

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM

KHOA ĐÀO TẠO CHẤT LƯỢNG CAO



ĐỒ ÁN 2

TÌM HIỂU THUẬT TOÁN ELASTIC NET REGRESSION

GVHD: ThS. Trần Nhật Quang

SVTH: MSSV

Đào Xuân Thủy 16110544

Lâm Phước Bảo 16110016

182PROJ312979_02CLC

Tp. Hồ Chí Minh, tháng 5 năm 2019

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Giáo viên hướng dẫn

ThS. Trần Nhật Quang

LỜI CAM ĐOAN

Chúng em xin cam đoan đồ án này do chính chúng em thực hiện. Chúng em không sao chép, sử dụng bất kỳ tài liệu, mã nguồn của người khác mà không ghi rõ nguồn gốc. Chúng em xin chịu hoàn toàn trách nhiệm nếu vi phạm.

Thành phố Hồ Chí Minh, ngày 19 tháng 05 năm 2019

Sinh viên thực hiện

Thuy

Bao

Đào Xuân Thủy

Lâm Phước Bảo

LỜI CẢM ƠN

Để hoàn thành đồ án này, chúng em xin gửi lời cảm ơn chân thành đến thầy Trần Nhật Quang, người đã hỗ trợ và giúp đỡ chúng em trong suốt quá trình thực hiện đồ án, nhận xét và góp ý cũng như cung cấp tài liệu tham khảo, giúp chúng em có thể hoàn thành một cách tốt nhất. Nếu không có sự hướng dẫn và kinh nghiệm thực tiễn của thầy, chúng em nghĩ rằng đồ án môn học của chúng em khó có thể hoàn thiện và hoàn thành đúng thời hạn. Một lần nữa chúng em xin cảm ơn thầy.

Chúng em xin gửi lời cảm ơn đến các bạn cùng khóa đã cung cấp nhiều thông tin và kiến thức hữu ích giúp cho chúng em hoàn thiện đồ án này.

Đồ án này được thực hiện trong thời gian có hạn, cùng với kiến thức còn hạn chế và còn nhiều bờ ngõ khác, do đó thiếu sót là điều không thể tránh khỏi nên chúng em rất mong nhận được những ý kiến đóng góp quý báu của mọi người để kiến thức của chúng em được hoàn thiện hơn sau này. Chúng em xin chân thành cảm ơn.

Thành phố Hồ Chí Minh, ngày 19 tháng 05 năm 2019

Sinh viên thực hiện

Thuy

Bao

Đào Xuân Thủy

Lâm Phước Bảo

MỤC LỤC

CHƯƠNG 1: TÌM HIỂU VỀ ĐẠO VĂN	2
1. Đạo văn là gì?	2
2. Những điều nên làm.....	2
3. Những điều không nên làm	2
CHƯƠNG 2: TÌM HIỂU LÝ THUYẾT.....	3
2.1 Mô tả thuật toán.....	3
2.2 Công thức tính	4
2.2.1 Tìm các hệ số β	4
2.2.2 Công thức tính cost function	5
2.2.3 Công thức tính hệ số phạt.....	5
2.3 Hiệu suất của thuật toán.....	5
2.3.1 Công thức R^2 ^[1]	5
2.3.2 Phương pháp MSE	6
2.4 So sánh với các thuật toán khác	6
2.5 Cross validation ^[3]	7
CHƯƠNG 3: LẬP TRÌNH	8
3.1 Bộ dữ liệu cho thuật toán	8
3.2 Mục tiêu thuật toán.....	11
3.3 Cài đặt thuật toán.....	11
3.4 Kết quả thu được	11
3.5 So sánh với thuật toán khác	12
3.6 Thư viện lập trình	15
BẢNG PHÂN CÔNG.....	16
TÀI LIỆU THAM KHẢO	17

CHƯƠNG 1: TÌM HIỂU VỀ ĐẠO VĂN

1. Đạo văn là gì?

Đạo văn với mức độ nghiên cứu khoa học của sinh viên có thể được hiểu là sử dụng công trình hay tác phẩm của người khác, lấy ý tưởng của người khác, sao chép nguyên bản từ ngữ của người khác mà không ghi nguồn, sử dụng cấu trúc và cách lý giải của người khác mà không ghi nhận họ, và lấy những thông tin chuyên ngành mà không đề rõ nguồn gốc.

Đạo văn được xem là hành vi thiếu trung thực về mặt học thuật và vi phạm đạo đức rất nghiêm trọng. Ở cấp độ sinh viên, đạo văn sẽ khiến kết quả nghiên cứu bị huỷ bỏ tùy thuộc vào mức độ nghiêm trọng của hành vi. Ở cấp độ nghiên cứu chuyên nghiệp, người đạo văn có thể bị buộc thôi việc, thu hồi công trình đã công bố hoặc huỷ bỏ chức danh.

