Using K-nearest-neighbors for Digit Recognition

Janani Ravi
CO-FOUNDER, LOONYCORN
www.loonycorn.com

Overview

Introduce the MNIST handwritten digit dataset

Understand the K-nearest-neighbors machine learning algorithm

Implement K-nearest-neighbors in TensorFlow to identify handwritten digits from 0 to 9

The MNIST Handwritten Digits Dataset

MNIST Dataset

Handwritten digits database

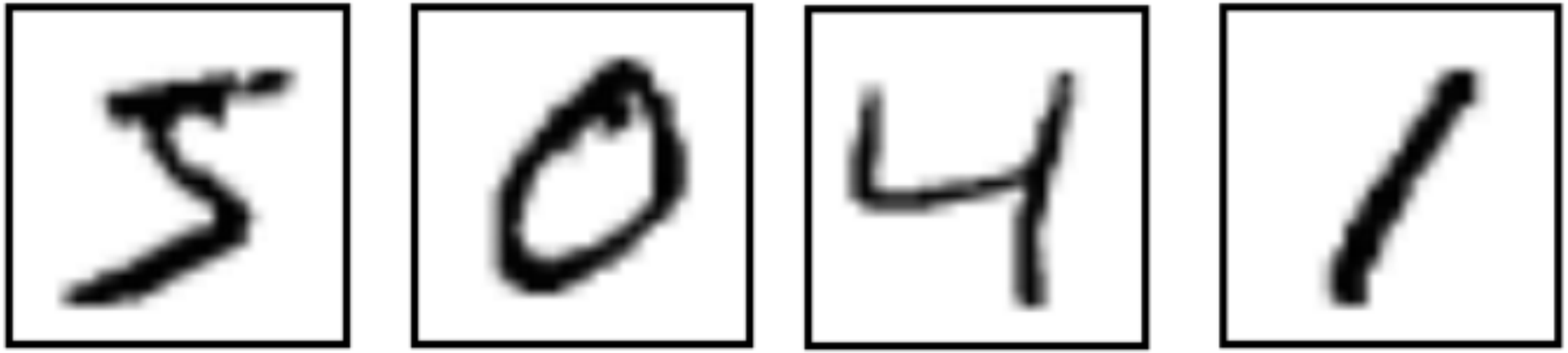
Large quantity of handwritten digits commonly used for training image processing systems

MNIST Dataset

Handwritten digits database

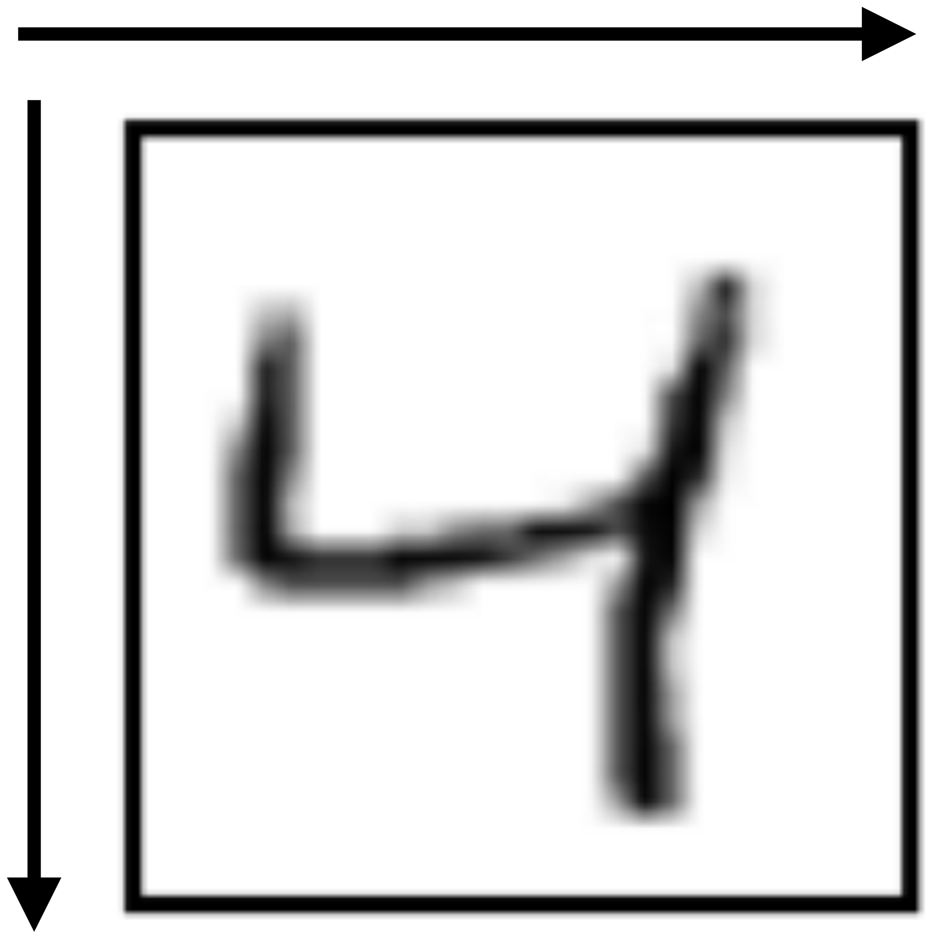
Modified National Institute of Standards and Technology

