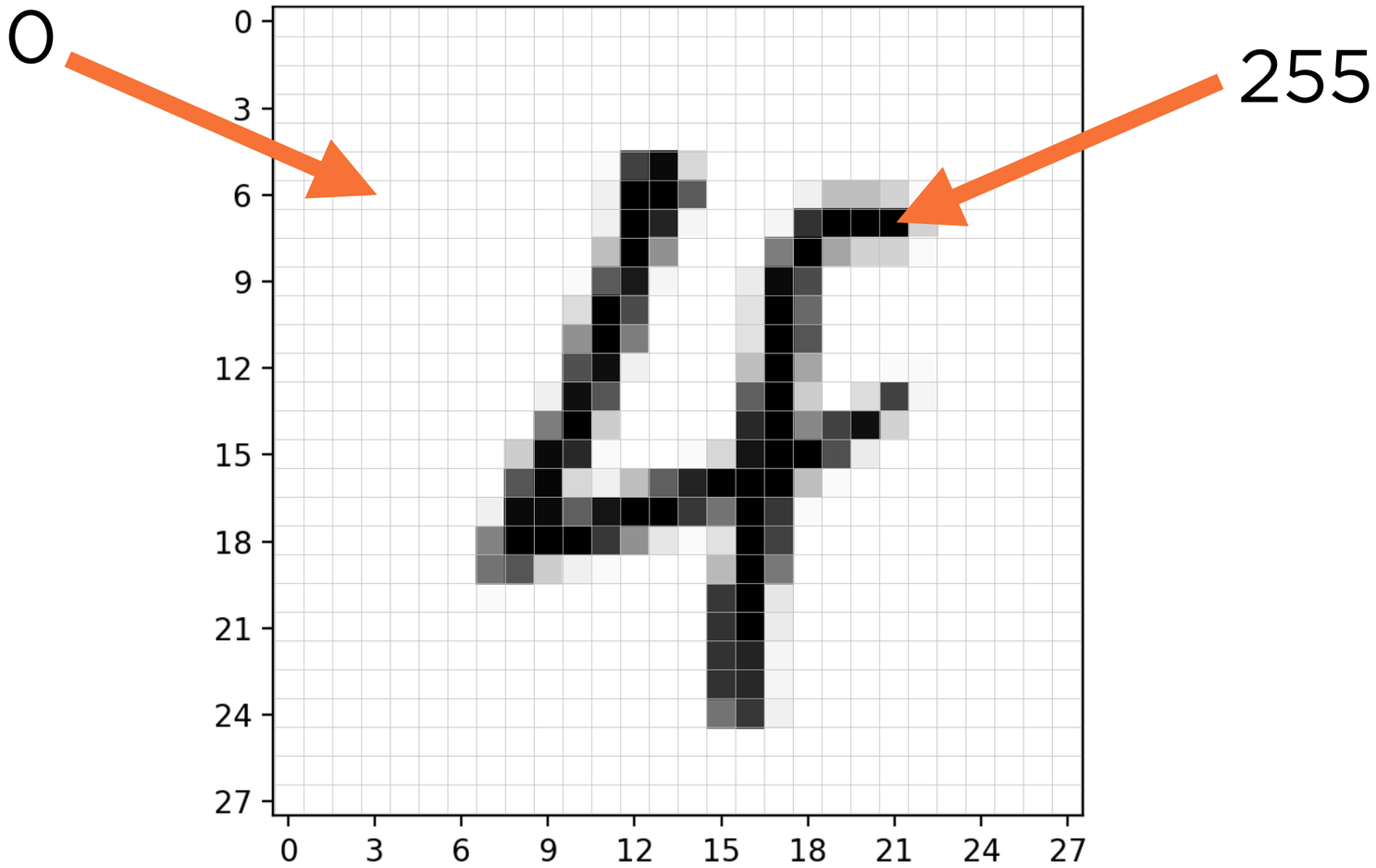
Recognizing Individual Digits

Paolo Perrotta
FREELANCE DEVELOPER
@nusco

MNIST Digits



Label: 4

Understanding the Test Set

A Fake Classifier

```
DICTIONARY = {}

def predict(X):
    result = []
    for image in X:
        result << DICTIONARY[image]
    return result

def train(X, Y):
    for image in X:
        DICTIONARY[image] = Y

train(MNIST_IMAGES, MNIST_LABELS)
predictions = predict(MNIST_IMAGES)
```

The Problem of Overfitting

Without Overfitting

The system *generalizes* the training data. When confronted with new data, it can still classify it.