2. Những điều nên làm

- Ghi rõ nguồn khi tham khảo tài liệu từ nguồn bên ngoài.
- Tôn trọng những sản phẩm, nội dung, công trình của người khác khi sử dụng.
- Nếu tài liệu muốn sử dụng quan trọng, cần xin ý kiến của chủ sở hữu trước khi sử dụng

3. Những điều không nên làm

- Sao chép công trình, tác phẩm mà không ghi rõ nguồn.
- Lấy ý tưởng của người khác để sử dụng cho mình.
- Sao chép cấu trúc và cách lý giải của người khác.

CHƯƠNG 2: TÌM HIỂU LÝ THUYẾT

2.1 Mô tả thuật toán

Trong quá trình huấn luyện cho model chúng ta sẽ không thể tránh được trường hợp **overfitting**, có nhiều cách để tránh overfitting:

- Giảm số lượng feature
- Sử dụng phương pháp Regularization:
 - + L1 Regularization (Lasso Regularization)
 - + L2 Regularization (Ridge Regularization)
 - + Elastic Net Regularization

Như chúng ta đã biết với phương pháp Lasso Regression thực hiện tốt nhất khi mà Model chứa nhiều giá trị thừa và không có ý nghĩa.

Ví dụ: Nếu đây là model sử dụng để dự đoán Size

$$\text{Size} = \text{y-intercept} + \text{slope} * \mathbf{\text{Weight}} + \text{diet difference} * \mathbf{\text{High Fat Diet}} + \text{astrological offset} * \mathbf{\text{Sign}} + \text{airspeed scalar} * \mathbf{\text{Airspeed of Swallow}}$$

Phương pháp Lasso Regression sẽ giữ giá trị **Weight** và **High Fat Diet** và ngược lại nó sẽ khử đi giá trị của **Sign** và **Airspeed of Swallow** vì 2 giá trị này không có ý nghĩa gì trong quá trình dự đoán **Size**

$$\text{Size} = \text{y-intercept} + \text{slope} * \mathbf{\text{Weight}} + \text{diet difference} * \mathbf{\text{High Fat Diet}} + \text{astrological offset} * \mathbf{\text{Sign}} + \text{airspeed scalar} * \mathbf{\text{Airspeed of Swallow}}$$

Kết quả là tạo ra một model đơn giản và phù hợp hơn

$$\text{Size} = \text{y-intercept} + \text{slope} * \mathbf{\text{Weight}} + \text{diet difference} * \mathbf{\text{High Fat Diet}}$$

Phương pháp Ridge Regression sẽ làm tốt nhất khi hầu hết các giá trị trong Model có ý nghĩa và hữu dụng

Ví dụ: Nếu đây là model sử dụng để dự đoán Size

$$\text{Size} = \text{y-intercept} + \text{slope} * \mathbf{\text{Weight}} + \text{diet difference} * \mathbf{\text{High Fat Diet}} + \text{Slope}_2 * \mathbf{\text{Age}} + \text{Slope}_3 * \mathbf{\text{Size of Father}}$$

Với hầu hết các giá trị trên đều mang ý nghĩa ảnh hưởng đến Size.

Ridge Regression sẽ làm giảm các giá trị xuống nhưng không hoàn toàn loại bỏ như Lasso Regression

Vì thế, nếu chúng ta biết rõ hết các dữ liệu chúng ta đưa vào là như thế nào thì chúng ta có thể dễ dàng quyết định được nên chọn sử dụng Lasso hay Ridge.

Tuy nhiên, chúng ta sẽ làm gì có quá nhiều các feature trong dataset?

Chúng ta không thể nào biết được tất cả chúng liệu rằng là những giá trị tốt cho việc huấn luyện hay chỉ là những giá trị vô nghĩa. Chính vì thế mà **Elastic-Net Regression** đã ra đời

Lasso Regression:

The sum of the squared residuals + $\lambda^* \text{variable}_1 + \dots + \text{variable}_2$
--

Ridge Regression:

The sum of the squared residuals + $\lambda^*(\text{variable}_1)^2 + \dots + (\text{variable}_2)^2$

Elastic-Net Regression:

The sum of the squared residuals + $\lambda^* \text{variable}_1 + \dots + \text{variable}_2 + \lambda^*(\text{variable}_1)^2 + \dots + (\text{variable}_2)^2$

2.2 Công thức tính

2.2.1 Tìm các hệ số β

Có nhiều cách khác nhau để tìm các hệ số β

Cách 1: Normal Equation

Với ma trận trên, ta có phương trình:

$$X.\beta = y \quad \leftrightarrow \quad X^T.X.\beta = X^T.y$$

$$\leftrightarrow \quad (X^T.X)^{-1}.X^T.X.\beta = (X^T.X)^{-1}.X^T.y$$

$$\leftrightarrow \quad \beta = (X^T.X)^{-1}.X^T.y$$

Cách 2: Gradient descent

Lặp lại tới khi hội tụ	$j = \overline{0, m}$: m là số lượng β
{	trong công thức

$\beta_j := \beta_j - \alpha \frac{\partial}{\partial \beta_j} J(\beta)$	α : learning rate
}	

Một số thuật toán khác như: Conjugate gradient, BFGS, L_BFGS...

