```
import tensorflow as tf
import keras
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

```
Using TensorFlow backend.
```

```
((train_data, train_labels),
 (eval_data, eval_labels)) = tf.keras.datasets.fashion_mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/trai
32768/29515 [==================================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/trai
26427392/26421880 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k
8192/5148 [==============================================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k
4423680/4422102 [==============================] - 0s 0us/step
```

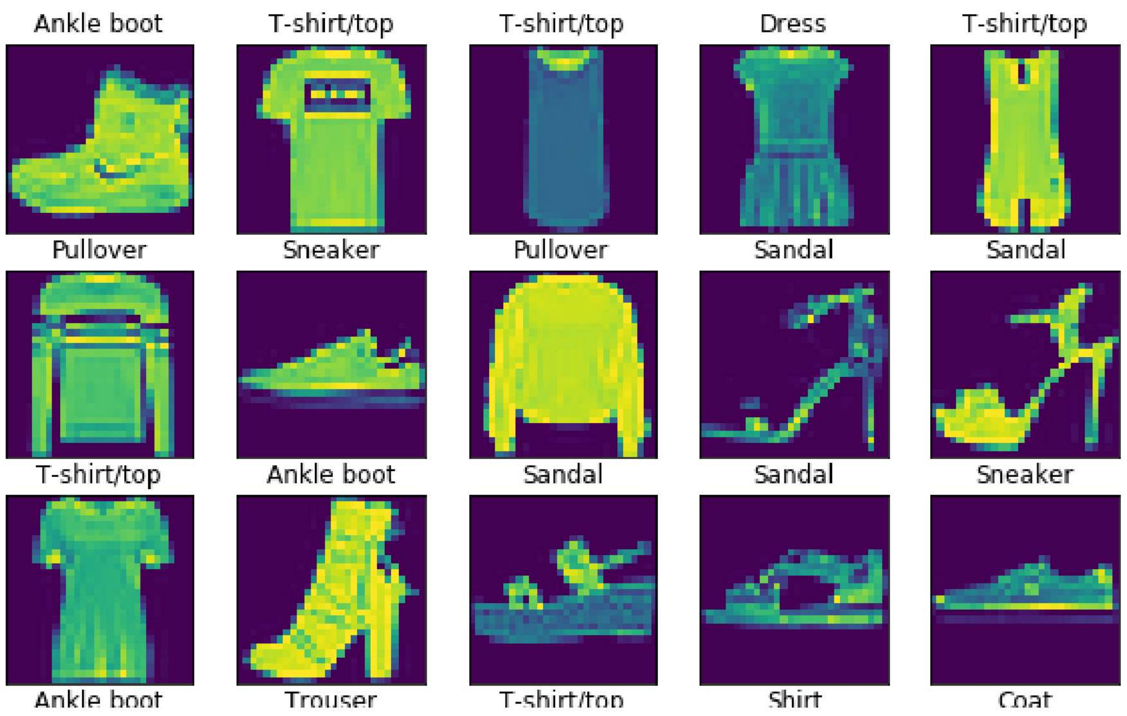
```
target_dict = {
 0: 'T-shirt/top',
 1: 'Trouser',
 2: 'Pullover',
 3: 'Dress',
 4: 'Coat',
 5: 'Sandal',
 6: 'Shirt',
 7: 'Sneaker',
 8: 'Bag',
 9: 'Ankle boot',
}
```

```
print(train_data.shape)
print(eval_data.shape)
```

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

Automatic document saving has been pending for 2832 minutes. Reloading may fix the problem. Save and reload the page.

```
plt.figure(figsize=(10,10))
for i in range(0,20):
    plt.subplot(5,5, i+1)
    plt.imshow(train_data[i] )
    plt.title( target_dict[(train_labels[i]) ])
    plt.xticks([])
    plt.yticks([])
```



```
train_data = train_data/np.float32(255)
train_labels = train_labels.astype(np.int32)
eval_data = eval_data/np.float32(255)
eval_labels = eval_labels.astype(np.int32)
```

```
def cnn_model(features, labels, mode):
    #Reshapinng the input
    input_layer = tf.reshape(features["x"], [-1, 28, 28, 1])

     # Convolutional Layer #1 and Pooling Layer #1
    conv1 = tf.layers.conv2d(
        inputs=input_layer,
        filters=32,
        kernel_size=[5, 5],
        padding="same",
        activation=tf.nn.relu)

    pool1 = tf.layers.max_pooling2d(inputs=conv1, pool_size=[2, 2], strides=2)

