset=['tpsa'])  
print(df[['canonical\_smiles', 'tpsa']].head())  
  
# Vietnamese  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
import os  
  
base\_path = "../data"  
csv\_file = "chembl\_data.csv"  
csv\_path = os.path.join(base\_path, csv\_file)  
  
df = pd.read\_csv(csv\_path)  
df = df.dropna(subset=['canonical\_smiles'])  
  
def calculate\_tpsa(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 return Descriptors.TPSA(mol)  
 else:  
 return None  
  
df['tpsa'] = df['canonical\_smiles'].apply(calculate\_tpsa)  
df = df.dropna(subset=['tpsa'])  
print(df[['canonical\_smiles', 'tpsa']].head())

1. **Calculate the number of hydrogen bond donors and acceptors (Tính toán số lượng chất cho và nhận liên kết hydro):**

# English  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
import os  
  
base\_path = "../data"  
csv\_file = "chembl\_data.csv"  
csv\_path = os.path.join(base\_path, csv\_file)  
  
df = pd.read\_csv(csv\_path)  
df = df.dropna(subset=['canonical\_smiles'])  
  
def calculate\_hbd(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 return Descriptors.NumHDonors(mol)  
 else:  
 return None  
  
def calculate\_hba(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 return Descriptors.NumHAcceptors(mol)  
 else:  
 return None  
  
df['hbd'] = df['canonical\_smiles'].apply(calculate\_hbd)  
df['hba'] = df['canonical\_smiles'].apply(calculate\_hba)  
  
df = df.dropna(subset=['hbd', 'hba'])  
print(df[['canonical\_smiles', 'hbd', 'hba']].head())  
  
# Vietnamese  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
import os  
  
base\_path = "../data"  
csv\_file = "chembl\_data.csv"  
csv\_path = os.path.join(base\_path, csv\_file)  
  
df = pd.read\_csv(csv\_path)  
df = df.dropna(subset=['canonical\_smiles'])  
  
def calculate\_hbd(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 return Descriptors.NumHDonors(mol)  
 else:  
 return None  
  
def calculate\_hba(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 return Descriptors.NumHAcceptors(mol)  
 else:  
 return None  
  
df['hbd'] = df['canonical\_smiles'].apply(calculate\_hbd)  
df['hba'] = df['canonical\_smiles'].apply(calculate\_hba)  
  
df = df.dropna(subset=['hbd', 'hba'])  
print(df[['canonical\_smiles', 'hbd', 'hba']].head())

1. **Train a Random Forest Regressor model (Huấn luyện mô hình Random Forest Regressor):**

# English  
import os  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import AllChem  
from sklearn.model\_selection import train\_test\_split  
from sklearn.ensemble import RandomForestRegressor  
from sklearn.metrics import mean\_squared\_error  
  
# Define base path  
base\_path = "../data"  
csv\_file = "chembl\_data.csv" # Replace with your actual file name  
csv\_path = os.path.join(base\_path, csv\_file)  
  
# Load data from CSV  
try:  
 df = pd.read\_csv(csv\_path)  
except FileNotFoundError:  
 print(f"Error: The file {csv\_path} was not found.")  
 exit()  
  
# Data Preprocessing  
df = df.dropna(subset=['canonical\_smiles', 'standard\_value'])  
  
# Convert SMILES to features using RDKit (Morgan Fingerprints)  
def smiles\_to\_features(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 fp = AllChem.GetMorganFingerprintAsBitVect(mol, 2, nBits=1024)  
 return list(fp)  
 else:  
 return None  
  
df['features'] = df['canonical\_smiles'].apply(smiles\_to\_features)  
df = df.dropna(subset=['features'])  
  
# Prepare data for modeling  
X = list(df['features'])  
y = df['standard\_value'].astype(float)  
  
# 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)  
  
# Build and train the Random Forest Regressor 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  
import os  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import AllChem  
from sklearn.model\_selection import train\_test\_split  
from sklearn.ensemble import RandomForestRegressor  
from sklearn.metrics import mean\_squared\_error  
  
# Định nghĩa đường dẫn cơ sở  
base\_path = "../data"  
csv\_file = "chembl\_data.csv" # Thay thế bằng tên file thực tế của bạn  
csv\_path = os.path.join(base\_path, csv\_file)  
  
# Tải dữ liệu từ CSV  
try:  
 df = pd.read\_csv(csv\_path)  
except FileNotFoundError:  
 print(f"Lỗi: Không tìm thấy file {csv\_path}.")  
 exit()  
  
# Tiền xử lý dữ liệu  
df = df.dropna(subset=['canonical\_smiles', 'standard\_value'])  
  
# Chuyển đổi SMILES thành đặc trưng sử dụng RDKit (Morgan Fingerprints)  
def smiles\_to\_features(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol is not None:  
 fp = AllChem.GetMorganFingerprintAsBitVect(mol, 2, nBits=1024)  
 return list(fp)  
 else:  
 return None  
  
df['features'] = df['canonical\_smiles'].apply(smiles\_to\_features)  
df = df.dropna(subset=['features'])  
  
# Chuẩn bị dữ liệu cho mô hình  
X = list(df['features'])  
y = df['standard\_value'].astype(float)  
  
# 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)  
  
# Xây dựng và huấn luyện mô hình Random Forest Regressor  
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"Sai số bình phương trung bình: {mse}")

**Lưu ý quan trọng:**

* Hãy thay thế "chembl\_data.csv" bằng tên file CSV thực tế của bạn.
* Đảm bảo bạn đã cài đặt tất cả các thư viện cần thiết (pandas, rdkit, scikit-learn, matplotlib).
* Điều chỉnh các tham số của mô hình (ví dụ: n\_estimators trong RandomForestRegressor) để đạt được hiệu năng tốt nhất.

Tôi hy vọng những hướng dẫn và ví dụ này sẽ giúp bạn tiến xa hơn trong dự án của mình. Nếu bạn có bất kỳ câu hỏi nào, đừng ngần ngại hỏi!