ĐẠI HỌC CÔNG NGHỆ THÔNG
ĐHQG TP. HỒ CHÍ MINH

NgÀNH: KHOA HỌC MÁY TÍNH

MÔN HỌC: LẬP TRÌNH PYTHON CHO MÁY HỌC

KERNEL SUPPORT VECTOR MACHINE

Sinh viên:
Phạm Trung Hiếu
Võ Khoa Nam
Trịnh Minh Hoàng

Giảng viên:
Nguyễn Vĩnh Tiệp

Contents

1 SVM là gì ?

Support vector machine (SVM) là một khái niệm trong thống kê và khoa học máy tính cho một tập hợp những phương pháp học có giám sát liên quan đến nhau để phân loại và phân tích hồi quy. SVM dạng chuẩn nhận dữ liệu vào và phân loại chúng vào hai lớp khác nhau. Do đó SVM là một thuật toán phân loại nhị phân. Với một bộ các ví dụ luyện tập thuộc hai thể loại cho trước, thuật toán luyện tập SVM xây dựng một mô hình SVM để phân loại các ví dụ khác vào hai thể loại đó. Một mô hình SVM là một cách biểu diễn các điểm trong không gian và lựa chọn ranh giới giữa hai thể loại sao cho khoảng cách từ các ví dụ luyện tập tới ranh giới là xa nhất có thể. Các ví dụ mới cũng được biểu diễn trong cùng một không gian và được thuật toán dự đoán thuộc một trong hai thể loại tùy vào ví dụ đó nằm ở phía nào của ranh giới.

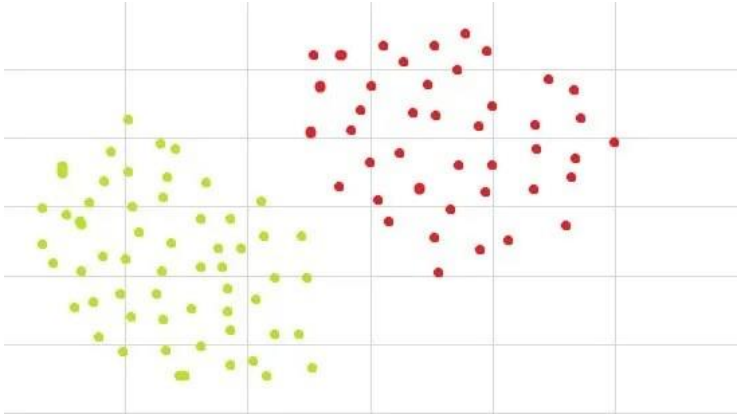

Figure 1: Thuật toán SVM

1.1 Các thành phần trong SVM

1.1.1 Support vectors

Các vectơ hỗ trợ là các điểm dữ liệu đặc biệt trong tập dữ liệu. Chúng chịu trách nhiệm về việc xây dựng siêu phẳng và là những điểm gần siêu phẳng

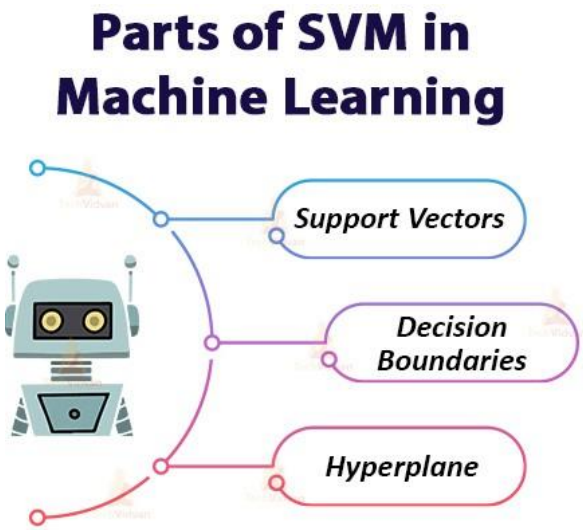

Figure 2: Các thành phần của SVM

nhất. Nếu những điểm này bị loại bỏ, vị trí của siêu phẳng sẽ bị thay đổi. Siêu phẳng có các ranh giới quyết định xung quanh nó.

Các vectơ hỗ trợ giúp giảm và tăng kích thước của các ranh giới. Chúng là những thành phần chính trong việc tạo ra một SVM.

Trong figure 2, các điểm màu vàng và màu xanh lá cây ở đây là các vector hỗ trợ. Các điểm màu xanh lam và đỏ là các lớp riêng biệt. Đường tối ở giữa là siêu phẳng trong 2-D và hai đường dọc theo siêu phẳng là ranh giới quyết định. Chúng tạo thành bề mặt quyết định chung.

