

TRƯỜNG ĐẠI HỌC HỌC VĂN LANG
KHOA CÔNG NGHỆ THÔNG TIN

VAN LANG
UNIVERSITY



BÁO CÁO ĐỒ ÁN MÔN HỌC 71ITDS40303 - HK223
NHẬP MÔN PHÂN TÍCH DỮ LIỆU LỚN

Chủ đề: *Ứng dụng công nghệ lớn trong việc quản lý hồ sơ bệnh nhân bị ung thư phổi*

1. Trần Hữu Luân – 2274802010520 (Trưởng nhóm)
2. Lê Duy Khang - 2274802010374
3. Nguyễn Hoàng Giới – 2174802010831
4. Huỳnh Linh Trung - 197CT22641
5. Nguyễn Thanh Trường – 207CT47995

GV: Hoàng Lê Minh

Lời Cảm Ơn

Trước tiên, chúng em xin cảm ơn sâu sắc đến thầy Hoàng Lê Minh – người thầy đã tận tâm truyền đạt những kiến thức, đồng thời luôn khích lệ và định hướng cho chúng em trong suốt quá trình học tập môn "Nhập môn Phân tích Dữ liệu Lớn". Môn học không chỉ trang bị cho chúng em nền tảng lý thuyết vững chắc mà còn mở ra cánh cửa ứng dụng thực tiễn, giúp chúng em nhận thức rõ hơn về tầm quan trọng của dữ liệu trong thời đại công nghệ số. Chúng em rất mong nhận được sự góp ý và chỉ dạy từ thầy để bài nghiên cứu được hoàn thiện hơn, đồng thời giúp chúng em tích lũy thêm nhiều bài học quý giá cho hành trang nghề nghiệp sau này.

Chúng em xin chân thành cảm ơn!

Nhận xét của giảng viên

MỤC LỤC

CHƯƠNG 1. GIỚI THIỆU CHỦ ĐỀ ĐỒ ÁN.	6
1.1. Bối cảnh và lý do chọn đề tài.	6
1.2. Mục tiêu nghiên cứu.	6
1.3. Phạm vi nghiên cứu.	7
1.4. Ý nghĩa khoa học và ứng dụng nghiên cứu	7
1.5. Phương pháp nghiên cứu và cấu trúc đồ án.	8
CHƯƠNG 2. CƠ SỞ LÝ THUYẾT.	9
2.1. Cơ sở lý thuyết:	9
2.2. Các thư viện hỗ trợ phân tích và trực quan hóa dữ liệu (Pandas, Matplotlib, Seaborn, Numpy)	12
2.3. Các công cụ hỗ trợ xây dựng mô hình học máy (Sklearn, Joblib)	13
2.4. Tổng kết và đánh giá các nghiên cứu liên quan	14
CHƯƠNG 3: HỆ THỐNG QUẢN LÝ THÔNG TIN BỆNH NHÂN UNG THƯ PHỔI.	15
3.1. Kiến trúc hệ thống	15
3.2. Quy trình thu thập và xử lý dữ liệu	15
3.3. Thiết kế cơ sở dữ liệu và lưu trữ	15
3.4. Giao diện người dùng và chức năng chính	15
3.5. Đánh giá khả năng mở rộng và bảo mật	16
CHƯƠNG 4: Triển khai và lập trình ứng dụng	17
4.1. Tổng quan quá trình	
4.2. Thực hiện xử lý trên Hadoop	17
4.3. Code thực thi hệ thống	17
CHƯƠNG 5: KẾT QUẢ SẢN PHẨM	32
5.1. Kết nối Hadoop trên fit-lab	32
5.2 Giao diện hệ thống quản lý	33
5.3 Dự đoán xu hướng giai đoạn ở độ tuổi	38
CHƯƠNG 6: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	40
1. Kết luận	40
2. Hạn chế của đồ án	40
3. Hướng phát triển hệ thống	41

Phân công công việc

Họ và tên	MSSV	Công việc	Mức độ hoàn thành
Lê Duy Khang	2274802010374	Code chính, xây dựng mô hình, Spark	100%
Trần Hữu Luân	2274802010520	Code chính , thu thập dữ liệu , phân tích dữ liệu, HDFS	100%
Nguyễn Hoàng Giới	2174802010831	word, code phụ	100%
Nguyễn Thanh Trường	207CT47995	word , code phụ , tìm thông tin	100%
Huỳnh Linh Trung	197CT22641	code phụ, xử lý dữ liệu	100%

CHƯƠNG 1. GIỚI THIỆU CHỦ ĐỀ ĐÒ ÁN.

- Khái quát một số nội dung cơ bản liên quan tới Phân tích dữ liệu lớn quản lý và phân tích hồ sơ bệnh nhân mắc ung thư phổi. Ung thư phổi là một trong những bệnh nguy hiểm và phức tạp, đòi hỏi các phương pháp điều trị chính xác, cá nhân hóa và dựa trên dữ liệu. Việc áp dụng dữ liệu lớn giúp tối ưu hóa quy trình lưu trữ, phân tích và sử dụng dữ liệu y tế nhằm nâng cao hiệu quả điều trị và dự đoán bệnh.

1.1. Bối cảnh và lý do chọn đề tài.

- Ung thư là một trong những căn bệnh nguy hiểm đầu tiên, có tỷ lệ tử vong cao và yêu cầu quy trình điều trị phức tạp. Việc quản lý hồ sơ bệnh nhân là một cách hiệu quả đóng vai trò quan trọng trong công việc mong đợi, điều trị và theo dõi tiến trình bệnh. Tuy nhiên, hệ thống lưu trữ lưu trữ thường gặp nhiều chế độ như:

- + **Dữ liệu phân tán, khó truy cập** : Hồ sơ bệnh nhân thường được lưu trữ dưới dạng giấy hoặc hệ thống địa phương, gây khó khăn trong công việc truy xuất thông tin.
- + **Thiếu sự hợp nhất giữa các hệ thống y tế** : Dữ liệu từ bệnh viện, phòng khám và cơ sở nghiên cứu không được liên kết chặt chẽ, dẫn đến khó khăn trong công việc theo dõi và điều phối điều trị.
- + **Chưa tận dụng được công nghệ lớn (Big Data) để hỗ trợ dự đoán và điều trị**: Công cụ xử lý dữ liệu lớn có thể giúp phân tích các mô hình bệnh lý, dự kiến nguy cơ và điều chỉnh sơ đồ tối ưu hóa

- Vì những lý do trên, việc ứng dụng công nghệ lớn (Big Data) để xây dựng một hệ thống quản lý hồ sơ bệnh nhân ung thư sẽ giúp cải thiện chất lượng điều trị, tăng cường khả năng lưu trữ và khai thác hiệu quả dữ liệu hơn.

1.2. Mục tiêu nghiên cứu.

- Hệ thống quản lý hồ sơ bệnh nhân được ứng dụng thư điện tử công nghệ nhằm đạt được các mục tiêu sau:

- + Xây dựng hệ thống lưu trữ tập trung sử dụng Hadoop HDFS để đảm bảo khả năng lưu trữ dữ liệu một cách an toàn và hiệu quả.
- + Tích hợp công nghệ xử lý dữ liệu thông tin như Apache Spark để phân tích dữ liệu bệnh nhân, giúp dự đoán cơ sở dữ liệu và hỗ trợ quyết định sẵn sàng cho lâm sàng.
- + Phát triển giao diện web thân thiện với Flask/Django, giúp bác sĩ và bệnh nhân dễ dàng truy cập thông tin.
- + Tăng cường khả năng bảo mật và quản lý dữ liệu y tế, bổ sung các tiêu chuẩn và toàn thông tin trong lĩnh vực y tế.

1.3. Phạm vi nghiên cứu.

Định hướng:

- Hệ thống được phát triển theo hướng ứng dụng các công nghệ Big Data để tối ưu hóa công việc lưu trữ, truy xuất và xử lý dữ liệu y tế. Các giải pháp được áp dụng bao gồm:

- + **Sử dụng Hadoop HDFS:** để lưu trữ và quản lý dữ liệu bệnh nhân.
- + **Tích hợp Apache Spark:** để xử lý và phân tích dữ liệu lớn.
- + **Phát triển giao diện web dựa trên Flask/Django:** để hỗ trợ truy cập dữ liệu một cách dễ dàng.

