

# MACHINE LEARNING: BÀI THỰC HÀNH SỐ 4 – PHẦN 1.

## PHÂN TÍCH THÀNH PHẦN CHÍNH (PRINCIPAL COMPONENTS ANALYSIS - PCA)

Trong phần này chúng ta sẽ thực hành một số nội dung liên quan đến PCA và ứng dụng trong việc giảm số chiều dữ liệu. Dữ liệu sau khi được giảm số chiều sẽ được sử dụng cho một số mô hình phân loại đã học (ví dụ Logistic Regression, SoftMax).

Do phần PCA sẽ cần khá nhiều tính toán với ma trận, nên chúng ta sẽ sử dụng thư viện Sci-Kit Learn (sklearn) hoặc ít nhất là numpy để tận dụng việc tính toán kiểu vector hóa.

**Ví dụ 1.** Trong ví dụ này, chúng ta sử dụng PCA để phân tích và giảm số chiều của bộ dữ liệu hoa IRIS (xem lại bài thực hành phần Naïve Bayes). Dữ liệu gồm có 150 mẫu hoa phân đều trong 03 loại (class), mỗi mẫu có 04 chiều. Dữ liệu có thể lấy từ tệp Iris.csv đính kèm hoặc link: [https://drive.google.com/file/d/1rb2ZQC1KCIP7Z1SZe9\\_T3XMKIUUFEB3/view](https://drive.google.com/file/d/1rb2ZQC1KCIP7Z1SZe9_T3XMKIUUFEB3/view). Thông tin thêm về dữ liệu có thể tham khảo lại trong bài hướng dẫn thực hành các phần Naïve Bayes hoặc SoftMax, hoặc trong link <https://archive.ics.uci.edu/ml/datasets/iris>. Do kích thước dữ liệu khá nhỏ nên trong ví dụ này chúng ta sẽ không sử dụng công cụ PCA của sklearn mà xây dựng từ các công cụ đại số tuyến tính của numpy. Ngoài ra chúng ta sẽ sử dụng công cụ PANDAS để đọc tệp CSV và một số công cụ vẽ để hiển thị kết quả (matplotlib.pyplot và seaborn).

### Code minh họa trong Python:

Đoạn chương trình đọc tệp dữ liệu, truy xuất một số thông tin thống kê trên dữ liệu. Khai báo thư viện và đường dẫn đến tệp dữ liệu. Cần sửa đường dẫn này đến vị trí đặt dữ liệu cụ thể.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

#Read csv data file, change to your location
df = pd.read_csv("D:\\Teach_n_Train\\Machine Learning\\code\\data\\Iris.csv")
```

Đoạn chương trình dưới đây chỉ thực hiện in thông tin hoặc hiển thị dữ liệu để chúng ta có cái nhìn tổng quan, trực giác

```
# show data information
df

df.describe()

sns.pairplot(df, hue = 'variety')
```

Chuẩn hóa và tính ma trận hiệp phương sai của dữ liệu:

```
# We're separating the species column
species = df["variety"].tolist()
X = df.drop("variety", 1)

# Standardize the data
X = (X - X.mean()) / X.std(ddof=0)

# Calculating the correlation matrix of the data
X_corr = (1 / 150) * X.T.dot(X)

# Plotting the correlation matrix
plt.figure(figsize=(10,10))
sns.heatmap(X_corr, vmax=1, square=True, annot=True)
plt.title('Correlation matrix')
```

Trong đoạn chương trình dưới đây, chúng ta tính hệ riêng của ma trận hiệp phương sai ( $X_{corr}$ ) bằng cả 2 công cụ của thư viện Numpy.Linalg là SVD (khai triển kỳ dị) và Eig (Khai triển hệ giá trị-vector riêng), sau đó in ra kết quả để so sánh.

```
# method1
u,s,v = np.linalg.svd(X_corr)
eig_values, eig_vectors = s, u
eig_values, eig_vectors

# method2
np.linalg.eig(X_corr)
```

Đoạn chương trình dưới đây hiển thị mức độ “quan trọng” của các trường dữ liệu (gồm 04 trường theo thứ tự như trong mô tả dữ liệu)

```
# plotting the variance explained by each PC
explained_variance=(eig_values / np.sum(eig_values))*100
plt.figure(figsize=(8,4))
plt.bar(range(4), explained_variance, alpha=0.6)
plt.ylabel('Percentage of explained variance')
plt.xlabel('Dimensions')
```

Đoạn chương trình chứa thủ tục chiếu dữ liệu (X) xuống không gian con 02 chiều ứng với 2 giá trị riêng lớn nhất (tức là trị riêng [0] và trị riêng [1])

```
# calculating our new axis
pc1 = X.dot(eig_vectors[:,0])
pc2 = X.dot(eig_vectors[:,1])
```

Vẽ dữ liệu trong không gian 2 chiều mới thành lập để quan sát:

```
# plotting in 2D
def plot_scatter(pc1, pc2):
    fig, ax = plt.subplots(figsize=(15, 8))

    species_unique = list(set(species))
    species_colors = ["r", "b", "g"]

    for i, spec in enumerate(species):
        plt.scatter(pc1[i], pc2[i], label = spec, s = 20,
c=species_colors[species_unique.index(spec)])
        ax.annotate(str(i+1), (pc1[i],pc2[i]))

    from collections import OrderedDict
    handles, labels = plt.gca().get_legend_handles_labels()
    by_label = OrderedDict(zip(labels, handles))
    plt.legend(by_label.values(), by_label.keys(), prop={'size': 15}, loc=4)

    ax.set_xlabel('Principal Component 1', fontsize = 15)
    ax.set_ylabel('Principal Component 2', fontsize = 15)
    ax.axhline(y=0, color="grey", linestyle="--")
    ax.axvline(x=0, color="grey", linestyle="--")

    plt.grid()
    plt.axis([-4, 4, -3, 3])
    plt.show()

plot_scatter(pc1, pc2)
```

Chúng ta có thể sử dụng thư viện sklearn để thực hiện yêu cầu trên. Toàn bộ đoạn lệnh như sau:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

#Read csv data file, change to your location
df = pd.read_csv("D:\\Teach_n_Train\\Machine Learning\\code\\data\\Iris.csv")

from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler

X = df.drop("variety", 1)
X = StandardScaler().fit_transform(X)
pca = PCA()
result = pca.fit_transform(X)
# Remember what we said about the sign of eigen vectors that might change ?
pc1 = - result[:,0]
pc2 = - result[:,1]
plot_scatter(pc1, pc2)
```

### Yêu cầu thực hành:

- 1) Sử dụng đoạn code chọn số chiều chính trong ví dụ 2, đưa tập dữ liệu đã đọc về còn 2 chiều, sau đó hiển thị lên màn hình để xem quan hệ giữa các lớp dữ liệu.
- 2) Với đoạn chương trình đọc dữ liệu đã có, hãy chạy lại ví dụ này với các thư viện của gói linear\_model, lớp LogisticRegression và so sánh kết quả, chia Train:Test = 4:1.

**Ví dụ 2 (Bài tập thực hành 1).** Chúng ta sử dụng thư viện PCA để giảm số chiều của tập dữ liệu bệnh nhân Parkinson, với 754 trường (thuộc tính) và 756 records. Trường 'class' chứa hai giá trị 0 và 1 xác định bệnh nhân có mắc Parkinson hay không (1 ~ có mắc). Dữ liệu có trong tệp đính kèm (dạng CSV) hoặc tại link: <https://www.kaggle.com/datasets/dipayanbiswas/parkinsons-disease-speech-signal-features>.

Đoạn chương trình dưới đây đọc dữ liệu. Hãy tham khảo code ở phần trước để giảm số chiều xuống còn 02 chiều, sau đó vẽ ra màn hình để có thể quan sát:

```
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')

df= pd.read_csv("D:\\Teach_n_Train\\Machine
Learning\\code\\data\\pd_speech_features.csv")

df.head()

df['class'].value_counts()
```

### Bài tập thực hành:

- 1) Sử dụng PCA để giữ lại 200 thành phần chính của dữ liệu

- 2) Chia dữ liệu thành 500 bản ghi training và phần còn lại cho Test. Sau đó sử dụng mô hình hồi quy Logistic để dự đoán phân loại dữ liệu cho phần Test.
- 3) Sử dụng độ đo tính chính xác trong sklearn để kiểm tra kết quả.

**Ví dụ 3 (Bài tập thực hành 2):** Hãy sử dụng đoạn mã lệnh giảm số chiều bằng phương pháp phân tích thành phần chính đã học và tham khảo phần đọc, hiển thị dữ liệu hình ảnh chữ số viết tay đã có trong bài thực hành phần Multinomial Logistic Regression, để áp dụng vào dữ liệu ảnh chữ số viết tay đã được cung cấp trong bài thực hành tuần trước:

- 1) Đọc dữ liệu ảnh, lấy tập dữ liệu 5000 ảnh bất kỳ; giảm số chiều dữ liệu xuống còn 100 chiều (từ  $28 \times 28 = 784$  chiều ban đầu).
- 2) Áp dụng phương pháp phân loại nhiều lớp Multinomial Logistic Regression (tham khảo bài trước) để phân loại để phân loại tập dữ liệu đã chọn trong ý 1), tỷ lệ train:validation là 0.7:0.3. Hãy so sánh kết quả (độ chính xác và thời gian) chạy mô hình phân loại trong hai trường hợp:
  - a) Dữ liệu nguyên bản (giữ nguyên  $28 \times 28 = 784$  chiều)
  - b) Dữ liệu đã qua giảm chiều (100 chiều). Chú ý cần thực hiện giảm chiều với tập ảnh trước, sau đó mới chia thành tập train và validation.