```
import pandas as pd
import numpy as np
import tensorflow as tf
from tensorflow.keras import models, datasets, layers
import matplotlib.pyplot as plt
import matplotlib.image as mp
```

```
(train_images,train_labels),(test_images,test_labels)=datasets.mnist.load_data()
```

```
print('x_tain: ', train_images.shape)
print('y_tain: ', train_labels.shape)
print('x_test: ', test_images.shape)
print('y_test: ', test_labels.shape)
```

```
x_tain:  (60000, 28, 28)
y_tain:  (60000,)
x_test:  (10000, 28, 28)
y_test:  (10000,)
```

```
pd.DataFrame(train_images[100])
```

	0	1	2	3	4	5	6	7	8	9	...	18	19	20	21	22	23	24	25	26	27
0	0	0	0	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	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	0	0	0	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	0	0
5	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	...	244	255	241	103	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	...	238	218	204	35	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	...	57	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	...	63	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	...	141	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	...	206	3	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	...	253	5	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	...	253	5	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	...	253	5	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	...	209	3	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	...	86	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

28 rows × 28 columns

```
train_images=train_images/255
test_images=test_images/255
```

```
model=models.Sequential()
model.add(layers.Flatten(input_shape=(28,28,1)))
model.add(layers.Dense(32,activation='relu'))
```

```
model.add(layers.Dense(16,activation='relu'))
model.add(layers.Dense(10,activation='softmax'))


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

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten (Flatten)           (None, 784)               0

 dense (Dense)               (None, 32)                25120

 dense_1 (Dense)             (None, 16)                528

 dense_2 (Dense)             (None, 10)                170

=================================================================
Total params: 25,818
Trainable params: 25,818
Non-trainable params: 0
_________________________________________________________________
```

```
h1 = model.fit(train_images,train_labels, epochs=4, validation_data = (test_images,test_labels))
```

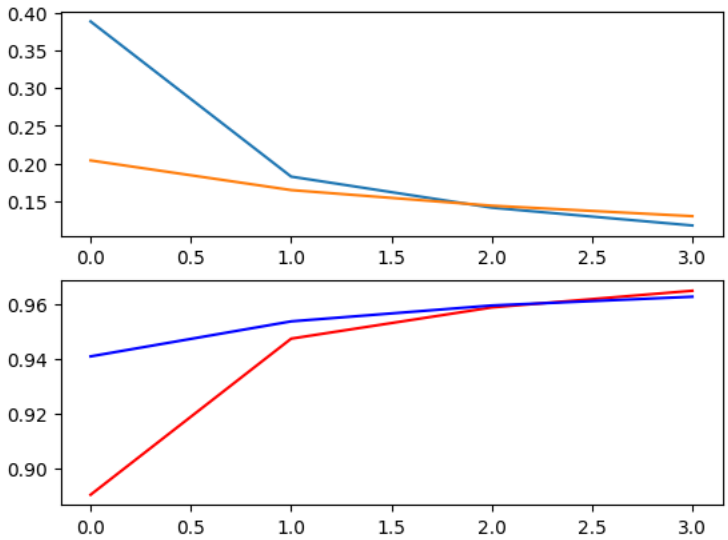
```
Epoch 1/4
1875/1875 [==============================] - 14s 3ms/step - loss: 0.3882 - accuracy: 0.8905 - val_loss: 0.2040 - val_accuracy: 0.941
Epoch 2/4
1875/1875 [==============================] - 6s 3ms/step - loss: 0.1826 - accuracy: 0.9475 - val_loss: 0.1648 - val_accuracy: 0.953
Epoch 3/4
1875/1875 [==============================] - 6s 3ms/step - loss: 0.1414 - accuracy: 0.9589 - val_loss: 0.1440 - val_accuracy: 0.959
Epoch 4/4
1875/1875 [==============================] - 6s 3ms/step - loss: 0.1178 - accuracy: 0.9650 - val_loss: 0.1302 - val_accuracy: 0.962
```

```
f,ax=plt.subplots(2,1)

ax[1].plot(h1.history['accuracy'],color='r', label='train accuracy')
ax[1].plot(h1.history['val_accuracy'],color='b', label='validation Accuracy')

#loss
ax[0].plot(h1.history['loss'], label='train loss')
ax[0].plot(h1.history['val_loss'], label='vall loss')
```

```
[<matplotlib.lines.Line2D at 0x7a7c701cd840>]
```