2.2.2 Công thức tính cost function

$$J(\beta) = \frac{1}{2n} \sum_{i=1}^n (h_{\theta}(x^{(i)}) - y^{(i)})^2 + \text{hệ số phạt}$$

n: số dòng dữ liệu mẫu

$h_{\theta}(x^{(i)})$: kết quả đầu ra tính được từ thuật toán

$y^{(i)}$: kết quả đúng từ dữ liệu mẫu

Penalty: hệ số phạt

2.2.3 Công thức tính hệ số phạt

$$P_{\alpha} = P_{Lasso} + P_{Ridge} = \lambda \cdot \sum_{i=1}^n \left[\alpha [\beta_j] + \frac{1}{2} (1 - \alpha) \beta_j^2 \right] \text{ với } 0 < \alpha < 1$$

P_{Lasso} : penalty của Lasso Regression

P_{Ridge} : Penalty của Ridge Regression

n: số dòng dữ liệu mẫu

2.3 Hiệu suất của thuật toán

2.3.1 Công thức R^2 [1]

$$R^2 = \frac{TSS - RSS}{TSS}$$

Trong đó:

- TSS: là một phép đo tổng biến thiên trong tỷ lệ đáp ứng / biến phụ thuộc Y và có thể được coi là số lượng biến thiên vốn có trong đáp ứng trước khi hồi quy được thực hiện.
- RSS đo lường lượng biến đổi còn lại không giải thích được sau khi thực hiện hồi quy.

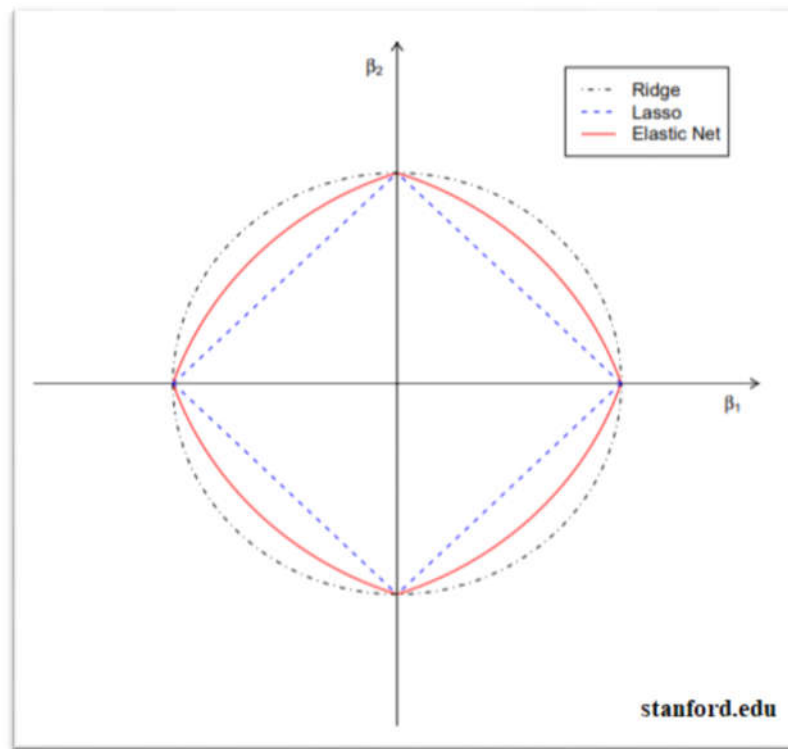
- (TSS - RSS) đo lường mức độ thay đổi trong đáp ứng được giải thích (hoặc loại bỏ) bằng cách thực hiện hồi quy.

2.3.2 Phương pháp MSE

$$MSE = \frac{1}{n} \sum_{i=1}^n (h(\beta)^{(i)} - y^{(i)})^2$$

MSE (Mean squared error) của một phép ước lượng là trung bình của bình phương của sai số, tức là sự khác biệt giữa các giá trị được mô hình dự đoán và giá trị thực. MSE là một hàm rủi ro, tương ứng với giá trị kỳ vọng của sự mất mát sai số bình phương hoặc mất mát bậc hai. MSE là moment bậc hai (về nguồn gốc) của sai số là moment bậc hai (về nguồn gốc) của sai số. [2]

2.4 So sánh với các thuật toán khác



Hình 1. So sánh Elastic Net với Lasso và Ridge

Như đã nêu trên, Elastic Net Regression sử dụng hàm số phạt từ hai hệ số phạt của mô hình Ridge Regression và Lasso Regression, vậy nên biểu đồ học được sẽ có sự chính xác hơn và khắc phục được điểm yếu từ 2 biểu đồ trước. Từ đó, ta thấy được thuật

toán này khắc phục được hầu hết các điểm yếu của các thuật toán trước kia và cho được kết quả đáng tin tưởng hơn.

