PharmApp Suite

🧠 AI for Drug Discovery and Development 💉

🧩 Topic: Xây dựng mô hình QSAR từ dữ liệu IC50 trong ChEMBL

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 nghiên cứu và phát triển dược học, tôi sẽ cung cấp cho bạn những phân tích, hướng dẫn và code mẫu để bạn có thể khai thác tối đa dữ liệu ChEMBL 35 phục vụ cho nghiên cứu và phát triển thuốc của mình.

Tổng quan dự án:

Dự án của bạn tập trung vào việc phân tích dữ liệu từ cơ sở dữ liệu ChEMBL 35, kết hợp với RDKit, để hỗ trợ các hoạt động nghiên cứu và phát triển thuốc. Bạn sử dụng PostgreSQL (psql) để lưu trữ dữ liệu và Jupyter Notebook để phân tích. Cấu trúc thư mục tuân theo chuẩn AIMLOps, giúp đảm bảo tính tổ chức và khả năng tái sử dụng của dự án.

Mô hình phân tích (Analysis Model):

Chúng ta có thể xây dựng một quy trình phân tích dữ liệu ChEMBL 35 theo các bước sau:

1. **Data Extraction (Trích xuất dữ liệu):** Sử dụng SQL để truy vấn và trích xuất dữ liệu cần thiết từ cơ sở dữ liệu ChEMBL 35. Dữ liệu này có thể bao gồm thông tin về các hợp chất, hoạt tính sinh học của chúng, các mục tiêu (targets) mà chúng tác động, và các thuộc tính lý hóa.
2. **Data Cleaning and Preprocessing (Làm sạch và tiền xử lý dữ liệu):**
    - Xử lý các giá trị thiếu (missing values).
    - Chuẩn hóa dữ liệu (ví dụ: chuyển đổi các giá trị hoạt tính về cùng một đơn vị).
    - Lọc bỏ các dữ liệu không hợp lệ hoặc không đáng tin cậy.
3. **Feature Engineering (Xây dựng đặc trưng):** Sử dụng RDKit để tính toán các đặc trưng phân tử từ cấu trúc hóa học của các hợp chất. Các đặc trưng này có thể bao gồm:
    - Các đặc trưng hình thái (ví dụ: diện tích bề mặt, thể tích).
    - Các đặc trưng điện tử (ví dụ: độ phân cực, điện tích).
    - Các đặc trưng cấu trúc (ví dụ: số lượng vòng, số lượng liên kết).
    - Fingerprints (ví dụ: Morgan fingerprints, MACCS keys).
4. **Data Analysis and Modeling (Phân tích dữ liệu và xây dựng mô hình):**
    - Phân tích thống kê mô tả (descriptive statistics) để hiểu rõ hơn về dữ liệu.
    - Phân tích tương quan (correlation analysis) để xác định mối quan hệ giữa các biến.
    - Xây dựng các mô hình học máy (machine learning models) để dự đoán hoạt tính sinh học của các hợp chất mới. Ví dụ:
        - Các mô hình hồi quy (regression models) để dự đoán giá trị IC50, Ki, EC50.
        - Các mô hình phân loại (classification models) để dự đoán khả năng một hợp chất có hoạt tính hay không.
5. **Model Validation and Evaluation (Xác thực và đánh giá mô hình):** Sử dụng các kỹ thuật xác thực chéo (cross-validation) để đánh giá hiệu năng của mô hình. Sử dụng các độ đo phù hợp (ví dụ: R-squared, RMSE, AUC) để so sánh các mô hình khác nhau.
6. **Interpretation and Visualization (Diễn giải và trực quan hóa):** Sử dụng các kỹ thuật trực quan hóa dữ liệu (data visualization) để trình bày kết quả phân tích một cách rõ ràng và dễ hiểu. Diễn giải kết quả để đưa ra các kết luận có ý nghĩa về mặt sinh học và hóa học.

