**MỤC LỤC**

[**LỜI CẢM ƠN** 7](#_Toc43449594)

[**MỞ ĐẦU** 8](#_Toc43449595)

[**CHƯƠNG I: THUẬT TOÁN PHÂN LỚP NHỊ PHÂN LOGISTIC REGRESSION** 9](#_Toc43449596)

[**1.** **Học máy (Machine Learning)** 9](#_Toc43449597)

[**1.1.** **Giới thiệu** 9](#_Toc43449598)

[**1.2.** **Bài toán phân cụm** 10](#_Toc43449599)

[**1.3.** **Bài toán phân lớp** 10](#_Toc43449600)

[**2.** **Thuật toán phân lớp nhị phân Logistic Regression** 15](#_Toc43449601)

[***2.1.*** **Giới thiệu** 15](#_Toc43449602)

[***2.2.*** **Ví dụ mô hình LogisticRegreession** 15](#_Toc43449603)

[***2.3.*** **Mô hình Logistic Regression** 17](#_Toc43449604)

[**CHƯƠNG II: ÁP DỤNG THUẬT TOÁN LOGISTIC REGREESION CHO BÀI TOÁN PHÂN LỚP ỨNG VIÊN** 22](#_Toc43449605)

[**1.** **Bài toán phân lớp ứng viên xét tuyển vào trường đại học** 22](#_Toc43449606)

[**1.1.** **Giới thiệu** 22](#_Toc43449607)

[**1.2.** **Mô hình bài toán** 22](#_Toc43449608)

[**2.** **Môi trường thử nghiệm** 23](#_Toc43449609)

[**2.1.** **Cài đặt python.** 23](#_Toc43449610)

[**2.2.** **Cài đặt pycharm.** 27](#_Toc43449611)

[**2.3.** **Cài đặt bộ thư viện** 31](#_Toc43449612)

[**a.** **NumPy** 31](#_Toc43449613)

[**b.** **SciPy** 31](#_Toc43449614)

[**c.** **Matplotlib** 31](#_Toc43449615)

[**d.** **Scikit-learn** 31](#_Toc43449616)

[**3.** **Cài đặt** 32](#_Toc43449617)

[**3.1.** **Bộ dữ liệu** 32](#_Toc43449618)

[**3.2.** **Cách thử nghiệm** 32](#_Toc43449619)

[**3.3.** **Kiểm tra dự đoán cho tập dữ liệu mới** 37](#_Toc43449620)

[**CHƯƠNG III: KẾT QUẢ CHƯƠNG TRÌNH** 40](#_Toc43449621)

[**1.** **Kết quả chương trình** 40](#_Toc43449622)

[**1.1.** **Ma trận Confusion Matrix kiểm tra độ chính xác** 40](#_Toc43449623)

[**1.2.** **Phân lớp trên tập dữ liệu ban đầu** 41](#_Toc43449624)

[**1.3.** **Kiểm tra cho tập dữ liệu mới** 41](#_Toc43449625)

[**2.** **Đánh giá** 42](#_Toc43449626)

[**TÀI LIỆU THAM KHẢO** 43](#_Toc43449627)

**DANH MỤC HÌNH ẢNH**

[H 1 Bài toán phân lớp 11](#_Toc43452787)

[H 2 Mô hình bài toán phân lớp 12](#_Toc43452788)

[H 3 Thuật toán PLA 13](#_Toc43452789)

[H 4 Thuật toán Support Vecto Machine 14](#_Toc43452790)

[H 5 Mô hình LogisticRegreession 16](#_Toc43452791)

[H 6 Mô hình LogisticRegreession 18](#_Toc43452792)

[H 7 Mô hình LogisticRegreession 18](#_Toc43452793)

[H 8 Mô hình LogisticRegreession 19](#_Toc43452794)

[H 9 Đồ thị biểu diễn 21](#_Toc43452795)

[H 10 Đồ thị biểu diễn 22](#_Toc43452796)

[H 11 Mô hình bài toán 23](#_Toc43452797)

[H 12 Bảng xếp hạng các ngôn ngữ lập trình được yêu thích nhất 25](#_Toc43452798)

[H 13 Cài đặt python 26](#_Toc43452799)

[H 14 Cài đặt python 26](#_Toc43452800)

[H 15 Cài đặt python 27](#_Toc43452801)

[H 16 Cài đặt python 27](#_Toc43452802)

[H 17 Cài đặt pyCharm 28](#_Toc43452803)

[H 18 Cài đặt pyCharm 28](#_Toc43452804)

[H 19 Cài đặt pyCharm 29](#_Toc43452805)

[H 20 Cài đặt pyCharm 29](#_Toc43452806)

[H 21 Cài đặt pyCharm 30](#_Toc43452807)

[H 22 Cài đặt pyCharm 30](#_Toc43452808)

[H 23 Cài đặt pyCharm 31](#_Toc43452809)

[H 24 Cài đặt pyCharm 31](#_Toc43452810)

[H 25 Cài đặt bộ dữ liệu 33](#_Toc43452811)

[H 26 Ma trận confusion Matrix 41](#_Toc43452812)

[H 27 Bộ dữ liệu thực tế 42](#_Toc43452813)

# 

# **LỜI CẢM ƠN**

Để hoàn thành đồ án tốt, lời đầu tiên chúng em xin chân thành cảm ơn đến toàn thể thầy cô trong trường Đại Học Điện Lực, đặc biệt hơn là các thầy cô trong ngành công nghệ thông tin, bộ môn học máy nâng cao nói riêng, những người đã tận tình hướng dẫn dạy dỗ và trang bị cho chúng em những kiến thức bổ ích trong năm vừa qua. Đặc biệt em xin chân thành gửi lời cảm ơn sâu sắc đến cô giáo Phạm Thị Kim Dung , người đã tận tình hướng dẫn, trực tiếp chỉ bảo và tạo mọi điều kiện giúp đỡ chúng em trong suốt quá trình làm báo cáo. Chúng em xin chân thành cảm ơn!!

# **MỞ ĐẦU**

 Lý do chọn đề tài Ngày nay, trí tuệ nhân tạo đang phát triển mạnh mẽ và xâm nhập vào rất nhiều lĩnh vực trong cuộc sống như tự động dịch thuật, nhận dạng giọng nói, điều khiển tự động, nhận dạng khuôn mặt, nhận dạng chữ viết tay. v.v. Nó giờ được coi là xu hướng công nghệ thế giới và nhiều người cho rằng đó là cuộc cách mạng công nghiệp lần thứ 4. Trong lĩnhvực AI, học máy (machine learning) là một lĩnh vực liên quan đến việc nghiên cứu các kỹ thuật và xây dựng các hệ thống có thể “học”tự động từ dữ liệu, từ đó giải quyết các vấn đề bài toán cụ thể. Hay nói cách khác học máy phần nào đó giúp cho máy tính có thể xử lý dữ liệu và đưa ra quyết định như con người. Ví dụ khi đưa một dữ liệu vào thì máy tính có thể xử lý dữ liệu đó thuộc loại nào. Trong các bài toán Machine learning có hai loại bài toán đặc trưng đó là bài toán phân lớp và phân cụm. Mỗi bài toán có những đặc trưng riêng và phạm vi áp dụng vào các loại bài toán thực tế khác nhau. Bên cạnh đó, bài toán phân lớp ứng viên là một bài toán hấp dẫn và có tính ứng dụng cao. Trong khuôn khổ báo cáo này, chúng em sẽ đi sâu vào nghiên cứu bài toán phân lớp và cụ hơn là thuật toán Logistic Regession để áp dụng vào nhận dạng giới tính qua dữ liệu ứng viên đầu vào, xem ứng viên đó là có được nhận vào một trường đại học danh tiếng hay không.

# **CHƯƠNG I: THUẬT TOÁN PHÂN LỚP NHỊ PHÂN LOGISTIC REGRESSION**

1. **Học máy (Machine Learning)**
   1. **Giới thiệu**

Những năm gần đây, Trí tuệ nhân tạo (AI - Artificial Intelligence) đang phát triển mạnh mẽ và xâm nhập và trở thành những công nghệ cốt lõi trên nhiều lĩnh vực của đời sống con người. Ta có thể bắt gặp sự hiện diện của AI ở khắp nơi. Ví dụ: Xe tự hành của Google và Tesla, hệ thống tự tag khuôn mặt trong ảnh của Facebook, trợ lý ảo Siri của Apple, hệ thống gợi ý sản phẩm của Amazon, hệ thống gợi ý phim của Netflix, máy chơi cờ vây AlphaGo của Google DeepMind, …, chỉ là một vài trong vô vàn những ứng dụng của AI/Machine Learning. Machine Learning là một lĩnh vực của AI, nó có khả năng tự học hỏi dựa trên dữ liệu đưa vào mà không cần phải được lập trình cụ thể. Machine learning còn cung cấp một phương pháp hiệu quả để học hỏi dữ liệu thay vì dựa vào con người để phân tích và dự đoán. Nhờ vào học máy, các máy tính có thể xử lý ảnh và đưa ra giới tính cho mỗi bức ảnh xem bức ảnh đưa vào là nam hay là nữ. Các loại học máy:

* Học có giám sát (Supervised Learning): Là phương pháp sử dụng những dữ liệu đã được gán nhãn từ trước để đưa ra các dự đoán giữa đầu vào và đầu ra. Các dữ liệu này được gọi là dữ liệu huấn luyện và chúng là cặp các đầu vào-đầu ra. Học có giám sát sẽ xem xét các tập huấn luyện này để từ đó có thể đưa ra dự đoán đầu ra cho 1 đầu vào mới chưa gặp bao giờ. Ví dụ một “email”có thể được gán nhãn “thứ rác” hoặc “không thư rác” và đưa vào mô hình Supervised Learing để phân loại.
* Học không giám sát (Unsupervised Learning): Khác với học có giám sát, học không giám sát sử dụng những dữ liệu chưa được gán nhãn từ trước để suy luận. Phương pháp này thường được sử dụng để tìm cấu trúc của tập dữ liệu. Tuy nhiên lại không có phương pháp đánh giá được cấu trúc tìm ra được là đúng hay sai. Ví dụ như phân cụm dữ liệu, triết xuất thành phần chính của một chất nào đó.
* Học nửa giám sát là một lớp của kỹ thuật học máy, sử dụng cả dữ liệu đã gán nhãn và chưa gán nhãn để huấn luyện - điển hình là một lượng nhỏ dữ liệu có gán nhãn cùng với lượng lớn dữ liệu chưa gán nhãn. Học nửa giám sát đứng giữa học không giám sát (không có bất kì dữ liệu có nhãn nào) và có giám sát (toàn bộ dữ liệu đều được gán nhãn). Nhiều nhà nghiên cứu nhận thấy dữ liệu không gán nhãn, khi được sử dụng kết hợp với một chút dữ liệu có gán nhãn, có thể cải thiện đáng kể độ chính xác. Để gán nhãn dữ liệu cho một bài toán học máy thường đòi hỏi một chuyên viên có kĩ năng để phân loại bằng tay các ví dụ huấn luyện. Trong học máy thì có hai bài toán đặc trưng:
* Bài toán phân cụm
* Bài toán phân lớp
  1. **Bài toán phân cụm**

Bài toán phân cụm Bài toán phân cụm là 1 trong những bài toán của lĩnh vực Unsupervised Learning (Học không giám sát), dữ liệu được mô tả trong bài toán không được dán nhãn hay nói cách khác thì bài toán này không có đầu ra. Trong trường hợp này, thuật toán sẽ tìm cách phân cụm - chia dữ liệu thành từng nhóm có đặc điểm tương tự nhau, nhưng đồng thời đặc tính giữa các nhóm đó lại phải càng khác biệt càng tốt. Ví dụ: Dữ liệu của chúng ta có thể là bất cứ thứ gì, chẳng hạn như dữ liệu về khách hàng: Thuật toán phân cụm sẽ rất hữu ích trong việc đánh giá và chia thành các nhóm người dùng khác nhau, rồi từ đó ta có thể đưa ra những chiến lược marketing phù hợp trên từng nhóm người dùng đó.

* 1. **Bài toán phân lớp**

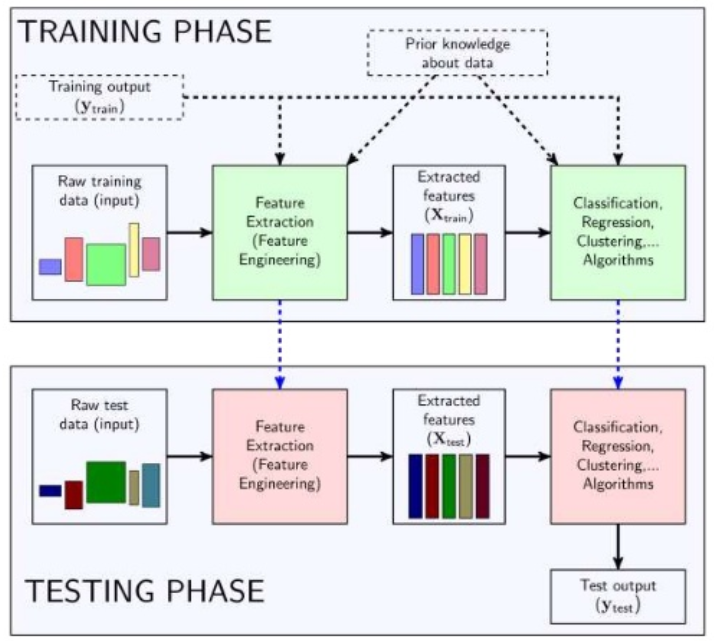
Bài toán phân lớp thuộc một nhánh ứng dụng chính của lĩnh vực Supervised Learning. Và bài toán là quá trình phân lớp một đối tượng dữ liệu vào hay nhiều lớp đã cho trước nhờ một mô hình phân lớp (model). Mô hình này được xây dựng dựa trên một tập dữ liệu được xây dựng trước đó có gián nhãn (hay còn gọi là huấn luyện). Quá trình phân lớp là quá trình gián nhãn cho đối tượng dữ liệu.



H 1 Bài toán phân lớp

Như vậy, nhiệm vụ của bài toán phân lớp là cần tìm một mô hình phân lớp để khi có dữ liệu mới thì có thể xác định được dữ liệu đó thuộc vào phân lớp nào. Ví dụ: Ta có bộ dữ liệu traning bao gồm có hạt đậu, hạt lạc, hạt lúa sau đó ta xây dựng mô hình phân lớp để gán nhãn cho bộ dữ liệu đó là hạt đậu được gán nhãn là A, hạt lúa có gán nhãn là B, hạt lạc gán nhãn là C. tiếp theo nhiệm vụ của bài toán phân lớp là tìm mô hình phân lớp để khi ta đưa 1 dữ liệu bị che nhãn vào mô hình phân lớp để phân lớp xem dữ liệu đó thuộc vào hạt nào trong 3 hạt đã được gán nhãn.

Mô hình bài toán phân lớp:



H 2 Mô hình bài toán phân lớp

Trong bài toán phân lớp dữ liệu thì có các bài toán sau đây:

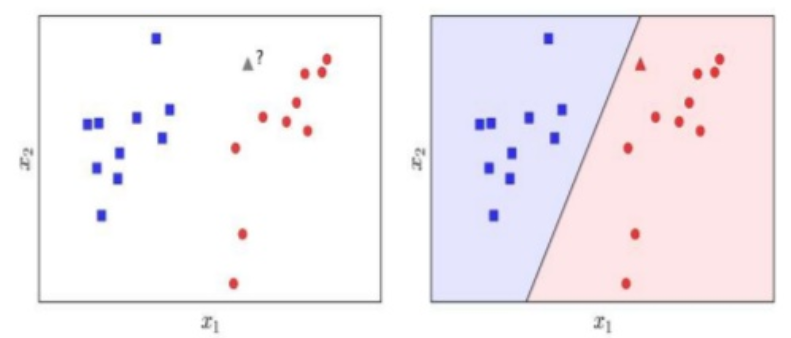
* Bài toán phân lớp đa lớp (multiclass).
* Bài toán phân lớp đa trị.
* Bài toán phân lớp nhị phân (binary).

Bài toán phân lớp nhị phân là bài toán gắn nhãn dữ liệu cho đối tượng vào một trong hai lớp khác nhau dựa vào việc dữ liệu đó có hay không có các đặc trưng (feature) của bộ phân lớp.

Một số thuật toán phổ biến:

* Thuật toán Perceptron Learning Algorithm.
* Thuật toán Support Vecto Machine
* Thuật toán Naive Bayes
* Thuật toán Logistic Regression.
  + 1. **Thuật toán Perceptron Learning Algorithm**

PLA là thuật toán classification nền tảng của các model Neuron Network và deeplearning. Ý tưởng của thuật toán đó là với các classes khác nhau, hãy tìm các đường biên để phân chia các classes này thành những vùng diện tích tách biệt. Trường hợp đơn giản nhất của thuật toán này là phân chia nhị phân (binary classification) bằng những đường biên tuyến tính. Bài toán được phát biểu như sau: Cho 2 class được dán nhãn khác nhau, tìm một đường thẳng sao cho toàn bộ các điểm thuộc class 1 nằm về 1 phía của đường thằng và toàn bộ các điểm thuộc class 2 sẽ nằm về phía còn lại với giả định luôn tồn tại 1 đường thẳng như thế (không rơi vào trường hợp 2 class nằm chồng lấn lên nhau dẫn tới không tồn tại đường biên). Ví dụ: Giả sử chúng ta có hai tập hợp dữ liệu đã được gán nhãn được minh hoạ trong hình 3 bên trái dưới đây. Hai class của chúng ta là tập các điểm màu xanh và tập các điểm màu đỏ. Bài toán đặt ra là: từ dữ liệu của hai tập được gán nhãn cho trước, hãy xây dựng một classifier (bộ phân lớp) để khi có một điểm dữ liệu hình tam giác màu xám mới, ta có thể dự đoán được màu (nhãn) của nó.



