```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
from sklearn.datasets import load_digits
```

```
data1=load_digits()
```

```
data1
```

```
{'data': array([[ 0.,  0.,  5., ...,  0.,  0.,  0.],
        [ 0.,  0.,  0., ..., 10.,  0.,  0.],
        [ 0.,  0.,  0., ..., 16.,  9.,  0.],
        ...,
        [ 0.,  0.,  1., ...,  6.,  0.,  0.],
        [ 0.,  0.,  2., ..., 12.,  0.,  0.],
        [ 0.,  0., 10., ..., 12.,  1.,  0.]]),
 'target': array([0, 1, 2, ..., 8, 9, 8]),
 'frame': None,
 'feature_names': ['pixel_0_0',
  'pixel_0_1',
  'pixel_0_2',
  'pixel_0_3',
  'pixel_0_4',
  'pixel_0_5',
  'pixel_0_6',
  'pixel_0_7',
  'pixel_1_0',
  'pixel_1_1',
  'pixel_1_2',
  'pixel_1_3',
  'pixel_1_4',
  'pixel_1_5',
  'pixel_1_6',
  'pixel_1_7',
  'pixel_2_0',
  'pixel_2_1',
  'pixel_2_2',
  'pixel_2_3',
  'pixel_2_4',
  'pixel_2_5',
  'pixel_2_6',
  'pixel_2_7',
  'pixel_3_0',
  'pixel_3_1',
  'pixel_3_2',
  'pixel_3_3',
  'pixel_3_4',
  'pixel_3_5',
  'pixel_3_6',
  'pixel_3_7',
  'pixel_4_0',
  'pixel_4_1',
  'pixel_4_2',
  'pixel_4_3',
  'pixel_4_4',
  'pixel_4_5',
  'pixel_4_6',
  'pixel_4_7',
  'pixel_5_0',
  'pixel_5_1',
  'pixel_5_2',
  'pixel_5_3',
  'pixel_5_4',
  'pixel_5_5',
  'pixel_5_6',
  'pixel_5_7',
  'pixel_6_0',
```

```
data1.data
```

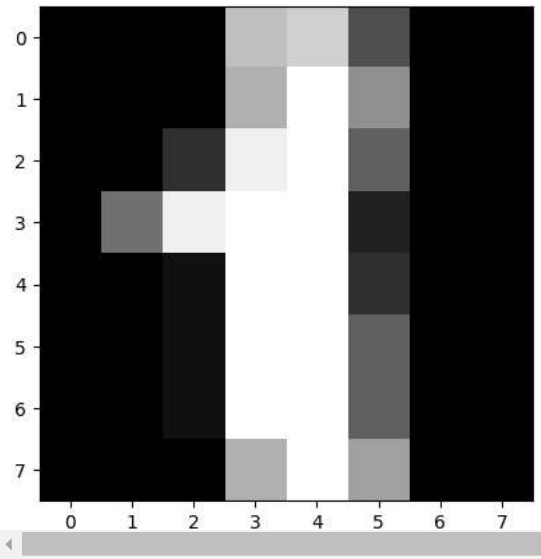
```
array([[ 0.,  0.,  5., ...,  0.,  0.,  0.],
       [ 0.,  0.,  0., ..., 10.,  0.,  0.],
       [ 0.,  0.,  0., ..., 16.,  9.,  0.],
       ...,
       [ 0.,  0.,  1., ...,  6.,  0.,  0.],
       [ 0.,  0.,  2., ..., 12.,  0.,  0.],
       [ 0.,  0., 10., ..., 12.,  1.,  0.]])
```

```
data1.target
```

```
array([0, 1, 2, ..., 8, 9, 8])
```

```
plt.imshow(data1.images[1],cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x7c71a1af4c70>
```




```
data2=pd.DataFrame(data1.data)
data2
```

	0	1	2	3	4	5	6	7	8	9	...	54	55	56	57	58	59	60	61	62	63
0	0.0	0.0	5.0	13.0	9.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	6.0	13.0	10.0	0.0	0.0	0.0
1	0.0	0.0	0.0	12.0	13.0	5.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	11.0	16.0	10.0	0.0	0.0
2	0.0	0.0	0.0	4.0	15.0	12.0	0.0	0.0	0.0	0.0	...	5.0	0.0	0.0	0.0	0.0	3.0	11.0	16.0	9.0	0.0
3	0.0	0.0	7.0	15.0	13.0	1.0	0.0	0.0	0.0	8.0	...	9.0	0.0	0.0	0.0	7.0	13.0	13.0	9.0	0.0	0.0
4	0.0	0.0	0.0	1.0	11.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	2.0	16.0	4.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1792	0.0	0.0	4.0	10.0	13.0	6.0	0.0	0.0	0.0	1.0	...	4.0	0.0	0.0	0.0	2.0	14.0	15.0	9.0	0.0	0.0
1793	0.0	0.0	6.0	16.0	13.0	11.0	1.0	0.0	0.0	0.0	...	1.0	0.0	0.0	0.0	6.0	16.0	14.0	6.0	0.0	0.0
1794	0.0	0.0	1.0	11.0	15.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	2.0	9.0	13.0	6.0	0.0	0.0
1795	0.0	0.0	2.0	10.0	7.0	0.0	0.0	0.0	0.0	0.0	...	2.0	0.0	0.0	0.0	5.0	12.0	16.0	12.0	0.0	0.0
1796	0.0	0.0	10.0	14.0	8.0	1.0	0.0	0.0	0.0	2.0	...	8.0	0.0	0.0	1.0	8.0	12.0	14.0	12.0	1.0	0.0

