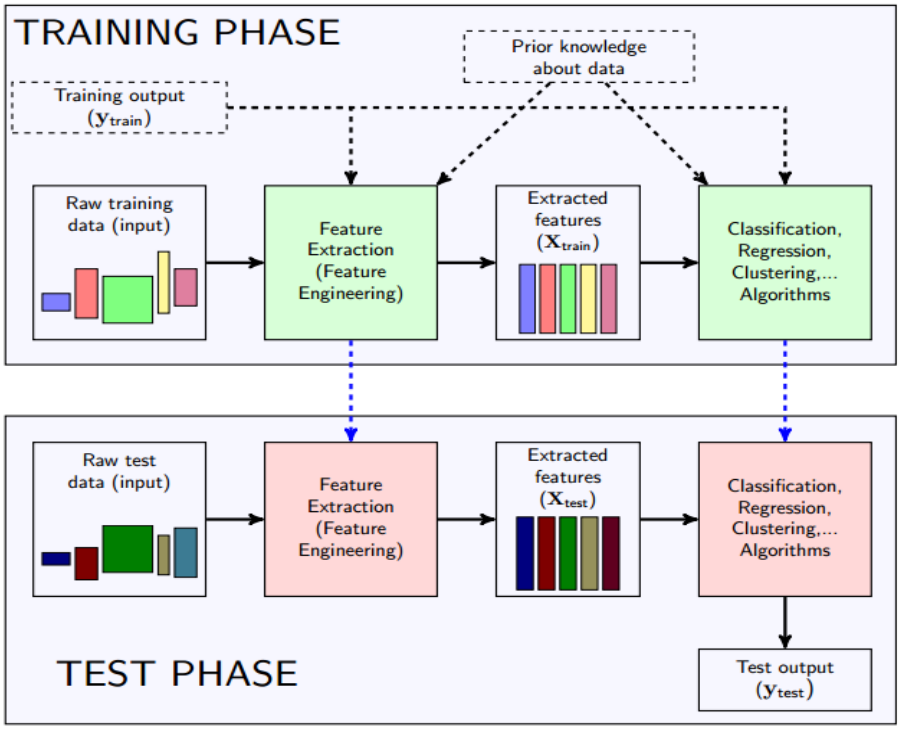
Mô hình chung cho bài toán Machine Learning



Thuật toán Máy vector hỗ trợ, Support Vector Machine (SVM)

Ví dụ: Sử dụng thuật toán SVM để xây dựng mô hình phân loại da (skin, non-skin)

File Dữ liệu:

https://archive.ics.uci.edu/ml/datasets/skin+segmentation

Mô tả dữ liệu:

Bộ dữ liệu về da được thu thập bằng cách lấy mẫu ngẫu nhiên các giá trị B, G, R từ hình ảnh khuôn mặt của các nhóm tuổi khác nhau (trẻ, trung niên và già), các nhóm chủng tộc (da trắng, da đen và châu Á) và giới tính thu được từ cơ sở dữ liệu FERET và cơ sở dữ liệu PAL. Tổng số lượng mẫu là 245057; trong đó 50859 là mẫu da và 194198 là mẫu không phải da.

Tập dữ liệu này có kích thước 245057 * 4 trong đó ba cột đầu tiên là các giá trị B, G, R (đặc trưng x1, x2 và x3) và cột thứ tư là nhãn lớp (biến quyết định y).

1. Sử dụng các thư viện: numpy, matplotlib, sklearn,… thông qua lệnh import

Import thư viện

```
import math as m
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
import seaborn as sns
```

2. Đọc file chứa dữ liệu và in ra tiêu đề các cột

```
data = pd.read_csv('datasets/uci-skin-segmentation/Skin_NonSkin.csv')
print(list(data.columns))
print(data.shape)
#targets - 'quality' column
#features - all other columns
```

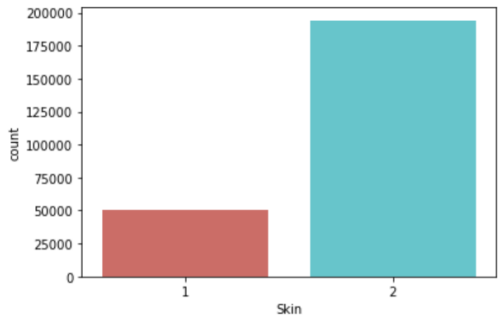
```
['Blue', 'Green', 'Red', 'Skin']
(245057, 4)
```

```
data.head()
```

	Blue	Green	Red	Skin
0	74	85	123	1
1	73	84	122	1
2	72	83	121	1
3	70	81	119	1
4	70	81	119	1

3. Vẽ đồ thị phân bổ dữ liệu stabf và số lượng mẫu

```
sns.countplot(x='Skin', data=data, palette='hls')
plt.show()
```



4. Chuyển đổi dữ liệu cột đầu ra như sau: 1->0, 2->1

```
data['Skin'] = data['Skin'] - 1
data.head()
```

	Blue	Green	Red	Skin
0	74	85	123	0
1	73	84	122	0
2	72	83	121	0
3	70	81	119	0
4	70	81	119	0

5. Chuyển dữ liệu thành mảng:

```
X = np.array(data.loc[:, data.columns != 'Skin'])
#X.head()
print(X)
```

```
[[ 74  85 123]
 [ 73  84 122]
 [ 72  83 121]
 ...
 [163 162 112]
 [163 162 112]
 [255 255 255]]
```

```
y = np.array(data.loc[:, data.columns == 'Skin'])
#y.head()
print(y)
```

```
[[0]
 [0]
 [0]
 ...
 [1]
 [1]
 [1]]
```

