PharmApp Suite

🧠 AI for Drug Discovery and Development 💉

🧩 Topic: So sánh thuốc lâm sàng và tiền lâm sàng

Tuyệt vời! Dự án của bạn rất thú vị và có tính ứng dụng cao trong nghiên cứu phát triển thuốc. Tôi sẽ đóng vai trò là chuyên gia R&D dược học và hỗ trợ bạn trong dự án này.

Tổng quan về quy trình phân tích:

Chúng ta sẽ thực hiện quy trình sau:

1. **Trích xuất dữ liệu từ cơ sở dữ liệu ChEMBL 35 (Data Extraction):** Sử dụng SQL để truy vấn và trích xuất thông tin cần thiết từ cơ sở dữ liệu ChEMBL 35. Dữ liệu này bao gồm thông tin về các hợp chất, hoạt tính sinh học của chúng (IC50, Ki, v.v.), và các thuộc tính khác.
2. **Tiền xử lý dữ liệu (Data Preprocessing):** Sử dụng RDKit và Pandas để làm sạch, biến đổi và chuẩn hóa dữ liệu. Các bước này bao gồm chuyển đổi SMILES thành fingerprint, xử lý giá trị hoạt tính, và loại bỏ dữ liệu trùng lặp hoặc không hợp lệ.
3. **Phân tích dữ liệu (Data Analysis):** Sử dụng các kỹ thuật thống kê và học máy để khám phá các mối quan hệ giữa cấu trúc hóa học và hoạt tính sinh học. Các phương pháp phổ biến bao gồm phân tích tương quan, hồi quy, và phân loại.
4. **Xây dựng mô hình (Model Building):** Xây dựng các mô hình dự đoán hoạt tính sinh học dựa trên cấu trúc hóa học. Các thuật toán học máy như Random Forest, Support Vector Machines (SVM), và Neural Networks thường được sử dụng.
5. **Đánh giá mô hình (Model Evaluation):** Đánh giá hiệu suất của mô hình bằng cách sử dụng các chỉ số phù hợp như RMSE, R-squared, AUC, và độ chính xác.
6. **Diễn giải kết quả (Interpretation):** Giải thích kết quả của mô hình và xác định các đặc điểm cấu trúc quan trọng ảnh hưởng đến hoạt tính sinh học.
7. **Ứng dụng (Application):** Sử dụng mô hình để sàng lọc ảo các hợp chất mới và đề xuất các ứng cử viên tiềm năng cho phát triển thuốc.

Phân tích và Hướng dẫn chi tiết (Analysis and Detailed Instructions):

1. Phân tích mô hình (Model Analysis):

- **Bài toán (Problem):** Dự đoán hoạt tính sinh học của các hợp chất dựa trên cấu trúc hóa học của chúng. Điều này giúp chúng ta xác định các hợp chất có tiềm năng trở thành thuốc.
- **Dữ liệu (Data):** Dữ liệu ChEMBL 35 chứa thông tin về cấu trúc hóa học (SMILES), hoạt tính sinh học (IC50, Ki, v.v.), và các thuộc tính khác của hàng triệu hợp chất.
- **Phương pháp (Method):** Chúng ta sẽ sử dụng các phương pháp học máy để xây dựng mô hình dự đoán hoạt tính sinh học. Các phương pháp phổ biến bao gồm:
    - **Hồi quy (Regression):** Sử dụng để dự đoán giá trị hoạt tính liên tục (ví dụ: IC50).
    - **Phân loại (Classification):** Sử dụng để dự đoán hoạt tính nhị phân (ví dụ: hoạt tính/không hoạt tính).
    - **Các thuật toán (Algorithms):** Random Forest, SVM, Neural Networks, v.v.
- **Đánh giá (Evaluation):** Sử dụng các chỉ số như RMSE, R-squared, AUC, độ chính xác để đánh giá hiệu suất của mô hình.

2. Hướng dẫn song ngữ (Bilingual Instructions):

2.1. SQL (Tiếng Anh/English):

- **Mục tiêu (Goal):** Trích xuất dữ liệu từ cơ sở dữ liệu ChEMBL 35. (Extract data from the ChEMBL 35 database.)
- **Lưu ý (Note):** Chỉ lấy 100 dòng dữ liệu để giảm tải cho máy tính. (Only retrieve 100 rows of data to reduce the load on the machine.)
- **Sửa lỗi (Error Correction):** Sửa lỗi liên quan đến so sánh chuỗi số. (Fix the error related to comparing numeric strings.)

```sql
-- SQL query to extract data from ChEMBL 35
-- This query retrieves 100 rows of data for analysis.
-- It selects specific columns from the activities, molecule_dictionary, and
target_dictionary tables.

SELECT DISTINCT
    act.molregno,
    md.chembl_id,
    td.chembl_id AS target_chembl_id,
    act.standard_type,
    act.standard_relation,
    act.standard_value,
    act.standard_units,
    md.canonical_smiles
FROM
    activities act
JOIN
    molecule_dictionary md ON act.molregno = md.molregno
JOIN
    target_dictionary td ON act.tid = td.tid
WHERE
    act.standard_type = 'IC50'
    AND act.standard_relation = '='
    AND act.standard_value IS NOT NULL
    AND md.canonical_smiles IS NOT NULL
    AND act.standard_value::text ~ '^[0-9\.]+$' -- Corrected line: convert to text for
regex comparison
LIMIT 100;
```

Giải thích (Explanation):

- `act.standard_value::text ~ '^[0-9\.]+$'`: Chuyển đổi `standard_value` sang kiểu text trước khi so sánh với regular expression. (Converts `standard_value` to text type before comparing with a regular expression.)

