In [ ]:
```
pip install --upgrade ipympl
```

In [3]:
```
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.activations import linear, relu, sigmoid
%matplotlib widget
import matplotlib.pyplot as plt
plt.style.use('./deeplearning.mplstyle')
```

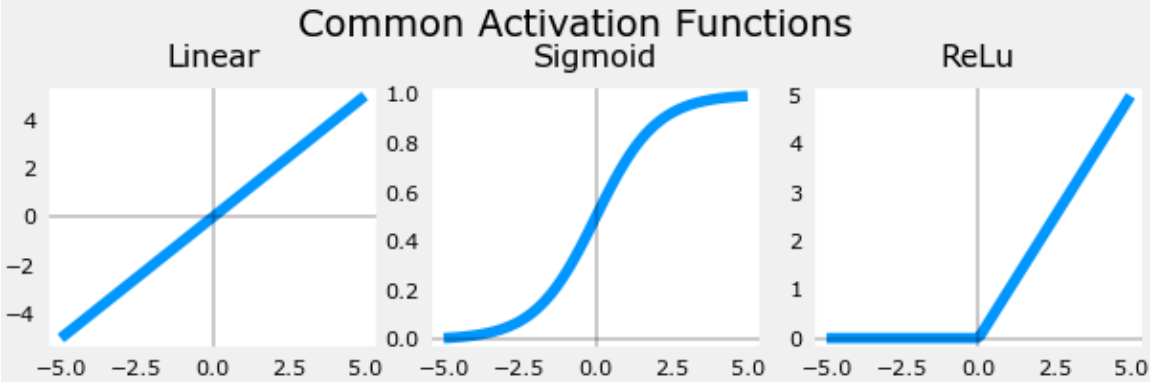
In [4]:
```
import logging
logging.getLogger("tensorflow").setLevel(logging.ERROR)
tf.autograph.set_verbosity(0)
```

In [5]:
```
from public_tests import *
```

In [7]:
```
from autils import*
from lab_utils_softmax import plt_softmax
np.set_printoptions(precision=2)
```

In [8]:
```
plt_act_trio()
```

Figure



In [9]:
```
def my_softmax(z):
    ez=np.exp(z)
    a=ez/np.sum(ez)
    return a
```

In [11]:
```
z=np.array([1,2,3,4])
a=my_softmax(z)
print(a)
```

```
[0.03 0.09 0.24 0.64]
```

In [12]:
```
X,y =load_data()
```

In [14]:
```
print("The first element of X is :", X[0])
```

```
The first element of X is : [ 0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00
e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  8.56e-06  1.94e-06 -7.37e-04
 -8.13e-03 -1.86e-02 -1.87e-02 -1.88e-02 -1.91e-02 -1.64e-02 -3.78e-03
  3.30e-04  1.28e-05  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  1.16e-04  1.20e-04 -1.40e-02 -2.85e-02  8.04e-02
  2.67e-01  2.74e-01  2.79e-01  2.74e-01  2.25e-01  2.78e-02 -7.06e-03
  2.35e-04  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  1.28e-17 -3.26e-04 -1.39e-02  8.16e-02  3.83e-01  8.58e-01  1.00e+00
  9.70e-01  9.31e-01  1.00e+00  9.64e-01  4.49e-01 -5.60e-03 -3.78e-03
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  5.11e-06  4.36e-04 -3.96e-03
 -2.69e-02  1.01e-01  6.42e-01  1.03e+00  8.51e-01  5.43e-01  3.43e-01
  2.69e-01  6.68e-01  1.01e+00  9.04e-01  1.04e-01 -1.66e-02  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  2.60e-05 -3.11e-03  7.52e-03  1.78e-01
  7.93e-01  9.66e-01  4.63e-01  6.92e-02 -3.64e-03 -4.12e-02 -5.02e-02
  1.56e-01  9.02e-01  1.05e+00  1.51e-01 -2.16e-02  0.00e+00  0.00e+00
  0.00e+00  5.87e-05 -6.41e-04 -3.23e-02  2.78e-01  9.37e-01  1.04e+00
  5.98e-01 -3.59e-03 -2.17e-02 -4.81e-03  6.17e-05 -1.24e-02  1.55e-01
  9.15e-01  9.20e-01  1.09e-01 -1.71e-02  0.00e+00  0.00e+00  1.56e-04
 -4.28e-04 -2.51e-02  1.31e-01  7.82e-01  1.03e+00  7.57e-01  2.85e-01
  4.87e-03 -3.19e-03  0.00e+00  8.36e-04 -3.71e-02  4.53e-01  1.03e+00
  5.39e-01 -2.44e-03 -4.80e-03  0.00e+00  0.00e+00 -7.04e-04 -1.27e-02
  1.62e-01  7.80e-01  1.04e+00  8.04e-01  1.61e-01 -1.38e-02  2.15e-03
 -2.13e-04  2.04e-04 -6.86e-03  4.32e-04  7.21e-01  8.48e-01  1.51e-01
 -2.28e-02  1.99e-04  0.00e+00  0.00e+00 -9.40e-03  3.75e-02  6.94e-01
  1.03e+00  1.02e+00  8.80e-01  3.92e-01 -1.74e-02 -1.20e-04  5.55e-05
 -2.24e-03 -2.76e-02  3.69e-01  9.36e-01  4.59e-01 -4.25e-02  1.17e-03
  1.89e-05  0.00e+00  0.00e+00 -1.94e-02  1.30e-01  9.80e-01  9.42e-01
  7.75e-01  8.74e-01  2.13e-01 -1.72e-02  0.00e+00  1.10e-03 -2.62e-02
  1.23e-01  8.31e-01  7.27e-01  5.24e-02 -6.19e-03  0.00e+00  0.00e+00
  0.00e+00  0.00e+00 -9.37e-03  3.68e-02  6.99e-01  1.00e+00  6.06e-01
  3.27e-01 -3.22e-02 -4.83e-02 -4.34e-02 -5.75e-02  9.56e-02  7.27e-01
  6.95e-01  1.47e-01 -1.20e-02 -3.03e-04  0.00e+00  0.00e+00  0.00e+00
  0.00e+00 -6.77e-04 -6.51e-03  1.17e-01  4.22e-01  9.93e-01  8.82e-01
  7.46e-01  7.24e-01  7.23e-01  7.20e-01  8.45e-01  8.32e-01  6.89e-02
 -2.78e-02  3.59e-04  7.15e-05  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  1.53e-04  3.17e-04 -2.29e-02 -4.14e-03  3.87e-01  5.05e-01  7.75e-01
  9.90e-01  1.01e+00  1.01e+00  7.38e-01  2.15e-01 -2.70e-02  1.33e-03
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  2.36e-04 -2.26e-03 -2.52e-02 -3.74e-02  6.62e-02  2.91e-01
  3.23e-01  3.06e-01  8.76e-02 -2.51e-02  2.37e-04  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  6.21e-18  6.73e-04 -1.13e-02 -3.55e-02 -3.88e-02
 -3.71e-02 -1.34e-02  9.91e-04  4.89e-05  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00  0.00e+00
  0.00e+00]
```

In [15]:
```
print("The first element of X is :", y[0,0])
print("The last element of y is :", y[-1,0])
```

```
The first element of X is : 0
The last element of y is : 9
```

In [17]:
```
str(X.shape)
```

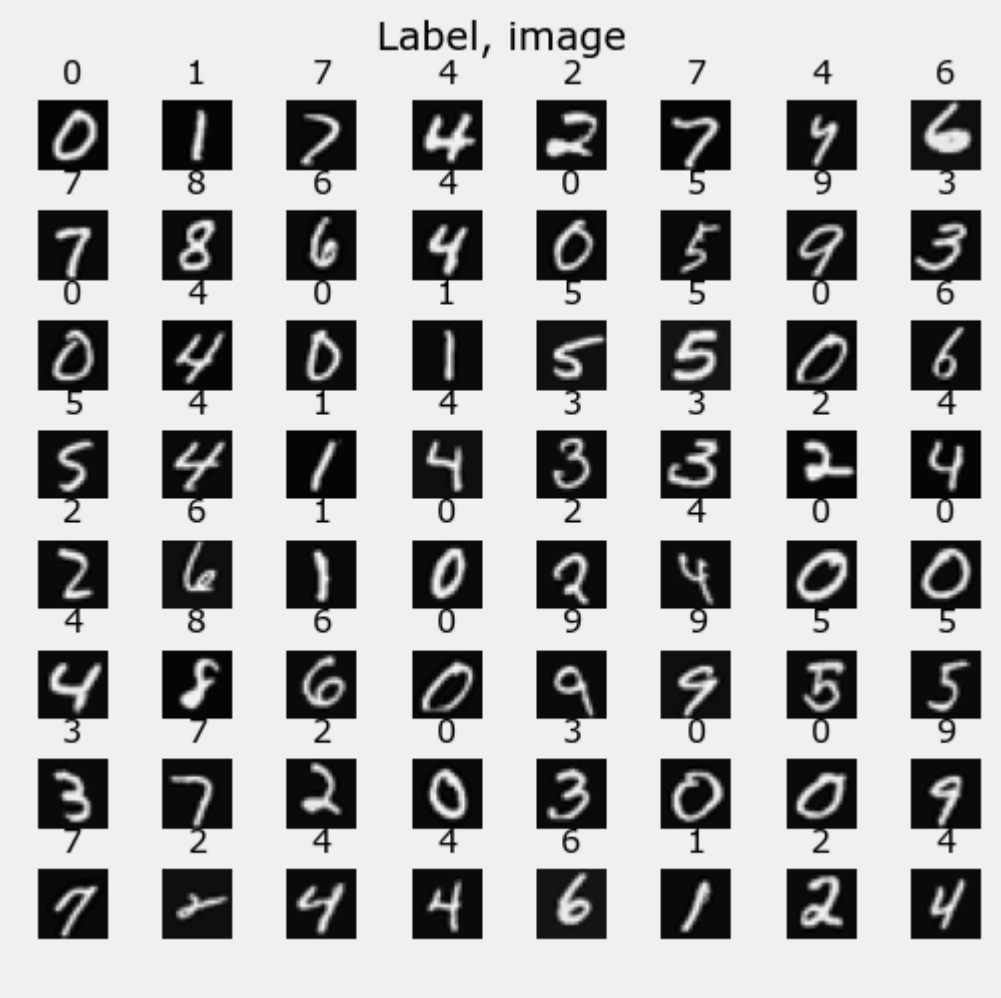
Out[17]:
```
'(5000, 400)'
```

In [19]:
```
str(y.shape)
```

Out[19]:
```
'(5000, 1)'
```

In [20]:
```
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
# You do not need to modify anything in this cell

m, n = X.shape

fig, axes = plt.subplots(8,8, figsize=(5,5))
fig.tight_layout(pad=0.13,rect=[0, 0.03, 1, 0.91]) #[left, bottom, right, to

#fig.tight_layout(pad=0.5)
widgvis(fig)
for i,ax in enumerate(axes.flat):
    # Select random indices
    random_index = np.random.randint(m)

    # Select rows corresponding to the random indices and
    # reshape the image
    X_random_reshaped = X[random_index].reshape((20,20)).T

    # Display the image
    ax.imshow(X_random_reshaped, cmap='gray')

    # Display the label above the image
    ax.set_title(y[random_index,0])
    ax.set_axis_off()
    fig.suptitle("Label, image", fontsize=14)
```

Figure



In [23]:
```
tf.random.set_seed(1234)
model=Sequential(
[
    tf.keras.layers.InputLayer((400)),
    tf.keras.layers.Dense(25, activation="relu", name="L1"),
    tf.keras.layers.Dense(15, activation="relu", name="L2"),
    tf.keras.layers.Dense(10, activation="linear", name="L3")
], name="my_model"
)
model.compile(loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits
```

In [24]:
```
model.summary()
```

```
Model: "my_model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 L1 (Dense)                  (None, 25)                10025

 L2 (Dense)                  (None, 15)                390

 L3 (Dense)                  (None, 10)                160

=================================================================
Total params: 10575 (41.31 KB)
Trainable params: 10575 (41.31 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [27]:
```
W1,b1 = layer1.get_weights()
W2,b2 = layer2.get_weights()
W3,b3 = layer3.get_weights()
print(f"W1 shape = {W1.shape}, b1 shape = {b1.shape}")
print(f"W2 shape = {W2.shape}, b2 shape = {b2.shape}")
print(f"W3 shape = {W3.shape}, b3 shape = {b3.shape}")
```

```
W1 shape = (400, 25), b1 shape = (25,)
W2 shape = (25, 15), b2 shape = (15,)
W3 shape = (15, 10), b3 shape = (10,)
```

[layer1, layer2, layer3] = model.layers

In [30]:
```
model.compile(
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
    optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
)
history=model.fit(X,y, epochs=40)
```

