# PharmApp Suite

## 🧠 AI for Drug Discovery and Development 🧪

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

## 🧩 Topic: Dự đoán hoạt tính sinh học từ các đặc trưng phân tử

Tuyệt vời! Dự án phân tích dữ liệu ChEMBL 35 kết hợp RDKit là một hướng đi rất tiềm năng trong nghiên cứu và phát triển thuốc. Dưới đây là phân tích chi tiết, hướng dẫn song ngữ và code mẫu để bạn tham khảo, tập trung vào chủ đề “Topic\_CheMBL\_35\_2”:

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

Chủ đề “Topic\_CheMBL\_35\_2” có thể tập trung vào nhiều khía cạnh khác nhau của dữ liệu ChEMBL 35. Để đưa ra một mô hình phân tích cụ thể, chúng ta cần xác định câu hỏi nghiên cứu (research question) mà bạn muốn trả lời. Dưới đây là một số gợi ý và hướng tiếp cận:

* **Dự đoán hoạt tính (Activity Prediction):** Xây dựng mô hình dự đoán hoạt tính của các hợp chất dựa trên cấu trúc hóa học của chúng.
  + **Phân tích:** Sử dụng các descriptor hóa học (tính chất vật lý, hóa học) được tính toán từ cấu trúc (ví dụ: MW, cLogP, HBA, HBD) làm đầu vào cho các mô hình machine learning (ví dụ: Random Forest, Support Vector Machine, Neural Networks).
  + **Mục tiêu:** Xác định các yếu tố cấu trúc quan trọng ảnh hưởng đến hoạt tính và dự đoán hoạt tính của các hợp chất mới.
* **Phân cụm hợp chất (Compound Clustering):** Phân nhóm các hợp chất có cấu trúc và/hoặc hoạt tính tương tự nhau.
  + **Phân tích:** Sử dụng các thuật toán clustering (ví dụ: k-means, hierarchical clustering) dựa trên các descriptor hóa học hoặc fingerprint cấu trúc.
  + **Mục tiêu:** Tìm kiếm các scaffold (khung cấu trúc) phổ biến, xác định các “druglike” compound và khám phá không gian hóa học.
* **Phân tích mối quan hệ cấu trúc-hoạt tính (Structure-Activity Relationship - SAR):** Nghiên cứu mối liên hệ giữa cấu trúc hóa học và hoạt tính sinh học của các hợp chất.
  + **Phân tích:** Sử dụng các phương pháp thống kê và machine learning để xác định các nhóm thế (substituents) quan trọng ảnh hưởng đến hoạt tính.
  + **Mục tiêu:** Tối ưu hóa cấu trúc của các hợp chất để tăng cường hoạt tính mong muốn.
* **Phân tích các đặc tính dược động học (ADMET Prediction):** Dự đoán các đặc tính hấp thụ, phân bố, chuyển hóa và thải trừ (ADMET) của các hợp chất.
  + **Phân tích:** Sử dụng các mô hình QSAR (Quantitative Structure-Activity Relationship) để dự đoán các đặc tính ADMET dựa trên cấu trúc hóa học.
  + **Mục tiêu:** Lọc các hợp chất có đặc tính ADMET không phù hợp trong giai đoạn đầu của quá trình phát triển thuốc.

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

Dưới đây là hướng dẫn chi tiết cho từng bước thực hiện, kèm theo giải thích bằng cả tiếng Anh và tiếng Việt:

**Bước 1: Truy vấn dữ liệu từ ChEMBL 35 (Querying Data from ChEMBL 35)**

* **Tiếng Anh:** Use PostgreSQL (psql) via pgAdmin to extract relevant data from the ChEMBL 35 database. Store the results in CSV files in the ../data/ directory.
* **Tiếng Việt:** Sử dụng PostgreSQL (psql) thông qua pgAdmin để trích xuất dữ liệu liên quan từ cơ sở dữ liệu ChEMBL 35. Lưu kết quả vào các file CSV trong thư mục ../data/.