H 3 Thuật toán PLA

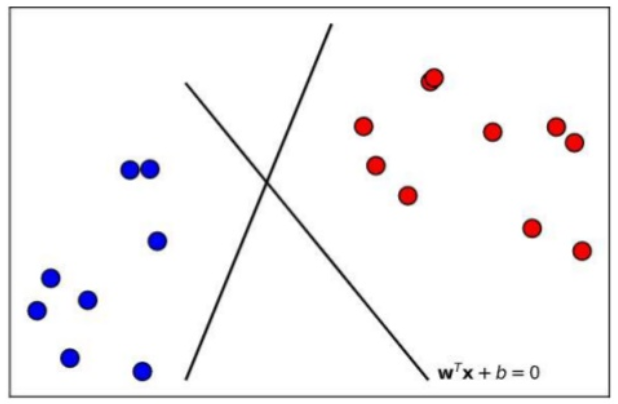
Chúng ta cần tìm lãnh thổ của mỗi class sao cho, với mỗi một điểm mới, ta chỉ cần xác định xem nó nằm vào lãnh thổ của class nào rồi quyết định nó thuộc class đó. Để tìm lãnh thổcủa mỗi class, chúng ta cần đi tìm biên giới (boundary) giữa hai lãnh thổ này. Hình 3 bên phải minh họa một đường thẳng phân chia hai class trong mặt phẳng. Phần có nền màu xanh được coi là lãnh thổ của lớp xanh, phần có nên màu đỏ được coi là lãnh thổ của lớp đỏ. Trong trường hợp này, điểm dữ liệu mới hình tam giác được phân vào class đỏ.

* + 1. **Thuật toán Logistic Regression**

Phương pháp hồi quy logistic là một thuật toán với mô hình dự đoán được sử dụng khi biến y là phân loại nhị phân. Nghĩa là chỉ lấy giá trị 1 và 0. Hiểu theo một cách khác nữa đó là mô hình này nhằm dự đoán đầu ra rời rạc y tương ứng với một vecto đâu vào X. Việc này tương đương với chuyện phân loại đầu vào X vào các nhóm y tương ứng.

* + 1. **Thuật toán Support Vecto Machine**

Support Vector Machine - SVM là một phương pháp học có giám sát trong các mô hình nhận dạng mẫu. Nó không chỉ hoạt động tốt với các dữ liệu được phân tách tuyến tính mà còn tốt với cả dữ liệu phân tách phi tuyến. Với nhiều bài toán, SVM mang lại kết quả tốt như mạng nơ-ron với hiệu quả sử dụng tài nguyên tốt hơn hẳn.



H 4 Thuật toán Support Vecto Machine

* + 1. **Thuật toán Naive Bayes**

Naive Bayes là một thuật toán phân loại cho các vấn đề phân loại nhị phân (hai lớp) và đa lớp. Kỹ thuật này dễ hiểu nhất khi được mô tả bằng các giá trị đầu vào nhị phân hoặc phân loại.

Có một thuật toán hay trong phân lớp nhị phân là thuật toán Logistic Regression áp dụng nhiều trong nhiều bài toán phân lớp. Chương tiếp theo chúng em sẽ trình bày kỹ hơn về thuật toán này.

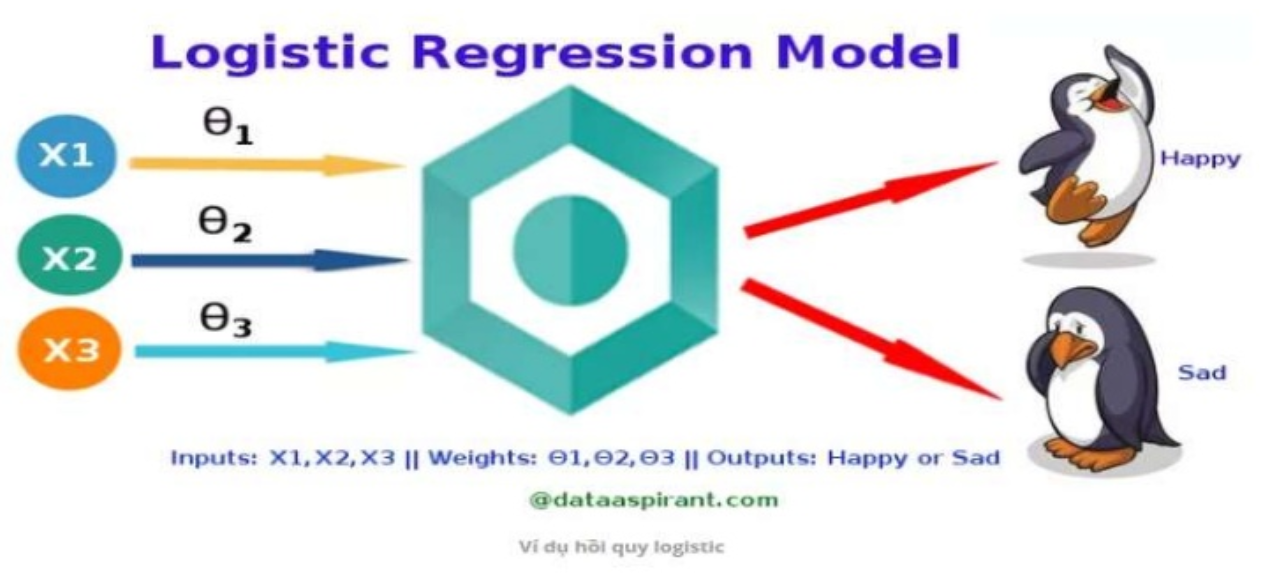
1. **Thuật toán phân lớp nhị phân Logistic Regression**
   1. **Giới thiệu**

 Mô hình hồi quy logistic thuộc vào bài toán học giám sát trong Machine Learning. Mô hình hồi quy logistic sẽ dùng hàm biến đổi logarit cho biến đầu ra để biến mô hình từ quan hệ phi tuyến tính sang tuyến tính. Hay nói cách khác, nó biểu diễn quan hệ hồi quy tuyên tính dưới hàm logarit. Mô hình logistic có 1 giả định rằng biến dự đoán có giá trị rời rạc. Nếu biến dự đoán chỉ lấy hai giá trị rời rạc, đó là mô hình Binary Logistic Regression.

Nếu biến dự đoán nhiều hơn hai giá trị thì đó là mô hình Multinomial Logistic Regression. Trong đồ án này, em chỉ tập trung vào Binary Logistic Regression vì nó rất hay được sử dụng trong thực tế dự đoán: nam/nữ, có/không, sống/chết, qua/hay trươt, bị bệnh/ không bị bệnh.

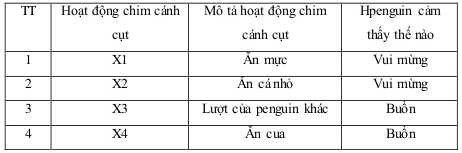
Phương pháp hồi quy logistic là một mô hình hồi quy nhằm dự đoán giá trị đầu ra rời rạc (discrete target variable) y ứng với một véc-tơ đầu vào x. Việc này tương đương với chuyện phân loại các đầu vào x vào các nhóm y tương ứng. Hay nói cách khác nó là một thuật toán với mô hình dự đoán được sử dụng khi biến y là phân loại nhị phân với đầu ra là [1] hoặc [0].

* 1. **Ví dụ mô hình LogisticRegreession**

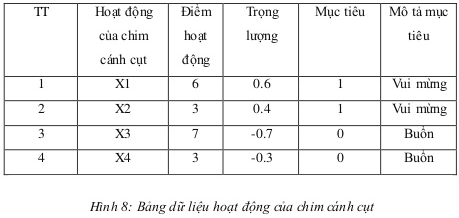


H 5 Mô hình LogisticRegreession

Giả sử HPenguin muốn biết, khả năng nó sẽ hạnh phúc như thế nào dựa trên các hoạt động hàng ngày của nó. Nếu chim cánh cụt muốn xây dựng mô hình hồi quy logistic để dự đoán hạnh phúc dựa trên các hoạt động hàng ngày của nó. Chim cánh cụt cần cả hoạt động vui và buồn. Trong thuật ngữ học máy, các hoạt động này được gọi là các tham số đầu vào ( tính năng ). Vì vậy, hãy tạo một bảng chứa các hoạt động chim cánh cụt và kết quả của hoạt động đó như vui hay buồn.



 Penguin sẽ sử dụng các hoạt động (tính năng) trên để huấn luyện mô hình hồi quy logistic. Sau đó, mô hình hồi quy logistic được đào tạo sẽ dự đoán cảm giác của chim cánh cụt đối với các hoạt động của chim cánh cụt mới. Vì không thể sử dụng bảng dữ liệu phân loại ở trên để xây dựng hồi quy logistic. Bảng dữ liệu hoạt động trên cần chuyển đổi thành điểm hoạt động, trọng lượng và mục tiêu tương ứng.



Các tập dữ liệu cập nhật trông như thế này. Trước khi đi vào bài toán, hãy hiểu thêm về bảng dữ liệu trên.

• Hoạt động chim cánh cụt:

Các hoạt động chim cánh cụt làm hàng ngày như ăn cá nhỏ, ăn cua .. vv.

• Điểm hoạt động:

Điểm hoạt động giống như số tương đương với hoạt động của chim cánh cụt. Đối với hoạt động ăn mực, điểm hoạt động tương ứng là 6 và tương tự, đối với các hoạt động khác, điểm số là 3, 7, 3.

• Trọng lượng:

Các trọng số giống như các trọng số tương ứng với mục tiêu cụ thể.

Giả sử cho hoạt động X1, chúng ta có trọng số là 0,6.

Nó có nghĩa là để nói nếu chim cánh cụt thực hiện hoạt động X1, người tự tin 60% để nói rằng chim cánh cụt sẽ hạnh phúc.

Nếu bạn quan sát các trọng số cho lớp mục tiêu thì hạnh phúc là tích cực và các trọng số cho lớp mục tiêu buồn là tiêu cực.

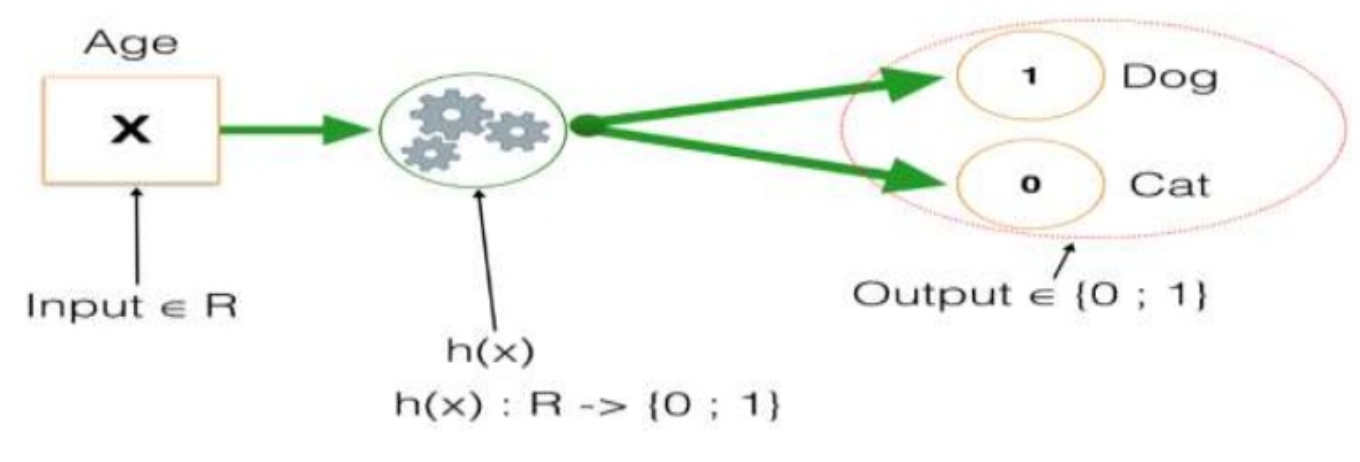
Điều này là do vấn đề chúng tôi đang giải quyết một phân loại nhị phân. Sẽ nói về phân loại nhị phân trong các phần tiếp theo của bài này.

• Mục tiêu:

Mục tiêu chỉ là các giá trị nhị phân. Giá trị 1 đại diện cho mục tiêu Hạnh phúc và giá trị 0 đại diện cho mục tiêu Buồn .

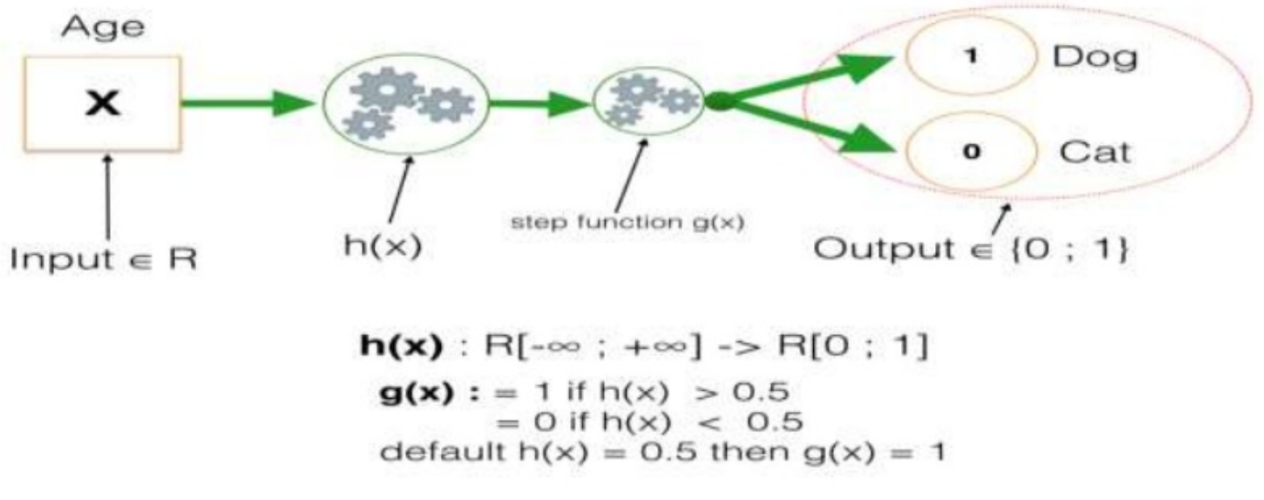
* 1. **Mô hình Logistic Regression**

Ta đi vào một bài toán cụ thể như sau:



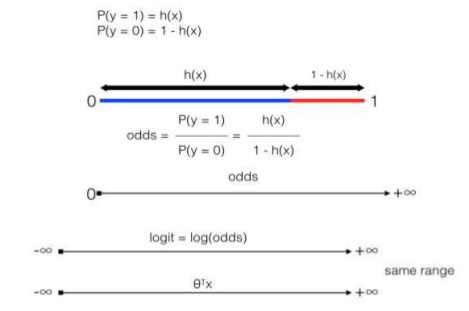
H 6 Mô hình LogisticRegreession

Trong đó input nhận giá là số thực (vecto giá trị thực), còn output nhận một trong hai giá trị 0 hoặc 1. Với những hàm thôngthường như (sin, cos) thì chỉ có thể biến đổi giá trị đầu vào từ giá trị liên tục sang output cũng là giá trị liên tục. Vậy nên thay giá trị output chỉ nhận giá trị như trên, ta tách h(x) ra thành hàm riêng biệt. Một hàm sẽ chuyển đồi input sang R [0,1], còn hàm thư hai ta thử nghiệm với Linear Regression, đặt một ngưỡng [0,1] nếu vượt qua ngưỡng thì trả về giá trị một còn ngược lại thì trả về 0. Hàm số này được gọi là Step Function.



H 7 Mô hình LogisticRegreession

Giá trị default khi h (x) có giá trị bằng chính xác ngưỡng đặt ra. Trong ví dụ này, nếu h(x) = threshold (ngưỡng) thì kết quả trả về là 1. Nếu để ý một chút, ta sẽ thấy giá trị nằm trong [0, 1], đây chính là xác suất xảy ra y = 1, Hay nói cách khác h(x) = P(y = 1). Nghĩa là ta đi xây dựng hàm số tính xác suất đối tượng có nhãn là “dog”, nếu xác suất >= 0.5 thì gán nhãn “dog” và ngược lại. Và vì đối tượng cần kiểm tra chỉ là “dog” or “cat” nên ta có xác suất P (y = 0 ) + P(y = 1 ) = 1.



H 8 Mô hình LogisticRegreession

Xác suất là tỉ lệ giữa số lần xảy ra và số lần thử nghiệm, tuy nhiên khi phép thử chỉ có 2 khả năng thì chúng ta còn có một đại lượng khác để đánh giá khả năng xảy ra của một biến cố, được gọi là odds (hay ratio). Trong bài toán này, ta có odds được tính như trên.

- Khi odds=1 nghĩa là h(x)=1- h(x). Đây chính là ngưỡng mà ta đặt ra.

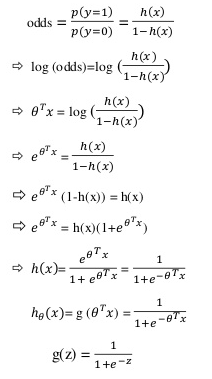
- Khi odds >1 nghĩa là h(x) >1-h(x) khi odds càng lớn thì h(x) càng lớn so với 1- h(x) hay h(x) càng tiến đến gần 1 thì xác suất y=1 càng cao.

- Khi odds <1 nghĩa là h(x) < 1 – h(x) thì xác suất y=0 càng cao.

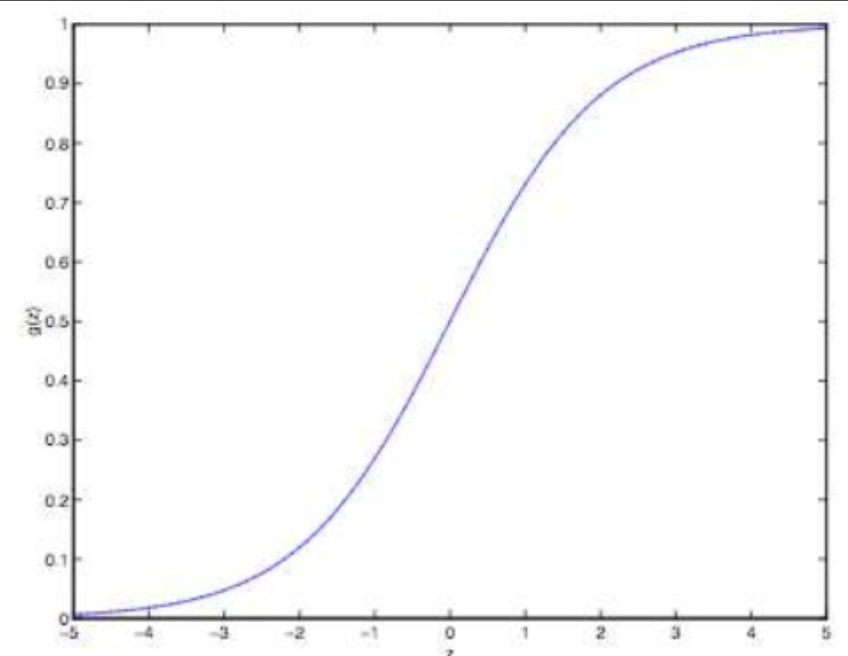
- Bên cạnh đó Logarit của một số nguyên dương có giá trị trong khoảng [-∞ , +∞], nên nếu lấy logarit của odds (được gọi là logit) ta sẽ có một đại lượng biến thiên như θTx (trong mô hình Linear Regression). Chú ý rằng khi input là vector thì θ cũng là vector, nên θx phải được viết là θTx (nghĩa là matrix chuyển vị θ nhân với vector x, vector là một trường hợp đặc biệt của matrix). Vậy nên ta có thể đặt tham số và input vào logit để máy có thể học được từ dữ liệu



Biến đổi một chút sẽ có hàm số của h(x):



Trong đó g (z) được gọi là sigmoid function. Đây là một trong những hàm được sử dụng phổ biến trong ML, dạng biểu diễn trên đồ thị như sau:



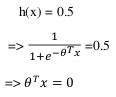
H 9 Đồ thị biểu diễn

Trong mô hình Logistic Regression, dữ liệu được phân tách các loại bởi một hàm số. Cụ thể:

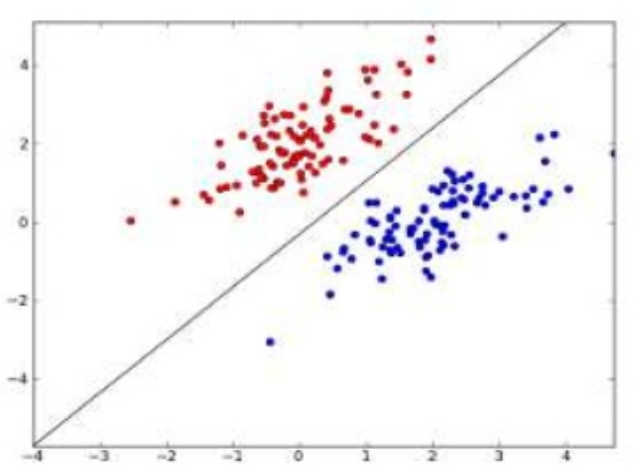
h(x) > 0.5 => y = 1

h(x) ﹤0.5 => y = 0

có nghĩa h(x) = 0.5 là ngưỡng phân biệt giữa y = 0 và y =1. Hay nói cách khác hàm số h(x) = 0.5 chính là đường ranh giới giữa những dữ liệu y = 0 và y = 1 (được gọi là Decision Boundary).



Khi ta biểu diễn dữ liệu chỉ với các trục của x (không bao gồm y), chẳng hạn trên dữ liệu có 2 features ta có:



H 10 Đồ thị biểu diễn

Trong trường hợp này, 𝜃T chỉ là hàm tuyến tính nên đường ranh giới chỉ là một đường thẳng.

# **CHƯƠNG II: ÁP DỤNG THUẬT TOÁN LOGISTIC REGREESION CHO BÀI TOÁN PHÂN LỚP ỨNG VIÊN**

1. **Bài toán phân lớp ứng viên xét tuyển vào trường đại học**
   1. **Giới thiệu**

Một trường đại học danh tiếng có một số lượng các ứng viên muốn được duyệt vào trường. Việc duyệt thông tin này cần phải được tổng hợp từ rất nhiều thông tin của ứng viên đó. Với thuật toán Logistic Regression ta có thể đánh giá các ứng viên đó nhanh chóng với độ chính xác cao chỉ cần với máy tính.

* 1. **Mô hình bài toán**

****

H 11 Mô hình bài toán

Phần này chúng em sẽ trình bày mô hình phân nhóm khách hàng được vay và không được vay. Đầu tiên ta đưa bộ dữ liệu ra làm 2 phần là train và test. Quá trình phân nhóm được thực hiện theo các bước sau:

* Bước 1: Huấn luyện mô hình phân lớp dữ liệu

Ta đưa dữ liệu của các khách hàng đã được gán nhãn vào quá trình phân lớp. Trích chọn đặc trưng cho mỗi khách hàng. Sau đó ta đưa dữ liệu các khách hàng vào mô hình phân lớp. Sau khi kết thúc quá trình huấn luyện, hệ thống sẽ lưu lại các tham số để thực hiện cho quá trình phân lớp tiếp theo.

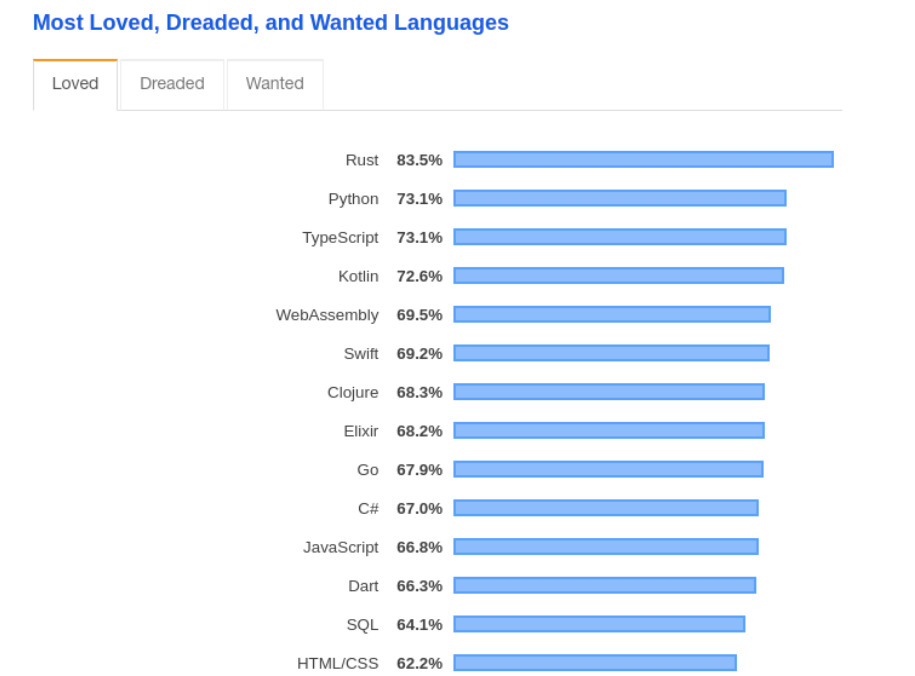
* Bước 2: Phân lớp dữ liệu – thử nghiệm mô hình

Sau khi xây dựng mô hình huấn luyện dữ liệu ở bước 1: Ta tiến hành phân lớp với dữ liệu của một khách hàng bị che nhãn. Ta trích chọn đặc trưng dữ liệu của khách hàng vừa mới đưa vào và tính toán các tham số của hàm quyết định (các tham số của mô hình huấn luyện) để xác lớp cho khách hàng đó.

1. **Môi trường thử nghiệm**
   1. **Cài đặt python.**

Python là một ngôn ngữ lập trình bậc cao được tạo bởi Guido Rossum và công bố lần đầu vào năm 1991. Ngôn ngữ Python này được thiết kế để giúp nhanh chóng xây dựng các ứng dụng phức tạp với số dòng code tối thiểu. Điểm mạnh của ngôn ngữ này là dễ đọc, dễ học và dễ nhớ. Do đó, đây là một ngôn ngữ lập trình có hình thức rất sáng sủa, cấu trúc rõ ràng, dễ học đối với người mới.

Python là ngôn ngữ lập trình có số lượng thư viện hỗ trợ phong phú, có thể chạy trên các hệ điều hành khác nhau. Rất nhiều công ty lớn sử dụng Python như là: Google, Youtube, BitTorrent,…



H 12 Bảng xếp hạng các ngôn ngữ lập trình được yêu thích nhất

Ngôn ngữ lập trình Python được sử dụng rộng rãi trong rất nhiều lĩnh vực Trí tuệ nhân tạo, học máy, khoa học máy tính, lập trình web, …

Để có thể code và chạy các chương trình với ngôn ngữ Python, chúng ta cần **cài đặt môi trường lập trình Python**(cài đặt python) cho máy và có thể cài thêm IDE để hỗ trợ chúng ta code nhanh hơn. Trong bài này, Lập Trình Không Khó sẽ hướng dẫn bạn từng bước cài đặt môi trường lập trình Python và cài thêm Pycharm IDE để hỗ trợ code Python trên Windows. Chúng ta sẽ cùng nhau đi qua từng bước chi tiết nhất để các bạn có thể hiểu rõ nhất.