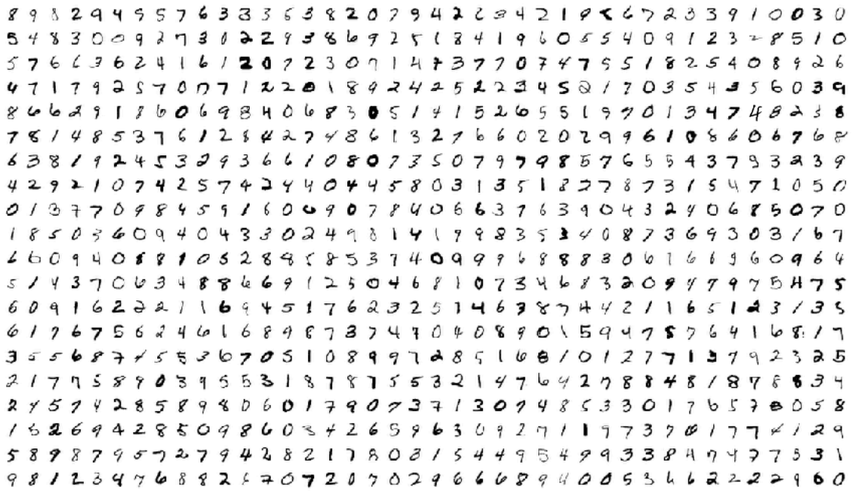
With Overfitting

The system *memorizes* the training data. When confronted with new data, it doesn't know what to do.

Never test a system on the same data that you used to train it.

Preparing MNIST Data for the Classifier

MNIST Images vs. the X Matrix



X

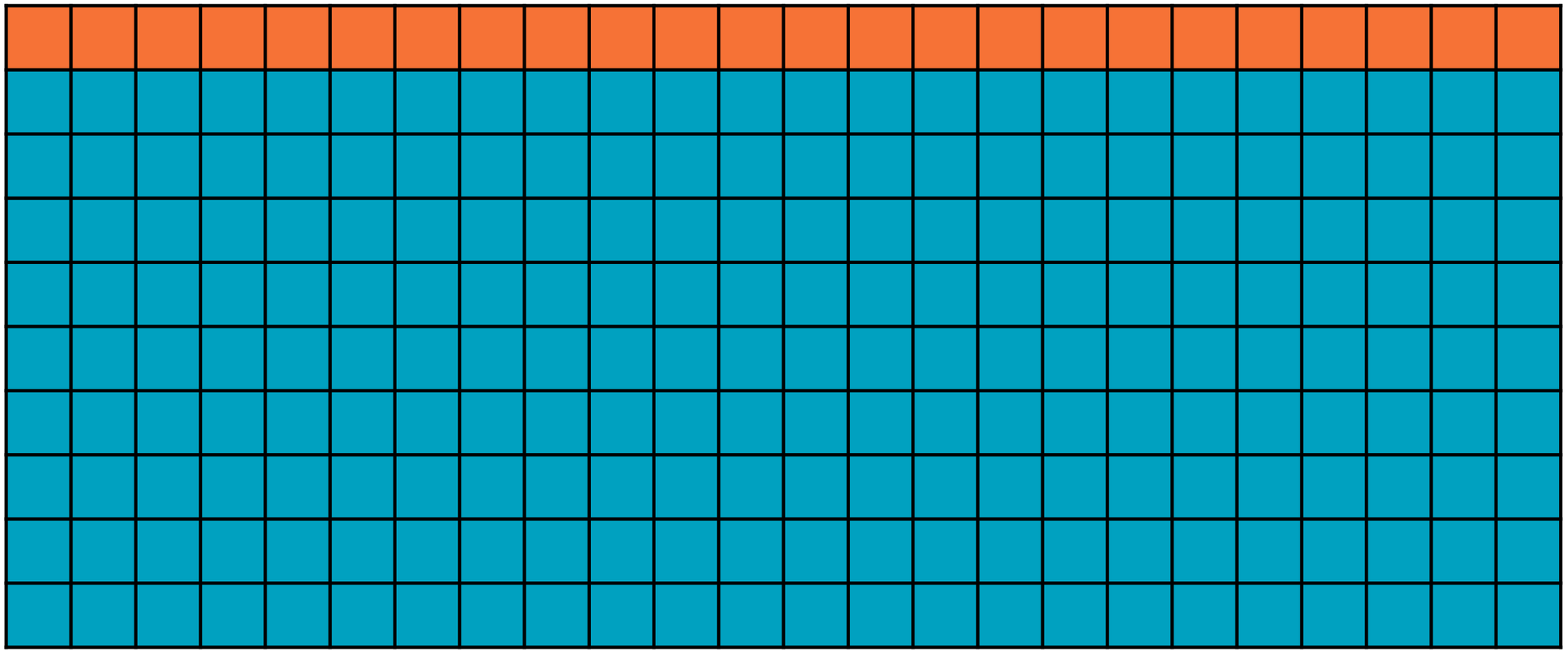
X_0	X_1	X_2	...
X_0	X_1	X_2	...
...	...	...	...

