PharmApp Suite

🧠 AI for Drug Discovery and Development 💉

🧩 Topic: Trích xuất tín hiệu cấu trúc gây độc

Tuyệt vời! Tôi rất vui được hỗ trợ bạn trong dự án phân tích dữ liệu ChEMBL 35 này. Với vai trò là một chuyên gia R&D dược học, tôi hiểu rõ tầm quan trọng của việc khai thác dữ liệu ChEMBL để khám phá và phát triển thuốc mới. Chúng ta sẽ cùng nhau phân tích dữ liệu, xây dựng mô hình, và giải quyết các vấn đề kỹ thuật một cách hiệu quả.

Dưới đây là phân tích, hướng dẫn song ngữ, code SQL và Python mẫu, cùng với các ví dụ cụ thể để bạn bắt đầu:

1. Phân tích mô hình và hướng dẫn song ngữ (Analysis and Bilingual Guidance)

Mô hình phân tích (Analysis Model):

Chúng ta sẽ tập trung vào việc sử dụng dữ liệu ChEMBL 35 để xây dựng các mô hình dự đoán hoạt tính sinh học của các hợp chất. Cụ thể, chúng ta có thể thực hiện các bước sau:

1. **Trích xuất dữ liệu (Data Extraction):** Sử dụng SQL để truy vấn và trích xuất thông tin về các hợp chất, hoạt tính sinh học, và mục tiêu (targets) từ cơ sở dữ liệu ChEMBL 35.
2. **Tiền xử lý dữ liệu (Data Preprocessing):** Sử dụng RDKit để tính toán các descriptor phân tử (molecular descriptors) từ cấu trúc hóa học của các hợp chất. Làm sạch và chuẩn hóa dữ liệu hoạt tính sinh học.
3. **Phân tích khám phá (Exploratory Data Analysis - EDA):** Phân tích thống kê mô tả, trực quan hóa dữ liệu để hiểu rõ hơn về phân phối, tương quan giữa các biến, và xác định các vấn đề tiềm ẩn.
4. **Xây dựng mô hình (Model Building):** Sử dụng các thuật toán học máy (machine learning) như hồi quy tuyến tính (linear regression), máy vector hỗ trợ (support vector machines), hoặc mạng nơ-ron (neural networks) để xây dựng mô hình dự đoán hoạt tính.
5. **Đánh giá mô hình (Model Evaluation):** Đánh giá hiệu suất của mô hình bằng các độ đo thích hợp như RMSE, R-squared, hoặc AUC.
6. **Diễn giải mô hình (Model Interpretation):** Tìm hiểu các yếu tố cấu trúc nào đóng vai trò quan trọng trong việc dự đoán hoạt tính.

Hướng dẫn song ngữ (Bilingual Guidance):

- **English:** We will focus on using ChEMBL 35 data to build models that predict the biological activity of compounds. This involves data extraction, preprocessing, exploratory analysis, model building, evaluation, and interpretation.
- **Tiếng Việt:** Chúng ta sẽ tập trung vào việc sử dụng dữ liệu ChEMBL 35 để xây dựng các mô hình dự đoán hoạt tính sinh học của các hợp chất. Quá trình này bao gồm trích xuất dữ liệu, tiền xử lý, phân tích khám phá, xây dựng mô hình, đánh giá và diễn giải mô hình.

2. Code SQL, Python mẫu (Sample SQL and Python Code)

SQL (English):

```sql
-- SQL query to extract data for a specific target (e.g., a protein)
-- and filter for activity values.
-- Lấy dữ liệu cho một target cụ thể và lọc theo giá trị hoạt tính.
SELECT
    md.chembl_id,
    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.chembl_id = 'CHEMBL205'  -- Replace with your target of interest. Thay bằng
target bạn muốn.
    AND act.standard_type = 'IC50' -- Lọc theo loại hoạt tính (ví dụ: IC50)
    AND act.standard_value IS NOT NULL
    AND act.standard_value > 0  -- Giá trị hoạt tính phải dương
    AND act.standard_units = 'nM'
    AND act.standard_value::text ~ '^[0-9\.]+$' -- only numeric value
LIMIT 100;
```

Python (English):

```
# Python code to read the extracted CSV data, calculate molecular descriptors using
RDKit,
# and prepare the data for machine learning.
# Đọc dữ liệu CSV, tính toán descriptor phân tử bằng RDKit, và chuẩn bị dữ liệu cho
học máy.
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
import os

base_path = "." # current directory

# Load the CSV file into a pandas DataFrame
# Load file CSV vào DataFrame
csv_file_path = os.path.join(base_path, "data", "CHEMBL205_IC50.csv")  # Replace with
your file path. Thay bằng đường dẫn file của bạn.
df = pd.read_csv(csv_file_path)

# Function to calculate molecular descriptors using RDKit
# Hàm tính toán descriptor phân tử bằng RDKit
def calculate_descriptors(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        descriptors = {}
        for name, func in Descriptors.descList:
            try:
                descriptors[name] = func(mol)
            except:
                descriptors[name] = np.nan
        return pd.Series(descriptors)
    else:
        return pd.Series([np.nan] * len(Descriptors.descList), index=[name for name,
func in Descriptors.descList])
```

```python
# Apply the descriptor calculation function to the 'canonical_smiles' column
# Áp dụng hàm tính toán descriptor cho cột 'canonical_smiles'
descriptors_df = df['canonical_smiles'].apply(calculate_descriptors)

# Concatenate the descriptors with the original DataFrame
# Kết hợp descriptor với DataFrame ban đầu
df = pd.concat([df, descriptors_df], axis=1)

# Data cleaning: Handle missing values and convert activity values to pIC50
# Làm sạch dữ liệu: Xử lý giá trị thiếu và chuyển đổi giá trị hoạt tính sang pIC50
df = df.dropna()
df['pIC50'] = -np.log10(df['standard_value'] / 1e9)  # Convert nM to M and then to
pIC50

# Prepare data for machine learning
# Chuẩn bị dữ liệu cho học máy
X = df.iloc[:, df.columns.get_loc('MolWt'):].values  # Molecular descriptors as
features. Descriptor phân tử làm features
y = df['pIC50'].values  # pIC50 values as target. Giá trị pIC50 làm target

# Split data into training and testing sets
# 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)

# Train a linear regression model
# Huấn luyện mô hình hồi quy tuyến tính
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions on the test set
# Dự đoán trên tập kiểm tra
y_pred = model.predict(X_test)

# Evaluate the model
# Đánh giá mô hình
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)

print(f"RMSE: {rmse}")
print(f"R-squared: {r2}")
```