2.5 Cross validation ^[3]

Nhiều trường hợp, chúng ta cần hạn chế số lượng dữ liệu để xây dựng mô hình. Nếu lấy quá nhiều dữ liệu trong tập training ra làm dữ liệu validation, phần dữ liệu còn lại của tập training là không đủ để xây dựng mô hình. Lúc này, tập validation phải thật nhỏ để giữ được lượng dữ liệu cho training đủ lớn. Tuy nhiên, một vấn đề khác nảy sinh. Khi tập validation quá nhỏ, hiện tượng overfitting lại có thể xảy ra với tập training còn lại.

Cross validation là một cải tiến của validation với lượng dữ liệu trong tập validation là nhỏ nhưng chất lượng mô hình được đánh giá trên nhiều tập validation khác nhau. Một cách thường dùng sử dụng là chia tập training ra thành các tập con không có phần tử chung, có kích thước gần bằng nhau. Tại mỗi lần kiểm thử, được gọi là run, một trong số các tập con được lấy ra làm validation set. Mô hình sẽ được xây dựng dựa vào tập của $k-1$ tập con còn lại. Mô hình cuối được xác định dựa trên trung bình của các train error và validation error. Cách làm này còn có tên gọi là k-fold cross validation.

Khi k bằng với số lượng phần tử trong tập training ban đầu, tức mỗi tập con có đúng 1 phần tử, ta gọi kỹ thuật này là leave-one-out.

CHƯƠNG 3: LẬP TRÌNH

3.1 Bộ dữ liệu cho thuật toán

Chủ đề: Đánh giá chất lượng của các loại rượu dựa trên nồng độ các thành phần chứa trong rượu.

Địa chỉ tải về: <https://www.kaggle.com/uciml/red-wine-quality-cortez-et-al-2009>

Kích thước bộ dữ liệu: 1599 x 12.

fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
7.4	0.7	0	1.9	0.076	11	34	0.9978	3.51	0.56	9.4	5
7.8	0.88	0	2.6	0.098	25	67	0.9968	3.2	0.68	9.8	5
7.8	0.76	0.04	2.3	0.092	15	54	0.997	3.26	0.65	9.8	5
11.2	0.28	0.56	1.9	0.075	17	60	0.998	3.16	0.58	9.8	6
7.4	0.7	0	1.9	0.076	11	34	0.9978	3.51	0.56	9.4	5
7.4	0.66	0	1.8	0.075	13	40	0.9978	3.51	0.56	9.4	5
7.9	0.6	0.06	1.6	0.069	15	59	0.9964	3.3	0.46	9.4	5
7.3	0.65	0	1.2	0.065	15	21	0.9946	3.39	0.47	10	7
7.8	0.58	0.02	2	0.073	9	18	0.9968	3.36	0.57	9.5	7
7.5	0.5	0.36	6.1	0.071	17	102	0.9978	3.35	0.8	10.5	5
6.7	0.58	0.08	1.8	0.097	15	65	0.9959	3.28	0.54	9.2	5
7.5	0.5	0.36	6.1	0.071	17	102	0.9978	3.35	0.8	10.5	5
5.6	0.615	0	1.6	0.089	16	59	0.9943	3.58	0.52	9.9	5
7.8	0.61	0.29	1.6	0.114	9	29	0.9974	3.26	1.56	9.1	5
8.9	0.62	0.18	3.8	0.176	52	145	0.9986	3.16	0.88	9.2	5
8.9	0.62	0.19	3.9	0.17	51	148	0.9986	3.17	0.93	9.2	5
8.5	0.28	0.56	1.8	0.092	35	103	0.9969	3.3	0.75	10.5	7
8.1	0.56	0.28	1.7	0.368	16	56	0.9968	3.11	1.28	9.3	5

Hình 2. Mô tả dữ liệu mẫu cho thuật toán

Trước khi đưa vào model chúng ta sẽ phải xử lý dữ liệu đầu vào:

Bước 1: Đầu tiên chúng ta sẽ khai báo những thư viện sẽ sử dụng

```
#Importing libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, mean_squared_error, r2_score
```