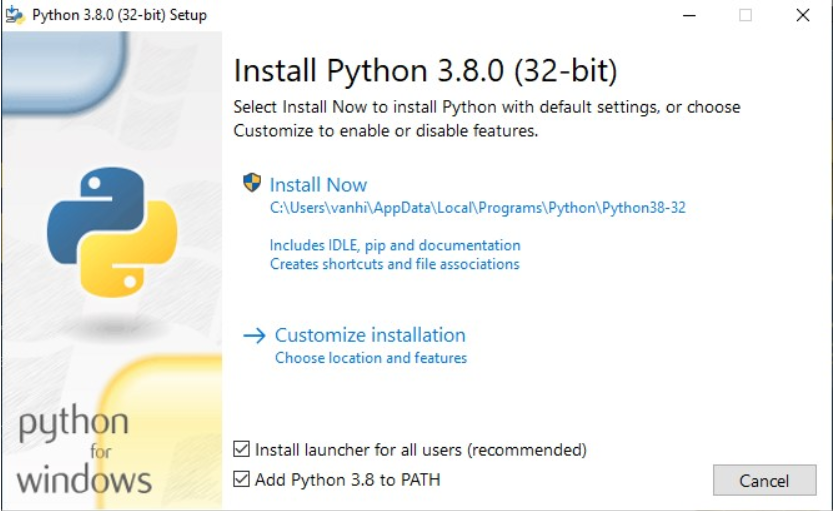
**Cài đặt Python trên Windows**

Bước 1. Bạn cần truy cập vào trang chủ của Python là [http://www.python.org/downloads/](https://www.python.org/downloads/) và lựa chọn phiên bản Python muốn cài. Tại thời điểm của bài hướng dẫn này, Python bản mới nhất đang là 3.8.0.



H 13 Cài đặt python

Bước 2. Khi tải xong, bạn tiến hành chạy file cài đặt có đuôi .exe vừa tải ở bước 1. Nhấp vào Install Now để cài đặt (Nhớ tích vào các lựa chọn như hình).



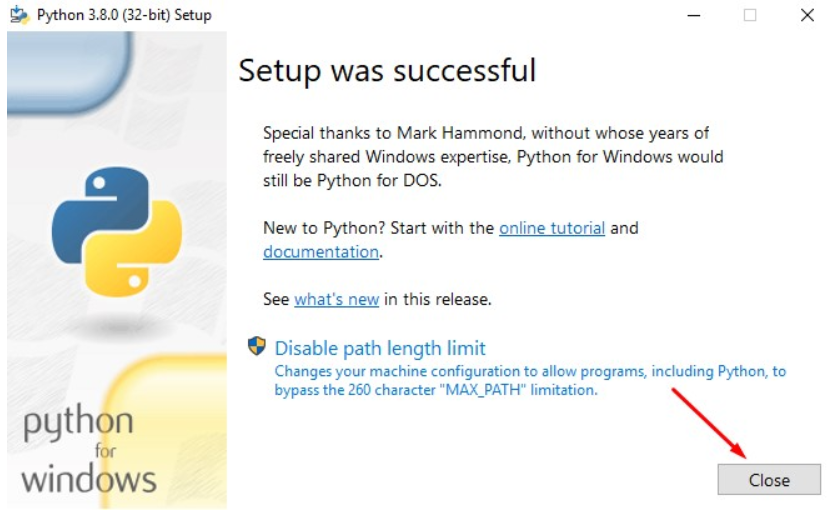
H 14 Cài đặt python

Bước 3. Bạn có thể nhìn thấy tiến trình cài đặt Python lên máy.



H 15 Cài đặt python

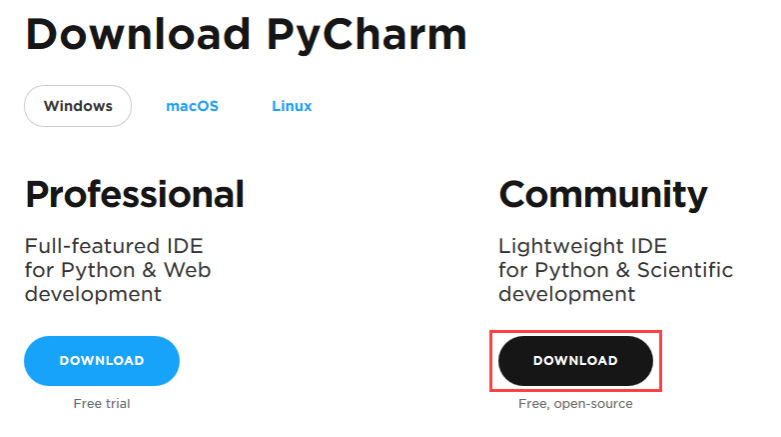
Bước 4. Khi cài đặt hoàn tất, bạn sẽ nhìn thấy một màn hình thông báo như dưới đây. Khi đó hãy click “Close” để hoàn tất cài đặt.



H 16 Cài đặt python

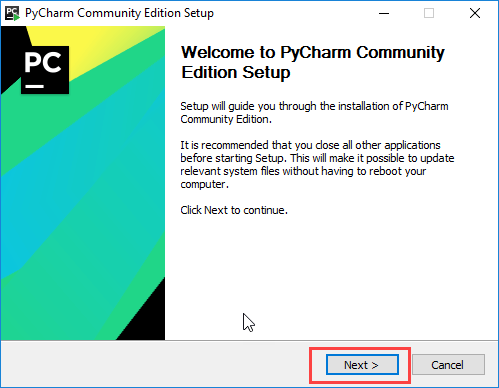
* 1. **Cài đặt pycharm.**

Bước 1. Bạn cần tải bản cài đặt Pycharm từ trang chủ tại <https://www.jetbrains.com/pycharm/download/> bằng cách click vào nút “DOWNLOAD”. Hãy nhớ download bản Community là bản miễn phí nhé.



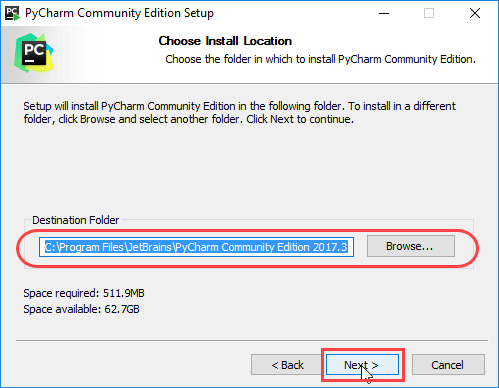
H 17 Cài đặt pyCharm

Bước 2. Khi tải hoàn tất, tiến hành chạy để cài đặt Pycharm. Một cửa sổ cài đặt sẽ hiện ra, hãy nhấp vào “Next”.



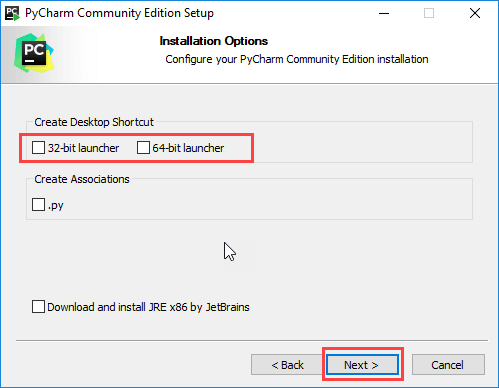
H 18 Cài đặt pyCharm

Bước 3. Ở màn hình tiếp theo, bạn có thể thay đổi đường dẫn để cài đặt Pycharm hoặc để mặc định. Sau đó nhấp vào “Next”.



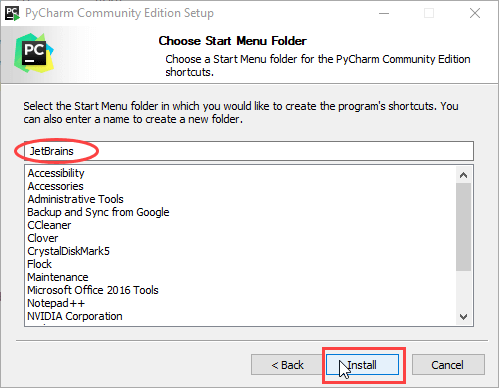
H 19 Cài đặt pyCharm

Bước 4. Bạn sẽ thấy một màn hình có một vài lựa chọn, nếu bạn muốn mở IDE từ màn hình Desktop thì hãy lựa chọn. Sau đó ấn vào “Next”.



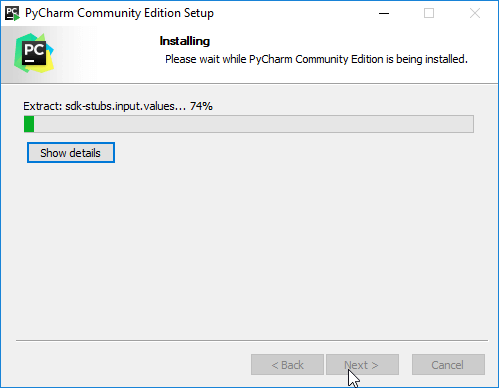
H 20 Cài đặt pyCharm

Bước 5. Chọn thư mục trong start menu của Windows. Hãy để mặc định là thư mục JetBrains vào nhấp vào “Install”.



H 21 Cài đặt pyCharm

Bước 6. Chờ quá trình cài đặt hoàn tất.



