ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH
KHOA ĐIỆN-ĐIỆN TỬ
BỘ MÔN VIỄN THÔNG

TIỂU LUẬN
XỬ LÝ ẢNH – EE3035

BỘ PHÂN LỚP NAVIE BAYES

GV hướng dẫn: **TS. VÕ TUẤN KIỆT**
SV thực hiện : Nguyễn Trí Dũng - 1910977
Nguyễn Huỳnh Nhật Duy - 1514017
Lê Phan Gia Nghiêm - 1710200

TP. HỒ CHÍ MINH, THÁNG 4 NĂM 2024

PHÂN CÔNG NHIỆM VỤ

STT	Họ và tên	MSSV	Phân công nhiệm vụ	Mức độ hoàn thành
1	Nguyễn Trí Dũng	1910977	Tìm hiểu lý thuyết, tìm source tham khảo ,viết code.	100%
2	Nguyễn Huỳnh Nhật Duy	1510471	Tìm hiểu lý thuyết,làm báo cáo , thuyết trình, làm powerpoint	100%
3	Lê Phan Gia Nghiêm	1710200	Tìm hiểu lý thuyết, tìm dataset, làm báo cáo, làm powerpoint	100%

Mục lục

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG BIỂU

1 CHƯƠNG 1. CƠ SỞ LÝ THUYẾT

1.1 Khái niệm

1.1.1 Thuật toán Naïve Bayes

Trong học máy, phân loại Naïve Bayes là một thành viên trong nhóm các phân loại có xác suất dựa trên việc áp dụng định lý Bayes khai thác mạnh giả định độc lập giữa các hàm, hay đặc trưng.

Mô hình Naïve Bayes cũng được biết đến với nhiều tên khác nhau, ví dụ: Simple Bayes hay independence Bayes hay phân loại Bayes.

Phân loại Bayes được đánh giá cao khả năng mở rộng, đòi hỏi một số thông số tuyến tính trong số lượng các biến (các tính năng, yếu tố dự báo) trong nhiều lĩnh vực khác nhau.

Một phân loại Naïve Bayes dựa trên ý tưởng nó là một lớp được dự đoán bằng các giá trị của đặc trưng cho các thành viên của lớp đó. Các đối tượng là một nhóm (group) trong các lớp nếu chúng có cùng đặc trưng chung. Có thể có nhiều lớp rời rạc hoặc lớp nhị phân.

Các luật Bayes dựa trên xác suất để dự đoán chúng về các lớp có sẵn dựa trên các đặc trưng được trích rút. Trong phân loại Bayes, việc học được coi như xây dựng một mô hình xác suất của các đặc trưng và sử dụng mô hình này để dự đoán phân loại cho một ví dụ mới.

1.1.2 Định lý Bayes

Giả sử A và B là hai sự kiện đã xảy ra. Xác suất có điều kiện A khi biết trước điều kiện B được cho bởi:

$$P(A|B) = \frac{P(B|A).P(A)}{P(B)}$$

Trong đó: $P(A)$ - xác suất của sự kiện A xảy ra.

$P(B)$ - xác suất của sự kiện B xảy ra.

$P(B|A)$ - xác suất (có điều kiện) của sự kiện B xảy ra, nếu biết rằng sự kiện A đã xảy ra.

$P(A|B)$ - xác suất (có điều kiện) của sự kiện A xảy ra, nếu biết rằng sự kiện B đã xảy ra.

1.2 Bộ phân loại Naïve Bayes

Xét các bài toán phân loại với C nhãn khác nhau. Thay vì tìm ra chính xác nhãn của mỗi điểm dữ liệu $x \in \mathbb{R}^d$, ta có thể đi tìm xác suất để kết quả rơi vào mỗi nhãn: $p(y = c|x)$ hoặc viết gọn thành $p(c|x)$. Biểu thức này được hiểu là xác suất để đầu ra là nhãn c biết rằng đầu vào là vector x. Nếu tính được biểu thức này, ta có thể giúp xác định nhãn của mỗi điểm dữ liệu bằng các chọn ra nhãn có xác suất rơi vào cao nhất:

$$c = \arg\max_{c \in (1,....,C)} p(c|x)$$

Tuy nhiên, việc tính toán trực tiếp $p(c|x)$. Thay vào đó, sử dụng quy tắc Bayes, ta được:

$$c = \arg\max_{c \in (1,....,C)} p(c|x) = \arg\max_{c \in (1,....,C)} \frac{p(x|c)p(c)}{p(x)} = \arg\max_{c \in (1,....,C)} p(x|c)p(c)$$

Trong đó: *p(x)* không phụ thuộc vào c

p(c) được hiểu là xác suất để một điểm bất kỳ rơi vào nhãn c. Nếu tập huấn luyện lớn, *p(c)* được xác định bằng phương pháp ước lượng hợp lý dực đại (MLE) – là tỷ lệ giữa số điểm thuộc nhãn c và số điểm trong tập huấn luyện. Nếu tập huấn luyện nhỏ, giá trị này được xác định bằng phương pháp ước lượng hậu nghiệm cực đại (MAP).

