In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
df = pd.read_csv("train.csv")
```

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

Out[3]:
```
(42000, 785)
```

In [4]:
```
df.head()
```

Out[4]:

	label	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	...	pixel774	pixe
0	1	0	0	0	0	0	0	0	0	0	...	0	
1	0	0	0	0	0	0	0	0	0	0	...	0	
2	1	0	0	0	0	0	0	0	0	0	...	0	
3	4	0	0	0	0	0	0	0	0	0	...	0	
4	0	0	0	0	0	0	0	0	0	0	...	0	

5 rows × 785 columns

In [5]:
```
df["label"].value_counts()
```

Out[5]:
```
1    4684
7    4401
3    4351
9    4188
2    4177
6    4137
0    4132
4    4072
8    4063
5    3795
Name: label, dtype: int64
```

In [6]:
```
import matplotlib.pyplot as plt
```

In [7]:
```
df.sample()
```

Out[7]:

	label	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	...	pixel774	pi
274	7	0	0	0	0	0	0	0	0	0	...	0	

1 rows × 785 columns

In [8]:

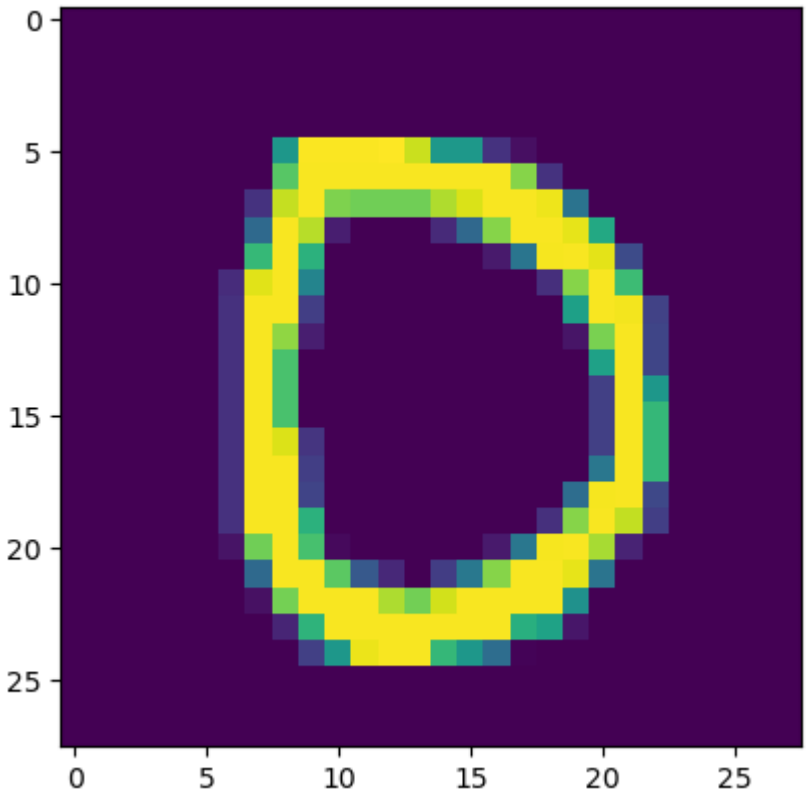
```
df.iloc[274,1:].values.reshape(28,28)
```

```
Out[8]: array([[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,  23,  79,  92, 216, 165, 141, 204, 178,
                141, 241, 255, 178, 229, 253, 126,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,  38, 234, 252, 253, 252, 252, 252, 253, 252,
                252, 252, 253, 252, 252, 252, 244, 131,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,  57, 252, 252, 253, 252, 252, 252, 253, 252,
                214, 252, 253, 252, 252, 252, 253, 196,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,  57, 252, 202, 128,  28,  28,  28,  28,  28,
                 15,  28,  28,  28, 203, 252, 253,  96,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0,   0,  32, 229, 253, 226,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0,   0,  57, 252, 252, 200,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0,  23, 209, 252, 252,  76,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0,  79, 252, 252, 151,   0,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0, 204, 253, 253, 128,   0,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,  76, 253, 252, 252,  28,   0,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0, 113, 253, 252, 148,   6,   0,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0, 163, 253, 252,  56,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                 10, 229, 254, 222,  25,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0],
                [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
```

