Cơ bản về học máy

Source : Chương 1,2
Gavin Hackeling, Mastering Machine Learning with scikit-learn
Presenter: A/Prof. Phuc Do
Year 2020

Học máy là gì ?

- Học máy là thiết kế và nghiên cứu cách tạo các phần mềm sử dụng kinh nghiệm trong quá khứ để đưa ra quyết định trong tương lai;
- Đó là nghiên cứu về các chương trình học từ dữ liệu.
- Mục tiêu cơ bản của học máy là để khái quát hóa, hoặc tạo quy tắc chưa biết về ứng dụng của quy tắc.
- Ví dụ điển hình của học máy là lọc thư rác. Bằng cách quan sát hàng ngàn email đã được gắn nhãn trước đó là thư rác hoặc ham, bộ lọc thư rác học cách phân loại thư mới.

Định nghĩa học máy

- Arthur Samuel, một nhà khoa học máy tính, người tiên phong nghiên cứu về trí tuệ nhân tạo, nói rằng học máy là "nghiên cứu mang lại cho máy tính khả năng học hỏi mà không được lập trình rõ ràng.»

Định nghĩa học máy

- Tom Mitchell định nghĩa học máy : "Một chương trình có học kinh nghiệm E đối với một số nhiệm vụ T với hiệu suất P. Hiệu suất thực hiện các nhiệm vụ trong T được đo bởi P nhằm cải thiện kinh nghiệm E.»
- **Ví dụ:** giả sử rằng bạn có một bộ sưu tập những bức ảnh. Mỗi bức tranh mô tả một con chó hoặc con mèo. Một nhiệm vụ có thể là sắp xếp các hình ảnh vào bộ sưu tập riêng biệt của hình ảnh chó và mèo. Một chương trình có thể học để thực hiện nhiệm vụ này bằng cách quan sát các hình ảnh đã được sắp xếp và nó có thể đánh giá hiệu suất bằng cách tính tỷ lệ phần trăm hình ảnh được phân loại chính xác.

Nhiệm vụ học máy

Phân loại

- Hai trong số các nhiệm vụ học máy giám sát phổ biến nhất là phân loại và hồi quy.
- Trong phân loại, chương trình phải học cách dự đoán rời rạc các giá trị cho các biến trả lời từ một hoặc nhiều biến giải thích. Chương trình phải dự đoán danh mục, lớp hoặc nhãn có thể xảy ra nhất cho quan sát mới.
- Các ứng dụng phân loại bao gồm dự đoán liệu một cổ phiếu giá sẽ tăng hoặc giảm, hoặc quyết định nếu một bài báo thuộc về thể loại chính trị hoặc giải trí

Hồi quy

- Trong các bài toán hồi quy, chương trình phải dự đoán giá trị của một liên tục biến phản ứng.
- Ví dụ về các vấn đề hồi quy bao gồm dự đoán doanh số cho một sản phẩm mới, hoặc mức lương cho một công việc dựa trên mô tả của nó.
- Tương tự như phân loại, vấn đề hồi quy đòi hỏi phải học có giám sát.

Training data and test data
Dữ liệu huấn luyện và kiểm tra

- Các quan sát trong tập huấn luyện bao gồm kinh nghiệm mà thuật toán sử dụng học.
- Trong học có giám sát, mỗi quan sát bao gồm một quan sát biến trả lời và một hoặc nhiều biến giải thích được quan sát.
- Bộ kiểm tra là một tập hợp các quan sát tương tự được sử dụng để đánh giá hiệu suất của mô hình bằng cách sử dụng một số số liệu hiệu suất.
- Điều quan trọng là không có quan sát nào từ tập huấn luyện nằm trong trong tập kiểm tra. Nếu tập kiểm tra có chứa ví dụ từ tập huấn luyện, sẽ rất khó để đánh giá thuật toán học
- Một chương trình khái quát hóa tốt sẽ có thể thực hiện hiệu quả nhiệm vụ với dữ liệu mới.

Các biện pháp thực hiện

Học giám sát và học không giám sát

- Các hệ học máy thường được mô tả là học từ kinh nghiệm có hoặc không có sự giám sát từ con người.
- Trong học có giám sát, một chương trình dự đoán đầu ra cho đầu vào bằng cách học hỏi từ các cặp đầu vào và đầu ra có nhãn (tập học);
- Trong học không giám sát, một chương trình không học từ tập học thay vào đó, nó cố gắng khám phá các mẫu trong dữ liệu.