$p(x|c)$ là phân phối của các điểm dữ liệu trong nhãn c. Thành phần này thường rất khó tính toán vì x là một biến ngẫu nhiên nhiều chiều. Để có thể ước lượng được phân phối đó, tập huấn luyện phải rất lớn. Nhằm đơn giản việc

tính toán, người ta thường giả sử rằng các thành phần của biến ngẫu nhiên x độc lập với nhau khi đã biết c:

$$p(x|c) = p(x_1, x_2, ..., x_d|c) = \prod_{i=1}^{d} p(x_i|c)$$

Giả thiết các chiều của dữ liệu độc lập với nhau là quá chặt, và trên thực tế, ít khi tìm được dữ liệu mà các thành phần hoàn toàn độc lập với nhau. Tuy nhiên, giả thiết ngây thơ (naïve) này đôi khi mang lại những kết quả tốt bất ngờ. Giả thiết về sự độc lập của các chiều dữ liệu này được gọi là naïve Bayes. Một phương pháp xác định nhãn của dữ liệu trên giả thiết này có tên là phân loại Naïve Bayes (NBC).

Nhờ giả thiết độc lập, NBC có tốc độ kiểm tra và huấn luyện rất nhanh. Việc này rất quan trọng trong các bài toán có dữ liệu lớn.

Ở bước huấn luyện, các phân phối p(c) và $p(x_i|c), i = 1, ..., d$ được xác định dựa vào dữ liệu huấn luyện. Việc xác định giá trị này có thể được thực hiện bằng MLE hoặc MAP.

Ở bước kiểm tra, nhãn của một điểm dữ liệu mới x được các định bởi:

$$c = \arg\max_{c \in (1,...,C)} p(c) \prod_{i=1}^{d} p(x_i|c) \quad (3.1)$$

Khi d lớn và các xác suất nhỏ, biểu thức trên cho kết quả là một số rất nhỏ, khi tính toán có thể gặp sai số. Để giải quyết việc này, ta có thể chuyển về dạng tương đương bằng cách lấy log vế phải:

$$c = \arg\max_{c \in (1,...,C)} \left(\log(p(c)) + \prod_{i=1}^{d} \log(p(x_i|c)) \right) \quad (3.2)$$

Việc này không ảnh hưởng tới kết quả vì log là hàm đồng biến trên tập các số dương.

1.3 Các phân phối thường dùng trong Naïve Bayes Classification

Việc tính toán $p(x_i|c)$ phụ thuộc vào loại dữ liệu. Có năm loại phân bố xác suất phổ biến là:

1.3.1 Gaussian Naïve Bayes

Mô hình này được sử dụng chủ yếu trong loại dữ liệu mà các thành phần là ***các biến liên tục***. Với mỗi chiều dữ liệu *i* và một nhãn *c*, x_i tuân theo phân phối chuẩn có kỳ vọng μ_{ci} và phương sai σ_{ci}^2:

$$p(x_i|c) = p(x_i|\mu_{ci}, \sigma_{ci}^2) = \frac{1}{\sqrt{2\pi\sigma_{ci}^2}} e^{\left(-\frac{(x_i-\mu_{ci})^2}{2\sigma_{ci}^2}\right)}$$

Trong đó, bộ tham số $\theta = \{\mu_{ci}, \sigma_{ci}^2\}$ được xác định bằng MLE dựa trên các điểm trong tập huấn luyện thuộc nhãn *c*.

1.3.2 Multinomial naïve Bayes

Mô hình này chủ yếu được sử dụng trong bài toán phân loại văn bản mà vector đặc trưng được xây dựng trên ý tưởng **bag of words (BoW)**. Lúc này, mỗi văn bản được biểu diễn bởi một vector có độ dài d là số từ trong từ điển. Giá trị của thành phần thứ i trong mỗi vector là số lần từ thứ i xuất hiện trong văn bản đó. Khi đó, $p(x_i|c)$ tỷ lệ với tần suất từ thứ i (hay đặc trưng thứ i trong trường hợp tổng quát) xuất hiện trong các văn bản có nhãn c. Giá trị này có thể được tính bởi:

$$\lambda_{ci} = p(x_i|c) = \frac{N_{ci}}{N_c} \qquad (3.3)$$

Trong đó, N_{ci} là tổng số lần từ thứ i xuất hiện trong các văn bản của nhãn c. Nó chính là tổng hợp tất cả thành phần thứ I của các vector đặc trưng ứng với nhãn N_c là tổng số từ, kể cả lặp, xuất hiện trong nhãn c. Nói cách khác, N_c là tổng độ dài của tất cả các văn bản thuộc nhãn c. Có thể suy ra rằng, $N_c = \sum_{i=1}^{d} N_{ci}$, từ đó $\sum_{i=1}^{d} \lambda_{ci} = 1$.