**Bước 3: Xử lý dữ liệu và xây dựng mô hình (Data Processing and Model Building)**

* **Tiếng Anh:** Use Jupyter Notebook with Python and RDKit to process the data, calculate molecular descriptors, and build machine learning models.
* **Tiếng Việt:** Sử dụng Jupyter Notebook với Python và RDKit để xử lý dữ liệu, tính toán các descriptor phân tử và xây dựng các mô hình machine learning.

**Bước 4: Đánh giá và giải thích mô hình (Model Evaluation and Interpretation)**

* **Tiếng Anh:** Evaluate the performance of the models using appropriate metrics (e.g., AUC, RMSE, R-squared). Interpret the results to gain insights into the SAR or other relationships.
* **Tiếng Việt:** Đánh giá hiệu suất của các mô hình bằng cách sử dụng các chỉ số phù hợp (ví dụ: AUC, RMSE, R-squared). Giải thích kết quả để hiểu rõ hơn về SAR hoặc các mối quan hệ khác.

**3. Code mẫu (Code Examples)**

Dưới đây là 5 ví dụ code mẫu, bao gồm cả SQL và Python, minh họa các bước khác nhau trong quy trình phân tích:

**Ví dụ 1: SQL - Trích xuất dữ liệu hoạt tính (Extract Activity Data)**

-- English  
-- Extracting activity data for a specific target (e.g., target\_chembl\_id = 'CHEMBL205')  
SELECT  
 act.molregno,  
 act.standard\_value,  
 act.standard\_units,  
 act.standard\_type,  
 md.chembl\_id  
FROM  
 activities act  
JOIN  
 molecule\_dictionary md ON act.molregno = md.molregno  
WHERE  
 act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 AND act.relation = '='  
 AND act.target\_chembl\_id = 'CHEMBL205'  
 AND act.standard\_value IS NOT NULL  
 AND act.standard\_value > 0  
 AND act.standard\_value ~ '^[0-9\.]+$' --Fix lỗi a  
LIMIT 100;  
  
-- Vietnamese  
-- Trích xuất dữ liệu hoạt tính cho một mục tiêu cụ thể (ví dụ: target\_chembl\_id = 'CHEMBL205')  
SELECT  
 act.molregno,  
 act.standard\_value,  
 act.standard\_units,  
 act.standard\_type,  
 md.chembl\_id  
FROM  
 activities act  
JOIN  
 molecule\_dictionary md ON act.molregno = md.molregno  
WHERE  
 act.standard\_type = 'IC50'  
 AND act.standard\_units = 'nM'  
 AND act.relation = '='  
 AND act.target\_chembl\_id = 'CHEMBL205'  
 AND act.standard\_value IS NOT NULL  
 AND act.standard\_value > 0  
 AND act.standard\_value ~ '^[0-9\.]+$' --Fix lỗi a  
LIMIT 100;

**Ví dụ 2: Python - Đọc dữ liệu từ CSV và tính toán descriptor (Read CSV Data and Calculate Descriptors)**

# English  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
  
# Define the base path  
base\_path = "." # Replace with your actual base path  
  
# Construct the file path  
csv\_file\_path = os.path.join(base\_path, "data", "activity\_data.csv")  
  
# Read the CSV file  
df = pd.read\_csv(csv\_file\_path)  
  
# Function to calculate molecular weight  
def calculate\_mw(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol:  
 return Descriptors.MolWt(mol)  
 else:  
 return None  
  