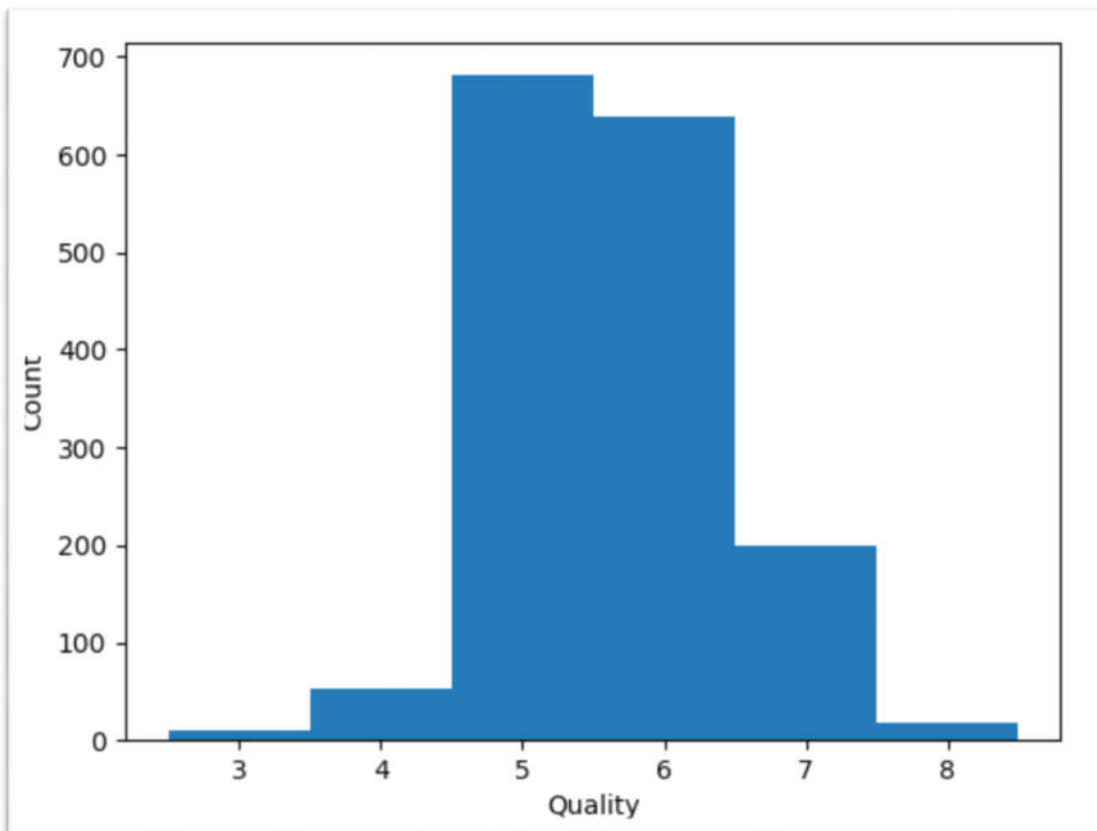
Bước 2: tiếp theo sẽ đọc file từ dataset (file csv)

```
#Initial dataset observations
df = pd.read_csv('Dataset/winequality-red.csv')
```

Bước 3: Vẽ ra biểu đồ thể hiện số lượng dataset cho từng label

```
#Vẽ ra thể hiện số lượng sản phẩm theo từng thông số chất lượng
def plot_wine_quality_histogram(quality):
    #chọn ra những cột unique và xếp lại theo thật tự
    unique_vals = df['quality'].sort_values().unique()
    plt.xlabel("Quality")
    plt.ylabel("Count")
    plt.hist(quality.values, bins=np.append(unique_vals, 9), align='left')
    plt.show()
plot_wine_quality_histogram(df['quality'])
```

Kết quả sẽ được:



Hình 3. Kết quả thu được khi chạy thuật toán

Bước 4: Tách feature và label

```
#y: label ( cột quality )
y = df.quality
#X: dataset bỏ cột quality
X = df.drop('quality', axis=1)
```

Bước 5: Chia số lượng data training và data test sử dụng để đánh giá model sau khi train

```
#tách bộ dữ liệu ra thành training (80%) and test (20%)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0, stratify=y)
```

Bước 6: Scaling dữ liệu

Khi nhìn vào các record của feature chúng ta sẽ thấy độ chênh lệch giữa các số khá lớn với nhau. Chính vì thế chúng ta cần thực hiện thêm một bước là Scale dữ liệu:

```
#scale dữ liệu train
scaler = preprocessing.StandardScaler().fit(X_train)
X_train_scaled = scaler.transform(X_train)
X_test_scaled = scaler.transform(X_test)
|
```

