In [1]:

```
# Importing Packages
import numpy as np
from matplotlib import pyplot
import keras
import tensorflow as tf
from keras.models import Sequential
from keras.layers import Flatten, Dense
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:

```
(X_train, y_train), (X_test, y_test) = tf.keras.datasets.fashion_mnist.load_data()
X_train.shape, y_train.shape, "____" , X_test.shape, y_test.shape
((60000, 28, 28), (60000,), '____', (10000, 28, 28), (10000,))
X_train[0]
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-labels-idx1-ubyte.gz
29515/29515 [==============================] - 0s 2us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-images-idx3-ubyte.gz
26421880/26421880 [==============================] - 32s 1us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-labels-idx1-ubyte.gz
5148/5148 [==============================] - 0s 0s/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-images-idx3-ubyte.gz
4422102/4422102 [==============================] - 3s 1us/step
```

```
         206, 115],
       [  0, 122, 219, 193, 179, 171, 183, 196, 204, 210, 213, 207, 211,
         210, 200, 196, 194, 191, 195, 191, 198, 192, 176, 156, 167, 177,
         210,  92],
       [  0,   0,  74, 189, 212, 191, 175, 172, 175, 181, 185, 188, 189,
         188, 193, 198, 204, 209, 210, 210, 211, 188, 188, 194, 192, 216,
         170,   0],
       [  2,   0,   0,   0,  66, 200, 222, 237, 239, 242, 246, 243, 244,
         221, 220, 193, 191, 179, 182, 182, 181, 176, 166, 168,  99,  58,
           0,   0],
       [  0,   0,   0,   0,   0,   0,   0,  40,  61,  44,  72,  41,  35,
           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]], dtype=uint8)
```

In [3]:

```
class_labels = ["Trouser", "Pullover" , "Dress" , "Coat" , "Sandal" , "Shirt" , "Sneaker",]
class_labels
```

Out[3]:

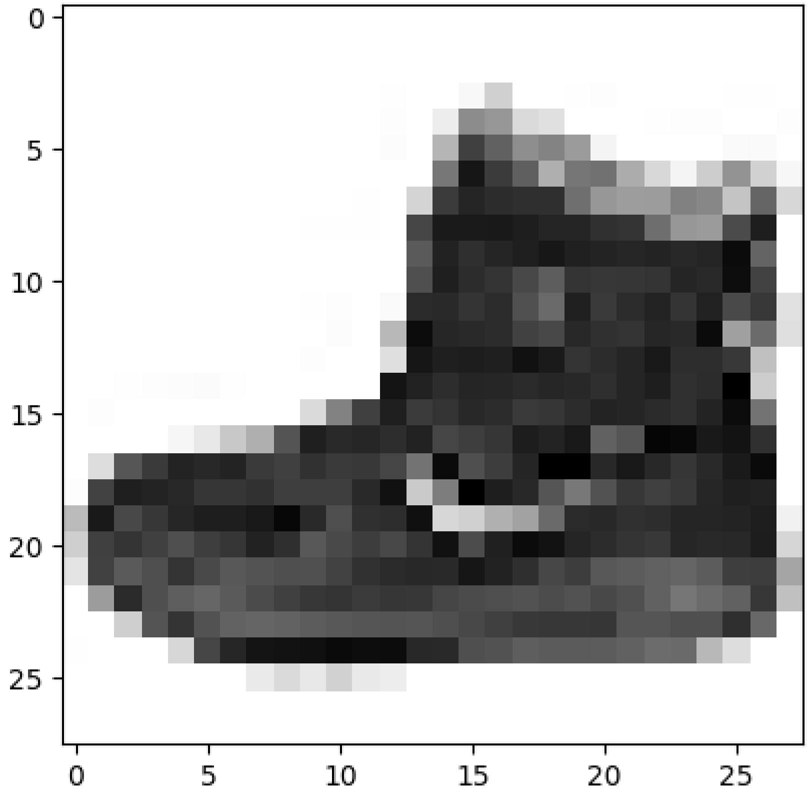
```
['Trouser', 'Pullover', 'Dress', 'Coat', 'Sandal', 'Shirt', 'Sneaker']
```

In [4]:

```
plt.imshow(X_train[0] , cmap="Greys")
```

Out[4]:

```
<matplotlib.image.AxesImage at 0x1f63152b9d0>
```



In [5]:

```
X_train. shape
```

Out[5]:

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

In [6]:

```
plt.figure(figsize=(16,16))
j=1
for i in np.random.randint(0,1000,25):
  plt.subplot(5,5,j)
  j+=1
  plt.imshow(X_train[i],cmap="Greys")
  plt.axis('off')
plt.title('{} / {}'.format(class_labels[y_train[i]], y_train[i]))
```

Out[6]:

```
Text(0.5, 1.0, 'Dress / 2')
```

In [7]:

```
X_train.ndim
```

Out[7]:

```
3
```

In [8]:

```
X_train = np.expand_dims(X_train, -1)
X_test =np.expand_dims(X_test, -1)
```

In [9]:

```
X_train.ndim
```

Out[9]:

```
4
```

In [10]:

```
X_train =X_train/255
X_test = X_test/255
```

In [11]:

```
from sklearn.model_selection import train_test_split
X_train, X_validation, y_train, y_validation = train_test_split(X_train, y_train)
```

In [12]:

```
X_train.shape, y_train.shape, X_validation.shape, y_validation.shape
```

```
Out[12]: ((45000, 28, 28, 1), (45000,), (15000, 28, 28, 1), (15000,))
```

```
In [13]: cnn = keras.models.Sequential([

             tf.keras.layers.Conv2D(filters=32, kernel_size=3,strides=(1,1),padding='valid', activatio
             tf.keras.layers.MaxPooling2D((2, 2)),
             tf.keras.layers.Conv2D(filters=64, kernel_size=3,strides=(2,2),padding='same', activation
             tf.keras.layers.MaxPooling2D((2, 2)),
             tf.keras.layers.Flatten(),
             tf.keras.layers.Dense(128, activation='relu'),
             tf.keras.layers.Dropout(0.25),
             tf.keras.layers.Dense(256, activation='relu'),
             tf.keras.layers.Dropout(0.25),
             tf.keras.layers.Dense(128, activation='relu'),
             tf.keras.layers.Dense(10, activation='softmax')

         ])
```

```
In [14]: cnn.compile(optimizer='adam',loss='sparse_categorical_crossentropy',metrics=['accuracy'])
```

