In [1]:

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

In [2]:

```
df = pd.read_csv("train.csv")
df
```

Out[2]:

	label	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	...	pixel774
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
...	...	...	...	...	...	...	...	...	...	...	...	...
41995	0	0	0	0	0	0	0	0	0	0	...	0
41996	1	0	0	0	0	0	0	0	0	0	...	0
41997	7	0	0	0	0	0	0	0	0	0	...	0
41998	6	0	0	0	0	0	0	0	0	0	...	0
41999	9	0	0	0	0	0	0	0	0	0	...	0

42000 rows × 785 columns

In [3]:

```
df.shape
```

Out[3]:

```
(42000, 785)
```

In [4]:

```
x = df.iloc[:,1:]
y = df.iloc[:,0]
```

In [5]:

```
y
```

Out[5]:

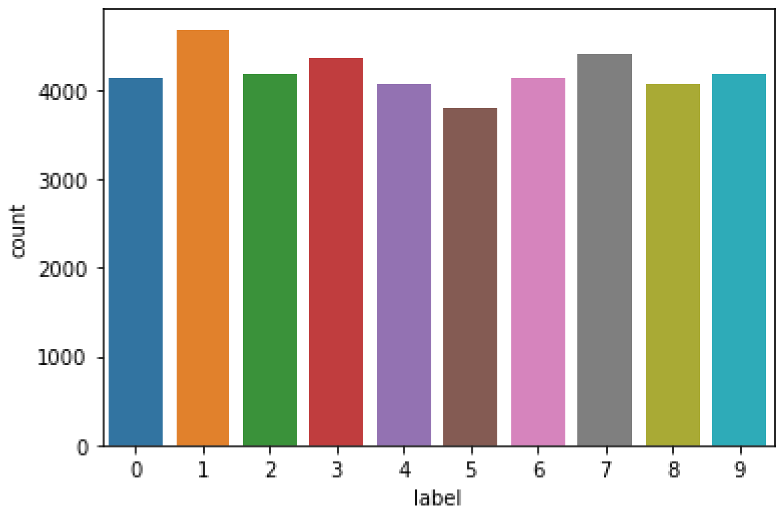
```
0        1
1        0
2        1
3        4
4        0
        ..
41995    0
41996    1
41997    7
41998    6
41999    9
Name: label, Length: 42000, dtype: int64
```

In [6]:

```
sns.countplot(data=df,x=y)
```

Out[6]:

```
<AxesSubplot:xlabel='label', ylabel='count'>
```



In [7]:

```
x.sample(1)
```

Out[7]:

	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	pixel9	...	pixel774
3652	0	0	0	0	0	0	0	0	0	0	...	0

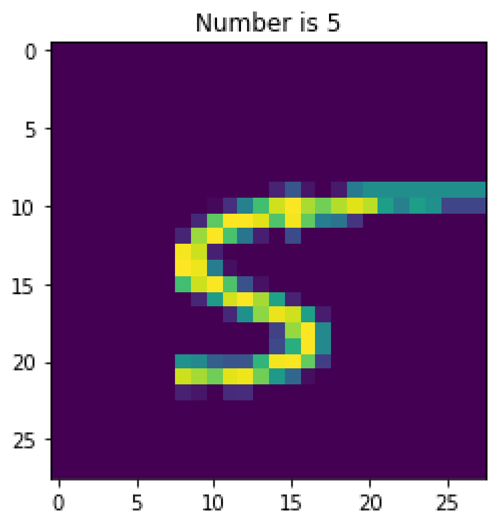
1 rows × 784 columns

In [8]:

```
plt.imshow(x.iloc[244,:].values.reshape(28,28))
plt.title(f"Number is {y[244]}")
```

Out[8]:

```
Text(0.5, 1.0, 'Number is 5')
```



In [9]:

```
x.sample()
```

Out[9]:

	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	pixel9	...	pixel774
22878	0	0	0	0	0	0	0	0	0	0	...	75

