**2022-2 Machine Learning Team Project 1**

**201835516 Ganghoon Jeon**

**201835512 Yunyoung Jang**

**202035362 Sohyun Lee**

**202035369 Jongeun Lee**



Print all looped model, scaler and hyperparameters and then print best score, used model and used scaler

Code)

import pandas as pd

from sklearn.preprocessing import StandardScaler

from sklearn.preprocessing import RobustScaler

from sklearn.preprocessing import Normalizer

from sklearn.preprocessing import MinMaxScaler

from sklearn.preprocessing import MaxAbsScaler

from sklearn.tree import DecisionTreeClassifier

from sklearn.linear\_model import LogisticRegression

from sklearn.svm import SVC

from sklearn.model\_selection import KFold

from sklearn.model\_selection import cross\_val\_score

from sklearn.metrics import accuracy\_score

import numpy as np

def best\_combi(x, y):

#5 scalers list

best\_score = 0

best\_model="m"

best\_scaler = "s"

#Various data scaling methods

scalers = ['StandardScaler()', 'RobustScaler()', 'Normalizer()', 'MinMaxScaler()', 'MaxAbsScaler()']

#breast\_cancer dataset don't need to encode

#encoder =[]

#4 models list

models = ['DecisionTreeClassifier(criterion="entropy",', 'DecisionTreeClassifier(criterion="gini",', 'LogisticRegression(solver="liblinear",', 'SVC(']

#model name is key and list index 0 means parameter name. index 1 means type of parameter and remainders are value

find\_param={

'DecisionTreeClassifier(criterion="entropy",': ['max\_depth', 'int', '2', '3', '4'],

'DecisionTreeClassifier(criterion="gini",':['max\_depth', 'int', '2', '3', '4'],

'LogisticRegression(solver="liblinear",' : ['penalty', 'str', 'l1', 'l2'],

'SVC(' : ['gamma', 'str', 'scale', 'auto']}

#loop 5 x 4

for s in scalers:

for m in models:

#various value of model with various values for the hyperparameters

for i in range(0, len(find\_param[m])-2):

#to send like DecisionTreeClassifier(criterion="entropy",max\_depth=int(2))

if(find\_param[m][1] == "str"):

model\_name = m+find\_param[m][0]+"="+find\_param[m][1]+'(\"'+find\_param[m][2+i]+"\"))"

else:#suppose int

model\_name = m+find\_param[m][0]+"="+find\_param[m][1]+'('+find\_param[m][2+i]+"))"

model = eval(model\_name)

scaler = eval(s)

#scaling

temp\_x = scaler.fit\_transform(X)

#Various number k for k-fold cross validation and send maximum score

scores = []

for i in range(5,8):

kfold = KFold(n\_splits=i, shuffle=True, random\_state=7)

score = cross\_val\_score(model, temp\_x,Y , cv = kfold)

score = np.mean(score)

scores.append(score)

score = max(scores)

print("model: "+model\_name+"\tscaler: "+s+"\tscore: "+str(score))

#update best\_score

if score>best\_score:

best\_score = score

best\_model = model\_name

best\_scaler = s

return best\_score, best\_model, best\_scaler

#We did data analysis in Lab1. so we'll skip it

#---Preprocessing---

#load data with column names

data = pd.read\_csv('./breast-cancer-wisconsin.data', names=["Sample code number", "Clump Thickness", "Uniformity of Cell Size", "Uniformity of Cell Shape", "Marginal Adhesion", "Single Epithelial Cell Size", "Bare Nuclei", "Bland Chromatin", "Normal Nucleoli", "Mitoses", "Class"])

#Delete missing value("?")

missing\_index = data[data['Bare Nuclei']=='?'].index

data = data.drop(missing\_index)

#Delete Sample code number(ID) column

data = data.drop(['Sample code number'], axis=1)

#Delete duplicated instances

#After delete sample code number(ID), I saw more duplicated instances(234).

data = data.drop\_duplicates()

#split features and target

X = data.drop('Class', 1)

Y = data['Class']

best\_score, best\_model, best\_scaler=best\_combi(X, Y)

print("Best Score:"+str(best\_score)+" Scaler:"+best\_scaler+" model:"+best\_model)

**Documentation**

Table of contents

1. Introduction of Dataset used
2. Framework of Major function best\_combi
3. Result of Example Dataset

1.Introduction of Dataset used

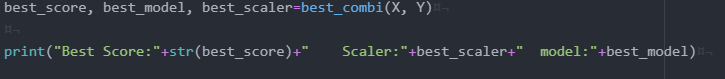
 The sample dataset used in this code is  Wisconsin Breast Cancer Database (January 8, 1991), which can be downloaded in this link : <https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/> .

텍스트이(가) 표시된 사진

자동 생성된 설명

Since the main function best\_score doesn’t have any preprocessing method, The code included a pre-processing process, which drops missing value ‘?’ in the original dataset and drop ID, the Sample code number , and redundancy records in the code.

2. Framework of Major function best\_combi



This is the used part of our major function, best\_combi.

best\_combi is a function that finds the best combination which results in the best accuracy for a specific pre-processed dataset.

텍스트이(가) 표시된 사진

자동 생성된 설명

At first, the function uses models and scalers to find the best combination which results in best accuracy. In this function, Various models with various hyperparameters and conditions such as criterion and gamma criterion used in models while various scalers such as standard scaler used for finding the best combination.

텍스트이(가) 표시된 사진

자동 생성된 설명

After setting models and scalers used for combination, the function starts to find the best combination by using KFold. After using kFold, the function finds the best score among the combinations and when it decides the best score, stores the condition to return.

after comparing combinations, the feature returns the best condition and its result.

텍스트이(가) 표시된 사진

자동 생성된 설명

This process is done by only one function, best\_combi. Users can find out and determine which combination results best by using this function.

3.Result of Example Dataset

텍스트, 신문, 영수증이(가) 표시된 사진

자동 생성된 설명

We show 4 machine learning model and 5 Scaler of combination. The results are as follows:

Model: 4

Scaler: 5

The number of cases : 20

Best of model: SVC(gamma=str(“auto”)

Best of Scaler: MaxAbsScaler

Highest Accuracy: 0.9576876876876876..

The result of Svc has highest accuracy of models, and MaxAbsScaler has highest accuracy of scalers. This is because SVC calculates Margin to classify the dataset well.

Evaluation:

Advantages

► show all number of cases with accuracy, so user can easily check what models are appropriate to datasets

►In major function, using double repeat statement to  reduce the amount of code

|  |  |  |
| --- | --- | --- |
| **Team Contribution percentage & Their roles** | | |
| **Name** | **Role Contributed** | **Percentage** |
| **Ganghoon jeon** | **Debugging/Code plan adjustment** | **25%** |
| **Yunyoung Jang** | **Code optimization** | **25%** |
| **Sohyun Lee** | **Documentation** | **25%** |
| **Jongeun Lee** | **Test / code review** | **25%** |
| **Common Role contributed** | **Programmed draft code** | |