Lưu ý về lỗi (Note on Errors):

- **Lỗi SQL (SQL Error):** Lỗi `ERROR: operator does not exist: numeric ~ unknown` xảy ra khi bạn cố gắng so sánh một kiểu dữ liệu số với một chuỗi. Để khắc phục, hãy đảm bảo rằng cột `act.standard_value` đã được ép kiểu về kiểu số trước khi so sánh. Bạn có thể dùng `act.standard_value::text ~ '^[0-9\.]+$'` để kiểm tra xem giá trị có phải là số hay không trước khi ép kiểu.
- **Lỗi Scikit-learn (Scikit-learn Error):** Nếu bạn gặp lỗi liên quan đến tham số `squared=False` trong hàm `mean_squared_error`, hãy cập nhật phiên bản scikit-learn của bạn lên phiên bản mới nhất, hoặc loại bỏ tham số `squared=False` (mặc định là `True`, trả về MSE, không phải RMSE).

3. Ví dụ code SQL và Python mẫu (Sample SQL and Python Code Examples)

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

Ví dụ 1: Trích xuất dữ liệu cơ bản (Basic Data Extraction)

SQL:

```sql
-- Lấy chembl_id, smiles và IC50 của 100 hợp chất.
SELECT md.chembl_id, 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'
LIMIT 100;
```

Python:

```python
# Đọc dữ liệu từ file CSV và in ra 5 dòng đầu tiên.
import pandas as pd
csv_file_path = os.path.join(base_path, "data", "basic_data.csv")
df = pd.read_csv(csv_file_path)
print(df.head())
```

Ví dụ 2: Tính toán logP (Calculating LogP)

SQL: (Không cần thiết, logP được tính bằng Python)

Python:

```python
# Tính toán LogP cho các hợp chất.
from rdkit import Chem
from rdkit.Chem import AllChem, Descriptors

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)
print(df[['canonical_smiles', 'logP']].head())
```

Ví dụ 3: Lọc dữ liệu theo khoảng giá trị IC50 (Filtering by IC50 Range)

SQL:

```sql
-- Lấy các hợp chất có giá trị IC50 từ 100 đến 1000 nM.
SELECT md.chembl_id, 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_value BETWEEN 100 AND 1000
LIMIT 100;
```

Python:

```python
# Lọc các hợp chất có giá trị IC50 từ 100 đến 1000 nM sau khi đã đọc dữ liệu từ CSV.
df_filtered = df[(df['standard_value'] >= 100) & (df['standard_value'] <= 1000)]
print(df_filtered[['chembl_id', 'standard_value']].head())
```

Ví dụ 4: Tính toán TPSA (Calculating TPSA)

SQL: (Không cần thiết, TPSA được tính bằng Python)

Python:

```python
# Tính toán TPSA (Topological Polar Surface Area) cho các hợp chất.
from rdkit.Chem import Descriptors

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)
print(df[['canonical_smiles', 'TPSA']].head())
```

Ví dụ 5: Xây dựng mô hình hồi quy đơn giản (Simple Regression Model)

SQL: (Không cần thiết, mô hình được xây dựng bằng Python)

Python:

```python
# Xây dựng mô hình hồi quy tuyến tính đơn giản giữa LogP và pIC50.
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
import numpy as np

# Chuẩn bị dữ liệu (giả sử đã có cột 'logP' và 'pIC50')
df = df.dropna(subset=['logP', 'pIC50'])  # Loại bỏ các hàng có giá trị NaN
X = df[['logP']].values
y = df['pIC50'].values

# 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
model = LinearRegression()
model.fit(X_train, y_train)

# Dự đoán và đánh giá
y_pred = model.predict(X_test)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)

print(f"RMSE: {rmse}")
print(f"R-squared: {r2}")
```

Lưu ý quan trọng:

- Đảm bảo rằng bạn đã cài đặt tất cả các thư viện cần thiết (RDKit, Pandas, Scikit-learn, v.v.).
- Thay đổi đường dẫn file và các tham số khác cho phù hợp với dự án của bạn.
- Đây chỉ là những ví dụ cơ bản. Bạn có thể mở rộng và tùy chỉnh chúng để phù hợp với mục tiêu nghiên cứu của mình.

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