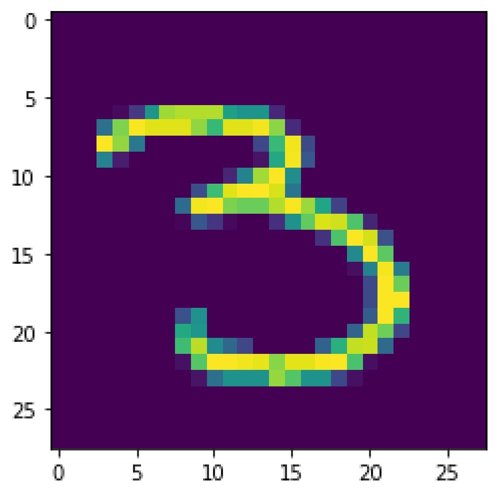
1 rows × 784 columns

In [10]:

```
plt.imshow(x.iloc[5483,:].values.reshape(28,28))
```

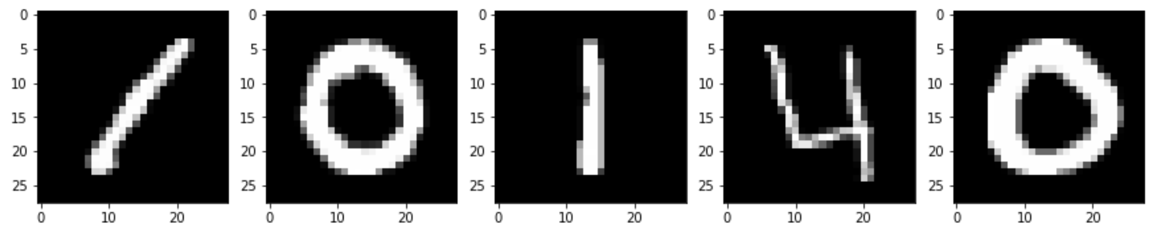
Out[10]:

```
<matplotlib.image.AxesImage at 0x223ac78feb0>
```



In [11]:

```
plt.figure(figsize=(15,10))
for i in range(5):
    plt.subplot(2,5,i+1)
    plt.imshow(x.iloc[i,:].values.reshape(28,28),cmap='gray')
plt.show()
```



In [28]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3,random_state=0)
```

In [29]:

```
from sklearn.neighbors import KNeighborsClassifier
```

In [30]:

```
knn = KNeighborsClassifier()
```

In [31]:

```
knn.fit(x_train,y_train)
```

Out[31]:

```
KNeighborsClassifier()
```

In [32]:

```
y_pred = knn.predict(x_test)
```

In [33]:

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[33]:

```
0.9657142857142857
```

In [34]:

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

In [35]:

```
std = StandardScaler()
x_train_trf = std.fit_transform(x_train)
x_test_trf = std.transform(x_test)
```

In [36]:

```
x_train_trf
```

Out[36]:

```
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.]])
```

In [37]:

```
from sklearn.decomposition import PCA
```

In [38]:

```
pca = PCA(n_components=100)
```

In [39]:

```
x_train_pca = pca.fit_transform(x_train_trf)
x_test_pca = pca.transform(x_test)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py:443: UserWarnin
g: X has feature names, but PCA was fitted without feature names
  warnings.warn(
```

In [40]:

```
x_train_trf
```

Out[40]:

```
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.]])
```

In [41]:

```
x_train_pca.shape
```

Out[41]:

```
(29400, 100)
```

In [42]:

```
knn_pca = KNeighborsClassifier()
knn_pca.fit(x_train_pca,y_train)
```

Out[42]:

```
KNeighborsClassifier()
```

In [43]:

```
y_pred_pca = knn_pca.predict(x_test_pca)
```

In [44]:

```
accuracy_score(y_test,y_pred_pca)
```

Out[44]:

```
0.7679365079365079
```

In [45]:

```
for i in range(1,785):
    pca=PCA(n_components=i)
    x_train_pca = pca.fit_transform(x_train_trf)
    x_test_pca = pca.transform(x_test)
    knn.fit(x_train_pca,y_train)
    y_pred_pca = knn.predict(x_test_pca)
    print(f"Iteration : {i} {accuracy_score(y_test,y_pred_pca)}")
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py:443: UserWarnin
g: X has feature names, but PCA was fitted without feature names
  warnings.warn(

Iteration : 1 0.18634920634920635

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py:443: UserWarnin
g: X has feature names, but PCA was fitted without feature names
  warnings.warn(

Iteration : 2 0.13523809523809524

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py:443: UserWarnin
g: X has feature names, but PCA was fitted without feature names
  warnings.warn(

Iteration : 3 0.2311111111111111

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py:443: UserWarnin
g: X has feature names, but PCA was fitted without feature names
  warnings.warn(
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Input In [45], in <cell line: 1>()
      4 x_test_pca = pca.transform(x_test)
      5 knn.fit(x_train_pca,y_train)
----> 6 y_pred_pca = knn.predict(x_test_pca)
      7 print(f"Iteration : {i} {accuracy_score(y_test,y_pred_pca)}")