# Apply the function to create a new column  
df['molecular\_weight'] = df['canonical\_smiles'].apply(calculate\_mw)  
  
# Print the first few rows with the new descriptor  
print(df.head())  
  
# Vietnamese  
import pandas as pd  
from rdkit import Chem  
from rdkit.Chem import Descriptors  
import os  
  
# Định nghĩa đường dẫn gốc  
base\_path = "." # Thay thế bằng đường dẫn gốc thực tế của bạn  
  
# Tạo đường dẫn đầy đủ đến file CSV  
csv\_file\_path = os.path.join(base\_path, "data", "activity\_data.csv")  
  
  
# Đọc file CSV  
df = pd.read\_csv(csv\_file\_path)  
  
# Hàm tính toán khối lượng phân tử  
def calculate\_mw(smiles):  
 mol = Chem.MolFromSmiles(smiles)  
 if mol:  
 return Descriptors.MolWt(mol)  
 else:  
 return None  
  
# Áp dụng hàm để tạo một cột mới  
df['molecular\_weight'] = df['canonical\_smiles'].apply(calculate\_mw)  
  
# In ra một vài dòng đầu tiên với descriptor mới  
print(df.head())

**Ví dụ 3: Python - Xây dựng mô hình Random Forest (Build Random Forest Model)**

# English  
from sklearn.model\_selection import train\_test\_split  
from sklearn.ensemble import RandomForestRegressor  
from sklearn.metrics import mean\_squared\_error, r2\_score  
import numpy as np  
  
# Assuming you have 'molecular\_weight' and 'standard\_value' columns  
# Prepare data  
df = df.dropna(subset=['molecular\_weight', 'standard\_value']) # Drop rows with missing values  
X = df[['molecular\_weight']] # Features  
y = np.log10(df['standard\_value']) # Target variable (log transformed)  
  
# Split data  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)  
  
# Train model  
model = RandomForestRegressor(n\_estimators=100, random\_state=42)  
model.fit(X\_train, y\_train)  
  
# Evaluate model  
y\_pred = model.predict(X\_test)  
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.model\_selection import train\_test\_split  
from sklearn.ensemble import RandomForestRegressor  
from sklearn.metrics import mean\_squared\_error, r2\_score  
import numpy as np  
  
# Giả sử bạn có các cột 'molecular\_weight' và 'standard\_value'  
# Chuẩn bị dữ liệu  
df = df.dropna(subset=['molecular\_weight', 'standard\_value']) # Loại bỏ các dòng có giá trị bị thiếu  
X = df[['molecular\_weight']] # Các đặc trưng  
y = np.log10(df['standard\_value']) # Biến mục tiêu (đã được logarit hóa)  
  
# Chia dữ liệu  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)  
  
# Huấn luyện mô hình  
model = RandomForestRegressor(n\_estimators=100, random\_state=42)  
model.fit(X\_train, y\_train)  
  
# Đánh giá mô hình  
y\_pred = model.predict(X\_test)  
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: Python - Phân cụm K-means (K-means Clustering)**

# English  
from sklearn.cluster import KMeans  
from sklearn.preprocessing import StandardScaler  
import matplotlib.pyplot as plt  
  
# Prepare data - using molecular weight and logP as example features  
df = df.dropna(subset=['molecular\_weight', 'alogp']) # Drop rows with missing values  
X = df[['molecular\_weight', 'alogp']]  
  
# Scale the data  
scaler = StandardScaler()  
X\_scaled = scaler.fit\_transform(X)  
  
# Determine optimal number of clusters using the Elbow method  
inertia = []  
for i in range(1, 11):  
 kmeans = KMeans(n\_clusters=i, random\_state=42,   
 n\_init='auto') # Explicitly set n\_init  
 kmeans.fit(X\_scaled)  
 inertia.append(kmeans.inertia\_)  
  
# Plot the Elbow method graph  
plt.plot(range(1, 11), inertia, marker='o')  
plt.title('Elbow Method for Optimal k')  
plt.xlabel('Number of Clusters')  
plt.ylabel('Inertia')  
plt.show()  
  
# Apply K-means clustering  
kmeans = KMeans(n\_clusters=3, random\_state=42,   
 n\_init='auto') # Explicitly set n\_init  
df['cluster'] = kmeans.fit\_predict(X\_scaled)  
  
print(df[['chembl\_id', 'cluster']].head())  
  
