In [1]:

```
# Importing Libraries


import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import os
from sklearn.model_selection import train_test_split
from keras.utils import to_categorical
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras.layers import Dense, Conv2D, Flatten, MaxPooling2D
from keras.callbacks import EarlyStopping, ModelCheckpoint, ReduceLROnPlateau


import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
# Importing Data

train_data = pd.read_csv("train.csv")
test_data = pd.read_csv("test.csv")
```

In [3]:

```
# Data exploration

print(train_data.columns)
print('-------------------------------------------------')
print(test_data.columns)
```

```
Index(['label', 'pixel0', 'pixel1', 'pixel2', 'pixel3', 'pixel4', 'pixel5',
       'pixel6', 'pixel7', 'pixel8',
       ...
       'pixel774', 'pixel775', 'pixel776', 'pixel777', 'pixel778', 'pixel779',
       'pixel780', 'pixel781', 'pixel782', 'pixel783'],
      dtype='object', length=785)
-------------------------------------------------
Index(['pixel0', 'pixel1', 'pixel2', 'pixel3', 'pixel4', 'pixel5', 'pixel6',
       'pixel7', 'pixel8', 'pixel9',
       ...
       'pixel774', 'pixel775', 'pixel776', 'pixel777', 'pixel778', 'pixel779',
       'pixel780', 'pixel781', 'pixel782', 'pixel783'],
      dtype='object', length=784)
```

In [4]:

```
print(train_data.shape)
print('-----------')
print(test_data.shape)
```

```
(42000, 785)
-----------
(28000, 784)
```

In [5]:

```
train_data.head(10)
```

Out[5]:

	label	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	...	pixel774	pixel775	pixel776	pixel777	pixel778	pixel779	pixel780	pixel781	pixel782	pixel783
0	1	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	1	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	4	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	7	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
7	3	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
8	5	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
9	3	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

10 rows × 785 columns

In [6]:

```
test_data.head(10)
```

Out[6]:

	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	pixel9	...	pixel774	pixel775	pixel776	pixel777	pixel778	pixel779	pixel780	pixel781	pixel782	pixel783
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	...	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	...	0	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

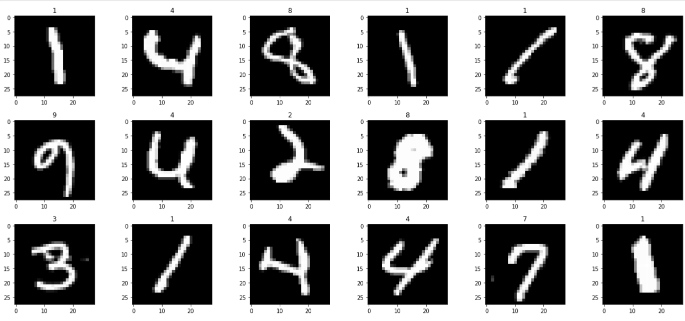
10 rows × 784 columns

In [7]:

```
# Showing images for example

def show_image(train_image, label, index):
    image_shaped = train_image.values.reshape(28,28)
    plt.subplot(3, 6, index+1)
    plt.imshow(image_shaped, cmap=plt.cm.gray)
    plt.title(label)


plt.figure(figsize=(18, 8))
sample_image = train_data.sample(18).reset_index(drop=True)
for index, row in sample_image.iterrows():
    label = row['label']
    image_pixels = row.drop('label')
    show_image(image_pixels, label, index)
plt.tight_layout()
```



In [8]:

```
# Reshaping the data

x = train_data.drop(columns=['label']).values.reshape(train_data.shape[0],28,28,1)
y = to_categorical(train_data['label'])
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.1, random_state=2)
```

In [9]:

```
# Setting epochs and batch_Size


fast_run=False
batch_size=32
epochs=32
if fast_run:
    epochs=1
```

In [10]:

```
# Image Data Generator

train_datagen = ImageDataGenerator(rotation_range=10,rescale=1./255,shear_range=0.1,zoom_range=0.1,width_shift_range=0.1,
    height_shift_range=0.1)
train_datagen.fit(x_train)
train_generator = train_datagen.flow(x_train,y_train,batch_size=batch_size)

validation_datagen = ImageDataGenerator(rescale=1./255) # Normalization
train_datagen.fit(x_test)
validation_generator = validation_datagen.flow(x_test,y_test)
```

In [11]:

```
# Model building

model = Sequential()
model.add(Conv2D(64, kernel_size=3, activation='relu', input_shape=(28,28,1)))
model.add(Conv2D(32, kernel_size=3, activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(10, activation='softmax'))
```