1.1.2 Hyperplane

Hyperplane (siêu phẳng) trong không gian mới được định nghĩa là tập hợp các điểm có tích vô hướng với một vectơ cố định trong không gian đó là một hằng số. Vectơ xác định một siêu phẳng sử dụng trong SVM là một tổ hợp

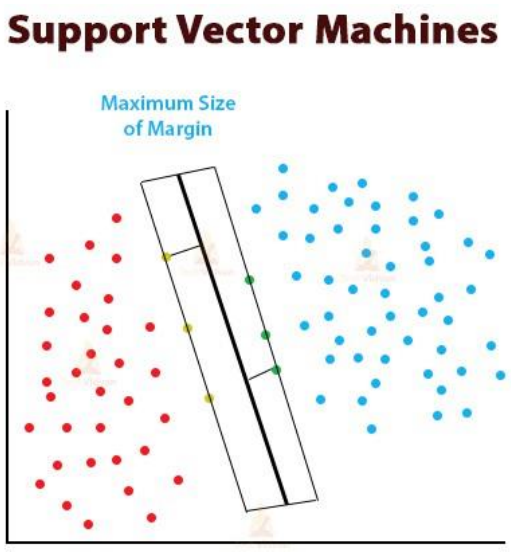

Figure 3: Vector hỗ trợ

tuyến tính của các vectơ dữ liệu luyện tập trong không gian mới. Siêu phẳng là không gian con có chiều nhỏ hơn một chiều so với không gian xung quanh nó.

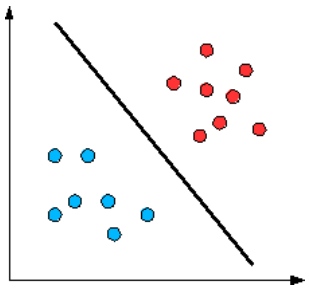

Figure 4: Siêu phẳng

Figure 3, siêu phẳng của không gian hai chiều là một đường một chiều phân chia các điểm xanh và đỏ

1.1.3 Decision boudaries

Decision boundaries (ranh giới quyết định) là hai đường thẳng được tạo ra từ các điểm vector hỗ trợ, dọc theo siêu phẳng chính là ranh giới quyết định trong SVM. Khoảng cách giữa hai vạch đậm nhạt gọi là biên độ. Một dạng siêu phẳng tối ưu hoặc tốt nhất khi kích thước ký quỹ là tối đa. Thuật toán SVM điều chỉnh siêu phẳng và lề của nó theo các vectơ hỗ trợ.

1.2 SVM hoạt động như thế nào ?

Cách hoạt động của SVM được hiểu đơn giản là. SVM là thuật toán tìm các điểm gần nhau nhất, sau đó nó sẽ vẽ một đường thẳng giữa chúng (trong không gian 2 chiều). Đường thẳng sẽ là tối ưu nế nó vuông góc với đường nối các điểm dữ liệu. Phần tốt nhất là hai lớp khác nhau được hình thành ở hai bên của đường thẳng. Bất kỳ điểm dữ liệu mới nào đi vào tập dữ liệu nó sẽ không ảnh hưởng đến siêu mặt phẳng này. Các điểm duy nhất sẽ ảnh hưởng đến siêu phẳng là các vectơ hỗ trợ. Siêu phẳng sẽ không cho phép dữ liệu từ cả hai lớp trộn lẫn trong hầu hết các trường hợp. Ngoài ra, siêu phẳng có thể tự điều chỉnh bằng cách tối đa hóa kích thước của lề của nó. Lề là không gian giữa siêu phẳng và các ranh giới quyết định.

2 Kernel là gì ?

Kernel hoặc phương thức hạt nhân (còn gọi là hàm Kernel) là tập hợp các loại thuật toán khác nhau đang được sử dụng để phân tích mẫu. Chúng được sử dụng để giải quyết một vấn đề phi tuyến tính bằng cách sử dụng một bộ phân loại tuyến tính. Phương thức Kernels được sử dụng trong SVM (Máy vectơ hỗ trợ) được sử dụng trong các bài toán phân loại và hồi quy. SVM sử dụng cái được gọi là “Kernel Trick”, nơi dữ liệu được chuyển đổi và một ranh giới tối ưu được tìm thấy cho các đầu ra có thể.

3 Cách hoạt động của kernel

Chức năng của kernel là lấy dữ liệu làm đầu vào và biến đổi nó thành dạng yêu cầu. Các thuật toán SVM khác nhau sử dụng các loại hàm nhân khác nhau. Các chức năng này có thể là nhiều loại khác nhau. Ví dụ: tuyến tính, phi tuyến, đa thức, hàm cơ sở xuyên tâm (RBF) và sigmoid.

Các hàm kernel được áp dụng trên mỗi cá thể dữ liệu để ánh xạ các dữ liệu phi tuyến tính ban đầu vào không gian có chiều cao hơn mà chúng có thể phân tách được.

Kernels cung cấp một giải pháp thay thế. Thay vì xác định một loạt các tính năng, bạn xác định một hàm nhân đơn lẻ để tính toán sự tương đồng giữa các hình ảnh. Bạn cung cấp hạt nhân này, cùng với hình ảnh và nhãn cho thuật toán học tập và đưa ra một bộ phân loại.

