TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**XỬ LÝ NGÔN NGỮ TỰ NHIÊN**

**WORD REPRESENTATION**

*Người hướng dẫn*: **Thầy Lê Anh Cường**

*Người thực hiện*: **Nguyễn Duy Hàn Lâm (MSHV: 196005004)**

Khoá  **: 2019**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2020**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**XỬ LÝ NGÔN NGỮ TỰ NHIÊN**

**WORD REPRESENTATION**

*Người hướng dẫn*: **Thầy Lê Anh Cường**

*Người thực hiện*: **Nguyễn Duy Hàn Lâm (MSHV: 196005004)**

Khoá  **: 2019**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2020**

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng tôi và được sự hướng dẫn của thầy Lê Anh Cường. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 30 tháng 04 năm 2020*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Nguyễn Duy Hàn Lâm*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

MỤC LỤC

[PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN ii](#_Toc38785913)

[MỤC LỤC 1](#_Toc38785914)

[CHƯƠNG 1: DỊCH MÁY NEURAL 4](#_Toc38785915)

[1.1 Giới thiệu về dịch máy neural 4](#_Toc38785916)

[1.2 Recurrent Neural Network trong mô hình 5](#_Toc38785917)

[1.2.1 Cổng update 7](#_Toc38785918)

[1.2.2 Cổng reset 7](#_Toc38785919)

[1.2.3 Xuất kết quả 8](#_Toc38785920)

[1.3 Convolutional Neural Network trong mô hình 9](#_Toc38785921)

[1.4 Mô hình Encoder-Decoder 11](#_Toc38785922)

[1.4.1 Word embedding 12](#_Toc38785923)

[1.4.2 Encoder 13](#_Toc38785924)

[1.4.3 Decoder 13](#_Toc38785925)

[1.4.4 Linguistic Input Feature 16](#_Toc38785926)

[1.5 Tóm tắt chương 1 18](#_Toc38785927)

[CHƯƠNG 2: THỰC HIỆN MÔ PHỎNG ENCODER-DECODER 19](#_Toc38785928)

[2.1 Chuẩn bị nguồn dữ liệu và thư viện 19](#_Toc38785929)

[2.2 Xử lý dữ liệu và tạo các đối tượng quản lý 19](#_Toc38785930)

[2.3 Xây dựng encoder-decoder 23](#_Toc38785931)

[2.4 Huấn luyện 27](#_Toc38785932)

[2.5 Đánh giá 33](#_Toc38785933)

[2.6 Kiểm thử 35](#_Toc38785934)

[2.7 Chạy chương trình và kết quả 36](#_Toc38785935)

[2.8 Tóm tắt 37](#_Toc38785936)

[TÀI LIỆU THAM KHẢO 39](#_Toc38785937)

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

**DANH MỤC HÌNH**

[Hình 1.1. Sơ lược cấu tạo hoạt động của RNN [1] 5](#_Toc38785871)

[Hình 1.2. Cấu trúc neural theo GRU [2] 6](#_Toc38785872)

[Hình 1.3. Mô tả cách tính của cổng update [2] 7](#_Toc38785873)

[Hình 1.4. Mô tả cách tính cổng reset [2] 8](#_Toc38785874)

[Hình 1.5. Mô tả cách tính [2] 8](#_Toc38785875)

[Hình 1.6. Cấu trúc của CNN trong encoder [3] 9](#_Toc38785876)

[Hình 1.7. Minh họa cách hoạt động của hàm activation ở mỗi node [3] 10](#_Toc38785877)

[Hình 1.8. Minh họa cho một số cấu trúc của CNN [3] 11](#_Toc38785878)

[Hình 1.9. Mô hình encoder-decoder cơ bản [3] 12](#_Toc38785879)

[Hình 1.10. Cấu trúc của một bộ GRU-RNN encoder [4] 13](#_Toc38785880)

[Hình 1.11. Minh họa cho decoder [4] 14](#_Toc38785881)

[Hình 1.12. Cơ chế attention [5] 15](#_Toc38785882)

[Hình 1.13. Cấu trúc chi tiết của một attention [6] 15](#_Toc38785883)

[Hình 2.1. Các thư viện cần chuẩn bị 19](#_Toc38785884)

[Hình 2.2. Các thư viện phục vụ linguistic input feature 19](#_Toc38785885)

[Hình 2.3. Class quản lý dữ liệu 20](#_Toc38785886)

[Hình 2.4. Xử lý và chuẩn hóa dữ liệu 21](#_Toc38785887)

[Hình 2.5. Hàm đọc dữ liệu 22](#_Toc38785888)

[Hình 2.6. Chuẩn hóa số lượng từ của câu và lemmas hóa dữ liệu 22](#_Toc38785889)

[Hình 2.7. Hàm đọc dữ liệu và thực hiện nạp dữ liệu vào chương trình 23](#_Toc38785890)

[Hình 2.8. Xây dựng bộ encoder 23](#_Toc38785891)

[Hình 2.9. Xây dựng bộ decoder cơ bản 24](#_Toc38785892)

[Hình 2.10. Cấu trúc hoạt động của attention 25](#_Toc38785893)