Cách tính này có hạn chế là nếu có một từ mới (trường hợp mới) chưa bao giờ xuất hiện trong nhãn c thì biểu thức (3.2) sẽ bằng không, dẫn đến vế phải của (3.1)

bằng 0 bất kể giá trị còn lại lớn như thế nào. Để giải quyết việc này, ta có thể sử dụng Laplace Smoothing:

$$\hat{\lambda}_{ci} = \frac{N_{ci} + \alpha}{N_c + d\alpha}$$

Với α là một số dương, thường bằng 1, để tránh trường hợp tỷ số bằng 0. Mẫu được cộng với $d\alpha$ để đảm bảo tổng xác suất $\sum_{i=1}^{d} \hat{\lambda}_{ci} = 1$. Như vậy, mỗi nhãn c được mô tả bởi một bộ các số dương có tổng bằng: $\hat{\lambda}_{ci} = \left\{ \hat{\lambda}_{c1}, ..., \hat{\lambda}_{c2} \right\}$.

1.3.3 Bernoulli Naïve Bayes

Mô hình này được áp dụng cho các loại dữ liệu mà mỗi thành phần là ***một giá trị nhị phân – bằng 0 hoặc 1***. Ví dụ, cũng với loại văn bản nhưng thay vì đếm tổng số lần xuất hiện một từ trong văn bản, ta chỉ cần quan tâm từ đó có xuất hiện hay không.

Khi đó, $p\left(x_i \middle| c\right)$ được tính bởi

$$p\left(x_i \middle| c\right) = p\left(i \middle| c\right)^{x_i} \left(1 - p\left(i \middle| c\right)\right)^{1-x_i}$$

Trong đó, $p\left(i \middle| c\right)$ được hiểu là xác suất từ thứ i xuất hiện trong các văn bản của class c.

x_i bằng 1 hoặc 0 tùy vào việc từ thứ i có xuất hiện hay không.

1.4 Ưu và nhược điểm

1.4.1 Ưu điểm:

Dự đoán nhanh chóng và dễ dàng. Nó cũng hoạt động tốt trong dự đoán đa lớp

Khi giả định về tính độc lập được giữ nguyên, bộ phân loại hoạt động tốt hơn so với các mô hình machine learninng khác như hồi quy logistic hoặc decision tree , nó cũng yêu cầu ít dữ liệu để train hơn.

1.4.2 Nhược điểm:

Nếu biến phân loại có một danh mục (trong dataset) không được tính toán trong tập dữ liệu training thì mô hình sẽ gán xác suất 0 (không) và sẽ không thể đưa ra dự đoán. Điều này được gọi là "Zero Frequency".

Độ phức tạp của dataset ảnh hưởng nghiêm trọng đến mô hình này.

Mặt khác, Naive Bayes còn được biết đến với độ tin cậy ko quá cao, vì vậy không nên quá coi trọng kết quả đầu ra xác suất từ nó,

Một hạn chế khác của thuật toán này là giả định về các yếu tố dự báo độc lập. Trong cuộc sống thực, gần như không thể có được một bộ dự đoán hoàn toàn độc lập.

1.5 Ứng dụng bộ phân lớp Naïve Bayes

1.5.1 Bộ Lọc TEXT:

Trình phân loại Naive Bayes thường được sử dụng trong các hệ thống lọc email để phân loại email rác. Bằng cách phân tích một số từ hoặc tính năng nhất định trong email, bộ phân loại sẽ tính toán xác suất email đó thuộc từng danh mục (thư rác hoặc không phải thư rác) dựa trên định lý Bayes. Sau đó, nó sẽ gán email vào danh mục có xác suất cao nhất.

1.5.2 Phân loại tài liệu:

Trình phân loại Naive Bayes có thể được sử dụng trong các tác vụ phân loại tài liệu, chẳng hạn như phân loại các bài báo thành các chủ đề khác nhau (ví dụ: chính trị, thể thao, giải trí). Trình phân loại kiểm tra sự xuất hiện của các từ hoặc đặc điểm trong tài liệu và tính toán khả năng xảy ra của từng danh mục dựa trên các đặc điểm của tài liệu. Cách tiếp cận này được sử dụng rộng rãi trong các hệ thống truy xuất thông tin và công cụ đề xuất nội dung.