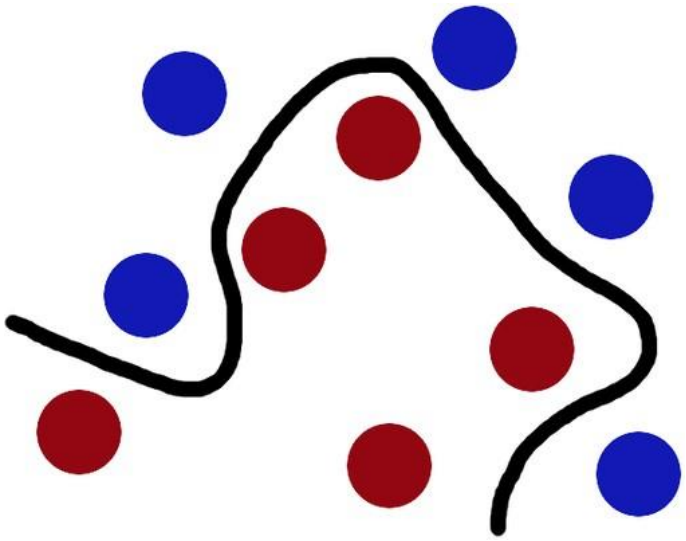

Figure 5: Dữ liệu phi tuyến tính

4 Các tham số trong SVM kernel

4.1 C

Tham số C cho biết mức độ tối ưu hoá SVM mà bạn muốn để tránh phân loại sai dữ liệu đào tạo. Đối với các C có giá trị lớn, việc tối ưu hoá chọn một siêu phẳng sẽ có lợi nhuận thấp nếu siêu phẳng đó phân loại chính xác các điểm dữ liệu. Ngược lại C có giá trị nhỏ sẽ tối ưu hoá việc tìm kiếm siêu

phẳng phân tách các điểm dữ liệu sẽ có lề lớn hơn, ngay cả khi siêu phẳng đó phân loại sai nhiều điểm hơn, ngay cả khi dữ liệu có thể phân tách tuyến tính.

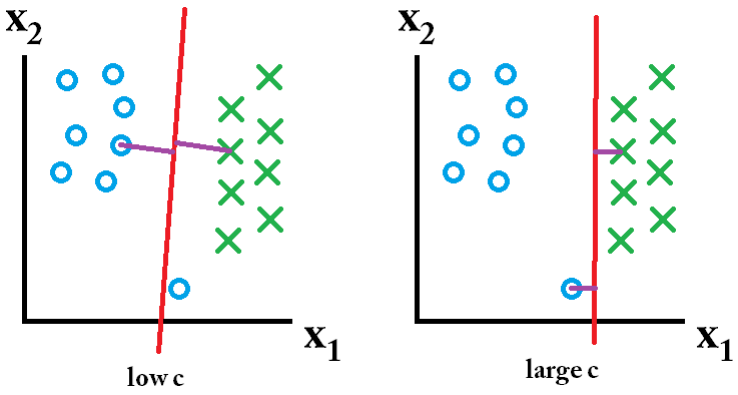

Figure 6: Cách chọn tham số C

Trong figure 6 bên trên. Ở bên trái tham số C có giá trị nhỏ hơn, thì ta có thể thấy đươc lề (đường màu tím) tạo ra giữa siêu phẳng (đường màu đỏ) sẽ có khoảng cách khá lớn và nó bỏ qua một điểm dữ liệu (điểm màu xanh lam) phân loại không chính xác. Ở bên phải tham số C có giá trị lớn hơn, bạn sẽ có được một siêu phẳng với lề nhỏ hơn và đạt độ chính xác cao hơn đối với việc phân loại.

Vậy nên chọn giá trị C thế nào cho phù hợp. Điều đó phụ thuộc vào dữ liệu mà bạn cần dự đoán nó phân bố thế nào. Trong figure 7 có thể thấy là việc chọn giá trị C thấp sẽ cho kết quả tốt hơn C có giá trị lớn.

4.2 Kernel

Chức năng chính của kernel là lấy không gian đầu vào có chiều thấp và biến nó thành không gian có chiều cao hơn. Nó chủ yếu hữu ích trong vấn đề phân tách phi tuyến tính.

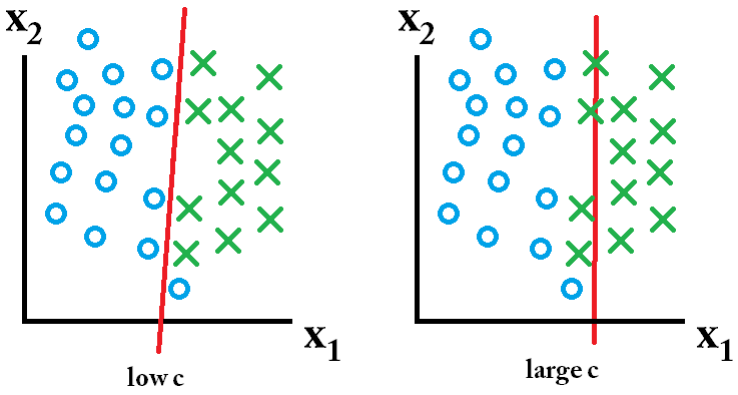

Figure 7: Cách chọn tham số C

4.2.1 Polynomial kernel

Polynomial kernel là một cách biểu diễn tổng quát hơn của linear kernel. Nó không được sử dụng nhiều bởi kém hiệu quả và chính xác hơn so với các hàm hạt nhân khác.