Phạm vi:

- **Dữ liệu :** Tập trung vào hồ sơ bệnh nhân ung thư phổi, bao gồm thông tin dự đoán, điều trị, xét nghiệm và lịch sử bệnh án.
- **Đối tượng sử dụng :** Hệ thống hướng dẫn bác sĩ, nhân viên y tế và bệnh nhân có nhu cầu nghiên cứu thông tin bệnh án.
- **Môi trường triển khai :** Hệ thống sẽ được thử nghiệm ở một số bệnh viện và phòng khám trước khi mở rộng mô.

1.4. Ý nghĩa khoa học và ứng dụng nghiên cứu

- Nghiên cứu này có ý nghĩa quan trọng trong việc ứng dụng công nghệ dữ liệu lớn và học máy vào lĩnh vực y tế. Các ứng dụng thực tiễn bao gồm:

- + **Hỗ trợ chẩn đoán:** Hệ thống có thể giúp các bác sĩ phân tích nhanh chóng dữ liệu bệnh nhân và đưa ra dự đoán về ung thư phổi.
- + **Nâng cao hiệu quả điều trị:** Bằng cách nhận diện sớm các yếu tố rủi ro, bệnh nhân có thể được tư vấn và điều trị kịp thời.
- + **Đóng góp vào nghiên cứu y học:** Cung cấp một công cụ hữu ích để phân tích dữ liệu ung thư phổi, giúp các nhà nghiên cứu có thêm thông tin để cải thiện phương pháp điều trị.
- + **Ứng dụng trong hệ thống y tế thông minh:** Góp phần vào sự phát triển của các hệ thống hỗ trợ ra quyết định trong lĩnh vực y tế, tối ưu hóa việc quản lý bệnh nhân.

1.5. Phương pháp nghiên cứu và cấu trúc đồ án.

- Phương pháp nghiên cứu:

- + Thu thập dữ liệu bệnh nhân ung thư phổi từ các nguồn y tế và nghiên cứu trước đó.
- + Tiền xử lý dữ liệu để loại bỏ nhiễu và tối ưu hóa độ chính xác.
- + Sử dụng các thuật toán Machine Learning để xây dựng mô hình dự đoán.
- + Đánh giá hiệu suất mô hình bằng các chỉ số như độ chính xác (Accuracy), độ nhạy (Recall), độ đặc hiệu (Specificity).

- Cấu trúc bài tiểu luận:

- + **Chương 1: Giới thiệu** - Trình bày bối cảnh, mục tiêu, phạm vi nghiên cứu và ý nghĩa thực tiễn của đề tài.
- + **Chương 2: Tổng quan lý thuyết** - Cung cấp kiến thức nền tảng về ung thư phổi, công nghệ dữ liệu lớn và các thuật toán Machine Learning.
- + **Chương 3: Phân tích tổng quan** - Giới thiệu tập dữ liệu, các bước tiền xử lý, và quy trình phát triển hệ thống.
- + **Chương 4: Kết quả thực hiện** - Minh họa các đoạn code thực thi mô hình, kết quả dự đoán và hiệu suất mô hình.
- + **Chương 5: Sản phẩm** - Phân tích dữ liệu thống kê về bệnh nhân ung thư phổi và ứng dụng kết quả vào thực tế.
- + **Chương 6: Kết luận và hướng phát triển** - Tổng kết kết quả chính, nhận xét về tính khả thi, hạn chế của đồ án và đề xuất hướng nghiên cứu trong tương lai.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT.

2.1. Cơ sở lý thuyết:

2.1.1. Tổng quan về bệnh ung thư phổi và tầm quan trọng của việc quản lý thông tin

- Ung thư phổi là một trong những loại ung thư nguy hiểm nhất với tỷ lệ tử vong cao do thường được phát hiện ở giai đoạn muộn. Một số điểm quan trọng về ung thư phổi:

- + **Nguyên nhân:** Hút thuốc lá, ô nhiễm không khí, yếu tố di truyền, tiếp xúc với hóa chất độc hại.
- + **Triệu chứng:** Ho kéo dài, khó thở, đau ngực, ho ra máu.
- + **Phương pháp chẩn đoán:** Chụp X-quang, CT scan, sinh thiết mô.
- + **Điều trị:** Phẫu thuật, xạ trị, hóa trị, liệu pháp nhắm đích.

- Tầm quan trọng của việc quản lý thông tin trong y tế:

- + **Lưu trữ dữ liệu bệnh nhân:** Giúp các bác sĩ theo dõi quá trình điều trị và đưa ra quyết định dựa trên dữ liệu lịch sử.
- + **Tăng độ chính xác trong chẩn đoán:** Ứng dụng Machine Learning giúp phân tích dữ liệu nhanh chóng, phát hiện mẫu bệnh lý chính xác hơn.
- + **Hỗ trợ ra quyết định lâm sàng:** Cung cấp mô hình dự đoán để bác sĩ có thêm thông tin tham khảo khi đưa ra phác đồ điều trị.

2.1.2. Các công nghệ lớn trong quản lý dữ liệu (Big Data, Hadoop, Spark, ...).

Big Data:

- Dữ liệu lớn của công nghệ (**Big Data**) đóng vai trò quan trọng trong lĩnh vực y tế hiện đại, đặc biệt trong công việc quản lý hồ sơ bệnh nhân. Big Data không chỉ

giúp lưu trữ số lượng lớn dữ liệu mà vẫn hỗ trợ phân tích, đưa ra các kỳ vọng và mức độ ưu tiên trong quá trình điều trị.

- Các đặc điểm của Big Data trong y tế:

- + **Dung lượng lớn (Tập)**: Hồ sơ bệnh nhân bao gồm chụp X-quang hình ảnh, chụp CT, kết quả xét nghiệm, đơn thuốc và thuốc điều trị qua nhiều năm.
- + **Xử lý tốc độ cao (Velocity)**: Dữ liệu bệnh nhân cần được cập nhật liên tục để hỗ trợ quyết định nhanh chóng của bác sĩ.
- + **Tính đa dạng (Variety)**: Dữ liệu y tế bao gồm cả dữ liệu có cấu trúc (hồ sơ bệnh án điện tử, kết quả xét nghiệm) và phi cấu trúc (hình ảnh dự đoán, ghi chú của bác sĩ).
- + **Độ tin cậy cao (Veracity)**: Yêu cầu chính xác dữ liệu, minh bạch, bảo mật và có thể truy xuất dễ dàng.

Hadoop:

- + Hệ thống lưu trữ phân tán (HDFS), cho phép quản lý dữ liệu y tế lớn.
- + Sử dụng MapReduce để xử lý dữ liệu nhanh chóng.

Spark:

- + Nhanh hơn Hadoop nhờ xử lý dữ liệu trong bộ nhớ (In-Memory Computing).
- + Hỗ trợ thư viện MLlib cho Machine Learning trên dữ liệu lớn.

2.1.3. Các mô hình nhận dạng và xử lý dữ liệu y tế:

- Trong hệ thống quản lý hồ sơ bệnh nhân được xử lý bằng thư phổi, các mô hình được xử lý và nhận dạng dữ liệu đóng vai trò quan trọng trong công việc tối ưu hóa quá trình quản lý và phân tích dữ liệu.

- **Mô hình dữ liệu:**

- + **Hệ thống quản lý hồ sơ bệnh án điện tử (EMR - Hồ sơ y tế điện tử):** Lưu trữ toàn bộ dữ liệu y khoa của bệnh nhân tại một cơ sở y tế, giúp bác sĩ dễ dàng truy cập và theo dõi tình trạng bệnh.
- + **Hệ thống hồ sơ sức khỏe điện tử (EHR - Electronic Health Record):** Kết nối dữ liệu từ nhiều bệnh viện, phòng khám, giúp quản lý thông tin bệnh nhân trên phạm vi rộng.
- + **Hệ thống quản lý thông tin bệnh viện (HIS - Hospital Information System):** Tích hợp các chức năng quản lý bệnh nhân, tài chính, nhân sự, xét nghiệm, dược phẩm trong một nền tảng duy nhất.

- Xử lý mô hình và phân tích dữ liệu y tế:

- + **Mô hình Machine Learning trong dự đoán bệnh:**
 - **Sử dụng AI và Machine Learning:** để phân tích dữ liệu y tế, phát hiện sớm ung thư từ hình ảnh X-quang, CT scan.
 - **Các thuật toán phỏ biến:** Random Forest, Support Vector Machine (SVM), Deep Learning (CNN - Convolutional Neural Network).
- + **Thực thi thời gian phân tích dữ liệu mô hình:**
 - **Kết hợp Apache Spark Streaming:** để xử lý dữ liệu bệnh nhân theo thời gian.
 - Hỗ trợ phát hiện các dấu hiệu bất thường và đưa ra cảnh báo sớm.

2.1.4. Các tiêu chuẩn và quy định về dữ liệu bảo mật.

- Dữ liệu bệnh nhân là loại dữ liệu nhạy cảm, cần kèm theo các tiêu chuẩn bảo mật nghiêm ngặt để đảm bảo an toàn thông tin. Một số tiêu chuẩn quan trọng bao gồm:

- + **HIPAA (Đạo luật về trách nhiệm giải trình và cung cấp thông tin bảo hiểm y tế - Hoa Kỳ):** Bảo vệ quyền riêng tư của bệnh nhân, quy định về lưu trữ và chia sẻ dữ liệu y tế.
- + **GDPR (Quy định chung về bảo vệ dữ liệu - Châu Âu):** Quy định bảo mật dữ liệu cá nhân, yêu cầu bệnh viện phải có đồng ý của bệnh nhân khi lưu trữ hoặc chia sẻ thông tin.

- + **HL7 (Health Level Seven):** Tiêu chuẩn trao đổi dữ liệu giữa các hệ thống khác nhau.
- + **ISO 27001:** Tiêu chuẩn quốc tế về toàn thông tin, giúp dữ liệu hệ thống bảo mật bệnh viện.

2.1.5. Kết luận.

- Cơ sở lý thuyết về công nghệ dữ liệu lớn, mô hình nhận dạng và bảo mật thông tin là nền tảng quan trọng trong việc xây dựng hệ thống quản lý hồ sơ bệnh nhân thư thư. Việc áp dụng Big Data, AI và các tiêu chuẩn bảo mật giúp nâng cao hiệu quả quản lý, hỗ trợ bác sĩ trong kỳ vọng và điều trị bệnh nhân một cách chính xác và đáp ứng kịp thời.

2.2. Các thư viện hỗ trợ phân tích và trực quan hóa dữ liệu (Pandas, Matplotlib, Seaborn, Numpy)

- Pandas

- + Hỗ trợ thao tác với dữ liệu dạng bảng (DataFrame), giúp làm sạch và tiền xử lý dữ liệu y tế.
- + Ví dụ: Xử lý dữ liệu thiếu trong tập bệnh nhân ung thư phổi.

```
import pandas as pd
data = pd.read_csv("lung_cancer_data.csv")
data.fillna(method="ffill", inplace=True) # Điền giá trị thiếu
```

- Numpy

- + Thư viện toán học hỗ trợ xử lý dữ liệu số học, đặc biệt là mảng dữ liệu lớn

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
mean_value = np.mean(arr) # Tính trung bình
```

- Matplotlib & Seaborn

- + Hỗ trợ trực quan hóa dữ liệu bằng biểu đồ, giúp phân tích xu hướng bệnh tật.

```

import matplotlib.pyplot as plt
import seaborn as sns

sns.histplot(data["age"], bins=30, kde=True)
plt.title("Phân bố độ tuổi bệnh nhân ung thư phổi")
plt.show()

```

2.3. Các công cụ hỗ trợ xây dựng mô hình học máy (Scikitlearn, Joblib)

- Scikit-learn (Scikitlearn)

- + Cung cấp các thuật toán Machine Learning như Decision Tree, Random Forest, SVM.
- + Hỗ trợ đánh giá mô hình bằng các chỉ số như Accuracy, Precision, Recall.

```

4   from sklearn.model_selection import train_test_split, GridSearchCV
5   from sklearn.ensemble import RandomForestClassifier
6   from sklearn.metrics import accuracy_score, classification_report

21  # Chia dữ liệu thành tập huấn luyện (80%) và kiểm tra (20%)
22  X_train_orig, X_test_orig, y_train_orig, y_test_orig = train_test_split(
23      X, y, test_size=0.2, random_state=42, stratify=y
24

54  y_pred = model.predict(X_test_selected)
55  accuracy = accuracy_score(y_test_orig, y_pred)
56  print(f"Độ chính xác của mô hình trên tập kiểm tra: {accuracy:.2f}")

```

- Joblib

- + Hỗ trợ lưu trữ và tải lại mô hình Machine Learning để sử dụng trong hệ thống thực tế.

```

3   import joblib

```

```
67     # Lưu mô hình  
68     joblib.dump(model, model_path)  
69     print(f"Mô hình đã được lưu tại {model_path}")
```

2.4. Tổng kết và đánh giá các nghiên cứu liên quan

- Các nghiên cứu trước đây chủ yếu sử dụng Logistic Regression và SVM để dự đoán ung thư phổi.
- Một số nghiên cứu mới ứng dụng Deep Learning để cải thiện độ chính xác.
- Đò án hiện tại kết hợp nhiều thuật toán Machine Learning và Big Data để tối ưu hóa dự đoán.

CHƯƠNG 3: HỆ THỐNG QUẢN LÝ THÔNG TIN BỆNH NHÂN UNG THƯ PHỔI.

3.1. Kiến trúc hệ thống

Hệ thống quản lý thông tin bệnh nhân ung thư phổi được thiết kế dựa trên mô hình kiến trúc tập trung, kết hợp các công nghệ lớn như Hadoop và Spark để xử lý dữ liệu quy mô lớn. Hệ thống bao gồm các thành phần chính:

- **Lớp thu thập dữ liệu:** Nhận dữ liệu từ các bệnh viện, phòng khám, thiết bị IoT.
- **Lớp lưu trữ dữ liệu:** Dựa trên Hadoop Distributed File System (HDFS) để đảm bảo tính phân tán và dễ mở rộng.
- **Lớp xử lý dữ liệu:** Sử dụng Apache Spark để phân tích và xử lý nhanh chóng.
- **Lớp truy vấn và giao diện người dùng:** Hỗ trợ truy vấn nhanh chóng và giao diện web/thanh điều hành trực quan.

3.2. Quy trình thu thập và xử lý dữ liệu

Quy trình thu thập và xử lý dữ liệu bao gồm:

1. **Thu thập dữ liệu:** Từ hồ sơ bệnh án, hệ thống PACS, và thiết bị y tế.
2. **Tiền xử lý:** Lọc trùng lặp, chuẩn hóa dữ liệu.
3. **Lưu trữ:** Lưu trữ trên HDFS.
4. **Xử lý và phân tích:** Dùng Spark MLlib để phân tích dữ liệu và hỗ trợ quyết định.

3.3. Thiết kế cơ sở dữ liệu và lưu trữ

- **HDFS:** Lưu trữ dữ liệu lớn, bao gồm các file DICOM, hồ sơ bệnh án.
- **Apache Hive:** Cung cấp khả năng truy vấn SQL trên dữ liệu lớn.
- **HBase:** Lưu trữ dữ liệu phi cấu trúc, chẳng hạn như dữ liệu IoT.

3.4. Giao diện người dùng và chức năng chính

Giao diện người cần được thiết kế thân thiện, dễ sử dụng và trực quan để hỗ trợ các bác sĩ, nhân viên y tế và bệnh nhân có thể dễ dàng thao tác. Hệ thống quản lý hồ sơ bệnh nhân ung thư là một giao diện trực tiếp, dễ sử dụng để hỗ trợ bác sĩ, nhân viên y tế và bệnh nhân trong công việc quản lý thông tin, theo dõi điều trị và đưa ra quyết định chính xác

hơn về lâm sàng. Ngoài ra, hệ thống cần tích hợp công nghệ AI và Big Data để hỗ trợ phân tích dữ liệu, dự đoán diễn biến bệnh và đưa ra phương pháp điều trị hiệu quả.

Với giao diện dễ hiểu, không quá nhiều thao tác phức tạp, chia giao diện thành các phần cụ thể cho từng nhóm chức năng, hoạt động tốt trên máy tính, máy tính bảng và điện thoại, hiển thị thông tin phù hợp với quyền của từng người dùng.

3.5. Đánh giá khả năng mở rộng và bảo mật

Hệ thống quản lý hồ sơ bệnh nhân ung thư phổi cần có khả năng mở rộng linh hoạt để đáp ứng nhu cầu ngày càng tăng của bệnh viện và các cơ sở y tế. Một số yếu tố quan trọng khi đánh giá khả năng mở rộng, với dung lượng lưu trữ lớn, mở rộng về số lượng người dùng càng ngày càng tăng.

Hồ sơ bệnh nhân là dữ liệu nhạy cảm và cần được bảo vệ chặt chẽ để tránh rủi ro rò rỉ hoặc bị truy cập trái phép. Mã hóa dữ liệu, hạn chế quyền truy cập vào cơ sở dữ liệu chỉ dành cho những người có thẩm quyền, giới hạn số lần nhập sai mật khẩu.

CHƯƠNG 4: Triển khai và lập trình ứng dụng

4.1. Tổng quan quá trình

Các giai đoạn triển khai xây dựng hệ thống:

- + Xác nhận và phân tích yêu cầu hệ thống quản lý.
- + Thu thập dữ liệu hồ sơ bệnh nhân ung thư phổi trên Kaggle.
- + Sử dụng numpy, pandas, matplotlib, seaborn, StandardScaler của sklearn để xử lý, phân tích và trực quan hóa dữ liệu.
- + Sử dụng thuật toán RandomForest để xây dựng và huấn luyện mô hình.
- + Sử dụng HDFS để lưu trữ dữ liệu.
- + Sử dụng Spark để thống kê và dự đoán xu hướng giai đoạn bệnh ở độ tuổi.
- + Tích hợp hệ thống và xây dựng web.

4.2. Thực hiện xử lý trên hadoop

Trong hệ thống quản lý hồ sơ bệnh nhân ung thư phổi, Hadoop được sử dụng để xử lý và lưu trữ dữ liệu lớn hiệu quả. Hai thành phần chính của Hadoop được ứng dụng là: HDFS (hệ thống tệp phân tán dùng để lưu trữ dữ liệu đầu vào) và Apache Spark (công cụ xử lý dữ liệu mạnh mẽ trên nền tảng Hadoop, hỗ trợ tính toán phân tán.)

4.2.1. Tải dữ liệu lên hadoop

Dữ liệu bệnh nhân được thu thập từ Kaggle dưới dạng tệp CSV. Dữ liệu này được tải lên hệ thống HDFS để xử lý phân tán.

```
# Tạo thư mục trên HDFS
```

```
hdfs dfs -mkdir /user/do_an/hiv_data
```

```
# Upload tệp CSV lên HDFS
```

```
hdfs dfs -put lung_cancer_data.csv /user/do_an/hiv_data/
```

4.2.2. Khởi chạy xử lý với spark

Apache Spark được sử dụng để xử lý dữ liệu đã lưu trên HDFS. File xử lý được viết bằng Python (PySpark) và nộp thông qua lệnh spark-submit.

```
spark-submit --master yarn \
```

```
--deploy-mode client \
```

```
--name LungCancerProcessing \
```

lung_cancer_analysis.py

4.3. Code thực thi hệ thống

Cấu trúc thư mục đồ án:

```
└── DOAN_DLL
    ├── .venv
    ├── data
    │   ├── lung_cancer_data_cleaned.csv
    │   ├── lung_cancer_data.csv
    │   └── prediction_history.csv
    ├── data_test
    ├── hadoop
    │   └── run_wordcount.sh
    ├── Spark.py
    ├── m1
    │   ├── lung_cancer_model.pkl
    │   ├── predict.py
    │   └── train_model.py
    ├── scripts
    │   ├── data_analysis.py
    │   └── data_processing.py
    └── web
        ├── static
        ├── templates
        │   ├── add_patient.html
        │   ├── index.html
        │   └── visualize.html
        └── app.py
```

4.3.1. Thu thập dữ liệu

The screenshot shows the Kaggle interface for the 'Lung Cancer Prediction' competition. At the top, there are buttons for 'Data Card', 'Code (10)', 'Discussion (4)', and 'Suggestions (1)'. A '48' button is also present. On the right, there are 'Download' and more options buttons. Below the header, a 'Cancer' tag is visible. The main area displays the 'lung_cancer_data.csv' file, which is 10.07 MB in size. It includes a preview section with 'Detail', 'Compact', and 'Column' buttons, and a status bar indicating '10 of 38 columns'. To the right, a 'Data Explorer' panel shows the same file with its size listed as 10.07 MB.

Hình 1: Thu thập dữ liệu trên Kaggle

4.3.2. Xử lý và phân tích dữ liệu

```
import pandas as pd
from sklearn.preprocessing import StandardScaler

def clean_data(input_file, output_file):
    # Đọc dữ liệu từ CSV
    df = pd.read_csv(input_file)

    # Xóa dòng có giá trị NULL
    df.dropna(inplace=True)

    # Chuẩn hóa tên cột
    df.columns = df.columns.str.strip().str.lower()

    # Đảm bảo cột age là numeric và giữ nguyên giá trị
    if 'age' in df.columns:
        age_values = df['age'].copy() # Lưu giữ giá trị gốc của age
        df['age'] = pd.to_numeric(df['age'], errors='coerce')

    # Loại bỏ cột patient_id nếu tồn tại
    df.drop(columns=[col for col in ['patient_id'] if col in df.columns], inplace=True)

    # Định dạng đúng các cột số (trừ age)
    numeric_columns = [col for col in df.columns if df[col].dtype in ['int64', 'float64'] and col != 'age']
    df[numeric_columns] = df[numeric_columns].apply(pd.to_numeric, errors='coerce')

    # Chuẩn hóa các cột số (trừ age)
    if numeric_columns: # Chỉ chuẩn hóa nếu có cột số nào khác age
        scaler = StandardScaler()
        df[numeric_columns] = scaler.fit_transform(df[numeric_columns])

    # Xử lý các cột phân loại
    categorical_mappings = {
        'gender': {'Female': 0, 'Male': 1},
        'smoking_history': {'Never Smoked': 0, 'Former Smoker': 1, 'Current Smoker': 2},
        'stage': {'Stage I': 0, 'Stage II': 1, 'Stage III': 2, 'Stage IV': 3}
    }

    for col, mapping in categorical_mappings.items():
        if col in df.columns:
            df[col] = df[col].map(mapping)

    categorical_columns = ['tumor_location', 'treatment', 'ethnicity', 'insurance_type', 'family_history']
    for col in categorical_columns:
        if col in df.columns:
            df[col], _ = pd.factorize(df[col])

    # Xử lý cột comorbidity
    comorbidity_columns = [col for col in df.columns if col.startswith('comorbidity_')]
    for col in comorbidity_columns:
        if col in df.columns:
            df[col] = df[col].map({'No': 0, 'Yes': 1})
```

```

# Xóa các dòng có giá trị NULL sau khi xử lý
df.dropna(inplace=True)

# Khôi phục giá trị gốc của age nếu cần
if 'age' in df.columns and 'age_values' in locals():
    df['age'] = age_values[df.index] # Gán lại giá trị gốc cho age

# Kiểm tra phân bố nhãn 'stage'
if 'stage' in df.columns:
    print("Phân bố nhãn 'stage' trong dữ liệu gốc:")
    print(df['stage'].value_counts(normalize=True) * 100)

# Kiểm tra age có bị thay đổi không
if 'age' in df.columns:
    print("\nKiểm tra giá trị age:")
    print("Giá trị age đầu tiên trong dữ liệu gốc:", age_values.iloc[0])
    print("Giá trị age đầu tiên sau xử lý:", df['age'].iloc[0])

