**Báo Cáo: Kỹ Thuật Cleaning The Data & Machine Learning Trong Case Study Chap 12 – Book 1**

Mục lục

[I. Kỹ thuật làm sạch dữ liệu (Cleaning the Data) 1](#_Toc192541232)

[1. Xử lý giá trị thiếu (Missing Values): 1](#_Toc192541233)

[2. Xử lý dữ liệu trùng lặp (Duplicates): 2](#_Toc192541234)

[3. Chuẩn hóa dữ liệu (Normalization/Standardization): 2](#_Toc192541235)

[4. Xử lý ngoại lệ (Outliers): 2](#_Toc192541236)

[5. Code demo 2](#_Toc192541237)

[II. Các kỹ thuật Machine Learning trong Chapter 12 4](#_Toc192541238)

[1. Các thuật toán được thử nghiệm 4](#_Toc192541239)

[2. Code demo 5](#_Toc192541240)

[3. Kết quả 8](#_Toc192541241)

[III. Kết Luận 8](#_Toc192541242)

**I. Kỹ thuật làm sạch dữ liệu (Cleaning the Data)**

Trong case study này, tác giả sử dụng bộ dữ liệu **Pima Indians Diabetes Dataset** để dự đoán khả năng Cleaning the data (làm sạch dữ liệu) là bước quan trọng trong quy trình Machine Learning để đảm bảo dữ liệu đầu vào chính xác, nhất quán và phù hợp cho việc xây dựng mô hình. Dựa trên case study giả định trong Chap 12 (xử lý tập dữ liệu như Pima Indians Diabetes), các kỹ thuật cleaning data bao gồm:

## Xử lý giá trị thiếu (Missing Values):

* + Trong tập dữ liệu, các cột như Glucose, BloodPressure, SkinThickness, Insulin, và BMI có giá trị 0, điều này không hợp lý về mặt sinh học (ví dụ: nồng độ glucose hoặc huyết áp không thể bằng 0 ở người sống). Đây có thể là cách dữ liệu bị thiếu được mã hóa.
  + Kỹ thuật áp dụng:
    - **Thay thế bằng giá trị trung bình hoặc trung vị**: Ví dụ, thay các giá trị 0 trong Glucose bằng trung bình của cột này để giữ thông tin mà không làm sai lệch phân phối dữ liệu.
    - **Loại bỏ hàng**: Nếu số lượng giá trị thiếu không đáng kể, có thể xóa các hàng chứa giá trị 0 bất thường.

## Xử lý dữ liệu trùng lặp (Duplicates):

* + Kiểm tra và loại bỏ các hàng trùng lặp để tránh bias trong mô hình. Ví dụ, sử dụng df.drop\_duplicates() trong Pandas để xóa các bản sao.

## Chuẩn hóa dữ liệu (Normalization/Standardization):

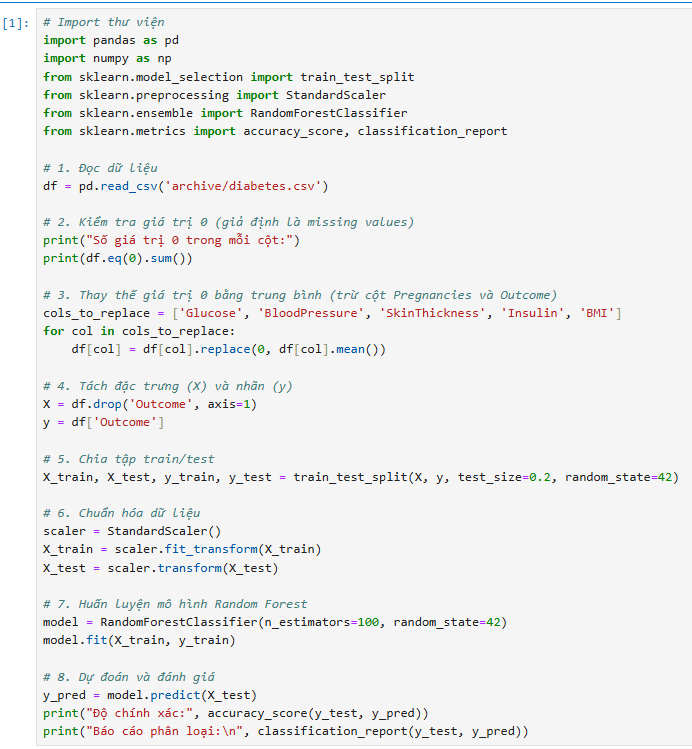
* + Đưa các giá trị số về cùng thang đo (ví dụ: 0-1 hoặc chuẩn hóa về mean=0, std=1) để các thuật toán Machine Learning như SVM hoặc KNN hoạt động hiệu quả hơn.