Công thức hàm polynomial kernel:

$$F(x, xj) = (x.xj + 1)^{d}$$

Trong công thức trên x.xj là tích vô hướng của hai tập dữ liệu đang xét. F(x,xj) đại diện cho ranh giới quyết định (decision boundaries) để phân tách các lớp đã cho.

Polynomial Kernel làm thay đổi chiều của tập dữ liệu phi tuyến tính, giúp ta có thể phân tách tuyến tính tập dữ liệu này.

Trên figure 8 ta thấy điểm màu xanh, điểm màu đỏ là hai lớp của dữ liệu và mục tiêu của ta là phân tách riêng biệt chúng ra. Ở hình bên trái ta có thể dễ dàng nhận thấy là không cách nào tách rời hai lớp. Ở hình bên phải sau khi ta biến đổi không gian chiều của tập dữ liệu, từ 2 chiều thành 3 chiều ta

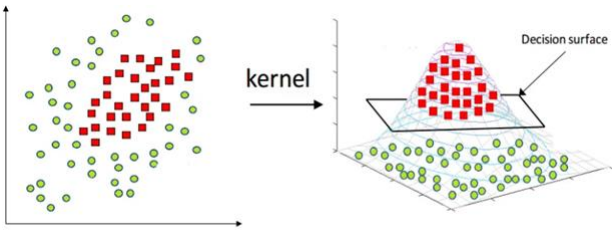

Figure 8: Polynomial kernel

có thể dễ dàng phân tách nó.

Khi sử dụng thư viện sklearn, kerrnel này được chọn bằng cách đặt kernel = 'poly'.

4.2.2 Gaussian rbf kernel

Gaussian rbf kernel là một trong các hàm hạt nhân (function kernel) được ưa thích nhất và được sử dụng nhiều nhất trong SVM. Nó thường được chọn cho dữ liệu phi tuyến tính. Nó giúp thực hiện phân tách thích hợp khi không có kiến thức trước về dữ liệu.

$F(x, xj) = \exp(-\text{gamma} * || x - xj ||^2)$

Giá trị của gamma thay đổi từ 0 đến 1 . Bạn phải cung cấp thủ công giá trị của gamma trong mã. Giá trị ưu tiên nhất cho gamma là 0,1.

4.2.3 Linear kernel

Hạt nhân tuyến tính(linear kernel) là một hàm hạt nhân (function kernel) phổ biến trong máy vector hỗ trợ. Đây là loại kernel cơ bản nhất, thường là một chiều trong tự nhiên. Nó chứng tỏ là một chức năng tốt nhất khi có rất nhiều tính năng. Hạt nhân tuyến tính chủ yếu được ưu tiên cho các bài toán phân loại văn bản vì hầu hết các loại bài toán phân loại này có thể được phân tách một cách tuyến tính.

Các hàm hạt nhân tuyến tính nhanh hơn các hàm khác. Công thức hàm hạt nhân tuyến tính:

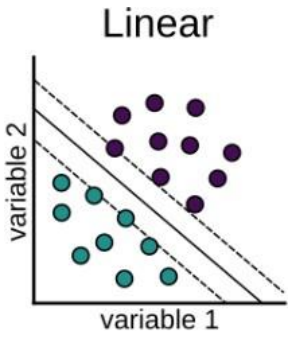

Figure 9: Linear kernel

F (x, xj) = sum (x.xj)

Trong công thức trên **x** và **xj** đại diện cho dữ liệu đang phân loại.

4.2.4 Sigmoid kernel

Sigmoid kernel là một hàm hạt nhân phổ biến trong máy vector hỗ trợ do nguồn gốc của nó bắt nguồn từ mạng nơ-ron. Được sử dụng trong các bài toán phân loại.

Nó chủ yếu được ưu tiên cho các mạng nơ-ron . Chức năng hạt nhân này tương tự như mô hình perceptron hai lớp của mạng nơ-ron, hoạt động như một chức năng kích hoạt các nơ-ron.

Hàm sigmoid kernel:

ˇF (x, xj) = tanh(xay + c)

4.3 Degree

Là cấp độ của hàm hạt nhân đa thưc ’poly’ bị bỏ qua bởi tất cả các hàm hạt nhân (kernel function) khác. Tham số degree thường có giá trị mặc định là 3.

4.4 Gamma

Gamma là một siêu tham số được sử dụng với SVM phi tuyến tính.Tham số gamma xác định mức độ ảnh hưởng của một ví dụ đào tạo. Điều này có nghĩa là gamma cao sẽ xem xét các điểm gần siêu phẳng và gamma thấp sẽ xem xét các điểm ở khoảng cách xa siêu phẳng hơn.