1797 rows × 64 columns

```
y= pd.DataFrame(data1.target,columns=['target'])
y
```

	target
0	0
1	1
2	2
3	3
4	4
...	...
1792	9
1793	0
1794	8
1795	9
1796	8

1797 rows × 1 columns

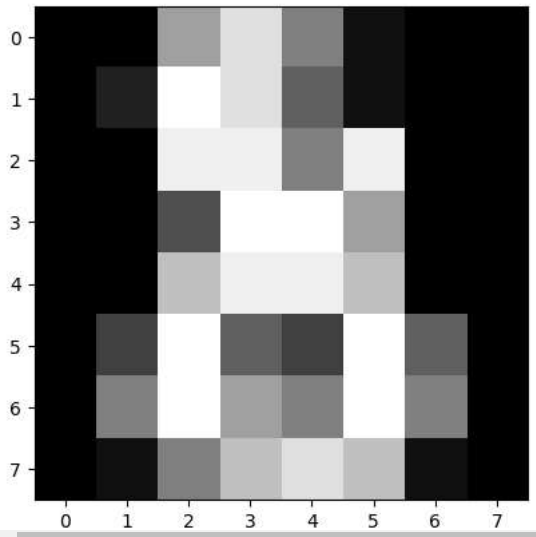
```
df=pd.concat([data2,y],axis=1)
df
```

	0	1	2	3	4	5	6	7	8	9	...	55	56	57	58	59	60	61	62	63	target
0	0.0	0.0	5.0	13.0	9.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	6.0	13.0	10.0	0.0	0.0	0.0	0
1	0.0	0.0	0.0	12.0	13.0	5.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	11.0	16.0	10.0	0.0	0.0	1
2	0.0	0.0	0.0	4.0	15.0	12.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	3.0	11.0	16.0	9.0	0.0	2
3	0.0	0.0	7.0	15.0	13.0	1.0	0.0	0.0	0.0	8.0	...	0.0	0.0	0.0	7.0	13.0	13.0	9.0	0.0	0.0	3
4	0.0	0.0	0.0	1.0	11.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	2.0	16.0	4.0	0.0	0.0	4
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1792	0.0	0.0	4.0	10.0	13.0	6.0	0.0	0.0	0.0	1.0	...	0.0	0.0	0.0	2.0	14.0	15.0	9.0	0.0	0.0	9
1793	0.0	0.0	6.0	16.0	13.0	11.0	1.0	0.0	0.0	0.0	...	0.0	0.0	0.0	6.0	16.0	14.0	6.0	0.0	0.0	0
1794	0.0	0.0	1.0	11.0	15.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	2.0	9.0	13.0	6.0	0.0	0.0	8
1795	0.0	0.0	2.0	10.0	7.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	5.0	12.0	16.0	12.0	0.0	0.0	9
1796	0.0	0.0	10.0	14.0	8.0	1.0	0.0	0.0	0.0	2.0	...	0.0	0.0	1.0	8.0	12.0	14.0	12.0	1.0	0.0	8

1797 rows × 65 columns

```
plt.imshow(data1.images[1796],cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x7c719fa90f70>
```



```
x=data1.data
y=data1.target
```

```
from sklearn.preprocessing import StandardScaler
```

```
sc=StandardScaler()
```

```
x_scaled=sc.fit_transform(x)
```

