|  |  |
| --- | --- |
| logo (CMYK)-01 | **BỘ GIÁO DỤC VÀ ĐÀO TẠO**  **TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM** |

**ĐỒ ÁN CƠ SỞ**

**PHÂN TÍCH RỦI RO TÍN DỤNG TRONG NGÂN HÀNG, SỬ DỤNG MÔ HÌNH HỌC MÁY ĐỂ ĐÁNH GIÁ KHẢ NĂNG VỠ NỢ CỦA KHÁCH HÀNG**

Ngành: **KHOA HỌC DỮ LIỆU**

Môn học: **ĐỒ ÁN CƠ SỞ**

Giảng viên hướng dẫn : THS. NGUYỄN QUANG PHÚC

Sinh viên thực hiện :

Nguyễn Anh Thao MSSV: 2286400030

Phạm Văn Thân MSSV: 2286400031

Phạm Ninh Thuận MSSV: 2286400037

Lớp: 22DKHA1

TP. Hồ Chí Minh, 2025

|  |  |
| --- | --- |
| logo (CMYK)-01 | **BỘ GIÁO DỤC VÀ ĐÀO TẠO**  **TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM** |

**ĐỒ ÁN CƠ SỞ**

**PHÂN TÍCH RỦI RO TÍN DỤNG TRONG NGÂN HÀNG, SỬ DỤNG MÔ HÌNH HỌC MÁY ĐỂ ĐÁNH GIÁ KHẢ NĂNG VỠ NỢ CỦA KHÁCH HÀNG**

Ngành: **KHOA HỌC DỮ LIỆU**

Môn học: **ĐỒ ÁN CƠ SỞ**

Giảng viên hướng dẫn : THS. NGUYỄN QUANG PHÚC

Sinh viên thực hiện :

Nguyễn Anh Thao MSSV: 2286400030

Phạm Văn Thân MSSV: 2286400031

Phạm Ninh Thuận MSSV: 2286400037

Lớp: 22DKHA1

TP. Hồ Chí Minh, 2025

**LỜI CAM ĐOAN**

Nhóm chúng tôi xin cam đoan rằng toàn bộ nội dung được trình bày trong báo cáo này là kết quả của quá trình học tập, nghiên cứu và thực hiện nghiêm túc của bản thân dưới sự hướng dẫn tận tình của giảng viên hướng dẫn. Trong suốt quá trình thực hiện, tôi đã cố gắng tiếp cận vấn đề một cách khoa học, trung thực và khách quan, đồng thời vận dụng các kiến thức đã học để phân tích và giải quyết bài toán một cách hiệu quả.

Báo cáo không sao chép bất kỳ nội dung nào từ các tài liệu, nguồn thông tin hoặc công trình nghiên cứu khác mà không trích dẫn rõ ràng. Tất cả dữ liệu, hình ảnh, bảng biểu và kết quả phân tích được sử dụng trong báo cáo đều có nguồn gốc rõ ràng và được thu thập, xử lý, phân tích bởi chính tôi hoặc thông qua các nguồn tài liệu hợp pháp, có dẫn chứng cụ thể.

Nhóm chúng tôi ý thức rõ tầm quan trọng của tính trung thực trong học thuật cũng như đạo đức nghề nghiệp. Do đó, tôi xin hoàn toàn chịu trách nhiệm trước Hội đồng đánh giá, Nhà trường và pháp luật về mọi nội dung trình bày trong báo cáo này. Nếu phát hiện có bất kỳ hành vi gian lận, sao chép, đạo văn hoặc vi phạm quy định học thuật nào, tôi xin chấp nhận mọi hình thức xử lý theo quy định hiện hành mà không có bất kỳ khiếu nại nào.

Chúng tôi kính mong nhận được sự xem xét, góp ý từ Quý Thầy Cô để tôi có thể hoàn thiện hơn về kiến thức chuyên môn và phương pháp nghiên cứu trong tương lai.

TP.HCM, Ngày...24..tháng...05..năm 2025

Sinh viên

Nguyễn Anh Thao

Phạm Văn Thân

Phạm Ninh Thuận

MỤC LỤC

[**Chương 1: TỔNG QUAN** 4](#_Toc199092091)

[**1.1 Giới thiệu về đề tài** 4](#_Toc199092092)

[**1.2 Nhiệm vụ của đề tài** 5](#_Toc199092093)

[*1.2.1 Tính cấp thiết của đề tài* 5](#_Toc199092094)

[*1.2.2 Ý nghĩa khoa học và thực tiễn của đề tài* 6](#_Toc199092095)

[**1.3 Mục tiêu** 6](#_Toc199092096)

[*1.3.1 Mục tiêu tổng quát* 7](#_Toc199092097)

[*1.3.2 Mục tiêu cụ thể* 7](#_Toc199092098)

[**1.4 Đối tượng và phạm vi** 7](#_Toc199092099)

[*1.4.1 Đối tượng* 7](#_Toc199092100)

[*1.4.2 Phạm vi* 7](#_Toc199092101)

[**1.5 Phương pháp nghiên cứu** 8](#_Toc199092102)

[*1.5.1 Phương pháp nghiên cứu số* 8](#_Toc199092103)

[*1.5.2 Phương pháp nghiên cứu tài liệu* 8](#_Toc199092104)

[*1.5.3 Phương pháp nghiên cứu thống kê* 8](#_Toc199092105)

[*1.5.4 Phương pháp nghiên cứu nghiệm thu* 9](#_Toc199092106)

[*1.5.5 Phương pháp đánh giá* 9](#_Toc199092107)

[**1.6 Những đóng góp nghiên cứu của đề tài** 9](#_Toc199092108)

[*1.6.1 Trong lĩnh vực học thuật* 9](#_Toc199092109)

[*1.6.2 Trong thực tiễn kinh doanh* 9](#_Toc199092110)

[**Chương 2: CƠ SỞ LÝ THUYẾT** 10](#_Toc199092111)

[**2.1 Logistic Regression** 10](#_Toc199092112)

[*2.1.1 Giới thiệu về thuật toán Logistic Regression* 10](#_Toc199092113)

[*2.1.2 Nền tảng toán học* 10](#_Toc199092114)

[*2.1.3 Diễn giải thuật toán* 12](#_Toc199092115)

[*2.1.4 Phân tích độ phức tạp* 12](#_Toc199092116)

[*2.1.5 Ưu điểm và hạn chế* 13](#_Toc199092117)

[*2.1.6 Ứng dụng* 13](#_Toc199092118)

[**2.2 Decision Tree** 13](#_Toc199092119)

[*2.2.1 Giới thiệu về thuật toán Decision Tree* 13](#_Toc199092120)

[*2.2.2 Nền tảng toán học* 14](#_Toc199092121)

[*2.2.3 Diễn giải thuật toán* 16](#_Toc199092122)

[*2.2.4 Phân tích độ phức tạp* 16](#_Toc199092123)

[*2.2.5 Ưu điểm và hạn chế* 17](#_Toc199092124)

[*2.2.6 Ứng dụng* 17](#_Toc199092125)

[**2.3 Random Forest** 17](#_Toc199092126)

[*2.3.1 Giới thiệu về thuật toán Random Forest* 17](#_Toc199092127)

[*2.3.2 Nền tảng toán học* 18](#_Toc199092128)

[*2.3.3 Diễn giải thuật toán* 20](#_Toc199092129)

[*2.3.4 Phân tích độ phức tạp* 20](#_Toc199092130)

[*2.3.5 Ưu điểm và hạn chế* 21](#_Toc199092131)

[*2.3.6 Ứng dụng* 21](#_Toc199092132)

[**2.4 Support Vector Machine (SVM)** 22](#_Toc199092133)

[*2.4.1 Giới thiệu về thuật toán Support Vector Machine* 22](#_Toc199092134)

[*2.4.2 Nền tảng toán học* 22](#_Toc199092135)

[*2.4.3 Diễn giải thuật toán* 24](#_Toc199092136)

[*2.4.4 Phân tích độ phức tạp* 25](#_Toc199092137)

[*2.4.5 Ưu điểm và hạn chế* 25](#_Toc199092138)

[*2.4.6 Ứng dụng* 25](#_Toc199092139)

[**2.5 Neural Network** 26](#_Toc199092140)

[*2.5.1 Giới thiệu về thuật toán Neural Network* 26](#_Toc199092141)

[*2.5.2 Nền tảng toán học* 26](#_Toc199092142)

[*2.5.3 Diễn giải thuật toán* 28](#_Toc199092143)

[*2.5.4 Phân tích độ phức tạp* 29](#_Toc199092144)

[*2.5.5 Ưu điểm và hạn chế* 29](#_Toc199092145)

[*2.5.6 Ứng dụng* 30](#_Toc199092146)

[**Chương 3: PHƯƠNG PHÁP THỰC NGHIỆM** 30](#_Toc199092147)

[**3.1 Dữ liệu** 30](#_Toc199092148)

[**3.2 Tiền Xử lý dữ liệu** 31](#_Toc199092149)

[**3.3 Chuẩn hóa dữ liệu** 32](#_Toc199092150)

[**3.4 Chia tập dữ liệu** 32](#_Toc199092151)

[**3.5 Cân bằng dữ liệu (SMOTE)** 33](#_Toc199092152)

[**3.6 Xây dựng mô hình** 34](#_Toc199092153)

[*3.6.1 Hồi quy logistic* 34](#_Toc199092154)

[*3.6.2 Cây quyết định* 34](#_Toc199092155)

[*3.6.3 Random Forest* 35](#_Toc199092156)

[*3.6.4 Support Vector Machine (SVM)* 35](#_Toc199092157)

[3.6.5 Mạng nơ-ron nhân tạo (Neural Network) 36](#_Toc199092158)

[**3.7 Đánh giá mô hình** 37](#_Toc199092159)

[Chương 4: KẾT QUẢ VÀ THẢO LUẬN 39](#_Toc199092160)

[**4.1 Kết quả thực nghiệm** 39](#_Toc199092161)

[**4.2 So sánh hiệu năng các mô hình** 40](#_Toc199092162)

[**Chương 5: KẾT LUẬN VÀ KIẾN NGHỊ** 41](#_Toc199092163)

[**5.1 Kết luận** 41](#_Toc199092164)

[**5.2 Kiến nghị** 42](#_Toc199092165)

[**TÀI LIỆU THAM KHẢO** 44](#_Toc199092166)

[**SOURCE CODE** 45](#_Toc199092167)

# **Chương 1: TỔNG QUAN**

**1.1 Giới thiệu về đề tài**Trong bối cảnh nền kinh tế toàn cầu hóa và sự phát triển mạnh mẽ của ngành tài chính - ngân hàng, rủi ro tín dụng đã trở thành một trong những thách thức lớn nhất đối với các tổ chức tín dụng. Rủi ro tín dụng xảy ra khi khách hàng không có khả năng thực hiện nghĩa vụ trả nợ đúng hạn, dẫn đến tổn thất tài chính cho ngân hàng. Việc quản lý và kiểm soát rủi ro tín dụng không chỉ giúp ngân hàng giảm thiểu tổn thất mà còn đảm bảo sự ổn định và bền vững trong hoạt động kinh doanh.

Với sự tiến bộ của công nghệ, đặc biệt là trí tuệ nhân tạo và học máy, các mô hình dự đoán hiện đại đã được áp dụng để đánh giá khả năng vỡ nợ của khách hàng một cách chính xác và hiệu quả hơn so với các phương pháp truyền thống. Đề tài “Phân tích rủi ro tín dụng trong ngân hàng, sử dụng mô hình học máy để đánh giá khả năng vỡ nợ của khách hàng” được thực hiện nhằm nghiên cứu và xây dựng một mô hình học máy để hỗ trợ các ngân hàng trong việc ra quyết định tín dụng.

**1.2 Nhiệm vụ của đề tài**Nhiệm vụ của đề tài là áp dụng các kỹ thuật xử lý dữ liệu và thuật toán học máy để phân tích và dự đoán khả năng vỡ nợ của khách hàng. Từ đó, xác định các yếu tố ảnh hưởng đến rủi ro tín dụng và đề xuất các giải pháp quản lý rủi ro hiệu quả. Điều này giúp các ngân hàng tối ưu hóa quy trình phê duyệt tín dụng, giảm thiểu tỷ lệ nợ xấu và tổn thất tài chính, đồng thời nâng cao khả năng quản lý rủi ro tín dụng trong bối cảnh kinh tế hiện tại và tương lai gần.

*1.2.1 Tính cấp thiết của đề tài*Trong những năm gần đây, tỷ lệ nợ xấu tại các ngân hàng thương mại có xu hướng gia tăng, đặc biệt trong bối cảnh kinh tế chịu ảnh hưởng bởi suy thoái, lạm phát và thị trường tài chính biến động mạnh mẽ. Việc đánh giá rủi ro tín dụng theo phương pháp truyền thống như hệ số tín dụng (credit scoring) thường không đủ chính xác và kịp thời để ngăn chặn các khoản vay có nguy cơ vỡ nợ cao. Sự thiếu chính xác và chậm trễ của các phương pháp này không đáp ứng được nhu cầu phân tích nhanh chóng với khối lượng dữ liệu khách hàng ngày càng lớn và phức tạp, dẫn đến việc bỏ lỡ các tín hiệu cảnh báo sớm về nguy cơ vỡ nợ.