Trong figure 10 trên ta có thể thấy được là khi giảm tham số gamma thì việc tìm siêu phẳng sẽ xem xét các điểm dữ liệu có khoảng cách xa hơn vì vậy sẽ càng nhiều điểm dữ liệu sẽ được sử dụng

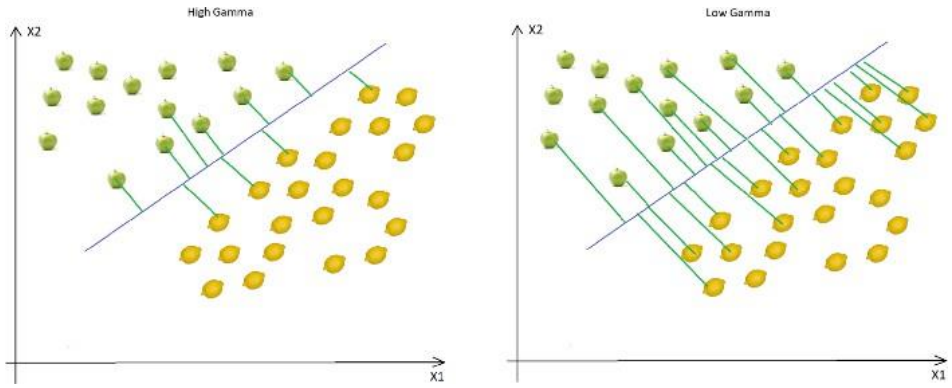

Figure 10: Tham số gamma

4.5 Coef0

Là một tham số có tính độc lập trong hàm kernel, nó chỉ có ý nghĩa trong 'poly' và 'sigmoid'.

5 Cài đặt SVM kernel và đánh giá

5.1 Mô tả bộ dữ liệu

5.1.1 Sơ lược về bộ dữ liệu

Tiểu đường là bệnh xảy ra khi mà mức độ đường trong máu trở nên cao, bệnh tiểu đường là bệnh gây ra nhiều bệnh khác như bệnh tim, bệnh thận,..

Bệnh tiểu đường gây ra bởi nhiều nguyên nhân như là thói quen tiêu thụ thực phẩm xấu, thói quen lười biếng, tuổi tác, lượng isulin,...

Trong đồ án này thì nhóm chúng em sử dụng tập dữ liệu được gọi là " Cơ sở dữ liệu về bệnh tiểu đường của người da đỏ Pima" được cung cập bởi kaggle

Mục tiêu của bộ dữ liệu này là dự đoán xem một bệnh nhân có mắc bệnh tiểu đường hay là không, dựa trên các thông số và phép đo chuẩn đoán trong tập dữ liệu. Đặc biệt là tất cả các bệnh nhân trong tập dữ liệu này đều là nữ ở độ tuổi 21 trở lên và thuộc tộc người Pima Ấn Độ.

Bộ data gồm các biến để dự đoán và biến mục tiêu. Các biến dự đoán bao gồm số lần mang thai mà bệnh nhân có, chỉ số BMI của họ, mức insulin, tuổi tác,..:

Các biến trong bộ dữ liệu

- Pregnancies: số lần mang thai.
- Glucose: nồng độ glucose trong máu trong 2 giờ thử nghiệm dung nạp glucose qua đường miệng.
- BloodPressure: huyết áp (mm Hg).
- SkinThickness: độ dày nếp gấp da cơ tam đầu (mm).
- Insulin: insulin trong máu (mu U/ml).
- BMI: body mass index (weigh in kg / (height in m)x2).
- DiabetesPedigreeFunction: di truyền bệnh tiểu đường.
- Age: tuổi (years).
- Outcome: đầu ra là biến có giá trị 0 hoặc 1 (268/768 giá trị 1, còn lại là 0).

5.1.2 Thông tin chung, một số khái niệm về các biến

Glucose Tolerance Test - Thử nghiệm dung nạp glucose Đây là một xét nghiệm máu, bao gồm việc lấy nhiều mẫu máu trong thời gian thường là 2

giờ, được sử dụng để chuẩn đoán bệnh tiểu đường. Kết quả có thể được phân loại là bình thường, suy giảm hoặc bất thường.

- Kết quả bình thường: mức đường huyết trong 2 giờ < 140 mg/dl
- Suy giảm: mức đường huyết trong 2 giờ trong khoảng 140 mg/dl đến 200 mg/dl.
- Kết quả bất thường: mức đường huyết trong 2 giờ lớn hơn 200 mg/dl

BloodPressure - huyết áp Chỉ số tâm trương, hoặc số dưới cùng, là áp suất trong động mạch khi tim nghỉ giữa các nhịp đập. Đây là thời gian tim nạp đầy máu và nhận oxy. Huyết áp tâm trương bình thường thấp hơn 80. Chỉ số 90 hoặc cao hơn có nghĩa là bạn bị cao huyết áp.

