# PharmApp Suite

## 🧠 AI for Drug Discovery and Development 💉



## 🧩 Topic: QSAR với Random Forest và đặc trưng từ ChEMBL

Tuyệt vời! Tôi rất vui được tham gia vào dự án phân tích dữ liệu ChEMBL 35 của bạn. Dựa trên thông tin bạn cung cấp, tôi sẽ giúp bạn phân tích, xây dựng mô hình, và cung cấp code mẫu SQL và Python để bạn có thể bắt đầu một cách hiệu quả.

**Tổng quan dự án:**

Dự án của bạn tập trung vào việc khai thác dữ liệu từ cơ sở dữ liệu ChEMBL 35, kết hợp với RDKit để hỗ trợ nghiên cứu và phát triển thuốc. Bạn đang sử dụng một cấu trúc thư mục theo chuẩn AIMLOps Template và sử dụng Jupyter Notebook để thực thi code.

**Mục tiêu:**

- Phân tích dữ liệu ChEMBL 35 để tìm ra các mối quan hệ tiềm năng giữa cấu trúc hóa học và hoạt tính sinh học của các hợp chất.
- Sử dụng RDKit để tính toán các descriptor phân tử từ cấu trúc hóa học.
- Xây dựng mô hình học máy để dự đoán hoạt tính sinh học của các hợp chất mới.

**Phân tích và Hướng dẫn:**

**1. Phân tích mô hình (Model Analysis):**

Chúng ta sẽ xây dựng một quy trình phân tích dữ liệu và xây dựng mô hình dự đoán, bao gồm các bước sau:

- **Data Extraction (Trích xuất dữ liệu):** Sử dụng SQL để truy vấn dữ liệu từ cơ sở dữ liệu ChEMBL 35.
- **Data Preprocessing (Tiền xử lý dữ liệu):**
    - Làm sạch dữ liệu, xử lý các giá trị thiếu (missing values).
    - Chuyển đổi dữ liệu về định dạng phù hợp cho việc phân tích.
- **Feature Engineering (Xây dựng đặc trưng):**
    - Sử dụng RDKit để tính toán các descriptor phân tử từ cấu trúc SMILES.
    - Chọn lọc các descriptor phù hợp.
- **Model Selection (Lựa chọn mô hình):**
    - Thử nghiệm với nhiều mô hình học máy khác nhau (ví dụ: Linear Regression, Random Forest, Support Vector Machines).
    - Sử dụng cross-validation để đánh giá hiệu suất của các mô hình.
- **Model Evaluation (Đánh giá mô hình):**
    - Sử dụng các metrics phù hợp (ví dụ: RMSE, R-squared) để đánh giá hiệu suất của mô hình trên tập test.
- **Model Deployment (Triển khai mô hình):**
    - Lưu mô hình đã huấn luyện để sử dụng trong tương lai.

**2. Hướng dẫn song ngữ (Bilingual Instructions):**

- **SQL:** Sử dụng SQL để truy vấn và trích xuất dữ liệu từ cơ sở dữ liệu ChEMBL.
- **Python:** Sử dụng Python với các thư viện như pandas, RDKit, scikit-learn để tiền xử lý dữ liệu, tính toán descriptor, xây dựng mô hình và đánh giá mô hình.

### 3. Code mẫu SQL và Python (Sample SQL and Python Code):

**SQL Code (Lấy 100 dòng dữ liệu):**

```sql
-- English
-- Extract 100 rows of data from the chembl_35 database, including molecule
information and activity data.

SELECT
    md.chembl_id,
    md.pref_name,
    ms.structure,
    act.standard_type,
    act.standard_value,
    act.standard_units
FROM
    molecule_dictionary md
JOIN
    compound_structures ms ON md.molregno = ms.molregno
JOIN
    activities act ON md.molregno = act.molregno
WHERE act.standard_type = 'IC50'
  AND act.standard_units = 'nM'
  AND act.standard_value IS NOT NULL
LIMIT 100;


-- Vietnamese
-- Trích xuất 100 dòng dữ liệu từ cơ sở dữ liệu chembl_35, bao gồm thông tin về phân
tử và dữ liệu hoạt tính.

SELECT
    md.chembl_id,
    md.pref_name,
    ms.structure,
    act.standard_type,
    act.standard_value,
    act.standard_units
FROM
    molecule_dictionary md
JOIN
    compound_structures ms ON md.molregno = ms.molregno
JOIN
    activities act ON md.molregno = act.molregno
WHERE act.standard_type = 'IC50'
  AND act.standard_units = 'nM'
  AND act.standard_value IS NOT NULL
LIMIT 100;
```