Hướng dẫn song ngữ (Bilingual Instructions):

1. Data Extraction (Trích xuất dữ liệu)

- **English:** Use SQL queries to extract relevant data from the ChEMBL 35 database. This may include compound information, bioactivity data, target information, and physicochemical properties.
- **Tiếng Việt:** Sử dụng các truy vấn SQL để trích xuất dữ liệu liên quan từ cơ sở dữ liệu ChEMBL 35. Dữ liệu này có thể bao gồm thông tin về hợp chất, dữ liệu hoạt tính sinh học, thông tin về mục tiêu và các thuộc tính lý hóa.

2. Data Cleaning and Preprocessing (Làm sạch và tiền xử lý dữ liệu)

- **English:** Handle missing values, standardize data units, and filter out invalid or unreliable data.
- **Tiếng Việt:** Xử lý các giá trị thiếu, chuẩn hóa đơn vị dữ liệu và lọc bỏ dữ liệu không hợp lệ hoặc không đáng tin cậy.

3. Feature Engineering (Xây dựng đặc trưng)

- **English:** Use RDKit to calculate molecular features from compound structures, such as shape features, electronic features, structural features, and fingerprints.
- **Tiếng Việt:** Sử dụng RDKit để tính toán các đặc trưng phân tử từ cấu trúc hợp chất, chẳng hạn như đặc trưng hình dạng, đặc trưng điện tử, đặc trưng cấu trúc và fingerprints.

4. Data Analysis and Modeling (Phân tích dữ liệu và xây dựng mô hình)

- **English:** Perform descriptive statistics, correlation analysis, and build machine learning models to predict bioactivity.
- **Tiếng Việt:** Thực hiện thống kê mô tả, phân tích tương quan và xây dựng các mô hình học máy để dự đoán hoạt tính sinh học.

5. Model Validation and Evaluation (Xác thực và đánh giá mô hình)

- **English:** Use cross-validation techniques to evaluate model performance and compare different models using appropriate metrics.
- **Tiếng Việt:** Sử dụng các kỹ thuật xác thực chéo để đánh giá hiệu suất mô hình và so sánh các mô hình khác nhau bằng cách sử dụng các chỉ số phù hợp.

6. Interpretation and Visualization (Diễn giải và trực quan hóa)

- **English:** Use data visualization techniques to present analysis results clearly and interpret the results to draw meaningful biological and chemical conclusions.
- **Tiếng Việt:** Sử dụng các kỹ thuật trực quan hóa dữ liệu để trình bày kết quả phân tích một cách rõ ràng và diễn giải kết quả để đưa ra các kết luận có ý nghĩa về mặt sinh học và hóa học.

Code mẫu (Code Samples):

1. SQL: Trích xuất dữ liệu hoạt tính sinh học (Extracting Bioactivity Data)

```sql
-- English
-- Extract compound IDs, standard values, and units for a specific target
SELECT
    cmp.chembl_id,
    act.standard_value,
    act.standard_units
FROM
    activities act
JOIN
    molecule_dictionary cmp ON act.molregno = cmp.molregno
WHERE
    act.target_chembl_id = 'CHEMBL205'  -- Replace with your target of interest
    AND act.standard_type = 'IC50'
    AND act.standard_relation = '='
```

```
    AND act.standard_value IS NOT NULL
LIMIT 100;

-- Vietnamese
-- Trích xuất ID hợp chất, giá trị tiêu chuẩn và đơn vị cho một mục tiêu cụ thể
SELECT
    cmp.chembl_id,
    act.standard_value,
    act.standard_units
FROM
    activities act
JOIN
    molecule_dictionary cmp ON act.molregno = cmp.molregno
WHERE
    act.target_chembl_id = 'CHEMBL205'  -- Thay thế bằng mục tiêu bạn quan tâm
    AND act.standard_type = 'IC50'
    AND act.standard_relation = '='
    AND act.standard_value IS NOT NULL
LIMIT 100;
```