- Bình thường: Tâm thu dưới 120 và tâm trương dưới 80
- Tăng cao: Tâm thu 120–129 và tâm trương dưới 80
- Tăng huyết áp giai đoạn 1: Tâm thu 130–139 và tâm trương 80–89
- Tăng huyết áp giai đoạn 2: Tâm thu 140 trở lên và tâm trương 90 trở lên trên 180 và tâm trương trên 120.

BMI Các loại trạng thái cân nặng tiêu chuẩn liên quan đến phạm vi BMI cho người lớn

- Dưới 18,5 -> Thiếu cân
- 18,5 - 24,9 -> Cân nặng bình thường hoặc khỏe mạnh
- 25,0 - 29,9 -> Thừa cân
- 30.0 trở lên -> Béo phì

5.1.3 Khám phá dữ liệu

Có thể thấy được từ figure 12, tập dữ liệu chúng ta có 768 hàng và 9 cột. Cột 'Outcome' là cột mà chúng tôi sẽ tiến hành dự đoán, cho biết bệnh nhân có bị bênh tiểu đường hay không giá trị 1 cho biết bị bệnh tiểu đường, giá trị 0 không bị bệnh tiểu đường.

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

Figure 11: 5 dòng đầu tiên của data diabites

```
1 df.groupby('Outcome').size()
```

```
Outcome
0    500
1    268
dtype: int64
```

Figure 12: Class distribute of dataset

Trong 768 người chúng ta có thể thấy được là có 500 người được gán nhãn là 0 (không mắc bệnh tiểu đường) và 268 người được gán nhãn là 1 (mắc bệnh tiểu đường).

Bộ dữ liệu chỉ có các giá trị số nguyên(interger) và kiểu số thực(float). Không tồn tại các cột có giá trị là null hoặc mất giá trị (missing value).

5.1.4 Xử lý các dữ liệu không mong đợi

Nhìn vào data và biểu đồ phân bố data ta có thể nhận thấy rằng một vài tiêu chí có giá trị không mong muốn.

Blood Pressure (huyết áp): nhìn vào biểu đồ có thể thấy được có giá trị huyết áp là 0. Tuy nhiên huyết áp của con người thì không thể mang giá trị 0 được. Dữ liệu blood pressure (figure 13) tồn tại 35 giá trị 0 trong cột dữ liệu.

Plasma glucose level (chỉ số đường huyết): có xuất hiện giá trị đường huyết

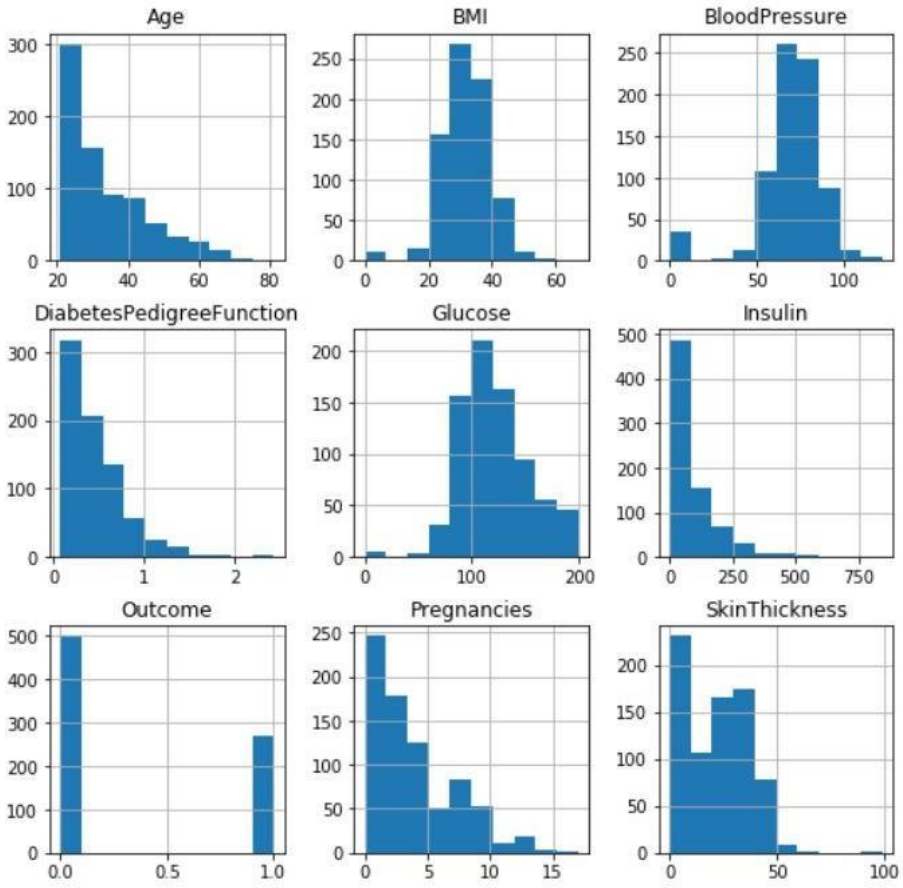

Figure 13: Info dataset

bằng 0. Chỉ số đường huyết của con người không thể bằng 0 được.