1.5.3 Chẩn đoán y tế:

Bộ phân loại Naive Bayes có thể hỗ trợ chẩn đoán y tế bằng cách phân tích các triệu chứng của bệnh nhân và dự đoán khả năng mắc các bệnh hoặc tình trạng khác nhau. Ví dụ: khi chẩn đoán một bệnh cụ thể dựa trên các triệu chứng do bệnh nhân báo cáo, bộ phân loại sẽ tính toán xác suất của từng bệnh dựa trên các triệu chứng quan sát được và đề xuất chẩn đoán có khả năng xảy ra nhất.

1.5.4 Chấm điểm tín dụng:

Trong ngành tài chính, bộ phân loại Naive Bayes có thể được sử dụng trong các mô hình chấm điểm tín dụng để đánh giá mức độ tin cậy của người xin vay. Bằng cách phân tích các yếu tố khác nhau như thu nhập, lịch sử tín dụng, tình trạng việc làm và thông tin người phụ thuộc, bộ phân loại dự đoán xác suất người nộp đơn sẽ vỡ nợ, giúp người cho vay đưa ra quyết định sáng suốt về việc phê duyệt khoản vay.

2 CHƯƠNG 2. ỨNG DỤNG PHÂN LOẠI

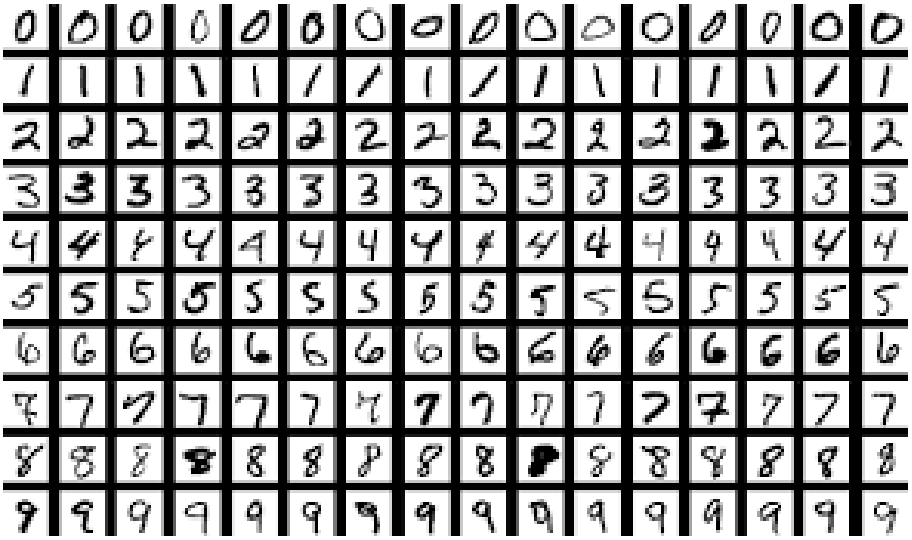
Để thuận tiện cho việc tìm hiểu mô hình thuật toán Naïve Bayes Classification và trên tinh thần của xử lý ảnh, ta sẽ thực hiện phân loại trên 2 tập dữ liệu ảnh thuộc bộ dữ liệu MNIST. Mô hình được xây dựng dựa trên kiến thức thu thập từ [1], [2], [3], [4], [5].

2.1 Dataset:

Tập dữ liệu đầu tiên là tập handwriting của MNIST (Modified National Institute of Standards and Technology) là một trong những tập dữ liệu phổ biến nhất trong lĩnh vực học máy và thị giác máy tính. Tập dữ liệu này chứa ảnh số viết tay từ 0 đến 9, mỗi ảnh có kích thước 28x28 pixel, tức là mỗi ảnh có tổng cộng 784 pixel.

Dưới đây là một số thông số cơ bản của tập dữ liệu MNIST:

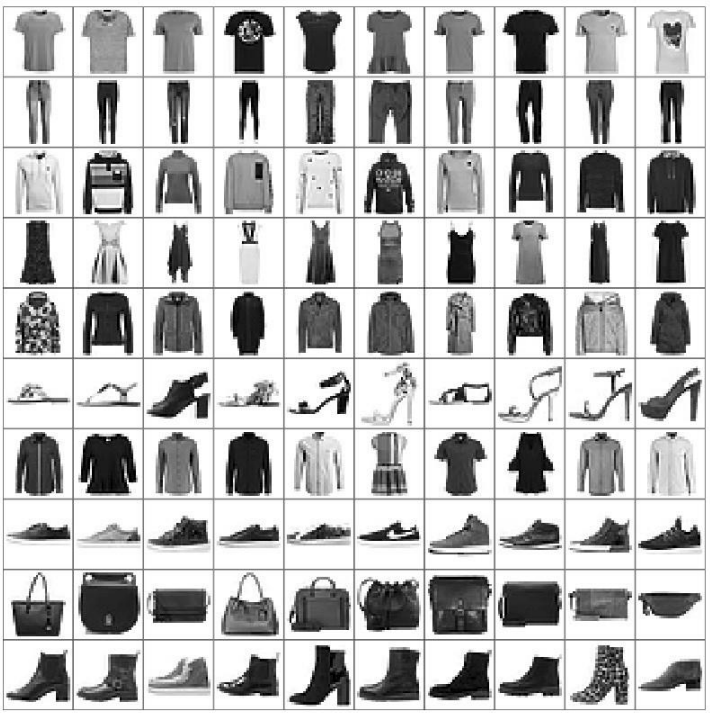
- Số lượng mẫu: Tập dữ liệu này bao gồm 60.000 ảnh cho việc huấn luyện và 10.000 ảnh cho việc kiểm tra.
- Định dạng ảnh: Mỗi ảnh được biểu diễn dưới dạng grayscale, nghĩa là mỗi pixel chỉ có giá trị từ 0 đến 255, thể hiện mức độ sáng tối.
- Số lượng lớp: Có tổng cộng 10 lớp trong tập dữ liệu này, mỗi lớp đại diện cho một số từ 0 đến 9.
- Phân phối: Số lượng mẫu cho mỗi lớp là cân bằng, có nghĩa là mỗi số có số lượng ảnh gần như bằng nhau trong tập dữ liệu.



Hình 2.1: Hình ảnh tập MNIST

Tập dữ liệu thứ hai là tập fashion MNIST, là một trong những tập dữ liệu phổ biến nhất trong lĩnh vực học máy và thị giác máy tính. Tập dữ liệu này chứa ảnh số viết tay từ 0 đến 9, mỗi ảnh có kích thước 28x28 pixel, tức là mỗi ảnh có tổng cộng 784 pixel. Tập dữ liệu nào bao gồm:

- Số lượng mẫu: Tập dữ liệu này bao gồm 60.000 ảnh cho việc huấn luyện và 10.000 ảnh cho việc kiểm tra.
- Định dạng ảnh: Mỗi ảnh được biểu diễn dưới dạng grayscale, nghĩa là mỗi pixel chỉ có giá trị từ 0 đến 255, thể hiện mức độ sáng tối.
- Số lượng lớp: Có tổng cộng 10 lớp trong tập dữ liệu này, mỗi lớp đại diện cho một số từ 0 đến 9.
- Phân phối: Số lượng mẫu cho mỗi lớp là cân bằng, có nghĩa là mỗi số có số lượng ảnh gần như bằng nhau trong tập dữ liệu.



Hình 2.2: Minh hoạ mô hình fashion MNIST

2.2 Mô tả chung:

Cấu trúc của mô hình được sử dụng để phân loại chỉ đơn giản lợi dụng khả năng trích xuất đặc trưng từ dữ liệu ảnh của bộ Autoencoder rồi đưa những đặc trưng này qua hai bộ phân loại nhằm đánh giá kết quả và so sánh chúng. Hai bộ phân loại này bao gồm Guassian Bayes classification và Logistic Classification.

Tập dữ liệu sẽ được đánh giá dựa trên khả năng vốn có của mô hình từ các tập code có sẵn đồng thời cũng được đưa qua bộ Encoder để đánh giá. Vì các file code là gần như nhau, báo cáo chỉ đề cập tới mô hình đánh giá bộ fashion MNIST.

Các phân fuction dùng để nhập dataset cũng như normalize data trên từng pixel đã được xây dựng sẵn từ trước, ta chỉ cần import file code và nhập hàm. Tập dữ liệu đã được tách sẵn từ tensor 3D (các ma trận pixel trong miền màu RBG),làm phẳng chúng thành tensor 2D thành các tập validation , train và test. Quá trình normalize từng pixel cho phép các giá trị pixel nằm trong khoảng 0 đến 1, tạo điều kiện thuận lợi hơn cho quá trình học của máy.

```
X_train, y_train, x_val, y_val, X_test, y_test = get_mnist_data()
```
✓ 0.2s

```
Reading fashion MNIST data...
<class 'numpy.ndarray'>
Done reading
train_x shape: (2500, 784)
train_y shape: (2500,)
val_x shape: (500, 784)
val_y shape: (500,)
test_x shape: (500, 784)
test_y shape: (500,)
```

```
# Preprocess data
X_train, X_test = normalize_all_pixel(X_train,X_test)
encoder_dim = X_train.shape[1]//2
print(X_train.shape)
```
✓ 0.0s

Hình 2.3: Nhập dataset và normalize dataset

Mô hình của bộ encoder bao gồm:

```
Encoder = Sequential([
    InputLayer(input_shape=X_train.shape[1:]),
    Dense(encoder_dim),
    BatchNormalization(),
    LeakyReLU(),
    Dense(encoder_dim),
    Dense(encoder_dim // 2)
], name="Encoder")
```
0.0s