MNIST Images vs. the X Matrix

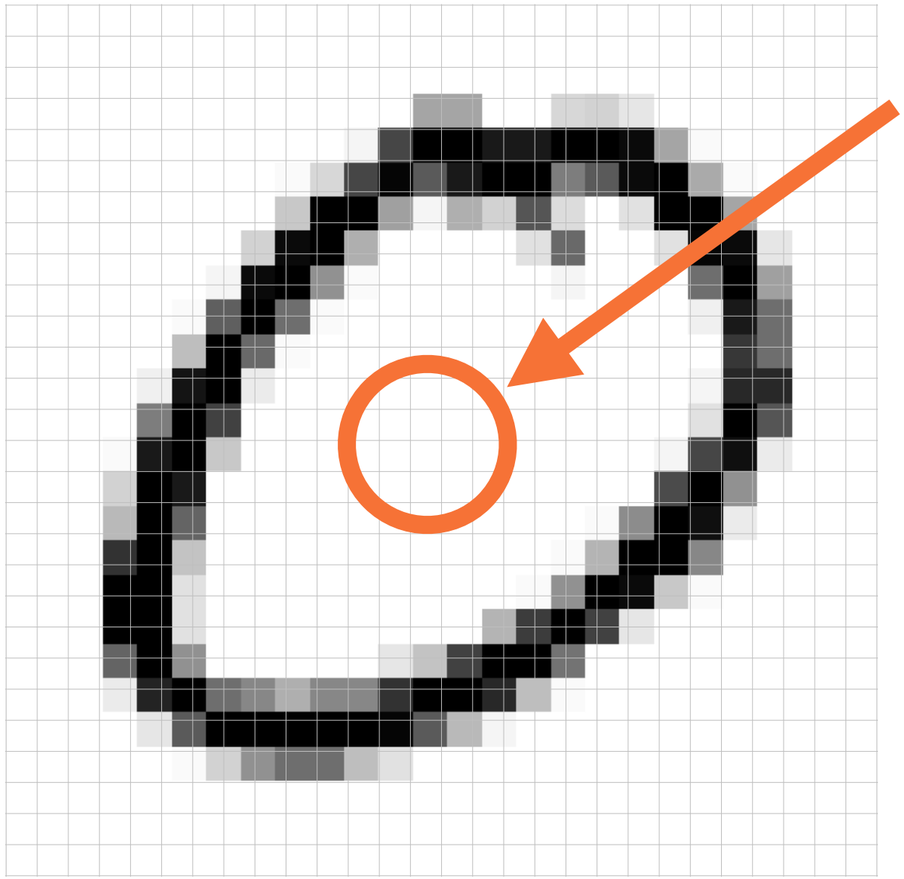
785
columns

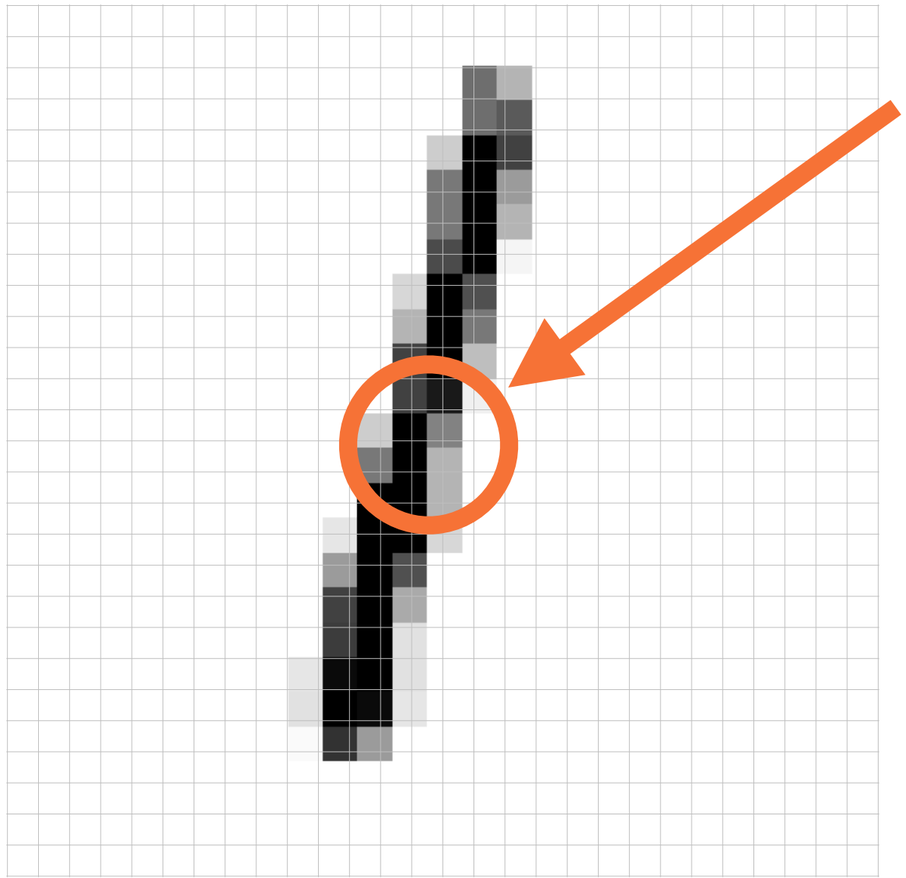
X

60K rows



Recognizing Images from Pixel Values



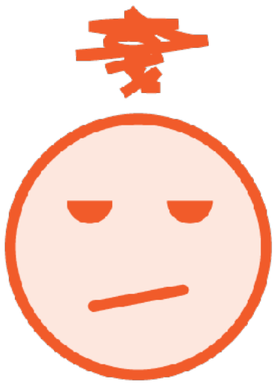


