6주. Decision Tree, RF, SVM			
학번	32152339	이름	송준영

PimaIndiansDiabetes dataset을 가지고 Classification 을 하고자 한다. (마지막의 diabetes 컬럼이 class label 임)

Q1 (4점) scikit-learn에서 제공하는 DecisionTree, RandonForest, support vector machine 알고리즘를 이용하여 **PimaIndiansDiabetes dataset**에 대한 분류 모델을 생성하고 accuracy를 비교하시오.
- 10-fold cross validation을 실시하여 mean accuracy를 비교한다
- 각 알고리즘의 hyper parameter 의 값은 default value를 이용한다.

Source code :

```
// source code 의 폰트는 Courier10 BT Bold으로 하시오
pip install pydot
import pandas as pd
import numpy as np
from sklearn.tree import DecisionTreeClassifier, export_graphviz
from sklearn.ensemble import RandomForestClassifier
from sklearn import svm
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
import pandas as pd
import pydot
from sklearn.model_selection import KFold
#decision tree
df=pd.read_csv('C:/Users/ATIV/Desktop/Deeplearning_Cloud/dataset_09
14/PimaIndiansDiabetes.csv')
df
df_X = df.loc[:, df.columns != 'diabetes']
df_y = df['diabetes']
df_X=df_X.values
df_y=df_y.values

seed=32152339
n_fold=10
kf = KFold(n_splits=n_fold, random_state=seed, shuffle=True)       #
10 fold

# Define learning model
DT_model = DecisionTreeClassifier(random_state=seed)
RF_model = RandomForestClassifier(random_state=seed)
```

```
SVM_model = svm.SVC()
models = [DT_model,RF_model,SVM_model]

for model in models:
    acc = np.zeros(n_fold)
    i = 0
    print(model)
    for train_index, test_index in kf.split(df_X):
        print("fold:", i)

        train_X, test_X = df_X[train_index], df_X[test_index]
        train_y, test_y =  df_y[train_index], df_y[test_index]

        # Train the model using the training sets
        model.fit(train_X, train_y)

        # Make predictions using the testing set
        pred_y = model.predict(test_X)
        #print(pred_y)

        # model evaluation: accuracy ##############
        acc[i] = accuracy_score(test_y, pred_y)
       # print('Accuracy : {0:3f}'.format(acc[i]))
        i += 1

    print(model)
    #print(f"{n_fold} fold :", acc)
    print("mean accuracy :", np.mean(acc))
    print('-----------------------------------------')
```

실행화면 캡쳐:

```
DecisionTreeClassifier(random_state=32152339)
fold: 0
fold: 1
fold: 2
fold: 3
fold: 4
fold: 5
fold: 6
fold: 7
fold: 8
fold: 9
DecisionTreeClassifier(random_state=32152339)
mean accuracy : 0.6980177717019821
-----------------------------------------
RandomForestClassifier(random_state=32152339)
fold: 0
fold: 1
fold: 2
fold: 3
fold: 4
fold: 5
fold: 6
fold: 7
fold: 8
fold: 9
RandomForestClassifier(random_state=32152339)
mean accuracy : 0.7617737525632263
-----------------------------------------
SVC()
fold: 0
fold: 1
fold: 2
fold: 3
fold: 4
fold: 5
fold: 6
fold: 7
fold: 8
fold: 9
SVC()
mean accuracy : 0.7564593301435407
-----------------------------------------
```

-> randomforest가 0.76으로 가장 성능이 좋다.

Q2. (3점) 다음의 조건에 따라 support vector machine 알고리즘를 이용하여 **PimaIndiansDiabetes dataset**에 대한 분류 모델을 생성하고 accuracy를 비교하시오.
- hyper parameter 중 kernel 에 대해 linear, poly, rbf, sigmoid, precomputed를 각각 테스트하여 어떤 kernel 이 가장 높은 accuracy를 도출하는지 확인하시오.
- 10-fold cross validation을 실시하여 mean accuracy를 비교한다

Source code :

```
// source code 의 폰트는 Courier10 BT Bold으로 하시오
kernels=['linear', 'poly', 'rbf', 'sigmoid','precomputed']

for k in kernels:
    print(f'svc kernels : {k}')
    model = svm.SVC(kernel=k)
    acc = np.zeros(n_fold)     # 10 fold
    i = 0                # fold no
    for train_index, test_index in kf.split(df_X):
        print("fold:", i)

        train_X, test_X = df_X[train_index], df_X[test_index]
        train_y, test_y =  df_y[train_index], df_y[test_index]

        if k != 'precomputed':
            model.fit(train_X, train_y)
            pred_y = model.predict(test_X)
        else :
            gram_train = np.dot(train_X, train_X.T)
            model.fit(gram_train, train_y)
            gram_test = np.dot(test_X, train_X.T)
            pred_y = model.predict(gram_test)

        acc[i] = accuracy_score(test_y, pred_y)
        i += 1

    print(model)
   # print(f"{n_fold} fold :", acc)
    print("mean accuracy :", np.mean(acc))
    print()
```

실행화면 캡쳐:

```
svc kernels : linear
fold: 0
fold: 1
fold: 2
fold: 3
fold: 4
fold: 5
fold: 6
fold: 7
fold: 8
fold: 9
SVC(kernel='linear')
mean accuracy : 0.7733937115516063

svc kernels : poly
fold: 0
fold: 1
fold: 2
fold: 3
fold: 4
fold: 5
fold: 6
fold: 7
fold: 8
fold: 9
SVC(kernel='poly')
mean accuracy : 0.7564422419685578

svc kernels : rbf
fold: 0
fold: 1
fold: 2
fold: 3
fold: 4
fold: 5
fold: 6
fold: 7
fold: 8
fold: 9
SVC()
mean accuracy : 0.7564593301435407

svc kernels : sigmoid
fold: 0
fold: 1
fold: 2
fold: 3
fold: 4
fold: 5
fold: 6
fold: 7
fold: 8
fold: 9
SVC(kernel='sigmoid')
mean accuracy : 0.49743677375256323

svc kernels : precomputed
fold: 0
fold: 1
fold: 2
fold: 3
fold: 4
fold: 5
fold: 6
fold: 7
fold: 8
fold: 9
SVC(kernel='precomputed')
mean accuracy : 0.7733937115516063
```

-> linear kernel과 precomputed kernel이 0.77로 accuracy 성능이 좋다.

Q3. (3점) 다음의 조건에 따라 Random Forest 알고리즘를 이용하여 **PimaIndiansDiabetes dataset**에 대한 분류 모델을 생성하고 accuracy를 비교하시오.
-다음의 hyper parameter를 테스트 하시오
. n_estimators : 100, 200, 300, 400, 500
. max_features : 1, 2, 3, 4, 5
어떤 조합이 가장 높은 accuracy를 도출하는지 확인하시오.
- 10-fold cross validation을 실시하여 mean accuracy를 비교한다

Source code :

```
// source code 의 폰트는 Courier10 BT Bold으로 하시오
n_estimators  = [100, 200, 300, 400, 500]
max_features = [1, 2, 3, 4, 5]
mean_acc=[]

for n_estimator in n_estimators:
    print(f'n_estimators : {n_estimator}')
    for max_feature in max_features:

model=RandomForestClassifier(n_estimators=n_estimator,max_features=
max_feature)
        acc = np.zeros(n_fold)     # 10 fold
        i = 0
        for train_index, test_index in kf.split(df_X):
            #print("fold:", i)

            train_X, test_X = df_X[train_index], df_X[test_index]
            train_y, test_y =  df_y[train_index], df_y[test_index]

            model.fit(train_X, train_y)

            pred_y = model.predict(test_X)

            acc[i] = accuracy_score(test_y, pred_y)
            i += 1

        print(model)
#        print(f"{n_fold} fold :", acc)
        print("mean accuracy :", np.mean(acc))

mean_acc.append([n_estimator,max_feature,round(np.mean(acc),5)])
        print()
```

```
mean_acc


mean_acc=pd.DataFrame(mean_acc)
mean_acc.columns=['n_estimators','max_features','mean_accuracy']


mean_acc[mean_acc.mean_accuracy==np.max(mean_acc.mean_accuracy)]
```

실행화면 캡쳐:

```
n_estimators : 100
RandomForestClassifier(max_features=1)
mean accuracy : 0.756578947368421

RandomForestClassifier(max_features=2)
mean accuracy : 0.7630382775119618

RandomForestClassifier(max_features=3)
mean accuracy : 0.7617395762132604

RandomForestClassifier(max_features=4)
mean accuracy : 0.7486671223513328

RandomForestClassifier(max_features=5)
mean accuracy : 0.7435064935064937

n_estimators : 200
RandomForestClassifier(max_features=1, n_estimators=200)
mean accuracy : 0.7540157211209844

RandomForestClassifier(max_features=2, n_estimators=200)
mean accuracy : 0.7682330827067669

RandomForestClassifier(max_features=3, n_estimators=200)
mean accuracy : 0.7630895420369106

RandomForestClassifier(max_features=4, n_estimators=200)
mean accuracy : 0.7604408749145593

RandomForestClassifier(max_features=5, n_estimators=200)
mean accuracy : 0.7487183868762817

n_estimators : 300
RandomForestClassifier(max_features=1, n_estimators=300)
mean accuracy : 0.7669856459330144

RandomForestClassifier(max_features=2, n_estimators=300)
mean accuracy : 0.7643882433356117

RandomForestClassifier(max_features=3, n_estimators=300)
mean accuracy : 0.7643540669856461

RandomForestClassifier(max_features=4, n_estimators=300)
mean accuracy : 0.7526144907723855
```

```
RandomForestClassifier(max_features=5, n_estimators=300)
mean accuracy : 0.7448051948051949

n_estimators : 400
RandomForestClassifier(max_features=1, n_estimators=400)
mean accuracy : 0.7591934381408065

RandomForestClassifier(max_features=2, n_estimators=400)
mean accuracy : 0.7630553656869445

RandomForestClassifier(max_features=3, n_estimators=400)
mean accuracy : 0.755228981544771

RandomForestClassifier(max_features=4, n_estimators=400)
mean accuracy : 0.7487012987012986

RandomForestClassifier(max_features=5, n_estimators=400)
mean accuracy : 0.7577922077922079

n_estimators : 500
RandomForestClassifier(max_features=1, n_estimators=500)
mean accuracy : 0.7591934381408065

RandomForestClassifier(max_features=2, n_estimators=500)
mean accuracy : 0.760475051264525

RandomForestClassifier(max_features=3, n_estimators=500)
mean accuracy : 0.7604408749145593

RandomForestClassifier(max_features=4, n_estimators=500)
mean accuracy : 0.7473855092276145

RandomForestClassifier(max_features=5, n_estimators=500)
mean accuracy : 0.7473855092276145

Out[10]:
   n_estimators  max_features  mean_accuracy
6           200             2        0.76823
```

n_estimators : 200

max_features: 2

의 hyper parametr가 가장 성능이 좋다.