Việc áp dụng học máy vào phân tích rủi ro tín dụng trở thành nhu cầu cấp thiết, giúp ngân hàng tăng cường hiệu quả quản lý tài sản bằng cách phân bổ nguồn vốn hợp lý, tránh đầu tư vào các khoản vay rủi ro cao. Học máy hỗ trợ ra quyết định chiến lược dựa trên dữ liệu thực tế, cho phép ngân hàng phản ứng linh hoạt với các thay đổi thị trường trong bối cảnh kinh tế hiện tại và tương lai gần. Tự động hóa quá trình đánh giá rủi ro giúp giảm chi phí vận hành, tiết kiệm thời gian và hạn chế sai sót do yếu tố con người. Các giải pháp tín dụng cá nhân hóa nâng cao trải nghiệm khách hàng, xây dựng lòng tin và thu hút khách hàng mới. Học máy còn cải thiện khả năng dự báo xu hướng tín dụng, đặc biệt khi các yếu tố như biến đổi khí hậu và công nghệ tài chính (fintech) tác động mạnh đến hành vi vay vốn. Đồng thời, nó tăng cường khả năng tuân thủ quy định pháp lý và các tiêu chuẩn quốc tế về quản lý rủi ro tín dụng, đặc biệt trong bối cảnh Việt Nam hội nhập sâu hơn vào nền kinh tế toàn cầu. Từ những lý do trên, đề tài mang lại lợi ích thiết thực cho các ngân hàng và đóng góp vào việc xây dựng một hệ thống tài chính bền vững.

*1.2.2 Ý nghĩa khoa học và thực tiễn của đề tài*Ý nghĩa khoa học: Đề tài đóng góp vào lĩnh vực phân tích dữ liệu và học máy thông qua việc áp dụng các thuật toán tiên tiến như hồi quy logistic, cây quyết định, rừng ngẫu nhiên và mạng nơ-ron để giải quyết bài toán dự đoán rủi ro tín dụng trong ngân hàng. Nghiên cứu cung cấp cơ sở lý thuyết mới về tích hợp học máy với dữ liệu tài chính, mở ra hướng tiếp cận hiện đại trong quản lý rủi ro tín dụng, đặc biệt trong bối cảnh Việt Nam đẩy mạnh chuyển đổi số. Kết quả nghiên cứu làm phong phú tài liệu học thuật về ứng dụng trí tuệ nhân tạo trong ngành tài chính, phát triển các phương pháp đánh giá hiệu suất mô hình thông qua các chỉ số như AUC-ROC, F1-score và ma trận nhầm lẫn. Việc so sánh hiệu quả giữa các thuật toán học máy cung cấp dữ liệu thực nghiệm giá trị, hỗ trợ các nhà nghiên cứu lựa chọn công cụ phù hợp cho các vấn đề tương tự.

Ý nghĩa thực tiễn: Đề tài hỗ trợ các ngân hàng ra quyết định tín dụng chính xác hơn thông qua dự đoán khả năng vỡ nợ, giảm tỷ lệ nợ xấu và bảo vệ nguồn vốn. Mô hình học máy tự động hóa quy trình đánh giá rủi ro, tiết kiệm thời gian, giảm chi phí vận hành và hạn chế sai sót. Nghiên cứu cung cấp cái nhìn chi tiết về các yếu tố ảnh hưởng đến rủi ro tín dụng, giúp xây dựng chính sách tín dụng linh hoạt theo từng nhóm khách hàng. Các giải pháp tín dụng cá nhân hóa cải thiện trải nghiệm khách hàng, tăng khả năng cạnh tranh của ngân hàng. Kết quả nghiên cứu không chỉ áp dụng cho các ngân hàng thương mại mà còn mở rộng sang các tổ chức tài chính khác, góp phần nâng cao chất lượng quản lý rủi ro trên toàn ngành. Các nhà quản lý và hoạch định chính sách có thể tận dụng nghiên cứu để thiết kế các quy định hỗ trợ ngành ngân hàng, đảm bảo môi trường tài chính lành mạnh và bền vững.

**1.3 Mục tiêu**

*1.3.1 Mục tiêu tổng quát*  
Phát triển một mô hình học máy tiên tiến và hiệu quả nhằm phân tích sâu sắc rủi ro tín dụng, dự đoán chính xác khả năng vỡ nợ của khách hàng dựa trên dữ liệu tài chính và hành vi vay vốn, từ đó cung cấp công cụ hỗ trợ các ngân hàng đưa ra các quyết định tín dụng tối ưu, giảm thiểu rủi ro tài chính và nâng cao hiệu quả quản lý trong bối cảnh kinh tế hiện tại và tương lai gần.

*1.3.2 Mục tiêu cụ thể*Nghiên cứu này thực hiện quá trình nghiên cứu toàn diện, bắt đầu bằng việc tìm hiểu và đánh giá các phương pháp học máy như hồi quy logistic, cây quyết định và mạng nơ-ron để áp dụng vào phân tích rủi ro tín dụng. Tiếp theo, thu thập và xử lý dữ liệu từ các tập dữ liệu công khai trên Kaggle liên quan đến lịch sử tín dụng, thu nhập, tài sản và các yếu tố khác của khách hàng, đảm bảo dữ liệu được làm sạch và chuẩn hóa. Sau đó, xây dựng và huấn luyện các mô hình học máy để dự đoán khả năng vỡ nợ, với sự hỗ trợ của các công cụ như Python và thư viện Scikit-learn. Cuối cùng, đánh giá hiệu suất của các mô hình thông qua các chỉ số như độ chính xác, độ nhạy và ma trận nhầm lẫn, đồng thời đề xuất các giải pháp cải tiến cụ thể nhằm tối ưu hóa quy trình quản lý rủi ro tín dụng trong bối cảnh thị trường tài chính biến động mạnh mẽ.

**1.4 Đối tượng và phạm vi**

*1.4.1 Đối tượng*Đối tượng nghiên cứu của đề tài là các khách hàng vay vốn tại các ngân hàng thương mại, bao gồm cả khách hàng cá nhân và doanh nghiệp nhỏ, với trọng tâm là các dữ liệu liên quan đến lịch sử tín dụng, thu nhập hàng tháng, tài sản đảm bảo, nợ hiện tại và các yếu tố kinh tế - xã hội ảnh hưởng đến khả năng trả nợ của họ. Ngoài ra, đề tài cũng xem xét các đặc điểm hành vi tài chính của khách hàng như tần suất giao dịch, lịch sử trả nợ và các khoản vay trước đó, nhằm cung cấp một bức tranh toàn diện về nguy cơ vỡ nợ.

*1.4.2 Phạm vi*Phạm vi nghiên cứu tập trung vào việc ứng dụng các thuật toán học máy hiện đại như hồi quy logistic, cây quyết định, rừng ngẫu nhiên và mạng nơ-ron để phân tích rủi ro tín dụng, với dữ liệu từ các tập dữ liệu công khai trên Kaggle, bao gồm khoảng 10.000-50.000 bản ghi về thông tin tài chính và phi tài chính của khách hàng vay vốn. Các thuật toán được chọn vì hồi quy logistic phù hợp với dữ liệu tuyến tính và dễ giải thích, cây quyết định và rừng ngẫu nhiên hiệu quả với dữ liệu phi tuyến và tương tác phức tạp, trong khi mạng nơ-ron xử lý tốt dữ liệu lớn và các mẫu phức tạp. Đề tài thực hiện các thí nghiệm trên các tập dữ liệu giới hạn để đảm bảo tính khả thi, với trọng tâm là xây dựng và kiểm tra mô hình dự đoán khả năng vỡ nợ trong bối cảnh kinh tế hiện tại và tương lai gần, đồng thời đề xuất các giải pháp quản lý rủi ro tín dụng có thể áp dụng ngay tại các ngân hàng trong ngắn hạn.

**1.5 Phương pháp nghiên cứu**

*1.5.1 Phương pháp nghiên cứu số* Sử dụng các công cụ lập trình tiên tiến như Python và R để xử lý và phân tích dữ liệu tín dụng, với sự hỗ trợ của các thư viện học máy như Scikit-learn, TensorFlow và Keras, nhằm xây dựng và huấn luyện các mô hình dự đoán rủi ro tín dụng dựa trên dữ liệu khách hàng vay vốn. Áp dụng các kỹ thuật tiền xử lý dữ liệu như làm sạch dữ liệu, chuẩn hóa và mã hóa biến mục tiêu, đảm bảo dữ liệu phù hợp với các thuật toán học máy, đặc biệt trong bối cảnh dữ liệu tài chính phức tạp.

*1.5.2 Phương pháp nghiên cứu tài liệu*Tham khảo các tài liệu học thuật, báo cáo nghiên cứu và sách chuyên ngành về rủi ro tín dụng, học máy và ứng dụng trí tuệ nhân tạo trong tài chính, bao gồm các bài báo từ các tạp chí quốc tế như Journal of Banking & Finance và các nghiên cứu nội địa. Phân tích các nghiên cứu trước đây về đánh giá rủi ro tín dụng để xác định khoảng trống kiến thức và áp dụng vào bối cảnh thực tế của ngành ngân hàng.

*1.5.3 Phương pháp nghiên cứu thống kê*Sử dụng các kỹ thuật thống kê mô tả để phân tích xu hướng rủi ro tín dụng, bao gồm tính toán các chỉ số như tỷ lệ nợ xấu, trung bình và độ lệch chuẩn của dữ liệu tín dụng. Áp dụng các phương pháp thống kê suy luận như kiểm định giả thuyết để đánh giá mối quan hệ giữa các biến độc lập (như thu nhập, lịch sử tín dụng) và biến phụ thuộc (khả năng vỡ nợ).

*1.5.4 Phương pháp nghiên cứu nghiệm thu*Thực hiện kiểm tra thực nghiệm trên các tập dữ liệu từ Kaggle, bao gồm dữ liệu lịch sử vay vốn và thông tin khách hàng, để đánh giá tính khả thi và hiệu quả của mô hình học máy trong việc dự đoán rủi ro tín dụng. So sánh kết quả từ mô hình học máy với các phương pháp truyền thống như điểm tín dụng thủ công để xác nhận ưu thế của cách tiếp cận mới trong quản lý rủi ro.

*1.5.5 Phương pháp đánh giá*Áp dụng ma trận nhầm lẫn để tính toán các chỉ số hiệu suất như Accuracy, Recall và F1-score, từ đó đánh giá khả năng phân loại đúng các trường hợp vỡ nợ và không vỡ nợ. Sử dụng chỉ số ROC để đo lường khả năng phân biệt của mô hình giữa các lớp vỡ nợ và không vỡ nợ, đảm bảo mô hình đạt hiệu suất cao trong các tình huống thực tế.

**1.6 Những đóng góp nghiên cứu của đề tài**

*1.6.1 Trong lĩnh vực học thuật*  
Đề tài đóng góp vào lý thuyết ứng dụng học máy trong phân tích rủi ro tín dụng, sử dụng các thuật toán như hồi quy logistic, cây quyết định và mạng nơ-ron để dự đoán khả năng vỡ nợ, đồng thời mở rộng kiến thức về tiền xử lý dữ liệu tài chính và đánh giá mô hình qua các chỉ số như AUC-ROC và F1-score, tạo cơ sở cho các nghiên cứu tiếp theo trong lĩnh vực tài chính và khoa học dữ liệu.

*1.6.2 Trong thực tiễn kinh doanh*  
Đề tài cung cấp công cụ dự đoán rủi ro tín dụng, giúp ngân hàng đưa ra quyết định tín dụng chính xác, giảm nợ xấu, tự động hóa quy trình đánh giá để tiết kiệm chi phí và nâng cao hiệu quả hoạt động, đồng thời hỗ trợ xây dựng chính sách tín dụng linh hoạt, tăng khả năng cạnh tranh và góp phần ổn định hệ thống tài chính.

# **Chương 2: CƠ SỞ LÝ THUYẾT**

**2.1 Logistic Regression**

*2.1.1 Giới thiệu về thuật toán Logistic Regression*  
Logistic Regression là một thuật toán học máy giám sát được sử dụng rộng rãi để giải quyết các bài toán phân loại nhị phân, đặc biệt phù hợp với các vấn đề như dự đoán khả năng vỡ nợ trong rủi ro tín dụng. Thuật toán này dự đoán xác suất một sự kiện thuộc vào một lớp cụ thể, chẳng hạn như khách hàng có vỡ nợ hay không, bằng cách sử dụng hàm sigmoid để ánh xạ giá trị đầu ra vào khoảng [0, 1]. Logistic Regression nổi bật nhờ tính đơn giản, dễ triển khai và khả năng giải thích cao, làm cho nó trở thành lựa chọn phổ biến trong ngành tài chính và ngân hàng. Với hiệu quả trong việc xử lý dữ liệu tuyến tính, thuật toán này là công cụ quan trọng để phân tích rủi ro tín dụng, hỗ trợ ngân hàng đưa ra quyết định tín dụng chính xác.[1]

*2.1.2 Nền tảng toán học*  
Logistic Regression dự đoán xác suất một điểm dữ liệu thuộc vào lớp dương (ví dụ: vỡ nợ) thông qua hàm sigmoid. Cho tập dữ liệu với n điểm {xi, yi}, trong đó xi là vector đặc trưng và yi là nhãn lớp (0 hoặc 1), mô hình được định nghĩa như sau:



Công thức 2.1

Trong đó:

* P(y=1|x): Xác suất điểm dữ liệu x thuộc lớp 1.
* σ(z): Hàm sigmoid, σ(z) = 1 / (1 + e^-z).
* w: Vector trọng số của các đặc trưng.
* b: Hệ số chặn (bias).
* w^T x: Tích vô hướng giữa vector trọng số và vector đặc trưng.

