การแบ่งข้อมูลเพื่อสอนและทดสอบด้วย Pandas

เขียนโดย ดร.จักรกฤษณ์ แสงแก้ว วันที่ 24 กรกฎาคม 2562

บทนำ

ในการสร้างโมเดลเพื่อพยากรณ์ จะทำการรวบรวมข้อมูลและนำข้อมูลเหล่านั้นมาแบ่งออกเป็น 2 ส่วน คือ
1) ข้อมูลที่ใช้สำหรับการสอน (Train) หมายถึง การนำข้อมูลไปสร้างสมการเพื่ออธิบายรูปแบบข้อมูลนั้น ๆ เรียกสมการที่สร้างขึ้นว่า โมเดลสำหรับแทนข้อมูลกลุ่มนี้
2) ข้อมูลที่ใช้สำหรับการทดสอบ (Test) หมายถึง การนำข้อมูลเพื่อป้อนให้กับสมการหรือโมเดลทางคณิตศาสตร์ เพื่อคำนวณหาประสิทธิภาพของการพยากรณ์

ในหัวข้อนี้เป็นการศึกษาใช้งานไลบรารี่ Pandas ซึ่งทำงานร่วมกับภาษาไพธอนและได้รับความนิยมในการจัดการข้อมูล อาทิ การเพิ่ม ลบ แก้ไข กรอง และแบ่งข้อมูลสำหรับการ Train และ Test ได้อย่างสะดวก รวดเร็ว และมีประสิทธิภาพ

การคืนค่า 2 ตัวแปรจากฟังก์ชั่น train_test_split()

ฟังก์ชั่น train_test_split() แบบคืนค่า 2 ตัวแปร จะมีหลักการแบ่งข้อมูลจากจำนวนแถวทั้งหมดออกเป็น 2 กลุ่ม ในข้อมูลชุดนี้จะประกอบด้วยทั้งตัวแปรอิสระและตัวแปรตามรวมด้วยกัน พิจารณาตัวอย่างต่อไปนี้

```
%pylab inline
import pandas as pd
import seaborn as sns
df = pd.read_csv("https://raw.githubusercontent.com/dsdi/dataset/master/th-advertising.csv", usecols=[1,2,3,4])
from sklearn.model_selection import train_test_split
train,test = train_test_split(df, train_size=0.7,random_state=7)
print(len(train), len(test))
train.head()
test.head()
```

คำอธิบาย :
บรรทัด 1 : ประกาศขอใช้ไลบรารี่ matplotlib และ numpy ด้วยคำสั่งลัด %pylab inline
บรรทัด 2-3 : ประกาศขอใช้ไลบรารี่ pandas และ seaborn สำหรับจัดการข้อมูลและแสดงผลข้อมูลแบบกราฟิก
บรรทัด 4 : โหลดข้อมูลจากเว็บเข้ามายังดาต้าเฟรมโดยใช้งานคอลัมน์ที่ 1,2,3 และ 4 ได้แก่ โทรทัศน์ วิทยุ หนังสือพิมพ์ ยอดขาย ตามลำดับ
บรรทัด 5 : ขอใช้คำสั่ง train_test_split จากไลบรารี่ scikit-learn
บรรทัด 6 : แบ่งข้อมูลสำหรับสอน (train) ออกเป็น 70% และสำหรับการทดสอบ (test) ออกเป็น 30% และกำหนด random_state=7 คือสุ่มทีละ 7 ตัว โดยสามารถเปลี่ยนแปลงค่า random_state ตามต้องการ
บรรทัด 7 : แสดงจำนวนข้อมูลภายในตัวแปร train และ test ด้วยคำสั่ง len() โดย 70% ของข้อมูลทั้งหมด 200 เรคคอร์ด มีค่าเท่ากับ 140 ดังนั้น ข้อมูลที่ใช้สอนมีจำนวน 140 แถวและข้อมูลที่ใช้ทดสอบมี 60 แถว ตามลำดับ
บรรทัด 8-9 : แสดงข้อมูลด้วยคำสั่ง head() เมื่อเลข 8 ที่ป้อนให้กับฟังก์ชั่น head() หมายถึงจำนวนแถวที่ต้องการนำมาแสดงผลลัพธ์

	โทรทัศน์	วิทยุ	หนังสือพิมพ์	ยอดขาย
88	88.3	25.5	73.4	12.9
58	210.8	49.6	37.7	23.8
113	209.6	20.6	10.7	15.9
149	44.7	25.8	20.6	10.1
36	266.9	43.8	5.0	25.4
192	17.2	4.1	31.6	5.9
182	56.2	5.7	29.7	8.7
51	100.4	9.6	3.6	10.7