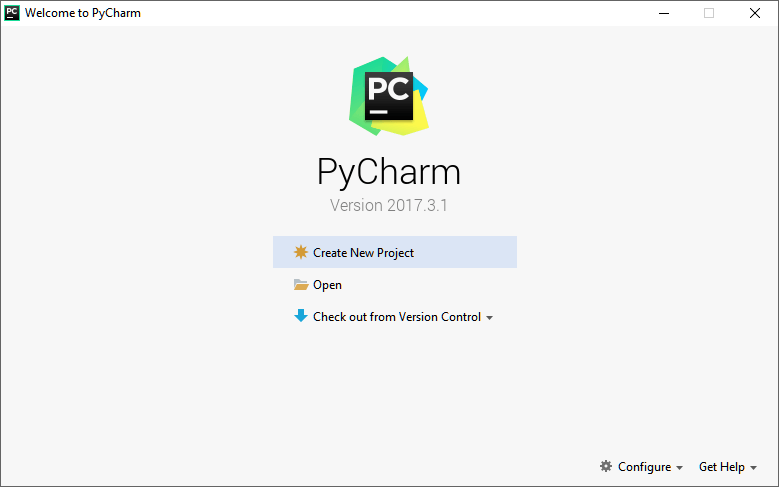
H 22 Cài đặt pyCharm

Bước 7. Khi cài đặt hoàn thành, bạn sẽ nhìn thấy màn hình thông báo như dưới đây. Nếu bạn muốn mở PyCharm luôn, hãy nhấp vào “Run PyCharm Community Edition” và nhấp vào “Finish” nhé.



H 23 Cài đặt pyCharm

Bước 8. Nếu tại bước 7 bạn chọn mở PyCharm, bạn sẽ nhìn thấy màn hình của PyCharm trông như thế này.



H 24 Cài đặt pyCharm

Vậy là chúng ta đã hoàn tất công việc cài đặt môi trường để lập trình Python trên Windows.

* 1. **Cài đặt bộ thư viện**

### **NumPy**

Được tạo bởi Travis Oliphant, **NumPy** là một “ngựa kéo” phân tích thực sự của Python. Nó cung cấp cho người dùng cách làm việc với các mảng nhiều chiều, cùng một số lượng lớn các hàm để xử lý trên các toán tử toán học nhiều chiều trên các mảng đó. Mảng là các khối dữ liệu được sắp xếp theo nhiều chiều dựa trên các véc tơ và ma trận trong toán học. Mảng thường hữu ích không chỉ trong việc lưu dữ liệu mà cả việc tính toán nhanh các ma trận, điều không thể thiếu khi giải quyết các vấn đề liên quan đến khoa học dữ liệu.

* Cài đặt: pip install numpy
* Trang chủ: [http://www.numpy.org](http://www.numpy.org/)

### **SciPy**

Là một dự án gốc bởi Travis Oliphant, Pearu Peterson, and Eric Jones, SciPy hoàn thiện các tính năng của NumPy, nhằm cung cấp các thuật toán cho đại số tuyến tính, không gian ma trận, xử lý tín hiệu và xử lý ảnh, tối ưu, biến đổi Fourier,…

* Cài đặt: pip install scipy
* Trang chủ: [http://www.scipy.org](http://www.scipy.org/)

### **Matplotlib**

Được phát triển bởi John Hunte, matplotlib là một thư viện xây dựng các khối cần thiết để tạo các biểu đồ chất lượng từ mảng và trực quan và tương tác với chúng.

* Cài đặt: pip install matplotlib
* Trang chủ: [http://matplotlib.org](http://matplotlib.org/)

### **Scikit-learn**

Bắt đầu như một phần của SciKits, Scikit-learn là lõi hoạt động của khoa học dữ liệu trên Python. Nó cung cấp tất cả những gì bạn cần để tiền xử lý dữ liệu, học giám sát và không giám sát, lựa chọn mô hình, validate và error metrics.

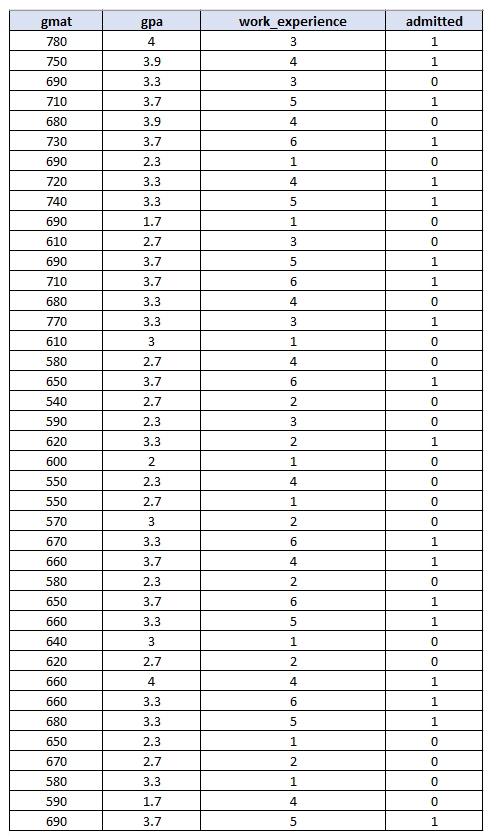
* Cài đặt: pip install scikit-learn

Trang chủ: <http://scikit-learn.org/stable>

1. **Cài đặt**

### **Bộ dữ liệu**

Đây là bộ dữ liệu của các ứng viên để xét vào một trường đại học danh tiếng. Để xác định, có 2 kết quả có thể xảy ra: Được thừa nhận (được biểu thị bằng giá trị '1') so với Bị từ chối (được biểu thị bằng giá trị '0').



H 25 Cài đặt bộ dữ liệu

* 1. **Cách thử nghiệm**

Để thử nghiệm cho thuật toán Logistic Regression, chúng em lấy 75% dữ liệu của tổng các ứng viên làm tập training set; 25% dữ liệu còn lại làm test set.

Chúng em sử dụng hàm sklearn.linear\_model.LogisticRegression trong thư viện sklearn cho bài thử nghiệm này.

Trước khi bắt đầu ta đảm bảo Python đã cài những gói sau:

* [pandas](https://pandas.pydata.org/) – được sử dụng để tạo DataFrame để lấy tập dữ liệu trong Python
* [sklearn](https://scikit-learn.org/stable/) – được sử dụng để xây dựng mô hình hồi quy logistic trong Python
* [seaborn](https://seaborn.pydata.org/) – được sử dụng để tạo [Ma trận nhầm lẫn](https://datatofish.com/confusion-matrix-python/)
* [matplotlib](https://datatofish.com/scatter-line-bar-charts-using-matplotlib/) – được sử dụng để hiển thị biểu đồ

Sau đó, ta sẽ cần phải nhập tất cả các gói như sau:

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

from sklearn import metrics

import seaborn as sn

import matplotlib.pyplot as plt

Tiếp theo, ta cần phải lấy tập dữ liệu từ các gói trên trong Python bằng cách sử dụng DataFrame

import pandas as pd

candidates = pd.read\_csv("export\_dataframe.csv", header=0)df = pd.DataFrame(candidates,columns= ['gmat', 'gpa','work\_experience','admitted'])

print (df)

Bây giờ, ta tạo Logistic Regression trong code. Đặt biến độc lập (đại diện là X), biến phụ thuộc (đại diện là Y):

X = df[['gmat', 'gpa','work\_experience']]

y = df['admitted']

Sau đó, áp dụng **train\_test\_split** . Ta đặt kích thước thử nghiệm thành **0,25** và do đó, thử nghiệm mô hình sẽ dựa trên 25% dữ liệu, trong khi đào tạo mô hình sẽ dựa trên 75% dữ liệu:

X\_train, X\_test, y\_train, y\_test = train\_test\_split (X, y, test\_size = 0.25, Random\_state = 0)

Áp dụng Logistic Regression như sau:

logistic\_regression = LogisticRegression ()

logistic\_regression.fit (X\_train, y\_train)

y\_pred = logistic\_regression.predict (X\_test)

Sau đó, sử dụng mã dưới đây để lấy [Confusion Matrix](https://datatofish.com/confusion-matrix-python/):

confusion\_matrix = pd.crosstab(y\_test, y\_pred, rownames=['Actual'], colnames=['Predicted'])

sn.heatmap(confusion\_matrix, annot=True)

Phần cuối cùng, ta in độ chính xác và vẽ [Confusion Matrix](https://datatofish.com/confusion-matrix-python/):

print('Accuracy: ',metrics.accuracy\_score(y\_test, y\_pred))

plt.show()

Gộp tất cả lại ta được một phần code hoàn chỉnh như sau:

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

from sklearn import metrics

import seaborn as sn

import matplotlib.pyplot as plt

candidates = pd.read\_csv("export\_dataframe.csv", header=0)df = pd.DataFrame(candidates,columns= ['gmat', 'gpa','work\_experience','admitted'])

#print (df)

X = df[['gmat', 'gpa','work\_experience']]

y = df['admitted']

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.25,random\_state=0)

logistic\_regression= LogisticRegression()

logistic\_regression.fit(X\_train,y\_train)

y\_pred=logistic\_regression.predict(X\_test)

confusion\_matrix = pd.crosstab(y\_test, y\_pred, rownames=['Actual'], colnames=['Predicted'])

sn.heatmap(confusion\_matrix, annot=True)

print('Accuracy: ',metrics.accuracy\_score(y\_test, y\_pred))

plt.show()

Chạy mã bằng Python và bạn sẽ nhận được [Confusion Matrix](https://datatofish.com/confusion-matrix-python/) sau với Độ chính xác là **0,8**

**Đi sâu hơn vào kết quả**

Bây giờ chúng ta in 2 thành phần trong mã Python:

* in (X\_test)
* in (y\_pred)

Đây là mã được sử dụng

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

candidates = pd.read\_csv("export\_dataframe.csv", header=0)

df = pd.DataFrame(candidates,columns= ['gmat', 'gpa', 'work\_experience', 'admitted'])

X = df[['gmat', 'gpa','work\_experience']]

y = df['admitted']

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.25,random\_state=0) #train is based on 75% of the dataset, test is based on 25% of dataset

logistic\_regression= LogisticRegression()

logistic\_regression.fit(X\_train,y\_train)

y\_pred=logistic\_regression.predict(X\_test)

print (X\_test) #test dataset

print (y\_pred) #predicted values

* 1. **Kiểm tra dự đoán cho tập dữ liệu mới**

Ta có một bộ dữ liệu với 5 ứng viên mới:



Mục tiêu của ta là sử dụng mô hình hồi quy logistic hiện tại để dự đoán liệu các ứng cử viên mới có được nhận hay không.

