PharmApp Suite

🧠 AI for Drug Discovery and Development 💉

🧩 Topic: Chú thích target ChEMBL bằng cấu trúc PDB

Tuyệt vời! Tôi rất vui được hỗ trợ bạn trong dự án phân tích dữ liệu ChEMBL 35 bằng RDKit. Dưới đây là phân tích chi tiết, hướng dẫn song ngữ, code mẫu SQL và Python để bạn tham khảo, cùng với cách xử lý các lỗi bạn đã gặp.

Tổng quan dự án:

Chúng ta sẽ tập trung vào việc khám phá và tiền xử lý dữ liệu từ ChEMBL 35, sau đó sử dụng RDKit để tạo các đặc trưng phân tử và xây dựng mô hình dự đoán hoạt tính sinh học. Việc sử dụng AIMLOps Template giúp đảm bảo tính nhất quán và khả năng tái sử dụng của dự án.

1. Phân tích mô hình (Analysis Model):

Mô hình phân tích của chúng ta sẽ bao gồm các bước chính sau:

- **Data Extraction and Preprocessing (Trích xuất và tiền xử lý dữ liệu):** Lấy dữ liệu từ cơ sở dữ liệu ChEMBL 35, lọc và làm sạch dữ liệu để loại bỏ các giá trị không hợp lệ hoặc thiếu.
- **Feature Engineering (Tạo đặc trưng):** Sử dụng RDKit để tính toán các đặc trưng hóa lý và cấu trúc phân tử từ SMILES.
- **Data Analysis and Visualization (Phân tích và trực quan hóa dữ liệu):** Sử dụng các kỹ thuật thống kê và trực quan hóa để hiểu rõ hơn về dữ liệu và mối quan hệ giữa các đặc trưng.
- **Model Building (Xây dựng mô hình):** Xây dựng mô hình dự đoán hoạt tính sinh học (ví dụ: hồi quy tuyến tính, Random Forest) dựa trên các đặc trưng đã tạo.
- **Model Evaluation (Đánh giá mô hình):** Đánh giá hiệu suất của mô hình bằng các chỉ số phù hợp (ví dụ: RMSE, R-squared).

2. Hướng dẫn song ngữ (Bilingual Guidance):

2.1. Kết nối và truy vấn dữ liệu (Connecting and Querying Data):

- **Tiếng Việt:** Sử dụng thư viện `psycopg2` trong Python để kết nối đến cơ sở dữ liệu PostgreSQL và thực hiện các truy vấn SQL.
- **English:** Use the `psycopg2` library in Python to connect to the PostgreSQL database and execute SQL queries.

2.2. Tạo đặc trưng với RDKit (Feature Engineering with RDKit):

- **Tiếng Việt:** Sử dụng RDKit để chuyển đổi chuỗi SMILES thành đối tượng phân tử và tính toán các đặc trưng như trọng lượng phân tử, logP, số lượng liên kết, v.v.
- **English:** Use RDKit to convert SMILES strings into molecule objects and calculate features such as molecular weight, logP, number of bonds, etc.

2.3. Xây dựng và đánh giá mô hình (Building and Evaluating Models):

- **Tiếng Việt:** Sử dụng thư viện `scikit-learn` để xây dựng các mô hình học máy và đánh giá hiệu suất của chúng.
- **English:** Use the `scikit-learn` library to build machine learning models and evaluate their performance.

