การบ้าน ML

Exploratory data

ทำการ Exploratory data ของ UCI Wine Dataset เพื่อหาข้อมูลเชิงลึกในการนำมาวิเคราะห์โดยข้อมูลมีชุดตัวอย่างดังรูปต่อไปนี้

	Wine	Alcohol	Malic.acid	Ash	Acl	Mg	Phenols	Flavanoids	Nonflavanoid.phenols	Proanth	Color.int	Hue	OD	Proline
2	1	14.23	1.71	2.43	15.6	127	2.8	3.06	.28	2.29	5.64	1.04	3.92	1065
3	1	13.2	1.78	2.14	11.2	100	2.65	2.76	.26	1.28	4.38	1.05	3.4	1050
4	1	13.16	2.36	2.67	18.6	101	2.8	3.24	.3	2.81	5.68	1.03	3.17	1185
5	1	14.37	1.95	2.5	16.8	113	3.85	3.49	.24	2.18	7.8	.86	3.45	1480
6	1	13.24	2.59	2.87	21	118	2.8	2.69	.39	1.82	4.32	1.04	2.93	735
7	1	14.2	1.76	2.45	15.2	112	3.27	3.39	.34	1.97	6.75	1.05	2.85	1450
8	1	14.39	1.87	2.45	14.6	96	2.5	2.52	.3	1.98	5.25	1.02	3.58	1290
9	1	14.06	2.15	2.61	17.6	121	2.6	2.51	.31	1.25	5.05	1.06	3.58	1295
10	1	14.83	1.64	2.17	14	97	2.8	2.98	.29	1.98	5.2	1.08	2.85	1045

รูปตัวอย่างชุดข้อมูล UCI Wine Dataset

โดยข้อมูลมี sample ทั้งหมด 178 ตัวอย่าง มีตัวแปร(Feature) ทั้งหมด 14 ตัวแปร โดยแบ่งเป็นชนิดของตัวแปรออกเป็นข้อมูลชนิด integer 3 ตัวแปร และ float 11 ตัวแปร ดังรูปต่อไปนี้

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 178 entries, 0 to 177
Data columns (total 14 columns):
Wine                  178 non-null int64
Alcohol               178 non-null float64
Malic.acid            178 non-null float64
Ash                   178 non-null float64
Acl                   178 non-null float64
Mg                    178 non-null int64
Phenols               178 non-null float64
Flavanoids            178 non-null float64
Nonflavanoid.phenols  178 non-null float64
Proanth               178 non-null float64
Color.int             178 non-null float64
Hue                   178 non-null float64
OD                    178 non-null float64
Proline               178 non-null int64
dtypes: float64(11), int64(3)
memory usage: 19.5 KB
```

รูปชนิดของ Feature

ข้อมูลดังกล่าว มีค่าสถิติพื้นฐาน ดังนี้

	Wine	Alcohol	Malic.acid	Ash	Acl	Mg	Phenols
count	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000
mean	1.938202	13.000618	2.336348	2.366517	19.494944	99.741573	2.295112
std	0.775035	0.811827	1.117146	0.274344	3.339564	14.282484	0.625851
min	1.000000	11.030000	0.740000	1.360000	10.600000	70.000000	0.980000
25%	1.000000	12.362500	1.602500	2.210000	17.200000	88.000000	1.742500
50%	2.000000	13.050000	1.865000	2.360000	19.500000	98.000000	2.355000
75%	3.000000	13.677500	3.082500	2.557500	21.500000	107.000000	2.800000
max	3.000000	14.830000	5.800000	3.230000	30.000000	162.000000	3.880000

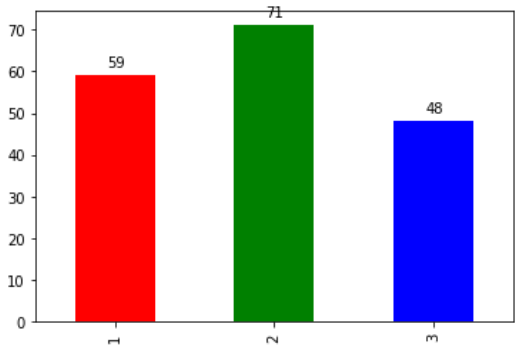
รูปสถิติพื้นฐานของ Feature ที่ 1-7

	Flavanoids	Nonflavanoid.phenols	Proanth	Color.int	Hue	OD	Proline
count	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000
mean	2.029270	0.361854	1.590899	5.058090	0.957449	2.611685	746.893258
std	0.998859	0.124453	0.572359	2.318286	0.228572	0.709990	314.907474
min	0.340000	0.130000	0.410000	1.280000	0.480000	1.270000	278.000000
25%	1.205000	0.270000	1.250000	3.220000	0.782500	1.937500	500.500000
50%	2.135000	0.340000	1.555000	4.690000	0.965000	2.780000	673.500000
75%	2.875000	0.437500	1.950000	6.200000	1.120000	3.170000	985.000000
max	5.080000	0.660000	3.580000	13.000000	1.710000	4.000000	1680.000000

รูปสถิติพื้นฐานของ Feature ที่ 8-14

Class หรือ Label หรือ Target ที่เราจะจำแนกมี 3 ค่าคือ

- ค่า 1 จำนวน 59 ตัวอย่าง
- ค่า 2 จำนวน 71 ตัวอย่าง
- ค่า 3 จำนวน 48 ตัวอย่าง



Decision Tree

ทำการสร้าง Model บน Google Colaboratory ด้วยภาษา Python โดยมีขั้นตอนการสร้าง ดังต่อไปนี้

ขั้นตอนที่ 1 : Import libraly ที่เกี่ยวข้อง