Trước đó X_train sẽ là:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
12	10.8	0.450	0.13	2.5	0.099	20.0	18.0	0.99818	3.14	0.71	10.000000
149	7.1	0.400	0.01	2.3	0.079	20.0	37.0	0.99514	3.40	0.61	10.000000
121	7.5	0.400	0.18	1.6	0.079	31.0	58.0	0.99450	3.34	0.58	9.000000
13	7.1	0.430	0.42	5.5	0.071	20.0	120.0	0.99730	3.42	0.72	10.500000
77	9.4	0.270	0.53	2.4	0.074	0.0	18.0	0.99420	3.20	1.13	11.000000
14	5.3	0.570	0.01	1.7	0.054	0.0	27.0	0.99340	3.57	0.84	12.500000
14	7.7	0.690	0.05	2.7	0.075	15.0	27.0	0.99740	3.20	0.61	9.100000
175	9.1	0.250	0.34	2.0	0.071	43.0	67.0	0.99760	3.44	0.86	10.100000
15	8.9	0.310	0.57	2.0	0.111	20.0	85.0	0.99710	3.30	0.53	9.700000
124	8.0	0.430	0.19	2.0	0.075	22.0	47.0	0.99522	3.39	0.70	10.000000
17	10.0	0.560	0.14	2.2	0.079	10.0	54.0	0.99510	3.18	0.56	10.100000
17	9.0	0.820	0.14	2.6	0.080	9.0	23.0	0.99840	3.39	0.63	9.000000
144	9.9	0.500	0.26	2.0	0.111	7.0	60.0	0.99700	3.04	0.30	10.100000
11	5.8	1.010	0.66	2.0	0.030	15.0	80.0	0.99357	3.66	0.60	11.500000
18	7.2	0.530	0.07	1.4	0.074	5.0	20.0	0.99730	3.12	0.81	9.000000
175	8.0	0.715	0.23	3.3	0.075	13.0	81.0	0.99480	3.34	0.54	9.500000

Sau khi X_train được Scale

```
D:\Python37-32\python.exe
[[ 1.49013921 -0.43246244  0.30109594 ... -0.4727928  0.2956587
  0.37235095]
 [-0.70033256  0.41953919 -1.34441809 ...  0.57114676 -0.28131334
  0.46712131]
 [-0.46352481 -0.71646299 -0.47023876 ...  0.17966942 -0.45440495
 -0.95443411]
 ...
 [ 0.30610041  0.4763393  -0.67592801 ... -0.79902391 -0.39670775
 -0.85966375]
 [-0.87793838  0.81713996 -1.3958404  ...  0.57114676  0.46875031
 -0.67012303]
```

3.2 Mục tiêu thuật toán

Xây dựng thuật toán dự đoán thước đo chất lượng của rượu dựa trên những tiêu chí về nồng độ các thành phần trong rượu.

3.3 Cài đặt thuật toán

Bước 1: Thiết lập Model

```
from sklearn.linear_model import ElasticNetCV
#n_alphas (int) số lượng số alphas trong quá trình regularization, được sử dụng ch
n_alphas = 300
#float giữa 0 và 1 truyền vào ElasticNet (scaling giữa L1 và L2 penalties)
l1_ratio = [.1, .3, .5, .7, .9]
#cv: số lượng k tập training được chia ra thành tập con => K-fold cross validation
Model = ElasticNetCV(n_alphas=n_alphas, l1_ratio=l1_ratio, cv=10, random_state=0)
```

Số lượng alphas là 300

L1_ratio là chỉ số λ giữa L1 và L2: 0.1, 0.3, 0.5, 0.7 và 0.9

```
Model = ElasticNetCV(n_alphas=n_alphas, l1_ratio=l1_ratio, cv=10, random_state=0)
```

Bước 2: Sau khi thiết lập hoàn tất chúng ta thực hiện training model bằng lệnh:

```
Model.fit(X_train_scaled, y_train)
```

Với X_train_scaled: là dữ liệu training đã được tách ra và scale lại ở phần trước

y_train là true answer

3.4 Kết quả thu được

Sau khi model đã được train ta tiến hành đánh giá model thu được bằng những phương pháp sau:

```
y_pred_train = Model.predict(X_train_scaled)
y_pred_test = Model.predict(X_test_scaled)
#metrics_en đánh giá giữa label có sẵn và label test đã được tách ra trước khi train
metrics_en = [accuracy_score(y_test, np.round(y_pred_test)),
              mean_squared_error(y_test, y_pred_test),
              r2_score(y_test, y_pred_test)]
```

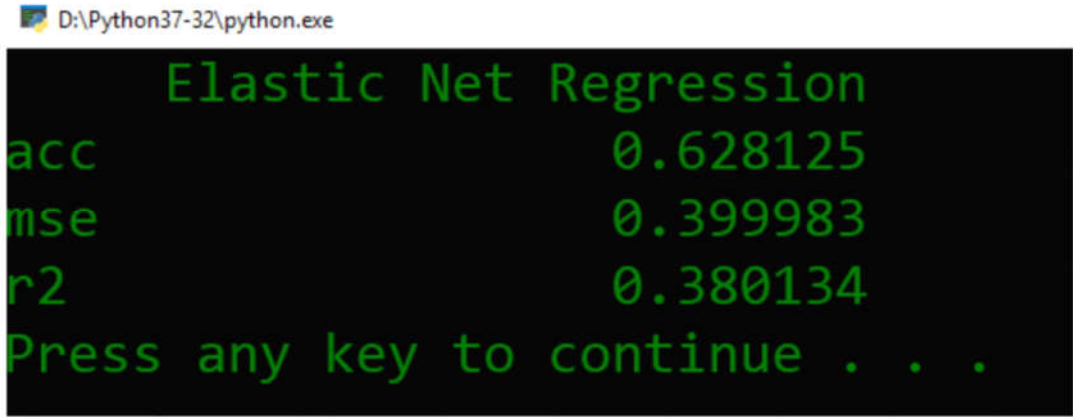
Bước 1: Dự đoán dữ liệu

Sử dụng trained model để dự đoán chính tập dữ liệu đã dùng để train cho Model, ta thu được **y_pred_train**

Sử dụng trained model để dự đoán cho tập dữ liệu X_test_scaled mà đã được ta tách ra từ data train ở phần xử lý data, thu được **y_pred_test**

Bước 2: Đánh giá Model

```
#kết quả
result = pd.DataFrame(list(zip(metrics)), columns = ['Elastic Net Regression'],
                      index = ['acc', 'mse', 'r2'])
```



```
D:\Python37-32\python.exe
Elastic Net Regression
acc                0.628125
mse                0.399983
r2                 0.380134
Press any key to continue . . .
```

3.5 So sánh với thuật toán khác

Để tiến hành so sánh giữa các giải thuật khác nhau cùng trên một dataset như ở mục 3.1, đầu tiên chúng ta sẽ xây dựng các model cho từng giải thuật khác nhau:

Linear Regression:

```
--Linear regression
#hàm này thực hiện train và trả về kết quả đánh giá
def linear_reg(X_train_scaled, X_test_scaled, y_train, y_test):
    # basic linear regression
    from sklearn.linear_model import LinearRegression
    lm = LinearRegression()
    lm.fit(X_train_scaled, y_train)
    y_pred_train = lm.predict(X_train_scaled)
    y_pred_test = lm.predict(X_test_scaled)
    #global metrics_lr lưu giữ ở đây cho lần so sánh kế tiếp
    global metrics_lr
    metrics_lr = [accuracy_score(y_test, np.round(y_pred_test)), mean_squared_error(y_test, y_pred_test), r:
    return scores_results(y_train, y_test, y_pred_train, y_pred_test)
```

Lasso Regression:

```
#Lasso Regression
def lasso_reg(X_train_scaled, X_test_scaled, y_train, y_test):
    from sklearn.linear_model import LassoCV
    n_alphas = 5000
    alpha_vals = np.logspace(-6, 0, n_alphas)
    lr = LassoCV(alphas=alpha_vals, cv=10, random_state=0)
    lr.fit(X_train_scaled, y_train)
    y_pred_train = lr.predict(X_train_scaled)
    y_pred_test = lr.predict(X_test_scaled)
    metrics_lasso = [accuracy_score(y_test, np.round(y_pred_test)), mean_squared_error(y_test, y_pred_test)
    return metrics_lasso
```

Ridge Regression:

```
def ridge_reg(X_train_scaled, X_test_scaled, y_train, y_test):
    from sklearn.linear_model import RidgeCV
    n_alphas = 100
    alpha_vals = np.logspace(-1, 3, n_alphas)
    rr = RidgeCV(alphas=alpha_vals, cv=10)
    rr.fit(X_train_scaled, y_train)
    y_pred_train = rr.predict(X_train_scaled)
    y_pred_test = rr.predict(X_test_scaled)
    metrics_ridge = [accuracy_score(y_test, np.round(y_pred_test)), mean_squared_error(y_test, y_pred_test)
    return metrics_ridge
```

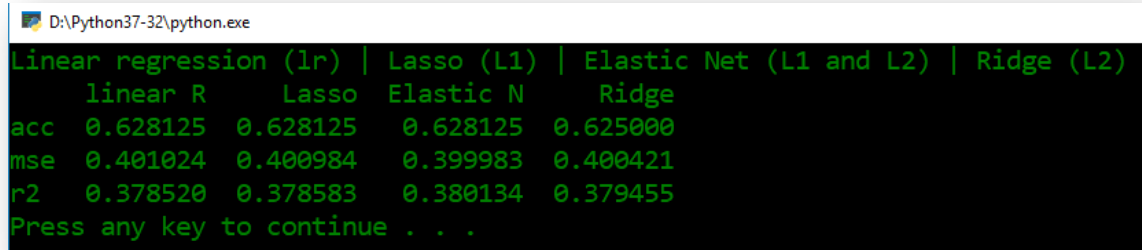
Elastic Net Regression (tương tự như lúc trước đã làm nhưng lần này đã được viết lại thành một function)

```
#Elastic Net Regression
def elastic_net_reg(X_train_scaled, X_test_scaled, y_train, y_test):
    from sklearn.linear_model import ElasticNetCV
    #n_alphas (int) số lượng số alphas trong quá trình regularization, được sử dụng cho mỗi l1_ratio
    n_alphas = 300
    #float between 0 and 1 passed to ElasticNet (scaling between l1 and l2 penalties)
    l1_ratio = [.1, .3, .5, .7, .9]
    #cv: chỉ định số lượng k-folds
    rr = ElasticNetCV(n_alphas=n_alphas, l1_ratio=l1_ratio, cv=10, random_state=0)
    rr.fit(X_train_scaled, y_train)
    y_pred_train = rr.predict(X_train_scaled)
    y_pred_test = rr.predict(X_test_scaled)
    metrics_en = [accuracy_score(y_test, np.round(y_pred_test)), mean_squared_error(y_test, y_pred_test)]
    return metrics_en
```