# Vietnamese  
from sklearn.cluster import KMeans  
from sklearn.preprocessing import StandardScaler  
import matplotlib.pyplot as plt  
  
# Chuẩn bị dữ liệu - sử dụng khối lượng phân tử và logP làm ví dụ  
df = df.dropna(subset=['molecular\_weight', 'alogp']) # Loại bỏ các dòng có giá trị bị thiếu  
X = df[['molecular\_weight', 'alogp']]  
  
# Chuẩn hóa dữ liệu  
scaler = StandardScaler()  
X\_scaled = scaler.fit\_transform(X)  
  
# Xác định số lượng cluster tối ưu bằng phương pháp Elbow  
inertia = []  
for i in range(1, 11):  
 kmeans = KMeans(n\_clusters=i, random\_state=42,   
 n\_init='auto') # Explicitly set n\_init  
 kmeans.fit(X\_scaled)  
 inertia.append(kmeans.inertia\_)  
  
# Vẽ biểu đồ Elbow method  
plt.plot(range(1, 11), inertia, marker='o')  
plt.title('Phương pháp Elbow để tìm k tối ưu')  
plt.xlabel('Số lượng Cluster')  
plt.ylabel('Inertia')  
plt.show()  
  
# Áp dụng phân cụm K-means  
kmeans = KMeans(n\_clusters=3, random\_state=42,   
 n\_init='auto') # Explicitly set n\_init  
df['cluster'] = kmeans.fit\_predict(X\_scaled)  
  
print(df[['chembl\_id', 'cluster']].head())

**Ví dụ 5: Python - Phân tích SAR đơn giản (Simple SAR Analysis)**

# English  
# Group by a specific structural feature (e.g., number of aromatic rings)  
df['num\_aromatic\_rings'] = df['canonical\_smiles'].apply(lambda x: Chem.MolFromSmiles(x).GetAromaticRingCount() if Chem.MolFromSmiles(x) else None)  
  
# Aggregate activity values for each group  
sar\_analysis = df.groupby('num\_aromatic\_rings')['standard\_value'].mean()  
  
print(sar\_analysis)  
  
# Vietnamese  
# Nhóm theo một đặc điểm cấu trúc cụ thể (ví dụ: số lượng vòng thơm)  
df['num\_aromatic\_rings'] = df['canonical\_smiles'].apply(lambda x: Chem.MolFromSmiles(x).GetAromaticRingCount() if Chem.MolFromSmiles(x) else None)  
  
# Tính toán giá trị hoạt tính trung bình cho mỗi nhóm  
sar\_analysis = df.groupby('num\_aromatic\_rings')['standard\_value'].mean()  
  
print(sar\_analysis)

**Lưu ý quan trọng:**

* **Lỗi a (Error a):** Lỗi “operator does not exist: numeric ~ unknown” xảy ra do bạn đang cố gắng so sánh một cột kiểu số (numeric) với một chuỗi (unknown). Để khắc phục, hãy đảm bảo rằng cột act.standard\_value thực sự là kiểu số. Nếu không, bạn có thể ép kiểu nó hoặc sử dụng một hàm khác để kiểm tra xem giá trị có phải là số hay không. Ví dụ: AND act.standard\_value::text ~ '^[0-9\.]+$'
* **Phiên bản scikit-learn (Scikit-learn version):** Nếu bạn đang sử dụng phiên bản scikit-learn cũ, hãy cân nhắc nâng cấp lên phiên bản mới hơn để hỗ trợ tham số squared=False trong hàm mean\_squared\_error. Nếu không thể nâng cấp, bạn có thể tính căn bậc hai của MSE để có được RMSE (Root Mean Squared Error).
* **Đường dẫn (Path):** Thay thế "./data/activity\_data.csv" bằng đường dẫn thực tế đến file CSV của bạn.

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