```
import numpy as np
import pandas as pd
from sklearn.tree import DecisionTreeClassifier, export_graphviz
from sklearn import tree
from sklearn.datasets import load_wine
from sklearn.model_selection import train_test_split
from sklearn import metrics
from IPython.display import SVG
from graphviz import Source
from IPython.display import display
```

ขั้นตอนที่ 2 : Prepare data

- โหลดข้อมูล wine จาก sklearn.datasets ด้วยคำสั่ง load_wine
- โหลดข้อมูลเข้า pandas dataframe

```
# load dataset
data = load_wine()

df = pd.DataFrame(data.data, columns=data.feature_names)
df.head()
```

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols	proanthocy
0	14.23	1.71	2.43	15.6	127.0	2.80	3.06	0.28	2.29
1	13.20	1.78	2.14	11.2	100.0	2.65	2.76	0.26	1.28
2	13.16	2.36	2.67	18.6	101.0	2.80	3.24	0.30	2.81
3	14.37	1.95	2.50	16.8	113.0	3.85	3.49	0.24	2.18
4	13.24	2.59	2.87	21.0	118.0	2.80	2.69	0.39	1.82

- สร้างตัวแปรสำหรับ feature , label , ชื่อของแต่ละ feature

```
# feature matrix
X = data.data

# target vector
y = data.target

# class labels
labels = data.feature_names
```

ขั้นตอนที่ 3 : Create model

- แบ่งข้อมูล train set 70% และ test set 30% ด้วย sklearn.model_selection

```
# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1) # 70% training and 30% test
```

- สร้าง decision tree โดย criterion แบบ entropy ด้วย sklearn.tree
- ทำการ predict ข้อมูล

```
# Create Decision Tree classifer object
clf = DecisionTreeClassifier(criterion="entropy", max_depth=None)

# Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)
y_score = clf.score(X,y)
```

ขั้นตอนที่ 4 : Model performance

- ดูค่า accuracy , precision , recall , f-measure และ confusion matrix