Chúng ta đều thấy các hàm đều trả về giá trị metrics cho chính giải thuật đó. Vì bây giờ chúng ta sẽ đưa dữ liệu ra nhờ sử dụng Dataframe trong thư viện Pandas

```
metrics_lasso = lasso_reg(X_train_scaled, X_test_scaled, y_train, y_test)
metrics_en = elastic_net_reg(X_train_scaled, X_test_scaled, y_train, y_test)
metrics_ridge = ridge_reg(X_train_scaled, X_test_scaled, y_train, y_test)
finalscores = pd.DataFrame(list(zip(metrics_lr, metrics_lasso, metrics_en, metrics_ridge)),
                             columns = ['lr', 'lasso', 'el net', 'ridge'], index = ['acc', 'mse', 'r2'])
print("Linear regression (lr) | Lasso (L1) | Ridge (L2) | Elastic Net (L1 and L2)")
print(finalscores)
```

Kết quả:



```
D:\Python37-32\python.exe
Linear regression (lr) | Lasso (L1) | Elastic Net (L1 and L2) | Ridge (L2)
      linear R      Lasso  Elastic N      Ridge
acc  0.628125  0.628125   0.628125  0.625000
mse  0.401024  0.400984   0.399983  0.400421
r2   0.378520  0.378583   0.380134  0.379455
Press any key to continue . . .
```

Nhận xét:

Theo như kết quả ta nhận được thì giữa các giải thuật chỉ có sự chênh lệch với nhau vô cùng nhỏ. Điều này có thể lí giải là model của chúng ta ban đầu đã không có hiện tượng overfitting. Rất có thể số lượng feature còn ít hoặc những giá trị đều tốt không có sự gây nhiễu. Dù vậy, chỉ có sự chênh lệch nhỏ nhưng chúng ta vẫn nhận ra hiệu suất, độ chính xác và tỉ lệ mất mát của Elastic Net vẫn tốt hơn hẳn so với 3 cái còn lại. Đó là nhờ vào hệ số penalty kết hợp của L1 và L2.

3.6 Thư viện lập trình

Tên hàm/thư viện	Mục đích
preprocessing	Thư viện chứa các hàm thực hiện kỹ thuật làm sạch dữ liệu từ dữ liệu gốc ví dụ như: minMaxScaler, binarizer
train_test_split	Hàm dùng để tách dữ liệu từ dataset phân ra thành dữ liệu x_train, x_test, y_train và y_test
accuracy_score	Hàm tính độ chính xác
StandardScaler	Là kỹ thuật giúp chuyển đổi dữ liệu gốc sang dữ liệu sạch sử dụng phân phối Gaussian
n_alphas	Số lượng alpha dùng cho mỗi l1_ratio
l1_ratio	Là 1 số nằm giữa [0;1] để cân bằng giữa hệ số penalti của L1 và L2. Nếu l1_ratio =0 thì nó chính là L2 penalty và nếu l1_ratio=1 thì chính là L1 penalty. $0 < l1_ratio < 1$ là hệ số penalty kết hợp giữa L1 và L2
ElasticNetCV	Chính là Elastic Net model với cross-validation estimator
cv	Xác định số lượng của k-folds
mse	Mean squared error: trung bình của bình phương của sai số
r2_score	Hiệu suất của mô hình

BẢNG PHÂN CÔNG

Họ tên và MSSV	Công việc	Đánh giá
Đào Xuân Thủy - 16110544	Tìm hiểu về Elastic net, phụ trách lí thuyết elastic net, code review và fix bug	50%
Lâm Phước Bảo- 16110016	Tìm hiểu về Elastic net, code thuật toán, trình bày quá trình cài đặt	50%

TÀI LIỆU THAM KHẢO

- [1] Duong Nguyen, *Linear Regression - Hồi quy tuyến tính trong Machine Learning*, <https://viblo.asia/p/linear-regression-hoi-quy-tuyen-tinh-trong-machine-learning-4P856akRIY3>, 30/05/2017.
- [2] Toan Pham, *Một vài hiểu nhầm khi mới học Machine Learning*, <https://viblo.asia/p/mot-vai-hieu-nham-khi-moi-hoc-machine-learning-4dbZNoDnIYM>, 27/02/2018.
- [3] Tiep Vu, *Overfitting*, <https://machinelearningcoban.com/2017/03/04/overfitting/>, 04/03/2017.