PharmApp Suite

🧠 AI for Drug Discovery and Development 💉

🧩 Topic: Dự đoán tương tác thuốc–đích bằng học máy

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 tìm kiếm các hợp chất tiềm năng. Chúng ta sẽ cùng nhau xây dựng một quy trình phân tích hiệu quả, tuân thủ theo chuẩn AIMLOps và giải quyết các vấn đề phát sinh.

1. Phân tích mô hình (Analysis Model)

Dựa trên thông tin bạn cung cấp, chúng ta sẽ xây dựng một quy trình phân tích dữ liệu ChEMBL 35 sử dụng PostgreSQL, RDKit và Jupyter Notebook. Mô hình phân tích sẽ bao gồm các bước sau:

- **Data Extraction (Trích xuất dữ liệu):** Sử dụng SQL queries để trích xuất thông tin cần thiết từ database ChEMBL 35. Các thông tin này có thể bao gồm cấu trúc hóa học, hoạt tính sinh học, thuộc tính vật lý hóa học, v.v.
- **Data Cleaning and Preprocessing (Làm sạch và tiền xử lý dữ liệu):** Loại bỏ các dữ liệu không hợp lệ, xử lý các giá trị thiếu, và chuẩn hóa dữ liệu.
- **Feature Engineering (Xây dựng đặc trưng):** Sử dụng RDKit để tính toán các descriptor phân tử (ví dụ: MW, LogP, TPSA, số lượng liên kết, vòng, v.v.) từ cấu trúc hóa học của các hợp chất.
- **Data Analysis and Modeling (Phân tích và mô hình hóa dữ liệu):** Sử dụng các kỹ thuật thống kê và machine learning để phân tích mối quan hệ giữa các descriptor phân tử và hoạt tính sinh học. Các mô hình có thể được sử dụng để dự đoán hoạt tính của các hợp chất mới, xác định các scaffold tiềm năng, và tối ưu hóa cấu trúc phân tử.
- **Visualization and Interpretation (Trực quan hóa và giải thích):** Sử dụng các công cụ trực quan hóa dữ liệu để hiểu rõ hơn về kết quả phân tích và trình bày thông tin một cách trực quan.

2. Hướng dẫn song ngữ (Bilingual Guidance)

Tôi sẽ cung cấp hướng dẫn chi tiết bằng cả tiếng Anh và tiếng Việt cho từng bước trong quy trình phân tích.

3. Code SQL và Python (SQL and Python Code)

Tôi sẽ cung cấp các ví dụ code SQL và Python để bạn có thể bắt đầu phân tích dữ liệu ChEMBL 35.

4. Ví dụ Code (Code Examples)

Tôi sẽ cung cấp 5 ví dụ code SQL và Python mẫu để bạn tham khảo.

Giải quyết các lỗi

- **Lỗi a:** ERROR: operator does not exist: numeric ~ unknown, LINE 12: AND act.standard_value ~ '^[0-9\.]+$'

  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 cần ép kiểu cột `standard_value` sang kiểu text trước khi so sánh với regular expression.

```sql
-- English
SELECT *
FROM activities act
WHERE act.standard_type = 'IC50'
```

```
  AND act.standard_relation = '='
  AND act.standard_value::text ~ '^[0-9\.]+$'
LIMIT 100;

-- Vietnamese
SELECT *
FROM activities act
WHERE act.standard_type = 'IC50'
  AND act.standard_relation = '='
  AND CAST(act.standard_value AS TEXT) ~ '^[0-9\.]+$'
LIMIT 100;
```

- **Lỗi b:** phiên bản scikit-learn cũ không hỗ trợ tham số squared=False trong hàm mean_squared_error

  Bạn cần cập nhật phiên bản scikit-learn của bạn lên phiên bản mới nhất hoặc loại bỏ tham số `squared=False` nếu không cần thiết.

```
# English
from sklearn.metrics import mean_squared_error

# If you can't update scikit-learn:
mse = mean_squared_error(y_true, y_pred)
rmse = mse**0.5

# If you can update scikit-learn:
rmse = mean_squared_error(y_true, y_pred, squared=False)

# Vietnamese
from sklearn.metrics import mean_squared_error

# Nếu bạn không thể cập nhật scikit-learn:
mse = mean_squared_error(y_true, y_pred)
rmse = mse**0.5

# Nếu bạn có thể cập nhật scikit-learn:
rmse = mean_squared_error(y_true, y_pred, squared=False)
```

