```
# importing libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow import keras
```

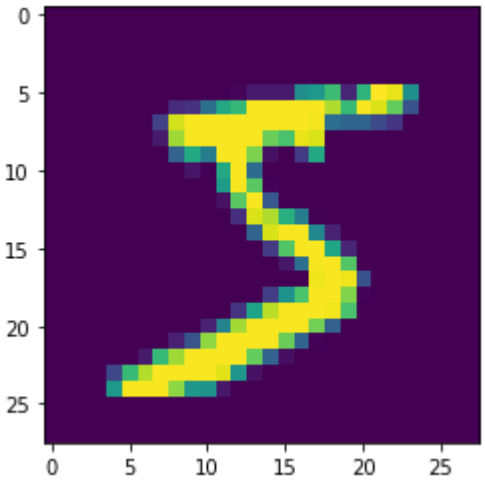
```
# importing dataset
digit_mnist = keras.datasets.mnist
(x_train_full, y_train_full), (x_test, y_test) = digit_mnist.load_data()
```

```
x_train_full.shape
```

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

```
plt.imshow(x_train_full[0])
```

```
<matplotlib.image.AxesImage at 0x12a396ed160>
```




```
# feature scaling
x_train_n = x_train_full / 255.
x_test_n = x_test / 255.
```

```
# Train_Valid_Test Split
x_valid, x_train = x_train_n[:6000],x_train_n[6000:]
y_valid, y_train = y_train_full[:6000],y_train_full[6000:]
x_test = x_test_n
```

```
np.random.seed(42)
tf.random.set_seed(42)
```

```
# Model Building
```

```
model = keras.models.Sequential()
model.add(keras.layers.Flatten(input_shape=[28,28]))
model.add(keras.layers.Dense(200,activation='relu'))
model.add(keras.layers.Dense(100,activation='relu'))
model.add(keras.layers.Dense(10,activation='softmax'))
```

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

```
# model training
model_r = model.fit(x_train,y_train,epochs=60,
                          validation_data=(x_valid,y_valid))
```

