**3. Phân Tích Mô Hình**

**3.1. Tiền xử lý dữ liệu**

Dữ liệu được phân chia thành 2 bộ dữ liệu riêng biệt cho việc đánh giá hiệu quả của mô hình:

* **Tập huấn luyện (Training set):** Dạy cho mô hình về các mối quan hệ giữa các dữ liệu với tỷ lệ 80% bộ dữ liệu gốc.
* **Tập kiểm tra (Testing set):** Đánh giá độ chính xác dự đoán của mô hình trên những dữ liệu mà nó chưa từng thấy trước đây, sử dụng 20% còn lại của bộ dữ liệu gốc.

**3.2. Phân tích, khám phá dữ liệu**

**Phân tích Mối tương quan giữa các biến:** Sử dụng ma trận tương quan và biểu đồ nhiệt để xem xét mối liên quan tuyến tính giữa các thuộc tính. Qua đó, nhóm đã xác định được các biến có ảnh hưởng mạnh nhất đến biến mục tiêu target như cp, thalach (tương quan dương) và exang, oldpeak (tương quan âm), giúp định hướng cho việc lựa chọn đặc trưng.

**Phân tích Phân bố của các biến quan trọng:**

* **Phân bố tuổi:** Bệnh nhân tập trung chủ yếu ở độ tuổi 51-62.
* **Phân bố biến mục tiêu:** Dữ liệu khá cân bằng với 165 ca mắc bệnh và 138 ca không mắc bệnh.
* **Phân tích theo nhóm tuổi:** Dữ liệu được chia thành các nhóm tuổi để phân tích tỷ trọng.

**3.3. Phân tích và lựa chọn các mô hình Machine Learning**

Để giải quyết bài toán phân loại nhị phân, một loạt các thuật toán học máy cổ điển đại diện cho các phương pháp tiếp cận khác nhau đã được lựa chọn. Việc lựa chọn đa dạng mô hình mang đến một góc nhìn toàn diện và so sánh được ưu, nhược điểm của từng phương pháp trên cùng một bộ dữ liệu.

Các mô hình được lựa chọn bao gồm:

1. **Logistic Regression (Hồi quy Logistic):**
   * **Phân tích:** Đây là một mô hình tuyến tính, tuy đơn giản nhưng rất hiệu quả và thường được sử dụng làm mô hình cơ sở trong các bài toán phân loại. Nó hoạt động bằng cách ước tính xác suất một mẫu thuộc về một lớp cụ thông qua hàm sigmoid. Ưu điểm của mô hình là dễ diễn giải và tính toán nhanh.
   * **Lý do lựa chọn:** Phù hợp để thiết lập một ngưỡng hiệu suất ban đầu để so sánh với các mô hình phức tạp hơn.
2. **K-Nearest Neighbors (KNN - K Láng giềng gần nhất):**
   * **Phân tích:** Thuật toán phân loại một điểm dữ liệu mới bằng cách dựa vào số lượng của K điểm dữ liệu gần nhất trong không gian đặc trưng. Đây là một thuật toán trực quan và dễ triển khai. Tuy nhiên, hiệu suất của nó phụ thuộc nhiều vào việc lựa chọn tham số K và độ đo khoảng cách.
   * **Lý do lựa chọn:** Để đánh giá một phương pháp tiếp cận phi tuyến, dựa trên sự tương đồng thay vì tìm một hàm phân tách rõ ràng.
3. **Support Vector Machine (SVM - Máy Véc-tơ Hỗ trợ):**
   * **Phân tích:** SVM là một thuật toán mạnh mẽ, hoạt động bằng cách tìm ra một siêu phẳng trong không gian nhiều chiều để phân tách các lớp dữ liệu một cách tối ưu nhất. Mục tiêu là tối đa hóa khoảng cách giữa siêu phẳng và các điểm dữ liệu gần nhất của mỗi lớp. SVM đặc biệt hiệu quả trong không gian có số chiều cao.
   * **Lý do lựa chọn:** Để thử nghiệm một mô hình dựa trên việc tối ưu hóa biên phân loại, có khả năng xử lý tốt các mối quan hệ phức tạp.
4. **Naive Bayes:**
   * **Phân tích:** Là một thuật toán phân loại dựa trên xác suất, áp dụng định lý Bayes với một giả định "ngây thơ" rằng các đặc trưng là độc lập với nhau. Mặc dù giả định này hiếm khi đúng trong thực tế, Naive Bayes vẫn hoạt động tốt một cách đáng ngạc nhiên trong nhiều bài toán, đặc biệt với các bộ dữ liệu nhỏ. Nó có ưu điểm là tốc độ huấn luyện rất nhanh.
   * **Lý do lựa chọn:** Để kiểm tra hiệu quả của một phương pháp tiếp cận dựa trên xác suất thống kê.
5. **Random Forest (Rừng Ngẫu nhiên):**
   * **Phân tích:** Đây là một thuật toán học tập hợp thuộc loại bagging. Random Forest xây dựng một "rừng" gồm nhiều cây quyết định trong quá trình huấn luyện. Để phân loại một mẫu mới, nó lấy kết quả dự đoán từ mỗi cây và chọn ra lớp được "bỏ phiếu" nhiều nhất. Kỹ thuật này giúp cải thiện đáng kể độ chính xác và giảm thiểu hiện tượng học vẹt so với việc chỉ dùng một cây quyết định đơn lẻ.
   * **Lý do lựa chọn:** Để đánh giá sức mạnh của các phương pháp học tập hợp, vốn thường cho kết quả hàng đầu trong nhiều cuộc thi học máy.

