```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import tensorflow as tf


mat = np.array([[1,2,3],
                [4,5,6],
                [7,8,9]])


mat
```

```
     array([[1, 2, 3],
            [4, 5, 6],
            [7, 8, 9]])
```

```
tensor = tf.convert_to_tensor(mat)


tensor
```

```
     <tf.Tensor: shape=(3, 3), dtype=int64, numpy=
     array([[1, 2, 3],
            [4, 5, 6],
            [7, 8, 9]])>
```

```
type(tensor)
```

```
     tensorflow.python.framework.ops.EagerTensor
```

```
a = tf.constant(5) #or u can just do like tf.covert_to_tensor(5)
a
```

```
     <tf.Tensor: shape=(), dtype=int32, numpy=5>
```

```
name = tf.constant("Asmeeta Bardiya")
name
```

```
     <tf.Tensor: shape=(), dtype=string, numpy=b'Asmeeta Bardiya'>
```

```
type(name.numpy())
```

```
     bytes
```

```
dir(name.numpy())
```

```
      '__hash__',
```

```
'__init__',
'__init_subclass__',
'__iter__',
'__le__',
'__len__',
'__lt__',
'__mod__',
'__mul__',
'__ne__',
'__new__',
'__reduce__',
'__reduce_ex__',
'__repr__',
'__rmod__',
'__rmul__',
'__setattr__',
'__sizeof__',
'__str__',
'__subclasshook__',
'capitalize',
'center',
'count',
'decode',
'endswith',
'expandtabs',
'find',
'fromhex',
'hex',
'index',
'isalnum',
'isalpha',
'isascii',
'isdigit',
'islower',
'isspace',
'istitle',
'isupper',
'join',
'ljust',
'lower',
'lstrip',
'maketrans',
'partition',
'replace',
'rfind',
'rindex',
'rjust',
'rpartition',
'rsplit',
'rstrip',
'split',
'splitlines',
'startswith',
'strip',
'swapcase',
'title',
'translate',
```

```
      'upper',
```

```
name.numpy().decode()
```

```
     'Asmeeta Bardiya'
```

```
type(name.numpy().decode())
```

```
     str
```

```
data = tf.keras.datasets.fashion_mnist
```

```
(x_train,y_train),(x_test,y_test) = data.load_data()
```

```
     Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-
     32768/29515 [================================] - 0s 0us/step
     40960/29515 [========================================] - 0s 0us/step
     Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-
     26427392/26421880 [==============================] - 0s 0us/step
     26435584/26421880 [==============================] - 0s 0us/step
     Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-l
     16384/5148 [================================================================================
     Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-i
     4423680/4422102 [==============================] - 0s 0us/step
     4431872/4422102 [==============================] - 0s 0us/step
```