[Hình 2.11. Class decoder sử dụng attention (hàm khởi tạo của class) 26](#_Toc38785894)

[Hình 2.12. Hàm khởi tạo tiến trình và hàm feedforward 27](#_Toc38785895)

[Hình 2.13. Các hàm tạo vector input và output cho quá trình huấn luyện 28](#_Toc38785896)

[Hình 2.14. Hàm thực hiện huấn luyện 29](#_Toc38785897)

[Hình 2.15. Giai đoạn tiền encoder 29](#_Toc38785898)

[Hình 2.16. Giai đoạn chạy encoder 30](#_Toc38785899)

[Hình 2.17. Giai đoạn hậu encoder và tiền decoder 30](#_Toc38785900)

[Hình 2.18. Giai đoạn chạy decoder 30](#_Toc38785901)

[Hình 2.19. Giai đoạn hậu decoder 31](#_Toc38785902)

[Hình 2.20. Hàm thông báo lượng thời gian chương trình đã và đang chạy 31](#_Toc38785903)

[Hình 2.21. Hàm vẽ biểu đồ thay đổi của giá trị loss 32](#_Toc38785904)

[Hình 2.22. Hàm huấn luyện tổng quát 33](#_Toc38785905)

[Hình 2.23. Hàm thực hiện đánh giá 34](#_Toc38785906)

[Hình 2.24. Hàm thực hiện chọn đánh giá ngẫu nhiên 35](#_Toc38785907)

[Hình 2.25. Các hàm phục vụ trình bày kết quả 35](#_Toc38785908)

[Hình 2.26. Tiến hành huấn luyện mô hình 36](#_Toc38785909)

[Hình 2.27. Kết quả của từng đợt huấn luyện 36](#_Toc38785910)

[Hình 2.28. Chạy thử 1 câu ví dụ 37](#_Toc38785911)

[Hình 2.29. Kết quả chạy thử 1 câu ví dụ 37](#_Toc38785912)

TÓM TẮT NỘI DUNG

Học máy là một lĩnh vực quan trọng trong thời đại công nghệ hiện nay. Trong đó dữ liệu là 1 phần cốt lõi của lĩnh vực vì chương trình sẽ phải dựa trên nguồn tập dữ liệu này mới có thể đào tạo, học và từ đó mới sinh ra được mô hình để có thể từ đó giải quyết bài toán.

Dữ liệu càng lớn, các đối tượng trong dữ liệu càng chứa nhiều thuộc tính (hay còn gọi là chiều dữ liệu) sẽ càng khiến chương trình trở nên nặng nề, khó đào tạo và từ đó sẽ khó có thể đào tạo được ra một mô hình chuẩn xác để giải quyết bài toán. Bằng các kĩ thuật tính toán ta có thể hồi quy đưa số chiều của dữ liệu lớn trở nên nhỏ hơn nhằm giảm bớt gánh nặng trong việc đào tạo ra mô hình. Kĩ thuật tính toán này được gọi là shrinkage method (các phương thức co rút chiều dữ liệu).

Tài liệu này bao gồm các phần chính:

* Giới thiệu lại về mô hình tuyến tính.
* Các phương pháp hồi quy thô.
* Các phương pháp hồi quy chính (shrinkage method).
* Áp dụng vào chương trình mẫu.

CHƯƠNG 1: BIỂU DIỄN TỪ VỰNG

* 1. Mô hình tuyến tính

Mô hình tuyến tính là mô hình có hàm số biểu diễn cho phương trình đường thẳng:

Trong đó:

* là giá trị mà mô hình tính toán được.
* là giá trị chặn (intercept) hay còn gọi là bias (phần bù đại số).
* là tập vector dữ liệu đầu vào.

Ta có thể thế và rút gọn biểu thức (1) về dạng:

Từ đó ta có thể thấy là tích vô hướng, và kết quả của nó nó tập chứa K các vector, sẽ là một ma trận hệ số có kích thước là . Trong không gian (p + 1) chiều ứng với tập dữ liệu đầu vào và tập dữ liệu đầu ra thì tập sẽ là 1 siêu phẳng. Nếu trong X có chứa hằng số thì siêu phẳng này sẽ chứa cả không gian gốc lẫn các không gian con liên quan. Nếu trong X không có hằng số thì siêu phẳng sẽ là một tập hợp các phép biến đổi cắt trục Y tại điểm .

Xét (1) trong mặt phẳng p chiều, ta có thể thấy nó được tổng quát hóa bằng hàm số , từ đó ta được đạo hàm của nó là là một tập hợp các điểm hình thành nên một nên một đường thẳng có dốc hướng lên.

Việc hiệu chỉnh mô hình đó chính là ta sẽ thay đổi đường thẳng sao cho khoảng cách từ tất các điểm dữ liệu đến đường thẳng là nhỏ nhất hay nói cách khác là làm sao cho giá trị của phương trình sau càng nhỏ càng tốt:

là 1 hàm bậc hai chính vì vậy nó luôn tồn tại giá trị nhỏ nhất, tuy nhiên trong 1 vài trường hợp giá trị này không phải duy nhất. ta có thể biểu diễn RSS về dạng phép tính của ma trận thành:

Trong đó:

* X là ma trận dữ liệu đầu vào có kích thước là . Mỗi dòng là 1 vector dữ liệu đầu vào.
* là tập chứa N vector kết quả mà mô hình tính toán được từ tập dữ liệu huấn luyện.