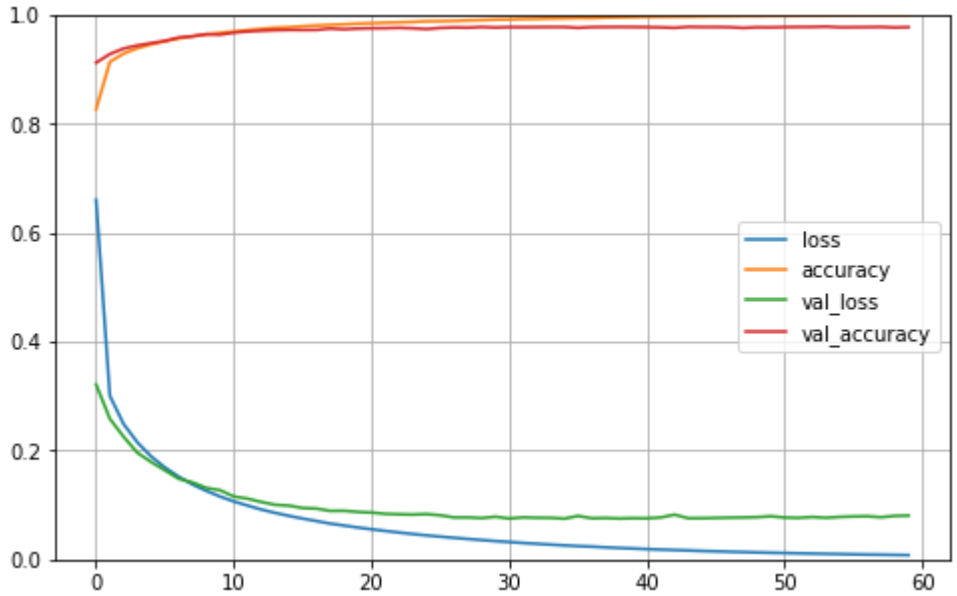
```
Epoch 1/60
1688/1688 [==============================] - 3s 1ms/step - loss: 0.6607 - accuracy: 0
Epoch 2/60
1688/1688 [==============================] - 2s 1ms/step - loss: 0.3006 - accuracy: 0
Epoch 3/60
1688/1688 [==============================] - 2s 1ms/step - loss: 0.2488 - accuracy: 0
Epoch 4/60
1688/1688 [==============================] - 2s 1ms/step - loss: 0.2151 - accuracy: 0
Epoch 5/60
1688/1688 [==============================] - 2s 1ms/step - loss: 0.1893 - accuracy: 0
Epoch 6/60
1688/1688 [==============================] - 2s 1ms/step - loss: 0.1690 - accuracy: 0
Epoch 7/60
1688/1688 [==============================] - 2s 1ms/step - loss: 0.1521 - accuracy: 0
Epoch 8/60
1688/1688 [==============================] - 2s 1ms/step - loss: 0.1378 - accuracy: 0
Epoch 9/60
1688/1688 [==============================] - 2s 1ms/step - loss: 0.1260 - accuracy: 0
Epoch 10/60
1688/1688 [==============================] - 2s 1ms/step - loss: 0.1157 - accuracy: 0
Epoch 11/60
1688/1688 [==============================] - 2s 1ms/step - loss: 0.1066 - accuracy: 0
Epoch 12/60
1688/1688 [==============================] - 3s 1ms/step - loss: 0.0991 - accuracy: 0
Epoch 13/60
1688/1688 [==============================] - 3s 2ms/step - loss: 0.0917 - accuracy: 0
Epoch 14/60
1688/1688 [==============================] - 2s 1ms/step - loss: 0.0857 - accuracy: 0
Epoch 15/60
1688/1688 [==============================] - 2s 1ms/step - loss: 0.0802 - accuracy: 0
Epoch 16/60
1688/1688 [==============================] - 3s 2ms/step - loss: 0.0752 - accuracy: 0
Epoch 17/60
1688/1688 [==============================] - 3s 1ms/step - loss: 0.0707 - accuracy: 0
Epoch 18/60
1688/1688 [==============================] - 3s 2ms/step - loss: 0.0659 - accuracy: 0
Epoch 19/60
1688/1688 [==============================] - 2s 1ms/step - loss: 0.0624 - accuracy: 0
Epoch 20/60
1688/1688 [==============================] - 2s 1ms/step - loss: 0.0587 - accuracy: 0
```

```
Epoch 21/60
1688/1688 [==============================] - 3s 2ms/step - loss: 0.0555 - accuracy: 0
Epoch 22/60
1688/1688 [==============================] - 3s 1ms/step - loss: 0.0525 - accuracy: 0
Epoch 23/60
1688/1688 [==============================] - 3s 1ms/step - loss: 0.0496 - accuracy: 0
Epoch 24/60
1688/1688 [==============================] - 3s 2ms/step - loss: 0.0468 - accuracy: 0
Epoch 25/60
1688/1688 [==============================] - 3s 2ms/step - loss: 0.0443 - accuracy: 0
Epoch 26/60
1688/1688 [==============================] - 3s 2ms/step - loss: 0.0420 - accuracy: 0
Epoch 27/60
1688/1688 [==============================] - 3s 2ms/step - loss: 0.0398 - accuracy: 0
Epoch 28/60
1688/1688 [==============================] - 3s 2ms/step - loss: 0.0378 - accuracy: 0
Epoch 29/60
```

```
model.evaluate(x_test,y_test)
```

```
313/313 [==============================] - 0s 1ms/step - loss: 0.0741 - accuracy: 0.9795
[0.07412999868392944, 0.9794999957084656]
```

```
# Val-loss
pd.DataFrame(model_r.history).plot(figsize=(8,5))
plt.grid(True)
plt.gca().set_ylim(0,1)
plt.show()
```




```
# predicting first five Digits
x_new=x_test[:5]
```

```
y_proba = model.predict(x_new)
y_proba.round(2)
```

```
array([[0., 0., 0., 0., 0., 0., 0., 1., 0., 0.],
       [0., 0., 1., 0., 0., 0., 0., 0., 0., 0.],
       [0., 1., 0., 0., 0., 0., 0., 0., 0., 0.],
       [1., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 1., 0., 0., 0., 0., 0.]], dtype=float32)
```

```
predict_x = model.predict(x_new)
classes_x=np.argmax(predict_x,axis=1)
```

```
# Output
classes_x
```

```
array([7, 2, 1, 0, 4], dtype=int64)
```