3. Code mẫu SQL và Python (SQL and Python Code Examples):

3.1. SQL (PostgreSQL):

```sql
-- Lấy 100 dòng dữ liệu từ bảng activities và molecule_dictionary
-- lấy các trường cần thiết: molecule_chembl_id, canonical_smiles, standard_value
SELECT md.molecule_chembl_id, md.canonical_smiles, act.standard_value
FROM activities act
JOIN molecule_dictionary md ON act.molregno = md.molregno
WHERE act.standard_type = 'IC50'  -- Lọc theo loại hoạt tính
  AND act.standard_relation = '=' -- Lọc theo quan hệ '='
  AND act.standard_value IS NOT NULL -- Loại bỏ giá trị NULL
  AND md.canonical_smiles IS NOT NULL -- Loại bỏ SMILES NULL
LIMIT 100;

-- Get 100 rows of data from the activities and molecule_dictionary tables
-- Select the necessary fields: molecule_chembl_id, canonical_smiles, standard_value
SELECT md.molecule_chembl_id, md.canonical_smiles, act.standard_value
FROM activities act
JOIN molecule_dictionary md ON act.molregno = md.molregno
WHERE act.standard_type = 'IC50'  -- Filter by activity type
  AND act.standard_relation = '=' -- Filter by relation '='
  AND act.standard_value IS NOT NULL -- Remove NULL values
  AND md.canonical_smiles IS NOT NULL -- Remove NULL SMILES
LIMIT 100;
```

3.2. Python:

```python
import os
import pandas as pd
import psycopg2
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, r2_score

# 1. Kết nối đến cơ sở dữ liệu (Connect to the database)
def connect_to_db(host, database, user, password):
    conn = psycopg2.connect(host=host, database=database, user=user,
password=password)
    return conn

# 2. Đọc dữ liệu từ file CSV (Read data from CSV file)
def read_data_from_csv(file_path):
    df = pd.read_csv(file_path)
    return df

# 3. Tính toán đặc trưng phân tử với RDKit (Calculate molecular features with RDKit)
def calculate_descriptors(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is None:
        return None
    descriptors = {}
    descriptors['mol_weight'] = Descriptors.MolWt(mol)
    descriptors['logp'] = Chem.Crippen.MolLogP(mol)
    descriptors['num_hba'] = Chem.Lipinski.NumHAcceptors(mol)
    descriptors['num_hbd'] = Chem.Lipinski.NumHDonors(mol)
    return descriptors

# 4. Tiền xử lý dữ liệu (Data preprocessing)
def preprocess_data(df):
    # Loại bỏ các dòng có SMILES không hợp lệ (Remove rows with invalid SMILES)
    df = df.dropna(subset=['canonical_smiles'])
    df = df[df['canonical_smiles'].apply(lambda x: Chem.MolFromSmiles(x) is not None)]
```

```
    # Tính toán các đặc trưng và thêm vào DataFrame (Calculate features and add to
DataFrame)
    descriptors = df['canonical_smiles'].apply(calculate_descriptors)
    df = pd.concat([df, descriptors.apply(pd.Series)], axis=1)
    df = df.dropna()
    return df

# 5. Xây dựng mô hình (Model building)
def build_model(df):
    X = df[['mol_weight', 'logp', 'num_hba', 'num_hbd']]
    y = df['standard_value']
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
    model = LinearRegression()
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    rmse = mean_squared_error(y_test, y_pred, squared=False)  # squared=False for RMSE
    r2 = r2_score(y_test, y_pred)
    print(f'RMSE: {rmse}')
    print(f'R2: {r2}')
    return model

# Đường dẫn cơ sở (Base path)
base_path = "."

# Tên file CSV (CSV file name)
csv_file = "chembl_data.csv"  # Thay đổi nếu cần

# Đường dẫn đầy đủ đến file CSV (Full path to CSV file)
csv_path = os.path.join(base_path, "data", csv_file)

# Thông tin kết nối cơ sở dữ liệu (Database connection information)
db_host = "192.168.206.136"
db_name = "chembl_35"
db_user = "rd"
db_password = "rd"

# Kết nối đến cơ sở dữ liệu (Connect to the database)
conn = connect_to_db(db_host, db_name, db_user, db_password)

# 1. Execute SQL query and save to CSV
sql_query = """
SELECT md.molecule_chembl_id, md.canonical_smiles, act.standard_value
FROM activities act
JOIN molecule_dictionary md ON act.molregno = md.molregno
WHERE act.standard_type = 'IC50'
  AND act.standard_relation = '='
  AND act.standard_value IS NOT NULL
  AND md.canonical_smiles IS NOT NULL
LIMIT 100;
"""

df = pd.read_sql_query(sql_query, conn)
df.to_csv(csv_path, index=False)

# Đọc dữ liệu (read data)
df = read_data_from_csv(csv_path)

# Tiền xử lý dữ liệu (Preprocess data)
```

```
df = preprocess_data(df)

# Xây dựng mô hình (Build model)
model = build_model(df)
```

4. Xử lý lỗi (Error Handling):

- **Lỗi SQL (SQL Error):** `ERROR: operator does not exist: numeric ~ unknown`. Lỗi này xảy ra do bạn đang cố gắng sử dụng toán tử `~` (regex match) trên kiểu dữ liệu `numeric`. Để khắc phục, bạn có thể chuyển đổi cột `standard_value` sang kiểu `TEXT` trước khi so sánh:

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


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


- **Lỗi Scikit-learn (Scikit-learn Error):** `squared=False` không được hỗ trợ. Hãy đảm bảo rằng bạn đang sử dụng phiên bản Scikit-learn đủ mới (>=0.22). Nếu không, hãy nâng cấp hoặc tính RMSE theo cách thủ công:

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

mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
```

5. Ví dụ code (Code Examples):

Dưới đây là 5 ví dụ code khác nhau để bạn có thể tham khảo và phát triển thêm:

Ví dụ 1: Tính toán nhiều đặc trưng RDKit (Calculating more RDKit features)

```
from rdkit.Chem import AllChem
def calculate_more_descriptors(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is None:
        return None
    descriptors = {}
    descriptors['tpsa'] = AllChem.CalcTPSA(mol)
    descriptors['qed'] = AllChem.qed.default(mol)
    return descriptors
```

Ví dụ 2: Sử dụng mô hình Random Forest (Using Random Forest model)

```
from sklearn.ensemble import RandomForestRegressor

def build_random_forest_model(df):
    X = df[['mol_weight', 'logp', 'num_hba', 'num_hbd']]
    y = df['standard_value']
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
    model = RandomForestRegressor(n_estimators=100, random_state=42)
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    rmse = mean_squared_error(y_test, y_pred, squared=False)
    r2 = r2_score(y_test, y_pred)
    print(f'RMSE: {rmse}')
    print(f'R2: {r2}')
    return model
```

Ví dụ 3: Lọc dữ liệu theo khoảng giá trị (Filtering data by value range)

```
SELECT md.molecule_chembl_id, md.canonical_smiles, act.standard_value
FROM activities act
JOIN molecule_dictionary md ON act.molregno = md.molregno
```

```
WHERE act.standard_type = 'IC50'
  AND act.standard_value BETWEEN 100 AND 1000
LIMIT 100;
```

Ví dụ 4: Lưu mô hình (Saving the model)

```
import pickle

def save_model(model, filepath):
    with open(filepath, 'wb') as f:
        pickle.dump(model, f)

# Sau khi xây dựng mô hình:
# After building the model:
model = build_model(df)
save_model(model, 'model.pkl')
```

Ví dụ 5: Đọc mô hình đã lưu (Loading the saved model)

```
import pickle

def load_model(filepath):
    with open(filepath, 'rb') as f:
        model = pickle.load(f)
    return model

# Để đọc mô hình:
# To load the model:
loaded_model = load_model('model.pkl')
```

Lưu ý quan trọng:

- Hãy nhớ thay đổi các đường dẫn file và thông tin kết nối cơ sở dữ liệu cho phù hợp với môi trường của bạn.
- Việc lựa chọn mô hình và đặc trưng phù hợp phụ thuộc vào đặc điểm của dữ liệu và mục tiêu nghiên cứu của bạn. Hãy thử nghiệm với các mô hình và đặc trưng khác nhau để tìm ra cấu hình tốt nhất.
- Luôn kiểm tra và làm sạch dữ liệu cẩn thận trước khi xây dựng mô hình. Dữ liệu sạch sẽ giúp cải thiện đáng kể hiệu suất của mô hình.

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