Ta đạo hàm (3) theo , ta được phương trình:

Từ (4) ta giải phương trình, ta ra được giải pháp ta cần tìm:

Giá trị mà chương trình tính toán được từ input thứ i sau khi hiệu chỉnh sẽ có dạng:

* 1. Mô hình thống kê

Cho là tập dữ liệu đầu vào chứa các giá trị số thực ngẫu nhiên.

Cho là tập dữ liệu kết quả chứa các giá trị số thực ngẫu nhiên.

Đồng thời ta có:

Trong đó:

Ta cần đi tìm một hàm f có khả năng từ các dữ liệu X dự đoán thành các dữ liệu Y. Sai số là điều không tránh khỏi vì vậy ta cần một hàm tính sai số (expected prediction error – EPE):

Ta có thể viết thành:

Kết hợp biểu thức (6) và (8) ta được:

Ta thấy ta có thể giảm bớt giá trị EPE theo từng điểm của f:

Từ đó ta tìm được giải pháp:

Trong đó:

* E là giá trị kỳ vọng

Biểu thức (12) được gọi là biểu thức kỳ vọng có điều kiện, còn được gọi là hàm hồi quy. Các giá trị dự đoán tốt nhất của Y tại bất kỳ điểm X = x nào được gọi là các giá trị trung bình có điều kiện, và giá trị tốt nhất sẽ được tính bằng sai số bình phương trung bình.

Trong bài toán the neareast-neighbor, các phương thức luôn cố gắng tìm cách triển khai trực tiếp quá trình của nó trên dữ liệu huấn luyện. Với mỗi điểm x đầu vào, ta có thể sẽ phải đi tính giá trị trung bình của tất cả các giá trị yi dự đoán được ứng với từng xi của nó.

Trong đó:

* Ave là hàm tính giá trị trung bình.
* là tập chứa các điểm k trong tập dữ liệu gần với x nhất.

Với công thức (13), với , khi thì , dẫn đến .

Khi kích thước số chiều p lớn, sẽ dẫn đến việc kích thước của ma trận sẽ lớn. Mặc dù sự hội tụ vẫn diễn ra như quy luật trên, tuy nhiên tốc độ hội tụ của nó lại diễn ra chậm.

Vì vậy ta đi xây dựng một mô hình hồi quy tuyến tính để giải quyết vấn đề này. Giả sử mô hình hồi quy tuyến tính sẽ có dạng:

Ta kết hợp biểu thức (14) với biểu thức (5) và (8) ta được:

Tuy nhiên về bản chất thì 2 cái hoàn toàn khác nhau, trong biểu thức (5) cố gắng dự đoán giá trị dựa trên một hàm tuyến tính chung tổng thể, còn trong biểu thức (8) lại cố gắng dự đoán bằng các giá trị tính toán nội bộ. Vì thế việc làm này sẽ khiến quá trình tính toán sẽ không ổn định mặc dù ta giải quyết được bớt một phần của vấn đề liên quan đến kích thước của chiều dữ liệu.

Giả sử:

Trong đó:

* là một biến lỗi ngẫu nhiên có và độc lập với X.
* phụ thuộc vào X thông qua điều kiện trung bình của .

Việc thêm sẽ giúp mô hình có giá trị gần hơn với giá trị thật. Vì đối với hầu hết các hệ thống, các cặp (X, Y) sẽ không có mối quan hệ xác định . Nói chung, ngoài giá trị tính toán được từ mô hình thì các yếu tố không liên quan cũng sẽ đóng góp một phần vào giá trị Y. Và chắc chắn giá trị lỗi phát sinh từ mô hình cũng sẽ đóng góp vào Y. Các giá trị này này ta sẽ ước lượng và quy kết bằng giá trị .

* 1. Đánh đổi giữa Bias và Variance

Ta xét lại mô hình phân loại hồi quy k-nearest-neighbor.

Giá trị thật Y được cấu thành như sau:

Trong đó:

* là một biến lỗi ngẫu nhiên có và độc lập với X.

Ta có giá trị lỗi kỳ vọng (Expected Prediction Error – EPE) tại biến đầu vào như sau:

Trong biểu thức trên giá trị là phương sai của giá trị lỗi sinh ra từ giá trị mới phát sinh từ mô hình. Mức độ phân tán này vượt khỏi tầm kiểm soát của ta.

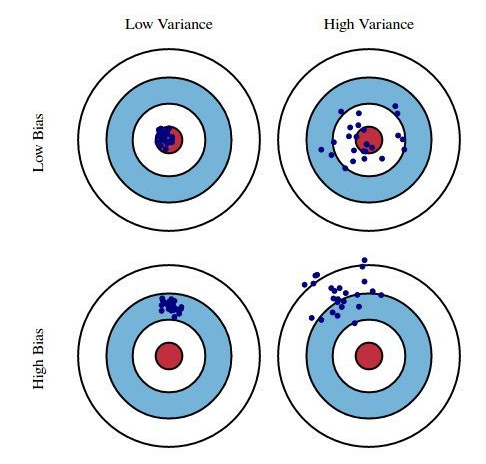
Hai giá trị còn lại trong biểu thức là Bias và Var (Variance).