MNIST Labels vs. the Y Matrix

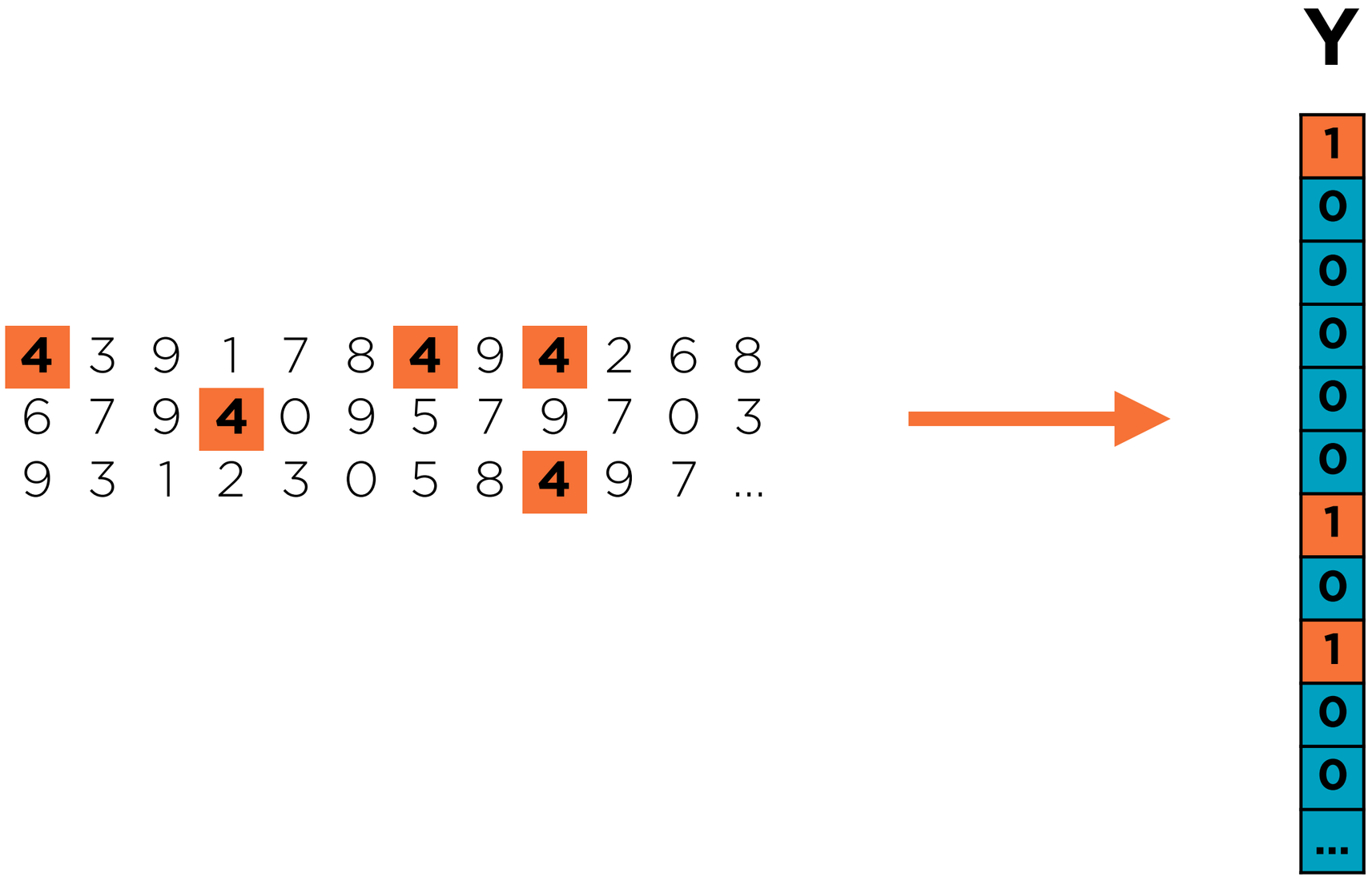
4 3 9 1 7 8 4 9 4 2 6 8
6 7 9 4 0 9 5 7 9 7 0 3
9 3 1 2 3 0 5 8 4 9 7 …

Y

4
3
9
1
7
8
4
9
4
2
6
…



MNIST Labels vs. the Y Matrix



Summary

We introduced the MNIST dataset

We learned what the test set is all about

We encoded MNIST for our classifier

- We flattened the images
- We encoded the “4” labels as 1s and the rest as 0s

We ran the binary classifier on MNIST