# GIẢI THÍCH NHẬP MÔN HỌC MÁY

**1. Đầu tiên thì mình sẽ khai báo các thư viện cần thiết cho bài toán của mình**

**2. Tạo ra 1 hàm để chạy cùng một lúc 2 dữ liệu**

- với tham số thứ nhất là file dữ liệu

- tham số thứ 2 là cột nhãn y

- tham số thứ 3 là 1 thuộc tính trong X

**3. Tiền xử lý:**

- sử dụng thư viện pandas sử dụng hàm read_csv để đọc file .csv

- sau đó mình hiển thị một chút dữ liệu cụ thể ở đây là hiển thị 5 mẫu dữ liệu ( 5 dòng dữ liệu )

- xác định X, X là dữ liệu với axis = 1 là xoá cột nhãn y ra khỏi dữ liệu

- xác định cột nhãn y

**4. chuẩn hoá dữ liệu sử dụng Standardisation**

- **C**huẩn hoá dữ liệu là việc giúp cho các giá trị đặc trưng trung bình = 0 và phương sai bằng 1.

- Để chuẩn hoá được dữ liệu thì phải xác định dược trung binh và độ lệch chuẩn cho phân phối của mỗi đặc trưng

- Công thức tính:

  x' = (x - average(x))/std(x)

  average(x)  = sum(x) / count(x)

  std(x) = sqrt(sum((x - average(x))^2)/count(x))

  Trong đó thì x là vecto đặc trưng ban đầu

  average(x): là trung bình của vecto đặc trưng đó

  std(x): là độ lệch chuẩn

**5. biểu diễn dữ liệu trên không gian 3 chiều và biểu diễn mối quan hệ giữa các thành phần chính với nhau (trực quan hoá dữ liệu)**

- Thuật toán PCA (học không có giám sát)

- ý tưởng của thuật toán PCA: là tạo ra các đặc trưng mới đọc lập là kết hợp tuyến tính của các đặc trưng cũ

- đặc trưng mới định nghĩa là 1 hình chiếu của dữ liệu lên không gian con sao cho khoảng cách giữa hình chiếu với dữ liệu gốc là nhỏ nhất

- có tác dụng làm giảm chiều dữ liệu, phương pháp này sẽ phân tích ra thành phần chính giảm bớt việc tính toán cũng như tốc độ xử lý

- bằng cách xử dụng PCA thì có thể mất phương sai ( tức là mất thông tin)

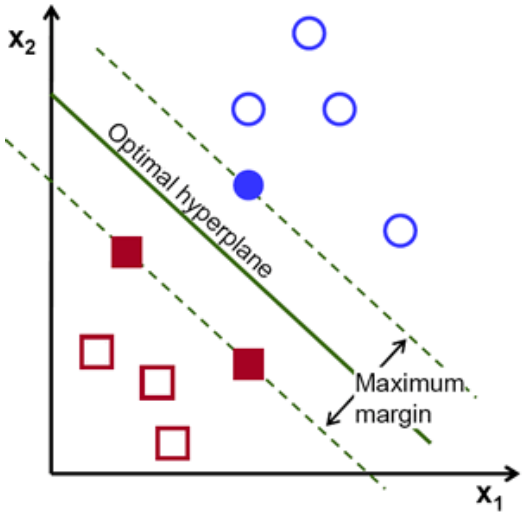
**Giải thích code:**

- Đầu tiên mình sử dụng thư viện PCA với số chiều là 3.
- Fit_transform(X): có nghĩa là chỉnh mô hình với dữ liệu X và áp dụng giảm kích thước trên X.
- Tạo hình và trục:
  - plt.figure(): tạo ra một hình với
  - fig.add_subplot(projection='3d'): tạo trục
- ax.scatter(x_pca, y_pca, z_pca, c=y, s=60): vẽ các điểm dữ liệu lên hình vẽ, với c = y là mảng danh sách màu, s = 60
- ax.legend(['Malign']): đối với bài toán khả năng sống sốt của bệnh nhân suy tim thì màu vàng là chết, còn với bài toán chuẩn đoán ung thư vú thì màu vàng là mắc bệnh).
- ax.set_xlabel: để set label cho các thành phần chính.
- sns.scatterplot(x = x_pca, y = z_pca, hue=y, palette='Set1'): các vecto vẽ trên trục x và trục y. hue = y tức là tạo ra các điểm có màu sắc khác nhau.

