|  |
| --- |
| **Machine Learning**  **Lab1** |

****

|  |  |  |  |
| --- | --- | --- | --- |
| **제출일** | **2022.9.13** | **전공** | **소프트웨어학과** |
| **과목명** | **머신러닝** | **이름** | **안해빈, 이지민,**  **임윤수, 윤주은** |

**1. Objective Setting**

We would like to create a “single major function” that will automatically run different combinations of the following : Data Scaling, Data Encoding, Classification Models with different K parameters, and various number k for k-fold cross validation.

We will use the Wisconsin Cancer Dataset to test the algorithm in the end-to-end process as used in Data Science.

The Program will run through loading data, data preparation, data analysis, data inspection automatically and show the best 5 results of combination.

**2. Data Curation**

Dataset Name : Wisconsin Cancer Dataset

Source : <https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/>

(breast-cancer-wisconsin.data)

**3. Data Inspection**

- Number of Instances: 699 (as of 15 July 1992)

- Number of Attributes: 10 plus the class attribute

- Attribute Information: (class attribute has been moved to last column)

# Attribute Domain

-- -----------------------------------------

1. Sample code number id number

2. Clump Thickness 1 - 10

3. Uniformity of Cell Size 1 - 10

4. Uniformity of Cell Shape 1 - 10

5. Marginal Adhesion 1 - 10

6. Single Epithelial Cell Size 1 - 10

7. Bare Nuclei 1 - 10

8. Bland Chromatin 1 - 10

9. Normal Nucleoli 1 - 10

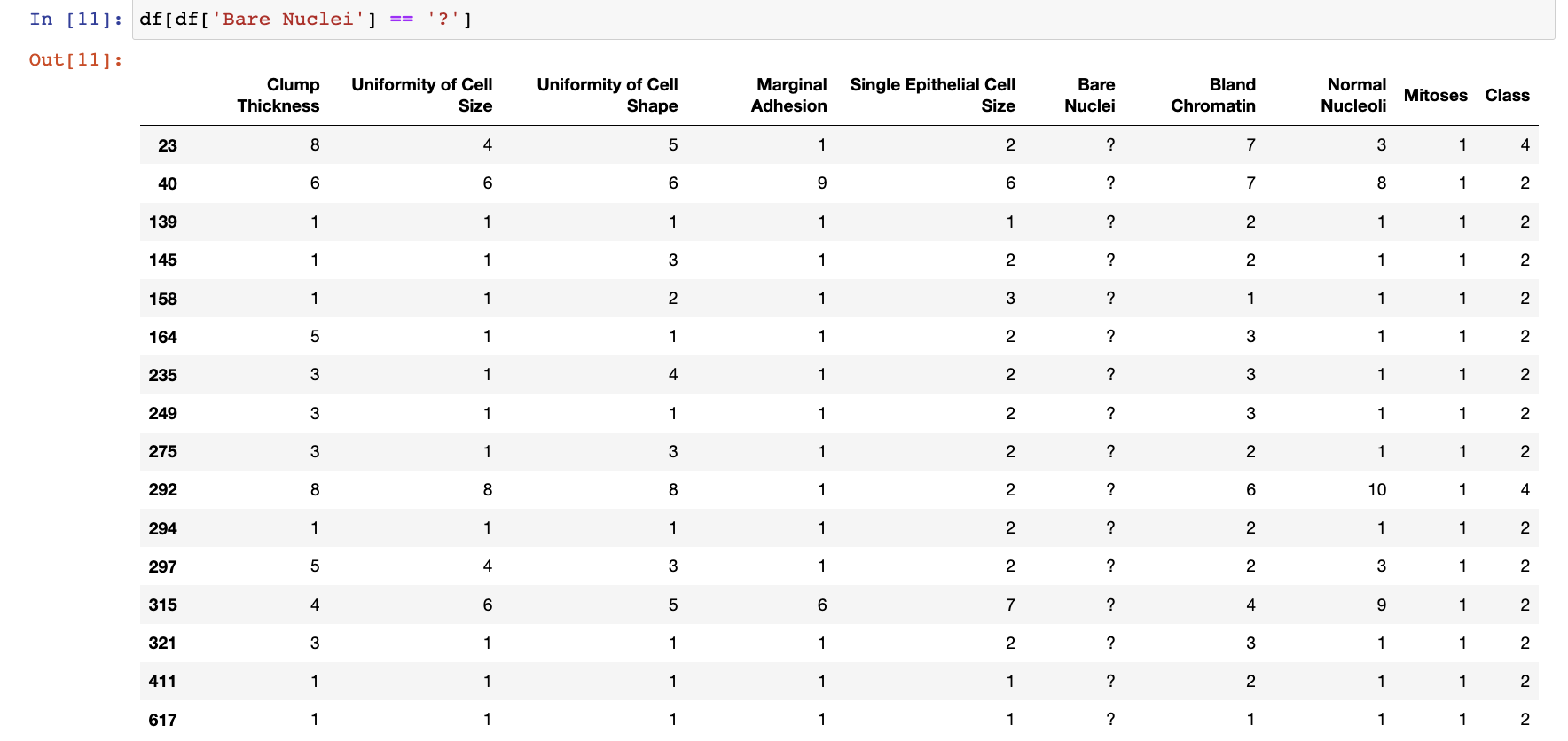
10. Mitoses 1 - 10

11. Class: (2 for benign, 4 for malignant)

- Missing attribute values: 16

There are 16 instances in Groups 1 to 6 that contain a single missing

(i.e., unavailable) attribute value, now denoted by "?".



- Class distribution:

Benign: 458 (65.5%)

Malignant: 241 (34.5%)

**4. Data Preparation - Plans**

Dirty data cleaning

Bare nuclei is the only attribute with missing values, ‘?’.

→ we will replace ‘?’ to mode value.

Feature engineering

Since the first attribute ‘sample code number’ is useless in analyzing process, we will drop this attribute. Every other attributes are necessary