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
```

```
lda=LinearDiscriminantAnalysis()
```

```
x_lda=lda.fit_transform(x_scaled,y)
```

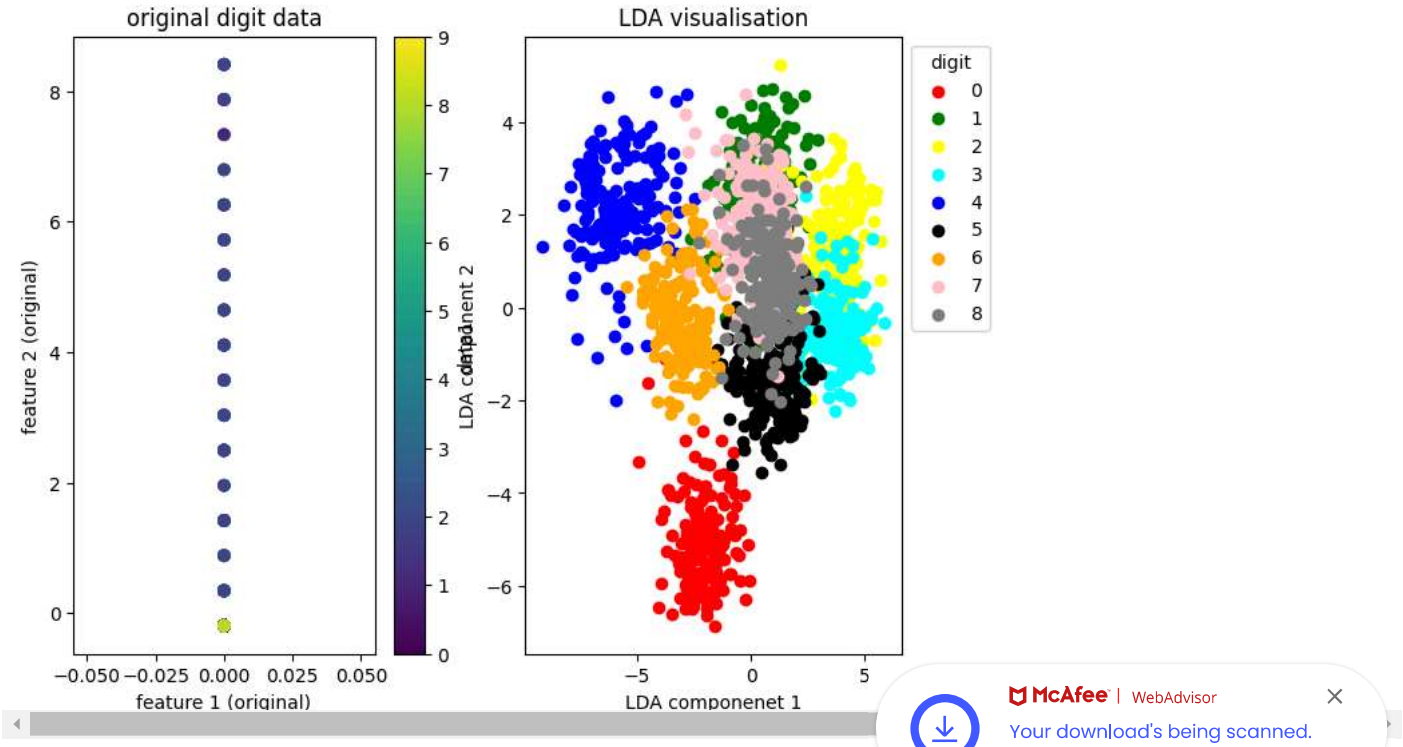
```
x_lda.shape
```

```
(1797, 9)
```

```
plt.figure(figsize=(8,6))
plt.subplot(1,2,1)
plt.scatter(x_scaled[:,0],x_scaled[:,-1],c=y,cmap='viridis')
plt.colorbar(label='data1')
plt.xlabel('feature 1 (original)')
plt.ylabel('feature 2 (original)')
plt.title('original digit data')

plt.subplot(1,2,2)
colors=['red','green','yellow','cyan','blue','black','orange','pink','gray']
for target_value,color in zip(np.unique(y),colors):
  plt.scatter(x_lda[y==target_value,0],x_lda[y==target_value,1],color=color,label=str(target_value))
plt.xlabel('LDA componenet 1')
plt.ylabel('LDA component 2')
plt.title('LDA visualisation')
plt.legend(title='digit',bbox_to_anchor=(1,1),loc='upper left')
```

```
<matplotlib.legend.Legend at 0x7c719f079600>
```




```
from sklearn.model_selection import train_test_split
```

```
xtrain,xtest,ytrain,ytest=train_test_split(x_lda,y,test_size=0.20,random_state=999)
```

```
from sklearn.ensemble import RandomForestClassifier
```

```
rfc=RandomForestClassifier(n_estimators=100)
```

```
rfc.fit(xtrain,ytrain)
```

```
▾ RandomForestClassifier
RandomForestClassifier()
```

```
y_pred=rfc.predict(xtest)
```

```
from sklearn.metrics import *
```

```
accuracy_score(ytest,y_pred)
```

```
0.95
```

```
confusion_matrix(ytest,y_pred)
```

```
array([[39,  0,  0,  0,  1,  0,  0,  0,  0,  0],
       [ 0, 34,  0,  0,  0,  0,  1,  0,  0,  0],
       [ 0,  1, 44,  0,  0,  0,  0,  0,  0,  0],
       [ 0,  0,  1, 30,  0,  0,  0,  0,  2,  2],
       [ 0,  0,  0,  0, 45,  0,  0,  0,  0,  0],
       [ 0,  0,  0,  1,  0, 25,  0,  0,  0,  1],
       [ 0,  0,  1,  0,  0,  0, 34,  0,  0,  0],
       [ 0,  0,  0,  0,  0,  0,  0, 27,  0,  1],
       [ 0,  2,  0,  0,  0,  0,  0,  0, 25,  0],
       [ 0,  0,  0,  2,  0,  1,  0,  0,  1, 39]])
```

```
from sklearn import datasets
```

```
digit=datasets.load_digits
```

```
Start coding or generate with AI.
```