```
Epoch 1/40
157/157 [==============================] - 1s 1ms/step - loss: 1.4380
Epoch 2/40
157/157 [==============================] - 0s 884us/step - loss: 0.5948
Epoch 3/40
157/157 [==============================] - 0s 872us/step - loss: 0.4125
Epoch 4/40
157/157 [==============================] - 0s 861us/step - loss: 0.3407
Epoch 5/40
157/157 [==============================] - 0s 882us/step - loss: 0.3005
Epoch 6/40
157/157 [==============================] - 0s 853us/step - loss: 0.2725
Epoch 7/40
157/157 [==============================] - 0s 854us/step - loss: 0.2494
Epoch 8/40
157/157 [==============================] - 0s 848us/step - loss: 0.2295
Epoch 9/40
157/157 [==============================] - 0s 863us/step - loss: 0.2168
Epoch 10/40
157/157 [==============================] - 0s 872us/step - loss: 0.2001
Epoch 11/40
157/157 [==============================] - 0s 897us/step - loss: 0.1873
Epoch 12/40
157/157 [==============================] - 0s 949us/step - loss: 0.1772
Epoch 13/40
157/157 [==============================] - 0s 930us/step - loss: 0.1676
Epoch 14/40
157/157 [==============================] - 0s 856us/step - loss: 0.1586
Epoch 15/40
157/157 [==============================] - 0s 892us/step - loss: 0.1496
Epoch 16/40
157/157 [==============================] - 0s 998us/step - loss: 0.1430
Epoch 17/40
157/157 [==============================] - 0s 915us/step - loss: 0.1335
Epoch 18/40
157/157 [==============================] - 0s 984us/step - loss: 0.1277
Epoch 19/40
157/157 [==============================] - 0s 898us/step - loss: 0.1207
Epoch 20/40
157/157 [==============================] - 0s 934us/step - loss: 0.1147
Epoch 21/40
157/157 [==============================] - 0s 935us/step - loss: 0.1105
Epoch 22/40
157/157 [==============================] - 0s 939us/step - loss: 0.1042
Epoch 23/40
157/157 [==============================] - 0s 912us/step - loss: 0.0993
Epoch 24/40
157/157 [==============================] - 0s 883us/step - loss: 0.0945
Epoch 25/40
157/157 [==============================] - 0s 873us/step - loss: 0.0906
Epoch 26/40
157/157 [==============================] - 0s 860us/step - loss: 0.0846
Epoch 27/40
157/157 [==============================] - 0s 864us/step - loss: 0.0805
Epoch 28/40
157/157 [==============================] - 0s 857us/step - loss: 0.0751
Epoch 29/40
157/157 [==============================] - 0s 892us/step - loss: 0.0696
Epoch 30/40
157/157 [==============================] - 0s 927us/step - loss: 0.0691
Epoch 31/40
```

```
157/157 [==============================] - 0s 929us/step - loss: 0.0624
Epoch 32/40
157/157 [==============================] - 0s 934us/step - loss: 0.0601
Epoch 33/40
157/157 [==============================] - 0s 897us/step - loss: 0.0573
Epoch 34/40
157/157 [==============================] - 0s 853us/step - loss: 0.0518
Epoch 35/40
157/157 [==============================] - 0s 954us/step - loss: 0.0499
Epoch 36/40
157/157 [==============================] - 0s 891us/step - loss: 0.0509
Epoch 37/40
157/157 [==============================] - 0s 906us/step - loss: 0.0443
Epoch 38/40
157/157 [==============================] - 0s 917us/step - loss: 0.0416
Epoch 39/40
157/157 [==============================] - 0s 901us/step - loss: 0.0388
Epoch 40/40
157/157 [==============================] - 0s 872us/step - loss: 0.0363
```

In [31]:

```
image_of_two=X[1015]
display_digit(image_of_two)
prediction=model.predict(image_of_two.reshape(1,400))
print(f" predicting a Two: \n{prediction}")
print(f" Largest Prediction index: {np.argmax(prediction)}")
```

Figure

```
1/1 [==============================] - 0s 74ms/step
 predicting a Two:
[[-10.18  -0.07   3.42  -0.18 -12.92  -5.98  -4.19   1.16  -5.36  -9.39]]
 Largest Prediction index: 2
```

In [ ]: