Topic

PHÂN TÍCH DỮ LIỆU Y TẾ ĐỂ DỰ ĐOÁN BỆNH TIM

Presented by: NHÓM 1

Group Member

- Nguyễn Ngọc Quỳnh Anh
- Hoàng Nguyên Vũ
- Lê Hoàng Gia Vĩ
- Phạm Trường Phát

PHÂN CHIA CÔNG VIỆC

QA

Viết báo cáo

Code
Làm slide

Vĩ

Viết báo cáo

Code
Làm slide

Vũ

Viết báo cáo

Code
Làm slide

Phát

Viết báo cáo

Code
Làm slide

NỘI DUNG

Bài toán

Bài toán: Dự đoán người đó có nguy cơ mắc bệnh tim dựa trên bộ dữ liệu có các thuộc tính đặc trưng liên về bệnh nhân mắc bệnh tim

Đầu vào: Các đặc trưng sinh lý và lâm sàng của bệnh nhân

Đầu ra: Xác định khả năng mắc bệnh tim

Tổng quan bộ data

Dữ liệu này bao gồm 920 dòng

Có 15 cột (chưa tiền xử lý dữ liệu)

Mô tả các biến:

- Id Số định danh bệnh nhân
- age Tuổi của bệnh nhân (tính theo năm)
- sex Giới tính của bệnh nhân
- cp Loại đau ngực
- trestbps Huyết áp khi nghỉ ngơi (mm Hg)
- chol Mức cholesterol trong máu (mg/dl)
- fbs Mức đường huyết lúc đói trên 120 mg/dl
- restecg Kết quả điện tâm đồ khi nghỉ ngơi

Tổng quan bộ data

- thalach Nhịp tim tối đa đạt được trong bài kiểm tra căng thẳng
- exang Đau thắt ngực do vận động
- oldpeak Mức độ suy giảm sóng ST khi tập thể dục so với khi nghỉ
- slope Độ dốc của đoạn sóng ST cao nhất khi tập thể dục
- ca Số lượng mạch máu chính được phát hiện bằng chụp huỳnh quang
- thal Kết quả kiểm tra stress thallium
- target Tình trạng bệnh tim (0 = không mắc, 1 = mắc bệnh)

```
heart_data.sample(10)
```

	id	age	sex	cp	trestbps	chol	fbs	restecg	thalch	exang	oldpeak	slope	ca	thal	target
344	345	40	Male	atypical angina	140.0	289.0	False	normal	172.0	False	0.0	NaN	NaN	NaN	0
870	871	41	Male	asymptomatic	150.0	171.0	False	normal	128.0	True	1.5	flat	NaN	NaN	0
817	818	62	Male	typical angina	112.0	258.0	False	st-t abnormality	150.0	True	NaN	NaN	NaN	NaN	1
237	238	46	Male	asymptomatic	120.0	249.0	False	lv hypertrophy	144.0	False	0.8	upsloping	0.0	reversable defect	1
27	28	66	Female	typical angina	150.0	226.0	False	normal	114.0	False	2.6	downsloping	0.0	normal	0
889	890	57	Male	atypical angina	180.0	285.0	True	st-t abnormality	120.0	False	0.8	NaN	NaN	NaN	1
194	195	68	Female	non-anginal	120.0	211.0	False	lv hypertrophy	115.0	False	1.5	flat	0.0	normal	0
379	380	45	Male	asymptomatic	120.0	225.0	False	normal	140.0	False	0.0	NaN	NaN	NaN	0
451	452	54	Male	atypical angina	110.0	208.0	False	normal	142.0	False	0.0	NaN	NaN	NaN	0
827	828	59	Male	asymptomatic	124.0	NaN	False	normal	117.0	True	1.0	flat	NaN	NaN	1

```
heart_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 920 entries, 0 to 919
Data columns (total 15 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   id        920 non-null    int64
 1   age       920 non-null    int64
 2   sex       920 non-null    object
 3   cp        920 non-null    object
 4   trestbps  861 non-null    float64
 5   chol      890 non-null    float64
 6   fbs       830 non-null    object
 7   restecg   918 non-null    object
 8   thalch    865 non-null    float64
 9   exang     865 non-null    object
 10  oldpeak   858 non-null    float64
 11  slope     611 non-null    object
 12  ca        309 non-null    float64
 13  thal      434 non-null    object
 14  target    920 non-null    int64
dtypes: float64(5), int64(3), object(7)
memory usage: 107.9+ KB
```

Nhóm nhận thấy rằng một số vấn đề cần giải quyết trước khi xây dựng mô hình dự đoán:

- Dữ liệu chứa nhiều loại biến khác nhau
- Dữ liệu còn chứa các giá trị thiếu khá nhiều

=> Việc tiền xử lý rất quan trọng để đảm bảo chất lượng cho các bước phân tích và mô hình hóa

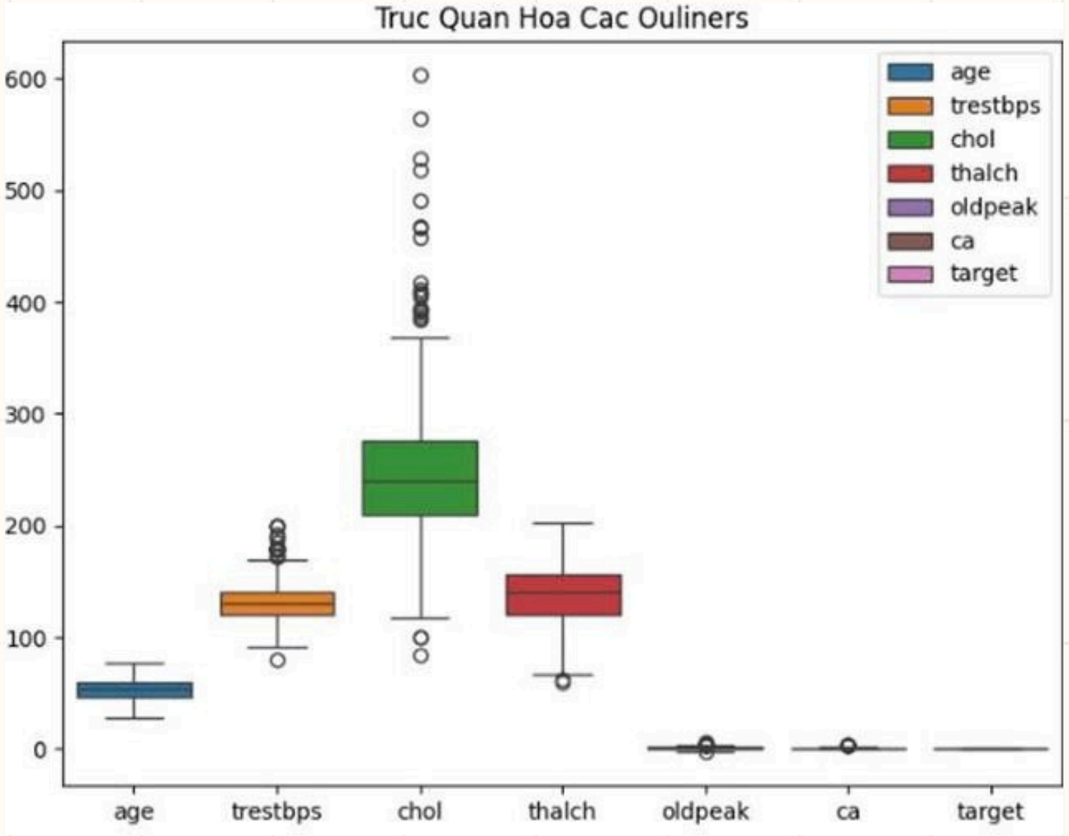
Data preprocessing

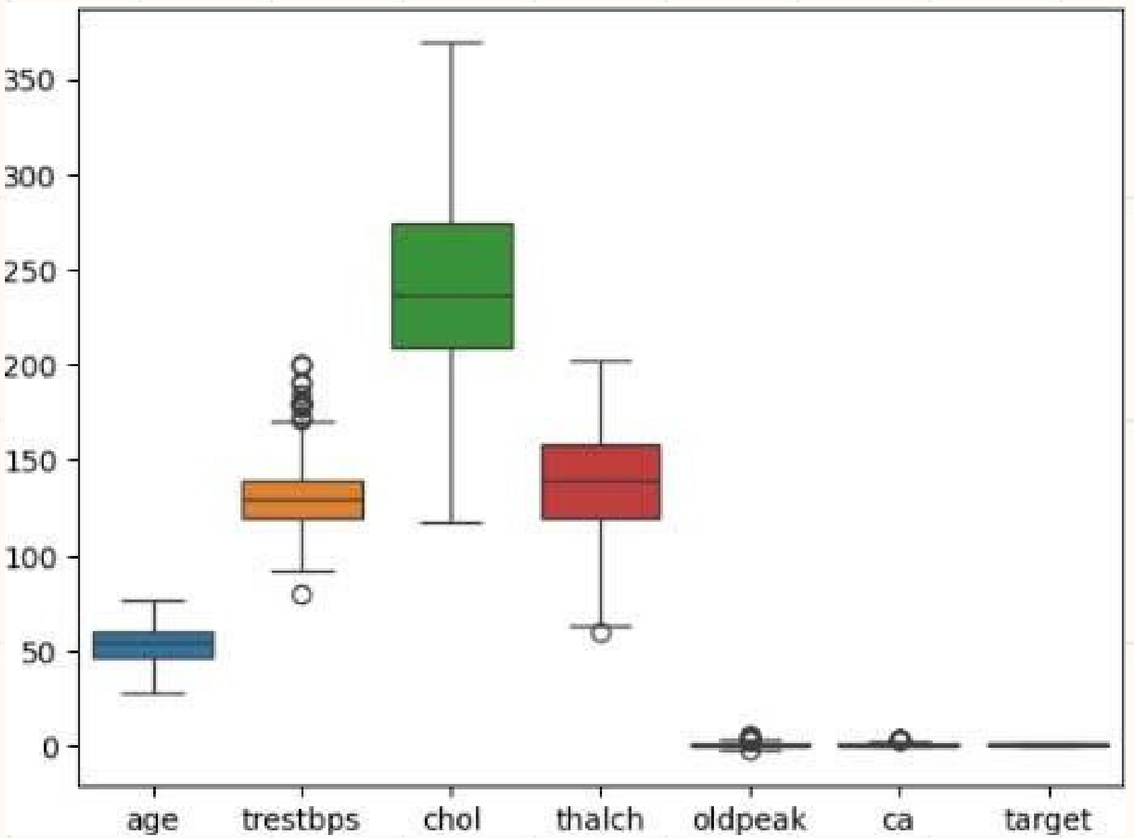
Làm sạch dữ liệu

Chuyển đổi dữ liệu

Giảm kích thước dữ liệu

Làm sạch dữ liệu





Giữ lại các giá trị ngoại lai của các thuộc tính trestbps, thalch, oldpeak, và ca.

Tiến hành tách biến phân loại

```
categories_col = [column for column in df_heart.columns if column not in numeric_col]
categories_col
```

```
['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'thal']
```

Sử dụng OrdinalEncoder để mã hóa

```
from sklearn.preprocessing import OrdinalEncoder
import numpy as np

ordinal_encoder = OrdinalEncoder()

# Chỉ encode các cột phân loại
df_heart[categories_col] = ordinal_encoder.fit_transform(df_heart[categories_col])
```

Chuyển đổi dữ liệu

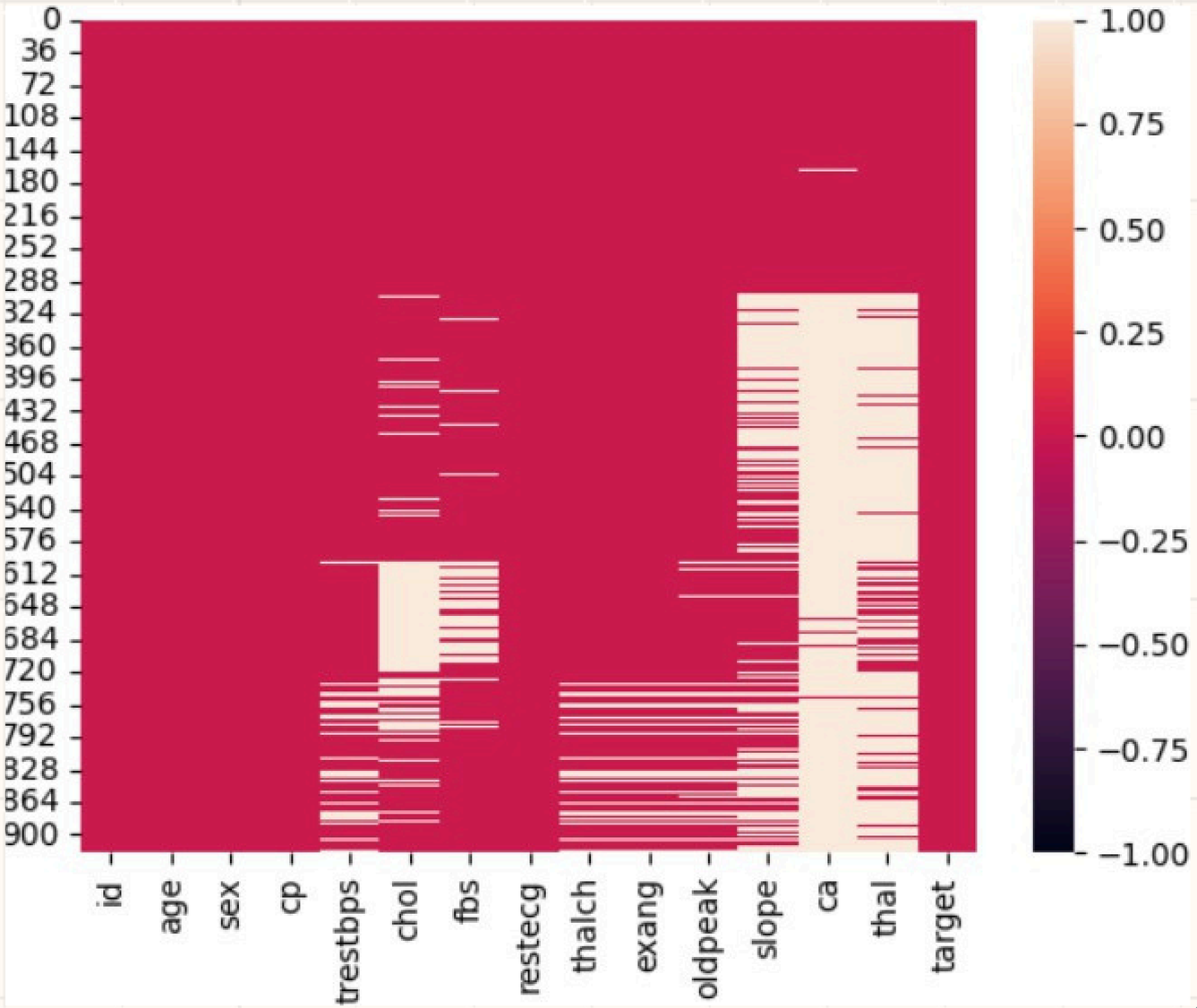
Mã hóa các biến phân loại

Làm sạch dữ liệu

Xử lý các giá trị khuyết

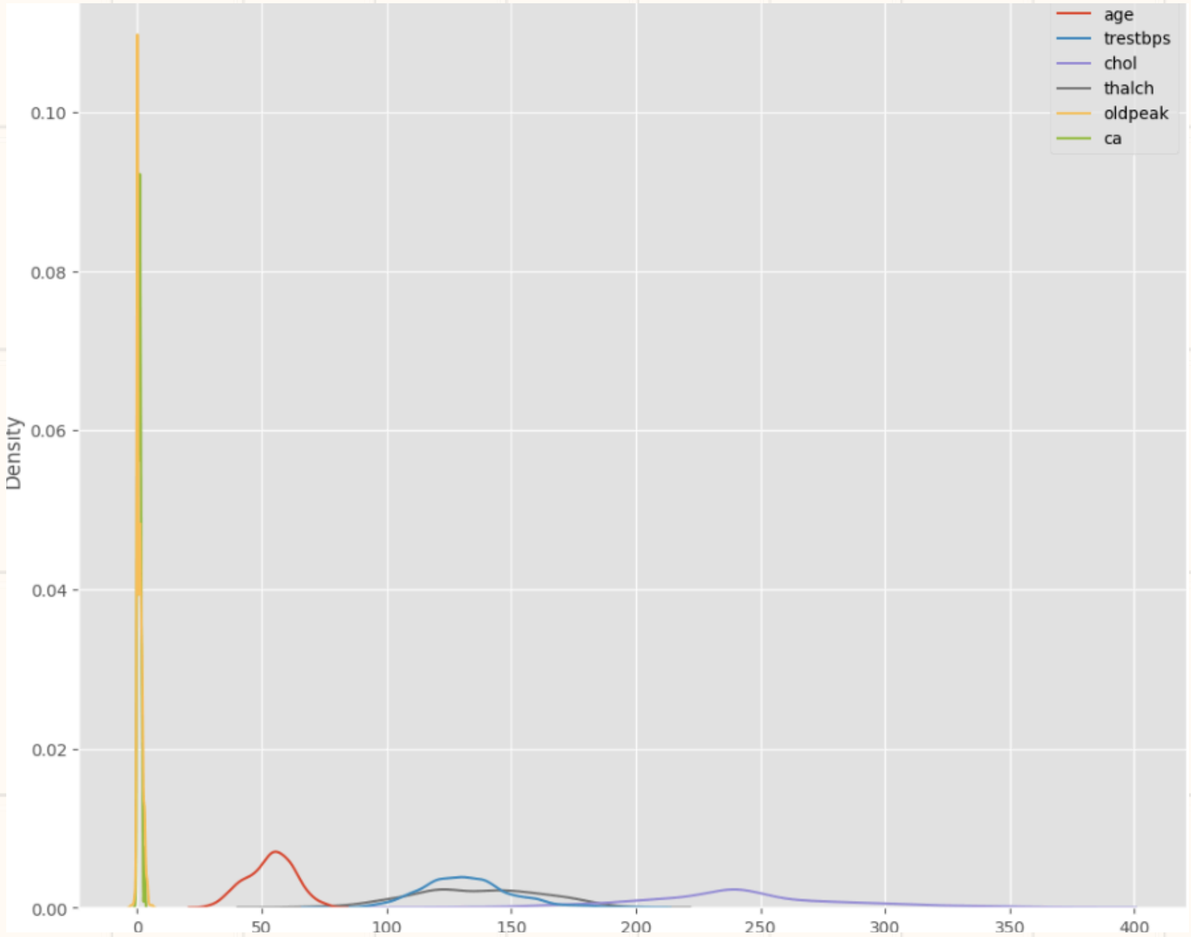
Biến số: sử dụng phương pháp MICE

Biến phân loại: điền giá trị thiếu bằng giá trị phổ biến thuộc tính

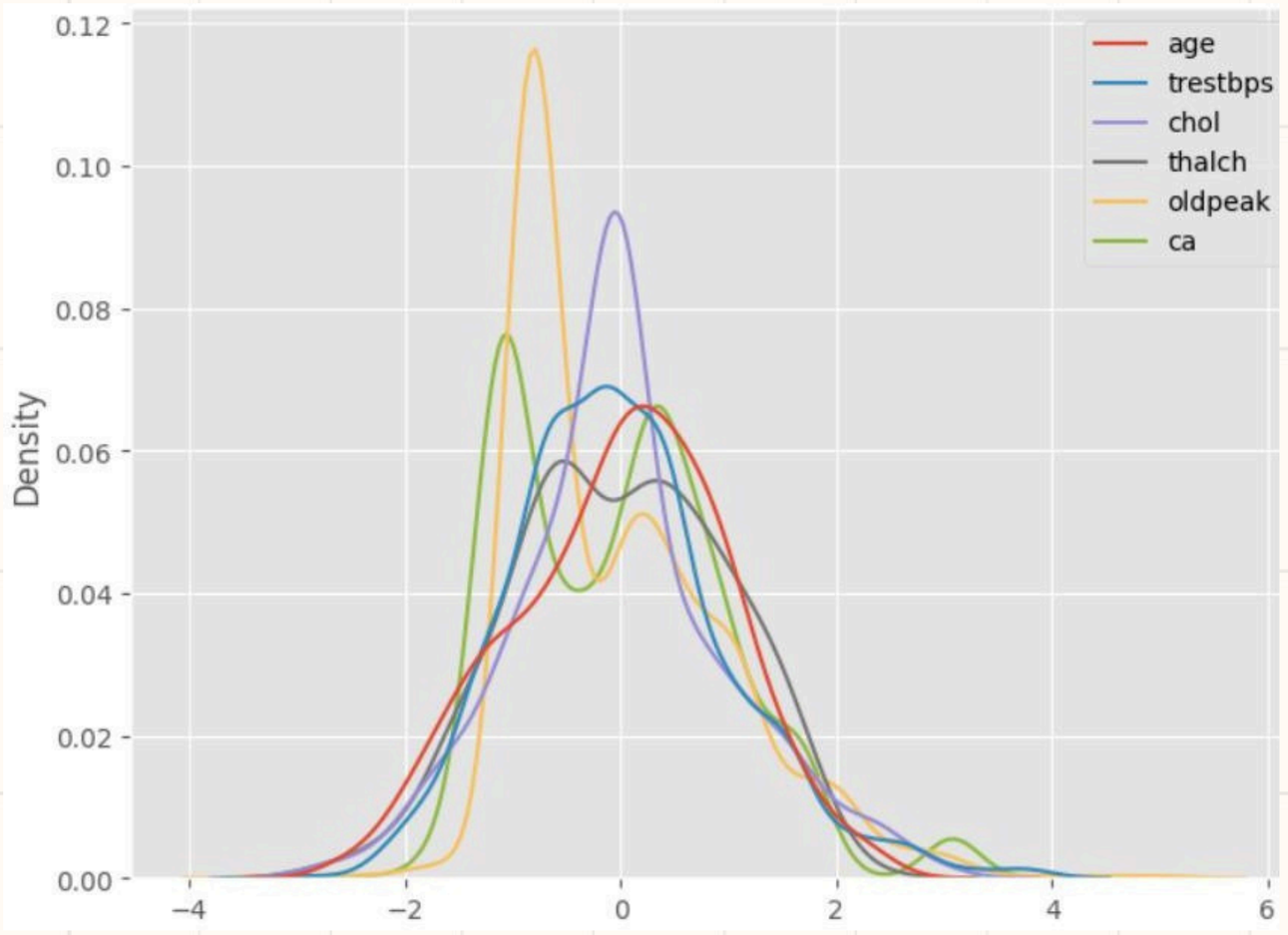


Chuyển đổi dữ liệu

Chuẩn hóa dữ liệu



Dữ liệu ban đầu khi chưa chuẩn hóa