Ngoài ra các cột giá trị như là chỉ số insulin, BMI và chỉ số Skin Fold Thickness cũng tồn tại các giá trị 0 không phù hợp với thực tế.

Các đề xuất xử lý dữ liệu trên:

1. Xoá/loại bỏ các trường hợp trên: tuy nhiên việc xoá hoặc loại bỏ các trường hợp mà có quá nhiều giá trị bằng 0, sẽ không khả thi và sẽ làm mất thông tin như các giá trị của các cột (SkinThickness hay Insulin có lần lượt là 227 và 334 giá trị bằng 0). Tuy nhiên, việc xoá loại bỏ này

```
1 print("Total : ", df[df.BloodPressure == 0].shape[0])
```

```
Total :  35
```

Figure 14: Tổng số giá trị 0 trong cột bloddpressure

```
1 print("Total : ", df[df.Glucose == 0].shape[0])
```

```
Total :  5
```

Figure 15: Tổng số giá trị 0 trong cột Glucose

thích hợp với các cột dữ liệu như là BMI, glucose hay BloodPressure.

2. Đặt các giá trị trung bình: điều này sẽ có hiệu quả đối với các bộ dữ liệu khác. Tuy nhiên đối với bộ dữ liệu này việc đặt các giá trị trung bình đối với cột BloodPressure (huyết áp) sẽ không cho kết quả tốt đối với mô hình

3. Không sử dụng các cột này làm thông số đánh giá: việc này là không khả quan bởi vì các yếu tố như Insulin, glucose hay BloodPressure,... Là những yếu tố trực tiếp gây ra bệnh tiểu đường.

Trong quá trình áp dụng model SVM trên tập dữ liệu thì nhóm tiến hành phương pháp xử lý thứ nhất đối với bộ dữ liệu này. Là sẽ thực hiện thao tác xoá các hàng BloodPressure, BMI và glucose có giá trị bằng 0.

5.2 Cài đặt và tunning các tham số với bộ dữ liệu

5.2.1 Cài đặt

Nhóm sẽ tiến hành cài đặt thuật toán SVM với bộ dữ liệu về bệnh tiểu đường bằng python với công cụ hỗ trợ là thư viện scikit-learn và googlecolab.

Thư viện sử dụng

Nhóm sử dụng các hàm có sẵn trong thư viện sklearn, pandas như: SVC, Kfold, Gridsearch,...

```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import numpy as np
from sklearn.metrics import accuracy_score, roc_auc_score, roc_curve, \
    classification_report
from sklearn.model_selection import KFold
from sklearn.model_selection import train_test_split, GridSearchCV, cross_val_score
from sklearn.svm import SVC
from sklearn import svm
from sklearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV
```

Figure 16: Một số thư viện được sử dụng

Dữ liệu

Bộ dữ liệu sử dụng là bộ dữ liệu về bệnh tiểu đường. Tập dữ liệu thu thập thông tin của 768 người. Từ 21 tuổi trở lên với các thông số như là tuổi, huyết áp, chỉ số BMI, lượng đường,...

Đây là một tập dữ liệu không cân bằng. Số bệnh nhân không mắc bệnh tiểu đường (0) thì gấp đôi số lượng bệnh nhân mắc bệnh tiểu đường (1) (figure 19).

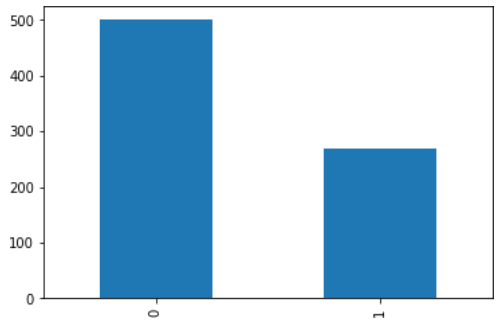

Figure 17: Biểu đồ số lượng bệnh nhân mắc và không mắc bệnh tiểu đường

Nhóm thực hiện thống kê các hàng, cột dữ liệu có giá trị là 0 không phù hợp với thực tế.

Nhóm tiến hành loại bỏ các hàng mà số lượng các giá trị bằng 0 ít như là

```
print("Total BloodPtrssure : ", df[df.BloodPressure == 0].shape[0])
print("Total Glucose : ", df[df.Glucose == 0].shape[0])
print("Total SkinThickness : ", df[df.SkinThickness == 0].shape[0])
print("Total BMI : ", df[df.BMI == 0].shape[0])
print("Total Insulin : ", df[df.Insulin == 0].shape[0])
```

```
Total BloodPtrssure :  35
Total Glucose :  5
Total SkinThickness :  227
Total BMI :  11
Total Insulin :  374
```

Figure 18: Số lượng giá trị bằng 0 trong các cột dữ liệu

BMI, glucose hay bloodpressure. Tiến hành chia tập dữ liệu thành các tập train và test với kích thước là 8-2.

```
df_mod = df[(df.BloodPressure != 0) & (df.BMI != 0) & (df.Glucose != 0)]
print(df_mod.shape)
```

```
(724, 9)
```

```
X = df.iloc[:, :-2]
y = df.iloc[:, -1]
```

Figure 19: Xoá và chia tập dữ liệu

Áp dụng model SVM đối với tập dữ liệu và đánh giá độ chính xác của model bằng cách sử dụng thư viện sklearn.