```
from sklearn.metrics import classification_report,confusion_matrix

print(confusion_matrix(y_test,y_pred))
```

```
[[22  1  0]
 [ 1 18  0]
 [ 0  0 12]]
```

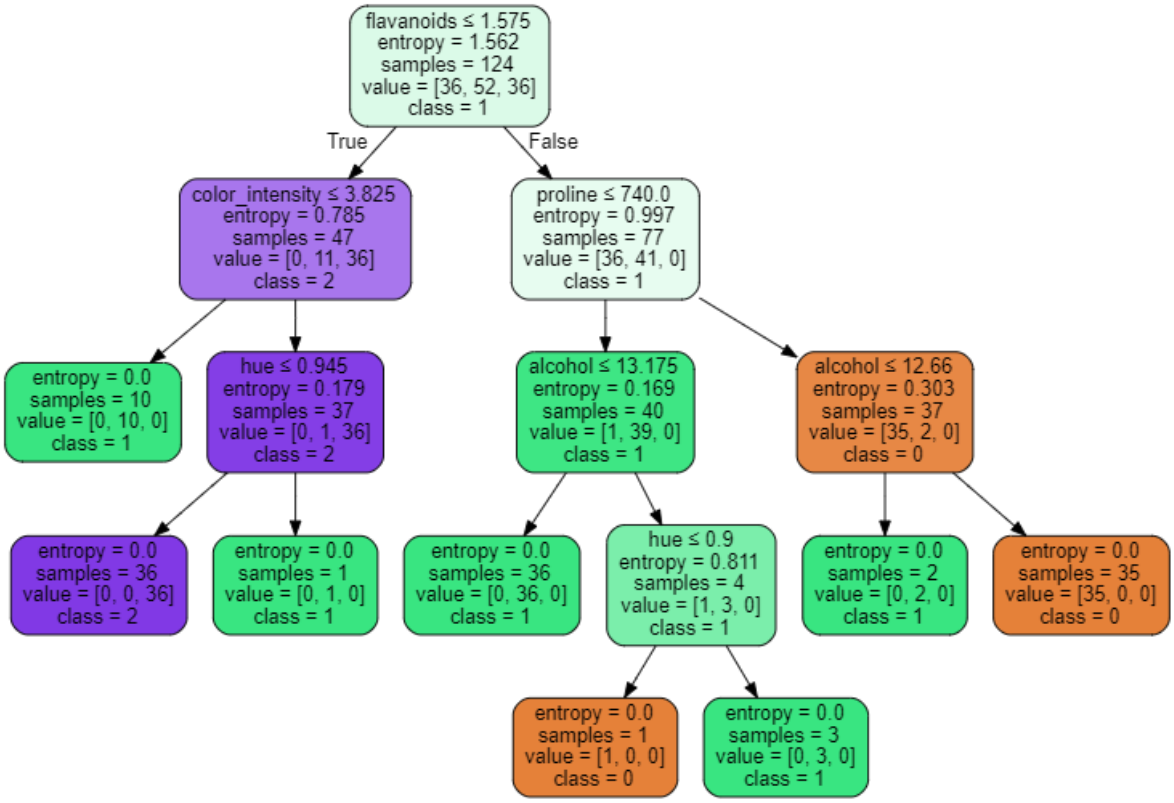
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.96      0.96      0.96        23
           1       0.95      0.95      0.95        19
           2       1.00      1.00      1.00        12

    accuracy                           0.96        54
   macro avg       0.97      0.97      0.97        54
weighted avg       0.96      0.96      0.96        54
```

ขั้นตอนที่ 5 : Visualize tree

- ทำการ visualization tree ด้วย graphviz

```
graph = Source(export_graphviz(clf, out_file=None,
                filled=True, rounded=True,
                special_characters=True,feature_names = labels,class_names=['0','1','2']))

display(SVG(graph.pipe(format='svg')))
```



Download Source Code - Decision Tree ได้ที่ : https://github.com/Kzis/ml-project/blob/master/decision%20tree/dicision_tree.ipynb

Artificial Neural Network

ทำการสร้าง Model บน Google Colaboratory ด้วยภาษา Python โดยมีขั้นตอนการสร้างใน 2 ขั้นตอนแรก เหมือนกับในส่วนของ Decision tree และมีส่วนที่แตกต่างกัน ตั้งแต่ขั้นตอนที่ 3 ดังต่อไปนี้

ขั้นตอนที่ 3 : Create model

- แบ่งข้อมูล train set 70% และ test set 30% ด้วย sklearn.model_selection

```
# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1) # 70% training and 30% test
```

- Standardized data ด้วย sklearn.preprocessing

```
##### Data Preprocessing #####

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

# Fit only to the training data
scaler.fit(X_train)
```

```
StandardScaler(copy=True, with_mean=True, with_std=True)
```

```
# Now apply the transformations to the data:
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

- สร้าง model ด้วย sklearn.nerual_network