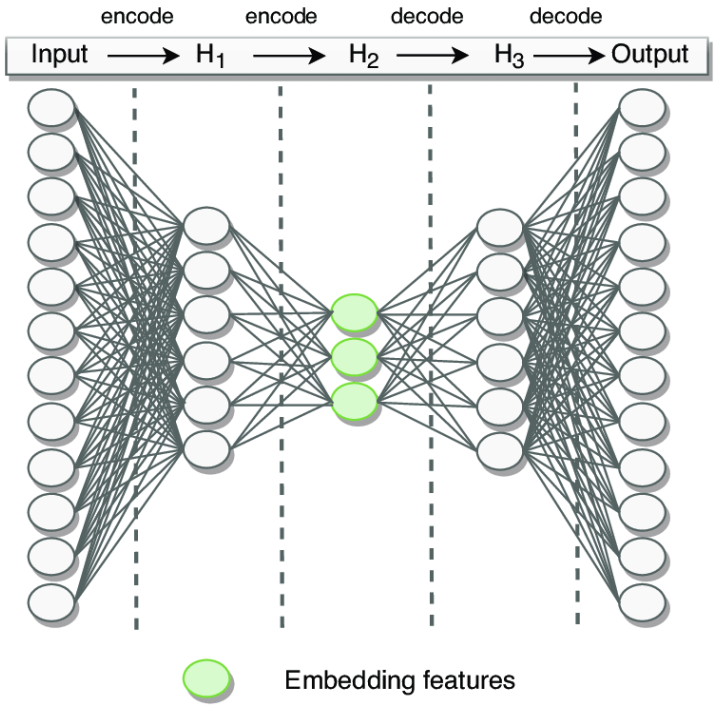
Hình 2.4: bộ Encoder

Cấu trúc của bộ Encoder bao gồm 2 lớp neuron có số node bằng một nửa size ảnh đầu vào và một lớp neuron, LeakyRelu. Bộ Decoder có cấu trúc tương tự nhưng ngược lại, điều này tạo ra hiện tượng bottleneck ở điểm bộ encoder và decoder giao nhau khi chúng được gom lại để tạo thành bộ encoder, khiến các đặc trưng được trích xuất ra có độ lớn nhỏ hơn so với data gốc. Nhiệm vụ của bộ encoder này là tạo ảnh giả từ ảnh thật trong dataset nhằm đánh giá cho thật tốt khả năng tạo ảnh giả này để có thể nắm được các đặc trưng của từng tập ảnh.

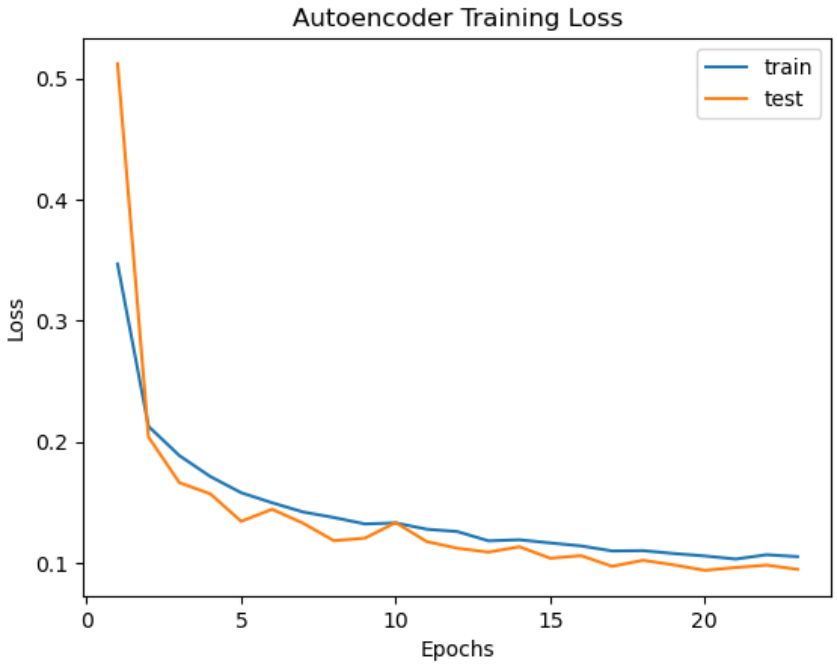
```
# Define Decoder model
Decoder = Sequential([
    InputLayer(input_shape=(encoder_dim // 2,)),
    Dense(encoder_dim),
    BatchNormalization(),
    LeakyReLU(),
    Dense(encoder_dim),
    Dense(X_train.shape[1])
], name="Decoder")
```

Hình 2.5: bộ Decoder

Sau khi đã hoàn thành model, việc train của model được kiểm soát bằng early stopping trong trường hợp hàm loss của tập validation không tiếp tục giảm quá 3 epoch.