**6. Chia tập dữ liệu thành tập train, test: sử dụng hàm train_test_split để chia dữ liệu**

**7. Mô hình học máy, huấn luyện mô hình**

- thuật toán SVM là phương pháp học có giám sát.
- Support vector machine xây dựng một siêu phẳng hyperplane để phân lớp tập dữ liệu thành các lớp riêng biệt.
- Siêu phẳng là cái gì? Là một hàm tương tự phương trình đường thẳng y = ax + b. Trong tập dữ liệu thì 2 lớp là lớp 0 và lớp 1 thì siêu phẳng này chính là một đường thẳng.
- Ý tưởng của thuật toán: dùng thủ thuật để ánh xạ tập dữ liệu ban đầu vào một không gian nhiều chiều hơn. Ví dụ để phân lớp các quả bóng xanh và quả bóng đỏ trên bàn thì nếu các quả bóng nó k đan xen nhau thì có thể dùng 1 cây que dài để phân chia 2 loại bóng này.
- Các quả bóng sẽ đại diện cho các điểm dữ liệu, màu xanh và màu đỏ đại diện đặc trưng cho 2 lớp. cái bàn sẽ đại diện cho 1 mặt phẳng. và cây que sẽ đại diện cho 1 đường thẳng
- Đối với các trường hợp phức tạp hơn là các quả bóng này đan xen nhau thì không thể dùng đường thẳng để phân tách 2 lớp này. Khi đó ta sẽ ánh xạ tập dữ liệu ban đầu vào không gian nhiều chiều bằng cách sử dụng kernel
- Margin trong SVM là khoảng cáchc giữa siêu phẳng đến 2 điểm gần nhất tương ứng với các lớp




- **<u>Giải thích code:</u>**
  - models = SVC(kernel='linear').fit(X_train, y_train): sử dụng hàm fit() để cho máy học với kernel = ‘linear’ là tạo ra 1 siêu phẳng để phân tách dữ liệu.

**8. Dự đoán mô hình**

- **sử dụng hàm predict () để dự đoán nhãn cho tập dữ liệu test**
- models.coef_: hệ số w có nghĩa là: hệ số ước lượng cho bài toán hồi quy tuyến tính
- models.coef_.shape: mảng một chiều có độ dài là 3
- models.intercept_: hệ số bias:
- models.classes_: số lớp cần dự đoán

**9. Đánh giá mô hình học máy dựa trên kết qura dự đoán**

- accuracy_score: tính toán độ chính xác, xem với tập dữ liệu dự đoán khớp chính xác với tập dữ liệu để thử nghiệm với các nhãn tương ứng
- classification_report: tổng hợp lại các chỉ số phân loại
  - với precision: độ chính xác tương ứng với nhãn 0 và nhãn 1

```
Ma trận dự doán:
 [[37  4]
 [13  6]]
```

- Có tổng cộng 60 dùng để test: gồm 41 mẫu khả năng sống sót là chết, 19 khả năng sống sót là sống

Trong đó có:

- Thực tế có 41 mẫu bệnh nhân có khả năng sống sót là chết thì có 37 mẫu được dự đoán đúng, còn 4 bệnh nhân thực tế là chết thì lại được dự đoán là có khả năng sống sót.

- Thực tế có 19 mẫu bệnh nhân có khả năng sống sót là sống thì có 6 mẫu được dự đoán đúng là khả năng sống sót là sống, còn 13 mẫu bệnh nhân thực tế có khả năng sống sót thì lại dự đoán là chết.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.74 | 0.90 | 0.81 | 41 |
| 1 | 0.60 | 0.32 | 0.41 | 19 |
| accuracy | | | 0.72 | 60 |
| macro avg | 0.67 | 0.61 | 0.61 | 60 |
| weighted avg | 0.70 | 0.72 | 0.69 | 60 |

- Độ chính xác được tính bằng = khả năng dự đoán sống chính xác + khả năng dự đoán chết chính xác/ tổng số mẫu.
- Precision với lớp 0 được tính = khả năng chết chính xác / (tổng chết) = 37 / (37+13)
- Recall: tỉ lệ quan sát chết chính xác / tổng dữ liệu dùng để test có khả năng là chết = 37 / (37 + 4) = 0. 90
- F1- score: được tính bằng = 2 * (precision * recall) / (precision + recall)
- Support: 41 bệnh nhân có khả năng sốt sót là chết, 19 bệnh nhân có khả năng sốt sót là sống.
- Accuracy: độ chính xác trên tổng số mẫu test
- Macro avg: độ chính xác trung bình của tất cả 2 lớp lớp 0 và lớp 1 = lớp 0 + lớp 1 / 2
- Weighted avg (trung bình có trọng số) = tỉ lệ số bệnh nhân chết * số lượng người chết + tỉ lệ bệnh nhân sống * số bệnh nhân sống / tổng số bệnh nhân