```
In [15]: cnn.fit(X_train, y_train, epochs=20 , batch_size=16 , verbose=1 , validation_data=(X_validati
Epoch 1/20
2813/2813 [==============================] - 47s 16ms/step - loss: 0.5674 - accuracy: 0.7893
- val_loss: 0.3913 - val_accuracy: 0.8577
Epoch 2/20
2813/2813 [==============================] - 47s 17ms/step - loss: 0.3755 - accuracy: 0.8636
- val_loss: 0.3322 - val_accuracy: 0.8803
Epoch 3/20
2813/2813 [==============================] - 48s 17ms/step - loss: 0.3296 - accuracy: 0.8808
- val_loss: 0.3171 - val_accuracy: 0.8859
Epoch 4/20
2813/2813 [==============================] - 46s 16ms/step - loss: 0.2997 - accuracy: 0.8890
- val_loss: 0.3176 - val_accuracy: 0.8835
Epoch 5/20
2813/2813 [==============================] - 40s 14ms/step - loss: 0.2797 - accuracy: 0.8980
- val_loss: 0.2929 - val_accuracy: 0.8935
Epoch 6/20
2813/2813 [==============================] - 42s 15ms/step - loss: 0.2670 - accuracy: 0.9029
- val_loss: 0.3052 - val_accuracy: 0.8913
Epoch 7/20
2813/2813 [==============================] - 44s 16ms/step - loss: 0.2537 - accuracy: 0.9072
- val_loss: 0.2897 - val_accuracy: 0.8961
Epoch 8/20
2813/2813 [==============================] - 41s 15ms/step - loss: 0.2420 - accuracy: 0.9131
- val_loss: 0.2979 - val_accuracy: 0.8937
Epoch 9/20
2813/2813 [==============================] - 43s 15ms/step - loss: 0.2289 - accuracy: 0.9156
- val_loss: 0.2920 - val_accuracy: 0.8965
Epoch 10/20
2813/2813 [==============================] - 46s 16ms/step - loss: 0.2214 - accuracy: 0.9173
- val_loss: 0.2878 - val_accuracy: 0.8978
Epoch 11/20
2813/2813 [==============================] - 47s 17ms/step - loss: 0.2136 - accuracy: 0.9212
- val_loss: 0.3027 - val_accuracy: 0.8903
Epoch 12/20
2813/2813 [==============================] - 46s 16ms/step - loss: 0.2055 - accuracy: 0.9247
- val_loss: 0.2749 - val_accuracy: 0.9036
Epoch 13/20
2813/2813 [==============================] - 40s 14ms/step - loss: 0.1991 - accuracy: 0.9259
- val_loss: 0.2858 - val_accuracy: 0.9019
Epoch 14/20
2813/2813 [==============================] - 42s 15ms/step - loss: 0.1920 - accuracy: 0.9278
- val_loss: 0.2790 - val_accuracy: 0.9042
Epoch 15/20
2813/2813 [==============================] - 49s 17ms/step - loss: 0.1849 - accuracy: 0.9322
- val_loss: 0.3083 - val_accuracy: 0.9029
Epoch 16/20
2813/2813 [==============================] - 48s 17ms/step - loss: 0.1829 - accuracy: 0.9318
- val_loss: 0.2945 - val_accuracy: 0.9031
Epoch 17/20
2813/2813 [==============================] - 46s 16ms/step - loss: 0.1771 - accuracy: 0.9360
- val_loss: 0.3130 - val_accuracy: 0.9038
Epoch 18/20
2813/2813 [==============================] - 48s 17ms/step - loss: 0.1755 - accuracy: 0.9359
- val_loss: 0.2987 - val_accuracy: 0.9007
Epoch 19/20
2813/2813 [==============================] - 47s 17ms/step - loss: 0.1701 - accuracy: 0.9371
- val_loss: 0.3227 - val_accuracy: 0.9014
Epoch 20/20
2813/2813 [==============================] - 40s 14ms/step - loss: 0.1636 - accuracy: 0.9406
- val_loss: 0.3009 - val_accuracy: 0.9048
```

Out[15]:
```
<keras.src.callbacks.History at 0x1f6519ca010>
```

In [16]:
```
y_pred = cnn.predict(X_test)
```

```
313/313 [==============================] - 2s 7ms/step
```

In [17]:

```
cnn.evaluate(X_test, y_test)
```

```
313/313 [==============================] - 2s 7ms/step - loss: 0.3154 - accuracy: 0.9020
```

Out[17]:

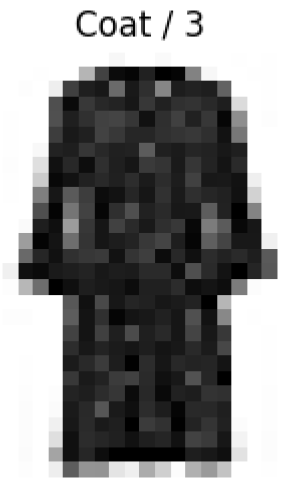
```
[0.3154495656490326, 0.9020000100135803]
```

In [19]:

```
plt.figure(figsize=(16,16))
j=1
for i in np.random.randint(0,1000,25):
 plt.subplot(5,5,j)
j+=1
plt.imshow(X_train[i],cmap="Greys")
plt.axis('off')
plt.title('{} / {}'.format(class_labels[y_train[i]], y_train[i]))
```

Out[19]:

```
Text(0.5, 1.0, 'Coat / 3')
```



Coat / 3