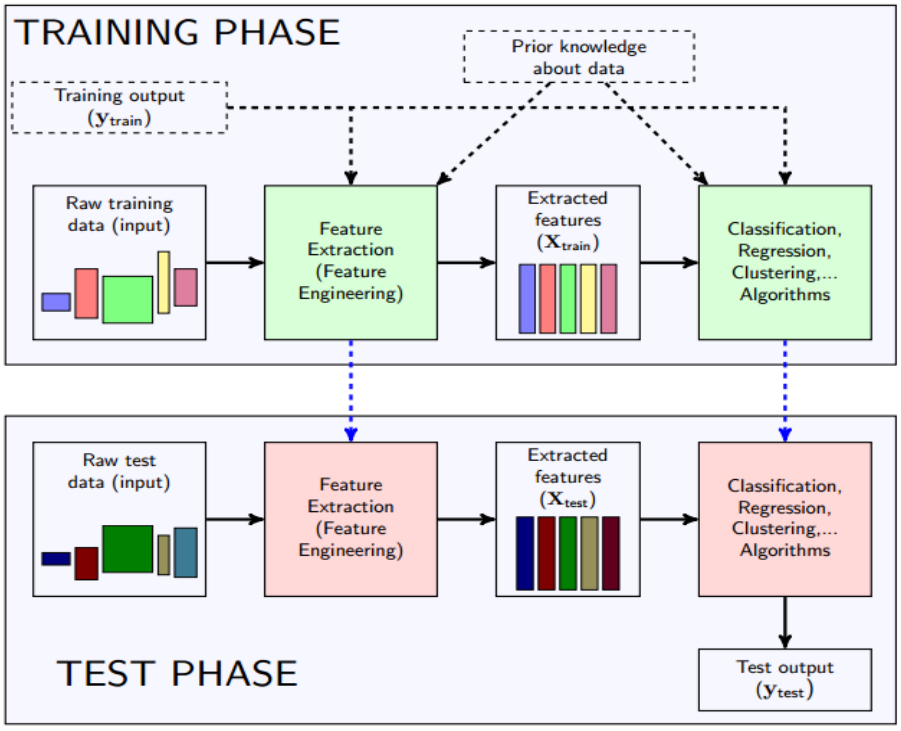
Mô hình chung cho bài toán Machine Learning



Thuật toán Softmax Regression

Ví dụ: Sử dụng thuật toán Softmax Regression để xây dựng mô hình hóa chất lượng rượu vang trắng dựa trên các thử nghiệm hóa lý

File Dữ liệu:

https://archive.ics.uci.edu/ml/datasets/wine+quality

Mô tả dữ liệu:

Attribute Information: For more information, read [Cortez et al., 2009].
Input variables (based on physicochemical tests):
1 - fixed acidity - độ axit cố định
2 - volatile acidity - tính axit dễ bay hơi
3 - citric acid - axit xitric
4 - residual sugar - đường dư
5 - chlorides - clorua
6 - free sulfur dioxide -
7 - total sulfur dioxide - tổng lưu huỳnh điôxit
8 - density - mật độ
9 - pH
10 - sulphates
11 - alcohol
Output variable (based on sensory data):
12 - quality (score between 0 and 10) - chất lượng được đánh giá bởi điểm nằm trong khoảng từ 0 đến 10

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.linear_model import LogisticRegression
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-wine-quality/winequality-white.csv')
print(list(data.columns))
#targets - 'quality' column
#features - all other columns
```