Bias của các thuộc tính được tính như sau:

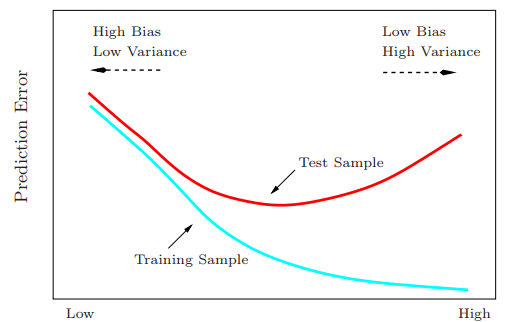
Bias là sự khác biệt bình phương giữa giá trị trung bình của và giá trị kỳ vọng của ước tính , trong đó giá trị kỳ vọng trung bình được tính ngẫu nhiên từ các dữ liệu huấn luyện. Bias sẽ tăng tỉ lệ thuận với k, nếu việc tính toán trở nên trơn tru. Đối với k nhỏ, một vài tập lân cận gần nhất sẽ có các giá trị gần với , vì vậy trung bình của chúng sẽ gần với . Khi k càng lớn các tâm của các cụm sẽ có khoảng cách với điểm càng lớn. Nói chung Bias biểu thị cho sự khác biệt giữa giá trị thực với giá trị tính toán được từ mô hình. Hay nói cách khác Bias biểu thị cho mức độ sai số của các giá trị dự đoán

Var (Variance) biểu thị cho mức độ phân tán của các giá trị tính toán được từ mô hình.

Như vậy, variance của các thuộc tính được tính như sau:



1. Minh họa về Bias và Variance [2]



1. Sự ảnh hưởng của Bias và Variance trên các tập dữ liệu [1]

Rõ ràng, cả sai lệch (bias) và phương sai (variance) đều có thể gây hại cho hiệu suất dự đoán của mô hình nếu chúng quá lớn. Hồi quy tuyến tính có thể sẽ bị ảnh hưởng bởi phương sai mặc dù chúng có độ sai lệch thấp. Điều này đặc biệt sẽ có thể xảy ra trong các mô hình có nhiều đặc tính (feature), hoặc các đặc tính của dữ liệu có mức độ liên quan với nhau cao.

Các phương pháp phân mảnh và chuẩn hóa sẽ được giới ở các phần sau có thể sẽ giúp làm giảm được phương sai của các giá trị dự đoán, tuy nhiên chúng sẽ làm tăng sai số nhưng thay vào đó giá trị sai số tổng thể lại giảm xuống.

* 1. Tóm tắt chương 1

Trong chương 1 ta đã ôn lại các khái niệm cơ bản về mô hình tuyến tính.

Ngoài ra ta còn biết các khái niệm về Bias và Variance. Ta còn biết thêm về mức độ ảnh hưởng của chúng lên mô hình. Ngoài ra ta hiểu được sự đánh đổi giữa Bias và Variance.

CHƯƠNG 2: MÔ HÌNH HỒI QUY VÀ PHƯƠNG PHÁP HỒI QUY CƠ BẢN

2.1 Mô hình hồi quy tuyến tính

Mô hình hồi quy tuyến tính sẽ có dạng giống với mô hình tuyến tính thuần:

Mô hình tuyến tính giả định rằng hàm hồi quy là tuyến tính hoặc mô hình tuyến tính là một hàm xấp xỉ hợp lý. Ở đây, là các tham số hoặc hệ số chưa biết và các biến có thể đến từ các nguồn khác nhau:

* Giá trị dữ liệu đầu vào gốc.
* Các giá trị tính toán thông qua từ phép tính biến đổi (log, bình phương, căn, …).
* Các phép tính mở rộng (lũy thừa), chẳng hạn như , , dẫn đến biểu diễn đa thức.
* tương tác giữa các biến, ví dụ, .

Cho dù đầu vào thuộc dạng nào đi chăng nữa vẫn không ảnh hưởng đến tuyến tính của mô hình.

Phương trình lỗi của mô hình:

Trong đó:

* tập chứa các đối tượng dữ liệu đầu vào.
* tập chứa các giá trị thực mong muốn ở đầu ra của mô hình.
* tập chứa các thuộc tính cho từng đối tượng đầu vào .

