In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [27]:
```
df = pd.read_csv('datasets/train.csv')
df.head()
```

Out[27]:

	label	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	...	pixel774	pixel775	pixel776	pixel777	pixel778	pixel779	pixel780	pixel781	pi
0	1	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	
2	1	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	
3	4	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	

5 rows × 785 columns

In [28]:
```
df.shape
```

Out[28]:
```
(42000, 785)
```

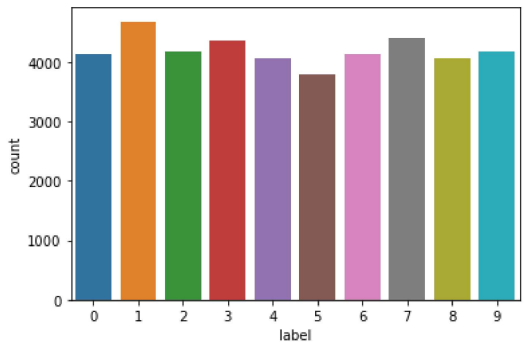
In [29]:
```
x = df.iloc[:,1:]
y = df.iloc[:,0]
```

In [30]:
```
y
```

Out[30]:
```
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 [31]:
```
sns.countplot(data=df,x=y)
```

Out[31]:
```
<AxesSubplot:xlabel='label', ylabel='count'>
```



In [32]:
```
x.sample(1)
```

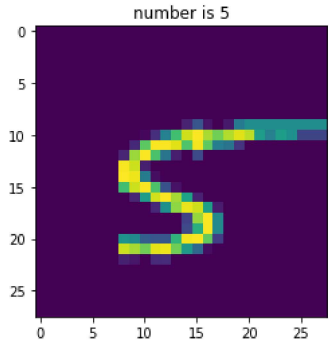
Out[32]:

	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	pixel9	...	pixel774	pixel775	pixel776	pixel777	pixel778	pixel779	pixel780	pixel78
31430	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	

1 rows × 784 columns

In [33]:
```
plt.imshow(x.iloc[244,:].values.reshape(28,28))
plt.title(f"number is {y[244]}")
```

Out[33]: Text(0.5, 1.0, 'number is 5')



In [34]:
```
x.sample()
```

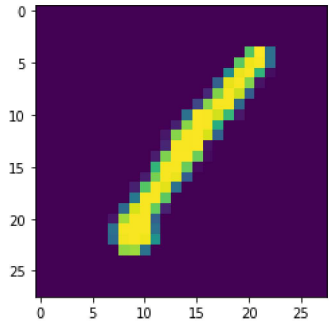
Out[34]:

	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	pixel9	...	pixel774	pixel775	pixel776	pixel777	pixel778	pixel779	pixel780	pixel78
34409	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	

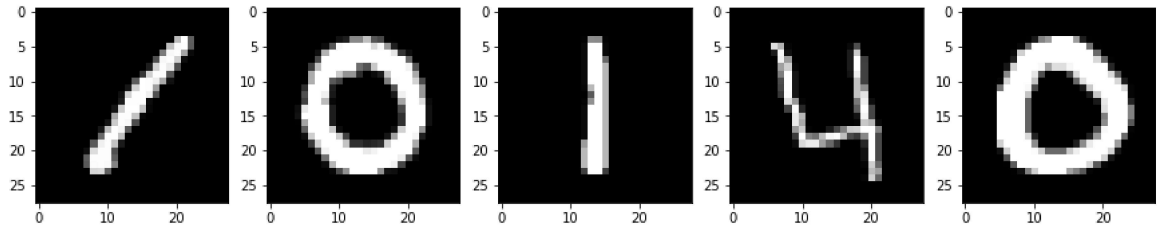
1 rows × 784 columns

In [35]:
```
plt.imshow(x.iloc[0,:].values.reshape(28,28))
```

Out[35]: <matplotlib.image.AxesImage at 0x1671c80d9d0>



In [36]:
```
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 [37]:
```
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 [38]:
```
from sklearn.neighbors import KNeighborsClassifier
```

In [39]:
```
knn = KNeighborsClassifier()
```

In [40]:
```
knn.fit(x_train,y_train)
```

Out[40]: KNeighborsClassifier()

In [41]:
```
y_pred = knn.predict(x_test)
```

In [42]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[42]:
```
0.9657142857142857
```

In [43]:
```
from sklearn.preprocessing import StandardScaler
```

In [44]:
```
std = StandardScaler()
x_train_trf = std.fit_transform(x_train)
x_test_trf = std.transform(x_test)
```

In [45]:
```
x_train_trf
```

Out[45]:
```
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 [46]:
```
from sklearn.decomposition import PCA
```