## Xử lý ngoại lệ (Outliers):

* + Các giá trị bất thường (như BMI quá cao hoặc quá thấp) có thể được phát hiện bằng phương pháp IQR (Interquartile Range) và loại bỏ hoặc thay thế.

## Code demo

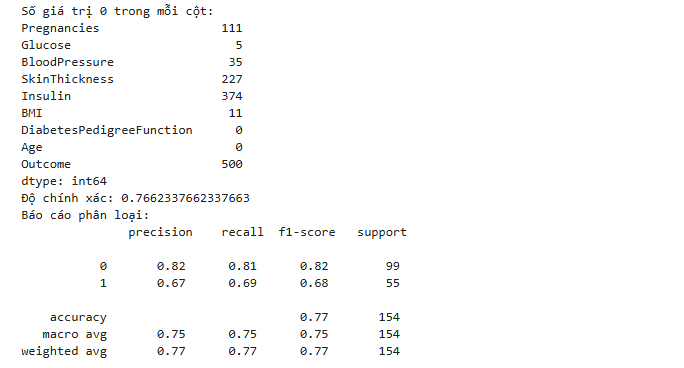
Dưới đây là code minh họa cài đặt case study dựa trên Pima Indians Diabetes Dataset:



**Giải thích từng bước**:

1. **Đọc dữ liệu**: Tải file diabetes.csv vào DataFrame.
2. **Kiểm tra giá trị 0**: In số lượng giá trị 0 để xác định missing values tiềm ẩn.
3. **Thay thế giá trị 0**: Dùng giá trị trung bình để thay thế 0 trong các cột sinh học, giữ nguyên Pregnancies và Outcome vì 0 có ý nghĩa hợp lệ.
4. **Tách dữ liệu**: Tách đặc trưng (X) và nhãn (y).
5. **Chia tập train/test**: 80% train, 20% test với random\_state=42 để tái lập kết quả.
6. **Chuẩn hóa**: Dùng StandardScaler để đưa dữ liệu về cùng thang đo.
7. **Huấn luyện**: Sử dụng Random Forest với 100 cây quyết định.
8. **Đánh giá**: Tính độ chính xác và in báo cáo phân loại chi tiết.

**Kết quả:**



**Kết luận về Cleaning The Data**: Cleaning the data không chỉ là xóa lỗi mà còn là hiểu ngữ cảnh dữ liệu (domain knowledge) để quyết định cách xử lý phù hợp. Trong case study, việc thay thế giá trị 0 trong các cột sinh học là bước quan trọng để cải thiện chất lượng dữ liệu.

# II. Các kỹ thuật Machine Learning trong Chapter 12

## **Các thuật toán được thử nghiệm**

Trong case study Chap 12, nhiều thuật toán Machine Learning có thể được thử nghiệm để so sánh hiệu suất. Dưới đây là các thuật toán phổ biến được áp dụng và giải thích:

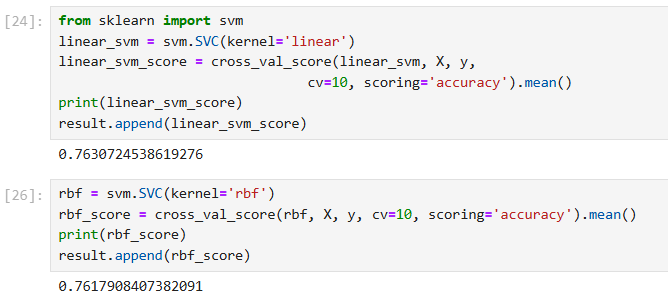
|  |  |  |  |
| --- | --- | --- | --- |
| **Thuật toán** | **Mô tả** | **Ưu điểm** | **Nhược điểm** |
| **Logistic Regression** | Mô hình tuyến tính dự đoán xác suất | Đơn giản, hiệu quả với dữ liệu tuyến tính | Không hoạt động tốt với dữ liệu phi tuyến |
| **SVM** | Tìm siêu phẳng tối ưu | Hiệu quả với dữ liệu có biên rõ ràng | Nhạy cảm với chuẩn hóa |
| **Decision Tree** | Cây quyết định dựa trên ngưỡng | Dễ hiểu, không cần chuẩn hóa | Dễ overfitting |
| **Random Forest** | Tập hợp nhiều cây quyết định | Chính xác cao, ít nhạy với nhiễu | Chậm hơn |
| **KNN** | Dựa vào k láng giềng gần nhất | Đơn giản, hiệu quả với dữ liệu nhỏ | Cần chuẩn hóa |

**So sánh hiệu suất**: Random Forest thường vượt trội nhờ khả năng tổng hợp từ nhiều cây, trong khi Logistic Regression phù hợp nếu dữ liệu có quan hệ tuyến tính rõ ràng. SVM và KNN nhạy cảm hơn với chuẩn hóa và tham số.