Để huấn luyện mô hình, Logistic Regression tối ưu hóa hàm mất mát (log-loss hoặc cross-entropy):



Công thức 2.2

Trong đó:

* L(w, b): Hàm mất mát trung bình trên tập dữ liệu.
* : Nhãn thực tế của điểm dữ liệu i (0 hoặc 1).
* : Xác suất dự đoán cho điểm dữ liệu i.

Quá trình tối ưu hóa sử dụng phương pháp gradient descent để cập nhật w và b sao cho hàm mất mát đạt giá trị nhỏ nhất:



Công thức 2.3 (Cập nhật gradient descent)

Trong đó:

* α: Tỷ lệ học (learning rate).
* : Đạo hàm riêng của hàm mất mát theo w và b.

*2.1.3 Diễn giải thuật toán*  
Logistic Regression hoạt động dựa trên nguyên tắc tối ưu hóa xác suất dự đoán đúng lớp của các điểm dữ liệu. Các bước chính bao gồm:

* Khởi tạo: Gán giá trị ban đầu ngẫu nhiên cho vector trọng số w và hệ số chặn b.
* Dự đoán: Với mỗi điểm dữ liệu, tính xác suất thuộc lớp dương bằng hàm sigmoid.
* Tính toán mất mát: Sử dụng hàm log-loss để đo lường sai số giữa dự đoán và nhãn thực tế.
* Cập nhật tham số: Áp dụng gradient descent để điều chỉnh w và b, giảm giá trị hàm mất mát.
* Lặp lại: Tiếp tục các bước dự đoán và cập nhật cho đến khi hàm mất mát hội tụ hoặc đạt số lần lặp tối đa.

Thuật toán đảm bảo mô hình học được mối quan hệ tuyến tính giữa các đặc trưng và xác suất xảy ra sự kiện, phù hợp với bài toán phân loại nhị phân như dự đoán vỡ nợ.

*2.1.4 Phân tích độ phức tạp*  
Độ phức tạp của Logistic Regression phụ thuộc vào số lượng điểm dữ liệu n, số chiều đặc trưng d và số lần lặp I của gradient descent. Trong mỗi lần lặp, việc tính toán xác suất và gradient có độ phức tạp O(n·d). Do đó, độ phức tạp thời gian tổng thể của thuật toán là O(I·n·d). Với các tập dữ liệu lớn, thuật toán vẫn hiệu quả nhờ tính đơn giản, nhưng có thể yêu cầu tối ưu hóa thêm (như sử dụng stochastic gradient descent) để giảm thời gian tính toán.

### *2.1.5 Ưu điểm và hạn chế*

* Ưu điểm: Logistic Regression dễ triển khai, có tốc độ huấn luyện nhanh và cho phép giải thích mối quan hệ giữa đặc trưng và kết quả dự đoán. Thuật toán hoạt động tốt với dữ liệu có mối quan hệ tuyến tính và phù hợp với bài toán phân loại nhị phân như rủi ro tín dụng.
* Hạn chế: Thuật toán giả định dữ liệu có mối quan hệ tuyến tính, do đó không hiệu quả với các bài toán có ranh giới quyết định phi tuyến phức tạp. Nó cũng nhạy cảm với dữ liệu mất cân bằng (ví dụ: số lượng khách hàng vỡ nợ quá ít so với không vỡ nợ) và có thể bị ảnh hưởng bởi các giá trị ngoại lai.

### *2.1.6 Ứng dụng*

* Trong ngành tài chính, Logistic Regression được sử dụng để dự đoán khả năng vỡ nợ của khách hàng dựa trên lịch sử tín dụng, thu nhập và các yếu tố kinh tế - xã hội, hỗ trợ ngân hàng phê duyệt khoản vay.
* Thuật toán được áp dụng trong phân tích rủi ro tín dụng để xác định các khách hàng có nguy cơ cao, giúp giảm tỷ lệ nợ xấu.
* Trong marketing, Logistic Regression phân loại khách hàng theo khả năng mua sản phẩm hoặc dịch vụ dựa trên hành vi và đặc điểm nhân khẩu học.
* Trong y tế, thuật toán dự đoán khả năng mắc bệnh dựa trên các chỉ số sức khỏe, tương tự như dự đoán vỡ nợ trong tài chính.

**2.2 Decision Tree**

*2.2.1 Giới thiệu về thuật toán Decision Tree*  
Decision Tree (Cây quyết định) là một thuật toán học máy giám sát được sử dụng phổ biến cho cả bài toán phân loại và hồi quy, đặc biệt hiệu quả trong các bài toán phân loại nhị phân như dự đoán khả năng vỡ nợ trong rủi ro tín dụng[14]. Thuật toán xây dựng một mô hình dạng cây, trong đó mỗi nút biểu diễn một quyết định dựa trên một đặc trưng, mỗi nhánh biểu thị kết quả của quyết định, và mỗi lá biểu thị một lớp hoặc giá trị dự đoán. Decision Tree nổi bật nhờ khả năng trực quan, dễ giải thích và xử lý tốt các mối quan hệ phi tuyến giữa các đặc trưng. Trong ngành tài chính, thuật toán này được ứng dụng rộng rãi để phân loại khách hàng dựa trên các yếu tố như lịch sử tín dụng, thu nhập, và nợ hiện tại, hỗ trợ ngân hàng đưa ra quyết định tín dụng chính xác và giảm thiểu rủi ro.[2]

*2.2.2 Nền tảng toán học*  
Decision Tree xây dựng cây bằng cách chia tập dữ liệu thành các tập con dựa trên các đặc trưng, tối ưu hóa một tiêu chí như entropy hoặc Gini index để đảm bảo các tập con đồng nhất nhất có thể. Các khái niệm toán học chính bao gồm entropy và information gain, được định nghĩa như sau:

Entropy đo lường mức độ không chắc chắn của một tập dữ liệu D với các lớp {c\_1, c\_2, ..., c\_m}:



Công thức 2.4 (Entropy)[3]

Trong đó:

* H(D): Entropy của tập dữ liệu D.
* pj: Tỷ lệ các điểm dữ liệu thuộc lớp cj trong 
* : Logarit cơ số 2.
* m: Số lớp (trong rủi ro tín dụng, m=2: vỡ nợ và không vỡ nợ).

Information gain đo lường mức độ giảm entropy khi tập dữ liệu D được chia thành các tập con {D\_1, D\_2, ..., D\_k} dựa trên một đặc trưng A:



Công thức 2.5 (Information gain)[4]

Trong đó:

* IG(D, A): Information gain khi chia D theo đặc trưng A.
* Dv: Tập con của D tương ứng với giá trị v của đặc trưng A.
* : Tỷ lệ số điểm dữ liệu trong Dv so với D.
* H(Dv): Entropy của tập con Dv.
* k: Số giá trị có thể của đặc trưng A.

Tại mỗi nút, thuật toán chọn đặc trưng A có information gain cao nhất để phân chia. Quá trình này được lặp lại đệ quy cho đến khi đạt tiêu chí dừng (ví dụ: tất cả điểm trong tập con thuộc cùng một lớp, hoặc độ sâu cây đạt giới hạn). Để tránh overfitting, cây có thể được cắt tỉa (pruning) bằng cách sử dụng tiêu chí như cost-complexity:



Công thức 2.6 (Cost-complexity)

Trong đó:

* Cα(T): Hàm chi phí tổng quát của cây T với tham số phạt α.
* C(T): Tổng lỗi phân loại của cây T (tỷ lệ dự đoán sai).
* |T|: Số nút lá của cây T.
* α: Tham số điều chỉnh mức độ cắt tỉa (α càng lớn, cây càng đơn giản).

*2.2.3 Diễn giải thuật toán*  
Decision Tree hoạt động bằng cách chia không gian đặc trưng thành các vùng dựa trên các điều kiện quyết định, được biểu diễn dưới dạng cây. Các bước chính bao gồm:

* Khởi tạo: Bắt đầu với toàn bộ tập dữ liệu D tại nút gốc.
* Chọn đặc trưng: Tính information gain (hoặc tiêu chí khác như Gini index) cho mỗi đặc trưng, chọn đặc trưng có giá trị cao nhất để phân chia tập dữ liệu thành các tập con.
* Phân nhánh: Tạo các nhánh con tương ứng với các giá trị của đặc trưng đã chọn, gán các tập con vào các nhánh.
* Lặp lại: Áp dụng quá trình trên cho mỗi nhánh con, đệ quy cho đến khi đạt tiêu chí dừng (ví dụ: tất cả điểm thuộc cùng lớp, hoặc độ sâu tối đa).
* Cắt tỉa: Loại bỏ các nhánh không cần thiết dựa trên tiêu chí như cost-complexity để giảm overfitting.  
  Kết quả là một cây quyết định, trong đó mỗi đường dẫn từ gốc đến lá biểu thị một quy tắc phân loại (ví dụ: “Nếu thu nhập < 50 triệu và nợ > 100 triệu, thì vỡ nợ”).

*2.2.4 Phân tích độ phức tạp*  
Độ phức tạp của Decision Tree phụ thuộc vào số lượng điểm dữ liệu n, số đặc trưng d, và độ sâu của cây h. Trong quá trình xây dựng cây, việc tính information gain cho mỗi đặc trưng tại mỗi nút yêu cầu duyệt qua tất cả điểm dữ liệu, dẫn đến độ phức tạp O(n·d·h) trong trường hợp xấu nhất (cây không cân bằng). Với cây cân bằng, độ phức tạp có thể giảm xuống O(n·d·log(n)). Quá trình cắt tỉa có độ phức tạp phụ thuộc vào số nút, thường là O(n). Trong giai đoạn dự đoán, việc đi từ gốc đến lá có độ phức tạp O(h), thường nhỏ so với n. Decision Tree hiệu quả với tập dữ liệu vừa và nhỏ, nhưng có thể tốn tài nguyên với dữ liệu lớn nếu không tối ưu.

### *2.2.5 Ưu điểm và hạn chế*

* Ưu điểm: Decision Tree dễ hiểu và trực quan, cho phép giải thích các quyết định phân loại thông qua các quy tắc đơn giản. Thuật toán xử lý tốt các mối quan hệ phi tuyến và không yêu cầu chuẩn hóa dữ liệu. Nó phù hợp với bài toán rủi ro tín dụng nhờ khả năng xác định các yếu tố quan trọng (như lịch sử tín dụng, thu nhập).
* Hạn chế: Decision Tree dễ bị overfitting, đặc biệt khi cây quá sâu và không được cắt tỉa. Thuật toán nhạy cảm với nhiễu trong dữ liệu và có thể tạo ra các cây khác nhau với các mẫu dữ liệu khác nhau. Nó không hiệu quả với các tập dữ liệu mất cân bằng (ví dụ: số khách hàng vỡ nợ quá ít), trừ khi kết hợp với kỹ thuật như cân bằng mẫu.

### *2.2.6 Ứng dụng*

* Trong ngành tài chính, Decision Tree được sử dụng để dự đoán khả năng vỡ nợ của khách hàng dựa trên các đặc trưng như lịch sử tín dụng, tỷ lệ nợ trên thu nhập, và tài sản đảm bảo, hỗ trợ ngân hàng phê duyệt khoản vay.
* Thuật toán giúp xây dựng hệ thống chấm điểm tín dụng (credit scoring), xác định các nhóm khách hàng rủi ro cao để giảm tỷ lệ nợ xấu.
* Trong marketing, Decision Tree phân loại khách hàng theo hành vi mua sắm hoặc khả năng phản hồi chiến dịch quảng cáo.
* Trong y tế, thuật toán dự đoán nguy cơ mắc bệnh dựa trên các chỉ số sức khỏe, tương tự như dự đoán rủi ro tín dụng trong tài chính.{Citation}

**2.3 Random Forest**

*2.3.1 Giới thiệu về thuật toán Random Forest*  
Random Forest là một thuật toán học máy giám sát thuộc nhóm ensemble, kết hợp nhiều cây quyết định (Decision Tree) để cải thiện độ chính xác và độ ổn định của mô hình[15] . Thuật toán sử dụng kỹ thuật bagging (Bootstrap Aggregating) và lựa chọn đặc trưng ngẫu nhiên để xây dựng các cây quyết định độc lập, sau đó tổng hợp kết quả bằng cách bỏ phiếu (cho phân loại) hoặc lấy trung bình (cho hồi quy). Random Forest đặc biệt hiệu quả trong các bài toán phân loại nhị phân như dự đoán khả năng vỡ nợ trong rủi ro tín dụng, nhờ khả năng xử lý dữ liệu phức tạp và giảm thiểu overfitting[10] . Trong ngành tài chính, Random Forest được ưa chuộng vì độ chính xác cao, khả năng xử lý các đặc trưng đa dạng (như lịch sử tín dụng, thu nhập, nợ), và tính ổn định khi phân tích dữ liệu tài chính lớn.