In [47]:
```
pca = PCA(n_components=100)
```

In [48]:
```
x_train_pca = pca.fit_transform(x_train_trf)
x_test_pca = pca.transform(x_test)
```

```
C:\Users\User15\anaconda3\lib\site-packages\sklearn\base.py:443: UserWarning: X has feature names, but PCA was fitted without f
eature names
  warnings.warn(
```

In [49]:
```
x_train_pca.shape
```

Out[49]:
```
(29400, 100)
```

In [50]:
```
knn_pca = KNeighborsClassifier()
knn_pca.fit(x_train_pca,y_train)
```

Out[50]:
```
KNeighborsClassifier()
```

In [52]:
```
y_pred_pca = knn_pca.predict(x_test_pca)
```

In [53]:
```
accuracy_score(y_test,y_pred_pca)
```

Out[53]:
```
0.768015873015873
```

In [54]:
```
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)}")
```

```
  warnings.warn(

Iteration : 7 0.43103174603174604

C:\Users\User15\anaconda3\lib\site-packages\sklearn\base.py:443: UserWarning: X has feature names, but PCA was fitted without
feature names
  warnings.warn(

Iteration : 8 0.4707142857142857

C:\Users\User15\anaconda3\lib\site-packages\sklearn\base.py:443: UserWarning: X has feature names, but PCA was fitted without
feature names
  warnings.warn(

Iteration : 9 0.4719047619047619

C:\Users\User15\anaconda3\lib\site-packages\sklearn\base.py:443: UserWarning: X has feature names, but PCA was fitted without
feature names
  warnings.warn(

Iteration : 10 0.4638095238095238
```