```
['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar', 'chlorides', 'free sulfur dioxide', 'total sulfur dioxide',
'density', 'pH', 'sulphates', 'alcohol', 'quality']
```

In ra dữ liệu 5 dòng đầu tiên:

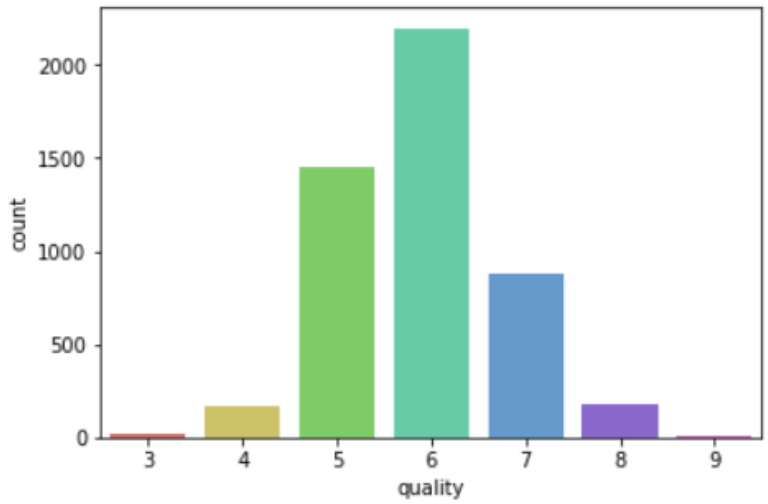
```
data.head()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.0010	3.00	0.45	8.8	6
1	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.9940	3.30	0.49	9.5	6
2	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.9951	3.26	0.44	10.1	6
3	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	6
4	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	6

3. **Vẽ đồ thị phân bố dữ liệu quality và số lượng mẫu**

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

Kết quả:



4. **Thêm cột category, với giá trị catergory cụ thể như sau:**
   quality < 6 thì category = 0
   quality > 6 thì category = 2
   quality = 6 thì category = 1

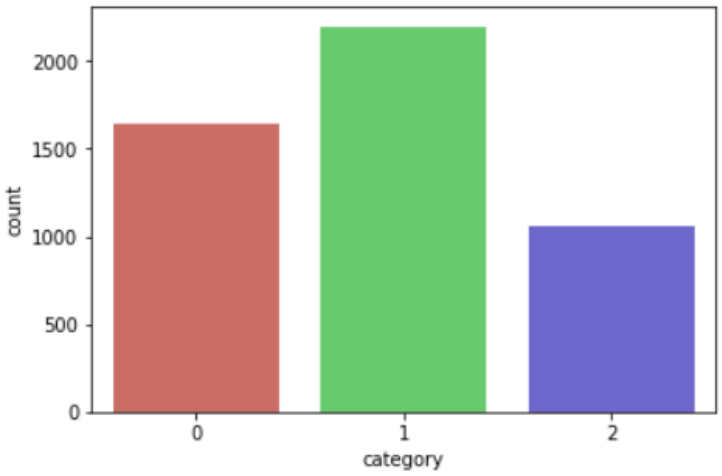
```
c = []
for q in data['quality'].values:
    if q < 6:
        c.append(0)
    elif q > 6:
        c.append(2)
    else:
        c.append(1)

data['category'] = c
data.head()
```

Kết quả:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	category
0	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.0010	3.00	0.45	8.8	6	1
1	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.9940	3.30	0.49	9.5	6	1
2	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.9951	3.26	0.44	10.1	6	1
3	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	6	1
4	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	6	1

5. **Vẽ đồ thị phân bổ dữ liệu category và số lượng mẫu**

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



6. Giá trị Đặc trưng, gán data_features giá trị từ các cột dữ liệu:

```
data_features = data.columns.values.tolist()
print(data_features)
remove_features = ['quality']
to_keep_features = [i for i in data_features if i not in remove_features]
data_final = data[to_keep_features]
data_final.head()
```

Kết quả:

```
['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar', 'chlorides', 'free sulfur dioxide', 'total sulfur dioxide',
'density', 'pH', 'sulphates', 'alcohol', 'quality', 'category']
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	category
0	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.0010	3.00	0.45	8.8	1
1	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.9940	3.30	0.49	9.5	1
2	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.9951	3.26	0.44	10.1	1
3	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	1
4	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	1

7. Chuyển dữ liệu thành mảng:

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

```
[[ 7.    0.27  0.36 ...  0.45  8.8   6.  ]
 [ 6.3   0.3   0.34 ...  0.49  9.5   6.  ]
 [ 8.1   0.28  0.4  ...  0.44 10.1   6.  ]
 ...
 [ 6.5   0.24  0.19 ...  0.46  9.4   6.  ]
 [ 5.5   0.29  0.3  ...  0.38 12.8   7.  ]
 [ 6.    0.21  0.38 ...  0.32 11.8   6.  ]]
```

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

```
[[1]
 [1]
 [1]
 ...
 [1]
 [2]
 [1]]
```

8. 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=0.3, 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:  3428
Test size:  1470
```