Ta có thể biểu diễn phương trình lỗi dưới dạng ma trận như sau:

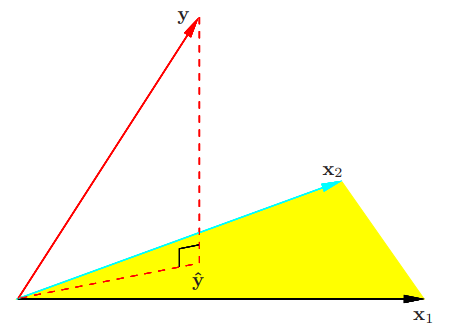
Sử dụng gradiant descent, ta đạo hàm phương trình lỗi theo :

Từ (22), ta lập được phương trình:

Ta giải phương trình (24), ta tìm được giải pháp cho mô hình:

Giá trị dự đoán trên đã hiệu chỉnh:

Với tập các vector đầu vào X, ta luôn muốn mô hình tính được các vector dự đoán trùng với các vector giá trị thật . Hay nói cách khác về mặt toán học ta luôn muốn và là 2 vector trực giao (tích vô hướng của 2 vector là 0) trong không gian tập các vector đầu vào X.



1. Vector dự đoán và vector giá trị thật là cặp vector trực giao trong không gian có tập vector dữ liệu đầu vào gồm 2 vector [1]

Tuy nhiên điều ta muốn chỉ xảy ra khi các thuộc tính của dữ liệu (các cột của ma trận X) không độc lập tuyến tính hay nói cách khác hạng của ma trận X không bảo toàn.

Ta giả định rằng các giá trị thật không quan hệ mật thiết với nhau và phương sai của chúng là hằng số . Phương sai của tập các trọng số được tính như sau:

Phương sai của các giá trị dự đoán được tính như sau:

sẽ giúp .

Từ mô hình trong biểu thức (19) ta có thể suy ra được mô hình cho trường hợp biến đầu ra phụ thuộc có điều kiện vào biến đầu vào như sau:

Trong đó:

* , là phân phối chi bình phương với bậc tự do .

Giá trị thử nghiệm giả thuyết (hypothesis) cho mỗi giá trị được tính thông qua hệ số chuẩn (Z-score):

Trong đó:

* là giá trị ở vị trí thứ j trong ma trận chéo

2.2 Lựa chọn các tập con

Có hai lý do tại sao chúng ta thường không hài lòng về việc hiệu chỉnh hệ số thông qua hàm tính lỗi RSS bằng biểu thức (25):

* Đầu tiên là độ chính xác dự đoán: việc hiệu chỉnh bằng RSS sẽ giúp ta có độ lệch thấp nhưng phương sai lớn giữa các các giá trị dự đoán. Độ chính xác dự đoán đôi khi có thể được cải thiện bằng cách thu nhỏ (giới hạn bớt một số thuộc tính) hoặc ta lược một số hệ số của thuộc tính về 0. Bằng cách làm như vậy, chúng ta sẽ hy sinh một chút về sai số để giảm phương sai của các giá trị dự đoán và do đó có thể cải thiện độ chính xác dự đoán tổng thể.
* Lý do thứ hai: Với một số lượng lớn các thuộc tính dự đoán, chúng ta thường muốn xác định một tập hợp con nhỏ hơn có mức độ ảnh hưởng lớn đến bài toán. Ta chỉ tập trung vào các thuộc tính trọng tâm của bài toán. Để có thể cải thiện độ chính xác dự đoán tổng thể, chúng ta sẵn sàng hy sinh một số thuộc tính còn lại.

Tập hợp các thuộc tính mà ta lấy ra như vậy được gọi là các tập con.

Trong phần này, chúng ta sẽ tìm hiểu một số cách tiếp cận để lựa chọn các tập hợp con và ta sẽ sử dụng các tập con này cho hồi quy tuyến tính.

Bằng việc lựa chọn các tập hợp con, chúng ta chỉ giữ lại một tập hợp con của các biến và loại bỏ phần còn lại khỏi mô hình. RSS được sử dụng để tính sai số của các tập con. Từ các sai số này ta sẽ có các chiến lược khác nhau để chọn ra tập hợp con phù hợp.

Hồi quy tuyến tính tốt nhất lựa chọn từ các tập hợp con có kích thước k ∈ {0,1,2, ..., p} là tập con có giá trị RSS nhỏ nhất. Cách lựa chọn tập con như này được gọi là lựa chọn tập con tốt nhất.

2.3 Tóm tắt chương 2

Trong chương 2 ta đã tìm hiểu được thế nào là 1 mô hình hồi quy tuyến tính cơ bản. Sau khi đưa dữ liệu đi vào mô hình, mô hình sẽ tính toán và đưa ra các giá trị dự đoán. Ta sẽ sử dụng lại Residual Sum of Squares (RSS hay hàm L2) để tính mức độ sai số của mô hình. Từ đó ta sẽ chỉnh lại các hệ số của các thuộc tính bằng guardian descent RSS, sau đó ta thực hiện phép trực giao giữa vector dự đoán với giá trị thực để hiệu chỉnh cho các hệ số.

Đồng thời qua chương 2 ta tìm hiểu một phương pháp thô nhất, cơ bản nhất của phương pháp rút gọn số chiều của bài toán gọi là lựa chọn các tập con. Trong các bài toán có số chiều lớn, mặc dù có bias thấp tuy nhiên sẽ có thể xảy ra giá trị phương sai của chúng là rất lớn. Điều này sẽ làm ảnh hưởng tới kết quả dự đoán tổng thể. Vì vậy ta sẽ lựa chọn ra các thuộc tính có tính ảnh hưởng cao của bài toán và loại bỏ các phần còn lại của dữ liệu với hy vọng phương sai giữa các dự đoán sẽ giảm và độ chính xác tổng thể sẽ tăng.