*2.3.2 Nền tảng toán học*  
Random Forest dựa trên cơ chế ensemble của Decision Tree, sử dụng bagging và lựa chọn đặc trưng ngẫu nhiên để xây dựng các cây. Các khái niệm toán học chính bao gồm entropy và information gain (đã trình bày trong phần 2.2.2, công thức 2.4, 2.5) để phân chia nút, cùng với cơ chế tổng hợp kết quả. Dưới đây là các công thức chính:

Cho tập dữ liệu D với n điểm dữ liệu, Random Forest tạo T cây quyết định từ T tập con D\_t (t=1,...,T) được lấy mẫu ngẫu nhiên từ D bằng bagging (lấy mẫu có thay thế). Entropy và information gain để phân chia nút trong mỗi cây được định nghĩa như trong công thức (2.4) và (2.5):



Công thức 2.7 (Entropy)[5]

Trong đó:

* H(Dt): Entropy của tập con Dt được lấy mẫu cho cây t.
* pj: Tỷ lệ các điểm dữ liệu thuộc lớp cj trong Dt
* log2: Logarit cơ số 2.
* m: Số lớp (m=2: vỡ nợ và không vỡ nợ).

Information gain được tính tương tự công thức (2.5) cho mỗi đặc trưng trong tập đặc trưng ngẫu nhiên tại mỗi nút. Để tổng hợp kết quả từ T cây, Random Forest sử dụng phương pháp bỏ phiếu đa số (majority voting) cho bài toán phân loại:



Công thức 2.8 (Majority voting)

Trong đó:

* : Nhãn dự đoán cuối cùng cho điểm dữ liệu x.
* : Nhãn dự đoán của cây t cho điểm x (0 hoặc 1).
* mode: Hàm chọn giá trị xuất hiện nhiều nhất (lớp đa số).
* T: Số cây trong rừng.

Để đánh giá độ quan trọng của đặc trưng A, Random Forest tính mức giảm trung bình của impurity (ví dụ: Gini index hoặc entropy) khi sử dụng A để phân chia:



Công thức 2.9 (Feature importance)

Trong đó:

* FI(A): Độ quan trọng của đặc trưng A (feature importance).
* Dnode: Tập dữ liệu tại nút sử dụng A để phân chia trong cây t.
* |: Tỷ lệ số điểm dữ liệu tại nút so với Dt.
* : Mức giảm impurity (entropy hoặc Gini) tại nút.
* T: Số cây trong rừng.

*2.3.3 Diễn giải thuật toán*  
Random Forest hoạt động bằng cách xây dựng nhiều cây quyết định từ các tập dữ liệu và đặc trưng ngẫu nhiên, sau đó tổng hợp kết quả để đưa ra dự đoán cuối cùng. Các bước chính bao gồm:

* Lấy mẫu: Tạo T tập con Dt từ tập dữ liệu D bằng bagging (lấy mẫu ngẫu nhiên có thay thế, mỗi Dt có kích thước tương đương D).
* Xây dựng cây: Với mỗi Dt, xây dựng một cây quyết định bằng cách:
  + Tại mỗi nút, chọn ngẫu nhiên một tập con đặc trưng (thường là sqrt(d) hoặc log2(d), với d là số đặc trưng).
  + Tính information gain (hoặc Gini index) để chọn đặc trưng tốt nhất từ tập con đặc trưng, phân chia tập dữ liệu thành các nhánh con.
  + Lặp lại cho đến khi đạt tiêu chí dừng (ví dụ: tất cả điểm thuộc cùng lớp, hoặc độ sâu tối đa).
* Tổng hợp: Đối với bài toán phân loại, thu thập dự đoán từ T cây và chọn lớp đa số bằng bỏ phiếu (majority voting).
* Đánh giá đặc trưng: Tính độ quan trọng của mỗi đặc trưng dựa trên mức giảm impurity trung bình.  
  Kết quả là một mô hình mạnh mẽ, ít bị overfitting hơn Decision Tree, phù hợp với các bài toán phức tạp như phân tích rủi ro tín dụng.

*2.3.4 Phân tích độ phức tạp*  
Độ phức tạp của Random Forest phụ thuộc vào số cây T, số điểm dữ liệu n, số đặc trưng d, và độ sâu trung bình h của mỗi cây. Xây dựng một cây quyết định có độ phức tạp O(n·d·h) trong trường hợp xấu nhất (như phần 2.2.4). Với T cây, độ phức tạp xây dựng Random Forest là O(T·n·d·h). Tuy nhiên, do chọn ngẫu nhiên một tập con đặc trưng (thường nhỏ hơn d), độ phức tạp thực tế có thể giảm. Trong giai đoạn dự đoán, việc đi qua T cây có độ phức tạp O(T·h). Random Forest tốn tài nguyên tính toán hơn Decision Tree, nhưng có thể song song hóa để tăng tốc độ, phù hợp với các tập dữ liệu lớn trong tài chính.

### *2.3.5 Ưu điểm và hạn chế*

* Ưu điểm: Random Forest có độ chính xác cao, ít bị overfitting nhờ ensemble và bagging. Thuật toán xử lý tốt các mối quan hệ phi tuyến, dữ liệu mất cân bằng, và các đặc trưng đa dạng. Nó cung cấp độ quan trọng của đặc trưng, hữu ích trong việc xác định các yếu tố chính ảnh hưởng đến rủi ro tín dụng (như lịch sử tín dụng, tỷ lệ nợ).
* Hạn chế: Random Forest phức tạp hơn Decision Tree, đòi hỏi tài nguyên tính toán lớn khi T hoặc n lớn. Mô hình khó giải thích hơn so với một cây quyết định đơn lẻ, có thể gây khó khăn trong việc trình bày các quy tắc cụ thể cho ngân hàng. Thuật toán cũng có thể chậm hơn khi dự đoán trên dữ liệu mới nếu số cây lớn.

### *2.3.6 Ứng dụng*

* Trong ngành tài chính, Random Forest được sử dụng để dự đoán khả năng vỡ nợ của khách hàng dựa trên các đặc trưng như lịch sử tín dụng, thu nhập, nợ hiện tại, và các yếu tố kinh tế - xã hội, hỗ trợ ngân hàng tối ưu hóa phê duyệt khoản vay.
* Thuật toán giúp xây dựng hệ thống chấm điểm tín dụng, xác định các khách hàng rủi ro cao để giảm tỷ lệ nợ xấu.
* Trong marketing, Random Forest phân loại khách hàng theo khả năng mua sản phẩm hoặc tham gia chương trình khuyến mãi.
* Trong y tế, thuật toán dự đoán nguy cơ mắc bệnh dựa trên các chỉ số sức khỏe, tương tự như dự đoán rủi ro tín dụng trong tài chính.[6]

**2.4 Support Vector Machine (SVM)**

### *2.4.1 Giới thiệu về thuật toán Support Vector Machine*

Support Vector Machine (SVM) là một thuật toán học máy giám sát mạnh mẽ, được sử dụng chủ yếu cho các bài toán phân loại nhị phân và có thể mở rộng cho phân loại đa lớp hoặc hồi quy . SVM tìm một siêu phẳng tối ưu để phân tách các lớp dữ liệu với khoảng cách margin lớn nhất, đảm bảo khả năng tổng quát hóa tốt. Thuật toán đặc biệt hiệu quả trong các bài toán như dự đoán khả năng vỡ nợ trong rủi ro tín dụng, nhờ khả năng xử lý dữ liệu phi tuyến thông qua kỹ thuật kernel trick . Trong ngành tài chính, SVM được ứng dụng để phân loại khách hàng dựa trên các đặc trưng như lịch sử tín dụng, thu nhập, và tỷ lệ nợ, giúp ngân hàng đưa ra quyết định tín dụng chính xác và giảm thiểu rủi ro nợ xấu.[7]

*2.4.2 Nền tảng toán học*  
SVM tìm siêu phẳng tối ưu w^T x + b = 0 để phân tách hai lớp dữ liệu (ví dụ: vỡ nợ và không vỡ nợ) với margin lớn nhất. Cho tập dữ liệu D = {(x\_i, y\_i) | i=1,...,n}, trong đó x\_i là vector đặc trưng (k chiều) và y\_i là nhãn lớp (y\_i ∈ {-1, 1}), bài toán tối ưu hóa của SVM (hard-margin) được định nghĩa như sau:



Công thức 2.10 [8]

Trong đó:

* w: Vector trọng số xác định hướng của siêu phẳng.
* b: Hệ số chặn (bias) xác định vị trí siêu phẳng.
* : Bình phương độ dài vector w, tỷ lệ nghịch với margin (khoảng cách từ siêu phẳng đến các điểm gần nhất).
* : Ràng buộc đảm bảo các điểm dữ liệu nằm đúng phía của siêu phẳng với margin ít nhất là 1.

Để xử lý dữ liệu không phân tách tuyến tính, SVM sử dụng soft-margin bằng cách thêm biến chùng (slack variables) ξ\_i và tham số phạt C:



Công thức 2.11[8]

Trong đó:

* ξi: Biến chùng, cho phép một số điểm nằm sai phía hoặc gần siêu phẳng.
* C: Tham số điều chỉnh giữa tối ưu hóa margin và giảm lỗi phân loại (C lớn ưu tiên phân loại đúng, C nhỏ ưu tiên margin lớn).

Bài toán tối ưu hóa được giải bằng cách sử dụng hàm Lagrange và chuyển sang dạng đối ngẫu:





Công thức 2.12 [8]

Trong đó:

* : Các hệ số Lagrange (α\_i > 0 cho các vector hỗ trợ - support vectors).
* : Hàm kernel (ví dụ: cho kernel tuyến tính, hoặc = cho kernel RBF).
* Siêu phẳng được xác định bởi  và b được tính từ các vector hỗ trợ.

Hàm kernel cho phép SVM ánh xạ dữ liệu vào không gian đặc trưng cao hơn, xử lý các bài toán phi tuyến mà không cần tính toán trực tiếp trong không gian đó.

*2.4.3 Diễn giải thuật toán*  
SVM hoạt động bằng cách tìm siêu phẳng tối ưu để phân tách các lớp dữ liệu với margin lớn nhất, sử dụng các vector hỗ trợ để xác định ranh giới. Các bước chính bao gồm:

* Khởi tạo: Chuẩn bị tập dữ liệu D với các nhãn {-1, 1} và chọn hàm kernel (tuyến tính, RBF, hoặc đa thức).
* Tối ưu hóa: Giải bài toán đối ngẫu (công thức 2.12) để tìm các hệ số α\_i, từ đó xác định vector trọng số w và hệ số chặn b.
* Xác định vector hỗ trợ: Các điểm dữ liệu có α\_i > 0 là vector hỗ trợ, quyết định vị trí siêu phẳng.
* Dự đoán: Với điểm dữ liệu mới x, tính để dự đoán nhãn lớp (1 hoặc -1).
* Điều chỉnh tham số: Tinh chỉnh tham số C và tham số kernel (như σ trong kernel RBF) để tối ưu hóa hiệu suất trên tập kiểm tra.  
  SVM đặc biệt phù hợp với các bài toán rủi ro tín dụng, nơi dữ liệu có thể có ranh giới phi tuyến phức tạp.

*2.4.4 Phân tích độ phức tạp*  
Độ phức tạp của SVM phụ thuộc vào số điểm dữ liệu n, số chiều đặc trưng k, và thuật toán giải bài toán tối ưu hóa (thường là Sequential Minimal Optimization - SMO). Trong trường hợp xấu nhất, độ phức tạp huấn luyện là đến do tính toán ma trận kernel và giải bài toán đối ngẫu. Với kernel tuyến tính và dữ liệu thưa, độ phức tạp có thể giảm xuống O(n·k). Trong giai đoạn dự đoán, độ phức tạp là , trong đó n\_sv là số vector hỗ trợ. SVM tốn tài nguyên tính toán với tập dữ liệu lớn, nhưng hiệu quả với dữ liệu có số chiều cao hoặc ranh giới lớp rõ ràng.

### *2.4.5 Ưu điểm và hạn chế*

* Ưu điểm: SVM có khả năng tổng quát hóa tốt nhờ tối ưu hóa margin, xử lý hiệu quả các bài toán phi tuyến thông qua kernel trick. Thuật toán hoạt động tốt với dữ liệu có số chiều cao và phù hợp với bài toán rủi ro tín dụng, nơi các đặc trưng như lịch sử tín dụng và thu nhập có thể tạo ranh giới phức tạp.
* Hạn chế: SVM yêu cầu tài nguyên tính toán lớn với tập dữ liệu lớn, nhạy cảm với việc chọn tham số (C, σ) và hàm kernel. Thuật toán khó giải thích so với Decision Tree và không hiệu quả với dữ liệu mất cân bằng trừ khi áp dụng kỹ thuật cân bằng mẫu.