```
##### Training the model #####

from sklearn.neural_network import MLPClassifier

mlp = MLPClassifier(hidden_layer_sizes=(13,13,13),max_iter=500)

mlp.fit(X_train,y_train)
```

ขั้นตอนที่ 4 : Model performance

- ดูค่า accuracy , precision , recall , f-measure และ confusion matrix

```
##### Predictions and Evaluation #####

predictions = mlp.predict(X_test)

from sklearn.metrics import classification_report,confusion_matrix

print(confusion_matrix(y_test,predictions))
```

```
[[17  0  0]
 [ 0 17  1]
 [ 0  2  8]]
```

```
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        17
           1       0.89      0.94      0.92        18
           2       0.89      0.80      0.84        10

    accuracy                           0.93        45
   macro avg       0.93      0.91      0.92        45
weighted avg       0.93      0.93      0.93        45
```

Download Source Code - ANN ได้ที่ : https://github.com/Kzis/ml-project/blob/master/nn/nn_wine.ipynb

K-mean

ทำการสร้าง Model บน Google Colaboratory ด้วยภาษา Python โดยมีขั้นตอนการสร้างใน 2 ขั้นตอนแรก เหมือนกับในส่วนของ Decision tree และมีส่วนที่แตกต่างกัน ตั้งแต่ขั้นตอนที่3 ดังต่อไปนี้

ขั้นตอนที่ 3 : Create model

- แบ่งข้อมูล train set 70% และ test set 30% ด้วย sklearn.model_selection

```
# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1) # 70% training and 30% test
```

- Standardized data ด้วย sklearn.preprocessing

```
##### Data Preprocessing #####

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

# Fit only to the training data
scaler.fit(X_train)
```

```
StandardScaler(copy=True, with_mean=True, with_std=True)
```

```
# Now apply the transformations to the data:
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

- สร้าง model ด้วย sklearn.cluster โดยกำหนด k=3 และทำการ predictive

```
##### Training the model #####
from sklearn.cluster import KMeans

model = KMeans(n_clusters=3, init = 'random', max_iter = 100, random_state = 5)
kmeans = model.fit(X)
kmeans
```

```
# Getting the cluster labels
output = kmeans.predict(X)
print(type(output))
print(output)
```

```
<class 'numpy.ndarray'>
[2 2 2 2 0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0 0 0 2 2 0 0 2 2 2 0 2 2 2 2 2 2 2 0 0
 2 2 0 0 2 2 0 0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 0 1 0 1 1 0 1 1 0 0 0 0 1 1 2
 0 1 1 1 0 1 1 0 0 1 1 1 1 1 0 0 1 1 1 1 1 0 0 1 0 1 0 1 1 1 1 0 1 1 1 1 0 1 0 1
 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0
 0 1 1 1 1 0 0 0 1 0 0 0 1 0 1 0 0 1 0 0 0 0 1 1 0 0 0 0 0 0 1]
```

จาก output จะเห็นได้ว่า model ทำการบอกว่าแต่ละ reccords อยู่ภายใน cluster ไหน แต่จะยังไม่สามารถสรุปได้ว่า cluster ที่แบ่งออกมานั้นคือ label class ไหน จึงต้องทำการหา distance ระหว่าง ข้อมูลค่าเฉลี่ยแต่ละ feature จัดกลุ่มด้วย label class ของข้อมูลตั้งต้น กับ ข้อมูลค่าเฉลี่ยแต่ละ feature จัดกลุ่มด้วย label class ของข้อมูลที่ออกมาจาก model

- ข้อมูลค่าเฉลี่ยแต่ละ feature จัดกลุ่มด้วย label class ของข้อมูลตั้งต้น

```
avg = wine.groupby('label').mean()
avg
```

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols
label								
0	13.744746	2.010678	2.455593	17.037288	106.338983	2.840169	2.982373	0.290000
1	12.278732	1.932676	2.244789	20.238028	94.549296	2.258873	2.080845	0.363662
2	13.153750	3.333750	2.437083	21.416667	99.312500	1.678750	0.781458	0.447500

- ข้อมูลค่าเฉลี่ยแต่ละ feature จัดกลุ่มด้วย label class ของข้อมูลที่ออกมาจาก model

```
avg_op = op_df.groupby('label').mean()
avg_op
```

