จันทรรัตน์ ชุมแสง 62070505216

Homework-Predictive Modeling

- Clean data

```
[36] heartData.isnull().any()
```

```
age       False
sex       False
cp        False
trestbps  False
chol      False
fbs       False
restecg   False
thalach   False
exang     False
oldpeak   False
slope     False
ca        False
thal      False
target    False
dtype: bool
```

สร้าง column ของข้อมูลจาก numerical เป็น string

```
[37] cata_column = ["sex","cp","fbs","restecg","exang","thal","target"]
```

```
[38] heartDatacat = heartData[cata_column].astype(str)
```

```
[39] heartDatacat.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 7 columns):
 #   Column   Non-Null Count  Dtype 
---  ------   --------------  ----- 
 0   sex      303 non-null    object
 1   cp       303 non-null    object
 2   fbs      303 non-null    object
 3   restecg  303 non-null    object
 4   exang    303 non-null    object
 5   thal     303 non-null    object
 6   target   303 non-null    object
dtypes: object(7)
memory usage: 16.7+ KB
```

สร้าง column ของ numerical

```
[40] heartData_num = heartData.drop(columns=heartDatacat)
```

```
heartData_num
```

	age	trestbps	chol	thalach	oldpeak	slope	ca
0	63	145	233	150	2.3	0	0
1	37	130	250	187	3.5	0	0
2	41	130	204	172	1.4	2	0
3	56	120	236	178	0.8	2	0
4	57	120	354	163	0.6	2	0
...	...	...	...	...	...	...	...

- Data preparation

```
[42] df_cate= pd.get_dummies(heartDatacat,drop_first=True)
```

```
df_cate
```

	sex_1	cp_1	cp_2	cp_3	fbs_1	restecg_1	restecg_2	exang_1	thal_1	thal_2	thal_3	target_1
0	1	0	0	1	1	0	0	0	1	0	0	1
1	1	0	1	0	0	1	0	0	0	1	0	1
2	0	1	0	0	0	0	0	0	0	1	0	1
3	1	1	0	0	0	1	0	0	0	1	0	1
4	0	0	0	0	0	1	0	1	0	1	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...
298	0	0	0	0	0	1	0	1	0	0	1	0
299	1	0	0	1	0	1	0	0	0	0	1	0
300	1	0	0	0	1	1	0	0	0	0	1	0

Train model

```
[46] from sklearn.model_selection import train_test_split
```

```
[47] train,test = train_test_split(df_final2,train_size=0.7)
```

```
[48] train.index
```

```
Int64Index([291,  54, 297, 276, 163, 137, 257, 271,  23, 147,
            ...
            278,  30, 117, 169,  59, 175,  71, 227, 228, 195],
           dtype='int64', length=212)
```

```
[49] test.index
```

```
Int64Index([170,  65, 154, 218, 132, 267,  61, 203, 223, 123, 204,  93, 126,
            131, 182, 219,  83,  88, 164, 113, 197, 202,  16, 261, 159, 282,
              2, 188,   4, 295,  91, 184,  58, 139, 213, 121, 157, 140, 275,
            249, 181,  99, 208, 221,  26,  84, 162,  28,  43,  18,  49,  11,
            180,  19, 156, 270, 205, 191, 158, 199, 246, 234, 105,  76, 287,
            146, 144, 263, 148, 206, 214, 138,  46, 130, 102,  55, 281,  96,
             29, 209, 236, 116,   7, 196, 134, 155, 119,  41,  86,   5,   0],
           dtype='int64')
```

Concat data

```
[44] df_final2 = pd.concat([heartData_num,df_cate],axis=1)
```

```
[45] df_final2
```

	age	trestbps	chol	thalach	oldpeak	slope	ca	sex_1	cp_1	cp_2	cp_3	fbs_1	restecg_1	restecg_2	exang_1	thal_1	thal_2	thal_3	target_1
0	63	145	233	150	2.3	0	0	1	0	0	1	1	0	0	0	1	0	0	1
1	37	130	250	187	3.5	0	0	1	0	1	0	0	1	0	0	0	1	0	1
2	41	130	204	172	1.4	2	0	0	1	0	0	0	0	0	0	0	1	0	1
3	56	120	236	178	0.8	2	0	1	1	0	0	0	1	0	0	0	1	0	1
4	57	120	354	163	0.6	2	0	0	0	0	0	0	1	0	1	0	1	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	57	140	241	123	0.2	1	0	0	0	0	0	0	1	0	1	0	0	1	0
299	45	110	264	132	1.2	1	0	1	0	0	1	0	1	0	0	0	0	1	0
300	68	144	193	141	3.4	1	2	1	0	0	0	1	1	0	0	0	0	1	0
301	57	130	131	115	1.2	1	1	1	0	0	0	0	1	0	1	0	0	1	0
302	57	130	236	174	0.0	1	1	0	1	0	0	0	0	0	0	0	1	0	0

```
[52] train.shape
```

```
(212, 19)
```

```
[53] test.shape
```

```
(91, 19)
```

```
[54] df_final2.shape
```

```
(303, 19)
```

Model

- Train a decision tree madel

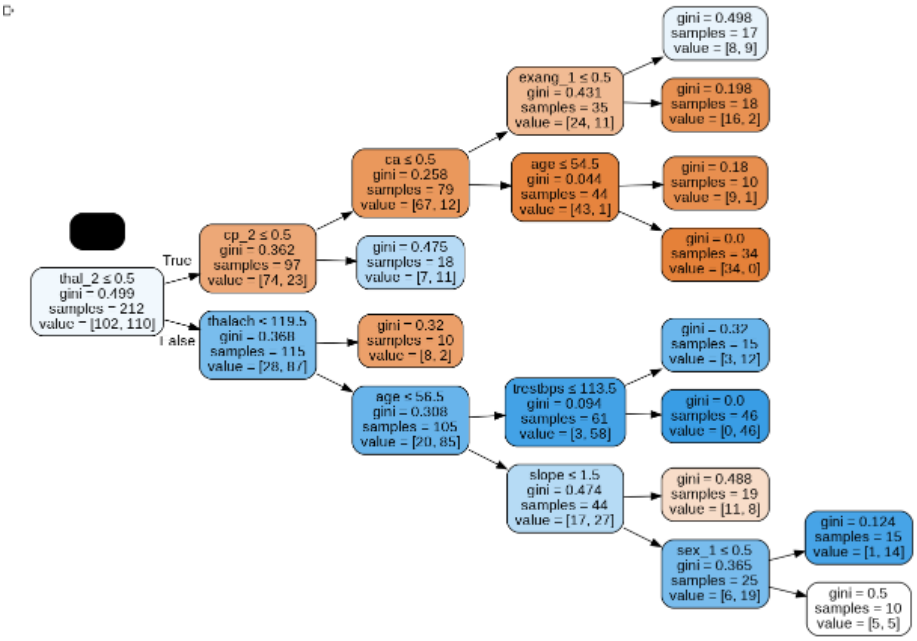
Train a decision tree model

```
[55] tree = DecisionTreeClassifier(min_samples_leaf=10)
     tree.fit(train.drop(columns='target_1'),
              train['target_1'])
```

```
DecisionTreeClassifier(min_samples_leaf=10)
```

Plot a decision tree

```
[56] dot_data = StringIO()
     export_graphviz(tree, out_file=dot_data,
                     filled=True, rounded=True,
                     special_characters=True,
                     rotate=True,
                     feature_names=train.columns[:-1])
     graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
     graph.write_png('tree.png')
     Image(graph.create_png())
```



Prediction

```
[136] tree.predict(test.drop(columns='target_1'))
```

```
array([0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1,
       1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1,
       1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0,
       0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1,
       1, 1, 1], dtype=uint8)
```

```
[137] tree.predict_proba(test.drop(columns='target_1'))
```

```
array([[0.63157895, 0.36842105],
       [0.15789474, 0.84210526],
       [0.21052632, 0.78947368],
       [0.        , 1.        ],
       [0.        , 1.        ],
       [0.        , 1.        ],
       [0.        , 1.        ],
       [0.84210526, 0.15789474],
       [0.        , 1.        ],
       [0.21052632, 0.78947368],
       [0.        , 1.        ],
       [1.        , 0.        ]
```

Classification report

```
from sklearn.metrics import classification_report
```

```
[140] res = tree.predict(test.drop(columns='target_1'))
      print(classification_report(y_true=test['target_1'].values, y_pred=res))
```

```
              precision    recall  f1-score   support

           0       0.83      0.66      0.73        44
           1       0.73      0.87      0.80        47

    accuracy                           0.77        91
   macro avg       0.78      0.77      0.77        91
weighted avg       0.78      0.77      0.77        91
```

F1-score = 0.80

Variable importance

```
pd.DataFrame(dict(Feature=train.columns[:-1],
                  Value=tree.feature_importances_))\
.sort_values(by='Value', ascending=False)
```

	Feature	Value
16	thal_2	0.478467
3	thalach	0.114333
9	cp_2	0.104197
0	age	0.100559
6	ca	0.055853
14	exang_1	0.051562
5	slope	0.041803
7	sex_1	0.037975
1	trestbps	0.015250

LDA

```
[58] lda = LinearDiscriminantAnalysis()
     lda.fit(train.drop(columns='target_1'), train['target_1'])
```

```
LinearDiscriminantAnalysis()
```

```
pd.DataFrame(dict(Feature = train.columns[:-1],
                  Coefficient = lda.coef_[0]))
```

	Feature	Coefficient
0	age	-0.001009
1	trestbps	-0.012131
2	chol	-0.003140
3	thalach	0.025436
4	oldpeak	-0.251708
5	slope	0.774619
6	ca	-0.777719

```
[60] res_lda = lda.predict(test.drop(columns='target_1'))
     print(classification_report(y_true=test['target_1'].values, y_pred=res_lda))
```

```
              precision    recall  f1-score   support

           0       0.82      0.75      0.78        36
           1       0.84      0.89      0.87        55

    accuracy                           0.84        91
   macro avg       0.83      0.82      0.82        91
weighted avg       0.83      0.84      0.83        91
```

F1-score = 0.87

Logistic regression

```
lr = LogisticRegression()
lr.fit(train.drop(columns='target_1'), train['target_1'])
```

```
pd.DataFrame(dict(Feature = train.columns[:-1],
                  Coefficient = lr.coef_[0]))
```

	Feature	Coefficient
0	age	0.001350
1	trestbps	-0.012605
2	chol	-0.005064
3	thalach	0.021839
4	oldpeak	-0.251361
5	slope	0.608166
6	ca	-0.737009

```
[63] res_lr = lr.predict(test.drop(columns='target_1'))
     print(classification_report(y_true=test['target_1'].values, y_pred=res_lr))
```

```
              precision    recall  f1-score   support

           0       0.88      0.78      0.82        36
           1       0.86      0.93      0.89        55

    accuracy                           0.87        91
   macro avg       0.87      0.85      0.86        91
weighted avg       0.87      0.87      0.87        91
```

F1-score = 0.89