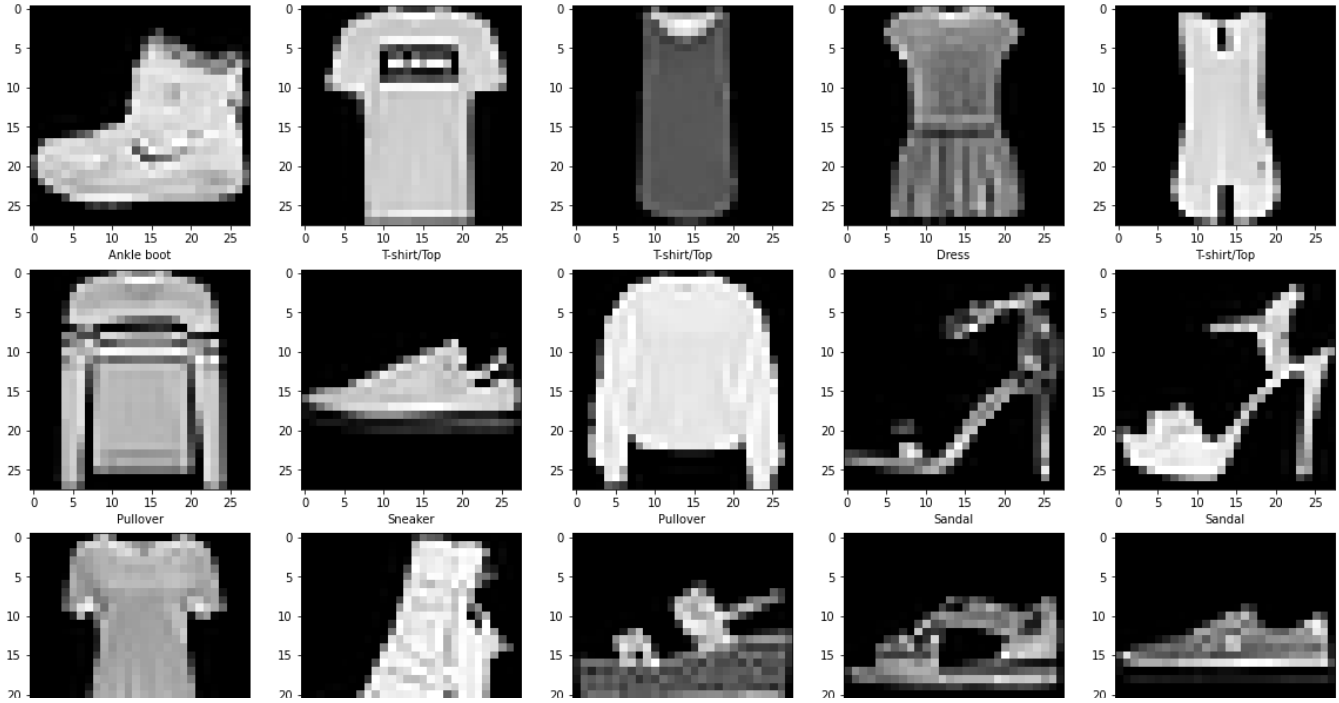
```
x_train.shape
```

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

```
class_names  = ['T-shirt/Top','Trouser','Pullover','Dress','Coat','Sandal','Shirt','Sneaker',
```

```
plt.imshow(x_train[0],cmap=plt.cm.binary_r)
plt.show()
```


```
plt.figure(figsize=(20,20))
for num in range(25):
  plt.subplot(5,5,num+1)
  plt.imshow(x_train[num],cmap=plt.cm.binary_r)
  plt.xlabel(class_names[y_train[num]])
plt.show()
```




```
#Tensorflow Keras Sequential API
model = tf.keras.Sequential([
              tf.keras.layers.Flatten(input_shape=(28,28)),
              tf.keras.layers.Dense(200,activation = 'relu'),
              tf.keras.layers.Dense(10,activation = 'softmax')
])
```

```
model.compile(
    loss='sparse_categorical_crossentropy',
    optimizer='adam',
    metrics=['accuracy'],
)
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
flatten (Flatten)            (None, 784)               0
_________________________________________________________________
dense (Dense)                (None, 200)               157000
_________________________________________________________________
dense_1 (Dense)              (None, 10)                2010
=================================================================
Total params: 159,010
Trainable params: 159,010
Non-trainable params: 0
_________________________________________________________________
```

```
model.fit(x_train,y_train,epochs=10,verbose=1)
```

```
Epoch 1/10
1875/1875 [==============================] - 6s 3ms/step - loss: 4.0579 - accuracy: 0.71
Epoch 2/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.6030 - accuracy: 0.78
Epoch 3/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.5509 - accuracy: 0.80
Epoch 4/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.5292 - accuracy: 0.81
Epoch 5/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.5128 - accuracy: 0.82
Epoch 6/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.4946 - accuracy: 0.82
Epoch 7/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.4791 - accuracy: 0.83
Epoch 8/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.4846 - accuracy: 0.83
Epoch 9/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.4713 - accuracy: 0.83
Epoch 10/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.4748 - accuracy: 0.83
<keras.callbacks.History at 0x7f1f35449dd0>
```

```
model.evaluate(x_test,y_test,verbose=1)
```

```
313/313 [==============================] - 1s 2ms/step - loss: 0.5238 - accuracy: 0.8250
[0.5238378047943115, 0.824999988079071]
```

```
model.evaluate(x_test,y_test,verbose=1)
```

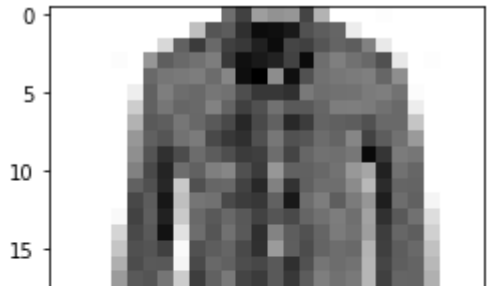
```
313/313 [==============================] - 0s 2ms/step - loss: 0.5238 - accuracy: 0.8250
[0.5238378047943115, 0.824999988079071]
```

```
img = x_test[10].reshape(1,28,28)


pred = np.argmax(model.predict(img))
print(pred)
```

```
4
```

```
plt.imshow(x_test[10],cmap=plt.cm.binary)
plt.show()
```




```
class_names[pred]
```

```
'Coat'
```

```
model.save('myfirstmodel.h5')
```