MNIST Dataset



Each digit is in grayscale

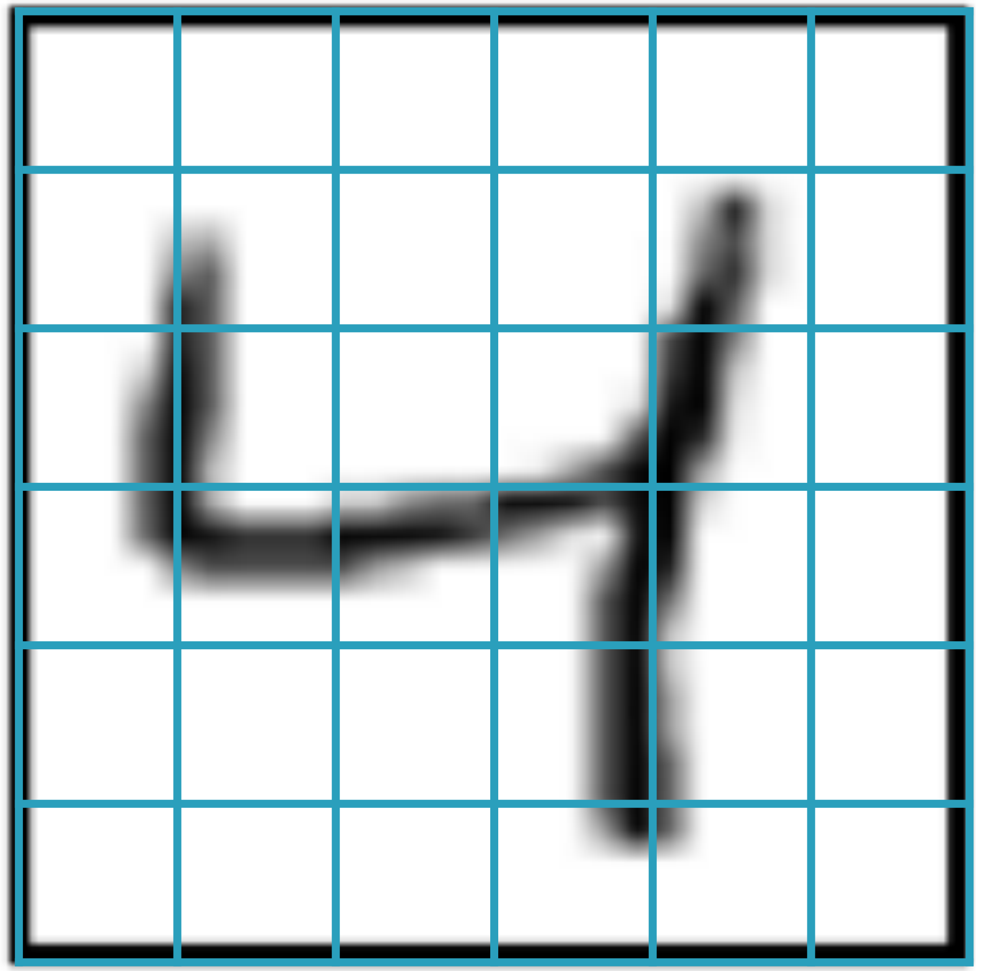
MNIST Dataset



Every image is standardized to be of size 28x28

= 784 pixels

MNIST Dataset



Every pixel holds a single value for intensity

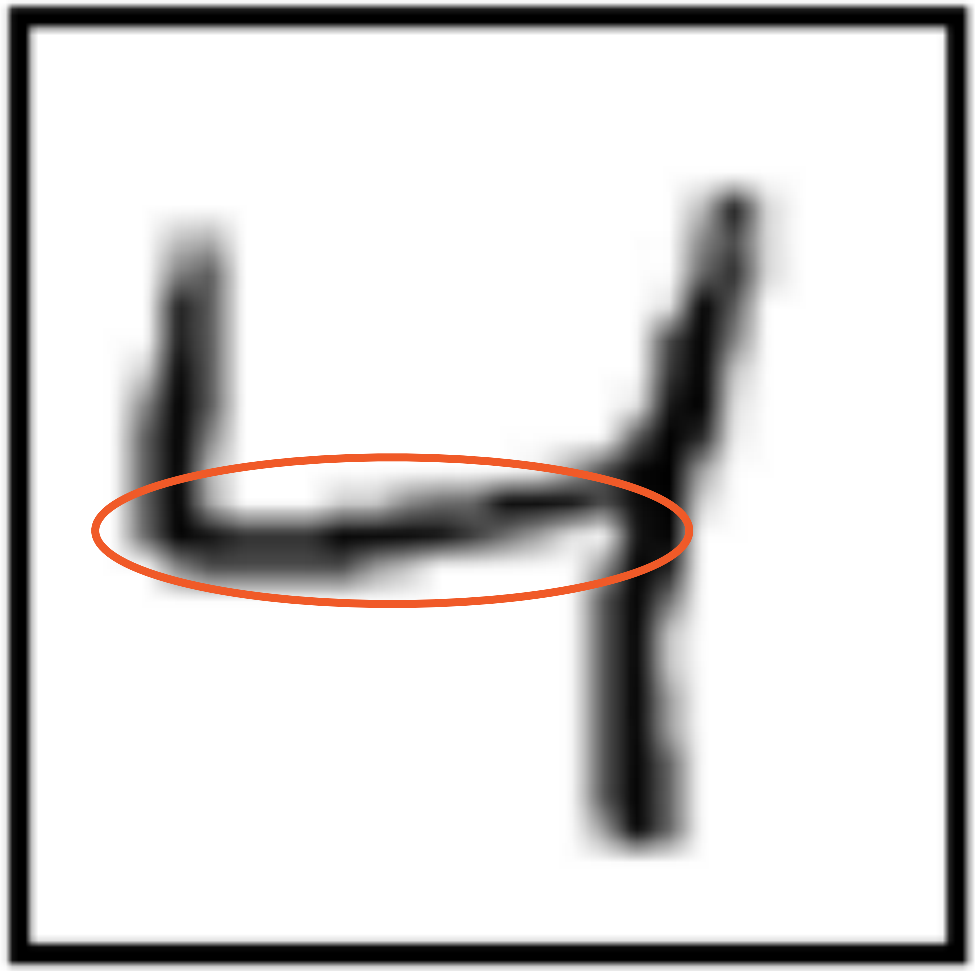
MNIST Dataset

0	0	0	0	0	0
0.2	0.8	0	0.3	0.6	0
0.2	0.9	0	0.3	0.8	0
0.3	0.8	0.7	0.8	0.9	0
0	0	0	0.2	0.8	0
0	0	0	0.2	0.2	0

MNIST Dataset

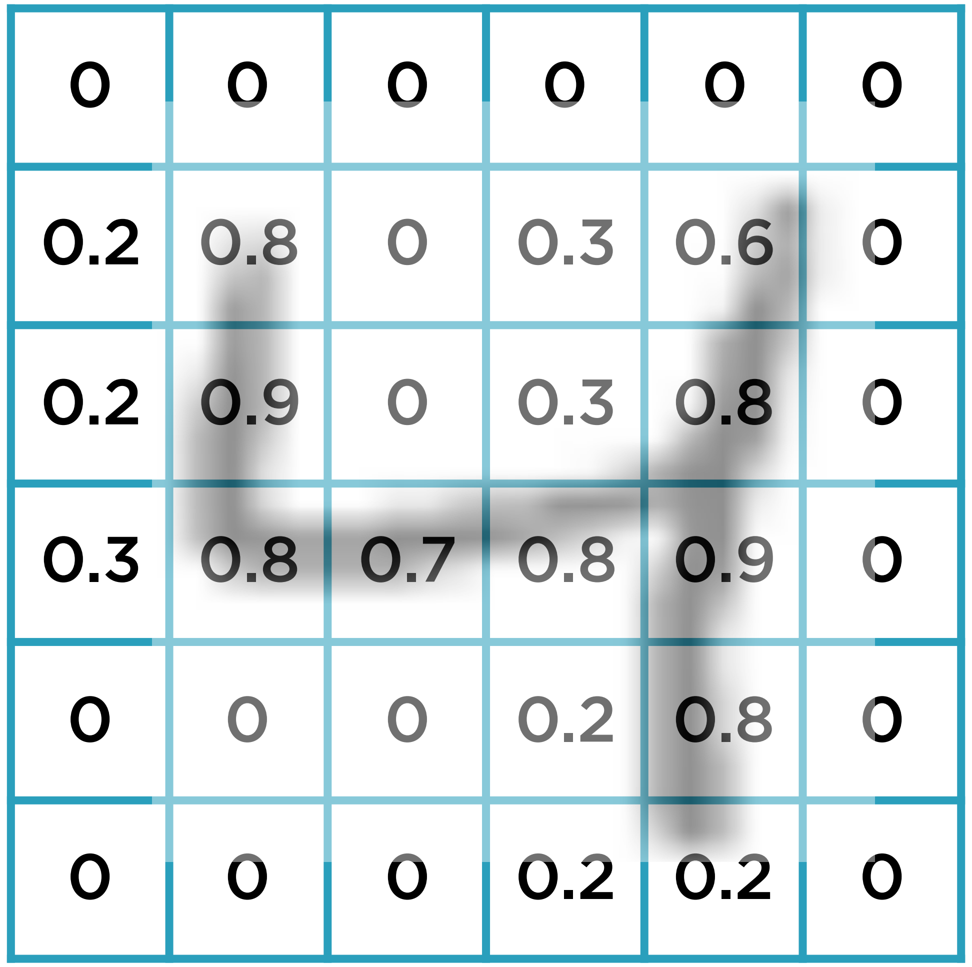
0	0	0	0	0	0
0.2	0.8	0	0.3	0.6	0
0.2	0.9	0	0.3	0.8	0
0.3	0.8	0.7	0.8	0.9	0
0	0	0	0.2	0.8	0
0	0	0	0.2	0.2	0

MNIST Dataset



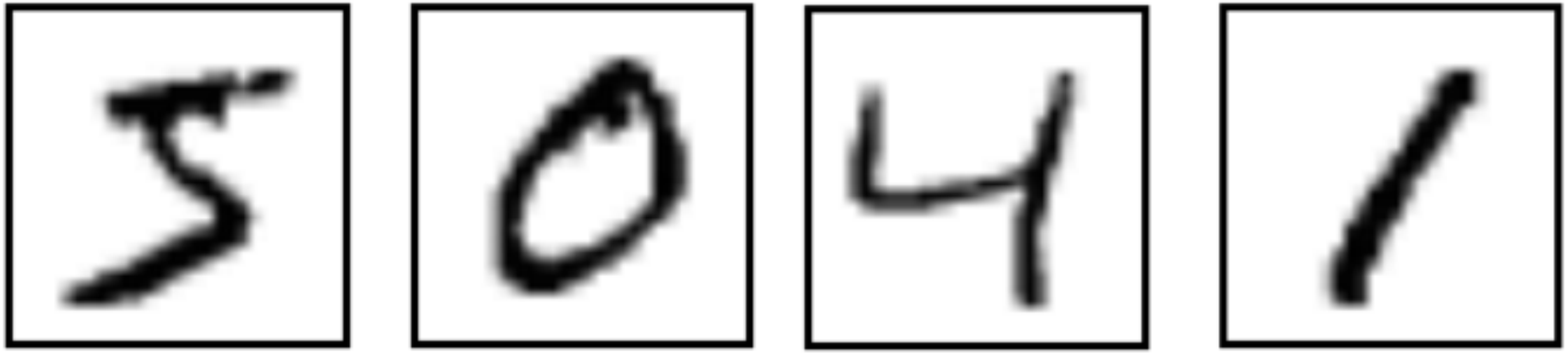
0	0	0	0	0	0
0.2	0.8	0	0.3	0.6	0
0.2	0.9	0	0.3	0.8	0
0.3	0.8	0.7	0.8	0.9	0
0	0	0	0.2	0.8	0
0	0	0	0.2	0.2	0

MNIST Dataset



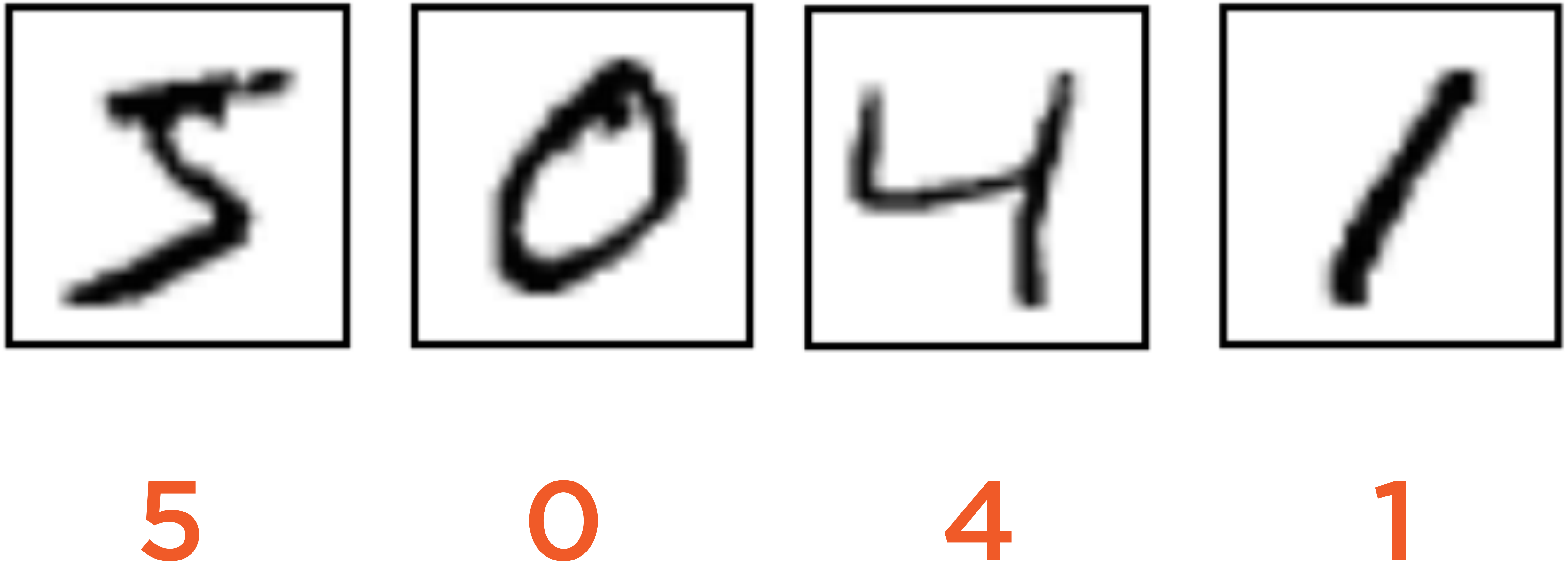
0	0	0	0	0	0
0.2	0.8	0	0.3	0.6	0
0.2	0.9	0	0.3	0.8	0
0.3	0.8	0.7	0.8	0.9	0
0	0	0	0.2	0.8	0
0	0	0	0.2	0.2	0

