5주. Decision Tree, RF, SVM

학번	32171373	이름	노병우

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.

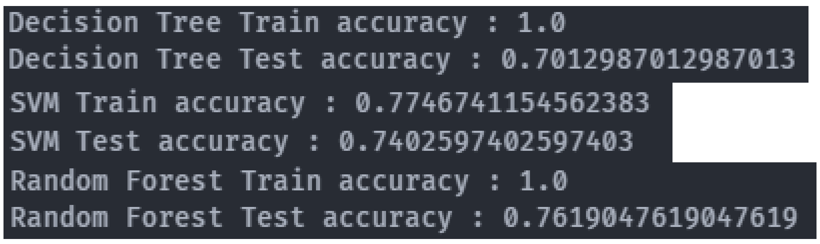
- 각 알고리즘의 hyper parameter 의 값은 default value를 이용한다.

Source code :

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split

df = pd.read_csv('../../data/PimaIndiansDiabetes.csv')

# diabetes Integer 형식으로 변경 pos = 1, neg = 0
label = {'diabetes' : {"pos" : 1, "neg" : 0}}
df = df.replace(label)

print(df)

df_X = df.loc [:, df.columns != 'diabetes']
df_y= df['diabetes']

train_X , test_X , train_y , test_y = train_test_split(df_X , df_y ,
test_size = 0.3, random_state = 1234)

#Decision Tree Model
from sklearn.tree import DecisionTreeClassifier
```

```
model = DecisionTreeClassifier()
model.fit(train_X, train_y)


print('Decision Tree Train accuracy :',model.score(train_X,
train_y))
print('Decision Tree Test accuracy :',model.score(test_X, test_y))

#Support Vector Machine Model
from sklearn import svm


model = svm.SVC()
model.fit(train_X, train_y)

print('SVM Train accuracy :',model.score(train_X, train_y))
print('SVM Test accuracy :',model.score(test_X, test_y))

#Radom Forest Model
from sklearn.ensemble import RandomForestClassifier


model = RandomForestClassifier()
model.fit(train_X, train_y)

print('Random Forest Train accuracy :',model.score(train_X,
train_y))
print('Random Forest Test accuracy :',model.score(test_X, test_y))
```

실행화면 캡처:



```
Decision Tree Train accuracy : 1.0
Decision Tree Test accuracy : 0.7012987012987013
SVM Train accuracy : 0.7746741154562383
SVM Test accuracy : 0.7402597402597403
Random Forest Train accuracy : 1.0
Random Forest Test accuracy : 0.7619047619047619
```

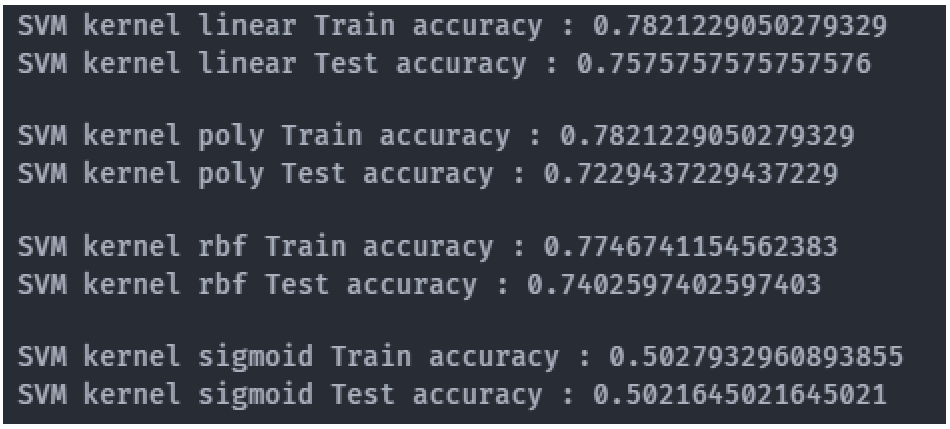
Q2. (3점) 다음의 조건에 따라 support vector machine 알고리즘를 이용하여 PimaIndiansDiabetes dataset에 대한 분류 모델을 생성하고 accuracy를 비교하시오.

- hyper parameter 중 kernel 에 대해 linear, poly, rbf, sigmoid를 각각 테스트하여 어떤 kernel 이 가장 높은 accuracy를 도출하는지 확인하시오.

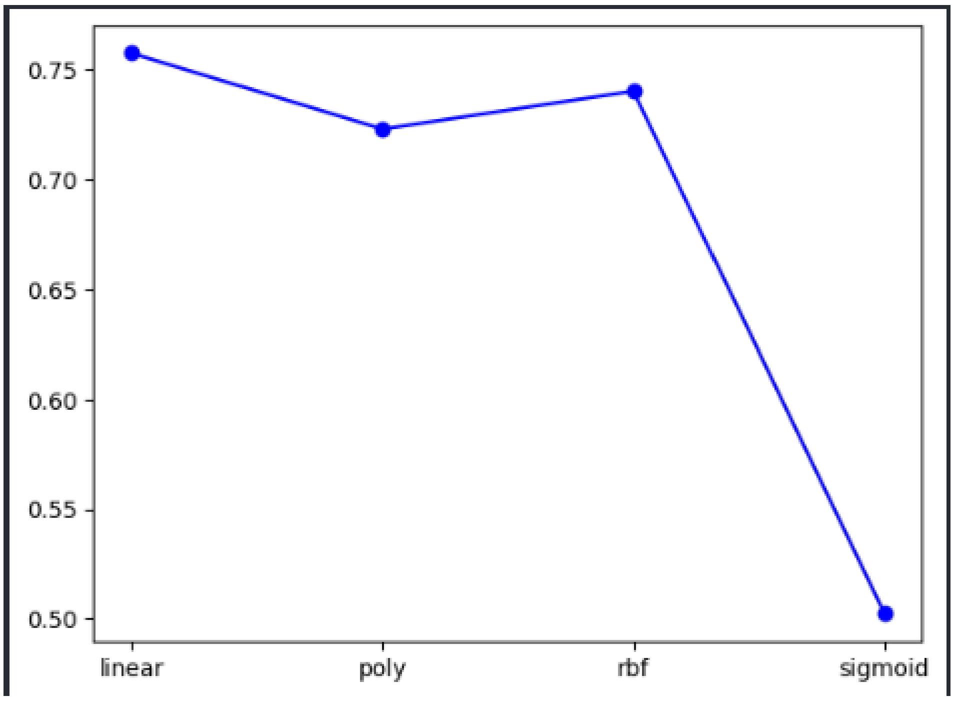
Source code :

```
model_name = ['linear', 'poly', 'rbf', 'sigmoid']

for mod in model_name:

    model = svm.SVC(kernel=mod)
    model.fit(train_X, train_y)

    accuracy.append(model.score(test_X, test_y))

    print(f'SVM kernel {mod} Train accuracy : {model.score(train_X, train_y)}')
    print(f'SVM kernel {mod} Test accuracy : {model.score(test_X, test_y)}\n')
```

실행화면 캡처:



```
SVM kernel linear Train accuracy : 0.7821229050279329
SVM kernel linear Test accuracy : 0.7575757575757576

SVM kernel poly Train accuracy : 0.7821229050279329
SVM kernel poly Test accuracy : 0.7229437229437229

SVM kernel rbf Train accuracy : 0.7746741154562383
SVM kernel rbf Test accuracy : 0.7402597402597403

SVM kernel sigmoid Train accuracy : 0.5027932960893855
SVM kernel sigmoid Test accuracy : 0.5021645021645021
```



Q3. (3점) 다음의 조건에 따라 Random Forest 알고리즘를 이용하여 PimaIndiansDiabetes dataset에 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

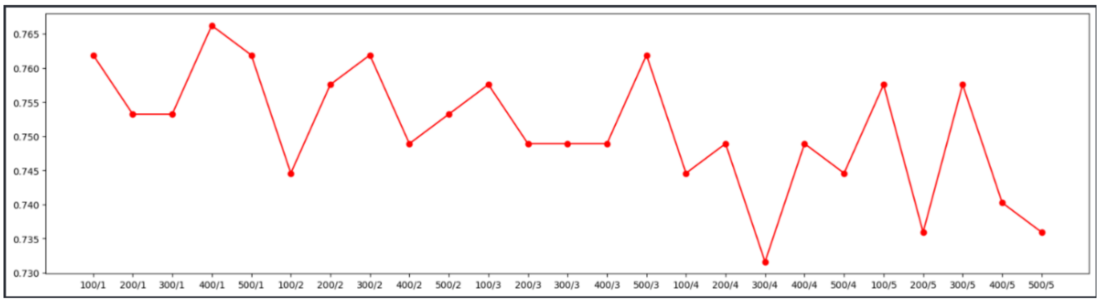
어떤 조합이 가장 높은 accuracy를 도출하는지 확인하시오.

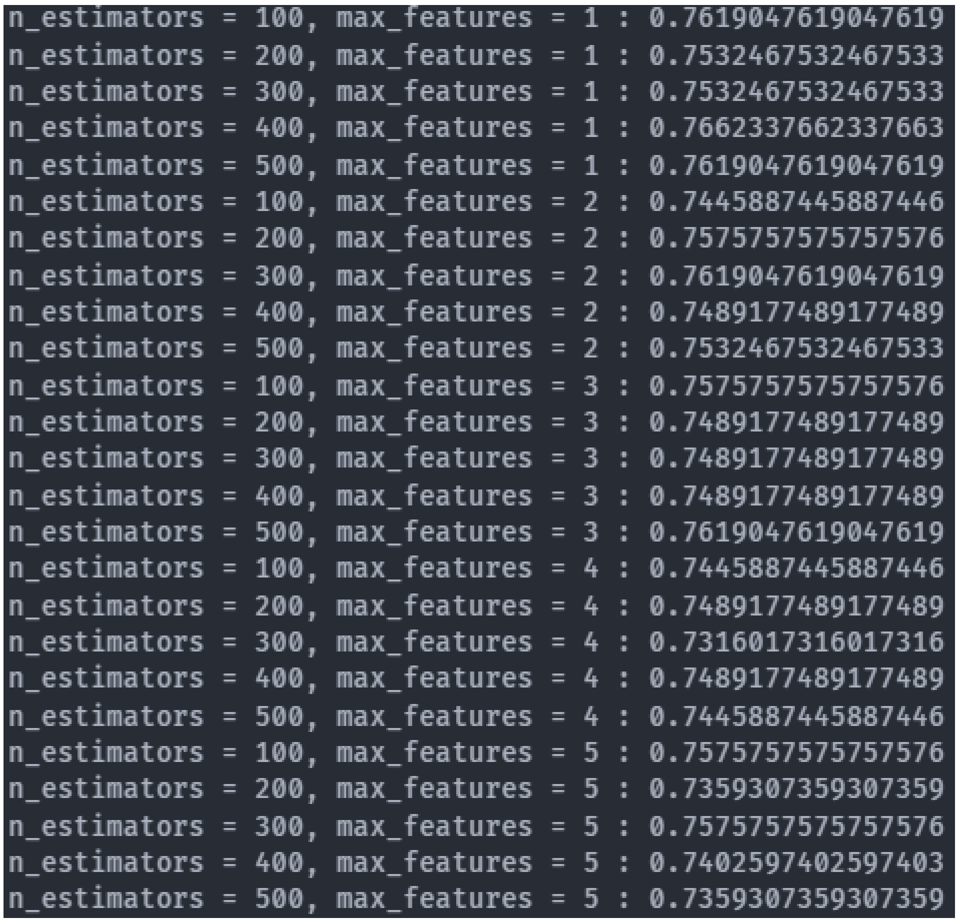
Source code :

```
estimate = [100, 200, 300, 400, 500]
max_feat = [1, 2, 3, 4, 5]

for feat in max_feat:
    for est in estimate:
        model = RandomForestClassifier(n_estimators=est,
max_features=feat)
        model.fit(train_X, train_y)
        print(f'n_estimators = {est}, max_features = {feat} :
```

```
{model.score(test_X, test_y)}')
```

실행화면 캡처:





```