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

```
h2 = model.fit(train_images,train_labels, epochs=3, validation_data = (test_images,test_labels))
```

```
Epoch 1/3
1875/1875 [==============================] - 6s 3ms/step - loss: 0.8445 - accuracy: 0.7546 - val_loss: 0.3967 - val_accuracy: 0.883
Epoch 2/3
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3500 - accuracy: 0.8999 - val_loss: 0.2921 - val_accuracy: 0.913
```

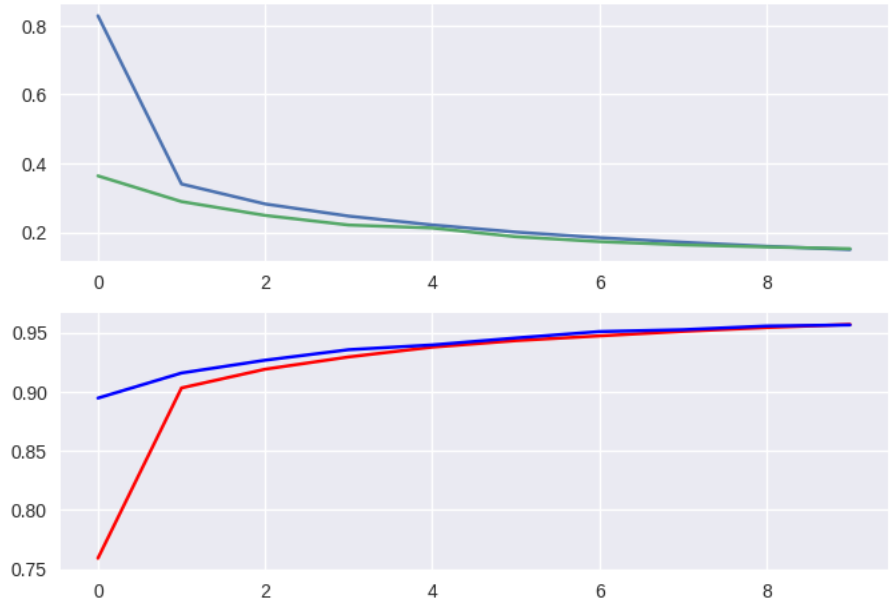
```
Epoch 3/3
1875/1875 [==============================] - 5s 3ms/step - loss: 0.2780 - accuracy: 0.9201 - val_loss: 0.2530 - val_accuracy: 0.923
```

```
f,ax=plt.subplots(2,1)

#loss
ax[0].plot(h2.history['loss'], label='train loss')
ax[0].plot(h2.history['val_loss'], label='vall loss')

ax[1].plot(h2.history['accuracy'],color='r', label='train accuracy')
ax[1].plot(h2.history['val_accuracy'],color='b', label='validation Accuracy')
```

```
[<matplotlib.lines.Line2D at 0x7bc2af0afe80>]
```




```
model.compile(optimizer='rmsprop', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
h3 = model.fit(train_images,train_labels, epochs=10, validation_data = (test_images,test_labels))
```