Hình 2.6: Minh hoạ mô hình encoder



Hình 2.7: Đường cong của hàm loss

```
Epoch 23/100
79/79 [==============================] - 0s 5ms/step - loss: 0.1050 - val_loss: 0.0945
```

Hình 2.8: Epoch cuối cùng

Sau khi hài lòng với kết quả của bộ Encoder, ta lưu lại model, lưu ý rằng chỉ lưu lại bộ encoder nhằm sử dụng nó như bộ trích xuất đặc trưng.

```
# Save Encoder model
encoder = Model(inputs=inp, outputs=Encoderoutput)
encoder.compile()
if os.path.exists('encoder_v2.keras'):
    os.remove('encoder_v2.keras')  # Deletes the existing file
encoder.save('encoder_v2.keras')
```

Hình 2.9: Lưu bộ encoder

```
# Load Encoder model for feature extraction
feature_extract = load_model('encoder_v2.keras')
X_train_encoded = feature_extract.predict(X_train)
X_test_encoded = feature_extract.predict(X_test)
```

```
79/79 [==============================] - 0s 4ms/step
16/16 [==============================] - 0s 1ms/step
```

Hình 2.10: Tạo ma trận đặc trưng

Lần này, ta tạo ma trận đặc trưng của chính tập train của dataset (được trích xuất ở trên). Sau đó ta thực hiện phân loại với bộ phân lớp Gaussain Bayes lần lượt với tập train chưa được mã hoá và tập train đã được mã hoá. Việc tạo mô hình phân loại với Gaussian Bayes được dễ dàng thực hiện nhờ sklearn.

```
gaussian = GaussianNB()
for i in range(100):
    gaussian.fit(X_train, y_train)
    print("Epoch: ",i )

y_pred = gaussian.predict(X_test)

print("Accuracy: %.2f%%" %(100*accuracy_score(y_test, y_pred)))
```

Hình 2.11: Phân loại với tập train không qua mã hoá

Bước cuối cùng sẽ là đưa tập dữ liệu chưa được mã hoá cũng như đã được mã hoá qua một bộ classifier khác nhằm so sánh độ hiệu quả của bộ Gaussain Bayes. Bộ phân loại được chọn là Multi class Logistic Regession. May mắn là Sklearn hỗ trợ ta việc này với model Logistic Regression với option multi class được bật lên.

```
# Train Logistic Regression classifier on original data
model_lr_original = LogisticRegression(multi_class="ovr")
model_lr_original.fit(X_train, y_train)
yhat_lr_original = model_lr_original.predict(X_test)
accuracy_lr_original = accuracy_score(y_test, yhat_lr_original)
print("Logistic Regression (Original data) Accuracy:", accuracy_lr_original)
```

Hình 2.12: Mo hình Linear regression đơn giản

2.3 Kết quả và nhận xét:

Kết quả thu được từ dự đoán của mô hình bao gồm:

	Non-encoded input	Encoded input
Gaussian Naïve Bayesian	56.60%	77.35%
Logistic	79.2%	83.87%

Bảng 2.1: bảng kết quả độ chính xác của 2 mô hình

	Precision	F1 Score
Gaussian Naïve Bayesian	0.7735	0.7610
Logistic	0.8387	0.8343

Bảng 2.2: bảng só sánh độ chính xác và F1 của 2 mô hình

Dựa trên các kết quả đánh giá hiệu suất của các mô hình trên tập dữ liệu Fashion MNIST, chúng ta có thể nhận xét như sau:

2.3.1 Gaussian Naïve Bayes Classifier:

GNB Classifier có hai phiên bản: phiên bản thông thường và phiên bản đã mã hóa.

Phiên bản đã mã hóa có hiệu suất tốt hơn rất nhiều so với phiên bản thông thường, với độ chính xác tăng từ 56.6% lên đến 77%.

Tuy nhiên, cả hai phiên bản GNB Classifier đều có hiệu suất thấp hơn so với các mô hình Logistic Regression, bao gồm cả phiên bản đã mã hóa.

