Learning the digits

You're going to build a model on the **digits dataset**, a sample dataset that comes pre-loaded with scikit learn. The **digits dataset** consist of **8x8 pixel handwritten digits from 0 to 9**:

You want to distinguish between each of the 10 possible digits given an image, so we are dealing with **multi-class classification**.

The dataset has already been partitioned into `X_train`, `y_train`, `X_test`, and `y_test` using 30% of the data as testing data. The labels are one-hot encoded vectors, so you don't need to use Keras `to_categorical()` function.

Let's build this new `model`!

- Add a `Dense` layer of 16 neurons with `relu` activation and `input_shape` being the **total number of pixels** of each image.
- Add a `Dense` layer with 10 outputs and `softmax` activation.
- Compile your model with `adam`, `categorical_crossentropy`, and `accuracy` metrics.
- Make sure your model works by predicting on `X_train`.

```
# Instantiate a Sequential model

model = Sequential()


# Input and hidden layer with input_shape, 16 neurons, and relu

model.add(Dense(16, input_shape = (64,), activation = 'relu'))


# Output layer with 10 neurons (one per digit) and softmax

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

```
# Compile your model
model.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])

# Test if your model works and can process input data
print(model.predict(X_train))
```