CHƯƠNG 3: SHRINKAGE METHOD

Bằng cách giữ lại một tập hợp con của các thuộc tính quan trọng và loại bỏ phần còn lại, phương pháp lựa chọn tập con có thể tạo ra một mô hình có giá trị sai số thấp hơn so với mô hình sử dụng dữ liệu gốc và đầy đủ các thuộc tính. Tuy nhiên, vì thuộc tính của các tập con này là các biến rời rạc, nên phương sai của nó sẽ cao và do đó vẫn không làm giảm lỗi dự đoán của mô hình đầy đủ. Các phương pháp shrinkage sẽ giúp các biến liên tục và không phải chịu đựng nhiều từ phương sai lớn.

3.1 Ridge Regression

Ridge regression thu nhỏ chiều của dữ liệu bằng cách áp dụng một giá trị penalty trên kích thước của chúng. Ta sẽ hiệu chỉnh hệ số dựa trên biểu thức mới này.

Sai số của ridge regression được tính như sau:

Trong đó:

* (trong lập trình ta sẽ thay bằng ) là một tham số phức tạp có khả năng điều khiển lượng co rút của bài toán: giá trị càng lớn thì lượng co rút sẽ càng cao. Lượng càng cao thì hệ số của các thuộc tính sẽ giảm dần về 0.

Ý tưởng chèn thêm một đại lượng nhằm làm giảm mức độ sai số của các hệ số được gọi là weight decay.

Hiệu chỉnh các hệ số theo biểu thức:

Hoặc ta có thể viết biểu thức (32) thành:

Phụ thuộc vào:

Có một sự tương ứng một-một giữa các tham số λ và t. Khi có nhiều biến tương quan trong mô hình hồi quy tuyến tính, các hệ số của chúng có thể gây hiệu suất kém trong việc tính toán, ngoài ra nó có thể làm giá trị phương sai trở nên cao. Một hệ số dương cực lớn trên một biến có thể bị hủy bởi một hệ số âm lớn của thuộc tính tương quan với nó. Bằng cách áp đặt một ràng buộc kích thước đối với các hệ số dưới một lượng t cho phép thì vấn đề này sẽ được giảm bớt.

Các giải pháp ridge regression sẽ không tương đương với nhau do phải chịu các phép tính tỉ lệ trên các giá trị dữ liệu đầu vào, và do đó, người ta thường chuẩn hóa dữ liệu đầu vào trước khi giải (32). Ngoài ra, ta có thể thấy hệ số không nằm trong sự kiểm soát của λ và t, hay nói cách ở mỗi lần tính nó sẽ được coi là 1 hằng số c ở mỗi lần tính. Việc thêm hằng số c ở mỗi lần tính như vậy sẽ làm ảnh hưởng lớn đến tổng thể của bài toán. Vì vậy ở mỗi bước thay vì ta lấy giá trị ta sẽ phải thay bằng (các input này được gọi là các centered input). Ta ước lượng bởi . Các hệ số còn lại sẽ được ước lượng bằng giá trị . Do ta đã giải quyết vấn đề về nên thay vì bài toán là p+1 chiều thì bài toán của ta chỉ còn là p chiều.

Ta biểu diễn (31) về dạng ma trận:

Hiệu chỉnh các trọng số theo (33) sẽ là:

Trong đó: I là ma trận đơn vị.

Với việc thêm giá trị vào khiến ridge regression trở về thành 1 hàm tuyến tính của y. Vì vậy ta phải cộng thêm vào trước khi thực hiện phép nghịch đảo.

Trong trường hợp các vector đầu vào trực giao với nhau, thì ta chỉ cần hiệu chỉnh hệ số theo tỉ lệ sau:

Ridge regression cũng có thể được coi là giá trị trung bình hoặc là 1 dạng của posterior distribution, đi kèm với nó là ta sẽ chọn prior distribution sao cho phù hợp.

Trong trường hợp đó ta giả định rằng các giá trị thực và các hệ số là các phân phối độc lập nằm trong khoảng . Giá trị log của mật độ phân phối của sẽ bằng với giá trị của nó trong biểu thức (32) với điều kiện mà ta giả sử các cận của bài toán là , và giá trị sẽ là . Do phân phối là phân phối Guassian, nên kết quả ước lượng sẽ là 1 dạng của posterior distribution hay còn gọi là posterior mean.

Một kĩ thuật thường được sử dụng trong ridge regression là SVD (Singular Value Decomposition). Ma trận X khi áp dụng SVD sẽ có dạng:

Trong đó:

* U và V là 2 ma trận trực giao (orthogonal matrix).
* D là ma trận đường chéo có kích thước là . Với các giá trị đường chéo được gọi là các giá trị dị biến của X. Nếu có một giá trị trong D, thì X sẽ được coi là dị biến (singular).