Tập hợp dữ liệu mới sau đó có thể được ghi lại trong Khung dữ liệu thứ hai được gọi là df2:

new\_candidates = {'gmat': [590,740,680,610,710],

'gpa': [2,3,7,3,3,2,3,3],

'work\_experience': [3,4,6,1,5]

}

df2 = pd.DataFrame (new\_candidates , columns = ['gmat', 'gpa', 'work\_experience'])

Và đây là mã hoàn chỉnh để có được dự đoán cho 5 ứng cử viên mới:

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

candidates = pd.read\_csv("export\_dataframe.csv", header=0)

df = pd.DataFrame(candidates,columns= ['gmat', 'gpa','work\_experience','admitted'])

X = df[['gmat', 'gpa','work\_experience']]

y = df['admitted']

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.25,random\_state=0) #in this case, you may choose to set the test\_size=0. You should get the same prediction here

logistic\_regression= LogisticRegression()

logistic\_regression.fit(X\_train,y\_train)

new\_candidates = {'gmat': [590,740,680,610,710],

'gpa': [2,3.7,3.3,2.3,3],

'work\_experience': [3,4,6,1,5]

}

df2 = pd.DataFrame(new\_candidates,columns= ['gmat', 'gpa','work\_experience'])

y\_pred=logistic\_regression.predict(df2)

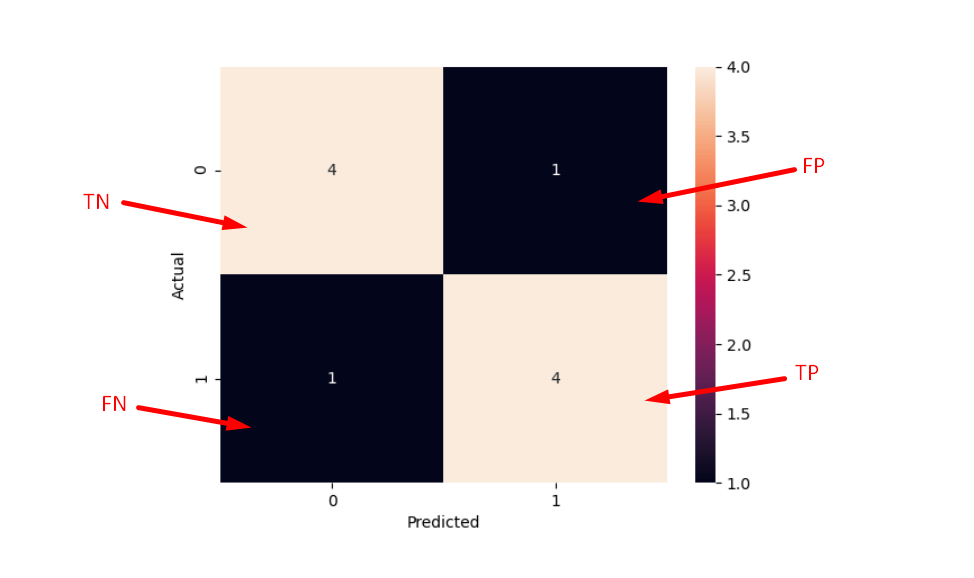
print (df2)

print (y\_pred)

# **CHƯƠNG III: KẾT QUẢ CHƯƠNG TRÌNH**

* 1. **Kết quả chương trình**
  2. **Ma trận Confusion Matrix kiểm tra độ chính xác**

Chạy mã bằng Python và ta sẽ nhận được [Confusion Matrix](https://datatofish.com/confusion-matrix-python/) sau với Độ chính xác là **0,8**

****

H 26 Ma trận confusion Matrix

**Ta có thể quan sát được từ ma trận:**

* **TP** = True Positives = 4
* **TN** = True Negatives = 4
* **FP** = False Positives = 1
* **FN** = False Negatives = 1

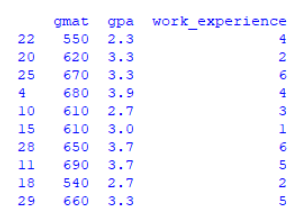
Sau đó ta cũng có thể nhận được độ chính xác bằng cách sử dụng công thức:

**Độ chính xác = (TP + TN) / Tổng** = (4 + 4) / 10 = 0,8

Do đó độ chính xác là 80% cho bộ thử nghiệm

* 1. **Phân lớp trên tập dữ liệu ban đầu**

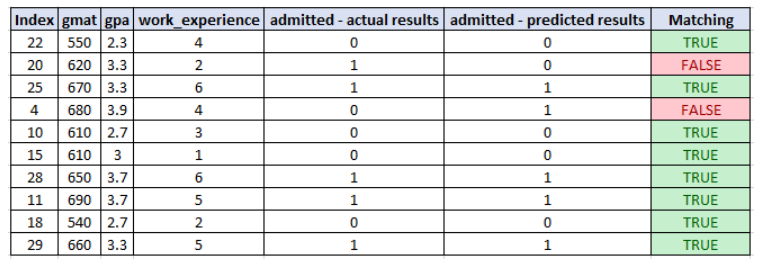
Tập dữ liệu ban đầu của chúng ta có 40 ứng viên. Vì chúng ta đặt kích thước thử nghiệm là 0,25, nên ma trận nhầm lẫn sẽ hiển thị kết quả cho 10 bản ghi (= 40 \* 0,25). Đây là 10 hồ sơ kiểm tra:



Dự đoán cũng được thực hiện cho 10 hồ sơ đó (trong đó 1 = thừa nhận, trong khi 0 = bị từ chối):

Screenshot_192

Trong bộ dữ liệu thực tế, ta sẽ thấy rằng đối với dữ liệu thử nghiệm, chúng ta đã nhận được kết quả chính xác 8 trên 10 lần:

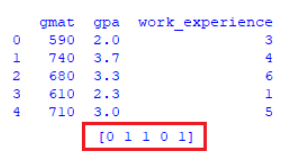


H 27 Bộ dữ liệu thực tế

Điều này phù hợp với mức độ chính xác 80%

* 1. **Kiểm tra cho tập dữ liệu mới**

Chạy mã và ta sẽ nhận được dự đoán sau:



Các ứng cử viên đầu tiên và thứ tư dự kiến ​​sẽ không được nhận, trong khi các ứng cử viên khác dự kiến ​​sẽ được nhận.

1. **Đánh giá**

Sau khi thử nghiệm với bộ cơ sở dữ liệu trên hoạt động bình thường, ổn định không phát sinh lỗi, kết quả phân lớp đạt 80%.

Đây là một kết quả khá cao đảm bảo giúp công việc phân lớp các ứng viên diễn ra tốt.

Những kết quả đạt được của đề tài:

- Đã tìm hiểu các kiến thức về Machine Learning.

- Các bài toán phân lớp đặc biệt là bài toán phân lớp nhị phân, cùng một số thuật toán phân lớp phổ biến. - Đã tìm hiểu kỹ thuật toán Logistic Regression

- Đã áp dụng thuật toán Logistic Regression để phân loại được ứng viên qua dữ liệu của họ với tỷ lệ cao 80 %.

Hạn chế: Trong qua trình làm báo cáo chúng em còn mắc phải nhiều vấn đề: Như tìm hiểu về thuật toán còn gặp nhiều khó khăn, kiến thức còn mới mẻ, thời gian còn hạn chế.

Hướng phát triền: em tiếp tục nghiên cứu các thuật toán Machine Leaning và áp dụng thuật toán Logicstic Regression trong các bài toán khác. Trong quá thực hiện báo cáo, chúng em đã cố gắng hết sức để tìm hiểu, học hỏi vì khả năng còn giới hạn, không tránh khỏi những sai sót, nên có thể chưa giải quyết được tất cả các vấn đề đặt ra. Chúng em rất mong nhận được sự thông cảm của quý thầy cô và các bạn. chúng em xin chân thành cảm ơn.

**TÀI LIỆU THAM KHẢO**

Sebastia Raschka, Python Machine Learning, 2015. [2] Vũ Hữu Tiệp, Machine Learning cơ bản, NXB Khoa Học Và Kỹ Thuật, 2018. [3] https://machinelearningcoban.com/2017/02/11/binaryclassifiers/ [4] https://machinelearningcoban.com/2017/01/27/logisticregression/ [5] https://en.wikipedia.org/wiki/Grayscale#Luma\_coding\_in\_video\_systems. [6] <https://machinelearningcoban.com/general/2017/02/06/featureengineering/>

<https://nguyenvanhieu.vn/khoa-hoc-lap-trinh-python/>