MNIST Dataset



Every image has an associated label

MNIST Dataset



5 0 4 1

MNIST for machine learning is the equivalent of the “Hello World” for programming

The K-nearest-neighbors Algorithm

Types of ML Algorithms

Supervised

Labels associated with the training data is used to correct the algorithm

Unsupervised

The model has to be set up right to learn structure in the data

Supervised Learning

Input variable x and output variable y

Learn the mapping function $y = f(x)$

Approximate the mapping function so for new values of x we can predict y

Use existing dataset to correct our mapping function approximation

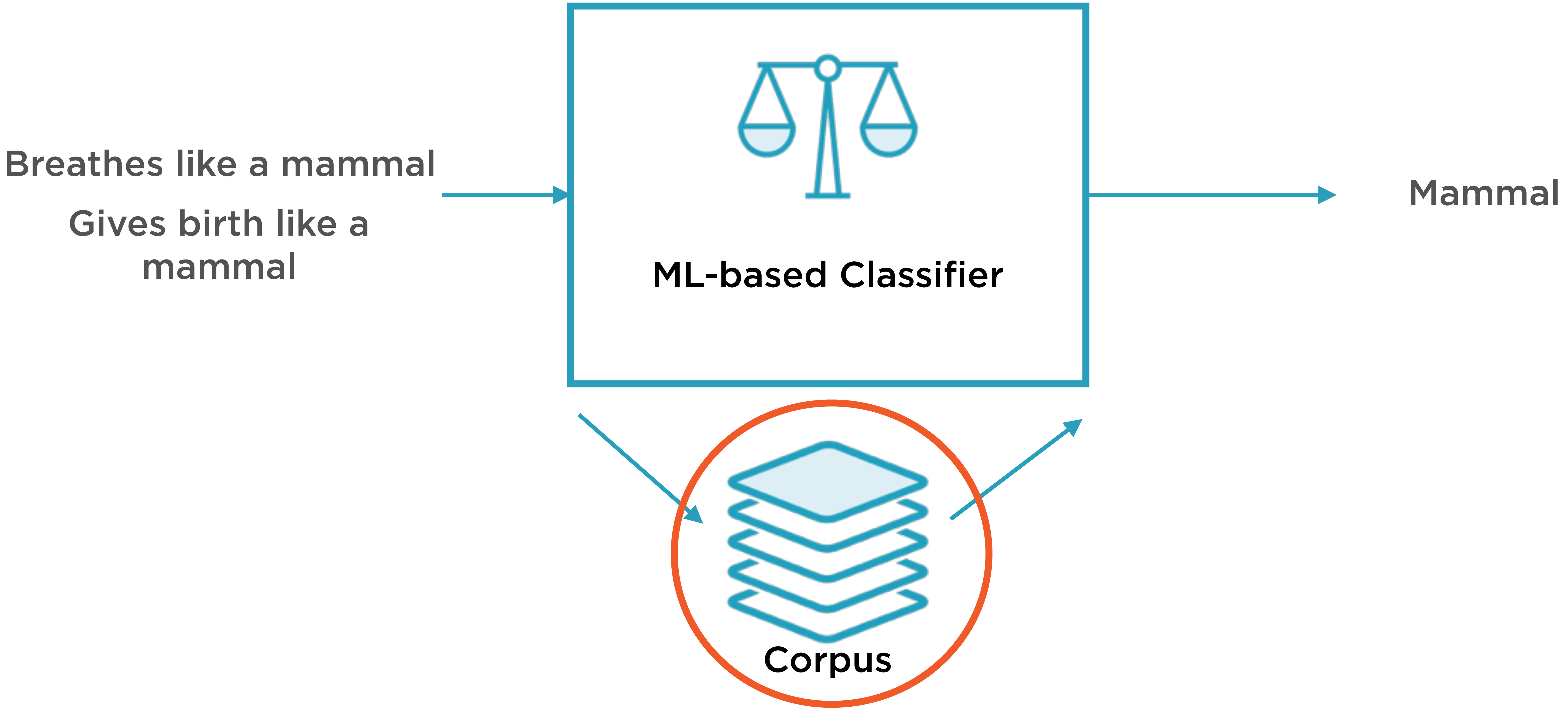
Unsupervised Learning

Only have input data x no output data

Model the underlying structure to learn more about data

Algorithms self discover the patterns and structure in the data

Training Data



Breathes like a mammal

Gives birth like a mammal

ML-based Classifier

Mammal

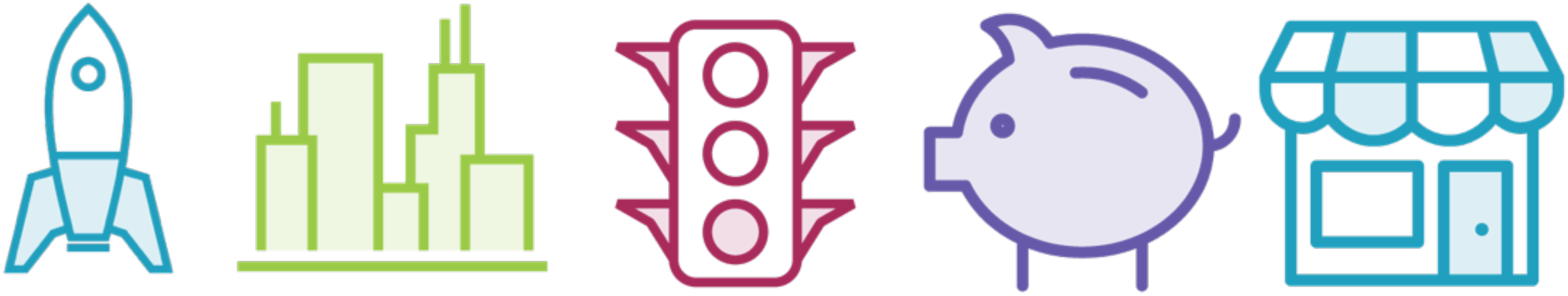
Corpus

KNN is an supervised learning algorithm which uses training data to find what is **most similar** to the current sample

K-nearest-neighbors

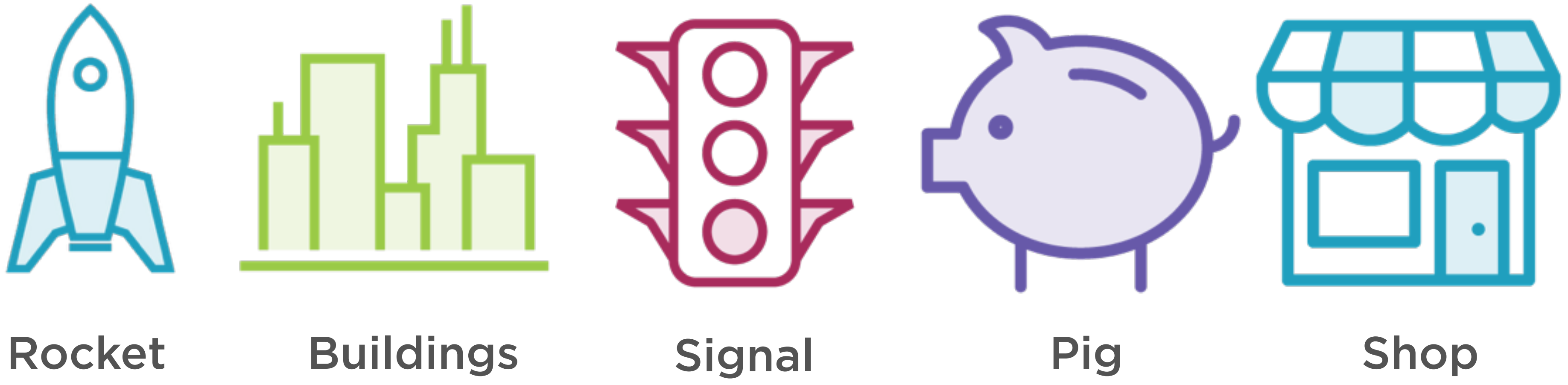
Uses the entire training dataset as a model