    # Convolutional Layer #2 and Pooling Layer #2
    conv2 = tf.layers.conv2d(
```

```
          padding="same",
          activation=tf.nn.relu)

    pool2 = tf.layers.max_pooling2d(inputs=conv2, pool_size=[2, 2], strides=2)

    dropout_1 = tf.layers.dropout(inputs=pool2, rate=0.25,training=mode == tf.estimator.ModeK

    # Convolutional Layer #2 and Pooling Layer #2
    conv3 = tf.layers.conv2d(
          inputs=dropout_1,
          filters=128,
          kernel_size=[5, 5],

          padding="same",
          activation=tf.nn.relu)

    pool3 = tf.layers.max_pooling2d(inputs=conv3, pool_size=[2, 2], strides=2)
    dropout_2 = tf.layers.dropout(inputs=pool3, rate=0.25,training=mode == tf.estimator.ModeK

    flatten_1= tf.reshape(dropout_2, [-1, 3*3*128])

    dense = tf.layers.dense(inputs= flatten_1,units=1024,activation=tf.nn.relu)

    dropout= tf.layers.dropout(inputs=dense, rate=0.4, training=mode == tf.estimator.ModeKeys

    output_layer = tf.layers.dense(inputs= dropout, units=10)
    predictions={
    "classes":tf.argmax(input=output_layer, axis=1),
    "probabilities":tf.nn.softmax(output_layer,name='softmax_tensor')
    }
    if mode==tf.estimator.ModeKeys.PREDICT:
        return tf.estimator.EstimatorSpec(mode=mode, predictions=predictions)

    loss= tf.losses.sparse_softmax_cross_entropy(labels=labels, logits= output_layer, scope='

    if mode== tf.estimator.ModeKeys.TRAIN:
        optimizer= tf.train.AdamOptimizer(learning_rate=0.001)
        train_op= optimizer.minimize(loss=loss, global_step=tf.train.get_global_step())
        return tf.estimator.EstimatorSpec(mode=mode, loss=loss,train_op=train_op )

    eval_metrics_op={ "accuracy":tf.metrics.accuracy(labels=labels,predictions=predictions["c
    return tf.estimator.EstimatorSpec(mode=mode, loss=loss, eval_metric_ops=eval_metrics_op)
```

```
fashion_classifier = tf.estimator.Estimator(model_fn = cnn_model)
```

```
WARNING: Logging before flag parsing goes to stderr.
W0617 06:35:56.366578 139661381625728 estimator.py:1811] Using temporary folder as mod
```

```
train_input_fn = tf.estimator.inputs.numpy_input_fn(
    x={"x": train_data},
    y=train_labels,
    batch_size=100,
    num_epochs=None,
    shuffle=True)
fashion_classifier.train(input_fn=train_input_fn, steps=1500)
```

Automatic document saving has been pending for 2832 minutes. Reloading may fix the problem. Save and reload the page.

```
W0617 06:35:56.438302 139661381625728 deprecation.py:323] From /usr/local/lib/python3.
Instructions for updating:
Use Variable.read_value. Variables in 2.X are initialized automatically both in eager
W0617 06:35:56.462234 139661381625728 deprecation.py:323] From /usr/local/lib/python3.
Instructions for updating:
To construct input pipelines, use the `tf.data` module.
W0617 06:35:56.464882 139661381625728 deprecation.py:323] From /usr/local/lib/python3.
Instructions for updating:
To construct input pipelines, use the `tf.data` module.
W0617 06:35:56.484174 139661381625728 deprecation.py:323] From <ipython-input-7-051681
Instructions for updating:
Use `tf.keras.layers.Conv2D` instead.
W0617 06:35:56.488184 139661381625728 deprecation.py:506] From /usr/local/lib/python3.
```

```
eval_input_fn = tf.estimator.inputs.numpy_input_fn(
    x={"x": eval_data},
    y=eval_labels,
    num_epochs=1,
    shuffle=False)
eval_results = fashion_classifier.evaluate(input_fn=eval_input_fn)
print(eval_results)
```

```
st-packages/tensorflow/python/training/saver.py:1276: checkpoint_exists (from tensorflo




W0617 06:35:58.499929 139661381625728 deprecation.py:323] From /usr/local/lib/python3.
Instructions for updating:
To construct input pipelines, use the `tf.data` module.
<tensorflow_estimator.python.estimator.estimator.Estimator at 0x7f053fd8cbe0>
```

Automatic document saving has been pending for 2832 minutes. Reloading may fix the problem. Save and reload the page.