Ví dụ Code (Code Examples)

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

SQL Examples

1. **Lấy thông tin cơ bản về các hợp chất có hoạt tính IC50:**

```
-- English
SELECT mol.molregno, act.standard_value, act.standard_units
FROM molecule_dictionary mol
JOIN activities act ON mol.molregno = act.molregno
WHERE act.standard_type = 'IC50'
LIMIT 100;

-- Vietnamese
SELECT mol.molregno, act.standard_value, act.standard_units
FROM molecule_dictionary mol
JOIN activities act ON mol.molregno = act.molregno
WHERE act.standard_type = 'IC50'
LIMIT 100;
```

2. **Lấy cấu trúc SMILES của các hợp chất:**

```sql
-- English
SELECT mol.molregno, cs.canonical_smiles
FROM molecule_dictionary mol
JOIN compound_structures cs ON mol.molregno = cs.molregno
LIMIT 100;

-- Vietnamese
SELECT mol.molregno, cs.canonical_smiles
FROM molecule_dictionary mol
JOIN compound_structures cs ON mol.molregno = cs.molregno
LIMIT 100;
```

3. **Lọc các hợp chất có hoạt tính IC50 dưới 100 nM:**

```sql
-- English
SELECT mol.molregno, act.standard_value, act.standard_units
FROM molecule_dictionary mol
JOIN activities act ON mol.molregno = act.molregno
WHERE act.standard_type = 'IC50'
  AND act.standard_value < 100
  AND act.standard_units = 'nM'
LIMIT 100;

-- Vietnamese
SELECT mol.molregno, act.standard_value, act.standard_units
FROM molecule_dictionary mol
JOIN activities act ON mol.molregno = act.molregno
WHERE act.standard_type = 'IC50'
  AND act.standard_value < 100
  AND act.standard_units = 'nM'
LIMIT 100;
```

4. **Tìm kiếm các hợp chất tương tự (similarity search) dựa trên cấu trúc SMILES:**

```sql
-- English
SELECT mol.molregno, cs.canonical_smiles
FROM molecule_dictionary mol
JOIN compound_structures cs ON mol.molregno = cs.molregno
WHERE morganbv_fp(cs.canonical_smiles) % morganbv_fp('CCOc1ccccc1Oc2ccccc2')
LIMIT 100;

-- Vietnamese
SELECT mol.molregno, cs.canonical_smiles
FROM molecule_dictionary mol
JOIN compound_structures cs ON mol.molregno = cs.molregno
WHERE morganbv_fp(cs.canonical_smiles) % morganbv_fp('CCOc1ccccc1Oc2ccccc2')
LIMIT 100;
```

5. **Đếm số lượng hợp chất cho mỗi standard_type:**

```sql
-- English
SELECT standard_type, COUNT(*)
FROM activities
GROUP BY standard_type
LIMIT 100;

-- Vietnamese
SELECT standard_type, COUNT(*)
FROM activities
GROUP BY standard_type
LIMIT 100;
```

Python Examples

1. **Kết nối đến database ChEMBL và lấy dữ liệu:**

```
# English
import psycopg2
import pandas as pd

# Database credentials
db_host = "192.168.206.136"
db_user = "rd"
db_pass = "rd"
db_name = "chembl_35"

# Connect to the database
conn = psycopg2.connect(host=db_host, user=db_user, password=db_pass,
database=db_name)

# SQL query
query = "SELECT mol.molregno, act.standard_value FROM molecule_dictionary mol
JOIN activities act ON mol.molregno = act.molregno WHERE act.standard_type =
'IC50' LIMIT 100;"

# Read data into a pandas DataFrame
df = pd.read_sql_query(query, conn)

# Close the connection
conn.close()

# Print the DataFrame
print(df.head())

# Vietnamese
import psycopg2
import pandas as pd

# Thông tin đăng nhập database
db_host = "192.168.206.136"
db_user = "rd"
db_pass = "rd"
db_name = "chembl_35"

# Kết nối đến database
conn = psycopg2.connect(host=db_host, user=db_user, password=db_pass,
database=db_name)

# Câu truy vấn SQL
query = "SELECT mol.molregno, act.standard_value FROM molecule_dictionary mol
JOIN activities act ON mol.molregno = act.molregno WHERE act.standard_type =
'IC50' LIMIT 100;"

# Đọc dữ liệu vào DataFrame của pandas
df = pd.read_sql_query(query, conn)

# Đóng kết nối
conn.close()

# In DataFrame
print(df.head())
```