	โทรทัศน์	วิทยุ	หนังสือพิมพ์	ยอดขาย
86	76.3	27.5	16.0	12.0
120	141.3	26.8	46.2	15.5
22	13.2	15.9	49.6	5.6
11	214.7	24.0	4.0	17.4
195	38.2	3.7	13.8	7.6
2	17.2	45.9	69.3	9.3
121	18.8	21.7	50.4	7.0
94	107.4	14.0	10.9	11.5

การคืนค่า 4 ตัวแปรจากฟังก์ชั่น train_test_split()

กรณีใช้งานฟังก์ชั่น train_test_split() แบบคืนค่า 4 ตัวแปร ประกอบด้วย train_x , train_y และ test_x, test_y วิธีการนี้ ข้อมูล train_x จะประกอบด้วยตัวแปรอิสระที่ถูกแบ่งออกมาใช้สำหรับการสอน ส่วน train_y คือตัวแปรตามที่ถูกแบ่งออกมาใช้สำหรับการสอน
ในทำนองเดียวกัน ตัวแปร test_x คือตัวแปรอิสระ และ test_y คือตัวแปรตาม ที่ถูกแบ่งออกมาสำหรับการทดสอบโมเดลที่สร้างขึ้น พิจารณาตัวอย่างต่อไปนี้

```
%pylab inline
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
df = pd.read_csv("https://raw.githubusercontent.com/dsdi/dataset/master/th-advertising.csv", usecols=[1,2,3,4])
X = df[['โทรทัศน์','วิทยุ','หนังสือพิมพ์']]
y = df[['ยอดขาย']]
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=7)
X_train.head(5)
X_test.head(5)
y_train.head(5)
y_train.head(5)
```

อธิบาย :
บรรทัด 1 : ประกาศขอใช้ไลบรารี่ matplotlib และ numpy ด้วยคำสั่งลัด %pylab inline
บรรทัด 2-3 : ประกาศขอใช้ไลบรารี่ pandas และ seaborn สำหรับจัดการข้อมูลและแสดงผลข้อมูลแบบกราฟิก
บรรทัด 4 : ขอใช้คำสั่ง train_test_split จากไลบรารี่ scikit-learn
บรรทัด 5 : โหลดข้อมูลจากเว็บเข้ามายังดาต้าเฟรมโดยใช้งานคอลัมน์ที่ 1,2,3 และ 4 ได้แก่ โทรทัศน์ วิทยุ หนังสือพิมพ์ ยอดขาย ตามลำดับ
บรรทัด 6 : สร้างตัวแปรอิสระ X ได้แก่ โทรทัศน์ วิทยุ หนังสือพิมพ์
บรรทัด 7 : สร้างตัวแปรตาม y ได้แก่ ยอดขาย
บรรทัด 8 : สร้างตัวแปร X_train, X_test , y_train และ y_test ด้วยฟังก์ชั่น train_test_split() กำหนด test_size 30% (0.3) และกำหนด random_state ถ้าไม่กำหนดจะสุ่มข้อมูลใหม่ทุกครั้งที่รันใหม่
บรรทัด 9-12 : แสดงข้อมูลของ X_train, X_test , y_train และ y_test ออกมาอย่างละ 5 แถว

	โทรทัศน์	วิทยุ	หนังสือพิมพ์
88	88.3	25.5	73.4
58	210.8	49.6	37.7
113	209.6	20.6	10.7
149	44.7	25.8	20.6
36	266.9	43.8	5.0

	โทรทัศน์	วิทยุ	หนังสือพิมพ์
86	76.3	27.5	16.0
120	141.3	26.8	46.2
22	13.2	15.9	49.6
11	214.7	24.0	4.0
195	38.2	3.7	13.8

	ยอดขาย
88	12.9
58	23.8
113	15.9
149	10.1
36	25.4

	ยอดขาย
86	12.0
120	15.5
22	5.6
11	17.4
195	7.6

การสร้างโมเดลด้วย Multiple Linear Regression จากข้อมูล 4 ตัวแปร (X_train,y_train) และ (X_test,y_test)