Để thuận tiện ta sẽ thực hiện một vài phép tính ma trận ta hay gặp:

Do U là ma trận trực giao nên ta được . Ta được:

Do D là ma trận đường chéo có kích thước là nên . Do đó ta được:

Biểu thức (35) hay được gọi là eigen decomposition. Các eigen-vector trong V còn được gọi là các thành phần chính (principal component) biểu thị cho các hướng của ma trận X. Trong đó hướng có thuộc tính có phương sai mẫu lớn nhất trong số tất cả các tổ hợp tuyến tính chuẩn hóa của các cột của X.

Ta nhân hai vế của biểu thức (25) với X và áp dụng X ở biểu thức (34) ta được:

Ta làm tương tự

3.2 The Lasso

3.3 Elastic Net

3.4 Least Angle Regression

3.5 Principle Components Regression

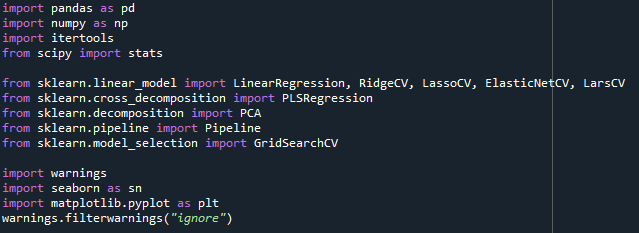
3.6 Partial Least Squares

3.7 Tổng kết lý thuyết

CHƯƠNG 4: LẬP TRÌNH TRÊN PYTHON

4.1 Dữ liệu

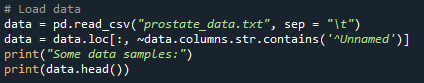
Trước hết ta cần chuẩn bị một số thư viện cho chương trình.



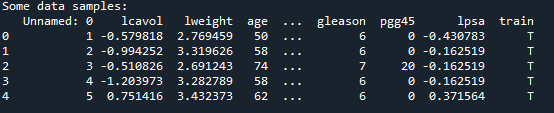
1. Chuẩn bị thư viện cho chương trình

Dữ liệu cho ví dụ này đến từ một nghiên cứu của Stamey et al. (1989). Họ đã kiểm tra mối tương quan giữa mức độ kháng nguyên (antigen) đặc hiệu tuyến tiền liệt (prostate) và một số biện pháp lâm sàng ở những người đàn ông sắp được phẫu thuật cắt bỏ triệt để tuyến tiền liệt. Các biến số (thuộc tính) gồm giá trị log của khối lượng ung thư (**lcavol**), giá trị log của trọng lượng của tuyến tiền liệt (**lweight**), tuổi, giá trị log của lượng kích thước thay đổi lành tính (không liên quan đến ung thư) của tuyến tiền liệt (**lbph –** log of benign prostatic hyperplasia), mức độ xâm lấn túi tinh (**svi** - seminal vesicle invasion), giá trị log của mức độ thâm nhập của nang (**lcp** – log of capsular penetration), giá trị Gleason (**gleason**) và phần trăm giá trị Gleason đạt mức 4 hoặc 5 (**pgg45**).

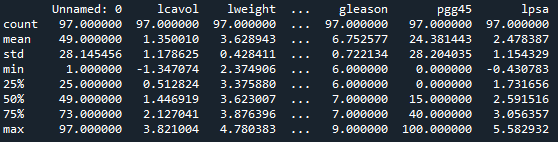
Đầu tiên ta sẽ nạp dữ liệu vào chương trình:



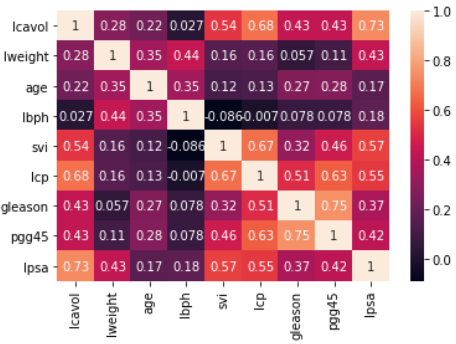
1. Thực hiện nạp dữ liệu vào chương trình



1. Một vài dữ liệu mẫu



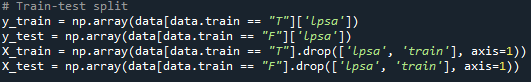
1. Một vài giá trị thống kê cơ bản về dữ liệu



1. Ma trận biểu thị sự tương quan giữa các thuộc tính

Từ hình 4.5 ta có thể thấy rằng khối lượng ung thư (lcavol) và mức độ thâm nhập của nang (lcp) có quan hệ mật thiết với chỉ số ung thư tuyến tiền liệt (lpsa). Đồng thời giữa 2 đại lượng lcavol và lcp cũng có mối quan hệ mật thiết với nhau.

Từ cột train trong dữ liệu, ta sẽ phân dữ liệu thành 2 tập dữ liệu là tập train (tập huấn luyện) ứng với giá trị T trong cột – tập này ta sẽ dùng để đi huấn luyện, xây dựng mô hình. Tập còn lại là tập test ứng với giá F trong cột train – tập này ta sẽ dùng để kiểm thử cũng như tính toán sai số của mô hình.

