1.Import Libraries

```
from keras import datasets
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.utils import to_categorical
from keras.layers import Dense,Flatten,MaxPooling2D,Conv2D,Dropout
from keras.models import Sequential
```

2.Import Datasets

```
(x_train, y_train), (x_test, y_test)=tf.keras.datasets.fashion_mnist.load_data() # Load Data
```

3.Data Undestanding

3.1 Check shape of train and test data

```
x_train.shape,y_train.shape,x_test.shape,y_test.shape   #check shape of input and output data
```

```
((60000, 28, 28), (60000,), (10000, 28, 28), (10000,))
```

```
x_train.std(),x_test.std()
```

```
(90.02118235130519, 89.87325907809718)
```

3.2 Normlize Input Datasets

```
x_train=x_train/255
x_test=x_test/255   # Normlize input Data
```

```
x_train=np.pad(x_train,pad_width=((0,0),(2,2),(2,2)))
x_test=np.pad(x_test,pad_width=((0,0),(2,2),(2,2)))   #Add Pad to all dimention
```

```
x_train.shape,x_test.shape
```

```
((60000, 32, 32), (10000, 32, 32))
```

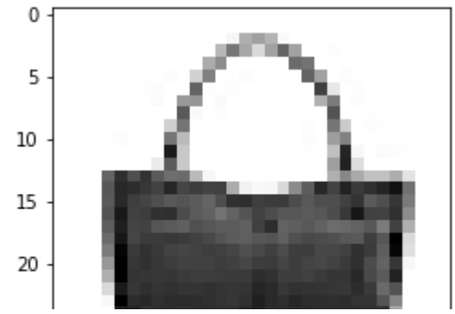
3.3 Reshape input Datasets

```
x_train_reshape=np.reshape(x_train,newshape=(60000,32,32,1))
x_test_reshape=np.reshape(x_test,newshape=(10000,32,32,1))
```

4.Data Testing

```
plt.imshow(x_train[100],cmap="Greys")
```

```
<matplotlib.image.AxesImage at 0x7f4b29679150>
```



```
y_train[100]
```

```
8
```

4.Add catagorical to output side to convert in catagory

```
y_train_encoder=to_categorical(y_train)
y_test_encoder=to_categorical(y_test)
```

5.Model Building

```
model=Sequential()
model.add(Conv2D(input_shape=(32,32,1),strides=1,filters=16, kernel_size=(3,3), padding='vali
model.add(MaxPooling2D(pool_size=(2,2),strides=2))
model.add(Conv2D(strides=1,filters=32, kernel_size=(3,3), padding='valid',kernel_initializer=
model.add(MaxPooling2D(pool_size=(2,2),strides=2))
model.add(Conv2D(strides=1,filters=32, kernel_size=(3,3), padding='valid',kernel_initializer=

model.add(Flatten())
model.add(Dense(100,activation='relu'))
model.add(Dense(40,activation='relu'))
model.add(Dense(10,activation='softmax'))
model.summary()
```

```
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_10 (Conv2D)          (None, 30, 30, 16)        160

 max_pooling2d_7 (MaxPooling  (None, 15, 15, 16)       0
 2D)

 conv2d_11 (Conv2D)          (None, 13, 13, 32)        4640

 max_pooling2d_8 (MaxPooling  (None, 6, 6, 32)         0
 2D)

 conv2d_12 (Conv2D)          (None, 4, 4, 32)          9248

 flatten_1 (Flatten)         (None, 512)               0

 dense_3 (Dense)             (None, 100)               51300

 dense_4 (Dense)             (None, 40)                4040

 dense_5 (Dense)             (None, 10)                410

=================================================================
Total params: 69,798
Trainable params: 69,798
Non-trainable params: 0
_________________________________________________________________
```

6.Model Compiling

```
model.compile(optimizer='adam',metrics="categorical_accuracy",loss='categorical_crossentropy'
```

7.Fit the Model

```
model_training=model.fit(x=x_train_reshape, y=y_train_encoder,batch_size=32,epochs=10,verbose
```

```
Epoch 1/10
1875/1875 [==============================] - 23s 6ms/step - loss: 0.5456 - categorical_a
Epoch 2/10
1875/1875 [==============================] - 11s 6ms/step - loss: 0.3349 - categorical_a
Epoch 3/10
1875/1875 [==============================] - 11s 6ms/step - loss: 0.2855 - categorical_a
Epoch 4/10
1875/1875 [==============================] - 11s 6ms/step - loss: 0.2569 - categorical_a
Epoch 5/10
1875/1875 [==============================] - 12s 6ms/step - loss: 0.2352 - categorical_a
Epoch 6/10
1875/1875 [==============================] - 12s 6ms/step - loss: 0.2171 - categorical_a
Epoch 7/10
1875/1875 [==============================] - 11s 6ms/step - loss: 0.2016 - categorical_a
Epoch 8/10
1875/1875 [==============================] - 11s 6ms/step - loss: 0.1890 - categorical_a
Epoch 9/10
1875/1875 [==============================] - 11s 6ms/step - loss: 0.1750 - categorical_a
Epoch 10/10
1875/1875 [==============================] - 11s 6ms/step - loss: 0.1650 - categorical_a
```

8. Evaluate The Model

```
model.evaluate(x_test_reshape,y_test_encoder)  # Check Accuracy of the model
```

```
313/313 [==============================] - 1s 4ms/step - loss: 0.2578 - categorical_accu
[0.2578224241733551, 0.9082000255584717]
```

9.Model Testing

```
y_test_pred=model.predict(x_test)
```

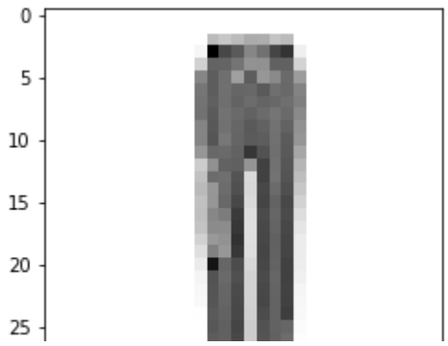
Example 1

```
class_name=['T-shirt/top',"Trouser","Pullover","Dress","Coat","Sandal",'Shirt','Sneaker','Bag
```

```
#We are taking example 100th data of x_train 100 image is showing bag
```

Example 1

```
plt.imshow(x_train[1000],cmap='Greys')
plt.show()
```

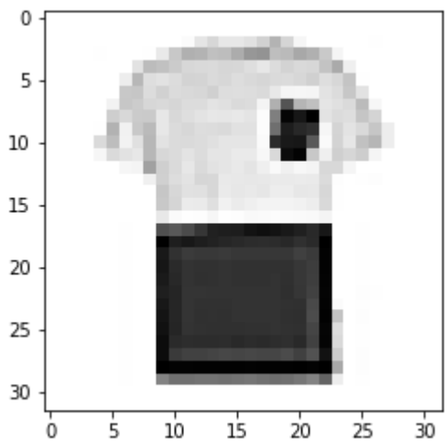



```
class_name[y_train[1000]]     #Prediction
```

```
'Trouser'
```

Example 2

```
plt.imshow(x_test[120],cmap="Greys")
plt.show()
```




```
class_name[np.argmax(y_test_pred[120])]
```

```
'T-shirt/top'
```