In [12]:

```
model.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy'])
```

In [13]:

```
# Callbacks

callbacks = [
    EarlyStopping(patience=10, verbose=1),
    ReduceLROnPlateau(factor=0.1, patience=3, min_lr=0.00001, verbose=1),
    ModelCheckpoint('model.h5', verbose=1, save_best_only=True, save_weights_only=True)
]
```

In [14]:

```
model.fit_generator(
    train_generator,
    steps_per_epoch=len(x_train) // batch_size,
    validation_data=validation_generator,
    validation_steps=len(x_test) // batch_size,
    epochs=epochs,
    callbacks=callbacks
)
```

```
WARNING:tensorflow:From <ipython-input-14-da6144225fb3>:1: Model.fit_generator (from tensorflow.python.keras.engine.training) is deprecated and will be removed in a future version.
Instructions for updating:
Please use Model.fit, which supports generators.
Epoch 1/32
1181/1181 [==============================] - ETA: 0s - loss: 0.3892 - accuracy: 0.8775
Epoch 00001: val_loss improved from inf to 0.10037, saving model to model.h5
1181/1181 [==============================] - 86s 73ms/step - loss: 0.3892 - accuracy: 0.8775 - val_loss: 0.1004 - val_accuracy: 0.9726
Epoch 2/32
1181/1181 [==============================] - ETA: 0s - loss: 0.1651 - accuracy: 0.9499
Epoch 00002: val_loss improved from 0.10037 to 0.07160, saving model to model.h5
1181/1181 [==============================] - 85s 72ms/step - loss: 0.1651 - accuracy: 0.9499 - val_loss: 0.0716 - val_accuracy: 0.9795
Epoch 3/32
1181/1181 [==============================] - ETA: 0s - loss: 0.1343 - accuracy: 0.9600 ETA: 0s - loss: 0.1341 - accuracy
Epoch 00003: val_loss improved from 0.07160 to 0.06869, saving model to model.h5
1181/1181 [==============================] - 85s 72ms/step - loss: 0.1343 - accuracy: 0.9600 - val_loss: 0.0687 - val_accuracy: 0.9823
Epoch 4/32
1181/1181 [==============================] - ETA: 0s - loss: 0.1158 - accuracy: 0.9649
Epoch 00004: val_loss improved from 0.06869 to 0.06107, saving model to model.h5
1181/1181 [==============================] - 85s 72ms/step - loss: 0.1158 - accuracy: 0.9649 - val_loss: 0.0611 - val_accuracy: 0.9843
Epoch 5/32
1181/1181 [==============================] - ETA: 0s - loss: 0.1045 - accuracy: 0.9694
Epoch 00005: val_loss did not improve from 0.06107
1181/1181 [==============================] - 85s 72ms/step - loss: 0.1045 - accuracy: 0.9694 - val_loss: 0.0626 - val_accuracy: 0.9840
Epoch 6/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0962 - accuracy: 0.9718
Epoch 00006: val_loss improved from 0.06107 to 0.05995, saving model to model.h5
1181/1181 [==============================] - 85s 72ms/step - loss: 0.0962 - accuracy: 0.9718 - val_loss: 0.0600 - val_accuracy: 0.9840
Epoch 7/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0943 - accuracy: 0.9731
Epoch 00007: val_loss improved from 0.05995 to 0.05382, saving model to model.h5
1181/1181 [==============================] - 104s 88ms/step - loss: 0.0943 - accuracy: 0.9731 - val_loss: 0.0538 - val_accuracy: 0.9854
Epoch 8/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0907 - accuracy: 0.9733
Epoch 00008: val_loss improved from 0.05382 to 0.04945, saving model to model.h5
1181/1181 [==============================] - 149s 126ms/step - loss: 0.0907 - accuracy: 0.9733 - val_loss: 0.0494 - val_accuracy: 0.9874
Epoch 9/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0878 - accuracy: 0.9751
Epoch 00009: val_loss improved from 0.04945 to 0.04936, saving model to model.h5
1181/1181 [==============================] - 156s 132ms/step - loss: 0.0878 - accuracy: 0.9751 - val_loss: 0.0494 - val_accuracy: 0.9859
Epoch 10/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0878 - accuracy: 0.9744
Epoch 00010: val_loss improved from 0.04936 to 0.04783, saving model to model.h5
1181/1181 [==============================] - 153s 130ms/step - loss: 0.0878 - accuracy: 0.9744 - val_loss: 0.0478 - val_accuracy: 0.9876
Epoch 11/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0859 - accuracy: 0.9748
Epoch 00011: val_loss did not improve from 0.04783
1181/1181 [==============================] - 154s 131ms/step - loss: 0.0859 - accuracy: 0.9748 - val_loss: 0.0524 - val_accuracy: 0.9862
Epoch 12/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0850 - accuracy: 0.9749
Epoch 00012: val_loss did not improve from 0.04783
1181/1181 [==============================] - 146s 123ms/step - loss: 0.0850 - accuracy: 0.9749 - val_loss: 0.0536 - val_accuracy: 0.9866
Epoch 13/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0834 - accuracy: 0.9760
Epoch 00013: ReduceLROnPlateau reducing learning rate to 0.00010000000474974513.

Epoch 00013: val_loss did not improve from 0.04783
1181/1181 [==============================] - 146s 123ms/step - loss: 0.0834 - accuracy: 0.9760 - val_loss: 0.0499 - val_accuracy: 0.9878
Epoch 14/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0720 - accuracy: 0.9787
Epoch 00014: val_loss improved from 0.04783 to 0.04727, saving model to model.h5
1181/1181 [==============================] - 146s 124ms/step - loss: 0.0720 - accuracy: 0.9787 - val_loss: 0.0473 - val_accuracy: 0.9876
Epoch 15/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0657 - accuracy: 0.9809