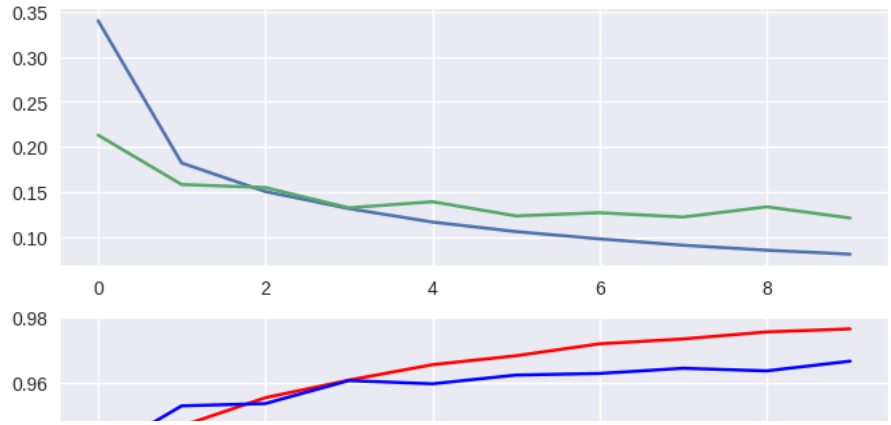
```
Epoch 1/10
1875/1875 [==============================] - 5s 2ms/step - loss: 0.3399 - accuracy: 0.9044 - val_loss: 0.2130 - val_accuracy: 0.937
Epoch 2/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.1821 - accuracy: 0.9465 - val_loss: 0.1582 - val_accuracy: 0.952
Epoch 3/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.1505 - accuracy: 0.9552 - val_loss: 0.1549 - val_accuracy: 0.953
Epoch 4/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.1316 - accuracy: 0.9606 - val_loss: 0.1324 - val_accuracy: 0.960
Epoch 5/10
1875/1875 [==============================] - 4s 2ms/step - loss: 0.1166 - accuracy: 0.9654 - val_loss: 0.1392 - val_accuracy: 0.959
Epoch 6/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.1060 - accuracy: 0.9682 - val_loss: 0.1234 - val_accuracy: 0.962
Epoch 7/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.0979 - accuracy: 0.9718 - val_loss: 0.1270 - val_accuracy: 0.962
Epoch 8/10
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0909 - accuracy: 0.9733 - val_loss: 0.1222 - val_accuracy: 0.964
Epoch 9/10
1875/1875 [==============================] - 5s 2ms/step - loss: 0.0854 - accuracy: 0.9755 - val_loss: 0.1335 - val_accuracy: 0.963
Epoch 10/10
1875/1875 [==============================] - 5s 2ms/step - loss: 0.0810 - accuracy: 0.9765 - val_loss: 0.1211 - val_accuracy: 0.966
```

```
f,ax=plt.subplots(2,1)

#loss
ax[0].plot(h3.history['loss'], label='train loss')
ax[0].plot(h3.history['val_loss'], label='vall loss')

ax[1].plot(h3.history['accuracy'],color='r', label='train accuracy')
ax[1].plot(h3.history['val_accuracy'],color='b', label='validation Accuracy')
```

```
[<matplotlib.lines.Line2D at 0x7bc2b76ff640>]
```




```
model.compile(optimizer='nadam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
h4 = model.fit(train_images,train_labels, epochs=10, validation_data = (test_images,test_labels))
```

```
Epoch 1/10
1875/1875 [==============================] - 8s 3ms/step - loss: 0.3799 - accuracy: 0.8916 - val_loss: 0.2104 - val_accuracy: 0.9397
Epoch 2/10
1875/1875 [==============================] - 11s 6ms/step - loss: 0.1865 - accuracy: 0.9459 - val_loss: 0.1713 - val_accuracy: 0.951
Epoch 3/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.1482 - accuracy: 0.9560 - val_loss: 0.1451 - val_accuracy: 0.9573
Epoch 4/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.1272 - accuracy: 0.9618 - val_loss: 0.1403 - val_accuracy: 0.9592
Epoch 5/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.1113 - accuracy: 0.9665 - val_loss: 0.1362 - val_accuracy: 0.9611
Epoch 6/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.1000 - accuracy: 0.9701 - val_loss: 0.1241 - val_accuracy: 0.9627
Epoch 7/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.0899 - accuracy: 0.9724 - val_loss: 0.1333 - val_accuracy: 0.9617
Epoch 8/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.0833 - accuracy: 0.9744 - val_loss: 0.1300 - val_accuracy: 0.9624
Epoch 9/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.0766 - accuracy: 0.9760 - val_loss: 0.1252 - val_accuracy: 0.9646
Epoch 10/10
1875/1875 [==============================] - 5s 2ms/step - loss: 0.0699 - accuracy: 0.9783 - val_loss: 0.1369 - val_accuracy: 0.9629
```

```
f,ax=plt.subplots(2,1)

#loss
ax[0].plot(h4.history['loss'], label='train loss')
ax[0].plot(h4.history['val_loss'], label='vall loss')

ax[1].plot(h4.history['accuracy'],color='r', label='train accuracy')
ax[1].plot(h4.history['val_accuracy'],color='b', label='validation Accuracy')
```

```
[<matplotlib.lines.Line2D at 0x7bc2b72803d0>]
```