**3.4. Phân tích và lựa chọn các mô hình Deep Learning**

Để khám phá khả năng nâng cao hiệu suất và tìm ra các mối quan hệ phi tuyến phức tạp hơn mà các mô hình cổ điển có thể bỏ qua, **nhóm chúng em đã quyết định xây dựng và bổ sung một mô hình Deep Learning**, cụ thể là mạng **Multi-layer Perceptron (MLP)**.

* **Phân tích:** MLP là một mạng nơ-ron truyền thẳng (Feedforward Network), rất phù hợp cho các bài toán với dữ liệu có cấu trúc dạng bảng như bộ dữ liệu bệnh tim. Với nhiều lớp ẩn, nó có khả năng học các đặc trưng ở nhiều mức độ trừu tượng khác nhau.
* **Lý do lựa chọn:**
  + **Khả năng học các mẫu phức tạp:** MLP có thể mô hình hóa các mối quan hệ phi tuyến tính phức tạp giữa các chỉ số y tế và nguy cơ mắc bệnh tim.
  + **Kiến trúc linh hoạt:**
    - Sử dụng hàm kích hoạt **ReLU** ở các lớp ẩn để giúp mô hình học nhanh hơn và tránh được vấn đề triệt tiêu đạo hàm (Vanishing Gradients).
    - Sử dụng hàm kích hoạt **Sigmoid** ở lớp đầu ra để đưa ra kết quả dưới dạng xác suất (từ 0 đến 1), rất phù hợp cho bài toán phân loại nhị phân.
  + **Tích hợp các kỹ thuật hiện đại:**
    - Sử dụng thuật toán tối ưu **Adam**, một lựa chọn phổ biến và mạnh mẽ, giúp mô hình hội tụ nhanh và ổn định.
    - Áp dụng kỹ thuật **Dropout** như một phương pháp điều chuẩn (Regularization) hiệu quả để chống lại hiện tượng học vẹt (Overfitting), đặc biệt hữu ích khi làm việc với các bộ dữ liệu có kích thước không quá lớn.

**4. Cài Đặt Thử Nghiệm**

Mã nguồn code được đẩy lên github tại link sau:

**4.1. Cài đặt môi trường**

* **Ngôn ngữ:** Python 3.
* **Thư viện:** TensorFlow, Keras, Scikit-learn, Pandas, NumPy, Matplotlib.

**4.2. Quy trình tiền xử lý dữ liệu**

1. **Phân chia dữ liệu:** Chia bộ dữ liệu thành 2 tập:
   * Tập huấn luyện (Training set): 80%
   * Tập kiểm thử (Test set): 20%
2. **Chuẩn hóa dữ liệu (Data Scaling):** Áp dụng StandardScaler từ Scikit-learn cho các thuộc tính số. Việc này giúp đưa tất cả các thuộc tính về cùng một thang đo, giúp quá trình huấn luyện mô hình hội tụ nhanh và ổn định hơn.

**4.3. Huấn luyện mô hình**

**4.3.1. Thực nghiệm mô hình Machine Learning (MLP)**

Kết quả đánh giá độ chính xác trên tập kiểm tra của các mô hình Machine Learning cơ sở như sau:

|  |  |  |
| --- | --- | --- |
| **STT** | **Mô hình** | **Độ chính xác trên tập Test** |
| 1 | Logistic Regression | **85.25%** |
| 2 | K-Nearest Neighbors (KNN) | 63.93% |
| 3 | Support Vector Machine (SVM) | 68.85% |
| 4 | Naive Bayes | **85.25%** |
| 5 | Random Forest | **85.25%** |

**Phân tích:** Các mô hình **Logistic Regression**, **Naive Bayes**, và **Random Forest** cho kết quả tốt nhất và đồng đều. Điều này cho thấy các phương pháp khác nhau có thể đạt được hiệu suất tương đương trên bộ dữ liệu này. Hiệu suất thấp của KNN và SVM có thể do sự nhạy cảm của chúng với việc co giãn các dữ liệu đặc trưng.

##### **4.3.2. Thực nghiệm mô hình Deep Learning (MLP)**

Mô hình MLP được xây dựng và huấn luyện bằng thư viện PyTorch.

* **Cài đặt thực nghiệm:**
  + **Kiến trúc:** Mạng nơ-ron truyền thẳng gồm 1 lớp đầu vào (13 nơ-ron), 2 lớp ẩn (64 và 32 nơ-ron với hàm kích hoạt ReLU), và 1 lớp đầu ra (1 nơ-ron).
  + **Hàm mất mát:** Binary Cross-Entropy (BCEWithLogitsLoss).
  + **Thuật toán tối ưu:** Adam với learning rate là 0.001.
  + **Kỹ thuật chống quá khớp:** Áp dụng Dropout với tỷ lệ 0.3.
  + **Huấn luyện:** Mô hình được huấn luyện trong 50 epochs với kích thước batch là 32.
* **Kết quả từ mô hình Deep Learning:** Sau khi huấn luyện, mô hình MLP đã cho thấy sự vượt trội so với các mô hình Machine Learning, đạt độ chính xác trên tập kiểm tra là **88.52%**.