K-nearest-neighbors



Each element in training data has a label

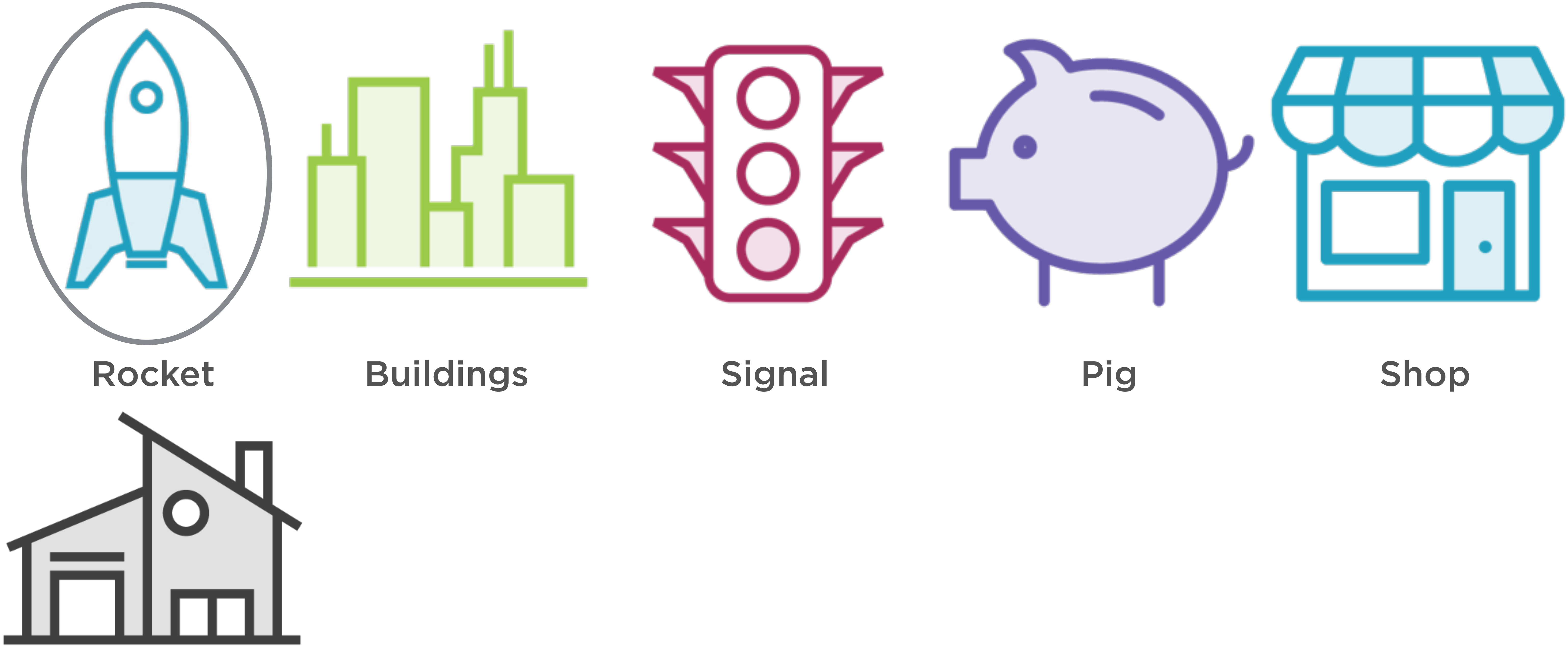
K-nearest-neighbors



Predictions for a new sample involves figuring out which element in the training data it is similar to