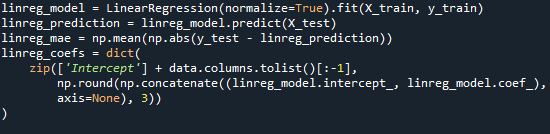


1. Chia dữ liệu thành tập train và tập test

Nhiệm vụ của ta đó là từ mô hình ta có thể xác định được mức độ rủi ro về ung thư cho từng đối tượng.

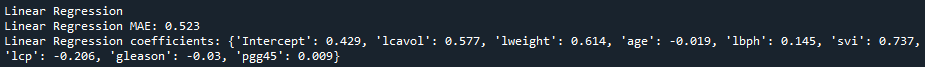
4.2 Triển khai mô hình hồi quy tuyến tính cơ bản

Đầu tiên ta sẽ dự đoán chỉ số lspa trên mô hình hồi quy tuyến tính cơ bản để có cái nhìn tổng quan về bài toán trước khi đi vào các phương pháp shrinkage.



1. Thực hiện dự đoán trên mô hình hồi quy tuyến tính cơ bản

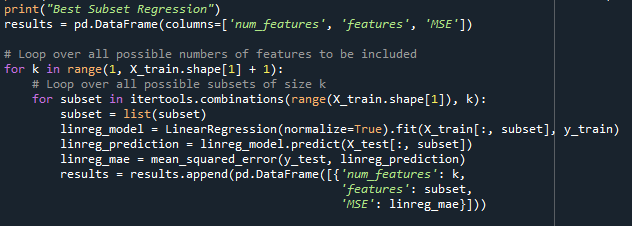
Ta thu được kết quả sau khi chạy chương trình:



1. Kết quả thu được sau khi thực thi mô hình hồi quy tuyến tính

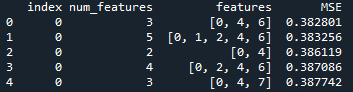
4.3 Triển khai phương pháp lựa chọn các tập con

Đầu tiên ta sẽ tiến hành chọn tổ hợp các thuộc tính (tạo các tập con các thuộc tính). Sau đó ta sẽ tiến hành đi xây dựng mô hình dựa trên các thuộc tính này. Ta sẽ đem các mô hình này đi dự đoán, sau đó ta tính mức độ sai số RSS.



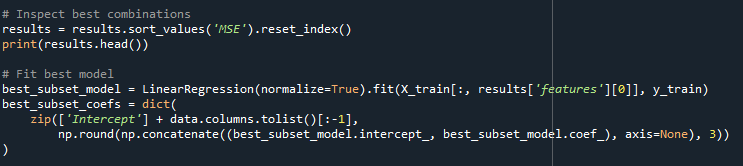
1. Tạo và tính lỗi RSS của từng tập con thông qua huấn luyện

Ta cho chạy chương trình ta thu được tập hợp các tập con kèm theo là giá trị lỗi RSS tương ứng.



1. Bốn tập con có sai số RSS thấp nhất

Sau đó ta sắp xếp các tập con dựa trên giá trị lỗi, từ đó ta chọn ra tập con có giá trị lỗi thấp nhất để đi xây dựng mô hình cuối cùng.



1. Chọn tập con có sai số thấp nhất và đem đi xây dựng mô hình

Sau khi chạy mô hình ta thu được kết quả cuối cùng:



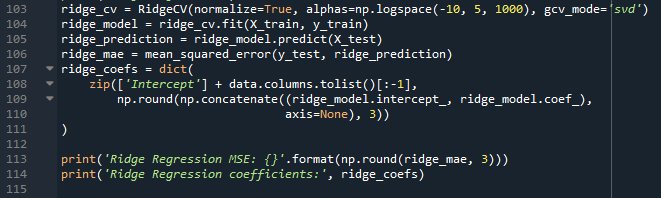
1. Kết quả của mô hình được huấn luyện tập con tốt nhất

4.4 Triển khai Ridge Regression

Ta cho chạy từ đến ứng với hàm np.logspace(start, end, số mẫu ).

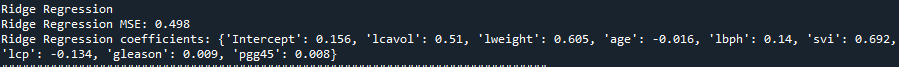
Sau đó ta sẽ tiến hành đi xây dựng mô hình ridge regression cho từng từ đó ta chọn ra mô hình có sai số thấp nhất.

Thư viện sklearn cho phép ta thực hiện quá trình này bằng hàm RidgeCV.



1. Xây dựng, chạy, xuất kết quả cho mô hình ridge regression

Ta thu được kết quả như sau:



1. Kết quả của mô hình ridge regression

4.5 Triển khai Lasso

4.6 Triển khai Elastic Net

4.7 Triển khai Least Angle Regression

4.8 Triển khai Principle Components Regression

4.9 Tổng kết tài liệu

TÀI LIỆU THAM KHẢO

**Tiếng Anh:**

1. Trevor Hastie, Robert Tibshirani, Jerome Friedman, 2008, The Elements of Statistical Learning: Data Mining, Inference, and Prediction (Second Edition).
2. <https://towardsdatascience.com/a-comparison-of-shrinkage-and-selection-methods-for-linear-regression-ee4dd3a71f16>