# Lưu dữ liệu đã làm sạch
df.to_csv(output_file, index=False)
print(f"\nDữ liệu đã được làm sạch và lưu vào {output_file}")
print("Cột age đã được giữ nguyên không thay đổi")

return df

```

```

if __name__ == "__main__":
    input_file = "data/lung_cancer_data.csv"
    output_file = "data/lung_cancer_data_cleaned.csv"
    df_cleaned = clean_data(input_file, output_file)

```

```
Phân bố nhãn 'stage' trong dữ liệu gốc:  
stage  
3    25.407896  
2    25.031702  
0    24.959844  
1    24.600558  
Name: proportion, dtype: float64  
  
Kiểm tra giá trị age:  
Giá trị age đầu tiên trong dữ liệu gốc: 68  
Giá trị age đầu tiên sau xử lý: 68  
  
Dữ liệu đã được làm sạch và lưu vào data/lung_cancer_data_cleaned.csv  
Cột age đã được giữ nguyên không thay đổi
```

Hình 2: Xử lý và phân tích dữ liệu

4.3.3. Trực quan hóa dữ liệu

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

def analyze_data(df):
    if 'age' in df.columns:
        print("Thống kê cơ bản về tuổi:")
        print(df['age'].describe())

    if 'gender' in df.columns:
        print("\nTỷ lệ giới tính (0: Female, 1: Male):")
        print(df['gender'].value_counts(normalize=True) * 100)

    if 'smoking_history' in df.columns:
        print("\nTỷ lệ tiền sử hút thuốc (0: Never, 1: Former, 2: Current):")
        print(df['smoking_history'].value_counts(normalize=True) * 100)

    if 'tumor_size_mm' in df.columns:
        print("\nKích thước khối u trung bình:")
        print(df['tumor_size_mm'].mean())

    if 'stage' in df.columns:
        print("\nPhân bố giai đoạn ung thư:")
        print(df['stage'].value_counts(normalize=True) * 100)

    if 'treatment' in df.columns:
        print("\nPhương pháp điều trị phổ biến:")
        print(df['treatment'].value_counts(normalize=True) * 100)

    if 'survival_months' in df.columns:
        print("\nSố tháng sống sót trung bình:")
        print(df['survival_months'].mean())
```

```

def visualize_data(df):
    if 'stage' in df.columns:
        plt.figure(figsize=(8,5))
        sns.countplot(x=df['stage'], palette='coolwarm')
        plt.title("Phân bố giai đoạn ung thư phổi")
        plt.xlabel("Giai đoạn")
        plt.ylabel("Số lượng bệnh nhân")
        plt.show()

    if 'age' in df.columns:
        plt.figure(figsize=(8,5))
        sns.histplot(df['age'], bins=20, kde=True, color='blue')
        plt.title("Phân bố tuổi của bệnh nhân")
        plt.xlabel("Tuổi")
        plt.ylabel("Số lượng bệnh nhân")
        plt.show()

    if {'tumor_size_mm', 'survival_months', 'stage'}.issubset(df.columns):
        plt.figure(figsize=(8,5))
        sample_df = df.sample(frac=0.1, random_state=42)
        sns.scatterplot(x=sample_df['tumor_size_mm'], y=sample_df['survival_months'],
                        hue=sample_df['stage'], palette='coolwarm')
        plt.title("Tương quan giữa kích thước khối u và số tháng sống sót")
        plt.xlabel("Kích thước khối u")
        plt.ylabel("Số tháng sống sót")
        plt.show()

    if 'treatment' in df.columns:
        plt.figure(figsize=(8,5))
        sns.countplot(x=df['treatment'], palette='coolwarm')
        plt.title("Phân bố phương pháp điều trị")
        plt.xlabel("Phương pháp điều trị")
        plt.ylabel("Số lượng bệnh nhân")
        plt.show()

if __name__ == "__main__":
    input_file = "data/lung_cancer_data_cleaned.csv"
    df = pd.read_csv(input_file)
    analyze_data(df)
    visualize_data(df)

```

Hình 3: Trực quan hóa dữ liệu đã xử lý

4.3.4. Xây dựng mô hình huấn luyện

```
import pandas as pd
import numpy as np
import joblib
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
from imblearn.over_sampling import SMOTE

def train_model(data_path, model_path):
    # Đọc dữ liệu từ CSV
    df = pd.read_csv(data_path)

    # Giữ nguyên toàn bộ dữ liệu thay vì lấy mẫu 50%
    X = df.drop(columns=['stage'])
    y = df['stage']

    # Kiểm tra phân bố nhãn 'stage' trong dữ liệu
    print("Phân bố nhãn 'stage' trong dữ liệu:")
    print(df['stage'].value_counts(normalize=True) * 100)

    # Chia dữ liệu thành tập huấn luyện (80%) và kiểm tra (20%)
    X_train_orig, X_test_orig, y_train_orig, y_test_orig = train_test_split(
        X, y, test_size=0.2, random_state=42, stratify=y
    )

    # Xử lý mất cân bằng bằng lớp bằng SMOTE
    smote = SMOTE(random_state=42, k_neighbors=3) # Giảm k_neighbors để phù hợp hơn
    X_train_balanced, y_train_balanced = smote.fit_resample(X_train_orig, y_train_orig)

    # Kiểm tra phân bố nhãn sau SMOTE
    print("\nPhân bố nhãn 'stage' sau SMOTE (tập huấn luyện):")
    print(pd.Series(y_train_balanced).value_counts(normalize=True) * 100)

    # Giữ nguyên các đặc trưng đầu vào
    X_train_selected = X_train_balanced
    X_test_selected = X_test_orig
```

```

# Tìm tham số tối ưu bằng GridSearchCV với không gian tham số hợp lý hơn
param_grid = {
    'n_estimators': [100, 200], # Thử nghiệm số cây khác nhau
    'max_depth': [10, 20],
    'min_samples_split': [2, 5],
    'min_samples_leaf': [1, 2]
}
rf = RandomForestClassifier(class_weight='balanced', random_state=42)
grid_search = GridSearchCV(rf, param_grid, cv=3, scoring='accuracy', n_jobs=-1)
grid_search.fit(X_train_selected, y_train_balanced)

# Lấy mô hình tốt nhất
model = grid_search.best_estimator_
print(f"\nTham số tốt nhất: {grid_search.best_params_}")

# Đánh giá mô hình trên tập kiểm tra
y_pred = model.predict(X_test_selected)
accuracy = accuracy_score(y_test_orig, y_pred)
print(f"Độ chính xác của mô hình trên tập kiểm tra: {accuracy:.2f}")

# In báo cáo phân loại chi tiết
print("\nBáo cáo phân loại chi tiết trên tập kiểm tra:")
print(classification_report(y_test_orig, y_pred, target_names=['Stage I', 'Stage II', 'Stage III', 'Stage IV']))

# Đánh giá trên toàn bộ tập dữ liệu gốc thay vì chỉ lấy mẫu 20%
y_pred_orig = model.predict(X)
accuracy_orig = accuracy_score(y, y_pred_orig)
print(f"Độ chính xác trên toàn bộ tập dữ liệu: {accuracy_orig * 100:.2f}%")

# Lưu mô hình
joblib.dump(model, model_path)
print(f"Mô hình đã được lưu tại {model_path}")

return model

if __name__ == "__main__":
    data_file = "data/lung_cancer_data_cleaned.csv" # Cập nhật đường dẫn file dữ liệu
    model_file = "ml/lung_cancer_model.pkl"
    train_model(data_file, model_file)

```

```
Phân bố nhãn 'stage' trong dữ liệu:  
stage  
3    25.407896  
2    25.031702  
0    24.959844  
1    24.600558  
Name: proportion, dtype: float64
```

```
Phân bố nhãn 'stage' sau SMOTE (tập huấn luyện):  
stage  
1    25.0  
3    25.0  
0    25.0  
2    25.0  
Name: proportion, dtype: float64
```

```
Tham số tốt nhất: {'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 200}  
Độ chính xác của mô hình trên tập kiểm tra: 0.24
```

```
Báo cáo phân loại chi tiết trên tập kiểm tra:  
precision    recall    f1-score   support  
  
Stage I       0.24      0.24      0.24      1181  
Stage II      0.23      0.24      0.24      1164  
Stage III     0.25      0.26      0.26      1185  
Stage IV      0.23      0.22      0.22      1202  
  
accuracy          0.24      0.24      0.24      4732  
macro avg       0.24      0.24      0.24      4732  
weighted avg    0.24      0.24      0.24      4732
```

```
Độ chính xác trên toàn bộ tập dữ liệu: 84.78%  
Mô hình đã được lưu tại ml/lung_cancer_model.pkl
```

Hình 4: Dùng RandomForest để huấn luyện

4.3.5. Kiểm tra mô hình huấn luyện

```
import pandas as pd
import joblib
import numpy as np

def predict_stage(input_data):
    # Load model
    model = joblib.load("ml/lung_cancer_model.pkl")

    # Lấy danh sách đặc trưng từ mô hình đã huấn luyện
    features = model.feature_names_in_

    # Đảm bảo input_data có đầy đủ các cột, nếu thiếu thì gán giá trị mặc định là 0
    for feature in features:
        if feature not in input_data:
            input_data[feature] = 0 # Giá định bệnh nền mặc định là không có (0)

    # Chuyển dữ liệu đầu vào thành DataFrame, đảm bảo cột đúng thứ tự
    input_df = pd.DataFrame([input_data])[list(features)]

    # Đảm bảo kiểu dữ liệu nhất quán với khi huấn luyện
    input_df = input_df.astype(np.float64)

    # Dự đoán giai đoạn ung thư
    prediction = model.predict(input_df)
    return prediction[0]
```

```
# Ví dụ sử dụng
data_example = {
    'age': 65, 'tumor_size_mm': 35, 'survival_months': 12, 'performance_status': 1,
    'blood_pressure_systolic': 120, 'blood_pressure_diastolic': 80, 'blood_pressure_pulse': 40,
    'hemoglobin_level': 13.5, 'white_blood_cell_count': 6.2, 'platelet_count': 250,
    'albumin_level': 3.8, 'alkaline_phosphatase_level': 90, 'alanine_aminotransferase_level': 25,
    'aspartate_aminotransferase_level': 30, 'creatinine_level': 1.1, 'ldh_level': 180,
    'calcium_level': 9.5, 'phosphorus_level': 3.5, 'glucose_level': 100, 'potassium_level': 4.2,
    'sodium_level': 140, 'smoking_pack_years': 20, 'gender': 1, 'smoking_history': 1,
    'tumor_location': 2, 'treatment': 1, 'ethnicity': 0, 'insurance_type': 1, 'family_history': 0,
    'comorbidity_autoimmune_disease': 0, 'comorbidity_chronic_lung_disease': 0,
    'comorbidity_diabetes': 0, 'comorbidity_heart_disease': 0, 'comorbidity_hypertension': 0,
    'comorbidity_kidney_disease': 0, 'comorbidity_other': 0
}

predicted_stage = predict_stage(data_example)
print(f'Giai đoạn ung thư dự đoán: {predicted_stage}')
```

Giai đoạn ung thư dự đoán: 3

Hình 5: Kiểm tra mô hình

4.3.6. Spark

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, count, row_number
from pyspark.sql.window import Window

# Tạo SparkSession
spark = SparkSession.builder.appName("LungCancerAnalysis").getOrCreate()

# Đọc dữ liệu từ file
df = spark.read.option("header", True).csv("hdfs://localhost:9870/user/hadoop/data/lung_cancer_data_cleaned.csv")

# Bước 1: CẬP NHẬT CHỖ NÀY - Nhóm theo age, stage và 2 cột quan trọng (gender, smoking_history)
df_grouped = df.groupBy("age", "stage", "gender", "smoking_history") \
| | | | .agg(count("*").alias("count"))

# Bước 2: Tìm tổ hợp phổ biến nhất theo từng độ tuổi
window_spec = Window.partitionBy("age").orderBy(col("count").desc())
df_ranked = df_grouped.withColumn("rank", row_number().over(window_spec))

# Lọc ra tổ hợp phổ biến nhất
df_top_per_age = df_ranked.filter(col("rank") == 1).drop("rank")

# Hiển thị kết quả
df_top_per_age.select("age", "gender", "smoking_history", "stage", "count").show()

# Lưu kết quả
df_top_per_age.write.mode("overwrite").csv("./data/lung_cancer_hadoop_data", header=True)

# Dừng Spark
spark.stop()
```

Hình 6: Sử dụng Spark để thống kê và dự đoán

4.3.6. Xây dựng web

```
from flask import Flask, render_template, request, redirect, url_for
import pandas as pd
import joblib
import seaborn as sns
import matplotlib.pyplot as plt
import os
from datetime import datetime
```

```

app = Flask(__name__)

# Load data
data_file = "data/lung_cancer_data.csv"
data = pd.read_csv(data_file)

# Load model
model = joblib.load("ml/lung_cancer_model.pkl")

# File lưu lịch sử dự đoán
prediction_history_file = "data/prediction_history.csv"

@app.route('/')
def index():
    # Đọc lịch sử dự đoán nếu file tồn tại
    prediction_history = pd.DataFrame()
    if os.path.exists(prediction_history_file):
        prediction_history = pd.read_csv(prediction_history_file)

    return render_template('index.html',
                           tables=[data.to_html(classes='data', index=False)],
                           titles=data.columns.values,
                           prediction_history=prediction_history.to_html(classes='data', index=False) if not prediction_history.empty else None,
                           prediction_table=None, # Đặt mặc định là None khi vào trang chính
                           patient_data=None,
                           prediction_result=None,
                           prediction_time=None,
                           confidence=None)

```

```

@app.route('/add_patient', methods=['GET', 'POST'])
def add_patient():
    if request.method == 'POST':
        new_patient = {col: request.form.get(col, 0) for col in data.columns}

        # Chuyển đổi kiểu dữ liệu phù hợp
        for key in new_patient.keys():
            try:
                new_patient[key] = float(new_patient[key])
            except ValueError:
                pass

        new_patient_df = pd.DataFrame([new_patient])
        new_patient_df.to_csv(data_file, mode='a', header=False, index=False)
        return redirect(url_for('index'))

    return render_template('add_patient.html')

```

```

@app.route('/plot')
def plot():
    plt.figure(figsize=(8, 6))
    sns.histplot(data['age'], bins=20, kde=True)
    plt.xlabel('Age')
    plt.ylabel('Frequency')
    plt.title('Age Distribution')

    # Tạo thư mục static nếu chưa tồn tại
    if not os.path.exists("static"):
        os.makedirs("static")

    plt.savefig("static/age_distribution.png")
    return redirect(url_for('index'))

```

```

@app.route('/visualize')
def visualize():
    images = [
        "age_distribution.png",
        "treatment_distribution.png",
        "tumor_size_survival.png",
        "stage_distribution.png"
    ]
    # Chỉ hiển thị các hình ảnh tồn tại
    existing_images = [img for img in images if os.path.exists(os.path.join("static", img))]

    return render_template('visualize.html', images=existing_images)

```

```

@app.route('/predict_file', methods=['POST'])
def predict_file():
    if 'file' not in request.files:
        return redirect(url_for('index'))

    file = request.files['file']

    if file.filename == '':
        return redirect(url_for('index'))

    # Đọc file
    ext = file.filename.split('.')[ -1]
    if ext == 'csv':
        df = pd.read_csv(file)
    elif ext in ['xls', 'xlsx']:
        df = pd.read_excel(file)
    else:
        return redirect(url_for('index'))

    # Kiểm tra cột đầu vào
    required_cols = ['age', 'gender', 'smoking_history', 'tumor_size_mm', 'tumor_location', 'treatment',
                     'survival_months', 'ethnicity', 'insurance_type', 'family_history', 'comorbidity_diabetes',
                     'comorbidity_hypertension', 'comorbidity_heart_disease', 'comorbidity_chronic_lung_disease',
                     'comorbidity_kidney_disease', 'comorbidity_autoimmune_disease', 'comorbidity_other',
                     'performance_status', 'blood_pressure_systolic', 'blood_pressure_diastolic',
                     'blood_pressure_pulse', 'hemoglobin_level', 'white_blood_cell_count', 'platelet_count',
                     'albumin_level', 'alkaline_phosphatase_level', 'alanine_aminotransferase_level',
                     'aspartate_aminotransferase_level', 'creatinine_level', 'ldh_level', 'calcium_level',
                     'phosphorus_level', 'glucose_level', 'potassium_level', 'sodium_level', 'smoking_pack_years']

    if not all(col in df.columns for col in required_cols):
        return "File không đúng định dạng!", 400

    # Tiền xử lý dữ liệu
    df.fillna(0, inplace=True)

    # Kiểm tra model trước khi dự đoán
    if model is None:
        return "Mô hình chưa được tải!", 500

    # Dự đoán
    predictions = model.predict(df)

    # Lấy dữ liệu của 1 bệnh nhân trong file
    patient_data = df.iloc[0]
    prediction_result = predictions[0]
    prediction_time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    confidence = 95 # Giả lập độ tin cậy cho dự đoán

    # Thêm kết quả dự đoán vào DataFrame lịch sử
    df['predicted_stage'] = predictions
    df['prediction_time'] = prediction_time

```

```
# Lưu kết quả dự đoán vào lịch sử
if not os.path.exists(prediction_history_file):
    df.to_csv(prediction_history_file, index=False)
else:
    df.to_csv(prediction_history_file, mode='a', header=False, index=False)

# Đọc lại lịch sử dự đoán
prediction_history = pd.read_csv(prediction_history_file)

# Render lại trang dự đoán ở Tab 2 mà không chuyển sang Tab 1
return render_template('index.html',
    tables=[df.to_html(classes='data', index=False)],
    titles=df.columns.values,
    prediction_table=df.to_html(classes='data', index=False),
    patient_data=patient_data,
    prediction_result=prediction_result,
    prediction_time=prediction_time,
    confidence=confidence,
    prediction_history=prediction_history.to_html(classes='data', index=False),
    active_tab=1) # Thêm biến active_tab để giữ tab 2

if __name__ == '__main__':
    app.run(debug=True, port=5004)
```

Hình 7: Xây dựng web bằng thư viện Flask

CHƯƠNG 5: KẾT QUẢ SẢN PHẨM

5.1. Kết nối Hadoop trên fit-lab

The screenshot shows a web-based Hadoop file browser interface. At the top, there is a navigation bar with links: Hadoop, Overview, Datanodes, Datanode Volume Failures, Snapshot, Startup Progress, Utilities, and a dropdown menu. Below the navigation bar, the title "Browse Directory" is displayed. A search bar with the path "/user/hadoop/data" and a "Go!" button is present. To the right of the search bar are several icons for file operations: copy, move, delete, and others. Below the search bar, there are buttons to "Show 25 entries" and a "Search" input field. The main area is a table listing three files:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	hadoop	supergroup	9.61 MB	Apr 02 14:06	1	128 MB	lung_cancer_data.csv
-rw-r--r--	hadoop	supergroup	10.02 MB	Apr 02 14:06	1	128 MB	lung_cancer_data_cleaned.csv
-rw-r--r--	hadoop	supergroup	1.57 KB	Apr 02 14:06	1	128 MB	prediction_history.csv

At the bottom left, it says "Showing 1 to 3 of 3 entries". On the right, there are "Previous" and "Next" buttons, with the number "1" highlighted in blue. The footer of the page displays the text "Hadoop, 2024."

5.2 Giao diện hệ thống quản lý

Hệ thống Quản lý Bệnh nhân & Dự Đoán Giai Đoạn Bệnh

[Danh sách Bệnh nhân](#)[Dự đoán từ File](#)

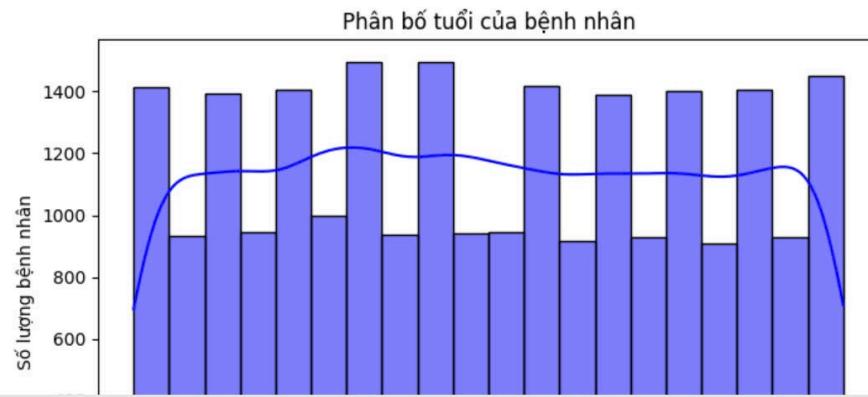
Danh sách Bệnh nhân

Patient_ID	Age	Gender	Smoking_History	Tumor_Size_mm	Tumor_Location	Stage	Treatment	Surv
Patient0000	68	Male	Current Smoker	81.678677	Lower Lobe	Stage III	Surgery	44
Patient0001	58	Male	Never Smoked	78.448272	Lower Lobe	Stage I	Radiation Therapy	101
Patient0002	44	Male	Former Smoker	67.714305	Lower Lobe	Stage I	Chemotherapy	69
Patient0003	72	Male	Current Smoker	70.806008	Lower Lobe	Stage III	Chemotherapy	95
Patient0004	37	Female	Never Smoked	87.272433	Lower Lobe	Stage IV	Radiation Therapy	105
Patient0005	50	Male	Never Smoked	72.148656	Lower Lobe	Stage I	Surgery	49
Patient0006	68	Female	Current Smoker	19.122175	Middle Lobe	Stage I	Radiation Therapy	63
Patient0007	48	Male	Current Smoker	68.095057	Lower Lobe	Stage IV	Chemotherapy	101
Patient0008	52	Female	Former Smoker	25.299440	Lower Lobe	Stage I	Targeted Therapy	35

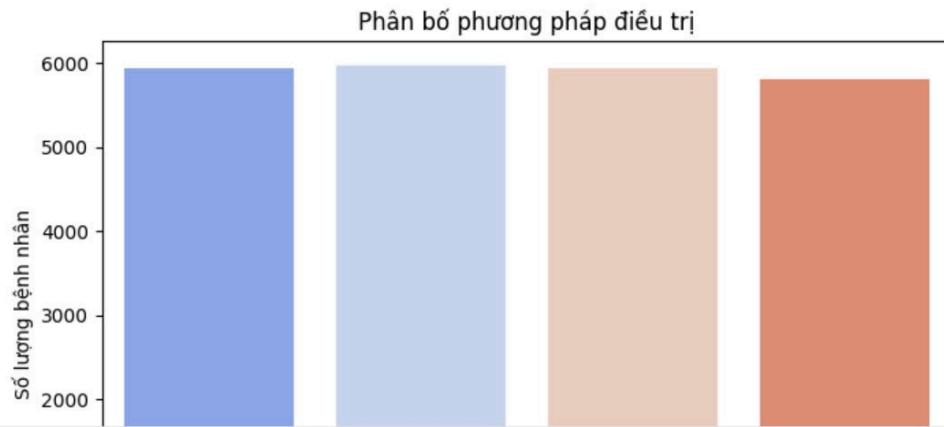
[Xem biểu đồ phân bố](#)

Thống kê Bệnh nhân

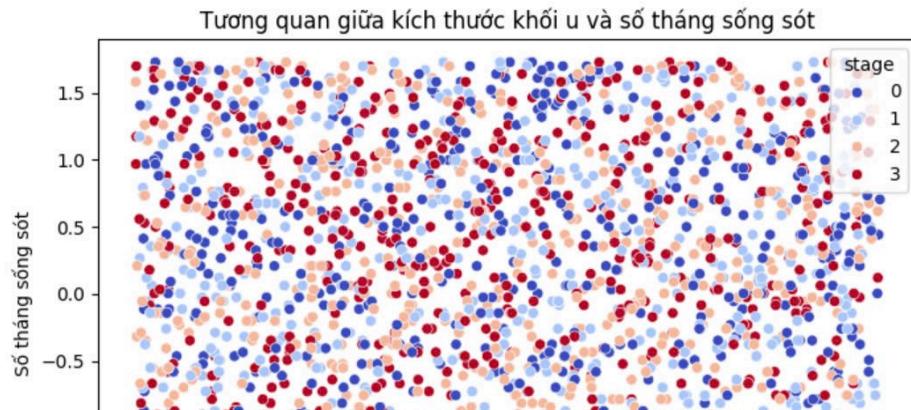
Phân bố tuổi của bệnh nhân



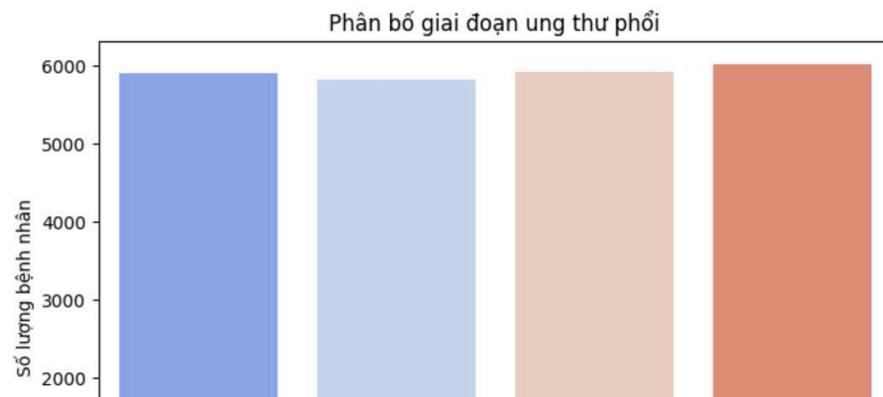
Phân bố phương pháp điều trị



Tương quan giữa kích thước khối u và số tháng sống sót



Phân bố giai đoạn ung thư phổi



[Quay lại danh sách](#)

Hệ thống Quản lý Bệnh nhân & Dự Đoán Giai Đoạn Bệnh

Danh sách Bệnh nhân

Dự đoán từ File

Tải lên file CSV/XLSX để Dự đoán

Chọn tệp data_test4.csv

Dự đoán

Hệ thống Quản lý Bệnh nhân & Dự Đoán Giai Đoạn Bệnh

Danh sách Bệnh nhân

Dự đoán từ File

Tải lên file CSV/XLSX để Dự đoán

Chọn tệp chưa chọn tệp nào

Dự đoán

Thông tin Bệnh nhân

Tuổi: 31.0

Giới tính: Nam

Hút thuốc: -0.6443752815136696 pack-years

Kích thước khối u: 0.8442794382994205 mm

Vị trí khối u: Thủy dưới

Kết quả Dự đoán

Giai đoạn: I

Ngày dự đoán: 2025-04-01 19:04:07

Độ tin cậy: 95%

Lịch sử Dự đoán

age	gender	smoking_history	tumor_size_mm	tumor_location	treatment	survival_mos
31	1	0	0.844279	0	1	1.201221

5.3 Dự đoán xu hướng giai đoạn ở độ tuổi

```
hadoop@hadoop-2274802010374:~/iDragonCloud$ spark-submit --version  
Welcome to
```

version 3.5.5

Using Scala version 2.12.18, OpenJDK 64-Bit Server VM, 11.0.26

```
hadoop@hadoop-2274802010374:~/iDragonCloud$ pyspark
Python 3.10.12 (main, Feb  4 2025, 14:57:36) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Welcome to
```

Using Python version 3.10.12 (main, Feb 4 2025 14:57:36)

Hình 8: kiểm tra spark

```
hadoop@hadoop-2274802010374:~/iDragonCloud/DOAN_DLL$ spark-submit hadoop/Spark.py
```

age	gender	smoking_history	stage	count
30	0		1	3
31	0		0	1
32	0		2	3
33	0		2	0
34	1		0	3
35	1		0	3
36	1		0	0
37	0		0	2
38	0		0	2
39	0		2	0
40	1		1	0
41	1		0	1
42	0		2	1
43	1		2	0
44	1		1	1
45	0		1	1
46	1		0	2
47	0		1	3
48	1		2	3
49	1		0	2

Hình 9: Thống kê số lượng, giới tính chiếm giai đoạn cao nhất trong độ tuổi

CHƯƠNG 6: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

1. Kết luận

Trong báo cáo này, chúng em đã tìm hiểu và ứng dụng công nghệ lớn (Hadoop) để triển khai hệ thống quản lý thông tin bệnh nhân ung thư phổi. Trong quá trình nghiên cứu và thực hiện, chúng em đã đạt được các kết quả quan trọng:

- Hiểu biết sâu sắc về dữ liệu y tế: Qua việc thu thập và xử lý dữ liệu từ nhiều nguồn khác nhau, đồ án đã giúp làm sáng tỏ các đặc trưng, xu hướng và mối liên hệ giữa các yếu tố liên quan đến bệnh ung thư phổi.
- Ứng dụng công nghệ Big Data: Việc tích hợp các công nghệ như Hadoop và Spark đã chứng minh khả năng xử lý lượng dữ liệu lớn một cách hiệu quả, đảm bảo dữ liệu được lưu trữ và truy xuất nhanh chóng.
- Phân tích và trực quan hóa dữ liệu: Sử dụng các thư viện như Pandas, Numpy, Matplotlib và Seaborn, đồ án đã xây dựng được các biểu đồ, báo cáo trực quan hỗ trợ cho việc đánh giá, so sánh và dự đoán xu hướng bệnh tật.
- Xây dựng mô hình học máy: Các mô hình học máy được huấn luyện và đánh giá qua Scikit-Learn cho thấy tiềm năng trong việc dự đoán các chỉ số quan trọng về tình trạng sức khỏe của bệnh nhân, đồng thời hỗ trợ cho việc ra quyết định lâm sàng.
- Khả năng áp dụng thực tế: quản lý thông tin bệnh nhân và chẩn đoán những người có khả năng mắc bệnh.

2. Hạn chế của đồ án

Mặc dù đã đạt được nhiều kết quả quan trọng, đồ án vẫn tồn tại một số hạn chế cần được khắc phục trong các nghiên cứu và phát triển sau này:

- Chất lượng dữ liệu: Một số dữ liệu thu thập được có thể không đầy đủ hoặc chưa được cập nhật liên tục, ảnh hưởng đến độ chính xác của mô hình phân tích và dự đoán.
- Hiệu năng xử lý: Trong trường hợp dữ liệu đầu vào với khối lượng rất lớn, hiệu năng xử lý của hệ thống, đặc biệt là trong giai đoạn phân tích và trực quan hóa, cần được tối ưu hóa thêm.

- Giao diện người dùng: Mặc dù giao diện web hiện tại đã thân thiện và dễ sử dụng, nhưng vẫn còn tồn tại những điểm có thể cải thiện về tính tương tác và trải nghiệm người dùng.
- Khả năng mở rộng: Hệ thống cần được thiết kế với khả năng mở rộng linh hoạt hơn để đáp ứng yêu cầu xử lý dữ liệu và ứng dụng trong môi trường thực tế có sự thay đổi liên tục.

3. Hướng phát triển hệ thống

Dựa trên các kết quả đạt được và hạn chế đã nêu, đồ án đề xuất một số hướng phát triển cho giai đoạn tiếp theo:

- Nâng cao hiệu suất: Tiếp tục tối ưu thuật toán và mô hình để giảm thời gian xử lý, cải thiện độ chính xác và tính ổn định của hệ thống.
- Mở rộng quy mô ứng dụng: Triển khai hệ thống trong các môi trường thực tế hơn, thu thập thêm dữ liệu để nâng cao tính tổng quát của mô hình.
- Cải thiện giao diện người dùng: Tích hợp các tính năng trực quan hóa dữ liệu, xây dựng giao diện thân thiện hơn để người dùng dễ dàng thao tác.
- Tích hợp với các công nghệ khác: Kết hợp với các hệ thống hiện có hoặc mở rộng tính năng để đáp ứng nhiều nhu cầu hơn, chẳng hạn như [liệt kê một số công nghệ có thể tích hợp].
- Tăng cường bảo mật và tính riêng tư: Đảm bảo hệ thống có khả năng bảo vệ dữ liệu người dùng, tuân thủ các tiêu chuẩn bảo mật hiện hành.