2. Python: Tính toán fingerprints sử dụng RDKit (Calculating Fingerprints using RDKit)

```
# English
from rdkit import Chem
from rdkit.Chem import AllChem
import pandas as pd

# Assuming you have a DataFrame 'df' with a column 'smiles'
# Replace 'path/to/your/data.csv' with the actual path to your CSV file
csv_path = 'path/to/your/data.csv'
df = pd.read_csv(csv_path)


def calculate_fingerprint(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol:
        fp = AllChem.GetMorganFingerprintAsBitVect(mol, 2, nBits=2048)
        return fp.ToBitString()
    else:
        return None

df['fingerprint'] = df['canonical_smiles'].apply(calculate_fingerprint)
df = df.dropna(subset=['fingerprint'])

print(df.head())

# Vietnamese
from rdkit import Chem
from rdkit.Chem import AllChem
import pandas as pd

# Giả sử bạn có DataFrame 'df' với cột 'smiles'
# Thay thế 'path/to/your/data.csv' bằng đường dẫn thực tế đến tệp CSV của bạn
csv_path = 'path/to/your/data.csv'
df = pd.read_csv(csv_path)

def calculate_fingerprint(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol:
        fp = AllChem.GetMorganFingerprintAsBitVect(mol, 2, nBits=2048)
```

```
        return fp.ToBitString()
    else:
        return None

df['fingerprint'] = df['canonical_smiles'].apply(calculate_fingerprint)
df = df.dropna(subset=['fingerprint'])

print(df.head())
```

3. SQL: Lọc các hợp chất có hoạt tính IC50 (Filtering Compounds with IC50 Activity)

```
-- English
-- Select compounds with IC50 values less than 100 nM
SELECT
    cmp.chembl_id,
    act.standard_value,
    act.standard_units
FROM
    activities act
JOIN
    molecule_dictionary cmp ON act.molregno = cmp.molregno
WHERE
    act.standard_type = 'IC50'
    AND act.standard_relation = '='
    AND act.standard_value IS NOT NULL
    AND act.standard_units = 'nM'
    AND act.standard_value < 100
LIMIT 100;

-- Vietnamese
-- Chọn các hợp chất có giá trị IC50 nhỏ hơn 100 nM
SELECT
    cmp.chembl_id,
    act.standard_value,
    act.standard_units
FROM
    activities act
JOIN
    molecule_dictionary cmp ON act.molregno = cmp.molregno
WHERE
    act.standard_type = 'IC50'
    AND act.standard_relation = '='
    AND act.standard_value IS NOT NULL
    AND act.standard_units = 'nM'
    AND act.standard_value < 100
LIMIT 100;
```

4. Python: Xây dựng mô hình hồi quy tuyến tính (Building a Linear Regression Model)

```
# English
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import numpy as np

# Assuming you have a DataFrame 'df' with 'fingerprint' and 'standard_value' columns
# Convert fingerprint strings to numpy arrays
df['fingerprint'] = df['fingerprint'].apply(lambda x: np.array([int(bit) for bit in
x]))

# Prepare data for the model
```

```
X = np.stack(df['fingerprint'].values)  # Convert list of arrays to 2D array
y = df['standard_value'].values

# 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 on the test set
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
print(f'Mean Squared Error: {mse}')

# Vietnamese
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import numpy as np

# Giả sử bạn có DataFrame 'df' với các cột 'fingerprint' và 'standard_value'
# Chuyển đổi chuỗi fingerprint thành mảng numpy
df['fingerprint'] = df['fingerprint'].apply(lambda x: np.array([int(bit) for bit in
x]))

# Chuẩn bị dữ liệu cho mô hình
X = np.stack(df['fingerprint'].values)  # Chuyển đổi danh sách mảng thành mảng 2D
y = df['standard_value'].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)

# Xây dựng và huấn luyện mô hình hồi quy tuyến tính
model = LinearRegression()
model.fit(X_train, y_train)

# Dự đoán trên tập kiểm tra
y_pred = model.predict(X_test)

# Đánh giá mô hình
mse = mean_squared_error(y_test, y_pred)
print(f'Lỗi bình phương trung bình: {mse}')
```

5. SQL: Đếm số lượng hợp chất cho mỗi mục tiêu (Counting Compounds per Target)

```
-- English
-- Count the number of compounds associated with each target
SELECT
    act.target_chembl_id,
    COUNT(DISTINCT cmp.chembl_id) AS num_compounds
FROM
    activities act
JOIN
    molecule_dictionary cmp ON act.molregno = cmp.molregno
GROUP BY
```

```
    act.target_chembl_id
ORDER BY
    num_compounds DESC
LIMIT 100;

-- Vietnamese
-- Đếm số lượng hợp chất liên quan đến mỗi mục tiêu
SELECT
    act.target_chembl_id,
    COUNT(DISTINCT cmp.chembl_id) AS num_compounds
FROM
    activities act
JOIN
    molecule_dictionary cmp ON act.molregno = cmp.molregno
GROUP BY
    act.target_chembl_id
ORDER BY
    num_compounds DESC
LIMIT 100;
```

Lưu ý về lỗi (Error Notes):

a. ERROR: operator does not exist: numeric ~ unknown

Lỗi này xảy ra khi bạn cố gắng sử dụng toán tử ~ (regular expression match) trên một cột kiểu số (numeric). Để khắc phục, bạn cần đảm bảo rằng cột `act.standard_value` có kiểu dữ liệu text hoặc varchar. Nếu không, bạn có thể ép kiểu nó trước khi sử dụng toán tử ~.

Ví dụ:

```
-- English
-- Try to cast the standard_value to TEXT
AND CAST(act.standard_value AS TEXT) ~ '^[0-9\.]+$'

-- Vietnamese
-- Thử ép kiểu standard_value sang TEXT
AND CAST(act.standard_value AS TEXT) ~ '^[0-9\.]+$'
```

Hoặc, tốt hơn là bạn nên sử dụng các hàm số học để so sánh giá trị trực tiếp thay vì sử dụng regular expression.

b. Phiên bản Scikit-learn cũ không hỗ trợ tham số `squared=False` trong hàm `mean_squared_error`

Nếu bạn đang sử dụng một phiên bản scikit-learn cũ, bạn sẽ gặp lỗi khi sử dụng tham số `squared=False` trong hàm `mean_squared_error`. Để khắc phục, bạn có thể tính căn bậc hai của MSE để có được RMSE (Root Mean Squared Error).

```
# English
from sklearn.metrics import mean_squared_error
import numpy as np

mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
print(f'Root Mean Squared Error: {rmse}')

# Vietnamese
from sklearn.metrics import mean_squared_error
import numpy as np

mse = mean_squared_error(y_test, y_pred)
```

```
rmse = np.sqrt(mse)
print(f'Lỗi bình phương trung bình gốc: {rmse}')
```

Hoặc, bạn nên nâng cấp phiên bản scikit-learn của mình lên phiên bản mới nhất để có thể sử dụng tham số `squared=False`.

Quan trọng:

- Hãy nhớ thay thế các đường dẫn tệp, ID mục tiêu và các tham số khác trong code mẫu bằng các giá trị phù hợp với dự án của bạn.
- Đảm bảo rằng bạn đã cài đặt tất cả các thư viện cần thiết (ví dụ: RDKit, scikit-learn, pandas) trước khi chạy code Python.
- Điều chỉnh các tham số của mô hình học máy (ví dụ: số lượng n_bits trong fingerprint, test_size trong train_test_split) để đạt được hiệu năng tốt nhất.
- Luôn luôn đánh giá kết quả của bạn một cách cẩn thận và diễn giải chúng trong bối cảnh của vấn đề nghiên cứu của bạn.

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