2.3.2 Logistic Regression:

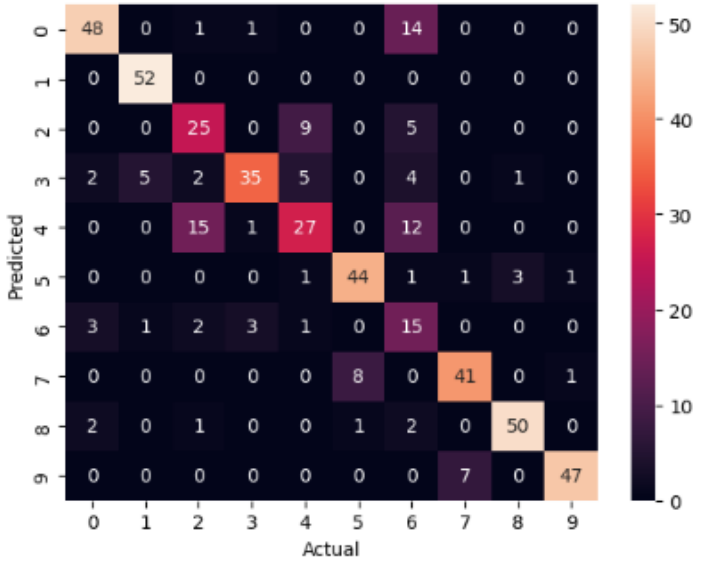
Mô hình Logistic Regression cũng có hai phiên bản: phiên bản thông thường và phiên bản đã mã hóa.

Cả hai phiên bản đều có hiệu suất cao hơn so với GB Classifier, với độ chính xác lần lượt là 79.2% và 83.8%.

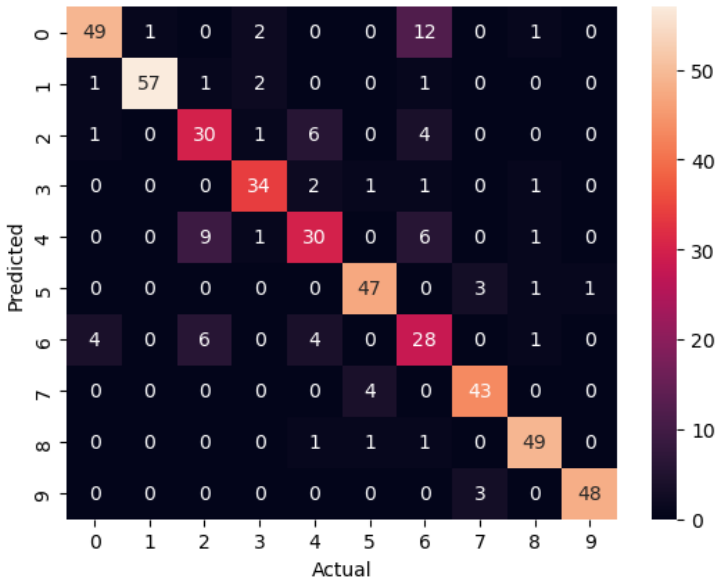
Phiên bản đã mã hóa có hiệu suất cao hơn so với phiên bản thông thường, ngụ ý rằng việc mã hóa các biến đầu vào có thể cải thiện hiệu suất của mô hình Logistic Regression.

Tuy nhiên, cần lưu ý rằng điểm mạnh của Gaussian Naive Bayes nằm ở tính đơn giản và hiệu quả tính toán, đặc biệt đối với các tập dữ liệu nhỏ hoặc đơn giản hơn.

Tổng quan, mô hình Logistic Regression (đặc biệt là phiên bản đã mã hóa) có hiệu suất tốt nhất trên tập dữ liệu đầu tiên, với độ chính xác cao nhất và các chỉ số đánh giá khác cũng cao. GNB Classifier có hiệu suất thấp hơn so với Logistic Regression, nhưng việc mã hóa các biến đầu vào có thể cải thiện hiệu suất của mô hình. Gaussian Naive Bayes cũng có hiệu suất khá tốt, nhưng thường không thể vượt qua các mô hình học máy phức tạp hơn như Logistic Regression trên các tập dữ liệu phức tạp.



Hình 2.13: confusion matrix của mô hình GNB



Hình 2.14: confusion matrix của mô hình logistic

3 THAM KHẢO

[1] J. Brownlee, “Autoencoder Feature Extraction for Classification,” MachineLearningMastery.com. Accessed: Apr. 02, 2024. [Online]. Available: https://machinelearningmastery.com/autoencoder-for-classification/
[2] H. Goonewardana, “Evaluating Multi-Class Classifiers,” Apprentice Journal. Accessed: Apr. 02, 2024. [Online]. Available: https://medium.com/apprentice-journal/evaluating-multi-class-classifiers-12b2946e755b
[3] “Autoencoders for Image Reconstruction in Python and Keras,” Stack Abuse. Accessed: Apr. 02, 2024. [Online]. Available: https://stackabuse.com/autoencoders-for-image-reconstruction-in-python-and-keras/
[4] S. Ray, “Naive Bayes Classifier Explained: Applications and Practice Problems of Naive Bayes Classifier,” Analytics Vidhya. Accessed: Apr. 02, 2024. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/
[5] “Machine Learning cơ bản.” Accessed: Apr. 02, 2024. [Online]. Available: https://machinelearningcoban.com/2017/08/08/nbc/

4 GITHUB

NgTrDunghcmut/Bayesmodelscomparision: This is an private git served the purpose of learning coding subjects. (github.com)