

In [1]:

```
import tensorflow as tf
(x_train, y_train), (x_test, y_test) = tf.keras.datasets.mnist.load_data()
```

The default version of TensorFlow in Colab will soon switch to TensorFlow 2.x.

We recommend you [upgrade \(https://www.tensorflow.org/guide/migrate\)](https://www.tensorflow.org/guide/migrate) now or ensure your notebook will continue to use TensorFlow 1.x via the %tensorflow_version 1.x magic: [more info \(https://colab.research.google.com/notebooks/tensorflow_version.ipynb\)](https://colab.research.google.com/notebooks/tensorflow_version.ipynb).

Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz

11493376/11490434 [=====] - 0s 0us/step

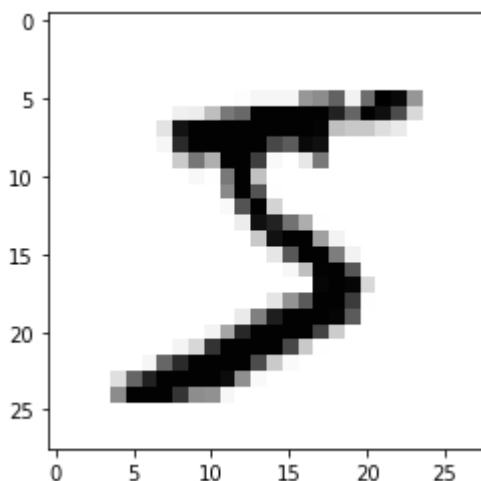
In [5]:

```
import matplotlib.pyplot as plt
#matplotlib inline # Only use this if using iPython
image_index = 0 # You may select anything up to 60,000
print(y_train[image_index]) # The label is 8
plt.imshow(x_train[image_index], cmap='Greys')
```

5

Out[5]:

<matplotlib.image.AxesImage at 0x7feeab312160>



In [8]:

```
print(x_train.shape)
x_test.shape
```

(60000, 28, 28, 1)

Out[8]:

(10000, 28, 28, 1)

In [7]:

```
# Reshaping the array to 4-dims so that it can work with the Keras API
x_train = x_train.reshape(x_train.shape[0], 28, 28, 1)
x_test = x_test.reshape(x_test.shape[0], 28, 28, 1)
input_shape = (28, 28, 1)
# Making sure that the values are float so that we can get decimal points after
division
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
# Normalizing the RGB codes by dividing it to the max RGB value.
x_train /= 255
x_test /= 255
print('x_train shape:', x_train.shape)
print('Number of images in x_train', x_train.shape[0])
print('Number of images in x_test', x_test.shape[0])
```

```
x_train shape: (60000, 28, 28, 1)
Number of images in x_train 60000
Number of images in x_test 10000
```

In [9]:

```
# Importing the required Keras modules containing model and layers
from keras.models import Sequential
from keras.layers import Dense, Conv2D, Dropout, Flatten, MaxPooling2D
# Creating a Sequential Model and adding the layers
model = Sequential()
model.add(Conv2D(28, kernel_size=(3,3), input_shape=input_shape))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten()) # Flattening the 2D arrays for fully connected layers
model.add(Dense(128, activation=tf.nn.relu))
model.add(Dropout(0.2))
model.add(Dense(10,activation=tf.nn.softmax))
```

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:66: The name tf.get_default_graph is deprecated. Please use tf.compat.v1.get_default_graph instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:541: The name tf.placeholder is deprecated. Please use tf.compat.v1.placeholder instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:4432: The name tf.random_uniform is deprecated. Please use tf.random.uniform instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:4267: The name tf.nn.max_pool is deprecated. Please use tf.nn.max_pool2d instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:148: The name tf.placeholder_with_default is deprecated. Please use tf.compat.v1.placeholder_with_default instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:3733: calling dropout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in a future version.

Instructions for updating:

Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.

Using TensorFlow backend.

In [10]:

```
model.compile(optimizer='adam',  
              loss='sparse_categorical_crossentropy',  
              metrics=['accuracy'])  
model.fit(x=x_train,y=y_train, epochs=10)
```

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/optimizers.py:793: The name tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:3622: The name tf.log is deprecated. Please use tf.math.log instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_core/python/ops/math_grad.py:1424: where (from tensorflow.python.ops.array_ops) is deprecated and will be removed in a future version.

Instructions for updating:

Use tf.where in 2.0, which has the same broadcast rule as np.where

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:1033: The name tf.assign_add is deprecated. Please use tf.compat.v1.assign_add instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:1020: The name tf.assign is deprecated. Please use tf.compat.v1.assign instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:3005: The name tf.Session is deprecated. Please use tf.compat.v1.Session instead.

Epoch 1/10

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:190: The name tf.get_default_session is deprecated. Please use tf.compat.v1.get_default_session instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:197: The name tf.ConfigProto is deprecated. Please use tf.compat.v1.ConfigProto instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:207: The name tf.global_variables is deprecated. Please use tf.compat.v1.global_variables instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:216: The name tf.is_variable_initialized is deprecated. Please use tf.compat.v1.is_variable_initialized instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:223: The name tf.variables_initializer is deprecated. Please use tf.compat.v1.variables_initializer instead.

60000/60000 [=====] - 37s 625us/step - loss: 0.1969 - acc: 0.9412

Epoch 2/10

60000/60000 [=====] - 37s 621us/step - loss: 0.0782 - acc: 0.9761

Epoch 3/10

60000/60000 [=====] - 37s 617us/step - loss: 0.0582 - acc: 0.9819

Epoch 4/10

60000/60000 [=====] - 37s 609us/step - loss: 0.0412 - acc: 0.9871

Epoch 5/10

```
60000/60000 [=====] - 37s 609us/step - los
s: 0.0335 - acc: 0.9890
Epoch 6/10
60000/60000 [=====] - 37s 613us/step - los
s: 0.0290 - acc: 0.9905
Epoch 7/10
60000/60000 [=====] - 37s 609us/step - los
s: 0.0231 - acc: 0.9919
Epoch 8/10
60000/60000 [=====] - 37s 612us/step - los
s: 0.0209 - acc: 0.9926
Epoch 9/10
60000/60000 [=====] - 36s 605us/step - los
s: 0.0187 - acc: 0.9939
Epoch 10/10
60000/60000 [=====] - 37s 624us/step - los
s: 0.0180 - acc: 0.9940
```

Out[10]:

<keras.callbacks.History at 0x7feeab3126d8>

In [11]:

```
model.evaluate(x_test, y_test)
#98.5% accuracy
```

```
10000/10000 [=====] - 2s 185us/step
```

Out[11]:

```
[0.06424489634489873, 0.9862]
```

In [0]:

```
image_index = 4444
plt.imshow(x_test[image_index].reshape(28, 28), cmap='Greys')
pred = model.predict(x_test[image_index].reshape(1, img_rows, img_cols, 1))
print(pred.argmax())
```

Reference: <https://towardsdatascience.com/image-classification-in-10-minutes-with-mnist-dataset-54c35b77a38d> (<https://towardsdatascience.com/image-classification-in-10-minutes-with-mnist-dataset-54c35b77a38d>)