### *2.4.6 Ứng dụng*

* Trong ngành tài chính, SVM được sử dụng để dự đoán khả năng vỡ nợ của khách hàng dựa trên các đặc trưng như lịch sử tín dụng, tỷ lệ nợ trên thu nhập, và tài sản đảm bảo, hỗ trợ ngân hàng phê duyệt khoản vay chính xác.
* Thuật toán giúp xây dựng hệ thống chấm điểm tín dụng, xác định các khách hàng rủi ro cao để giảm tỷ lệ nợ xấu.
* Trong xử lý hình ảnh, SVM phân loại các đối tượng dựa trên đặc trưng hình ảnh, tương tự như phân loại khách hàng trong tài chính.
* Trong y tế, SVM dự đoán nguy cơ mắc bệnh dựa trên các chỉ số sức khỏe, có thể áp dụng tương tự cho dự đoán rủi ro tín dụng.[9]

**2.5 Neural Network**

*2.5.1 Giới thiệu về thuật toán Neural Network*  
Neural Network (Mạng nơ-ron nhân tạo) là một thuật toán học máy giám sát mạnh mẽ, lấy cảm hứng từ cấu trúc nơ-ron trong não bộ, được sử dụng cho cả bài toán phân loại và hồi quy . Thuật toán bao gồm các lớp nơ-ron (input layer, hidden layers, output layer) kết nối với nhau thông qua trọng số, xử lý dữ liệu thông qua lan truyền thuận và học thông qua lan truyền ngược. Neural Network đặc biệt hiệu quả trong các bài toán phức tạp như dự đoán khả năng vỡ nợ trong rủi ro tín dụng, nhờ khả năng học các mối quan hệ phi tuyến và xử lý dữ liệu lớn. Trong ngành tài chính, Neural Network được ứng dụng để phân loại khách hàng dựa trên các đặc trưng như lịch sử tín dụng, thu nhập, và tỷ lệ nợ, hỗ trợ ngân hàng đưa ra quyết định tín dụng chính xác và giảm tỷ lệ nợ xấu.[10]

*2.5.2 Nền tảng toán học*  
Neural Network xử lý dữ liệu thông qua các lớp nơ-ron, sử dụng hàm kích hoạt để ánh xạ đầu vào thành đầu ra. Cho tập dữ liệu D = {(x\_i, y\_i) | i=1,...,n}, trong đó xi là vector đặc trưng (k chiều) và yi là nhãn lớp (ví dụ: yi ∈ {0, 1} cho phân loại nhị phân), lan truyền thuận tính đầu ra của một nơ-ron trong lớp l như sau:





Công thức 2.13 (Lan truyền thuận)[11]

Trong đó:

* : Giá trị đầu vào của nơ-ron j trong lớp l trước khi áp dụng hàm kích hoạt.
* : Giá trị đầu ra của nơ-ron j trong lớp l sau khi áp dụng hàm kích hoạt.
* : Trọng số kết nối từ nơ-ron i ở lớp l-1 đến nơ-ron j ở lớp l.
* : Hệ số chặn (bias) của nơ-ron j trong lớp l.
* : Đầu ra của nơ-ron i ở lớp trước (l-1).
* σ(z): Hàm kích hoạt (ví dụ: cho phân loại nhị phân,

hoặc ReLU 

Để huấn luyện mạng, Neural Network tối ưu hóa hàm mất mát (ví dụ: binary cross-entropy cho phân loại nhị phân):



Công thức 2.14 (Hàm mất mát)

Trong đó:

* L(W, b): Hàm mất mát trung bình trên tập dữ liệu, với W là tập hợp tất cả trọng số và b là tất cả bias.
* : Đầu ra dự đoán cho điểm dữ liệu xi (giá trị của nơ-ron đầu ra sau lan truyền thuận).
* yi: Nhãn thực tế của điểm dữ liệu i (0 hoặc 1).
* n: Số lượng điểm dữ liệu trong tập D.

Tối ưu hóa được thực hiện bằng gradient descent, cập nhật trọng số và bias theo đạo hàm riêng của hàm mất mát thông qua lan truyền ngược:





Công thức 2.15 (Lan truyền ngược)[12]

Trong đó:

* α: Tỷ lệ học (learning rate), điều chỉnh tốc độ cập nhật.
* : Đạo hàm riêng của hàm mất mát theo trọng số và bias , được tính bằng quy tắc chuỗi (chain rule) trong lan truyền ngược.
* :=: Toán tử gán, biểu thị cập nhật giá trị mới.

Lan truyền ngược tính gradient từ lớp đầu ra về lớp đầu vào, sử dụng quy tắc chuỗi để cập nhật từng tham số, đảm bảo hàm mất mát hội tụ.

*2.5.3 Diễn giải thuật toán*  
Neural Network hoạt động bằng cách ánh xạ đầu vào qua các lớp nơ-ron để tạo đầu ra, sau đó điều chỉnh trọng số và bias để giảm hàm mất mát. Các bước chính bao gồm:

* Khởi tạo: Gán ngẫu nhiên các trọng số W và bias b, chọn kiến trúc mạng (số lớp, số nơ-ron mỗi lớp, hàm kích hoạt).
* Lan truyền thuận: Với mỗi điểm dữ liệu xi, tính đầu ra bằng cách truyền xi qua các lớp, áp dụng công thức (2.13) tại mỗi nơ-ron.
* Tính mất mát: Sử dụng hàm mất mát (2.14) để đo sai số giữa và yi.
* Lan truyền ngược: Tính gradient của hàm mất mát đối với mỗi trọng số và bias bằng quy tắc chuỗi, sau đó cập nhật tham số theo công thức (2.15).
* Lặp lại: Tiếp tục lan truyền thuận và ngược qua nhiều epoch cho đến khi hàm mất mát hội tụ hoặc đạt số lần lặp tối đa.  
  Neural Network phù hợp với các bài toán rủi ro tín dụng nhờ khả năng học các mô hình phức tạp từ dữ liệu tài chính đa dạng.

*2.5.4 Phân tích độ phức tạp*  
Độ phức tạp của Neural Network phụ thuộc vào số điểm dữ liệu n, số lớp L, số nơ-ron mỗi lớp m\_l, và số epoch E. Trong lan truyền thuận, việc tính đầu ra cho mỗi điểm dữ liệu có độ phức tạp do các phép nhân ma trận. Lan truyền ngược có độ phức tạp tương tự khi tính gradient. Với n điểm dữ liệu và E epoch, độ phức tạp huấn luyện là . Trong giai đoạn dự đoán, độ phức tạp là  cho mỗi điểm dữ liệu. Neural Network đòi hỏi tài nguyên tính toán lớn, đặc biệt với mạng sâu và dữ liệu lớn, nhưng có thể tăng tốc bằng phần cứng chuyên dụng (như GPU).

### *2.5.5 Ưu điểm và hạn chế*

* Ưu điểm: Neural Network có khả năng học các mối quan hệ phi tuyến phức tạp, phù hợp với dữ liệu tài chính đa dạng. Thuật toán xử lý tốt các tập dữ liệu lớn và có thể mở rộng với kiến trúc sâu (deep learning). Trong rủi ro tín dụng, Neural Network có thể xác định các mẫu ẩn trong lịch sử tín dụng và hành vi tài chính.
* Hạn chế: Neural Network yêu cầu tài nguyên tính toán lớn và thời gian huấn luyện lâu. Mô hình khó giải thích (black-box), gây khó khăn trong việc trình bày quy tắc phân loại cho ngân hàng. Thuật toán nhạy cảm với việc chọn siêu tham số (số lớp, số nơ-ron, tỷ lệ học) và dễ bị overfitting nếu không có đủ dữ liệu hoặc kỹ thuật điều chuẩn (regularization).

### *2.5.6 Ứng dụng*

* Trong ngành tài chính, Neural Network được sử dụng để dự đoán khả năng vỡ nợ của khách hàng dựa trên các đặc trưng như lịch sử tín dụng, thu nhập, tỷ lệ nợ, và các yếu tố kinh tế - xã hội, hỗ trợ ngân hàng tối ưu hóa phê duyệt khoản vay.
* Thuật toán giúp xây dựng hệ thống chấm điểm tín dụng, xác định các khách hàng rủi ro cao để giảm tỷ lệ nợ xấu.
* Trong nhận diện hình ảnh, Neural Network phân loại đối tượng dựa trên đặc trưng hình ảnh, tương tự như phân loại khách hàng trong tài chính.
* Trong y tế, thuật toán dự đoán nguy cơ mắc bệnh dựa trên các chỉ số sức khỏe, có thể áp dụng tương tự cho dự đoán rủi ro tín dụng.

# **Chương 3: PHƯƠNG PHÁP THỰC NGHIỆM**

## **3.1 Dữ liệu**

Trong nghiên cứu này, chúng tôi sử dụng bộ dữ liệu khoản vay thực tế loan\_data.csv gồm **37.066 khoản vay**, mỗi khoản được mô tả bởi 13 đặc trưng đầu vào và 1 nhãn đầu ra (not\_fully\_paid).

**Danh sách các đặc trưng:**

* credit\_policy: Chính sách tín dụng (1: đạt yêu cầu, 0: không đạt)
* purpose: Mục đích khoản vay (debt\_consolidation, credit\_card,…)
* int\_rate: Lãi suất (%)
* installment: Khoản trả hàng tháng
* log\_annual\_inc: Log(thu nhập năm)
* dti: Tỷ lệ nợ trên thu nhập
* fico: Điểm FICO
* days\_with\_cr\_line: Số ngày có lịch sử tín dụng
* revol\_bal: Số dư tín dụng quay vòng
* revol\_util: % sử dụng tín dụng quay vòng
* inq\_last\_6mths: Số yêu cầu tín dụng trong 6 tháng
* delinq\_2yrs: Số lần chậm trả trong 2 năm
* pub\_rec: Số hồ sơ công khai bất lợi

**Nhãn:**  
not\_fully\_paid: 1 nếu vỡ nợ, 0 nếu trả đủ

## **3.2 Tiền Xử lý dữ liệu**

Nhìn chung, hầu hết các đặc trưng đầu vào đều là biến số (numeric) dạng liên tục hoặc rời rạc, ngoại trừ purpose là biến phân loại danh mục (categorical). Biến credit\_policy tuy được biểu diễn dưới dạng số 0/1 nhưng thực chất mang ý nghĩa phân loại nhị phân (Boolean). Nhãn not\_fully\_paid là biến phân loại nhị phân biểu thị kết quả như đã mô tả ở trên.

Một điểm cần lưu ý là bộ dữ liệu không chứa giá trị thiếu ở bất kỳ cột nào. Tỷ lệ các khoản vay bị vỡ nợ trong dữ liệu gốc khá thấp so với tổng số (là nhóm thiểu số), cụ thể chỉ chiếm khoảng chưa đến 20% tổng số mẫu. Điều này dẫn đến hiện tượng mất cân bằng lớp (imbalanced classes) mà chúng tôi sẽ đề cập hướng xử lý ở mục 3.5.

Trước khi xây dựng mô hình, dữ liệu thô cần được tiền xử lý để đảm bảo chất lượng và định dạng phù hợp. Do bộ dữ liệu đã được thu thập sẵn và không có giá trị thiếu, các bước tiền xử lý chủ yếu tập trung vào việc mã hóa biến phân loại và làm sạch định dạng dữ liệu nếu cần.

Cụ thể, bộ dữ liệu có một biến dạng phân loại là purpose mô tả mục đích khoản vay với nhiều giá trị danh mục (như liệt kê ở trên). Để thuận tiện cho việc đưa vào mô hình học máy, biến purpose được mã hóa thành dạng số. Thay vì sử dụng one-hot encoding cho tất cả các danh mục, nghiên cứu này tập trung vào mục đích phổ biến nhất là debt\_consolidation (hợp nhất nợ). Do đó, chúng tôi tạo ra một thuộc tính nhị phân mới purpose\_debt\_consolidation, với giá trị 1 nếu purpose của khoản vay là debt\_consolidation, và 0 nếu purpose thuộc các loại khác.

Việc mã hóa nhị phân như vậy giúp đơn giản hóa biến mục đích khoản vay, đồng thời vẫn giữ lại được thông tin về trường hợp mục đích phổ biến nhất. Ngoài ra, do các biến khác đều ở dạng số hoặc nhị phân sẵn, không có ký hiệu hay định dạng đặc biệt, nên không cần thực hiện thêm các bước làm sạch dữ liệu nào khác (ví dụ: loại bỏ trùng lặp, xử lý outlier đặc biệt, v.v.) ở giai đoạn này. Dữ liệu sau khi mã hóa biến purpose sẽ sẵn sàng cho bước chuẩn hóa và tách tập ở các mục tiếp theo.