Giới thiệu về scikit-learn

Giới thiệu về scikit-learn

- Kể từ khi phát hành vào năm 2007, scikit-learn đã trở thành một trong những phần mềm mở phổ biến nhất
- Thư viện máy học cho Python.
- scikit-learn cung cấp các thuật toán cho nhiệm vụ học máy bao gồm phân loại, hồi quy, tút gọn chiều, và phân cụm.

Hồi quy tuyến tính
Linear Regression

Mục tiêu

- Giả sử bạn muốn biết giá của một chiếc bánh pizza.
- Bạn có thể chỉ cần nhìn vào một menu và sử dụng hồi quy tuyến tính đơn giản dự đoán giá của một chiếc bánh pizza dựa trên một thuộc tính của chiếc bánh pizza mà chúng ta có thể quan sát

Dữ liệu học

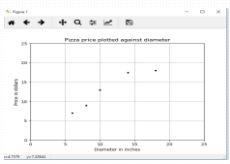
Training instance	Diameter (in inches)	Price (in dollars)
1	6	7
2	8	9
3	10	13
4	14	17.5
5	18	18

Bạn đã ghi lại đường kính và giá của pizza mà bạn đã ăn pizza trước đây. Đường kính (tính bằng inch); Giá (tính bằng đô la)

Trực quan hóa dữ liệu học bằng cách vẽ biểu đồ bằng matplotlib

```
import matplotlib.pyplot as plt
X = [[6], [8], [10], [14], [18]]
y = [[7], [9], [13], [17.5], [18]]
plt.figure()
plt.title('Pizza price plotted against diameter')
plt.xlabel('Diameter in inches')
plt.ylabel('Price in dollars')
plt.plot(X, y, 'k.')
plt.axis([0, 25, 0, 25])
plt.grid(True)
plt.show()
```


Trực quan hóa dữ liệu học bằng cách vẽ biểu đồ bằng matplotlib



Chúng ta có thể thấy từ biểu đồ của dữ liệu học rằng có một mối quan hệ tích cực giữa đường kính của một chiếc bánh pizza và giá của nó, cần được chứng thực bằng kinh nghiệm ăn pizza của chính chúng ta.
Khi đường kính của một chiếc bánh pizza tăng lên, giá của nó thường cũng tăng theo.

Chương trình dự đoán giá pizza sử dụng hồi quy tuyến tính.

```
from sklearn.linear_model import LinearRegression
# Tập học
X = [[6], [8], [10], [14], [18]]
y = [[7], [9], [13], [17.5], [18]]
# Tạo model hồi quy tuyến tính
model = LinearRegression()
model.fit(X, y)
print ('A 12" pizza should cost: $%.2f' %
model.predict([[12]]))
Kết quả #### A 12" pizza should cost: $13.68
```


Ước tính

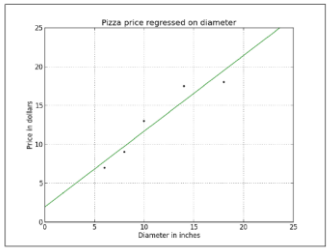
- Lớp sklearn.linear_model.LinearRegression là một công cụ ước tính.
- Ước tính dự đoán một giá trị dựa trên dữ liệu quan sát.
- Trong scikit-learn, tất cả các công cụ ước tính đều thực hiện các phương thức fit () và predict().
- Phương pháp fit() được dùng để học các tham số của mô hình
- Phương thức predict() được dùng để dự đoán giá trị của biến trả lời cho biến giải thích bằng các tham số đã học.

- Phương thức phù hợp của linearRegression tìm hiểu các tham số của mô hình sau để hồi quy tuyến tính đơn giản:

$$y = \alpha + \beta x$$

- y là giá của pizza.
- x là biến giải thích.
- Thuật ngữ intercept term α và hệ số β là các tham số của mô hình được học bằng thuật toán học.

- Dòng vẽ trong hình dưới đây mô tả mối quan hệ giữa kích thước của pizza và giá của nó.
- Sử dụng mô hình này, chúng tôi dự kiến giá của một chiếc bánh pizza 8 inch sẽ vào khoảng 7,33 đô la và giá của một chiếc bánh pizza 20 inch là 18,75 đô la.



Hồi quy tuyến tính đa biến Multiple Linear Regression

Hồi quy tuyến tính đa biến

- Giá cả thường phụ thuộc vào số lượng toppings trên pizza.
- Hãy thêm số lượng toppings vào dữ liệu học như một biến giải thích thứ hai.
- Có nhiều biến giải thích được gọi là hồi quy tuyến tính đa biến.

