5 주. Decision Tree, RF, SVM			
학번	32191197	이름	김채은

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

Q1 (4 점) scikit-learn 에서 제공하는 DecisionTree, RandonForest, support vector machine 알고리즘를 이용하여 **PimaIndiansDiabetes dataset** 에 대한 분류 모델을 생성하고 accuracy 를 비교하시오.

- 10-fold cross validation 을 실시하여 mean accuracy 를 비교한다
- 각 알고리즘의 hyper parameter 의 값은 default value 를 이용한다.

1) DecisionTree 모델

Source code :

```
import pandas as pd
import numpy as np

import pydot

from sklearn.tree import DecisionTreeClassifier, export_graphviz
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.ensemble import RandomForestClassifier

from sklearn.model_selection import cross_validate
from IPython.display import Image

df                                                                    =
pd.read_csv('C:/Users/82104/Desktop/deeplearning/dataset/PimaIndiansDiabetes.
csv')

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)

model = DecisionTreeClassifier(random_state=1234)
scores = cross_validate(model, train_X, train_y, cv=10)

print(np.mean(scores['test_score']))
```

실행화면 캡쳐:

```
In [30]: runcell(0, 'C:/Users/82104/.spyder-py3/temp.py')
DecisionTree mean accuracy
0.6982529699510832
```

2) RandomForest 모델

Source code :

```
import pandas as pd
import numpy as np

import pydot

from sklearn.tree import DecisionTreeClassifier, export_graphviz
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.ensemble import RandomForestClassifier

from sklearn.model_selection import cross_validate
from IPython.display import Image

df                                                                    =
pd.read_csv('C:/Users/82104/Desktop/deeplearning/dataset/PimaIndiansDi
abetes.csv')

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)

model = RandomForestClassifier(n_estimators=10, random_state=1234)
scores = cross_validate(model, train_X, train_y, cv=10)

print("Random Forest mean accuracy")
print(np.mean(scores['test_score']))
```

실행화면 캡쳐:

```
In [31]: runcell(0, 'C:/Users/82104/.spyder-py3/temp.py')
Random Forest mean accuracy
0.7504891684136967
```

3) Support Vector Machine 모델

Source code :

```
import pandas as pd
import numpy as np

import pydot

from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix

from sklearn import svm

from sklearn.model_selection import cross_validate
from IPython.display import Image

df                                                                    =
pd.read_csv('C:/Users/82104/Desktop/deeplearning/dataset/PimaIndiansDi
abetes.csv')

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)

model = svm.SVC()
scores = cross_validate(model, train_X, train_y, cv=10)

print("Support Vector Machine mean accuracy")
print(np.mean(scores['test_score']))
```

실행화면 캡쳐:

```
In [32]: runcell(0, 'C:/Users/
82104/.spyder-py3/temp.py')
Support Vector Machine mean accuracy
0.7653389238294899
```

테스트 결과 해당 문제에 대한 분류 모델의 정확도는 Support Vector Machine, Random Forest, Decision Tree 순으로 높다.

Q2. (3 점) 다음의 조건에 따라 support vector machine 알고리즘를 이용하여 PimaIndiansDiabetes dataset 에 대한 분류 모델을 생성하고 accuracy 를 비교하시오.

- hyper parameter 중 kernel 에 대해 linear, poly, rbf, sigmoid, precomputed 를 각각 테스트하여 어떤 kernel 이 가장 높은 accuracy 를 도출하는지 확인하시오.
- 10-fold cross validation 을 실시하여 mean accuracy 를 비교한다

Source code :

```
import pandas as pd
import numpy as np

import pydot

from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix

from sklearn import svm

from sklearn.model_selection import cross_validate
from IPython.display import Image

df                                                                    =
pd.read_csv('C:/Users/82104/Desktop/deeplearning/dataset/PimaIndiansDi
abetes.csv')

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)


kernels = ['linear', 'poly', 'rbf', 'sigmoid', 'precomputed']


for idx, i in enumerate(kernels):

    model = svm.SVC(kernel=i)
    if idx == 4:
        x = np.dot(train_X, train_X.T)
        scores = cross_validate(model, x, train_y, cv=10)
        print(i+": "+str(np.mean(scores['test_score'])))

    else:
        scores = cross_validate(model, train_X, train_y, cv=10)
        print(i+": "+str(np.mean(scores['test_score'])))
```