## **3.3 Chuẩn hóa dữ liệu**

Để các mô hình học máy hoạt động hiệu quả, đặc biệt là với các mô hình sử dụng khoảng cách hoặc gradient (như SVM, mạng nơ-ron), chúng tôi thực hiện chuẩn hóa các đặc trưng đầu vào về cùng một quy mô. Cụ thể, kỹ thuật chuẩn hóa Z-score được áp dụng cho các biến số đầu vào. Phương pháp này (còn được gọi là *StandardScaler*) chuyển đổi giá trị của mỗi đặc trưng thành dạng điểm chuẩn hóa với trung bình = 0 và độ lệch chuẩn = 1.

Công thức chuẩn hóa mỗi giá trị của biến theo Z-score được định nghĩa như sau:

Trong đó:

* : giá trị trung bình của biến trên tập dữ liệu huấn luyện
* : độ lệch chuẩn của biến trên tập dữ liệu huấn luyện

Sau khi chuẩn hóa, các đặc trưng sẽ không còn chênh lệch về đơn vị hay khoảng giá trị, giúp thuật toán học tập hội tụ nhanh hơn và tránh việc đặc trưng có giá trị lớn lấn át đặc trưng có giá trị nhỏ.

Lưu ý rằng việc tính và cho mỗi đặc trưng chỉ được thực hiện trên tập huấn luyện, sau đó áp dụng để chuẩn hóa cho cả tập huấn luyện và tập kiểm tra, nhằm tránh rò rỉ thông tin từ tập kiểm tra vào quá trình huấn luyện.

## **3.****4 Chia tập dữ liệu**

Sau khi tiền xử lý và chuẩn hóa, dữ liệu được chia thành hai phần: tập huấn luyện và tập kiểm tra. Tỷ lệ chia được lựa chọn là 80% cho huấn luyện và 20% cho kiểm tra. Cụ thể, trong 37.066 quan sát, có khoảng 29.653 quan sát được dùng để huấn luyện mô hình, và 7.413 quan sát được giữ lại để đánh giá mô hình sau khi huấn luyện. Việc chia dữ liệu được thực hiện một cách ngẫu nhiên, đồng thời chúng tôi sử dụng tham số random\_state cố định (ví dụ: 42) để đảm bảo có thể tái lập lại kết quả phân chia. Hơn nữa, do dữ liệu có hiện tượng mất cân bằng lớp (số lượng khoản vay vỡ nợ ít hơn hẳn so với không vỡ nợ), chúng tôi áp dụng kỹ thuật chia dữ liệu có stratify theo nhãn. Điều này có nghĩa là tỷ lệ vỡ nợ trong tập huấn luyện và tập kiểm tra gần như tương đương với tỷ lệ vỡ nợ trong toàn bộ dữ liệu, giúp đảm bảo cả hai tập đều có phân bố lớp tương tự nhau. Cách chia có stratify tránh trường hợp tập kiểm tra tình cờ không có (hoặc có quá ít) khoản vay vỡ nợ, gây khó khăn cho việc đánh giá mô hình.

Kết quả của bước này là chúng tôi thu được hai tập dữ liệu: tập huấn luyện (dùng để huấn luyện mô hình) và tập kiểm tra (dùng để đánh giá mô hình cuối cùng). Tập kiểm tra được giữ nguyên và hoàn toàn không được sử dụng trong quá trình huấn luyện hoặc tối ưu siêu tham số, nhằm đảm bảo đánh giá mô hình là khách quan và phản ánh đúng khả năng tổng quát hóa trên dữ liệu chưa thấy.

## **3.5 Cân bằng dữ liệu (SMOTE)**

Như đã đề cập, dữ liệu huấn luyện bị mất cân bằng với tỷ lệ các mẫu dương tính (vỡ nợ) rất thấp so với mẫu âm tính (trả đủ). Sự mất cân bằng này có thể khiến mô hình học tập thiên lệch: mô hình có thể dự đoán tất cả các khoản vay là “không vỡ nợ” và đạt độ chính xác cao, nhưng lại không thực sự học được cách nhận diện các trường hợp vỡ nợ (do quá ít mẫu vỡ nợ để học).

Để khắc phục, chúng tôi áp dụng kỹ thuật **SMOTE** (viết tắt của *Synthetic Minority Over-sampling Technique*) trên tập huấn luyện nhằm cân bằng dữ liệu giữa hai lớp. SMOTE là phương pháp tăng cường mẫu thiểu số bằng cách tổng hợp các điểm dữ liệu mới cho lớp thiểu số (ở đây là lớp vỡ nợ) thay vì chỉ nhân bản ngẫu nhiên các mẫu cũ.

Cụ thể, SMOTE sẽ chọn một mẫu thiểu số hiện có, sau đó tìm *k* láng giềng gần nhất của nó trong không gian đặc trưng, và tạo ra các điểm dữ liệu mới nằm giữa mẫu đó với một trong những láng giềng gần nhất được chọn ngẫu nhiên.

Quá trình này lặp lại cho đến khi số lượng mẫu thiểu số được tăng lên đáng kể, cân bằng với số lượng mẫu đa số. Nói cách khác, thay vì lặp lại các trường hợp vỡ nợ đã có (dễ gây *overfitting*), SMOTE sinh ra các trường hợp vỡ nợ giả lập bằng cách nội suy đặc trưng của các trường hợp gần giống nhau, giúp mở rộng không gian mẫu dương một cách hợp lý.

Trong nghiên cứu này, chúng tôi sử dụng SMOTE để tăng kích thước lớp vỡ nợ trong tập huấn luyện cho đến khi số mẫu vỡ nợ xấp xỉ bằng số mẫu không vỡ nợ. Sau khi áp dụng, tập huấn luyện trở nên cân bằng: mỗi lớp đều có số lượng quan sát xấp xỉ ngang nhau.

**Lưu ý:** SMOTE chỉ được áp dụng trên tập huấn luyện. Tập kiểm tra vẫn giữ nguyên phân bố ban đầu (mất cân bằng) để phản ánh đúng phân bố thực tế và để việc đánh giá mô hình được khách quan.

Nhờ việc cân bằng dữ liệu bằng SMOTE, mô hình huấn luyện sẽ ít bị thiên lệch và có cơ hội học tốt hơn các mẫu thuộc lớp vỡ nợ (vốn là quan tâm chính trong bài toán này).

## **3.6 Xây dựng mô hình**

Sau khi chuẩn bị dữ liệu, chúng tôi tiến hành xây dựng và huấn luyện các mô hình dự báo vỡ nợ. Trong nghiên cứu này, chúng tôi triển khai một bộ mô hình đa dạng bao gồm cả mô hình tuyến tính đơn giản, mô hình cây quyết định giải thích được, mô hình ensemble, mô hình máy học phân biệt tuyến tính kernel, và mô hình học sâu.

Việc thử nghiệm nhiều thuật toán khác nhau cho phép so sánh hiệu quả dự báo và lựa chọn mô hình phù hợp nhất.

Danh sách các mô hình được xây dựng gồm:

* **Hồi quy Logistic** – *Logistic Regression* (với regularization L2, solver liblinear).
* **Cây quyết định** – *Decision Tree* (giới hạn độ sâu cây max\_depth = 6).
* **Rừng ngẫu nhiên** – *Random Forest* (gồm 100 cây quyết định, n\_estimators = 100).
* **Máy Vector Hỗ trợ (SVM)** – *Support Vector Machine* (thử nghiệm với kernel tuyến tính và kernel Gaussian RBF).
* **Mạng nơ-ron nhân tạo** – *Neural Network* (mạng truyền thẳng 2 lớp ẩn, kích hoạt ReLU và sigmoid, huấn luyện bằng Keras – optimizer Adam, 20 epoch, batch size 32).

Dưới đây là mô tả chi tiết và cấu hình của từng mô hình:

### *3.6.1 Hồi quy logistic*

Hồi quy logistic là mô hình phân loại tuyến tính phổ biến, sử dụng hàm **sigmoid (logistic)** để ước tính xác suất một quan sát thuộc về lớp “vỡ nợ” (nhãn 1). Mô hình này giả định mối quan hệ tuyến tính giữa các biến đầu vào và logit của xác suất kết quả.

Hệ số đầu ra của hồi quy logistic dễ diễn giải, cho biết mức độ ảnh hưởng của mỗi đặc trưng đến xác suất vỡ nợ.

Trong thực nghiệm, chúng tôi sử dụng biến thể logistic với:

* **Regularization L2 (Ridge)** để tránh overfitting
* **Solver: liblinear** – thuật toán tối ưu phù hợp cho bài toán phân loại nhị phân với L2 trên tập dữ liệu cỡ vừa
* **Các tham số khác** sử dụng giá trị mặc định trong thư viện scikit-learn

### *3.6.2 Cây quyết định*

Cây quyết định là một mô hình phân loại dựa trên cấu trúc cây nhị phân, trong đó dữ liệu được phân tách theo các điều kiện trên thuộc tính. Mô hình này có ưu điểm là dễ hiểu và giải thích: ta có thể trực quan hóa cây để thấy được logic ra quyết định dựa trên các đặc trưng.

Tuy nhiên, cây quyết định đơn lẻ có thể dễ bị overfit nếu không được tỉa (*prune*) hoặc giới hạn độ phức tạp. Do đó, chúng tôi giới hạn độ sâu tối đa của cây ở mức 6 (max\_depth = 6) nhằm kiểm soát độ phức tạp, ngăn việc cây phân chia quá chi tiết theo dữ liệu huấn luyện.

Tiêu chí chọn điểm chia tại mỗi nút là chỉ số Gini (mặc định trong scikit-learn). Các tham số khác giữ mặc định.

Mô hình Decision Tree được huấn luyện trên tập huấn luyện đã cân bằng (sau SMOTE). Với cấu hình này, cây quyết định có thể học được các quy tắc phân loại tổng quát, đồng thời tránh được việc quá khớp với nhiễu của dữ liệu.

### *3.6.3 Random Forest*

Rừng ngẫu nhiên (*Random Forest*) là mô hình ensemble (tập hợp) gồm nhiều cây quyết định, giúp cải thiện độ chính xác và tính tổng quát so với một cây đơn lẻ.

Ý tưởng của Random Forest là xây dựng nhiều cây quyết định độc lập trên các mẫu dữ liệu con (được lấy ngẫu nhiên có hoàn lại từ tập huấn luyện – *bootstrap*) và chỉ sử dụng một tập ngẫu nhiên các đặc trưng tại mỗi lần chia nhánh.

Cách làm này giúp các cây trong rừng đa dạng và ít tương quan với nhau, từ đó khi lấy trung bình phiếu bầu (hoặc đa số phiếu) của các cây, sai số chung sẽ giảm đi (giảm phương sai).

Trong triển khai, chúng tôi sử dụng 100 cây quyết định (n\_estimators = 100) trong rừng – số lượng cây đủ lớn để mô hình hội tụ ổn định. Mỗi cây được giới hạn độ sâu tự động tùy theo dữ liệu (hoặc có thể sử dụng tham số mặc định của thư viện, thường cho phép cây phát triển cho đến khi lá thuần nhất hoặc số mẫu lá tối thiểu).

* Tiêu chí chia nhánh: Gini (mặc định)
* Tập dữ liệu: Đã cân bằng bằng SMOTE

Mô hình này kỳ vọng sẽ cho độ chính xác caovàkhả năng tổng quát tốt, do kết hợp được sức mạnh của nhiều mô hình cơ bản.

Tuy nhiên, nhược điểm là khó diễn giải hơn so với một cây đơn lẻ, và thời gian huấn luyện – dự đoán cũng lâu hơn (nhưng với 100 cây và dữ liệu ~30.000 mẫu thì vẫn chấp nhận được).

### *3.6.**4 Support Vector Machine (SVM)*

Support Vector Machine (*SVM*) là một thuật toán học máy giám sát mạnh mẽ cho bài toán phân loại, đặc biệt hiệu quả trong các không gian đặc trưng có kích thước lớn. SVM tìm kiếm một siêu phẳng (hyperplane) tối ưu để phân tách hai lớp dữ liệu, sao cho khoảng cách biên (margin) giữa các điểm dữ liệu gần nhất của hai lớp và siêu phẳng là lớn nhất.

Trong nghiên cứu này, chúng tôi thử nghiệm hai biến thể **SVM** vớihàm kernel khác nhau:

* **SVM tuyến tính**: sử dụng kernel tuyến tính (*linear*), tương đương với việc tìm một hyperplane trong không gian ban đầu. Mô hình này đơn giản và hiệu quả khi dữ liệu có thể phân tách tuyến tính (sau khi đã chuẩn hóa và biến đổi thích hợp).
* **SVM phi tuyến (kernel RBF)**: sử dụng hàm kernel Gauss RBF (*Radial Basis Function*) để ánh xạ dữ liệu vào không gian đặc trưng phi tuyến cao hơn, cho phép phân tách những trường hợp mà trong không gian gốc không tuyến tính được. Kernel RBF thường hiệu quả trên nhiều dạng dữ liệu nhưng cần chọn tham số gamma phù hợp.