The nearest neighbor

K-nearest-neighbors



Rocket

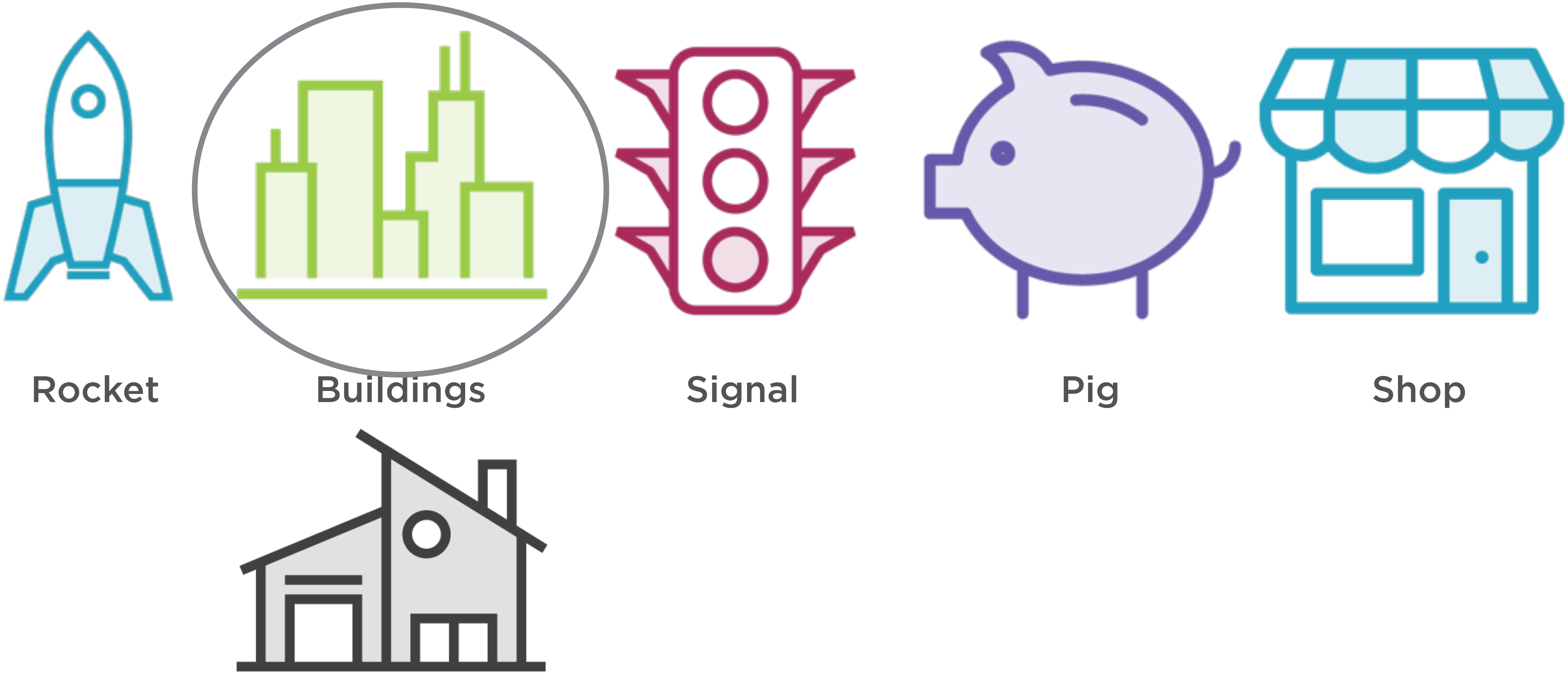
Buildings

Signal

Pig

Shop

K-nearest-neighbors



Rocket

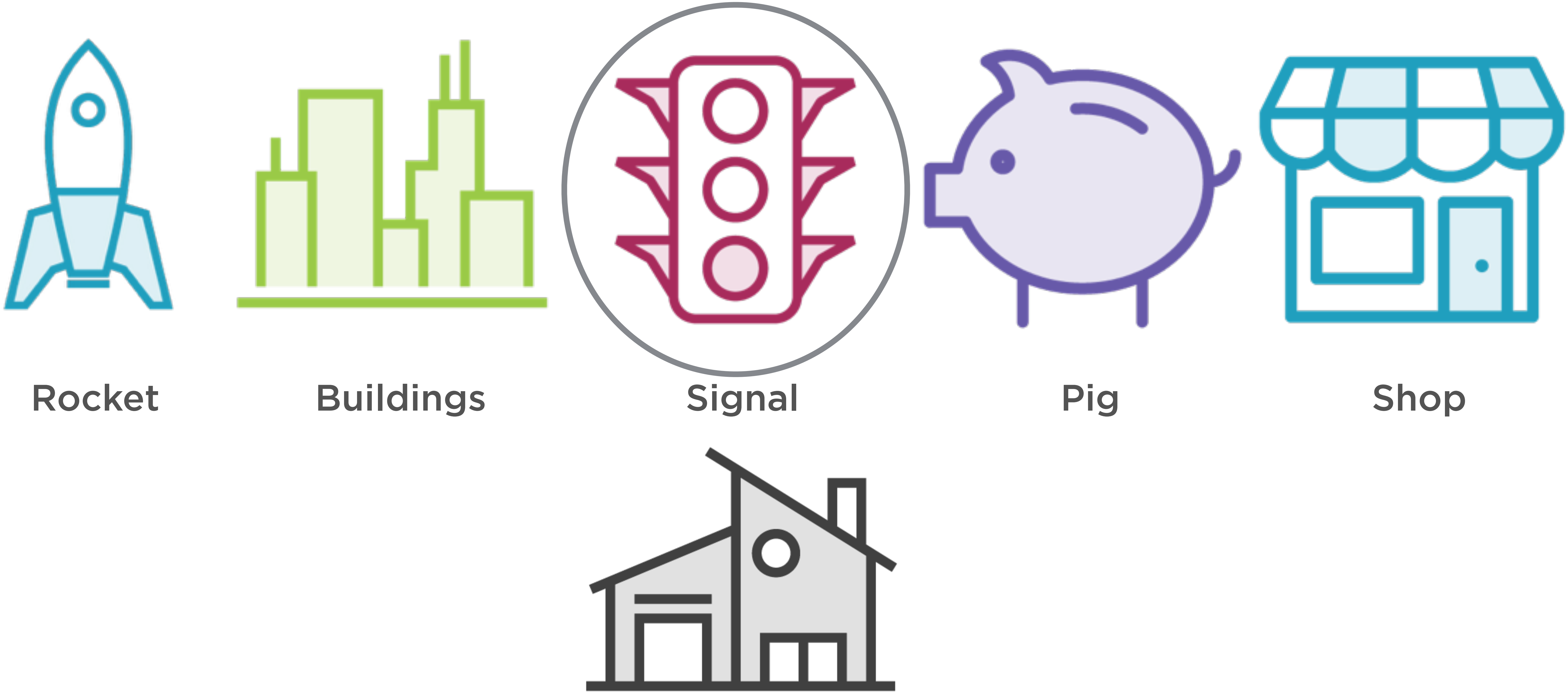
Buildings

Signal

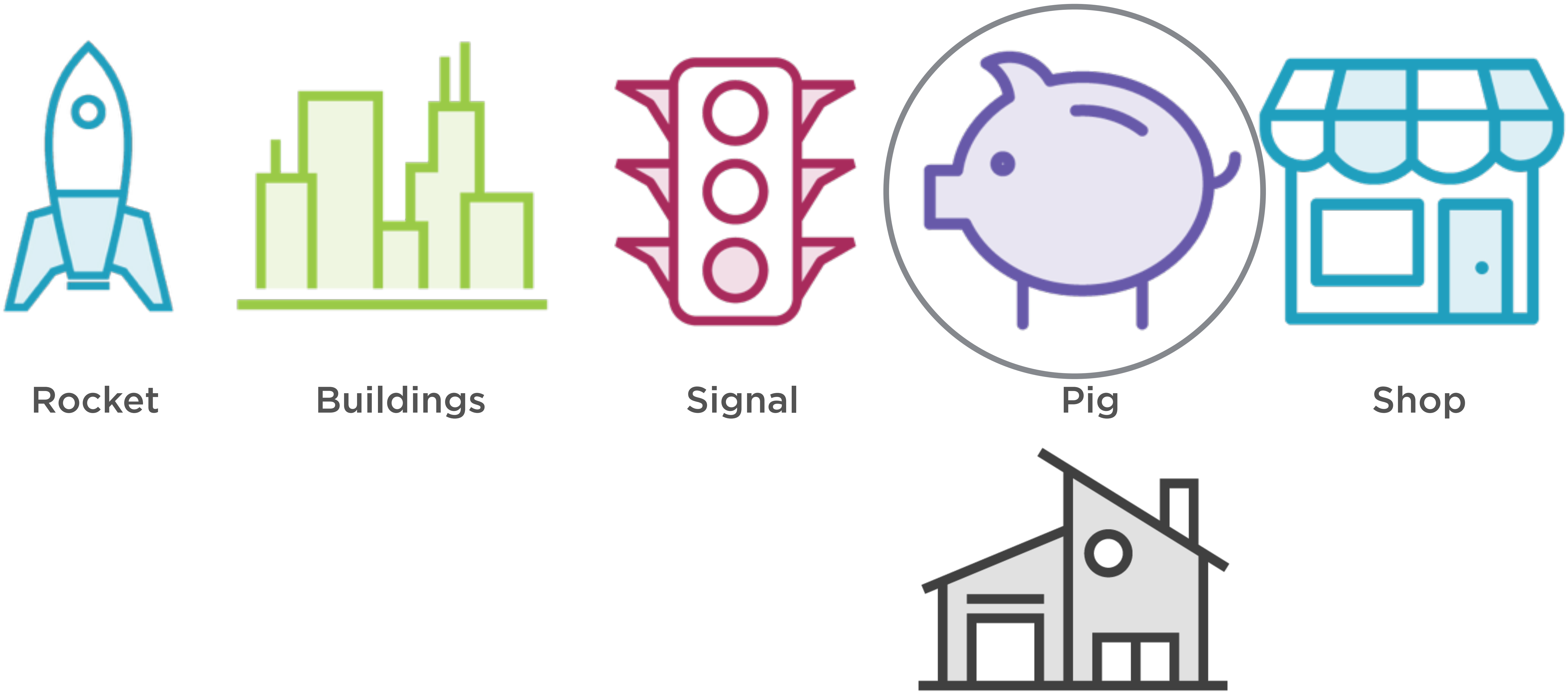
Pig

Shop

K-nearest-neighbors