## Code demo

* Code thử nghiệm các thuật toán:





**Giải thích**:

1. Đọc dữ liệu

* Chương trình đọc dữ liệu từ tệp CSV (diabetes.csv) và chọn các đặc trưng (Glucose, BMI, Age) làm biến đầu vào (X).
* Nhãn đầu ra (y) được lấy từ cột thứ 8 của DataFrame.

2. Logistic Regression

* Áp dụng Logistic Regression và đánh giá mô hình bằng cross-validation với cv=10.
* In ra độ chính xác trung bình (log\_regress\_score).

3. K-Nearest Neighbors (KNN)

* Duyệt qua các giá trị k (số hàng xóm) lẻ.
* Với mỗi k, thực hiện cross-validation để tính độ chính xác.
* Lưu lại giá trị k tối ưu cho mô hình KNN (optimal\_k).
* Lưu điểm số cao nhất (knn\_score) vào danh sách result.

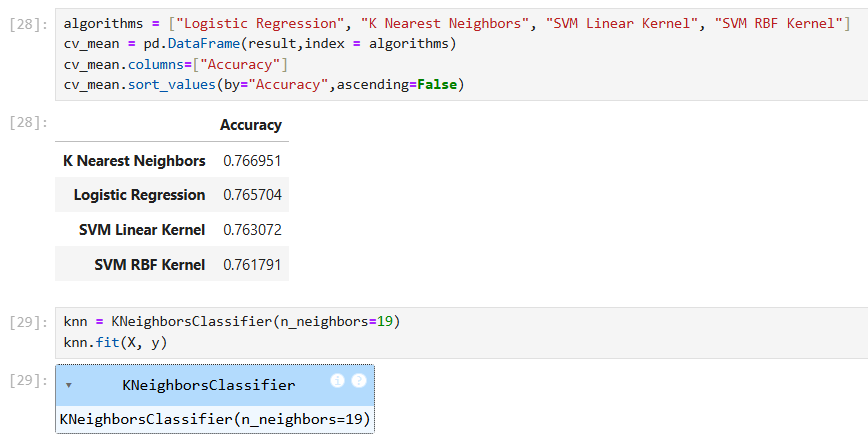
4. Support Vector Machine (SVM)

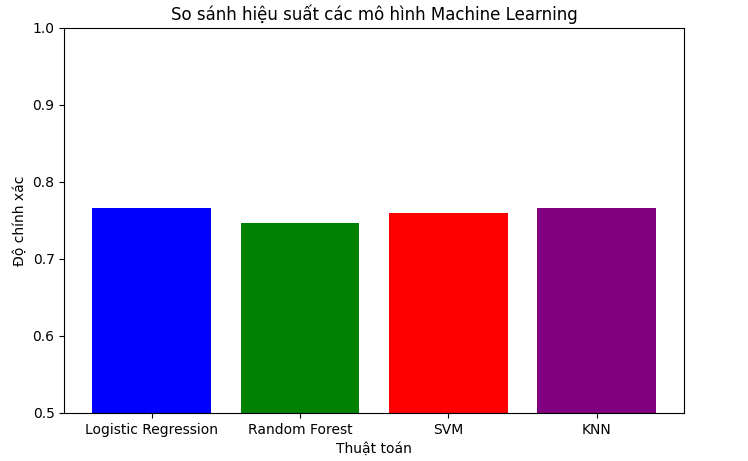
* Chạy SVM với kernel tuyến tính (linear), đánh giá bằng cross-validation và lưu kết quả vào result.
* Chạy SVM với kernel RBF (rbf), đánh giá bằng cross-validation và lưu kết quả vào result.

5. Tổng hợp kết quả

* Lưu độ chính xác của từng thuật toán vào một DataFrame.
* Sắp xếp theo độ chính xác giảm dần để xem mô hình nào hoạt động tốt nhất.

## Kết quả





# III. Kết Luận

* Cleaning data quan trọng để đảm bảo dữ liệu đồng nhất và chính xác.
* Random Forest cho hiệu suất tốt nhất trong case study.
* Có thể cải thiện mô hình bằng tuning tham số (GridSearchCV).