9. 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)
```

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

10. Tạo mô hình sử dụng thuật toán Softmax Regression, huấn luyện mô hình:

```
#change max_iter lower and higher to see results
classifier_logreg = LogisticRegression(multi_class='multinomial', solver='lbfgs', max_iter=1000000)
classifier_logreg.fit(train_features, train_targets)
predictions = classifier_logreg.predict(test_features)
print(classifier_logreg.coef_)
print(classifier_logreg.intercept_)
```

Kết quả:

```
[[ 7.95930228e-02  4.10107137e+00  2.99314781e-01 -5.12251957e-02
   6.25722809e-01 -1.28034618e-02  2.65924298e-03  3.29338339e-01
  -3.33755517e-01 -9.84292018e-01 -7.93993014e-01]
 [-8.56717198e-02 -1.19987537e+00  1.22718677e-01  5.52658087e-03
   4.61446869e-01  5.59068617e-04  2.03994345e-05  9.77283435e-02
  -4.19512823e-01  1.14040226e-01  3.18997948e-02]
 [ 6.07869698e-03 -2.90119600e+00 -4.22033459e-01  4.56986148e-02
  -1.08716968e+00  1.22443932e-02 -2.67964241e-03 -4.27066682e-01
   7.53268340e-01  8.70251791e-01  7.62093219e-01]]
[  8.12408444   2.2426032  -10.36668763]
```

11. Hiển thị nhãn và độ chính xác của mô hình đối với tập huấn luyện:

```
1 train_predictions = classifier_logreg.predict(train_features)
2 print("##### Training - Prediction results of Softmax Regression #####")
3 print("Target labels:     ", train_targets)
4 print("Prediction labels: ", train_predictions)
```

```
##### Training - Prediction results of Softmax Regression #####
Target labels:      [0 1 0 ... 1 1 0]
Prediction labels:  [0 2 1 ... 0 1 0]
```

```
1 accuracy = 100 * accuracy_score(train_targets, train_predictions)
2 print("##### Training - Prediction accuracy of Softmax Regression #####")
3 print("Accuracy of Logistic Regression:     ", accuracy)
4 print(classification_report(train_targets, train_predictions))
```

```
##### Training - Prediction accuracy of Softmax Regression #####
Accuracy of Logistic Regression:      57.90548424737456
              precision    recall  f1-score   support

           0       0.65      0.56      0.60      1113
           1       0.54      0.71      0.61      1568
           2       0.59      0.35      0.44       747

    accuracy                           0.58      3428
   macro avg       0.60      0.54      0.55      3428
weighted avg       0.59      0.58      0.57      3428
```

12. Hiển thị kết quả đích và kết quả dự đoán của mô hình đối với tập huấn luyện:

```
1 test_predictions = classifier_logreg.predict(test_features)
2 print("##### Testing - Prediction results of Softmax Regression #####")
3 print("Target labels:     ", test_targets)
4 print("Prediction labels: ", test_predictions)
```

```
##### Testing - Prediction results of Softmax Regression #####
Target labels:      [0 1 2 ... 1 1 1]
Prediction labels:  [0 0 1 ... 1 1 1]
```

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

14. Hiển thị kết quả đích và kết quả dự đoán của mô hình đối với tập kiểm tra:

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

```
###### Testing - Prediction results of MLP Backpropagation ######
Target labels:      [0 1 2 ... 1 1 1]
Prediction labels:  [0 1 2 ... 1 1 1]
```

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

```
:
1 accuracy = 100 * accuracy_score(test_targets, test_predictions)
2 print("##### Training - Prediction accuracy of Softmax Regression #####")
3 print("Accuracy of Logistic Regression:     ", accuracy)
4 print(classification_report(test_targets, test_predictions))
```

```
##### Training - Prediction accuracy of Softmax Regression #####
Accuracy of Logistic Regression:      54.21768707482993
              precision    recall  f1-score   support

           0       0.66      0.53      0.59       527
           1       0.49      0.68      0.57       630
           2       0.50      0.28      0.36       313

    accuracy                           0.54      1470
   macro avg       0.55      0.50      0.51      1470
weighted avg       0.56      0.54      0.53      1470
```

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