Kết quả của mô hình SVM sử dụng trên tập dữ liệu bệnh tiểu đường cho kết quả đạt 0.8 (80 phần trăm).

Kfold

Nhóm áp dụng Kfold với k = 5 để đánh giá mô hình.

```python
x_train, x_test, y_train, y_test = train_test_split(X, y, random_state = 0, test_size = 0.2)

# Creating the SVM model.
clf = svm.SVC()
clf.fit(x_train,y_train)
y_pred = clf.predict(x_test)

print(classification_report(y_pred, y_test))
```

```
              precision    recall  f1-score   support

           0       0.92      0.82      0.86       120
           1       0.53      0.74      0.62        34

    accuracy                           0.80       154
   macro avg       0.72      0.78      0.74       154
weighted avg       0.83      0.80      0.81       154
```

Figure 20: Áp dụng SVM và kết quả mô hình

```python
kfold = KFold(n_splits=5, shuffle=True, random_state=1)
accuracy_list = []
X_data = X
y_data = y
for train, test in kfold.split(X_data, y_data):
    clf = SVC()
    X_train, X_test = X_data.iloc[train], X_data.iloc[test];
    y_train, y_test = y_data.iloc[train], y_data.iloc[test]

    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    accuracy_list.append(accuracy_score(y_test, y_pred))

print("Accuracy: ", np.mean(accuracy_list) * 100)
```

```
Accuracy:  75.78134284016637
```

Figure 21: Áp dụng Kfold

5.2.2 Tunning tham số

Áp dụng gridsearch để tunning các siêu tham số:

- C (0.1 , 1, 100, 1000)
- Kernel (linear, rbf, sigmoid, poly)
- gamma (1, 0.1, 0.01, 0.001, 0.0001)

Kết quả thực nghiệm:

- Không sử dụng gridsearch:

- Accuracy với các tham số mặc định: 0.8
- Accuracy với kfold (k=5): 0.757813

- Sử dụng gridsearch:
  - Accuracy: 0.805194
  - Best paramater: { ‘C’ : 1, ‘kernel’: linear, ‘gamma’: 1}

Kết quả accuracy khi tunning tham số trong SVM của tập dữ liệu này thay đổi không quá nhiều đối với việc sử dụng mô hình SVM bình thường. Nhưng việc tunning các tham mất rất nhiều thời gian.

6 Ưu nhược điểm và fíng dụng của SVM kernel

6.1 Ưu điểm

SVM xử lý hiệu quả các bộ dữ liệu nhiều chiều và không phân biệt tuyến tính.

SVM hoạt động tốt với cả ba loại dữ liệu là (có cấu trúc, bán cấu trúc và phi cấu trúc).

Có nhiều hàm hạt nhân (function kernel) cho nhiều quyết định khác nhau.

6.2 Nhược điểm

Với nhiều hàm hạt nhân việc chọn hàm hạt nhân phù hợp với từng loại dữ liệu là điều không dễ dàng.

SVM mất nhiều thời gian đào tạo hơn đối với các tập dữ liệu lớn.

Bộ dữ liệu có quá nhiều tính năng. SVM sẽ cho kết quả kém hơn bình thường.

6.3 Bảng so sánh

Bảng so sánh kết quả các mô hình. Sử dụng Kfold với k = 5. Kết quả là accuracy trung bình của 5 fold trên tập test của bộ dữ liệu. Kết quả của mô

hình với các siêu tham số là mặc định ngoại trừ SVM tunning.

Từ bảng kết quả có thể thấy được mô hình SVM cho kết quả cao nhất trong các mô hình, nó hoạt động tốt trên bộ dữ liệu về bệnh tiểu đường.

Model	Accuracy
SVM	0.757813
RF	0.750055
KNN	0.700569
LR	0.766989
XGBM	0.752653

Table 1: Bảng so sánh kết quả các mô hình

7 Tài liệu tham khảo

1. Machine Learning Workflow on Diabetes Data, 12/12/2021, từ https://towardsdatascience.com/machine-learning-workflow-on-diabetes-data-part-01-573864fcc6b8
2. SVM in Machine learning, 12/12/2021, từ https://techvidvan.com/tutorials/svm-in-machine-learning/?fbclid=IwAR3X-cni4xLHUwrQZVxYg4Buwszd1-pksGemx6_VaPJwQzqJy12FfdzOdZ0
3. Support Vector Machine, 12/12/2021, từ https://machinelearningcoban.com/2017/04/09/smv/
4. Diabetes Prediction Using Classification Models, 12/12/2021, từ https://www.kaggle.com/simgeerek/diabetes-prediction-using-classification-models/data?fbclid=IwAR17dRosVAcAlgPD-VB1x_JDS0NggRMn2rH4frepudwcB7GmBEwFUjV0tQg