**Scaling (5)** - Standard, MinMax, Robust, MaxAbsScaler, Normalizer

**Encoding (0)** – All data are numerical numbers, so no Encoding is needed.

**5. Data Analysis - Plans**

Decision Tree (entropy)

Decision Tree (gini index)

Logistic Regression

Support Vector Machine

----------------------------------

+ try different hyperparameters

**6. Evaluation**

K fold validation (k= 5, 8, 10)

**7. Combination cases**

Scaling(5) \* Encoding(-) \* Algorithms(4) \* K fold(3) = 60

We plan to make **60** cases.

**8. Algorithm Structure**

scaler = [‘StandardScaler()’, ‘MinMaxScaler()’, ‘RobustScaler()’, ‘MaxAbsScaler()’, ‘Normalizer()’]

model = [‘DecisionTreeClassifier(criterion=’gini)’, ‘DecisionTreeClassifier(criterion=’entrophy)’, …]

def best\_comb(scaler, model):

"""Train model and find best combination of classifier and scaler

Args:

scaler: Array of scaler you want to use

model: Array of classifier model you want to use

Returns:

best\_acc: return best accuracy among combination of scaler and model

best\_scaler: return scaler which has best score

best\_model: return model which has best score

x\_bTest: return x\_test which has best score

y\_Btest: return y\_test which has best score

y\_Bpred: return y\_pred which has best score

"""

best\_acc = 0

for element in scaler:

scaler = eval(element)

scaled = scaler.fit\_transform(X\_clf)

x\_train, x\_test, y\_train, y\_test = train\_test\_split(scaled, y\_clf, test\_size = 0.2, random\_state=42) #Using random\_state to fixing random rate

for element2 in model:

classifier = eval(element2)

classifier = classifier.fit(x\_train,y\_train)

y\_pred = classifier.predict(x\_test)

acc = accuracy\_score(y\_test, y\_pred)

print(f'Using {classifier} in {scaler} score : {acc}')

if acc > best\_acc:

best\_acc = acc

best\_scaler = element

best\_model = classifier

x\_bTest = x\_test

y\_bTest = y\_test

y\_bPred = y\_pred

print('')

return best\_acc, best\_scaler, best\_model, x\_bTest, y\_bTest, y\_bPred

and calculate k-fold validation