PharmApp Suite

🧠 AI for Drug Discovery and Development 💉

🧩 Topic: Liên kết ChEMBL với dữ liệu nhãn FDA

Tuyệt vời! Tôi rất vui được hỗ trợ bạn trong dự án phân tích dữ liệu ChEMBL 35 này. Với vai trò là chuyên gia nghiên cứu và phát triển dược học, tôi sẽ giúp bạn xây dựng quy trình phân tích hiệu quả, giải quyết các vấn đề kỹ thuật và cung cấp các ví dụ code để bạn tham khảo.

Tổng quan dự án:

Dự án này tập trung vào việc khai thác dữ liệu từ cơ sở dữ liệu ChEMBL 35, một nguồn thông tin phong phú về các hợp chất hóa học và hoạt tính sinh học của chúng. Sử dụng RDKit để xử lý thông tin hóa học và Python để phân tích dữ liệu, bạn có thể khám phá các mối quan hệ giữa cấu trúc phân tử và hoạt tính, từ đó hỗ trợ quá trình thiết kế và phát triển thuốc.

Phân tích và Hướng dẫn:

Chúng ta sẽ tập trung vào chủ đề `Topic_CheMBL_35_84`. Để bắt đầu, chúng ta cần xác định mục tiêu cụ thể của chủ đề này. Ví dụ:

- **Mục tiêu:** Xây dựng mô hình dự đoán hoạt tính ức chế enzyme (ví dụ: IC50) dựa trên cấu trúc phân tử của các hợp chất.
- **Mô hình phân tích:**
    1. **Chuẩn bị dữ liệu:**
        - Kết nối đến cơ sở dữ liệu ChEMBL 35.
        - Lọc dữ liệu theo enzyme mục tiêu (target).
        - Làm sạch dữ liệu hoạt tính (activity data), loại bỏ các giá trị không hợp lệ hoặc không đầy đủ.
        - Tính toán các đặc trưng phân tử (molecular descriptors) bằng RDKit.
    2. **Xây dựng mô hình:**
        - Chia dữ liệu thành tập huấn luyện (training set) và tập kiểm tra (test set).
        - Chọn thuật toán học máy phù hợp (ví dụ: Random Forest, Support Vector Machine).
        - Huấn luyện mô hình trên tập huấn luyện.
        - Đánh giá hiệu năng của mô hình trên tập kiểm tra.
    3. **Diễn giải kết quả:**
        - Xác định các đặc trưng phân tử quan trọng ảnh hưởng đến hoạt tính.
        - Đề xuất các cải tiến cấu trúc để tăng cường hoạt tính.

Hướng dẫn song ngữ:

1. Data Preparation (Chuẩn bị dữ liệu):

- **SQL:** Extract relevant data from ChEMBL database.
- **SQL:** Trích xuất dữ liệu liên quan từ cơ sở dữ liệu ChEMBL.
- **Python:** Calculate molecular descriptors using RDKit.
- **Python:** Tính toán các đặc trưng phân tử bằng RDKit.

2. Model Building (Xây dựng mô hình):

- **Python:** Train a machine learning model to predict activity.
- **Python:** Huấn luyện mô hình học máy để dự đoán hoạt tính.

- **Python:** Evaluate model performance using appropriate metrics.
- **Python:** Đánh giá hiệu năng của mô hình bằng các độ đo phù hợp.

3. Interpretation (Diễn giải):

- **Python:** Identify key molecular features influencing activity.
- **Python:** Xác định các đặc trưng phân tử quan trọng ảnh hưởng đến hoạt tính.
- **Based on the model, propose structural modifications to improve activity.
- Dựa trên mô hình, đề xuất các cải tiến cấu trúc để tăng cường hoạt tính.

Code SQL:

```sql
-- Lấy 100 hợp chất có hoạt tính trên một mục tiêu cụ thể (ví dụ: CHEMBL205)
SELECT DISTINCT mol.molregno,
                md.chembl_id,
                act.standard_value,
                act.standard_units
FROM molecule_dictionary mol
    JOIN activities act ON mol.molregno = act.molregno
    JOIN assay_xref ax ON act.assay_id = ax.assay_id
    JOIN target_dictionary td ON ax.tid = td.tid
    JOIN compound_structures cs ON mol.molregno = cs.molregno
WHERE td.chembl_id = 'CHEMBL205'
  AND act.standard_type = 'IC50'
  AND act.standard_relation = '='
  AND act.standard_value IS NOT NULL
LIMIT 100;

-- Sửa lỗi liên quan đến kiểu dữ liệu trong điều kiện lọc
-- Chuyển đổi giá trị standard_value sang kiểu numeric trước khi so sánh
SELECT DISTINCT mol.molregno,
                md.chembl_id,
                act.standard_value,
                act.standard_units
FROM molecule_dictionary mol
    JOIN activities act ON mol.molregno = act.molregno
    JOIN assay_xref ax ON act.assay_id = ax.assay_id
    JOIN target_dictionary td ON ax.tid = td.tid
    JOIN compound_structures cs ON mol.molregno = cs.molregno
WHERE td.chembl_id = 'CHEMBL205'
  AND act.standard_type = 'IC50'
  AND act.standard_relation = '='
  AND act.standard_value IS NOT NULL
  AND act.standard_value::TEXT ~ '^[0-9\.]+$' -- Kiểm tra xem giá trị có phải là số
LIMIT 100;
```

Code Python:

```python
import os
import pandas as pd
from rdkit import Chem
from rdkit.Chem import AllChem
from rdkit.Chem import Descriptors
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score

# Cấu hình đường dẫn
base_path = os.getcwd()  # Đường dẫn thư mục gốc của dự án
data_path = os.path.join(base_path, 'data')
```

```
notebook_path = os.path.join(base_path, 'notebooks')

# Đọc dữ liệu từ file CSV
data = pd.read_csv(os.path.join(data_path, 'chembl_data.csv'))

# Hàm tính toán đặc trưng phân tử
def calculate_descriptors(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is None:
        return None
    descriptors = {}
    descriptors['MolLogP'] = Descriptors.MolLogP(mol)
    descriptors['MolWt'] = Descriptors.MolWt(mol)
    descriptors['NumHAcceptors'] = Descriptors.NumHAcceptors(mol)
    descriptors['NumHDonors'] = Descriptors.NumHDonors(mol)
    return descriptors

# Áp dụng hàm tính toán đặc trưng
data['descriptors'] = data['canonical_smiles'].apply(calculate_descriptors)
data = data.dropna(subset=['descriptors'])

# Chuyển đổi đặc trưng thành DataFrame
descriptors_df = pd.DataFrame(data['descriptors'].tolist())
data = pd.concat([data, descriptors_df], axis=1)

# Chuẩn bị dữ liệu cho mô hình
X = data[['MolLogP', 'MolWt', 'NumHAcceptors', 'NumHDonors']]
y = data['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)

# Xây dựng mô hình Random Forest
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Dự đoán và 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'R2 Score: {r2}')
```

Giải thích code:

1. **Import thư viện:** Nhập các thư viện cần thiết như `os`, `pandas`, `rdkit`, và `sklearn`.
2. **Cấu hình đường dẫn:** Xác định đường dẫn đến các thư mục chứa dữ liệu và notebook.
3. **Đọc dữ liệu:** Đọc dữ liệu từ file CSV vào DataFrame.
4. **Tính toán đặc trưng phân tử:**
    - Định nghĩa hàm `calculate_descriptors` để tính toán các đặc trưng phân tử từ SMILES.
    - Áp dụng hàm này cho cột `canonical_smiles` trong DataFrame.
5. **Chuẩn bị dữ liệu:**
    - Chọn các đặc trưng phân tử làm biến độc lập (X) và giá trị hoạt tính (IC50) làm biến phụ thuộc (y).
    - Chia dữ liệu thành tập huấn luyện và tập kiểm tra.

6. **Xây dựng mô hình:**
    - Khởi tạo mô hình Random Forest.
    - Huấn luyện mô hình trên tập huấn luyện.
7. **Đánh giá mô hình:**
    - Dự đoán giá trị hoạt tính trên tập kiểm tra.
    - Tính toán Mean Squared Error (MSE) và R2 Score để đánh giá hiệu năng của mô hình.

Ví dụ code bổ sung:

Ví dụ 1: Lọc dữ liệu theo khoảng giá trị hoạt tính:

```
-- Lấy các hợp chất có IC50 từ 100 nM đến 1000 nM
SELECT DISTINCT mol.molregno,
                md.chembl_id,
                act.standard_value,
                act.standard_units
FROM molecule_dictionary mol
    JOIN activities act ON mol.molregno = act.molregno
    JOIN assay_xref ax ON act.assay_id = ax.assay_id
    JOIN target_dictionary td ON ax.tid = td.tid
WHERE td.chembl_id = 'CHEMBL205'
  AND act.standard_type = 'IC50'
  AND act.standard_relation = '='
  AND act.standard_value BETWEEN 100 AND 1000
  AND act.standard_units = 'nM'
LIMIT 100;

# Lọc dữ liệu theo khoảng giá trị hoạt tính
data_filtered = data[(data['standard_value'] >= 100) & (data['standard_value'] <=
1000)]
```

Ví dụ 2: Tính toán TPSA bằng RDKit:

```
from rdkit.Chem import rdMolDescriptors

def calculate_tpsa(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is None:
        return None
    tpsa = rdMolDescriptors.CalcTPSA(mol)
    return tpsa

data['tpsa'] = data['canonical_smiles'].apply(calculate_tpsa)
```

Ví dụ 3: Sử dụng Support Vector Machine (SVM) cho mô hình hóa:

```
from sklearn.svm import SVR

# Khởi tạo mô hình SVM
model = SVR(kernel='rbf')
model.fit(X_train, y_train)

# Dự đoán và 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'R2 Score: {r2}')
```

Ví dụ 4: Lưu trữ kết quả vào file CSV:

```
results = pd.DataFrame({'actual': y_test, 'predicted': y_pred})
results.to_csv(os.path.join(data_path, 'predictions.csv'), index=False)
```

Ví dụ 5: Sử dụng PCA để giảm chiều dữ liệu:

```
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler

# Chuẩn hóa dữ liệu
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Giảm chiều dữ liệu xuống 2 chiều
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

# Tạo DataFrame từ dữ liệu PCA
X_pca_df = pd.DataFrame(data=X_pca, columns=['PC1', 'PC2'])
```

Lưu ý về lỗi:

- **Lỗi SQL:** Lỗi `operator does not exist: numeric ~ unknown` xảy ra do bạn đang cố gắng so sánh một giá trị số (`act.standard_value`) với một chuỗi (`'^[0-9\.]+$'`). Để khắc phục, bạn cần đảm bảo rằng cả hai vế của phép so sánh đều có cùng kiểu dữ liệu. Trong trường hợp này, bạn có thể chuyển đổi `act.standard_value` sang kiểu chuỗi bằng cách sử dụng `act.standard_value::TEXT`.
- **Lỗi scikit-learn:** Nếu bạn gặp lỗi `squared=False` không được hỗ trợ, hãy nâng cấp phiên bản scikit-learn của bạn lên phiên bản mới nhất.

Kết luận:

Hy vọng những hướng dẫn và ví dụ này sẽ giúp bạn bắt đầu dự án phân tích dữ liệu ChEMBL 35 một cách hiệu quả. Hãy nhớ rằng, việc thử nghiệm và điều chỉnh mô hình là rất quan trọng để đạt được kết quả tốt nhất. Chúc bạn thành công!