K-nearest-neighbors



Rocket

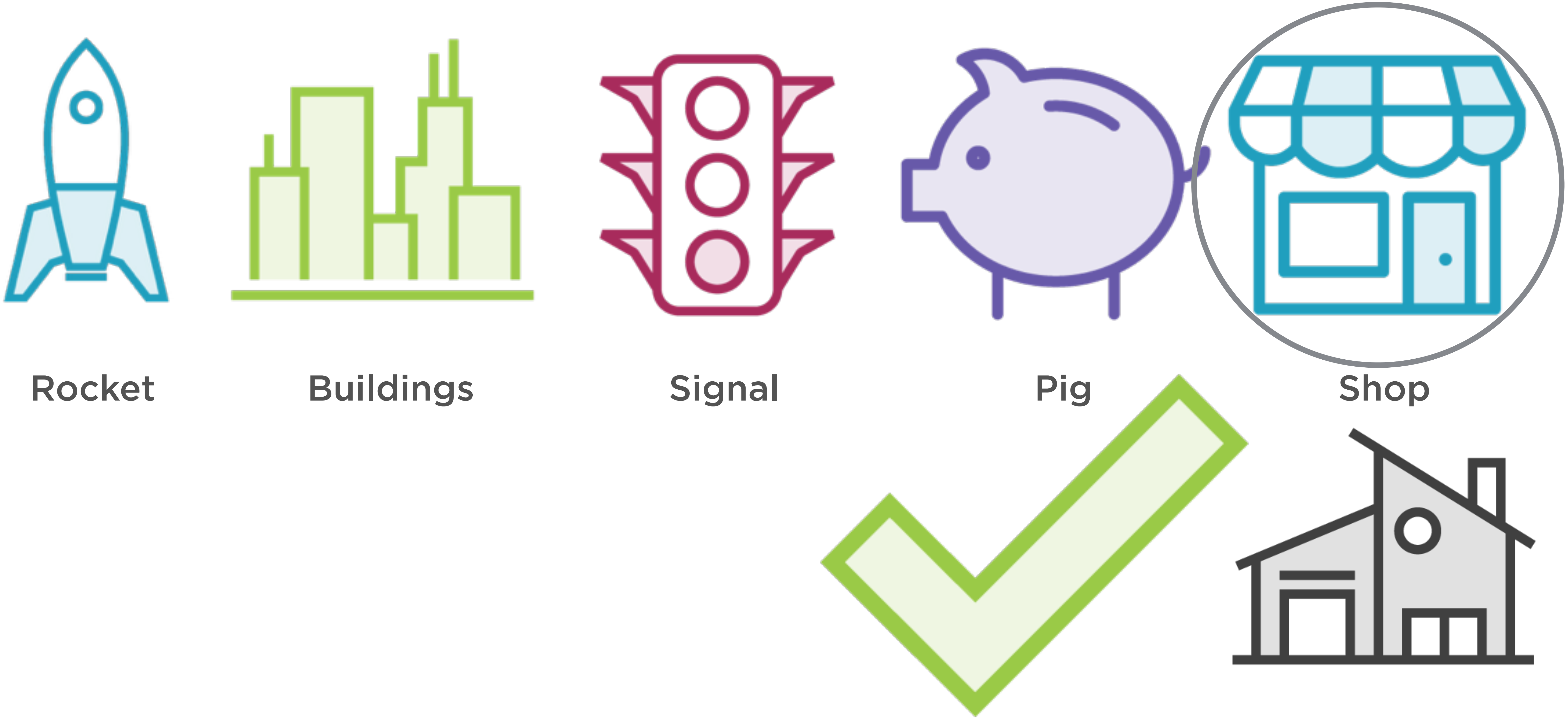
Buildings

Signal

Pig

Shop

K-nearest-neighbors

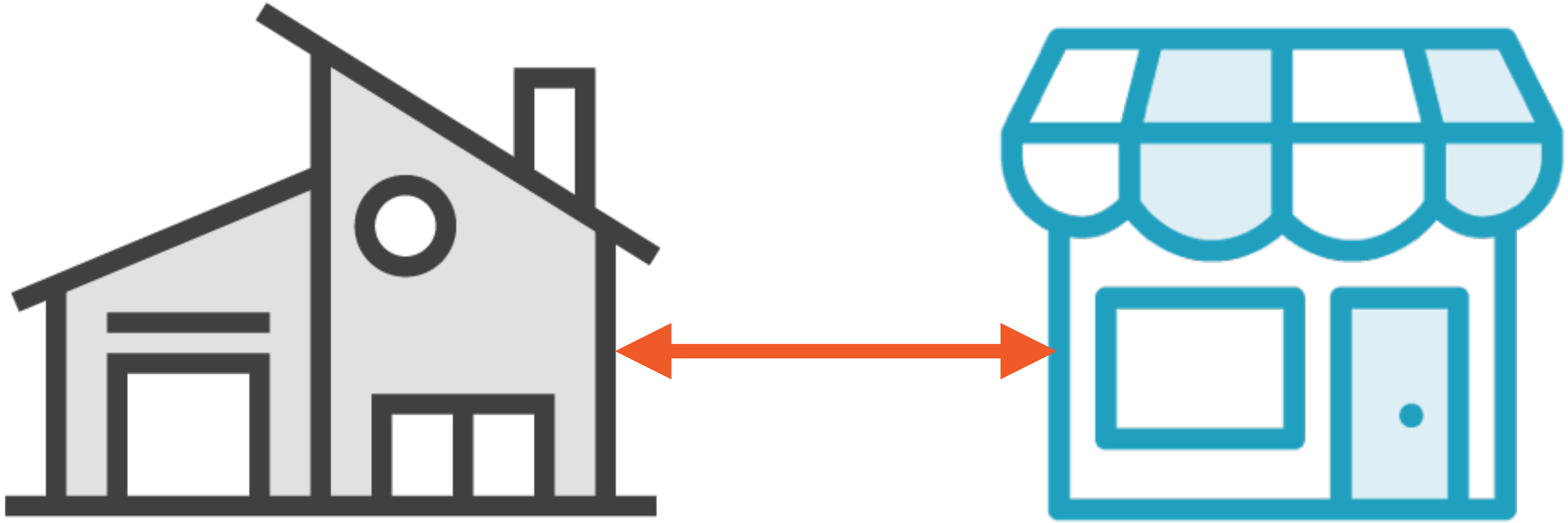


Rocket Buildings Signal Pig Shop

K-nearest-neighbors

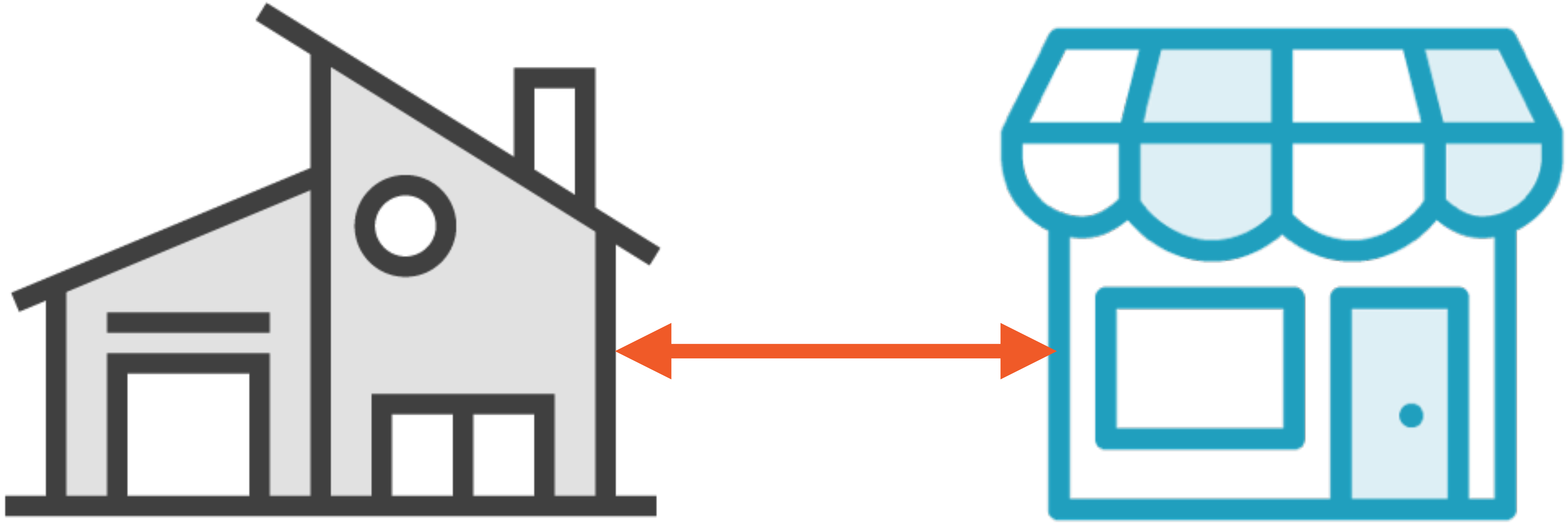
How do we calculate neighbors of a sample?