File C:\ProgramData\Anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228, in KNeighborsClassifier.predict(self, X)
    226 for k, classes_k in enumerate(classes_):
    227     if weights is None:
--> 228         mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
    229     else:
    230         mode, _ = weighted_mode(_y[neigh_ind, k], weights, axis=1)

File C:\ProgramData\Anaconda3\lib\site-packages\scipy\stats\stats.py:448, in mode(a, axis, nan_policy)
    446 counts = np.empty(a_view.shape[:-1], dtype=np.int_)
    447 for ind in inds:
--> 448     modes[ind], counts[ind] = _mode1D(a_view[ind])
    449 newshape = list(a.shape)
    450 newshape[axis] = 1

File C:\ProgramData\Anaconda3\lib\site-packages\scipy\stats\stats.py:434, in mode.<locals>._mode1D(a)
    433 def _mode1D(a):
--> 434     vals, cnts = np.unique(a, return_counts=True)
    435     return vals[cnts.argmax()], cnts.max()

File <__array_function__ internals>:5, in unique(*args, **kwargs)

File C:\ProgramData\Anaconda3\lib\site-packages\numpy\lib\arraysetops.py:272, in unique(ar, return_index, return_inverse, return_counts, axis)
    270 ar = np.asanyarray(ar)
    271 if axis is None:
--> 272     ret = _unique1d(ar, return_index, return_inverse, return_counts)
    273     return _unpack_tuple(ret)
    275 # axis was specified and not None

File C:\ProgramData\Anaconda3\lib\site-packages\numpy\lib\arraysetops.py:359, in _unique1d(ar, return_index, return_inverse, return_counts)
    357     ret += (inv_idx,)
    358 if return_counts:
--> 359     idx = np.concatenate(np.nonzero(mask) + ([mask.size],))
    360     ret += (np.diff(idx),)
    361 return ret

File <__array_function__ internals>:5, in concatenate(*args, **kwargs)

KeyboardInterrupt: 
```

In [58]:

```python
pca_dim = PCA(n_components=3)
x_train_pca = pca_dim.fit_transform(x_train_trf)
x_test_pca = pca_dim.transform(x_test)
x_train_pca.shape
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py:443: UserWarning:
X has feature names, but PCA was fitted without feature names
```

Out[58]:

```
(29400, 3)
```

In [59]:

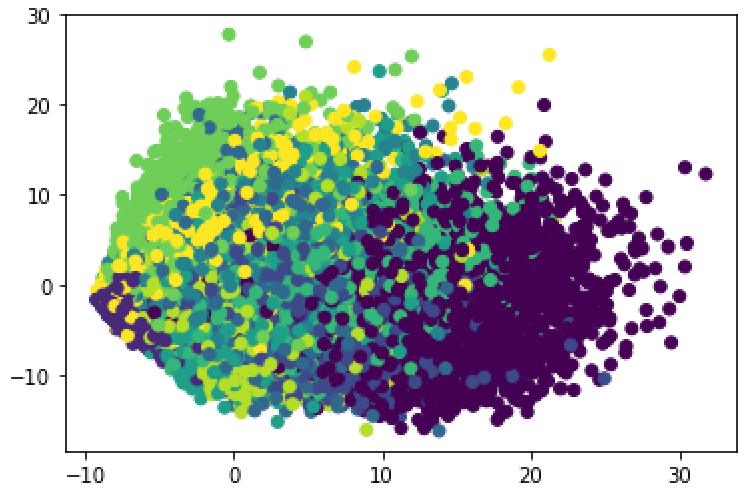
```
#pip install plotly
import plotly.express as px
px.scatter(x_train_pca[:,0],x_train_pca[:,1])
```

In [60]:

```
# import plotly
plt.scatter(x_train_pca[:,0],x_train_pca[:,1],c=y_train)
```

Out[60]:

```
<matplotlib.collections.PathCollection at 0x223dfe37d30>
```



In [61]:

```
d=y_train.astype(str)
```

In [62]:

```
import plotly.express as px
px.scatter_3d(x=x_train_pca[:,0],y=x_train_pca[:,1],z=x_train_pca[:,2],color=d)
```

In [63]:

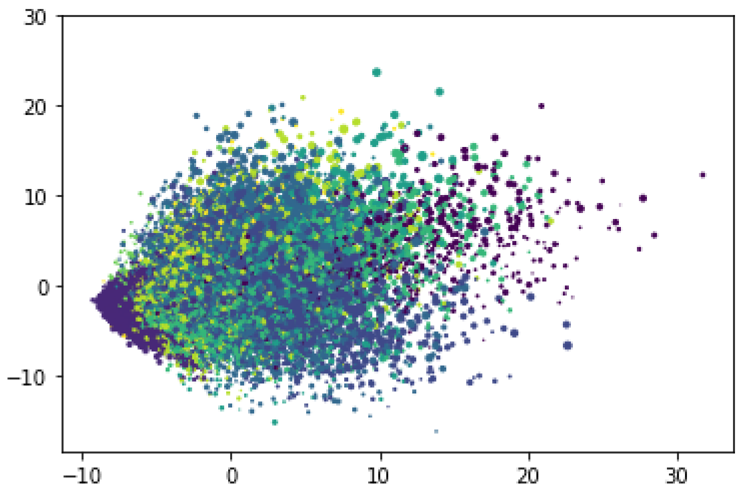
```
plt.scatter(x_train_pca[:,0],x_train_pca[:,1],x_train_pca[:,2],c=y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\matplotlib\collections.py:982:
RuntimeWarning:

invalid value encountered in sqrt
```

Out[63]:

```
<matplotlib.collections.PathCollection at 0x223dfe5ba90>
```



In [64]:

```
pca.explained_variance_
```

Out[64]:

```
array([40.59588171, 29.31939625, 26.70539897, 20.7906109 ])
```

In [65]:

```
pca.explained_variance_ratio_*100
```

Out[65]:

```
array([5.84935171, 4.22455317, 3.84790931, 2.99566336])
```

In [66]:

```
d = y_train.astype(str)
pca_dim = PCA()
x_train_pca = pca_dim.fit_transform(x_train_trf)
x_test_pca = pca_dim.transform(x_test)
pca_dim.explained_variance_ratio_*100
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py:443: UserWarn
ing:

X has feature names, but PCA was fitted without feature names
```

Out[66]:

```
array([5.84935171e+00, 4.22455317e+00, 3.84790932e+00, 2.99566411e+00,
       2.60283768e+00, 2.27062325e+00, 2.00722706e+00, 1.80402768e+00,
       1.60330442e+00, 1.45271448e+00, 1.40019450e+00, 1.25521838e+00,
       1.15606423e+00, 1.13774951e+00, 1.07853036e+00, 1.03298491e+00,
       9.70395520e-01, 9.57399383e-01, 9.34237163e-01, 9.12905801e-01,
       8.58260664e-01, 8.33084537e-01, 8.04735721e-01, 7.70710198e-01,
       7.48672901e-01, 7.13847382e-01, 7.08900817e-01, 6.86255797e-01,
       6.51284381e-01, 6.37463674e-01, 6.28489824e-01, 6.22514458e-01,
       6.00492008e-01, 5.86809038e-01, 5.80058199e-01, 5.67056254e-01,
       5.49320588e-01, 5.35830886e-01, 5.23765424e-01, 5.10018441e-01,
       4.98834932e-01, 4.93105367e-01, 4.77954113e-01, 4.67097346e-01,
       4.62488902e-01, 4.57313741e-01, 4.54632903e-01, 4.41634541e-01,
```

In [67]:

```
np.cumsum(pca.explained_variance_ratio_)
```

Out[67]:

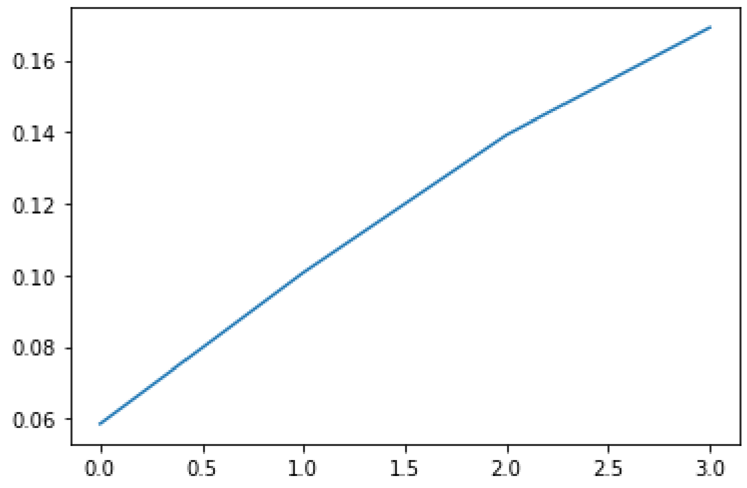
```
array([0.05849352, 0.10073905, 0.13921814, 0.16917478])
```

In [68]:

```
plt.plot(np.cumsum(pca.explained_variance_ratio_))
```

Out[68]:

```
[<matplotlib.lines.Line2D at 0x223ee23c940>]
```



In [ ]:

```

```