	0	1	2	3	4	5	6	7	8	9	10
label											
0	12.929839	2.504032	2.408065	19.890323	103.596774	2.111129	1.584032	0.388387	1.503387	5.650323	0.883968
1	12.516667	2.494203	2.288551	20.823188	92.347826	2.070725	1.758406	0.390145	1.451884	4.086957	0.941159
2	13.804468	1.883404	2.426170	17.023404	105.510638	2.867234	3.014255	0.285319	1.910426	5.702553	1.078298

- หา distance โดย euclidiance distance ด้วย sklearn.metrics.pairwise

```
from sklearn.metrics.pairwise import euclidean_distances
vector_master0 = avg.to_numpy()[0]
vector_master1 = avg.to_numpy()[1]
vector_master2 = avg.to_numpy()[2]

vector_predict0 = avg_op.to_numpy()[0]
vector_predict1 = avg_op.to_numpy()[1]
vector_predict2 = avg_op.to_numpy()[2]

vector00 = np.round( np.sum( np.sqrt((vector_master0-vector_predict0)**2)) )
vector01 = np.round( np.sum( np.sqrt((vector_master0-vector_predict1)**2)) )
vector02 = np.round( np.sum( np.sqrt((vector_master0-vector_predict2)**2)) )
print("--------")
print("0-0,",vector00)
print("0-1,",vector01)
print("0-2,",vector02)

vector10 = np.round( np.sum( np.sqrt((vector_master1-vector_predict0)**2)) )
vector11 = np.round( np.sum( np.sqrt((vector_master1-vector_predict1)**2)) )
vector12 = np.round( np.sum( np.sqrt((vector_master1-vector_predict2)**2)) )
print("--------")
print("1-0,",vector10)
print("1-1,",vector11)
print("1-2,",vector12)

vector20 = np.round( np.sum( np.sqrt((vector_master2-vector_predict0)**2)) )
vector21 = np.round( np.sum( np.sqrt((vector_master2-vector_predict1)**2)) )
vector22 = np.round( np.sum( np.sqrt((vector_master2-vector_predict2)**2)) )
print("--------")
print("2-0,",vector20)
print("2-1,",vector21)
print("2-2,",vector22)
```

จะได้ว่า

- cluster ที่ 0 ความคล้ายกับ label class ที่ 2
- cluster ที่ 1 ความคล้ายกับ label class ที่ 1
- cluster ที่ 2 ความคล้ายกับ label class ที่ 0

ขั้นตอนที่ 4 : Model performance

- ดูค่า accuracy , precision , recall , f-measure และ confusion matrix

```
from sklearn.metrics import classification_report,confusion_matrix

print(confusion_matrix(y,output))
```

```
[[46  0 13]
 [ 1 50 20]
 [ 0 19 29]]
```

```
print(classification_report(y,output))
```

```
              precision    recall  f1-score   support

           0       0.98      0.78      0.87        59
           1       0.72      0.70      0.71        71
           2       0.47      0.60      0.53        48

    accuracy                           0.70       178
   macro avg       0.72      0.70      0.70       178
weighted avg       0.74      0.70      0.71       178
```

Download Source Code - KMean ได้ที่ : https://github.com/Kzis/ml-project/blob/master/kmeans/k_mean1.ipynb

Model Performace ของแต่ละ Model

- Decision tree

	precision	recall	f1-score	support
0	0.96	0.96	0.96	23
1	0.95	0.95	0.95	19
2	1.00	1.00	1.00	12
accuracy			0.96	54
macro avg	0.97	0.97	0.97	54
weighted avg	0.96	0.96	0.96	54

- ANN

	precision	recall	f1-score	support
0	1.00	1.00	1.00	17
1	0.89	0.94	0.92	18
2	0.89	0.80	0.84	10
accuracy			0.93	45
macro avg	0.93	0.91	0.92	45
weighted avg	0.93	0.93	0.93	45

- K-mean

	precision	recall	f1-score	support
0	0.98	0.78	0.87	59
1	0.72	0.70	0.71	71
2	0.47	0.60	0.53	48
accuracy			0.70	178
macro avg	0.72	0.70	0.70	178
weighted avg	0.74	0.70	0.71	178