- Mô hình hồi quy đa biến:

$$y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

  - Hồi quy đa biến sử dụng một hệ số cho mỗi số lượng các biến giải thích tùy ý.

$$Y = X\beta$$

$$\begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} \alpha + \beta X_1 \\ \alpha + \beta X_2 \\ \vdots \\ \alpha + \beta X_n \end{bmatrix} = \begin{bmatrix} 1 & X_1 \\ 1 & X_2 \\ \vdots & \vdots \\ 1 & X_n \end{bmatrix} \times \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

Y là một vectơ cột giá trị của các biến kết quả cho các mẫu học
β là một vectơ cột của các giá trị của các tham số của mô hình.
X, là ma trận thứ nguyên m × n giá trị của các biến giải thích của tập mẫu học
m là số mẫu học và n là số biến giải thích.

Tập học và tập kiểm tra

Tập học

Training Example	Diameter (in inches)	Number of toppings	Price (in dollars)
1	6	2	7
2	8	1	9
3	10	0	13
4	14	2	17.5
5	18	0	18

Tập kiểm tra

Test Instance	Diameter (in inches)	Number of toppings	Price (in dollars)
1	8	2	11
2	9	0	8.5
3	11	2	15
4	16	2	18
5	12	0	11

Chương trình hồi quy tuyến tính đa biến

```
from sklearn.linear_model import LinearRegression
X = [[6, 2], [8, 1], [10, 0], [14, 2], [18, 0]]
y = [[7], [9], [13], [17.5], [18]]
model = LinearRegression()
model.fit(X, y)
X_test = [[8, 2], [9, 0], [11, 2], [16, 2], [12, 0]]
y_test = [[11], [8.5], [15], [18], [11]]
predictions = model.predict(X_test)
for i, prediction in enumerate(predictions):
    print ('Predicted: %s, Target: %s' % (prediction, y_test[i]))
    print ('R-squared: %.2f' % model.score(X_test, y_test))
```

Kết quả

- Predicted: [10.0625], Target: [11]
- Predicted: [10.28125], Target: [8.5]
- Predicted: [13.09375], Target: [15]
- Predicted: [18.14583333], Target: [18]
- Predicted: [13.3125], Target: [11]
- R-squared: 0.77

Hồi quy đa thức
Polynomial regression

Hồi quy đa thức

- Hồi quy đa thức, một trường hợp đặc biệt của hồi quy đa biến tuyến tính bổ sung các số hạng có độ lớn hơn một cho mô hình
- Hồi quy bậc hai, hoặc hồi quy với đa thức bậc hai, được
- xác định bởi công thức sau:

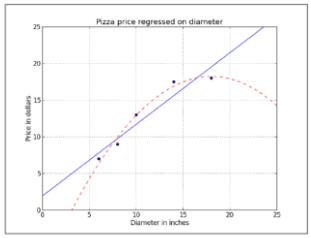
$$y = \alpha + \beta_1 x + \beta_2 x^2$$

Chương trình thực hiện

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
X_train = [[6], [8], [10], [14], [18]]
y_train = [[7], [9], [13], [17.5], [18]]
X_test = [[6], [8], [11], [16]]
y_test = [[8], [12], [15], [18]]
regressor = LinearRegression()
regressor.fit(X_train, y_train)
xx = np.linspace(0, 26, 100)
yy = regressor.predict(xx.reshape(xx.shape[0], 1))
plt.plot(xx, yy)
quadratic_featurizer = PolynomialFeatures(degree=2)
X_train_quadratic = quadratic_featurizer.fit_transform(X_train)
X_test_quadratic = quadratic_featurizer.transform(X_test)
regressor_quadratic = LinearRegression()
regressor_quadratic.fit(X_train_quadratic, y_train)
xx_quadratic = quadratic_featurizer.transform(xx.reshape(xx.shape[0], 1))
plt.plot(xx, regressor_quadratic.predict(xx_quadratic), c='r',linestyle='--')
plt.title('Pizza price regressed on diameter')
plt.xlabel('Diameter in inches')
plt.ylabel('Price in dollars')
plt.axis([0, 25, 0, 25])
plt.grid(True)
plt.scatter(X_train, y_train)
plt.show()
print (X_train)
print (X_train_quadratic)
print (X_test)
print (X_test_quadratic)
print ('Simple linear regression r-squared', regressor.score(X_test, y_test))
print ('Quadratic regression r-squared', regressor_quadratic.score(X_test_quadratic, y_test))
```



```
plt.title('Pizza price regressed on diameter')
plt.xlabel('Diameter in inches')
plt.ylabel('Price in dollars')
plt.axis([0, 25, 0, 25])
plt.grid(True)
plt.scatter(X_train, y_train)
plt.show()
print (X_train)
print (X_train_quadratic)
print (X_test)
print (X_test_quadratic)
print ('Simple linear regression r-squared', regressor.score(X_test, y_test)
print 'Quadratic regression r-squared', regressor_quadratic.score(X_test_quadratic, y_test))
```

Áp dụng hồi quy tuyến tính

Áp dụng hồi quy tuyến tính

- Chúng tôi sẽ sử dụng máy học để dự đoán chất lượng rượu dựa trên thuộc tính hóa lý của nó.

Exploring the data

Fixed acidity	Volatile acidity	Citric acidity	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

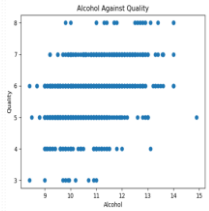
Dữ liệu học

```
>>> import pandas as pd
>>> df = pd.read_csv('winequality-red.csv', sep=';')
>>> df.describe()
```

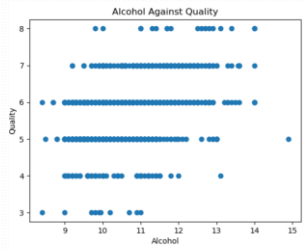
	pH	sulphates	alcohol	quality
count	1599.000000	1599.000000	1599.000000	1599.000000
mean	3.311113	0.658149	10.422983	5.636023
std	0.154386	0.169507	1.065668	0.807569
min	2.740000	0.330000	8.400000	3.000000
25%	3.210000	0.550000	9.500000	5.000000
50%	3.310000	0.620000	10.200000	6.000000
75%	3.400000	0.730000	11.100000	6.000000
max	4.010000	2.000000	14.900000	8.000000

Vẽ biểu đồ

```
import matplotlib.pylab as plt
plt.scatter(df['alcohol'], df['quality'])
plt.xlabel('Alcohol')
plt.ylabel('Quality')
plt.title('Alcohol Against Quality')
plt.show()
```



Kết quả



Chương trình

```
### Wine Regression ###
###Date:5/07/2020
import pandas as pd
df = pd.read_csv('data/winequality-red.csv', sep=';')
df.describe()
### Vẽ đồ thị
import matplotlib.pylab as plt
plt.scatter(df['alcohol'], df['quality'])
plt.xlabel('Alcohol')
plt.ylabel('Quality')
plt.title('Alcohol Against Quality')
 plt.show()
### Tính hồi qui
from sklearn.linear_model import LinearRegression
import pandas as pd
import matplotlib.pylab as plt
 from sklearn.model_selection import train_test_split
df = pd.read_csv('data/winequality-red.csv', sep=';')
X = df[list(df.columns)[:-1]]
y = df['quality']
X_train, X_test, y_train, y_test = train_test_split(X, y)
regressor = LinearRegression()
regressor.fit(X_train, y_train)
y_predictions = regressor.predict(X_test)
print ('R-squared:', regressor.score(X_test, y_test))
```


Kết quả dự đoán

- the predicted scores:
- Predicted: 4.89907499467 True: 4
- Predicted: 5.60701048317 True: 6
- Predicted: 5.92154439575 True: 6
- Predicted: 5.54405696963 True: 5
- Predicted: 6.07869910663 True: 7
- Predicted: 6.036656327 True: 6
- Predicted: 6.43923020473 True: 7
- Predicted: 5.80270760407 True: 6
- Predicted: 5.92425033278 True: 5
- Predicted: 5.31809822449 True: 6
- Predicted: 6.34837585295 True: 6

Biểu diễn trực quan kết quả

Tóm lược

- Trong chương này, chúng tôi đã thảo luận về ba trường hợp hồi quy tuyến tính.
- Chúng tôi đã làm việc thông qua một ví dụ về hồi quy tuyến tính đơn giản, mô hình hóa mối quan hệ giữa một một biến giải thích và một biến trả lời sử dụng một dòng.
- Sau đó chúng tôi đã thảo luận hồi quy tuyến tính đa biến, trong đó khái quát hóa hồi quy tuyến tính đơn giản để mô hình hóamối quan hệ giữa nhiều biến giải thích và một biến trả lời.
- Cuối cùng,chúng tôi đã mô tả hồi quy đa thức, một trường hợp đặc biệt của mô hình hồi quy tuyến tính đa biến mô hình mối quan hệ phi tuyến tính giữa các biến giải thích và phản hồi.

Bài tập buổi 3

- Sưu tầm các dữ liệu tài chính
- Thiết lập ba mô hình dự báo đã học trên dữ liệu tài chính