เมื่อทำการแบ่งข้อมูลออกเป็นสองส่วน คือ 1) Train 2) Test ด้วยฟังก์ชั่น train_test_split() แล้ว ต่อไปเป็นการนำเอาข้อมูลที่แบ่งออกมาแล้วไปใช้สร้างโมเดลและทดสอบโมเดล ขอให้พิจารณาตัวอย่างต่อไปนี้

```
%pylab inline
import pandas as pd
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split

df = pd.read_csv("https://raw.githubusercontent.com/dsdi/dataset/master/th-advertising.csv", usecols=[1,2,3,4])
X = df[['โทรทัศน์','วิทยุ','หนังสือพิมพ์']]
y = df[['ยอดขาย']]
X_train,X_test,y_train,y_test = train_test_split(X,y,train_size=0.7,random_state=7)

model = LinearRegression()
model.fit(X,y)
print("R-squared : ",model.score(X,y))
print("สัมประสิทธิ์ : ",model.coef_)
print("พยากรณ์ : ",model.predict([[200,40,70]]))

y2 = model.predict(X_train).ravel()
train = pd.concat([X_train, y_train], axis='columns')
dc=pd.concat([train.reset_index(), pd.Series(y2, name='พยากรณ์')], axis='columns')
dc.corr()
```

ผลลัพธ์ :

```
Populating the interactive namespace from numpy and matplotlib
R-squared :  0.8972106381789521
สัมประสิทธิ์ :  [[ 0.04576465  0.18853002 -0.00103749]]
พยากรณ์ :  [[19.56039462]]
```

	index	โทรทัศน์	วิทยุ	หนังสือพิมพ์	ยอดขาย	พยากรณ์
index	1.000000	-0.056881	-0.111713	-0.214504	-0.100151	-0.110548
โทรทัศน์	-0.056881	1.000000	0.012879	0.038569	0.768956	0.808029
วิทยุ	-0.111713	0.012879	1.000000	0.342437	0.562808	0.599485
หนังสือพิมพ์	-0.214504	0.038569	0.342437	1.000000	0.218680	0.228583
ยอดขาย	-0.100151	0.768956	0.562808	0.218680	1.000000	0.947097
พยากรณ์	-0.110548	0.808029	0.599485	0.228583	0.947097	1.000000

อธิบาย :
บรรทัด 1 : ประกาศขอใช้ไลบรารี่ matplotlib และ numpy ด้วยคำสั่งลัด %pylab inline
บรรทัด 2-3 : ประกาศขอใช้ไลบรารี่ pandas และ seaborn สำหรับจัดการข้อมูลและแสดงผลข้อมูลแบบกราฟิก
บรรทัด 4 : ขอใช้คลาส LinearRegression จากไลบรารี่ sklearn
บรรทัด 5 : ขอใช้คำสั่ง train_test_split จากไลบรารี่ scikit-learn
บรรทัด 7 : โหลดข้อมูลจากเว็บเข้ามายังดาต้าเฟรมโดยใช้งานคอลัมน์ที่ 1,2,3 และ 4 ได้แก่ โทรทัศน์ วิทยุ หนังสือพิมพ์ ยอดขาย ตามลำดับ
บรรทัด 8 : สร้างตัวแปรอิสระ X ได้แก่ โทรทัศน์ วิทยุ หนังสือพิมพ์
บรรทัด 9 : สร้างตัวแปรตาม y ได้แก่ ยอดขาย
บรรทัด 10 : สร้างตัวแปร X_train, X_test , y_train และ y_test ด้วยฟังก์ชั่น train_test_split() กำหนด test_size 30% (0.3) และกำหนด random_state ถ้าไม่กำหนดจะสุ่มข้อมูลใหม่ทุกครั้งที่รันใหม่
บรรทัด 12 : สร้างอ็อบเจ็ค model จากคลาส LinearRegression()
บรรทัด 13 : สร้างโมเดลจากข้อมูล X และ y ด้วยคำสั่ง fit()
บรรทัด 14-16 : พิมพ์ค่า R-squared , สัมประสิทธิ์ และค่าการพยากรณ์ที่ได้จากโมเดล เมื่อป้อนข้อมูล โทรทัศน์=200 วิทยุ=40 และหนังสือพิมพ์ = 70
บรรทัด 18 : สร้างตัวแปร y2 สำหรับเก็บผลการพยากรณ์ที่ได้จากโมเดลและแปลงให้อยู่ในรูปอาร์เรย์ 1 มิติ โดยใช้ฟังก์ชั่น ravel()
บรรทัด 19 : สร้างตัวแปร train โดยนำข้อมูลจาก X_train สามคอลัมน์ (โทรทัศน์,วิทยุ,หนังสือพิมพ์ และ y_train (ยอดขาย) มาต่อกัน รวมเป็น 4 คอลัมน์ และเก็บไว้ในตัวแปร train
บรรทัด 20 : สร้างตัวแปร dc โดยรวมคอลัมน์ y2 ซึ่งเป็นผลการพยากรณ์ ต่อเพิ่มเป็นอีก 1 คอลัมน์ด้านขวา
บรรทัด 21 : พิมพ์ค่า สหสัมพันธ์ (Correlation) ด้วยฟังก์ชั่น corr()

การสร้างโมเดลด้วย Multiple Linear Regression จากข้อมูล 2 ตัวแปร train และ test

ไลบรารี่ statsmodels เหมาะกับข้อมูลที่แบ่งเป็น 2 ส่วน คือ train และ test พิจารณาตัวอย่างต่อไปนี้

```
import pandas as pd
from sklearn.model_selection import train_test_split
import statsmodels.formula.api as smf

df = pd.read_csv("https://raw.githubusercontent.com/dsdi/dataset/master/en-advertising.csv", usecols=[1,2,3,4])
train,test = train_test_split(df, train_size=0.7,random_state=7)
model = smf.ols(formula="Sales ~ TV + Radio + Newspaper",data=train).fit()
model.summary()
```

ผลลัพธ์ :

OLS Regression Results

Dep. Variable:	Sales	**R-squared:**	0.897
Model:	OLS	**Adj. R-squared:**	0.895
Method:	Least Squares	**F-statistic:**	395.0
Date:	Wed, 24 Jul 2019	**Prob (F-statistic):**	6.42e-67
Time:	05:12:09	**Log-Likelihood:**	-271.78
No. Observations:	140	**AIC:**	551.6
Df Residuals:	136	**BIC:**	563.3
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	2.5972	0.398	6.533	0.000	1.811	3.383
TV	0.0471	0.002	27.669	0.000	0.044	0.050
Radio	0.1910	0.010	18.885	0.000	0.171	0.211
Newspaper	-1.938e-05	0.007	-0.003	0.998	-0.014	0.014

Omnibus:	49.785	**Durbin-Watson:**	2.062
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	135.914
Skew:	-1.399	**Prob(JB):**	3.07e-30
Kurtosis:	6.933	**Cond. No.**	487.

อธิบาย :
บรรทัด 1 : ขอใช้ไลบรารี่ pandas สำหรับจัดการข้อมูล เพิ่ม ลบ แก้ไข และตั้งชื่อย่อว่า pd
บรรทัด 2 : ขอใช้งานคลาส train_test_split() ภายในไลบรารี่ scikit-learn
บรรทัด 3 : ขอใช้งานคลาส smf ภายในไลบรารี่ statsmodels
บรรทัด 4 : สร้างตัวแปร df โหลดข้อมูลจากไฟล์ที่กำหนด โดยเลือกเอาเฉพาะคอลัมน์ 1,2,3,4 ได้แก่ โทรทัศน์ วิทยุ หนังสือพิมพ์ และ ยอดขาย
บรรทัด 5 : สร้างตัวแปร train,test ด้วยการแบ่งข้อมูลจากดาต้าเฟรม ด้วยขนาดข้อมูล train 70% และ test 30% ตัวแปร random_state ถ้าไม่กำหนดจะสุ่มข้อมูล train/test ใหม่ทุกครั้ง
บรรทัด 6 : สร้างโมเดลด้วยค่ำสั่ง ols (ordinal least square) โดยใช้ตัวแปรพยากรณ์ Sales จากตัวแปรอิสระ Tv + Radio + Newspaper
บรรทัด 7 : แสดงค่าสถิติของโมเดล ได้แก่ R-Squared, Adjusted R-Squared , p-values เป็นต้น

แหล่งอ้างอิง

- หลักการทำงานเบื้องต้นของ Logistic Regression -> https://www.youtube.com/watch?v=zhkTD7rNEBk
- การทำ Logistic Regression ด้วย statsmodels -> https://www.youtube.com/watch?v=SM1W2SQOD7I
- การทำ Logistic Regression (binary classification) ด้วย scikit-learn -> https://www.youtube.com/watch?v=l1OWNtuAUUg
- สอน R: การทำ Logistic Regression เบื้องต้น -> https://www.youtube.com/watch?v=nFJgel5Cv0E
- สอน Azure Machine Learning: Binary Logistic Regression -> https://www.youtube.com/watch?v=xetCfuHr4DY