실행화면 캡쳐:

```
In [42]: runcell(0, 'C:/Users/82104/.spyder-py3/temp.py')
linear: 0.7823899371069183
poly: 0.7711390635918937
rbf: 0.7653389238294899
sigmoid: 0.504367575122292
precomputed: 0.7823899371069183
```

테스트 결과 해당 문제에 대한 정확도는 kernel 속성을 linear 또는 precomputed 로 두었을 때 가장 높다. 이어 poly, rbf, sigmoid 순으로 정확도가 높게 나타난다.

Q3. (3 점) 다음의 조건에 따라 Random Forest 알고리즘를 이용하여 PimaIndiansDiabetes dataset 에 대한 분류 모델을 생성하고 accuracy 를 비교하시오.

-다음의 hyper parameter 를 테스트 하시오

. n_estimators : 100, 200, 300, 400, 500

. max_features : 1, 2, 3, 4, 5

어떤 조합이 가장 높은 accuracy 를 도출하는지 확인하시오.

- 10-fold cross validation 을 실시하여 mean accuracy 를 비교한다

Source code :

```
import pandas as pd
import numpy as np

import pydot

from sklearn.tree import DecisionTreeClassifier, export_graphviz
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.ensemble import RandomForestClassifier

from sklearn.model_selection import cross_validate
from IPython.display import Image

df                                                                    =
pd.read_csv('C:/Users/82104/Desktop/deeplearning/dataset/PimaIndiansDi
abetes.csv')

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)


for i in range(1, 6):
    for j in range(1, 6):
        model       =       RandomForestClassifier(n_estimators=i*100,
max_features=j, random_state=1234)
        scores = cross_validate(model, train_X, train_y, cv=10)

        print("n_estimators | max_features | accuracy")
        print("{}\t\t\t            {}\t\t\t            ".format(i*100,
j)+str(np.mean(scores['test_score'])))
```

실행화면 캡쳐:

```
In [61]: runfile('C:/Users/82104/.spyder-py3/temp.py', wdir='C:/Users/82104/.spyder-py3')
n_estimators | max_features | accuracy
100            1              0.7654437456324249
n_estimators | max_features | accuracy
100            2              0.7561495457721873
n_estimators | max_features | accuracy
100            3              0.7745632424877709
n_estimators | max_features | accuracy
100            4              0.7820055904961566
n_estimators | max_features | accuracy
100            5              0.767120894479385
n_estimators | max_features | accuracy
200            1              0.7580363382250174
n_estimators | max_features | accuracy
200            2              0.7580013976240391
n_estimators | max_features | accuracy
200            3              0.7653039832285115
n_estimators | max_features | accuracy
200            4              0.7801537386443047
n_estimators | max_features | accuracy
200            5              0.7653389238294899
n_estimators | max_features | accuracy
300            1              0.7579664570230608
n_estimators | max_features | accuracy
300            2              0.776554856743536
n_estimators | max_features | accuracy
300            3              0.7653039832285115
n_estimators | max_features | accuracy
300            4              0.7745632424877708
n_estimators | max_features | accuracy
300            5              0.7708595387840671
n_estimators | max_features | accuracy
400            1              0.7671907756813416
n_estimators | max_features | accuracy
400            2              0.7709294199860237
n_estimators | max_features | accuracy
400            3              0.7690426275331936
n_estimators | max_features | accuracy
400            4              0.7689727463312369
n_estimators | max_features | accuracy
400            5              0.772711390635919
500            1              0.7653738644304682
n_estimators | max_features | accuracy
500            2              0.7691125087351502
n_estimators | max_features | accuracy
500            3              0.7653039832285116
n_estimators | max_features | accuracy
500            4              0.7708944793850454
n_estimators | max_features | accuracy
500            5              0.770859538784067
```

해당 문제에서 Random Forest 분류 모델의 속성이 n_estimators=200, max_features=4 일 때 가장 높은 정확도를 보인다.