2. **Tính toán descriptor phân tử sử dụng RDKit:**

```
# English
from rdkit import Chem
from rdkit.Chem import Descriptors

def calculate_descriptors(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        mw = Descriptors.MolWt(mol)
        logp = Descriptors.MolLogP(mol)
        return mw, logp
    else:
        return None, None

# Example usage
smiles = "CCOc1ccccc1Oc2ccccc2"
mw, logp = calculate_descriptors(smiles)
print(f"Molecular Weight: {mw}, LogP: {logp}")

# Vietnamese
from rdkit import Chem
from rdkit.Chem import Descriptors

def tinh_toan_descriptors(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        mw = Descriptors.MolWt(mol)
        logp = Descriptors.MolLogP(mol)
        return mw, logp
    else:
        return None, None

# Ví dụ sử dụng
smiles = "CCOc1ccccc1Oc2ccccc2"
mw, logp = tinh_toan_descriptors(smiles)
print(f"Khối lượng phân tử: {mw}, LogP: {logp}")
```

3. **Kết hợp dữ liệu từ database và descriptor phân tử:**

```
# English
import psycopg2
import pandas as pd
from rdkit import Chem
from rdkit.Chem import Descriptors

# Database credentials
db_host = "192.168.206.136"
db_user = "rd"
db_pass = "rd"
db_name = "chembl_35"

# Connect to the database
conn = psycopg2.connect(host=db_host, user=db_user, password=db_pass,
database=db_name)

# SQL query
query = "SELECT mol.molregno, cs.canonical_smiles, act.standard_value FROM
molecule_dictionary mol JOIN compound_structures cs ON mol.molregno =
cs.molregno JOIN activities act ON mol.molregno = act.molregno WHERE
```

```
act.standard_type = 'IC50' LIMIT 100;"

# Read data into a pandas DataFrame
df = pd.read_sql_query(query, conn)

# Close the connection
conn.close()

# Function to calculate descriptors
def calculate_descriptors(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        mw = Descriptors.MolWt(mol)
        logp = Descriptors.MolLogP(mol)
        return mw, logp
    else:
        return None, None

# Apply the function to each SMILES string
df[['mw', 'logp']] = df['canonical_smiles'].apply(lambda x:
pd.Series(calculate_descriptors(x)))

# Print the DataFrame
print(df.head())

# Vietnamese
import psycopg2
import pandas as pd
from rdkit import Chem
from rdkit.Chem import Descriptors

# Thông tin đăng nhập database
db_host = "192.168.206.136"
db_user = "rd"
db_pass = "rd"
db_name = "chembl_35"

# Kết nối đến database
conn = psycopg2.connect(host=db_host, user=db_user, password=db_pass,
database=db_name)

# Câu truy vấn SQL
query = "SELECT mol.molregno, cs.canonical_smiles, act.standard_value FROM
molecule_dictionary mol JOIN compound_structures cs ON mol.molregno =
cs.molregno JOIN activities act ON mol.molregno = act.molregno WHERE
act.standard_type = 'IC50' LIMIT 100;"

# Đọc dữ liệu vào DataFrame của pandas
df = pd.read_sql_query(query, conn)

# Đóng kết nối
conn.close()

# Hàm tính toán descriptors
def tinh_toan_descriptors(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        mw = Descriptors.MolWt(mol)
        logp = Descriptors.MolLogP(mol)
```

```
        return mw, logp
    else:
        return None, None

# Áp dụng hàm cho mỗi chuỗi SMILES
df[['mw', 'logp']] = df['canonical_smiles'].apply(lambda x:
pd.Series(tinh_toan_descriptors(x)))

# In DataFrame
print(df.head())
```