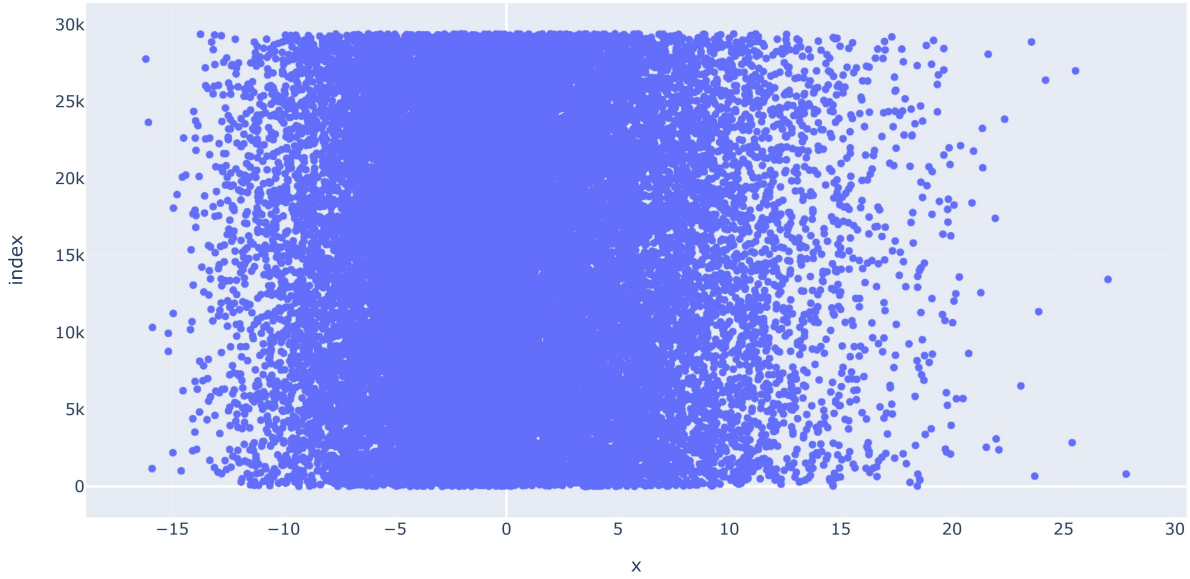
In [63]:
```
pca_dim= PCA(n_components=3)
x_train_pca = pca_dim.fit_transform(x_train_trf)
x_test_pca = pca_dim.transform(x_test)
x_train_pca.shape
```

```
C:\Users\User15\anaconda3\lib\site-packages\sklearn\base.py:443: UserWarning:

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

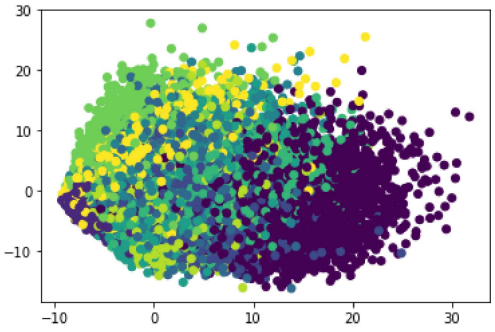
Out[63]: (29400, 3)

In [64]:
```
# !pip install plotly
import plotly.express as px
px.scatter(x_train_pca[:,0],x_train_pca[:,1])
```



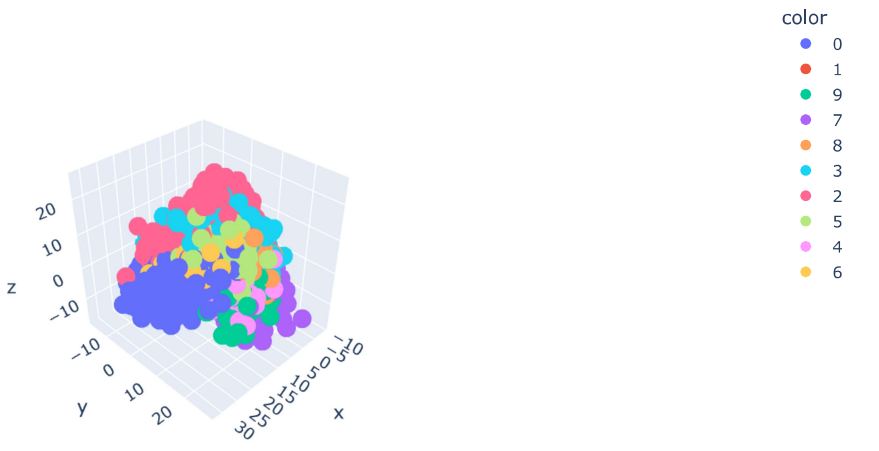
In [65]:
```
#import plotly.express as px
plt.scatter(x_train_pca[:,0],x_train_pca[:,1],c=y_train)
```

Out[65]: <matplotlib.collections.PathCollection at 0x1675d322640>



In [66]:
```
d=y_train.astype(str)
```

In [67]:
```
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)
```

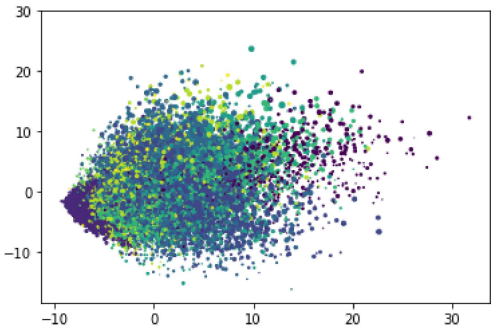


In [68]:
```
plt.scatter(x_train_pca[:,0],x_train_pca[:,1],x_train_pca[:,2],c=y_train)
```

