ASSIGNMENT 3.2

on

Supervised Machine Learning Algorithms

Submitted by:

Haseebullah Shaikh (2303.KHI.DEG.015)

and

Faiza Gulzar Ahmed (2303.khi.deg.001)

Dated : 19th Apr 2023

Task 01: Implement a single classification model of your choice and try to achieve at least an 80% F1 score on the wine dataset provided by Sklearn.

Solution:

In [1]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn import datasets
from sklearn.metrics import f1_score
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

In [2]:
```
wine_x, wine_y = datasets.load_wine(return_X_y=True)
```

There are 13 feautures, I am using six of them to train the model

In [3]:
```
wine_x.shape
```

Out[3]:
```
(178, 13)
```

In [4]:
```
wine_x = wine_x[:,:6]
```

In [5]:
```
wine_x.shape
```

Out[5]:
```
(178, 6)
```

Spliting dataset into 80/20

In [6]:
```
x_train, x_test, y_train, y_test = train_test_split(wine_x, wine_y, test_size=0.2)
```

Implementing Random Forest

In [7]:
```
rfc_model = RandomForestClassifier()
rfc_model.fit(x_train, y_train)
y_pred = rfc_model.predict(x_test)

f1_score(y_test, y_pred, average="micro")
```

Out[7]: 0.8888888888888888

Implementing K means

In [8]:
```
km_model = KNeighborsClassifier()
km_model.fit(x_train, y_train)
y_pred = km_model.predict(x_test)
f1_score(y_test, y_pred, average="micro")
```

Out[8]: 0.6388888888888888

Since K means is giving less f1 score than 80%, training the model again after scaling.

In [9]:
```
scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)
```

In [10]:
```
km_model = KNeighborsClassifier()
km_model.fit(x_train, y_train)
y_pred = km_model.predict(x_test)
f1_score(y_test, y_pred, average="micro")
```

Out[10]: 0.8611111111111112

Random forest is still giving good results than K means :)