4. **Xây dựng mô hình machine learning để dự đoán hoạt tính:**

```
# English
import psycopg2
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

# Database credentials
db_host = "192.168.206.136"
db_user = "rd"
db_pass = "rd"
db_name = "chembl_35"

# Connect to the database
conn = psycopg2.connect(host=db_host, user=db_user, password=db_pass,
database=db_name)

# SQL query
query = "SELECT mol.molregno, cs.canonical_smiles, act.standard_value FROM
molecule_dictionary mol JOIN compound_structures cs ON mol.molregno =
cs.molregno JOIN activities act ON mol.molregno = act.molregno WHERE
act.standard_type = 'IC50' AND act.standard_value IS NOT NULL LIMIT 100;"

# Read data into a pandas DataFrame
df = pd.read_sql_query(query, conn)

# Close the connection
conn.close()

# Function to calculate descriptors
def calculate_descriptors(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        mw = Descriptors.MolWt(mol)
        logp = Descriptors.MolLogP(mol)
        return mw, logp
    else:
        return None, None

# Apply the function to each SMILES string
df[['mw', 'logp']] = df['canonical_smiles'].apply(lambda x:
pd.Series(calculate_descriptors(x)))

# Drop rows with NaN values
df = df.dropna()
```

```
# Prepare data for machine learning
X = df[['mw', 'logp']]
y = df['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)

# Create a linear regression model
model = LinearRegression()

# Train the model
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
rmse = mean_squared_error(y_test, y_pred, squared=False)
print(f"Root Mean Squared Error: {rmse}")

# Vietnamese
import psycopg2
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

# Thông tin đăng nhập database
db_host = "192.168.206.136"
db_user = "rd"
db_pass = "rd"
db_name = "chembl_35"

# Kết nối đến database
conn = psycopg2.connect(host=db_host, user=db_user, password=db_pass,
database=db_name)

# Câu truy vấn SQL
query = "SELECT mol.molregno, cs.canonical_smiles, act.standard_value FROM
molecule_dictionary mol JOIN compound_structures cs ON mol.molregno =
cs.molregno JOIN activities act ON mol.molregno = act.molregno WHERE
act.standard_type = 'IC50' AND act.standard_value IS NOT NULL LIMIT 100;"

# Đọc dữ liệu vào DataFrame của pandas
df = pd.read_sql_query(query, conn)

# Đóng kết nối
conn.close()

# Hàm tính toán descriptors
def tinh_toan_descriptors(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        mw = Descriptors.MolWt(mol)
        logp = Descriptors.MolLogP(mol)
```

```
            return mw, logp
        else:
            return None, None

    # Áp dụng hàm cho mỗi chuỗi SMILES
    df[['mw', 'logp']] = df['canonical_smiles'].apply(lambda x:
    pd.Series(tinh_toan_descriptors(x)))

    # Loại bỏ các hàng có giá trị NaN
    df = df.dropna()

    # Chuẩn bị dữ liệu cho machine learning
    X = df[['mw', 'logp']]
    y = df['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)

    # Tạo mô hình hồi quy tuyến tính
    model = LinearRegression()

    # Huấn luyện mô hình
    model.fit(X_train, y_train)

    # Dự đoán
    y_pred = model.predict(X_test)

    # Đánh giá mô hình
    rmse = mean_squared_error(y_test, y_pred, squared=False)
    print(f"Sai số bình phương gốc trung bình: {rmse}")
```

5. **Trực quan hóa kết quả:**

```
    # English
    import matplotlib.pyplot as plt

    # Scatter plot of predicted vs. actual values
    plt.scatter(y_test, y_pred)
    plt.xlabel("Actual Values")
    plt.ylabel("Predicted Values")
    plt.title("Predicted vs. Actual Values")
    plt.show()

    # Vietnamese
    import matplotlib.pyplot as plt

    # Biểu đồ phân tán giữa giá trị thực tế và giá trị dự đoán
    plt.scatter(y_test, y_pred)
    plt.xlabel("Giá trị thực tế")
    plt.ylabel("Giá trị dự đoán")
    plt.title("Giá trị dự đoán so với giá trị thực tế")
    plt.show()
```

Lưu ý:

- Bạn cần cài đặt các thư viện cần thiết (psycopg2, pandas, rdkit, scikit-learn, matplotlib) trước khi chạy code Python.
- Bạn cần thay đổi thông tin đăng nhập database cho phù hợp với môi trường của bạn.

- Các ví dụ code trên chỉ mang tính chất minh họa. Bạn cần điều chỉnh code cho phù hợp với mục tiêu phân tích của bạn.
- Việc lựa chọn mô hình machine learning phù hợp phụ thuộc vào dữ liệu và mục tiêu của bạn. Bạn có thể thử nghiệm với các mô hình khác nhau để tìm ra mô hình tốt nhất.

Chúc bạn thành công với dự án của mình! Hãy cho tôi biết nếu bạn có bất kỳ câu hỏi nào.