**Python Code:**

```python
# English
import os
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, r2_score
import numpy as np
```

```python
# Define base path
base_path = "../data"  # Adjust if necessary

# Load data from CSV
csv_file = "chembl_35_data.csv"  # Replace with your actual file name
data = pd.read_csv(os.path.join(base_path, csv_file))

# Data Preprocessing
data = data.dropna(subset=['structure', 'standard_value'])
data['standard_value'] = pd.to_numeric(data['standard_value'], errors='coerce')
data = data.dropna(subset=['standard_value'])

# RDKit Feature Calculation
def calculate_descriptors(smiles):
    try:
        mol = Chem.MolFromSmiles(smiles)
        if mol is None:
            return None
        descriptors = {}
        for name, func in Descriptors.descList:
            try:
                descriptors[name] = func(mol)
            except:
                descriptors[name] = np.nan
        return pd.Series(descriptors)
    except:
        return None

data['descriptors'] = data['structure'].apply(calculate_descriptors)
data = data.dropna(subset=['descriptors'])

# Convert descriptors to DataFrame
descriptors_df = pd.DataFrame(data['descriptors'].tolist())
data = pd.concat([data, descriptors_df], axis=1)

# Drop rows with NaN values in descriptors
data = data.dropna(subset=descriptors_df.columns)

# Prepare data for modeling
X = data[descriptors_df.columns]
y = data['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)

# Train a linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

```python
# Vietnamese
import os
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, r2_score
import numpy as np

# Định nghĩa đường dẫn gốc
base_path = "../data"  # Điều chỉnh nếu cần thiết

# Tải dữ liệu từ file CSV
csv_file = "chembl_35_data.csv"  # Thay thế bằng tên file thực tế của bạn
data = pd.read_csv(os.path.join(base_path, csv_file))

# Tiền xử lý dữ liệu
data = data.dropna(subset=['structure', 'standard_value'])
data['standard_value'] = pd.to_numeric(data['standard_value'], errors='coerce')
data = data.dropna(subset=['standard_value'])

# Tính toán các descriptor phân tử sử dụng RDKit
def calculate_descriptors(smiles):
    try:
        mol = Chem.MolFromSmiles(smiles)
        if mol is None:
            return None
        descriptors = {}
        for name, func in Descriptors.descList:
            try:
                descriptors[name] = func(mol)
            except:
                descriptors[name] = np.nan
        return pd.Series(descriptors)
    except:
        return None

data['descriptors'] = data['structure'].apply(calculate_descriptors)
data = data.dropna(subset=['descriptors'])

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

# Loại bỏ các hàng có giá trị NaN trong descriptors
data = data.dropna(subset=descriptors_df.columns)

# Chuẩn bị dữ liệu cho mô hình hóa
X = data[descriptors_df.columns]
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)

# Huấn luyện mô hình hồi quy tuyến tính
```

```
model = LinearRegression()
model.fit(X_train, y_train)

# Dự đoán
y_pred = model.predict(X_test)

# Đánh giá mô hình
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

**4. Ví dụ code .sql và python mẫu (Sample SQL and Python Code Examples):**

**SQL Examples:**

1. **Lọc các hợp chất có trọng lượng phân tử (MW) dưới 500:**

```
-- English
SELECT md.chembl_id, ms.structure FROM molecule_dictionary md JOIN compound_structures
ms ON md.molregno = ms.molregno WHERE md.mw_freebase < 500 LIMIT 100;

-- Vietnamese
SELECT md.chembl_id, ms.structure FROM molecule_dictionary md JOIN compound_structures
ms ON md.molregno = ms.molregno WHERE md.mw_freebase < 500 LIMIT 100;
```

2. **Tìm các hợp chất có chứa một khung cấu trúc cụ thể (ví dụ: benzene ring):**

```
-- English
SELECT md.chembl_id, ms.structure FROM molecule_dictionary md JOIN compound_structures
ms ON md.molregno = ms.molregno WHERE ms.structure LIKE '%c1ccccc1%' LIMIT 100;

-- Vietnamese
SELECT md.chembl_id, ms.structure FROM molecule_dictionary md JOIN compound_structures
ms ON md.molregno = ms.molregno WHERE ms.structure LIKE '%c1ccccc1%' LIMIT 100;
```

3. **Lấy thông tin về các mục tiêu (targets) liên quan đến một hợp chất cụ thể:**

```
-- English
SELECT td.chembl_id, td.pref_name FROM target_dictionary td JOIN target_components tc
ON td.tid = tc.tid JOIN component_sequences cs ON tc.component_id = cs.component_id
WHERE cs.sequence LIKE '%amino acid sequence%' LIMIT 100;

-- Vietnamese
SELECT td.chembl_id, td.pref_name FROM target_dictionary td JOIN target_components tc
ON td.tid = tc.tid JOIN component_sequences cs ON tc.component_id = cs.component_id
WHERE cs.sequence LIKE '%amino acid sequence%' LIMIT 100;
```

4. **Tìm các hoạt động (activities) có giá trị IC50 nhỏ hơn 100 nM:**

```
-- English
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 < 100 LIMIT 100;

-- Vietnamese
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 < 100 LIMIT 100;
```

5. **Thống kê số lượng hợp chất cho mỗi loại hoạt động (activity type):**

```
-- English
SELECT act.standard_type, COUNT(*) FROM activities act GROUP BY act.standard_type
```

```
LIMIT 100;

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

**Python Examples:**

1. **Tính toán logP cho các hợp chất:**

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

def calculate_logp(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        return Descriptors.MolLogP(mol)
    else:
        return None

# Apply the function to a DataFrame column
data['logP'] = data['structure'].apply(calculate_logp)

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

def calculate_logp(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        return Descriptors.MolLogP(mol)
    else:
        return None

# Áp dụng hàm cho một cột DataFrame
data['logP'] = data['structure'].apply(calculate_logp)
```

2. **Tính toán số lượng vòng (ring count) trong mỗi hợp chất:**

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

def calculate_ring_count(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        return Descriptors.RingCount(mol)
    else:
        return None

# Apply the function to a DataFrame column
data['ring_count'] = data['structure'].apply(calculate_ring_count)

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

def calculate_ring_count(smiles):
    mol = Chem.MolFromSmiles(smiles)
    if mol is not None:
        return Descriptors.RingCount(mol)
```

```
    else:
        return None

# Áp dụng hàm cho một cột DataFrame
data['ring_count'] = data['structure'].apply(calculate_ring_count)
```

3. **Chuẩn hóa dữ liệu hoạt tính (ví dụ: chuyển đổi IC50 sang pIC50):**

```
# English
import numpy as np

def convert_ic50_to_pic50(ic50):
    if ic50 is not None and ic50 > 0:
        return -np.log10(ic50 / 1e9)  # Convert nM to M
    else:
        return None

# Apply the function to a DataFrame column
data['pIC50'] = data['standard_value'].apply(convert_ic50_to_pic50)

# Vietnamese
import numpy as np

def convert_ic50_to_pic50(ic50):
    if ic50 is not None and ic50 > 0:
        return -np.log10(ic50 / 1e9)  # Chuyển đổi nM sang M
    else:
        return None

# Áp dụng hàm cho một cột DataFrame
data['pIC50'] = data['standard_value'].apply(convert_ic50_to_pic50)
```

4. **Xây dựng mô hình Random Forest để dự đoán hoạt tính:**

```
# English
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

# Prepare data (assuming X and y are already defined)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Create and train the model
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Make predictions
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.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

# Chuẩn bị dữ liệu (giả sử X và y đã được định nghĩa)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
```

```
random_state=42)

# Tạo và huấn luyện mô hình
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Dự đoán
y_pred = model.predict(X_test)

# Đánh giá mô hình
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse}")
```

5. **Sử dụng PCA để giảm số chiều của dữ liệu:**

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

# Scale the data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Apply PCA
pca = PCA(n_components=10)  # Reduce to 10 components
X_pca = pca.fit_transform(X_scaled)

# Vietnamese
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler

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

# Áp dụng PCA
pca = PCA(n_components=10)  # Giảm xuống 10 thành phần
X_pca = pca.fit_transform(X_scaled)
```

**Lưu ý về lỗi:**

- **Lỗi SQL "operator does not exist: numeric ~ unknown":** 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 có thể cast cột `standard_value` sang kiểu text trước khi so sánh:

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

- **Lỗi "squared=False" trong mean_squared_error:** Nếu bạn đang sử dụng phiên bản scikit-learn cũ, hãy loại bỏ tham số `squared=False` hoặc nâng cấp lên phiên bản mới hơn.

**Lưu ý quan trọng:**

- Hãy nhớ thay thế các tên file và đường dẫn trong code mẫu bằng các giá trị phù hợp với cấu trúc thư mục và tên file của bạn.
- Đây chỉ là những ví dụ cơ bản. Bạn có thể cần điều chỉnh code để phù hợp với yêu cầu cụ thể của dự án.
- Việc lựa chọn mô hình phù hợp phụ thuộc vào đặc điểm của dữ liệu và mục tiêu của bạn. Hãy thử nghiệm với nhiều mô hình khác nhau và đánh giá hiệu suất của chúng một cách cẩn thận.

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