Cả hai mô hình SVM đều được huấn luyện với dữ liệu đã chuẩn hóa **Z-score** (điều kiện tiên quyết để SVM tối ưu tốt). Chúng tôi sử dụng các tham số **SVM** mặc định của scikit-learn (ví dụ C = 1.0 cho độ mềm của margin).

Mục đích của việc huấn luyện cả **SVM tuyến tính và SVM RBF** là để so sánh:

* Nếu dữ liệu không quá phi tuyến, **SVM** tuyến tính có thể đủ tốt
* Ngược lại, **SVM** với **kernel RBF** có thể nắm bắt quan hệ phức tạp hơn giữa các biến

Tuy nhiên, **SVM** trên tập dữ liệu lớn có thể đòi hỏi thời gian tính toán cao, do đó chúng tôi cần cân nhắc giữa độ chính xác và hiệu năng khi so sánh kết quả.

### 3.6.5 Mạng nơ-ron nhân tạo (Neural Network)

Mạng nơ-ron nhân tạo (ANN) là một mô hình học sâu (deep learning) phổ biến, mô phỏng cấu trúc các tế bào thần kinh liên kết.  
Trong bài toán này, chúng tôi xây dựng một mạng nơ-ron dạng feed-forward (mạng truyền thẳng) với 2 lớp ẩn để dự đoán khả năng vỡ nợ.

Cấu trúc tổng quát của mạng như sau:

* Lớp đầu vào gồm 13 neuron tương ứng với 13 đặc trưng đã chuẩn hóa.
* Tiếp đó là 2 lớp ẩn.
* Cuối cùng là một neuron đầu ra cho dự đoán nhãn (0/1).

Các lớp ẩn sử dụng hàm kích hoạt ReLU (Rectified Linear Unit) – đây là hàm kích hoạt phi tuyến phổ biến, giúp mạng học được các quan hệ phức tạp và giảm thiểu hiện tượng vanishing gradient trong quá trình huấn luyện.  
Lớp đầu ra dùng hàm sigmoid để đưa ra xác suất của lớp “vỡ nợ” (giá trị trong khoảng 0 đến 1).  
Quyết định dùng sigmoid ở đầu ra phù hợp với bài toán nhị phân và cho phép chúng tôi ngưỡng xác suất để phân loại.

Mô hình mạng nơ-ron được hiện thực hóa bằng thư viện Keras (chạy trên backend TensorFlow).  
Hàm mất mát sử dụng để huấn luyện là binary cross-entropy (độ lệch chuẩn binary), phù hợp cho bài toán phân loại nhị phân.  
Quá trình tối ưu trọng số sử dụng thuật toán Adam (Adaptive Moment Estimation) – một phương pháp gradient descent tối ưu hiện đại, tự điều chỉnh tốc độ học (learning rate) cho mỗi tham số.

Chúng tôi huấn luyện mạng trong 20 epoch (20 lượt qua toàn bộ dữ liệu huấn luyện) với batch size = 32.  
Với kích thước dữ liệu huấn luyện (~30k mẫu) và batch 32, mỗi epoch sẽ gồm khoảng 926 bước cập nhật.  
Sau 20 epoch, mạng thường đã hội tụ hoặc cải thiện chậm, do đó chúng tôi giới hạn tại đó để tránh overfitting.

Trong quá trình huấn luyện, chúng tôi có thể tách một phần nhỏ dữ liệu huấn luyện làm tập validation để theo dõi hiệu năng và áp dụng kỹ thuật dừng sớm (early stopping) nếu cần, tuy nhiên chi tiết này tùy thuộc quá trình tinh chỉnh mô hình.

Kết quả cuối cùng của mô hình mạng nơ-ron sẽ được so sánh với các mô hình truyền thống ở trên để đánh giá liệu mô hình học sâu có mang lại lợi ích đáng kể cho bài toán dự báo vỡ nợ hay không.

## **3.7 Đánh giá mô hình**

Để đánh giá hiệu quả của các mô hình đã xây dựng, chúng tôi áp dụng các thước đo đánh giá phổ biến cho bài toán phân loại nhị phân, bao gồm: **Accuracy**, **Precision**, **Recall** và **F1-score**.

Các chỉ số này được tính dựa trên ma trận kết quả **(confusion matrix)** của mô hình trên **tập kiểm tra** (20% dữ liệu chưa được nhìn thấy trong quá trình huấn luyện).  
Dưới đây là định nghĩa ngắn gọn của từng chỉ số:

* **Accuracy (Độ chính xác)**: Tỷ lệ dự đoán đúng trên tổng số mẫu kiểm tra.  
  Công thức:
* Trong đó:
  + TP: True Positive – dự đoán vỡ nợ và thực tế vỡ nợ
  + TN: True Negative – dự đoán không vỡ nợ và thực tế không vỡ nợ
  + FP: False Positive – dự đoán vỡ nợ nhưng thực tế trả đủ
  + FN: False Negative – dự đoán không vỡ nợ nhưng thực tế bị vỡ nợ
* **Precision (Độ chính xác dự đoán dương)**: Tỷ lệ mẫu dự đoán vỡ nợ (dương tính) thực sự đúng là vỡ nợ.
* Precision cao nghĩa là ít báo động nhầm.
* **Recall (Độ nhạy, TPR)**: Tỷ lệ mẫu vỡ nợ thực sự được mô hình dự đoán đúng.
* Recall cao nghĩa là mô hình ít bỏ sót các trường hợp vỡ nợ.
* **F1-score**: Trung bình điều hòa của Precision và Recall.
* F1-score cung cấp một chỉ số cân bằng, hữu ích khi muốn đánh giá tổng quát hơn trong trường hợp dữ liệu mất cân bằng.

Sau khi huấn luyện, chúng tôi sẽ áp dụng mỗi mô hình trên tập kiểm tra để tạo dự đoán, sau đó tính các chỉ số trên.  
Kết quả các mô hình sẽ được so sánh dựa trên những thước đo này.

Đáng chú ý, trong bài toán dự báo rủi ro tín dụng, việc đánh đổi giữa Precision và Recall cần được cân nhắc kỹ.

Thông thường, Recall được coi là quan trọng hơn Precision trong ngữ cảnh này.

Lý do là một false negative (FN) – tức một khoản vay thực tế sẽ vỡ nợ nhưng mô hình dự đoán nhầm là “an toàn” – có thể gây thiệt hại nghiêm trọng (ngân hàng cấp khoản vay mà không thu hồi được nợ, mất vốn).

Ngược lại, một false positive (FP) – khoản vay thực tế trả đủ nhưng bị mô hình dự đoán nhầm là sẽ vỡ nợ và có thể bị từ chối – tuy gây mất cơ hội hoặc làm khách hàng không hài lòng, nhưng không gây thiệt hại trực tiếp lớn bằng việc mất vốn.

Do đó, việc ưu tiên Recall (tức cố gắng phát hiện đầy đủ các trường hợp rủi ro) giúp tổ chức tín dụng giảm thiểu rủi ro và bảo vệ tài sản tốt hơn.

Nói cách khác, mô hình dự báo vỡ nợ nên hạn chế bỏ sót các trường hợp vỡ nợ (tăng Recall), dù có thể phải chấp nhận một tỷ lệ nào đó các báo động giả (giảm Precision).

Tất nhiên, chúng tôi vẫn xem xét cả hai chỉ số cùng với F1-score để có cái nhìn toàn diện, nhưng Recallsẽ được nhấn mạnh hơn trong việc lựa chọn mô hình tối ưu cho bài toán.

# Chương 4: KẾT QUẢ VÀ THẢO LUẬN

## **4.1 Kết quả thực nghiệm**

Bảng tổng hợp kết quả:

| Thuật toán | Accuracy | Precision | Recall | F1-Score |
| --- | --- | --- | --- | --- |
| Logistic Regression | 0.82 | 0.88 | 0.87 | 0.87 |
| Decision Tree | 0.73 | 0.82 | 0.80 | 0.81 |
| Random Forest | 0.85 | 0.88 | 0.90 | 0.89 |
| SVM (RBF Kernel) | 0.82 | 0.86 | 0.90 | 0.88 |
| SVM (Linear Kernel) | 0.82 | 0.84 | 0.92 | 0.88 |
| Neural Network (Keras) | 0.84 | 0.88 | 0.90 | 0.89 |

Nhìn vào bảng kết quả, có thể thấy:

* Random Forest đạt độ chính xác (Accuracy) cao nhất, đồng thời có F1-Score = 0.89 – cao nhất trong tất cả các mô hình.
* Neural Network (mạng nơ-ron) cũng cho kết quả rất tốt với F1-Score = 0.89, tương đương Random Forest, và Accuracy = 0.84.
* SVM: cả hai phiên bản SVM tuyến tính và SVM RBF đều đạt F1-Score = 0.88, chỉ kém nhẹ so với Random Forest và ANN. Đặc biệt, SVM tuyến tính có Recall cao nhất (0.92) trong các mô hình

– mô hình này nhận diện được nhiều nhất các trường hợp vỡ nợ, dù Precision của nó chỉ 0.84.

* Logistic Regression đạt F1-Score = 0.87 – một kết quả khá tốt đối với mô hình tuyến tính đơn giản, cho thấy hồi quy logistic vẫn là baseline mạnh. Precision của logistic cao (0.88) nhưng recall hơi thấp hơn (0.87) so với các mô hình phức tạp hơn.
* Decision Tree có hiệu năng thấp nhất trong các mô hình: Accuracy chỉ 0.73 và F1-Score = 0.81. Mô hình cây đơn cho thấy hạn chế khi so sánh với các mô hình khác, dù Precision không quá tệ (0.82) nhưng Recall thấp nhất (0.80), nghĩa là bỏ sót khá nhiều trường hợp vỡ nợ.

## **4.2 So sánh hiệu năng các mô hình**

Dựa trên kết quả trên, chúng tôi phân tích cụ thể ưu nhược điểm của từng mô hình trong bối cảnh bài toán:

**Random Forest**: Đây là mô hình có hiệu năng cao nhất trên tập kiểm tra, với Accuracy đạt 0.85 và F1-Score 0.89. Random Forest đạt sự cân bằng tốt giữa Precision (0.88) và Recall (0.90), cho thấy vừa hạn chế được báo động giả, vừa phát hiện hầu hết các khoản vay rủi ro. Ưu điểm của Random Forest là khả năng giảm overfitting so với cây quyết định đơn, đồng thời tận dụng được sức mạnh kết hợp của nhiều mô hình. Kết quả này khẳng định Random Forest rất phù hợp cho bài toán, khi dữ liệu có nhiều đặc trưng phức tạp mà một cây đơn khó nắm bắt hết.

**Neural Network (Keras)**: Mạng nơ-ron đạt F1-Score 0.89, ngang ngửa với Random Forest, và Recall cao (0.90) – nghĩa là mô hình rất nhạy trong việc phát hiện khoản vay xấu. Điều này cho thấy với dữ liệu đã được xử lý tốt (chuẩn hóa, cân bằng), mạng nơ-ron có thể học được các quan hệ phức tạp ẩn trong dữ liệu. Tuy nhiên, nhược điểm là mô hình hoạt động như một “hộp đen”, rất khó giải thích tại sao một khoản vay bị dự đoán vỡ nợ (mạng không cung cấp lý do rõ ràng như cây quyết định hoặc trọng số đặc trưng như logistic).

**Support Vector Machine (SVM)**: Cả hai phiên bản SVM đều cho F1-Score ~0.88. SVM với kernel phi tuyến (RBF) xử lý tốt biên phân cách phi tuyến, trong khi SVM tuyến tính lại đạt Recall cao nhất (0.92) trong số các mô hình. Điều này gợi ý rằng SVM tuyến tính cực kỳ hữu ích nếu mục tiêu đặt ra là giảm thiểu tối đa việc bỏ sót các khoản vay rủi ro (coi trọng Recall). Tuy nhiên, cái giá phải trả là Precision của SVM tuyến tính thấp nhất (0.84), nghĩa là cảnh báo nhầm nhiều hơn các mô hình khác. Ngoài ra, trong quá trình huấn luyện, SVM (đặc biệt là RBF) mất nhiều thời gian hơn so với các mô hình còn lại do tập dữ liệu khá lớn.

**Decision Tree**: Đây là mô hình kém hiệu quả nhất trong thí nghiệm, với Accuracy chỉ 0.73 và F1-Score 0.81. Precision của cây quyết định tương đối cao (0.82) – tức mô hình đưa ra các dự báo vỡ nợ khá thận trọng, ít nhầm lẫn – nhưng Recall thấp (0.80) cho thấy cây bỏ sót nhiều khoản vay xấu. Mô hình cây có ưu điểm về mặt dễ hiểu (có thể trích xuất các quy tắc if-then đơn giản), nhưng nhược điểm cố hữu là dễ overfit và thiếu ổn định. Trong thí nghiệm, chúng tôi có giới hạn độ sâu cây, nhưng có lẽ do bản chất dữ liệu phức tạp, một cây duy nhất không đủ sức mô tả, dẫn đến hiệu năng thấp nhất.