6. Phân chia tập dữ liệu ra thành 2 tập: tập huấn luyện và tập kiểm tra:

```
train_features, test_features, train_targets, test_targets = train_test_split(X, y, test_size=0.95, random_state=0)
print("##### Training and test datasets #####")
print("Training size: ", len(train_targets))
print("Test size: ", len(test_targets))
```

```
##### Training and test datasets #####
Training size:  12252
Test size:  232805
```

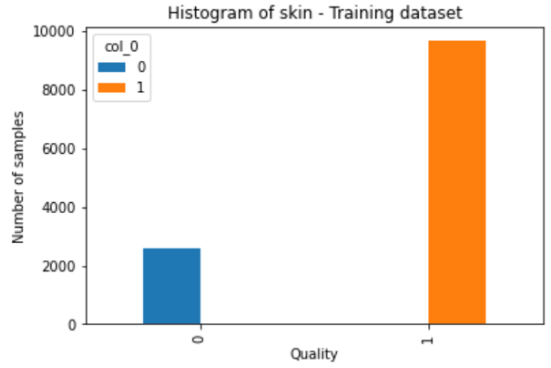
7. Biến đổi kích thước ma trận dữ liệu huấn luyện kiểm tra:

```
train_targets = train_targets.reshape(train_targets.shape[0])
test_targets = test_targets.reshape(test_targets.shape[0])
print(train_targets)
```

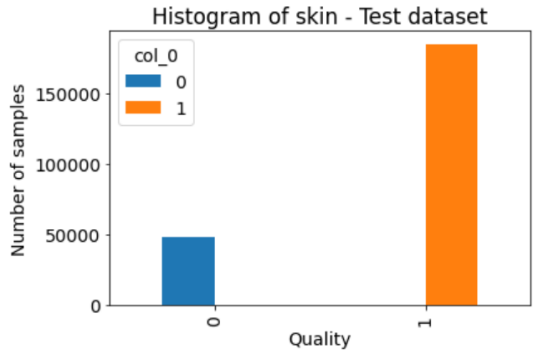
```
[1 1 1 ... 1 0 1]
```

8. Vẽ đồ thị

```
pd.crosstab(train_targets,train_targets).plot(kind='bar')
plt.title('Histogram of skin - Training dataset')
plt.xlabel('Quality')
plt.ylabel('Number of samples')
plt.rc("font", size=14)
```




```
pd.crosstab(test_targets,test_targets).plot(kind='bar')
plt.title('Histogram of skin - Test dataset')
plt.xlabel('Quality')
plt.ylabel('Number of samples')
plt.rc("font", size=14)
```



9. Tạo mô hình sử dụng thuật toán SVM, huấn luyện mô hình:

```
svmClassifier = SVC(kernel = 'linear', C = 10)
svmClassifier.fit(train_features, train_targets)
print(svmClassifier.coef_)
print(svmClassifier.intercept_)
```

```
[[ 0.01654937 -0.00482597 -0.01950881]]
[1.89284263]
```

10. Hiển thị độ chính xác của mô hình đối với tập huấn luyện:

```
train_predictions = svmClassifier.predict(train_features)
print("##### Training - Prediction results of SVM #####")
print("Target labels:     ", train_targets)
print("Prediction labels: ", train_predictions)
```

```
##### Training - Prediction results of SVM #####
Target labels:      [1 1 1 ... 1 0 1]
Prediction labels:  [1 1 1 ... 1 0 1]
```

```
accuracy = 100 * accuracy_score(train_targets, train_predictions)
print("##### Training - Prediction accuracy of SVM #####")
print("Accuracy of Logistic Regression:     ", accuracy)
print(classification_report(train_targets, train_predictions))
```

```
##### Training - Prediction accuracy of SVM #####
Accuracy of Logistic Regression:      92.67058439438459
              precision    recall  f1-score   support

           0       0.77      0.94      0.84      2601
           1       0.98      0.92      0.95      9651

    accuracy                           0.93     12252
   macro avg       0.88      0.93      0.90     12252
weighted avg       0.94      0.93      0.93     12252
```

11. Hiển thị độ chính xác của mô hình đối với tập kiểm tra:

```
test_predictions = svmClassifier.predict(test_features)
print("##### Testing - Prediction results of SVM #####")
print("Target labels:     ", test_targets)
print("Prediction labels: ", test_predictions)
```

```
##### Testing - Prediction results of SVM #####
Target labels:      [1 0 1 ... 1 1 1]
Prediction labels:  [1 0 1 ... 1 0 1]
```

```
accuracy = 100 * accuracy_score(test_targets, test_predictions)
print("##### Testing - Prediction accuracy of SVM #####")
print("Accuracy of Logistic Regression:     ", accuracy)
print(classification_report(test_targets, test_predictions))
```

```
##### Testing - Prediction accuracy of SVM #####
Accuracy of Logistic Regression:      92.8270440926956
              precision    recall  f1-score   support

           0       0.77      0.94      0.84     48258
           1       0.98      0.93      0.95    184547

    accuracy                           0.93    232805
   macro avg       0.88      0.93      0.90    232805
weighted avg       0.94      0.93      0.93    232805
```

Bài tập: Sinh viên nhận xét về độ chính xác của thuật toán.