```
C:\Users\User15\anaconda3\lib\site-packages\matplotlib\collections.py:982: RuntimeWarning:

invalid value encountered in sqrt
```

Out[68]:
```
<matplotlib.collections.PathCollection at 0x1675b3d3460>
```



In [69]:
```
pca.explained_variance_
```

Out[69]:
```
array([40.59588171, 29.31939629, 26.705399  , 20.79061603, 18.06430776,
       15.75866099, 13.93062652, 12.52034899, 11.12707722, 10.08205158,
        9.71636769,  8.71101004,  8.02118987])
```

In [71]:
```
pca.components_.shape
```

Out[71]:
```
(13, 784)
```

In [70]:
```
pca.explained_variance_ratio_*100
```

Out[70]:
```
array([5.84935171, 4.22455317, 3.84790932, 2.9956641 , 2.60283766,
       2.2706232 , 2.00722661, 1.8040235 , 1.60327071, 1.45269577,
       1.40000536, 1.25514607, 1.15575173])
```

In [72]:
```
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
```

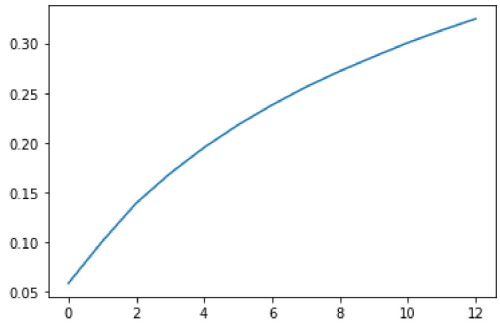
```
       4.32328909e-02, 4.28325503e-02, 4.24215749e-02, 4.23117403e-02,
       4.17064664e-02, 4.16260548e-02, 4.11475248e-02, 4.09359363e-02,
       4.05625845e-02, 4.01519850e-02, 3.95911497e-02, 3.94476652e-02,
       3.92191366e-02, 3.86050492e-02, 3.84249733e-02, 3.80918035e-02,
       3.77628757e-02, 3.71936334e-02, 3.69879309e-02, 3.68579897e-02,
       3.64615502e-02, 3.63987922e-02, 3.61405266e-02, 3.57138679e-02,
       3.53294409e-02, 3.47620994e-02, 3.46146215e-02, 3.43886661e-02,
       3.42368515e-02, 3.40180210e-02, 3.36906914e-02, 3.33562580e-02,
       3.31623982e-02, 3.31271853e-02, 3.28604615e-02, 3.26594121e-02,
       3.25167883e-02, 3.23123427e-02, 3.19364628e-02, 3.19172888e-02,
       3.14522092e-02, 3.11163581e-02, 3.09712565e-02, 3.07832778e-02,
       3.06647068e-02, 3.03927400e-02, 3.00391359e-02, 2.99056593e-02,
       2.97362867e-02, 2.94108538e-02, 2.92677622e-02, 2.91480363e-02,
       2.90066592e-02, 2.87276254e-02, 2.85630711e-02, 2.82133520e-02,
       2.79770121e-02, 2.78266770e-02, 2.76401346e-02, 2.73711712e-02,
       2.72352578e-02, 2.66693490e-02, 2.65974412e-02, 2.62724506e-02,
       2.61508241e-02, 2.58354688e-02, 2.56189813e-02, 2.55608242e-02,
       2.53986565e-02, 2.51836673e-02, 2.51626387e-02, 2.48756948e-02,
       2.46801735e-02, 2.45245248e-02, 2.43535391e-02, 2.42756781e-02,
       2.41080800e-02, 2.38236384e-02, 2.36396748e-02, 2.35774771e-02,
```

In [73]:
```
np.cumsum(pca.explained_variance_ratio_)
```

Out[73]:
```
array([0.05849352, 0.10073905, 0.13921814, 0.16917478, 0.19520316,
       0.21790939, 0.23798166, 0.25602189, 0.2720546 , 0.28658156,
       0.30058161, 0.31313307, 0.32469059])
```

In [74]:
```
plt.plot(np.cumsum(pca.explained_variance_ratio_))
```

Out[74]:
```
[<matplotlib.lines.Line2D at 0x16767469c70>]
```



In [ ]:
```

```