2.2. Python (Tiếng Anh/English):

- **Mục tiêu (Goal):** Đọc dữ liệu từ file CSV, tiền xử lý dữ liệu, và xây dựng mô hình học máy. (Read data from CSV file, preprocess data, and build a machine learning model.)
- **Lưu ý (Note):** Sử dụng `os.path.join` để tạo đường dẫn. (Use `os.path.join` to create paths.)
- **Sửa lỗi (Error Correction):** Cập nhật scikit-learn hoặc sử dụng phiên bản cũ hơn của hàm `mean_squared_error`. (Update scikit-learn or use an older version of the `mean_squared_error` function.)

```python
# Python code to read data, preprocess, and build a machine learning model
import os
import pandas as pd
from rdkit import Chem
from rdkit.Chem import AllChem
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
```

```
# Define the base path
base_path = '../data'

# Construct the file path
file_path = os.path.join(base_path, 'chembl_35_ic50_data.csv')

# Read the CSV file into a Pandas DataFrame
try:
    df = pd.read_csv(file_path)
except FileNotFoundError:
    print(f"Error: The file '{file_path}' was not found.")
    exit()

# Data Preprocessing
df.dropna(subset=['canonical_smiles', 'standard_value'], inplace=True)

def generate_fingerprint(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        fp = AllChem.GetMorganFingerprintAsBitVect(mol, 2, nBits=2048)
        return np.array(fp)
    else:
        return None

df['fingerprint'] = df['canonical_smiles'].apply(generate_fingerprint)
df.dropna(subset=['fingerprint'], inplace=True)

# Convert IC50 values to numeric, handling potential errors
df['standard_value'] = pd.to_numeric(df['standard_value'], errors='coerce')
df.dropna(subset=['standard_value'], inplace=True)

# Split data into features (fingerprints) and target (IC50)
X = np.array(list(df['fingerprint']))
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)

# Model Building (Random Forest Regressor)
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Model Evaluation
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}")
```

Giải thích (Explanation):

- `os.path.join(base_path, 'chembl_35_ic50_data.csv')`: Tạo đường dẫn đến file CSV. (Creates the path to the CSV file.)
- `mean_squared_error(y_test, y_pred, squared=False)`: Tính toán RMSE (Root Mean Squared Error). Nếu phiên bản scikit-learn cũ, bỏ `squared=False`. (Calculates RMSE. If the scikit-learn version is old, remove `squared=False`.)

3. Ví dụ (Examples):

3.1. SQL:

```sql
-- 1. Get the number of unique molecules in the database
SELECT COUNT(DISTINCT molregno) FROM molecule_dictionary;

-- 2. Find molecules with a specific SMILES string
SELECT chembl_id FROM molecule_dictionary WHERE canonical_smiles = 'CCO';

-- 3. Find activities for a specific target
SELECT act.standard_value, act.standard_units FROM activities act JOIN
target_dictionary td ON act.tid = td.tid WHERE td.chembl_id = 'CHEMBL205';

-- 4. Find molecules with IC50 values less than 1000
SELECT md.chembl_id, act.standard_value FROM molecule_dictionary md JOIN activities
act ON md.molregno = act.molregno WHERE act.standard_type = 'IC50' AND
act.standard_value < 1000 LIMIT 100;

-- 5. List the standard types and their counts
SELECT standard_type, COUNT(*) FROM activities GROUP BY standard_type ORDER BY
COUNT(*) DESC LIMIT 10;
```

3.2. Python:

```python
# 1. Calculate the average IC50 value
print(f"Average IC50: {df['standard_value'].mean()}")

# 2. Filter data for IC50 values less than 100
filtered_df = df[df['standard_value'] < 100]
print(f"Number of molecules with IC50 < 100: {len(filtered_df)}")

# 3. Find the molecule with the smallest IC50 value
min_ic50_molecule = df.loc[df['standard_value'].idxmin()]
print(f"Molecule with smallest IC50: {min_ic50_molecule['chembl_id']}")

# 4. Create a scatter plot of IC50 values vs molecule weight
from rdkit.Chem import Descriptors
df['mol_weight'] = df['canonical_smiles'].apply(lambda x:
Descriptors.MolWt(Chem.MolFromSmiles(x)))

import matplotlib.pyplot as plt
plt.scatter(df['mol_weight'], df['standard_value'])
plt.xlabel('Molecular Weight')
plt.ylabel('IC50 Value')
plt.show()

# 5. Calculate the correlation between molecular weight and IC50
correlation = df['mol_weight'].corr(df['standard_value'])
print(f"Correlation between molecular weight and IC50: {correlation}")
```

Lưu ý quan trọng (Important Notes):

- **Phiên bản thư viện (Library versions):** Đảm bảo rằng bạn đã cài đặt các phiên bản thư viện tương thích. (Ensure that you have installed compatible library versions.)
- **Hiệu suất (Performance):** Phân tích trên toàn bộ dữ liệu ChEMBL 35 có thể tốn nhiều thời gian và tài nguyên. Hãy cân nhắc sử dụng các kỹ thuật tối ưu hóa và tính toán song song nếu cần. (Analysis on the entire ChEMBL 35 dataset can be time-consuming and resource-intensive. Consider using optimization techniques and parallel computing if necessary.)

- **Mô hình hóa (Modeling):** Thử nghiệm với nhiều thuật toán học máy khác nhau và tinh chỉnh các tham số để đạt được hiệu suất tốt nhất. (Experiment with different machine learning algorithms and fine-tune the parameters to achieve the best performance.)

Tôi hy vọng hướng dẫn này sẽ giúp bạn trong dự án của mình. Hãy cho tôi biết nếu bạn có bất kỳ câu hỏi nào khác!