Epoch 00015: val_loss improved from 0.04727 to 0.04592, saving model to model.h5
1181/1181 [==============================] - 148s 125ms/step - loss: 0.0657 - accuracy: 0.9809 - val_loss: 0.0459 - val_accuracy: 0.9885
Epoch 16/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0646 - accuracy: 0.9806
Epoch 00016: val_loss improved from 0.04592 to 0.04423, saving model to model.h5
1181/1181 [==============================] - 146s 123ms/step - loss: 0.0646 - accuracy: 0.9806 - val_loss: 0.0442 - val_accuracy: 0.9878
Epoch 17/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0682 - accuracy: 0.9800
Epoch 00017: val_loss did not improve from 0.04423
1181/1181 [==============================] - 149s 126ms/step - loss: 0.0682 - accuracy: 0.9800 - val_loss: 0.0458 - val_accuracy: 0.9876
Epoch 18/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0691 - accuracy: 0.9796
Epoch 00018: val_loss did not improve from 0.04423
1181/1181 [==============================] - 148s 125ms/step - loss: 0.0691 - accuracy: 0.9796 - val_loss: 0.0454 - val_accuracy: 0.9881
Epoch 19/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0675 - accuracy: 0.9802
Epoch 00019: val_loss improved from 0.04423 to 0.04386, saving model to model.h5
1181/1181 [==============================] - 148s 126ms/step - loss: 0.0675 - accuracy: 0.9802 - val_loss: 0.0439 - val_accuracy: 0.9883
Epoch 20/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0648 - accuracy: 0.9803
Epoch 00020: val_loss did not improve from 0.04386
1181/1181 [==============================] - 149s 126ms/step - loss: 0.0648 - accuracy: 0.9803 - val_loss: 0.0449 - val_accuracy: 0.9876
Epoch 21/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0635 - accuracy: 0.9811
Epoch 00021: val_loss did not improve from 0.04386
1181/1181 [==============================] - 147s 125ms/step - loss: 0.0635 - accuracy: 0.9811 - val_loss: 0.0441 - val_accuracy: 0.9881
Epoch 22/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0668 - accuracy: 0.9811
Epoch 00022: val_loss improved from 0.04386 to 0.04249, saving model to model.h5
1181/1181 [==============================] - 147s 124ms/step - loss: 0.0668 - accuracy: 0.9811 - val_loss: 0.0425 - val_accuracy: 0.9883
Epoch 23/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0680 - accuracy: 0.9797
Epoch 00023: val_loss improved from 0.04249 to 0.04238, saving model to model.h5
1181/1181 [==============================] - 148s 125ms/step - loss: 0.0680 - accuracy: 0.9797 - val_loss: 0.0424 - val_accuracy: 0.9881
Epoch 24/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0632 - accuracy: 0.9810
Epoch 00024: val_loss did not improve from 0.04238
1181/1181 [==============================] - 147s 124ms/step - loss: 0.0632 - accuracy: 0.9810 - val_loss: 0.0450 - val_accuracy: 0.9881
Epoch 25/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0649 - accuracy: 0.9812
Epoch 00025: val_loss improved from 0.04238 to 0.04179, saving model to model.h5
1181/1181 [==============================] - 148s 125ms/step - loss: 0.0649 - accuracy: 0.9812 - val_loss: 0.0418 - val_accuracy: 0.9895
Epoch 26/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0648 - accuracy: 0.9812
Epoch 00026: val_loss did not improve from 0.04179
1181/1181 [==============================] - 148s 125ms/step - loss: 0.0648 - accuracy: 0.9812 - val_loss: 0.0425 - val_accuracy: 0.9888
Epoch 27/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0647 - accuracy: 0.9814
Epoch 00027: val_loss did not improve from 0.04179
1181/1181 [==============================] - 146s 124ms/step - loss: 0.0647 - accuracy: 0.9814 - val_loss: 0.0437 - val_accuracy: 0.9895
Epoch 28/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0643 - accuracy: 0.9817
Epoch 00028: ReduceLROnPlateau reducing learning rate to 1.0000000474974514e-05.

Epoch 00028: val_loss did not improve from 0.04179
1181/1181 [==============================] - 149s 126ms/step - loss: 0.0643 - accuracy: 0.9817 - val_loss: 0.0428 - val_accuracy: 0.9895
Epoch 29/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0666 - accuracy: 0.9809
Epoch 00029: val_loss did not improve from 0.04179
1181/1181 [==============================] - 149s 126ms/step - loss: 0.0666 - accuracy: 0.9809 - val_loss: 0.0438 - val_accuracy: 0.9885
Epoch 30/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0658 - accuracy: 0.9821
Epoch 00030: val_loss did not improve from 0.04179
1181/1181 [==============================] - 146s 124ms/step - loss: 0.0658 - accuracy: 0.9821 - val_loss: 0.0435 - val_accuracy: 0.9881
Epoch 31/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0633 - accuracy: 0.9818
Epoch 00031: ReduceLROnPlateau reducing learning rate to 1e-05.

Epoch 00031: val_loss did not improve from 0.04179
1181/1181 [==============================] - 143s 121ms/step - loss: 0.0633 - accuracy: 0.9818 - val_loss: 0.0434 - val_accuracy: 0.9888
Epoch 32/32
1181/1181 [==============================] - ETA: 0s - loss: 0.0623 - accuracy: 0.9818
Epoch 00032: val_loss did not improve from 0.04179
1181/1181 [==============================] - 139s 117ms/step - loss: 0.0623 - accuracy: 0.9818 - val_loss: 0.0442 - val_accuracy: 0.9885
```