Dữ liệu sau khi được chuẩn hóa

```
# Tỉ lệ phương sai giải thích
explained_var = pca.explained_variance_ratio_
print(f'Ty le phuong sai giai thich', explained_var)
```

```
Ty le phuong sai giai thich [0.25350351 0.10924564 0.09599767 0.07592013 0.07093689 0.0687054
 0.05942983 0.0553865  0.05057277 0.04896809 0.03660282 0.02898128
 0.02751443 0.01823505]
```

```
# Phương sai tích luỹ
cumsum_explained_var = np.cumsum(pca.explained_variance_ratio_)
print(f'Phuong sai tich luy', cumsum_explained_var)
```

```
Phuong sai tich luy [0.25350351 0.36274915 0.45874682 0.53466696 0.60560385 0.67430925
 0.73373908 0.78912557 0.83969834 0.88866643 0.92526925 0.95425053
 0.98176495 1.         ]
```

Giảm kích thước dữ liệu

```
# Lựa chọn thanh phần chính
n_components = np.argmax(cumsum_explained_var >= 0.8)+1
print(f'So luong thanh phan chinh duoc chon la: {n_components}'
```

```
luong thanh phan chinh duoc chon la: 9
```

Giảm kích thước dữ liệu

XÂY DỰNG VÀ THỬ NGHIỆM MÔ HÌNH

Chia tập dữ liệu

Huấn luyện mô hình

So sánh các mô hình

XÂY DỰNG VÀ THỬ NGHIỆM MÔ HÌNH

Chia tập dữ liệu

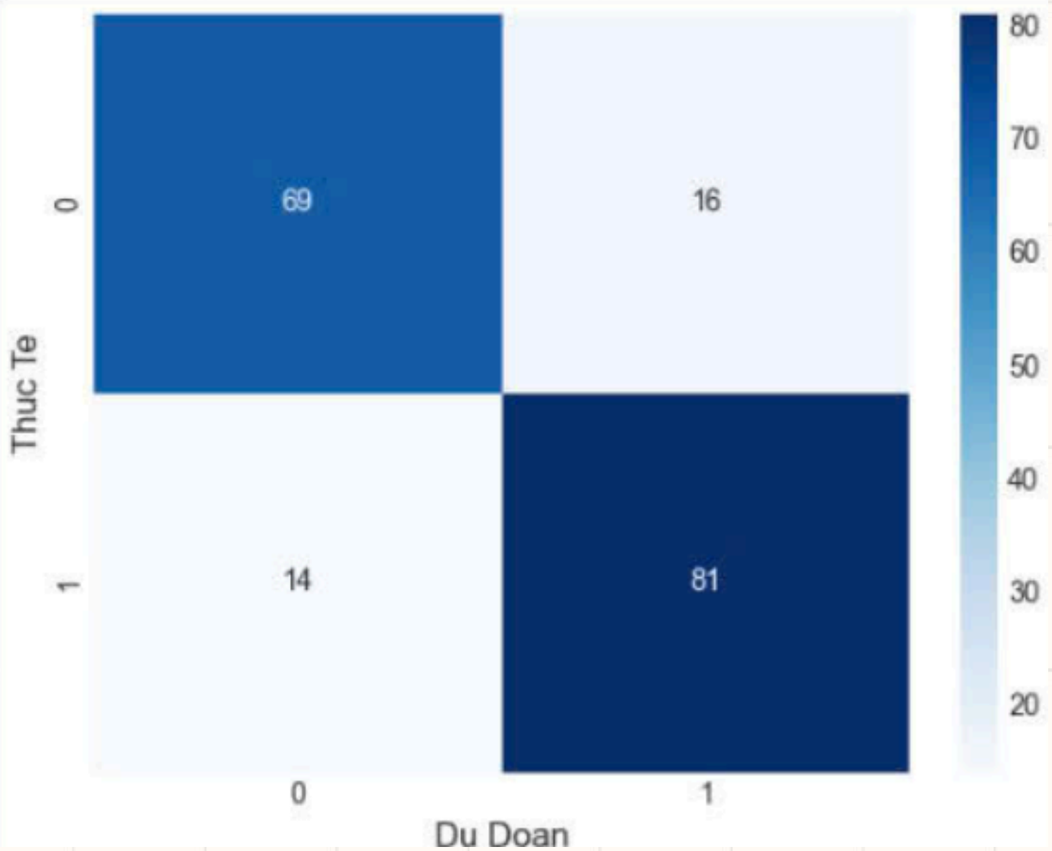
```
X_train, X_test, y_train, y_test =
train_test_split(X_pca_reduced, y, test_size=0.2,
random_state=42)
```

Dữ liệu được chia thành hai tập: tập huấn luyện chiếm 80% để đào tạo mô hình, và tập kiểm tra chiếm 20% để đánh giá hiệu năng

Logistic Regression

```python
from sklearn.linear_model import LogisticRegression
# Huan Luyen Model
model = LogisticRegression()
model.fit(X_train, y_train)

# Du doan tren tap test
pred_y = model.predict(X_test)
print(pred_y)
```

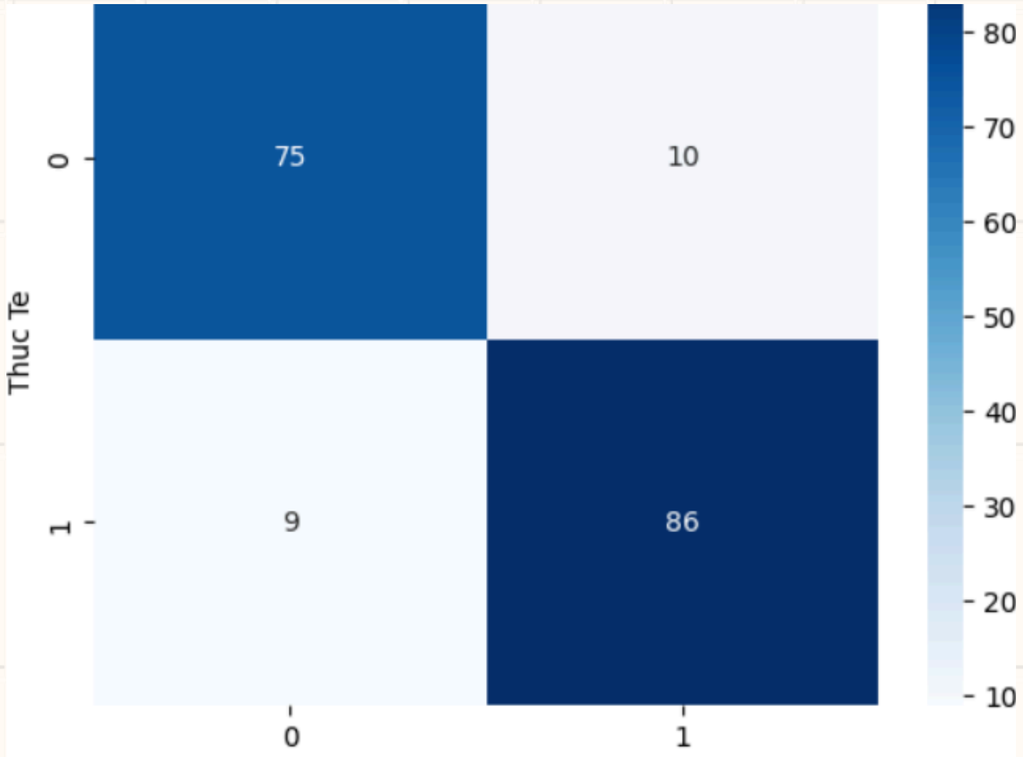
```
--------------------------------------
Accuracy: 0.8333333333333334
--------------------------------------
Classification Report:
              precision    recall  f1-score   support

         0.0       0.83      0.81      0.82        85
         1.0       0.84      0.85      0.84        95

    accuracy                           0.83       180
   macro avg       0.83      0.83      0.83       180
weighted avg       0.83      0.83      0.83       180
```