**Logistic Regression**: Mặc dù là mô hình đơn giản nhất, Logistic Regression cho kết quả khá tốt (F1 = 0.87) – chỉ thấp hơn khoảng 0.02 so với Random Forest và ANN. Precision đạt 0.88 và Recall 0.87, cho thấy mô hình logistic đưa ra quyết định khá cân bằng giữa hai loại lỗi. Điều này có nghĩa là logistic có xu hướng thận trọng (độ chính xác cao) nhưng đồng thời vẫn tìm được phần lớn các trường hợp vỡ nợ. Tuy nhiên, do giả định tuyến tính, mô hình này có thể bỏ lỡ các quan hệ phi tuyến giữa đặc trưng và khả năng vỡ nợ. Logistic Regression tỏ ra phù hợp nếu ưu tiên một mô hình đơn giản, dễ triển khai và giải thích trực quan (ví dụ: trong môi trường yêu cầu giải trình rõ ràng, logistic có lợi thế về interpretability).

Tóm lại, Random Forest được đánh giá là mô hình tốt nhất trong bài toán này, xét trên cả hiệu năng dự báo và tính ổn định. Neural Network và SVM cũng cho kết quả rất cao, chứng tỏ những phương pháp hiện đại này có thể bắt kịp Random Forest khi dữ liệu được xử lý phù hợp. Logistic Regression vượt kỳ vọng khi vẫn giữ hiệu năng khá dù đơn giản. Decision Tree cá biệt cho thấy hiệu quả kém, củng cố quan điểm rằng việc dùng một mô hình ensemble như Random Forest sẽ cải thiện đáng kể so với dùng một cây đơn lẻ.

# **Chương 5: KẾT LUẬN VÀ KIẾN NGHỊ**

## **5.1 Kết luận**

Trong nghiên cứu này, chúng tôi đã triển khai và so sánh hiệu năng của sáu thuật toán học máy – Logistic Regression, Decision Tree, Random Forest, Support Vector Machine (kernel tuyến tính và kernel RBF), và Neural Network – trong nhiệm vụ phân loại các khoản vay có khả năng không được hoàn trả đầy đủ (vỡ nợ). Dữ liệu thực nghiệm gồm 37 nghìn khoản vay với đầy đủ các thông tin đặc trưng về người vay và khoản vay. Các bước tiền xử lý như chuẩn hóa đặc trưng và dùng SMOTE để cân bằng lớp đã được áp dụng nhằm đảm bảo điều kiện huấn luyện tốt nhất cho các mô hình.

Kết quả thực nghiệm cho thấy **Random Forest** là mô hình hoạt động hiệu quả nhất trên tập dữ liệu kiểm tra, đạt Accuracy 0.85 và F1-Score 0.89. Đây cũng là mô hình cân bằng tốt nhất giữa Precision (0.88) và Recall (0.90), tức vừa hạn chế cảnh báo nhầm vừa phát hiện hầu hết các khoản vay rủi ro.

**Neural Network (Keras)** đạt F1-Score tương đương (0.89), với khả năng phát hiện khoản vay xấu rất mạnh (Recall 0.90), chứng tỏ hiệu quả của mô hình này trong việc xử lý các quan hệ phi tuyến phức tạp trong dữ liệu.

**SVM** (cả kernel tuyến tính và RBF) cũng hoạt động tốt với F1-Score 0.88 cho cả hai trường hợp. Đặc biệt, **SVM tuyến tính** đạt Recall cao nhất (0.92), phù hợp nếu mục tiêu là giảm thiểu tối đa việc bỏ sót các khoản vay rủi ro.

**Logistic Regression** cho kết quả khá ổn (F1-Score 0.87) đối với một mô hình tuyến tính đơn giản, tuy nhiên mô hình này có giới hạn khi dữ liệu mang tính phi tuyến rõ rệt.

**Decision Tree** có hiệu năng thấp nhất (F1-Score 0.81) và dễ bị overfit nếu không được kiểm soát độ sâu, điều này phù hợp với kỳ vọng do mô hình cây đơn lẻ kém mạnh mẽ hơn so với các phương pháp khác.

Lý do chọn Random Forest làm mô hình tối ưu:

* **Random Forest** kết hợp nhiều cây quyết định và sử dụng kỹ thuật lấy mẫu ngẫu nhiên (bagging), giúp giảm phương sai, tăng tính ổn định của dự báo và hạn chế overfitting so với một cây đơn lẻ.
* Mô hình không đòi hỏi giả định tuyến tính giữa các biến đầu vào và đầu ra, do đó phù hợp với dữ liệu phi tuyến – đặc biệt trong trường hợp quan hệ giữa các đặc trưng tín dụng và rủi ro vỡ nợ có thể rất phức tạp, phi tuyến tính.
* **Random Forest** duy trì hiệu năng tốt kể cả khi có nhiều đặc trưng hoặc dữ liệu nhiễu, nhờ cơ chế trung bình hóa kết quả của nhiều mô hình con (mỗi cây quyết định chỉ “nhìn” một phần dữ liệu và tập đặc trưng).
* Mặc dù là mô hình ensemble “hộp đen”, Random Forest vẫn có thể cung cấp một số thông tin giải thích, chẳng hạn như mức độ quan trọng của từng đặc trưng (feature importance) thông qua việc đo lường giảm độ lỗi do mỗi đặc trưng mang lại.

Từ những kết quả và phân tích trên, có thể kết luận rằng Random Forest nổi trội hơn các mô hình còn lại cho bài toán phân loại rủi ro khoản vay trong phạm vi dữ liệu và điều kiện thí nghiệm đã thực hiện. Mô hình mạng nơ-ron và SVM cũng chứng tỏ hiệu quả cao, nhưng xét về độ ổn định và dễ sử dụng, Random Forest là lựa chọn phù hợp và cân bằng nhất.

## **5.2 Kiến nghị**

Dựa trên kết quả nghiên cứu, chúng tôi đưa ra một số kiến nghị như sau:

* **Ứng dụng mô hình trong thực tế**: Đối với các hệ thống chấm điểm tín dụng hoặc quản trị rủi ro khoản vay, nên ưu tiên triển khai mô hình Random Forest hoặc Neural Network nhằm đạt độ chính xác phân loại cao và ổn định. Hai mô hình này có khả năng nhận diện tốt các khoản vay rủi ro, giúp tổ chức tín dụng có biện pháp phòng ngừa kịp thời (ví dụ: thắt chặt điều kiện cho vay đối với hồ sơ có rủi ro cao, yêu cầu tài sản đảm bảo, v.v.).
* **Lựa chọn mô hình tùy tình huống**: Trong trường hợp yêu cầu hệ thống triển khai nhanh và có khả năng giải thích trực quan (ví dụ: môi trường đòi hỏi tuân thủ giải trình với quy định), Logistic Regression vẫn là lựa chọn phù hợp. Mặc dù độ chính xác không cao bằng các mô hình phức tạp, logistic đơn giản, dễ hiểu và dễ bảo trì. Tương tự, nếu mục tiêu kinh doanh đặt nặng việc không bỏ sót khách hàng rủi ro, có thể cân nhắc sử dụng SVM tuyến tính do mô hình này cho Recall rất cao (đổi lại cần chấp nhận một tỷ lệ báo động nhầm nhất định).
* **Mở rộng nghiên cứu**: Nghiên cứu có thể được mở rộng bằng cách thử thêm các mô hình nâng cao như thuật toán Gradient Boosting (ví dụ: XGBoost, LightGBM), vốn nổi tiếng với hiệu năng cao trong các cuộc thi phân tích dữ liệu. Những mô hình boosting có thể khắc phục một số hạn chế của Random Forest bằng cách tập trung vào các mẫu khó. Bên cạnh đó, việc tinh chỉnh siêu tham số (hyperparameter tuning) cho các mô hình hiện tại cũng là hướng quan trọng để cải thiện thêm hiệu năng – chẳng hạn tối ưu số lượng cây và độ sâu trong Random Forest, hoặc kiến trúc mạng nơ-ron.
* **Bổ sung dữ liệu và đặc trưng**: Để tăng cường sức mạnh dự báo, có thể xem xét tích hợp thêm các đặc trưng mới hoặc nguồn dữ liệu khác vào mô hình. Ví dụ: dữ liệu kinh tế vĩ mô (tình hình thị trường, lãi suất chung), thông tin định tính về người vay (mức độ uy tín, lịch sử giao dịch khác) có thể ảnh hưởng đến xác suất vỡ nợ. Việc kết hợp các yếu tố này (nếu thu thập được) có thể giúp mô hình toàn diện hơn.
* **Cân nhắc giữa Precision và Recall**: Tùy theo chiến lược quản trị rủi ro của tổ chức, có thể ưu tiên mô hình với Precision cao (ít báo động giả, tránh từ chối oan các khách hàng tốt) hoặc Recall cao (ưu tiên phát hiện tối đa rủi ro, chấp nhận từ chối nhầm một số khách hàng). Ví dụ, nếu tổn thất từ một khoản vay vỡ nợ lớn hơn nhiều so với lợi nhuận bỏ lỡ từ một khoản vay bị từ chối nhầm, thì nên ưu tiên mô hình có Recall cao. Ngược lại, nếu muốn mở rộng danh mục cho vay và chấp nhận rủi ro hơn, mô hình có Precision cao sẽ phù hợp.

**Tóm lại**, nghiên cứu đã cho thấy tiềm năng của các thuật toán ML trong việc dự đoán rủi ro tín dụng. Random Forest nổi lên như một giải pháp hứa hẹn với hiệu suất cao và dễ triển khai. Để tiếp tục nâng cao kết quả, các nhà nghiên cứu và thực hành có thể thử nghiệm các kỹ thuật tiên tiến hơn và bổ sung phạm vi dữ liệu lớn hơn. Việc lựa chọn thuật toán ML phù hợp cần dựa vào mục tiêu bài toán (ưu tiên Recall, Precision hay F1), tính chất dữ liệu, yêu cầu diễn giải và khả năng triển khai trong môi trường thực tế.

# **TÀI LIỆU THAM KHẢO**

[1] “Ví dụ Logistic regression ứng dụng trong ngân hàng (P.3) – Big Data Uni.” Accessed: May 24, 2025. [Online]. Available: https://bigdatauni.com/tin-tuc/vi-du-logistic-regression-ung-dung-trong-ngan-hang-p-3.html

[2] “Decision Tree algorithm — Machine Learning cho dữ liệu dạng bảng.” Accessed: May 24, 2025. [Online]. Available: https://machinelearningcoban.com/tabml\_book/ch\_model/decision\_tree.html

[3] T. Vu, “Bài 34: Decision Trees (1): Iterative Dichotomiser 3,” Tiep Vu’s blog. Accessed: May 25, 2025. [Online]. Available: https://machinelearningcoban.com/2018/01/14/id3/

[4] “Information gain (decision tree),” *Wikipedia*. Dec. 17, 2024. Accessed: May 25, 2025. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Information\_gain\_(decision\_tree)&oldid=1263575143

[5] admin, “Thuật toán Random Forest: Giải thích chi tiết và ứng dụng,” AIcandy. Accessed: May 25, 2025. [Online]. Available: https://aicandy.vn/thuat-toan-random-forest-giai-thich-chi-tiet-va-ung-dung/

[6] “Decision Tree là gì? Lý do sử dụng & Ứng dụng cây quyết định.” Accessed: May 24, 2025. [Online]. Available: https://interdata.vn/blog/decision-tree-la-gi/

[7] T. Vu, “Bài 19: Support Vector Machine,” Tiep Vu’s blog. Accessed: May 24, 2025. [Online]. Available: https://machinelearningcoban.com/2017/04/09/smv/

[8] “Sử dụng Hard Margin so với Soft Margin trong SVM | Baeldung về Khoa học Máy tính.” Accessed: May 25, 2025. [Online]. Available: https://www.baeldung.com/cs/svm-hard-margin-vs-soft-margin

[9] “Giới thiệu về Support Vector Machine (SVM).” Accessed: May 24, 2025. [Online]. Available: https://viblo.asia/p/gioi-thieu-ve-support-vector-machine-svm-6J3ZgPVElmB

[10] nttuan8, “Bài 3: Neural network,” Deep Learning cơ bản. Accessed: May 24, 2025. [Online]. Available: https://nttuan8.com/bai-3-neural-network/

[11] “Ly thuyet mang neural - Lý Thuyết Mạng Nơron Tổng quan về mạng nơron Mạng nơron nhân tạo Mạng noron,” Studocu. Accessed: May 25, 2025. [Online]. Available: https://www.studocu.vn/vn/document/dai-hoc-bach-khoa-tphcm/phap-luat-viet-nam-dai-cuong/ly-thuyet-mang-neural/81167909

[12] “4.7. Lan truyền xuôi, Lan truyền ngược và Đồ thị tính toán — Đắm mình vào Học Sâu 0.14.4 documentation.” Accessed: May 25, 2025. [Online]. Available: https://d2l.aivivn.com/chapter\_multilayer-perceptrons/backprop\_vn.html

# **SOURCE CODE**

https://github.com/Than22012004/DACS.git