```
 160, 252, 253, 171,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
 197, 252, 244,  56,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
 197, 252, 225,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
 198, 253, 254,  97,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
 197, 252, 253, 221,  25,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
 197, 252, 253, 196,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
 147, 252, 241,  59,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0]], dtype=int64)
```

In [11]:
```
1 plt.imshow(df.iloc[260,1:].values.reshape(28,28))
```

Out[11]: <matplotlib.image.AxesImage at 0x293469d8e10>



In [ ]:
```
1 42000*784
```

In [ ]:
```
1 42000*200
```

In [12]:
```
X = df.iloc[:,1:]
y = df.iloc[:,0]
```

In [13]:
```
from sklearn.model_selection import train_test_split
```

In [14]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.2,
```

In [15]:
```
from sklearn.preprocessing import StandardScaler
scalar = StandardScaler()
```

In [16]:
```
scalar.fit(X_train)
X_train = scalar.transform(X_train)
X_test = scalar.transform(X_test)
```

In [17]:
```
from sklearn.neighbors import KNeighborsClassifier
```

In [18]:
```
knn = KNeighborsClassifier()
```

In [19]:
```
knn.fit(X_train,y_train)
```

Out[19]:
```
KNeighborsClassifier

KNeighborsClassifier()
```

In [20]:
```
y_pred = knn.predict(X_test)
```

In [21]:
```
y_pred
```

Out[21]:
```
array([8, 1, 9, ..., 3, 0, 9], dtype=int64)
```

In [22]:
```
y_test
```

Out[22]:
```
5457     8
38509    1
25536    9
31803    9
39863    8
        ..
8388     4
29359    9
40276    3
18421    0
4335     9
Name: label, Length: 8400, dtype: int64
```

In [23]:
```
from sklearn.metrics import accuracy_score
```

In [24]:
```
accuracy_score(y_test, y_pred)
```

Out[24]: 0.9391666666666667

In [25]:
```
# Apply PCA
```

In [26]:
```
from sklearn.decomposition import PCA
```

In [27]:
```
pca = PCA(n_components=200)
```

In [28]:
```
X_train_trf = pca.fit_transform(X_train)
X_test_trf = pca.transform(X_test)
```

In [29]:
```
X_train_trf.shape
```

Out[29]: (33600, 200)

In [30]:
```
X_test.shape
```

Out[30]: (8400, 784)

In [31]:
```
knn = KNeighborsClassifier()
```

In [32]:
```
knn.fit(X_train_trf,y_train)
```

Out[32]:
▾ KNeighborsClassifier

KNeighborsClassifier()

In [33]:
```
y_pred = knn.predict(X_test_trf)
```

In [34]:
```
accuracy_score(y_test, y_pred)
```

Out[34]: 0.9502380952380952

In [35]:

```
for i in range(1,785):
    pca = PCA(n_components=i)
    X_train_trf = pca.fit_transform(X_train)
    X_test_trf = pca.transform(X_test)
    knn = KNeighborsClassifier()

    knn.fit(X_train_trf,y_train)

    y_pred = knn.predict(X_test_trf)

    print(accuracy_score(y_test, y_pred))
```

```
0.25833333333333336
0.32392857142857145
0.5107142857142857
0.6660714285714285
0.7378571428571429
0.8225
0.8438095238095238
0.8721428571428571
0.8863095238095238
0.9064285714285715
0.9120238095238096
0.9184523809523809
0.9283333333333333
0.9342857142857143
0.9388095238095238
0.9367857142857143
0.94
0.940952380952381
0.9426190476190476
```

In [37]:

```
from sklearn.decomposition import PCA

pca = PCA(n_components=None)

X_train_trf = pca.fit_transform(X_train)
X_test_trf = pca.transform(X_test)

```

In [43]:
```
list(enumerate(np.cumsum(pca.explained_variance_ratio_)))
```

```
(677, 0.9994690827787005),
(678, 0.999504270309631),
(679, 0.9995389616997464),
(680, 0.9995731289694155),
(681, 0.9996070274444568),
(682, 0.9996405498365394),
(683, 0.9996736869345986),
(684, 0.9997063544787866),
(685, 0.9997389027004104),
(686, 0.9997710516318596),
(687, 0.9998022602475863),
(688, 0.9998326025117951),
(689, 0.9998623799430802),
(690, 0.9998917295631716),
(691, 0.9999201264974633),
(692, 0.9999462093150049),
(693, 0.9999716252622665),
(694, 0.9999866516954188),
(695, 0.9999975192373468),
(696, 0.9999999999999987),
```

In [44]:
```
pca = PCA(n_components=222)

X_train_trf = pca.fit_transform(X_train)
X_test_trf = pca.transform(X_test)

X_train_trf.shape

X_test.shape

knn = KNeighborsClassifier()

knn.fit(X_train_trf,y_train)

y_pred = knn.predict(X_test_trf)

print(accuracy_score(y_test, y_pred))
```

```
0.9492857142857143
```

In [ ]:
```

```