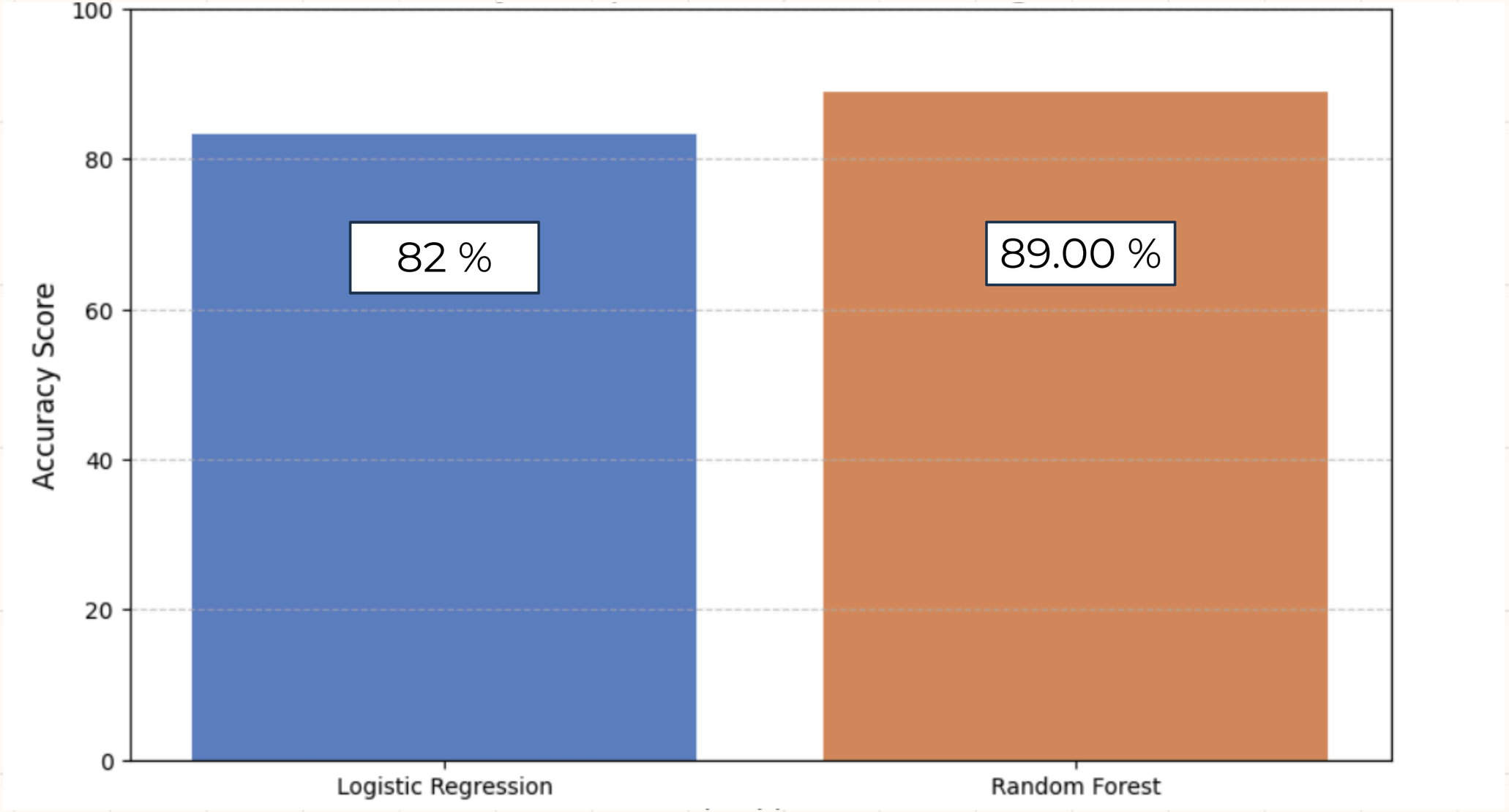


Random Forest

```python
from sklearn.ensemble import RandomForestClassifier
# Huan luyen Model
model = RandomForestClassifier()
model.fit(X_train, y_train)

# Du doan tren tap test
pred_y = model.predict(X_test)
print(pred_y)
```

```
Accuracy: 0.89
Classification Report:
              precision    recall  f1-score   support

         0.0       0.89      0.88      0.89        85
         1.0       0.90      0.91      0.90        95

    accuracy                           0.89       180
   macro avg       0.89      0.89      0.89       180
weighted avg       0.89      0.89      0.89       180
```




100
80
60
40
20
0
Accuracy Score
82 %
89.00 %
Logistic Regression
Random Forest

KẾT LUẬN

Mô hình Random Forest Classifier được xác định là phù hợp hơn cho bài toán

Việc thu thập dữ liệu cho bài toán ở cỡ mẫu nhỏ chưa tổng quát hết kết quả

Hướng phát triển nhóm sẽ thu thập thêm nhiều bộ dữ liệu và mở rộng thử nghiệm với các thuật toán phân loại GradientBoosting, XGBoost

Thank You

Presented by

NHÓM 1