K-nearest-neighbors



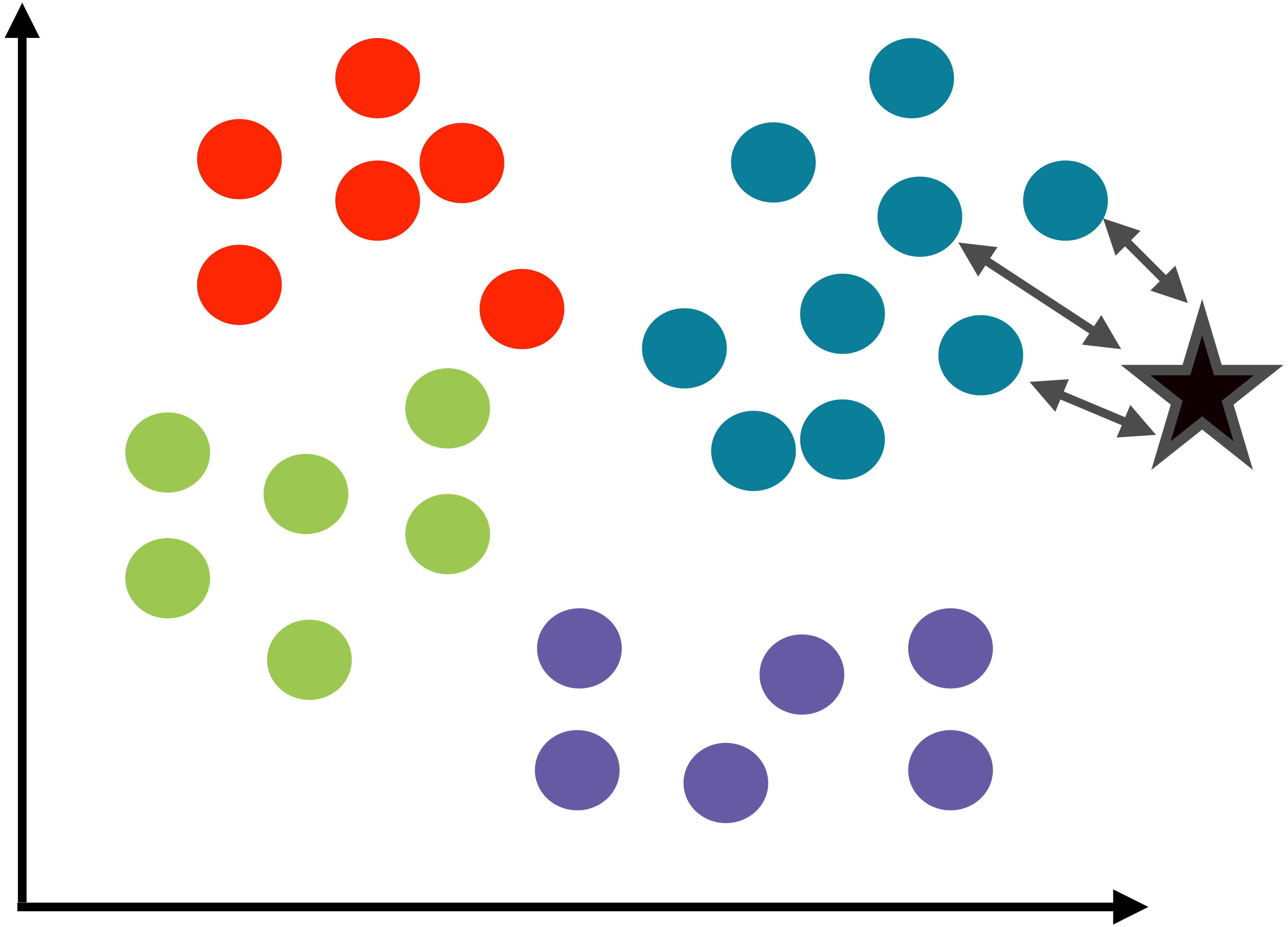
Distance measures

K-nearest-neighbors

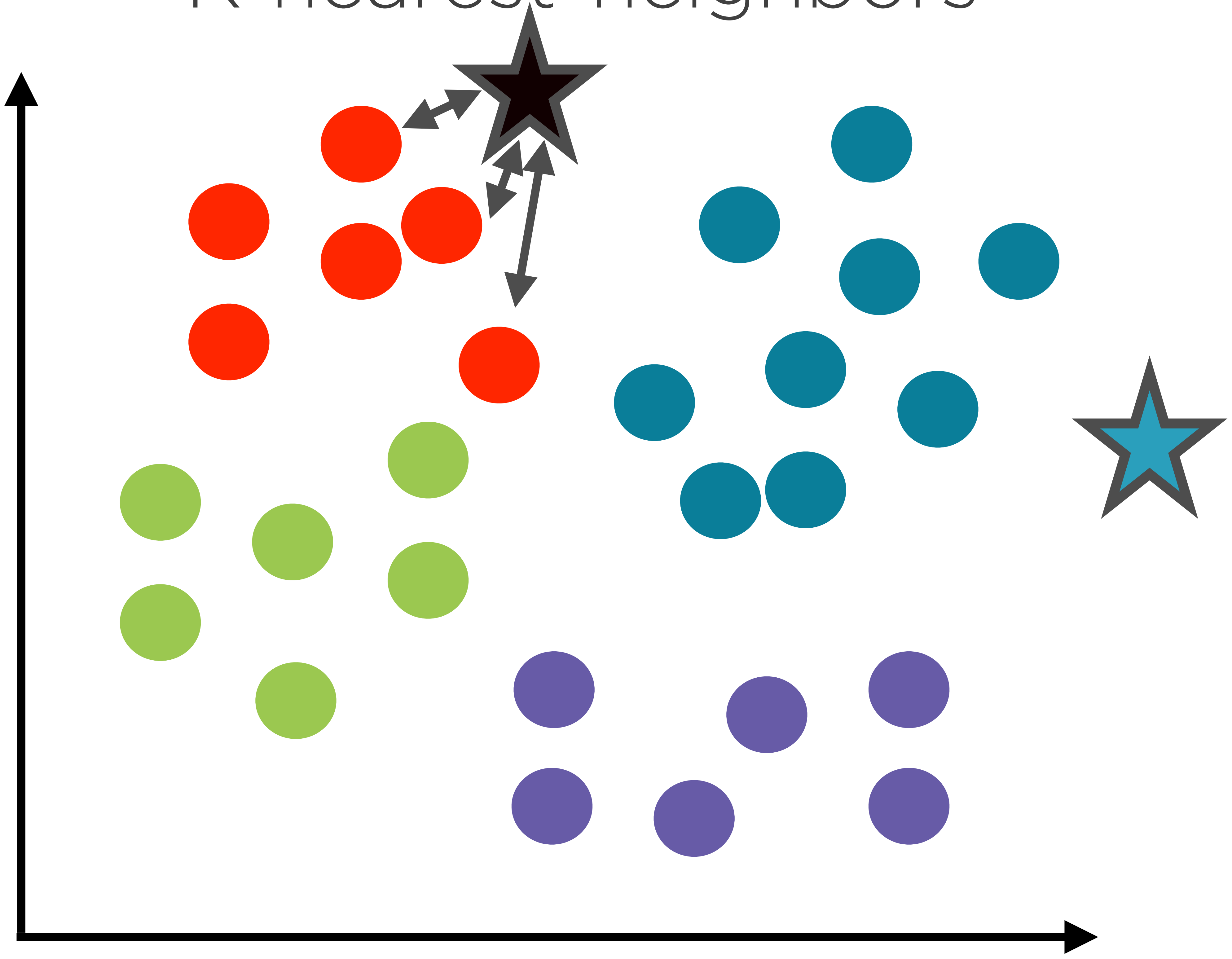


Euclidean distance, Hamming distance, Manhattan distance

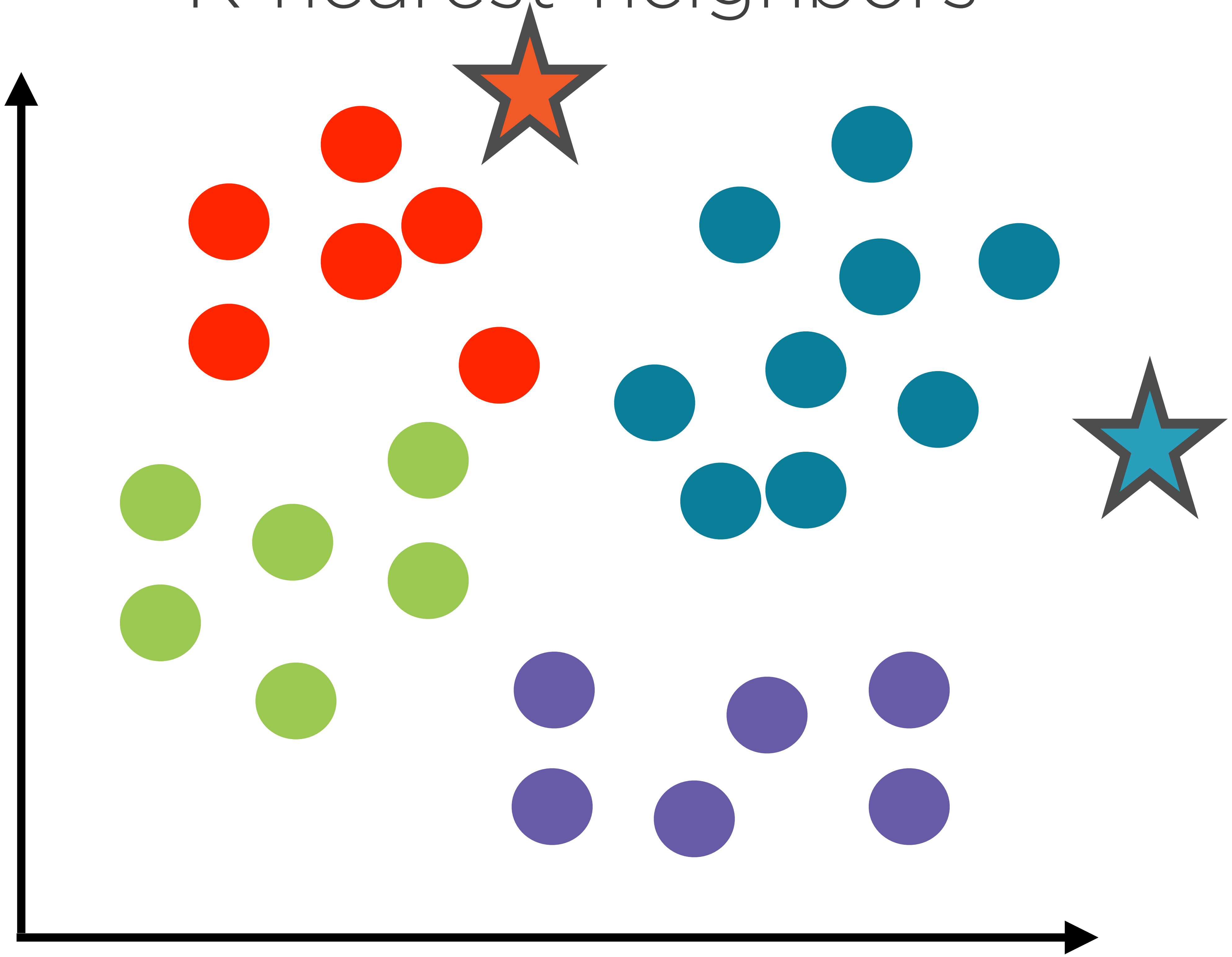
K-nearest-neighbors



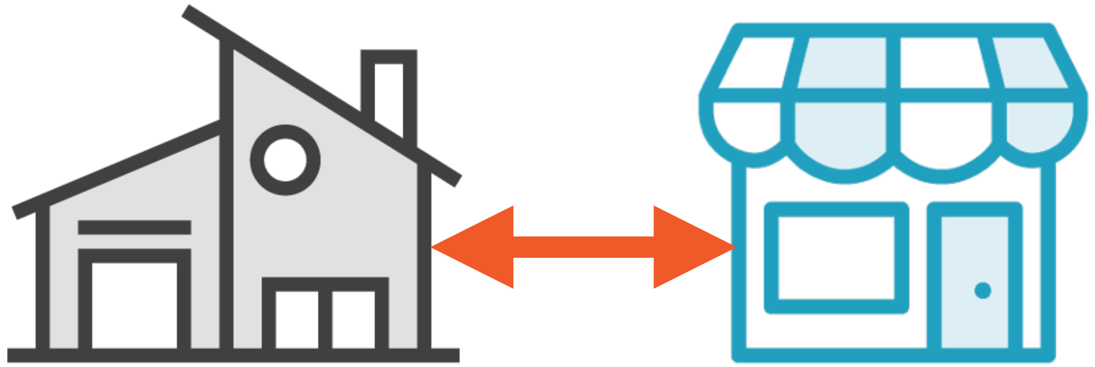
K-nearest-neighbors



K-nearest-neighbors

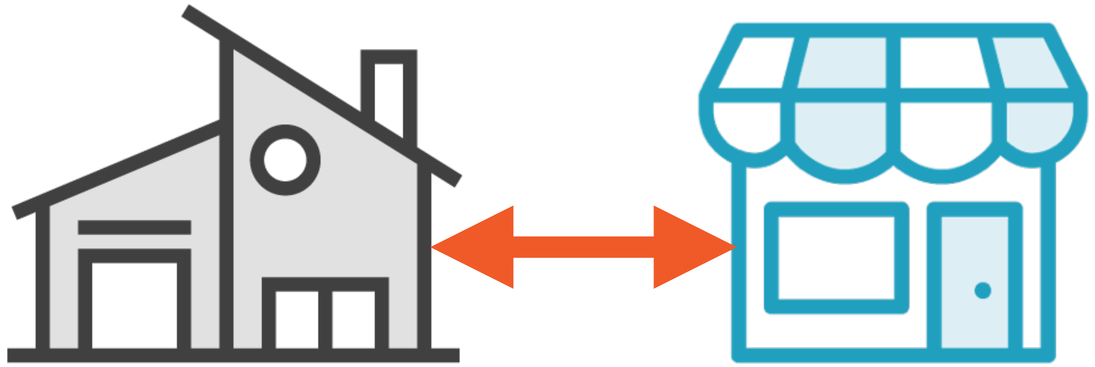


Distance Measures



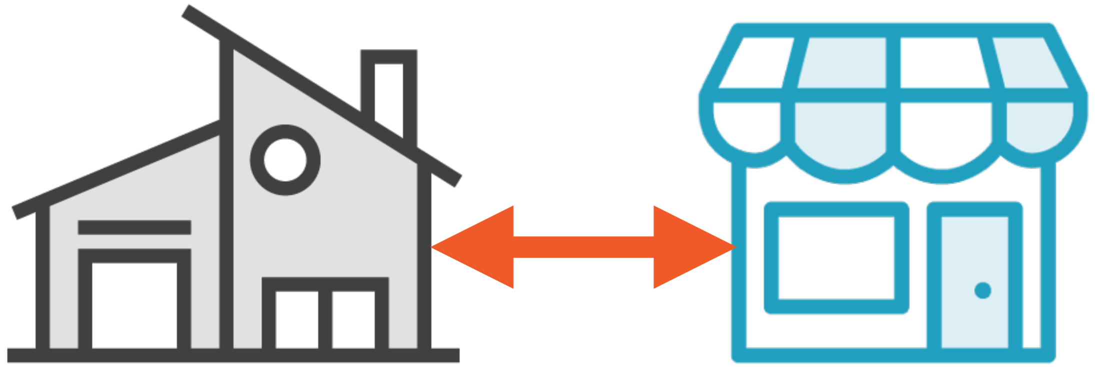
```
EuclideanDistance(x, xi) = sqrt(sum((xj - xij)^2))
```

Distance Measures



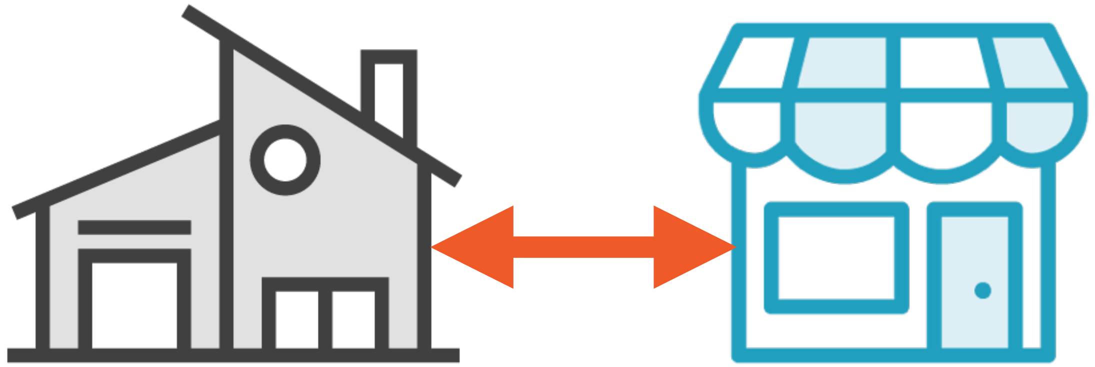
```
EuclideanDistance(x, xi) = sqrt(sum((xj - xij)^2))
```

Distance Measures



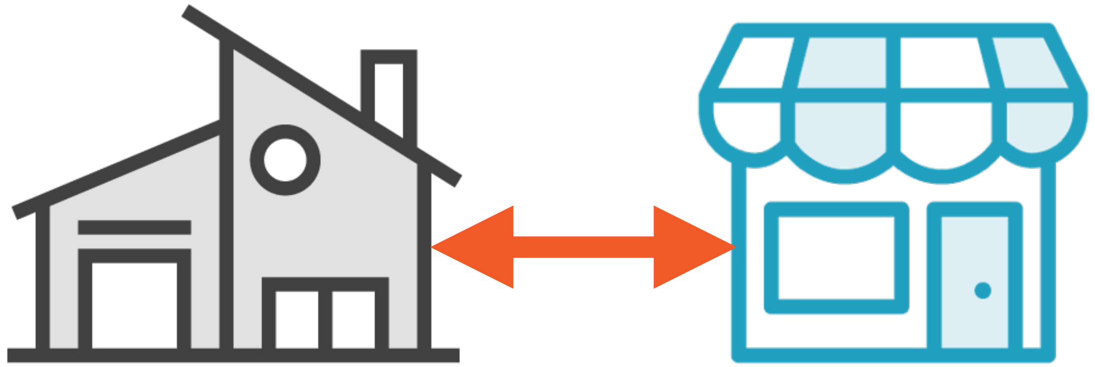
```
EuclideanDistance(x, xi) = sqrt(sum((xj – xij)^2))
```