n_estimators = 100, max_features = 1 : 0.7619047619047619
n_estimators = 200, max_features = 1 : 0.7532467532467533
n_estimators = 300, max_features = 1 : 0.7532467532467533
n_estimators = 400, max_features = 1 : 0.7662337662337663
n_estimators = 500, max_features = 1 : 0.7619047619047619
n_estimators = 100, max_features = 2 : 0.7445887445887446
n_estimators = 200, max_features = 2 : 0.7575757575757576
n_estimators = 300, max_features = 2 : 0.7619047619047619
n_estimators = 400, max_features = 2 : 0.7489177489177489
n_estimators = 500, max_features = 2 : 0.7532467532467533
n_estimators = 100, max_features = 3 : 0.7575757575757576
n_estimators = 200, max_features = 3 : 0.7489177489177489
n_estimators = 300, max_features = 3 : 0.7489177489177489
n_estimators = 400, max_features = 3 : 0.7489177489177489
n_estimators = 500, max_features = 3 : 0.7619047619047619
n_estimators = 100, max_features = 4 : 0.7445887445887446
n_estimators = 200, max_features = 4 : 0.7489177489177489
n_estimators = 300, max_features = 4 : 0.7316017316017316
n_estimators = 400, max_features = 4 : 0.7489177489177489
n_estimators = 500, max_features = 4 : 0.7445887445887446
n_estimators = 100, max_features = 5 : 0.7575757575757576
n_estimators = 200, max_features = 5 : 0.7359307359307359
n_estimators = 300, max_features = 5 : 0.7575757575757576
n_estimators = 400, max_features = 5 : 0.7402597402597403
n_estimators = 500, max_features = 5 : 0.7359307359307359
```