Out[14]:

```
<tensorflow.python.keras.callbacks.History at 0x1fd07e69e20>
```

In [15]:

```
x_test_recaled = (x_test.astype("float32") / 255)
scores = model.evaluate(x_test_recaled, y_test, verbose=0)
print("%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
print("%s: %.2f%%" % (model.metrics_names[0], scores[0]*100))
```

```
accuracy: 98.86%
loss: 4.41%
```

In [16]:

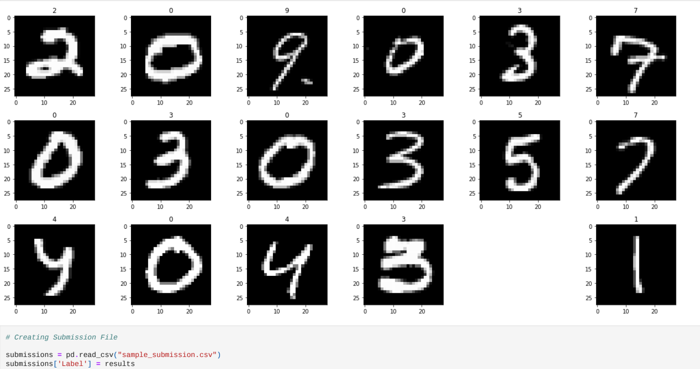
```
# testing on test_data

test_digit_data = test_data.values.reshape(test_data.shape[0],28,28,1).astype("float32") / 255
predictions = model.predict(test_digit_data)
results = np.argmax(predictions, axis = 1)
```

In [17]:

```
# checking the accurate of my result

plt.figure(figsize=(18, 8))
sample_test = test_data.head(18)
for index, image_pixels in sample_test.iterrows():
    label = results[index]
    show_image(image_pixels, label, index)
    plt.tight_layout()
```



In [18]:

```
# Creating Submission File

submissions = pd.read_csv("sample_submission.csv")
submissions['Label'] = results
submissions.to_csv('submission.csv', index = False)
```

In [ ]: