PharmApp Suite

🧠 AI for Drug Discovery and Development 💉

🧩 Topic: Phân tích thuốc thần kinh – tâm thần

Tuyệt vời! Với vai trò là chuyên gia R&D dược học, tôi rất vui được hỗ trợ bạn trong dự án phân tích dữ liệu ChEMBL 35. Dưới đây là phân tích chi tiết, hướng dẫn song ngữ và các ví dụ code để bạn tham khảo, tập trung vào chủ đề "Topic_CheMBL_35_54".

1. Phân tích mô hình (Analysis Model)

Chủ đề "Topic_CheMBL_35_54" có thể liên quan đến nhiều khía cạnh khác nhau trong phát triển thuốc, ví dụ:

- **Phân tích SAR/QSAR (Structure-Activity Relationship/Quantitative Structure-Activity Relationship):** Xác định mối liên hệ giữa cấu trúc hóa học của các phân tử và hoạt tính sinh học của chúng.
- **Phân tích sàng lọc ảo (Virtual Screening Analysis):** Sử dụng các mô hình máy tính để dự đoán khả năng liên kết của các phân tử với mục tiêu (target) và ưu tiên các ứng viên tiềm năng.
- **Phân tích tính chất dược động học (Pharmacokinetic Property Analysis):** Nghiên cứu cách cơ thể hấp thụ, phân phối, chuyển hóa và thải trừ thuốc (ADME).
- **Phân tích độc tính (Toxicity Analysis):** Đánh giá khả năng gây hại của các phân tử đối với cơ thể.

Mô hình phân tích đề xuất:

Dựa trên kinh nghiệm của tôi, tôi đề xuất một quy trình phân tích kết hợp các bước sau:

1. **Thu thập và chuẩn bị dữ liệu (Data Collection and Preparation):**
    - Truy vấn dữ liệu từ cơ sở dữ liệu ChEMBL 35 bằng SQL.
    - Sử dụng RDKit để tính toán các descriptor phân tử (ví dụ: trọng lượng phân tử, logP, diện tích bề mặt phân cực).
    - Làm sạch và tiền xử lý dữ liệu (xử lý giá trị thiếu, loại bỏ outlier).
2. **Phân tích mô tả (Descriptive Analysis):**
    - Thống kê mô tả các descriptor và hoạt tính sinh học.
    - Trực quan hóa dữ liệu (ví dụ: biểu đồ phân tán, biểu đồ hộp) để khám phá các xu hướng và mối quan hệ tiềm năng.
3. **Xây dựng mô hình SAR/QSAR (SAR/QSAR Model Building):**
    - Chọn các descriptor phù hợp làm đầu vào cho mô hình.
    - Sử dụng các thuật toán học máy (ví dụ: hồi quy tuyến tính, cây quyết định, mạng nơ-ron) để xây dựng mô hình dự đoán hoạt tính.
    - Đánh giá hiệu suất của mô hình bằng các chỉ số phù hợp (ví dụ: R-squared, RMSE, AUC).
4. **Giải thích mô hình và rút ra kết luận (Model Interpretation and Conclusion):**
    - Xác định các descriptor quan trọng ảnh hưởng đến hoạt tính.
    - Đề xuất các hướng đi tiềm năng để tối ưu hóa cấu trúc phân tử.

2. Hướng dẫn song ngữ (Bilingual Guidance)

- **Tiếng Anh (English):**
    - **Data Retrieval:** Use SQL queries to extract relevant data from the ChEMBL 35 database.

- **Molecular Descriptors:** Utilize RDKit to compute molecular descriptors that capture the structural and physicochemical properties of the compounds.
- **Model Building:** Employ machine learning algorithms to build predictive models that relate molecular descriptors to biological activity.
- **Model Validation:** Evaluate the performance of the models using appropriate metrics and validation techniques.
- **Interpretation:** Interpret the models to identify key structural features that influence activity and guide further optimization efforts.

- **Tiếng Việt (Vietnamese):**
  - **Truy xuất dữ liệu:** Sử dụng truy vấn SQL để trích xuất dữ liệu liên quan từ cơ sở dữ liệu ChEMBL 35.
  - **Descriptor phân tử:** Sử dụng RDKit để tính toán các descriptor phân tử, nắm bắt các đặc tính cấu trúc và lý hóa của các hợp chất.
  - **Xây dựng mô hình:** Sử dụng các thuật toán học máy để xây dựng các mô hình dự đoán liên hệ giữa các descriptor phân tử và hoạt tính sinh học.
  - **Xác thực mô hình:** Đánh giá hiệu suất của các mô hình bằng cách sử dụng các số liệu và kỹ thuật xác thực phù hợp.
  - **Giải thích:** Giải thích các mô hình để xác định các đặc điểm cấu trúc chính ảnh hưởng đến hoạt động và hướng dẫn các nỗ lực tối ưu hóa hơn nữa.

3. Code mẫu (Code Examples)

3.1 SQL (lấy 100 dòng dữ liệu, tránh lỗi `numeric ~ unknown`)

```sql
-- English
-- Select 100 rows of data, ensuring that standard_value is numeric
SELECT
    act.molregno,
    act.standard_value,
    act.standard_units,
    cmp.smiles
FROM
    activities act
JOIN
    molecule_dictionary cmp ON act.molregno = cmp.molregno
WHERE
    act.standard_type = 'IC50'  -- Example: Filter by IC50 values
    AND act.standard_relation = '='
    AND act.standard_value IS NOT NULL
    AND act.standard_value::TEXT ~ '^[0-9\.]+$' -- Ensure numeric value
LIMIT 100;

-- Vietnamese
-- Chọn 100 dòng dữ liệu, đảm bảo rằng standard_value là số
SELECT
    act.molregno,
    act.standard_value,
    act.standard_units,
    cmp.smiles
FROM
    activities act
JOIN
    molecule_dictionary cmp ON act.molregno = cmp.molregno
WHERE
    act.standard_type = 'IC50'  -- Ví dụ: Lọc theo giá trị IC50
    AND act.standard_relation = '='
    AND act.standard_value IS NOT NULL
```

```
    AND act.standard_value::TEXT ~ '^[0-9\.]+$' -- Đảm bảo giá trị là số
LIMIT 100;
```

3.2 Python (Jupyter Notebook)

```
# 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

# Define base path
base_path = "../data"  # Adjust this path

# Load data from CSV
data_file = "chembl_ic50_data.csv"  # Replace with your CSV file name
data_path = os.path.join(base_path, data_file)
df = pd.read_csv(data_path)

# Function to calculate molecular descriptors using RDKit
def calculate_descriptors(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is None:
        return None
    descriptors = {}
    descriptors["MolWt"] = Descriptors.MolWt(mol)
    descriptors["LogP"] = Descriptors.MolLogP(mol)
    descriptors["HBD"] = Descriptors.NumHDonors(mol)
    descriptors["HBA"] = Descriptors.NumHAcceptors(mol)
    return descriptors

# Apply descriptor calculation to the DataFrame
df['descriptors'] = df['smiles'].apply(calculate_descriptors)
df = df.dropna(subset=['descriptors'])  # Drop rows with None descriptors

# Convert descriptors to columns
df = pd.concat([df.drop(['descriptors'], axis=1), df['descriptors'].apply(pd.Series)],
axis=1)

# Prepare data for modeling
X = df[["MolWt", "LogP", "HBD", "HBA"]]  # Use calculated descriptors
y = df["standard_value"].astype(float)  # Ensure numeric type

# Remove infinite or NaN values
X = X[~np.isinf(X).any(axis=1)]
y = y[~np.isinf(y)]
X = X.dropna()
y = y.dropna()

# 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)
```

```python
# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred) # No squared=False needed
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")

# Vietnamese
# Nhập các thư viện cần thiết
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

# Định nghĩa đường dẫn gốc
base_path = "../data"  # Điều chỉnh đường dẫn này

# Tải dữ liệu từ file CSV
data_file = "chembl_ic50_data.csv"  # Thay thế bằng tên file CSV của bạn
data_path = os.path.join(base_path, data_file)
df = pd.read_csv(data_path)

# Hàm tính toán descriptor phân tử bằng RDKit
def calculate_descriptors(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is None:
        return None
    descriptors = {}
    descriptors["MolWt"] = Descriptors.MolWt(mol)
    descriptors["LogP"] = Descriptors.MolLogP(mol)
    descriptors["HBD"] = Descriptors.NumHDonors(mol)
    descriptors["HBA"] = Descriptors.NumHAcceptors(mol)
    return descriptors

# Áp dụng tính toán descriptor cho DataFrame
df['descriptors'] = df['smiles'].apply(calculate_descriptors)
df = df.dropna(subset=['descriptors'])  # Loại bỏ các hàng có descriptor là None

# Chuyển đổi descriptor thành các cột
df = pd.concat([df.drop(['descriptors'], axis=1), df['descriptors'].apply(pd.Series)],
axis=1)

# Chuẩn bị dữ liệu cho mô hình hóa
X = df[["MolWt", "LogP", "HBD", "HBA"]]  # Sử dụng các descriptor đã tính
y = df["standard_value"].astype(float)  # Đảm bảo kiểu số

# Loại bỏ các giá trị vô cực hoặc NaN
X = X[~np.isinf(X).any(axis=1)]
y = y[~np.isinf(y)]
X = X.dropna()
y = y.dropna()
```

```
# 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) # Không cần squared=False
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

4. Năm ví dụ code (Five code examples)

Ví dụ 1: Tính số lượng vòng (ring count)

```
-- SQL
SELECT molregno, smiles, ring_count FROM molecule_dictionary LIMIT 5;
```

```
# Python
from rdkit import Chem
from rdkit.Chem import rdMolDescriptors

def get_ring_count(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol:
        return rdMolDescriptors.CalcNumRings(mol)
    else:
        return None

df['ring_count'] = df['smiles'].apply(get_ring_count)
```

Ví dụ 2: Lọc theo trọng lượng phân tử (molecular weight)

```
-- SQL
SELECT molregno, smiles, mol_weight FROM molecule_dictionary WHERE mol_weight > 500
LIMIT 5;
```

```
# Python
df_filtered = df[df['MolWt'] > 500]
```

Ví dụ 3: Tính TPSA (Topological Polar Surface Area)

```
# Python
from rdkit.Chem import Descriptors3D

def calculate_tpsa(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol:
        return Descriptors3D.TPSA(mol)
    else:
        return None

df['tpsa'] = df['smiles'].apply(calculate_tpsa)
```

Ví dụ 4: Phân tích tần suất của các đơn vị cấu trúc (frequent substructure analysis)

```
# Python
from rdkit.Chem import AllChem
from collections import defaultdict

def frequent_substructures(smiles_list, min_support=0.2):
    mols = [Chem.MolFromSmiles(s) for s in smiles_list]
    patterns = defaultdict(int)
    for mol in mols:
        if mol:
            info = {}
            fp = AllChem.GetMorganFingerprint(mol, 2, bitInfo=info)
            for bit, atom_radius in info.items():
                env = Chem.FindAtomEnvironmentOfRadiusN(mol, atom_radius[1],
atom_radius[0])
                amap = {}
                submol = Chem.PathToSubmol(mol, env, atomMap=amap)
                 Smiles = Chem.MolToSmiles(submol)
                patterns[Smiles] += 1

    # Filter by minimum support
    filtered_patterns = {k: v for k, v in patterns.items() if v / len(smiles_list) >=
min_support}
    return filtered_patterns

# Get a list of SMILES strings from your DataFrame
smiles_list = df['smiles'].tolist()

# Find frequent substructures
frequent_patterns = frequent_substructures(smiles_list)

print("Frequent Substructures:")
for pattern, count in frequent_patterns.items():
    print(f"Substructure: {pattern}, Support: {count / len(smiles_list):.2f}")
```

Ví dụ 5: Tính số lượng liên kết quay tự do (number of rotatable bonds)

```
# Python
from rdkit.Chem import Descriptors

def calculate_rotatable_bonds(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol:
        return Descriptors.NumRotatableBonds(mol)
    else:
        return None

df['rotatable_bonds'] = df['smiles'].apply(calculate_rotatable_bonds)
```

Lưu ý quan trọng:

- **Đường dẫn:** Hãy chắc chắn rằng đường dẫn `base_path` trong code Python trỏ đến đúng thư mục chứa dữ liệu của bạn.
- **Tên file:** Thay thế `"chembl_ic50_data.csv"` bằng tên file CSV thực tế của bạn.
- **Chọn descriptor:** Lựa chọn descriptor phân tử phù hợp phụ thuộc vào mục tiêu cụ thể của bạn. Hãy tham khảo các tài liệu về SAR/QSAR để biết thêm thông tin.
- **Điều chỉnh tham số:** Các tham số như `test_size` trong `train_test_split` và `min_support` trong `frequent_substructures` có thể cần được điều chỉnh để phù hợp với dữ liệu của bạn.

- **Xử lý lỗi:** Luôn kiểm tra và xử lý các lỗi tiềm ẩn (ví dụ: phân tử không hợp lệ, giá trị thiếu) để đảm bảo tính ổn định của quy trình phân tích.
- **Phiên bản thư viện:** Đảm bảo rằng bạn đã cài đặt các phiên bản tương thích của các thư viện (RDKit, scikit-learn, pandas, numpy).

Chúc bạn thành công với dự án phân tích dữ liệu ChEMBL 35! Nếu bạn có bất kỳ câu hỏi nào khác, đừng ngần ngại hỏi.