Distance Measures



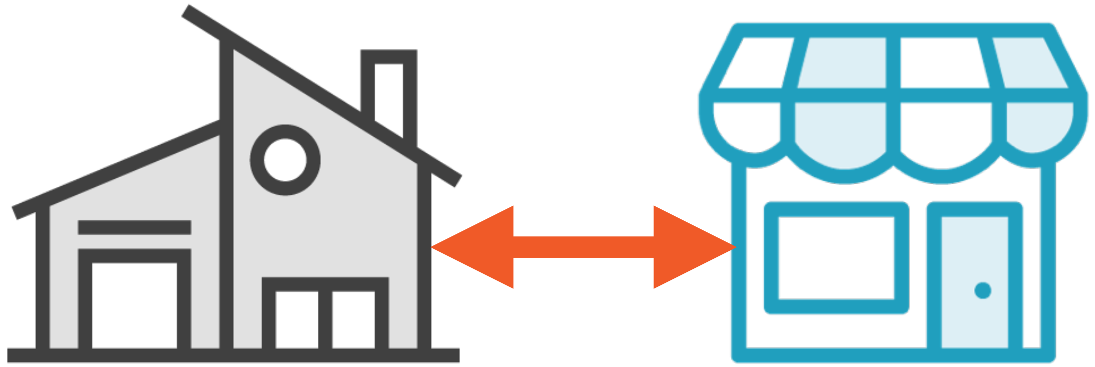
```
EuclideanDistance(x, xi) = sqrt(sum((xj - xij)^2))
```

Distance Measures



```
EuclideanDistance(x, xi) = sqrt(sum((xj – xij)^2))
```

Distance Measures



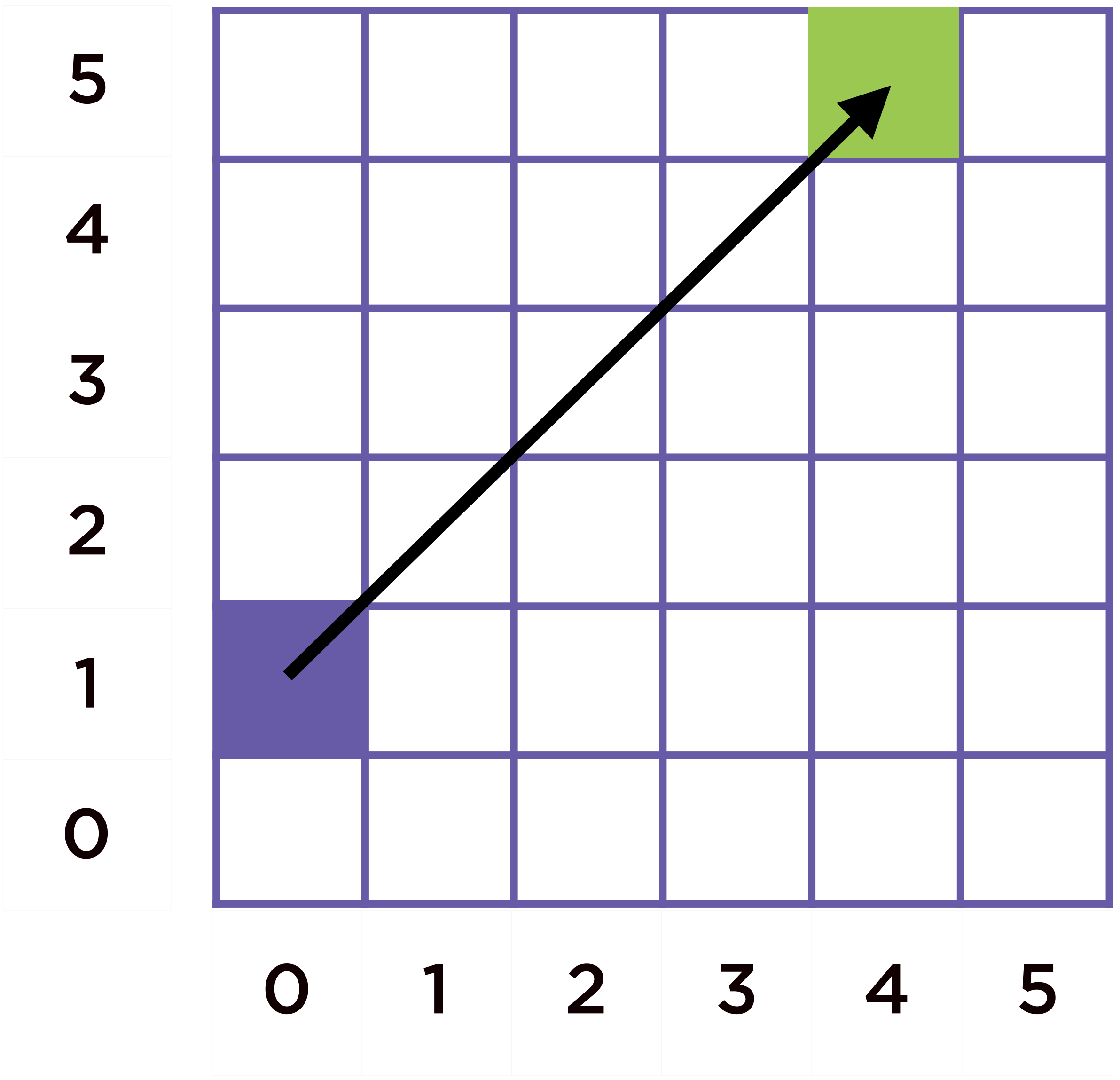
```
EuclideanDistance(x, xi) = sqrt(sum((xj – xij)^2))
```

Distance Measures

Distance Measure

Euclidean Distance

As the crow flies



L1 Distance

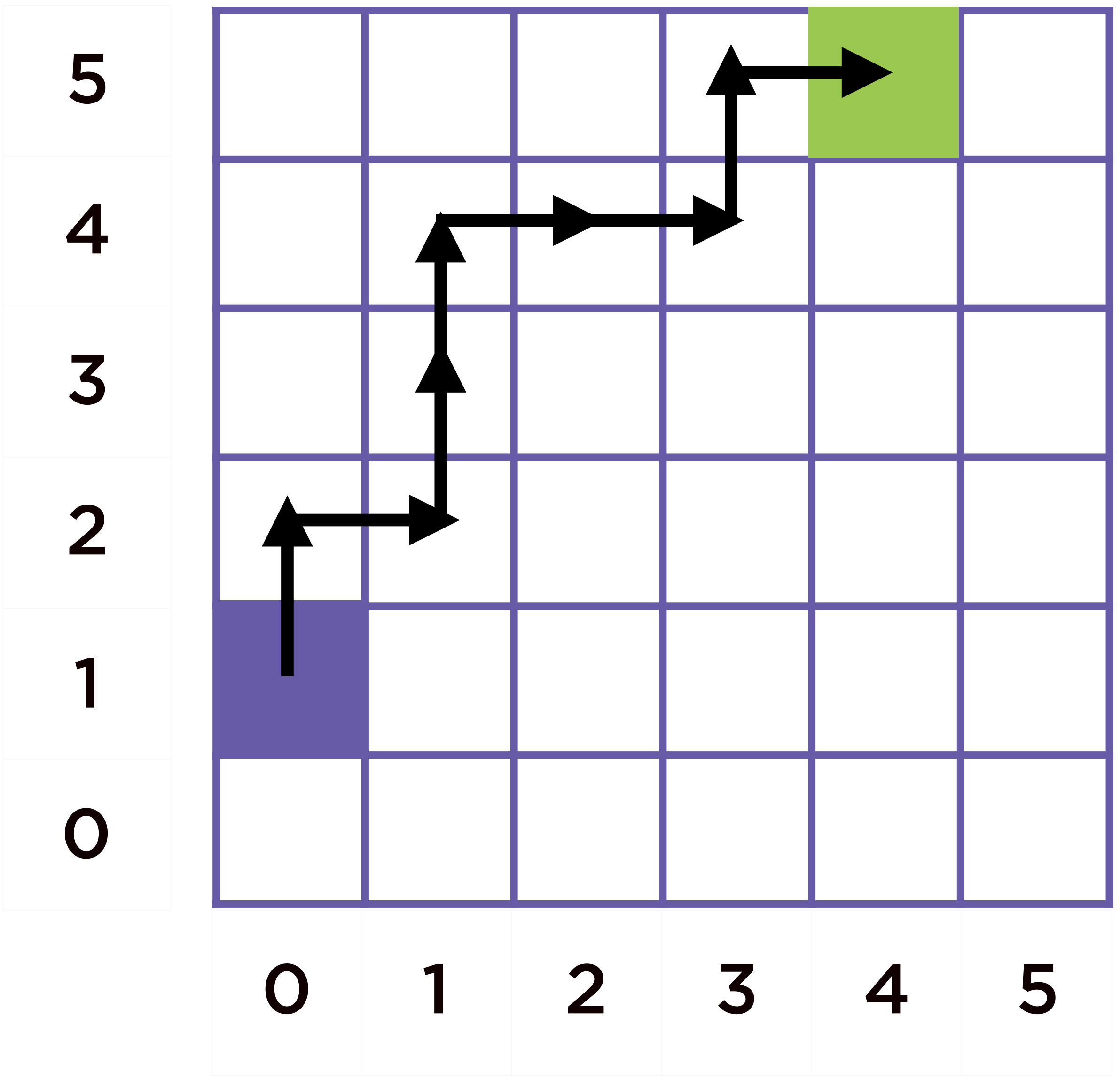
Distance Measure

L1 distance

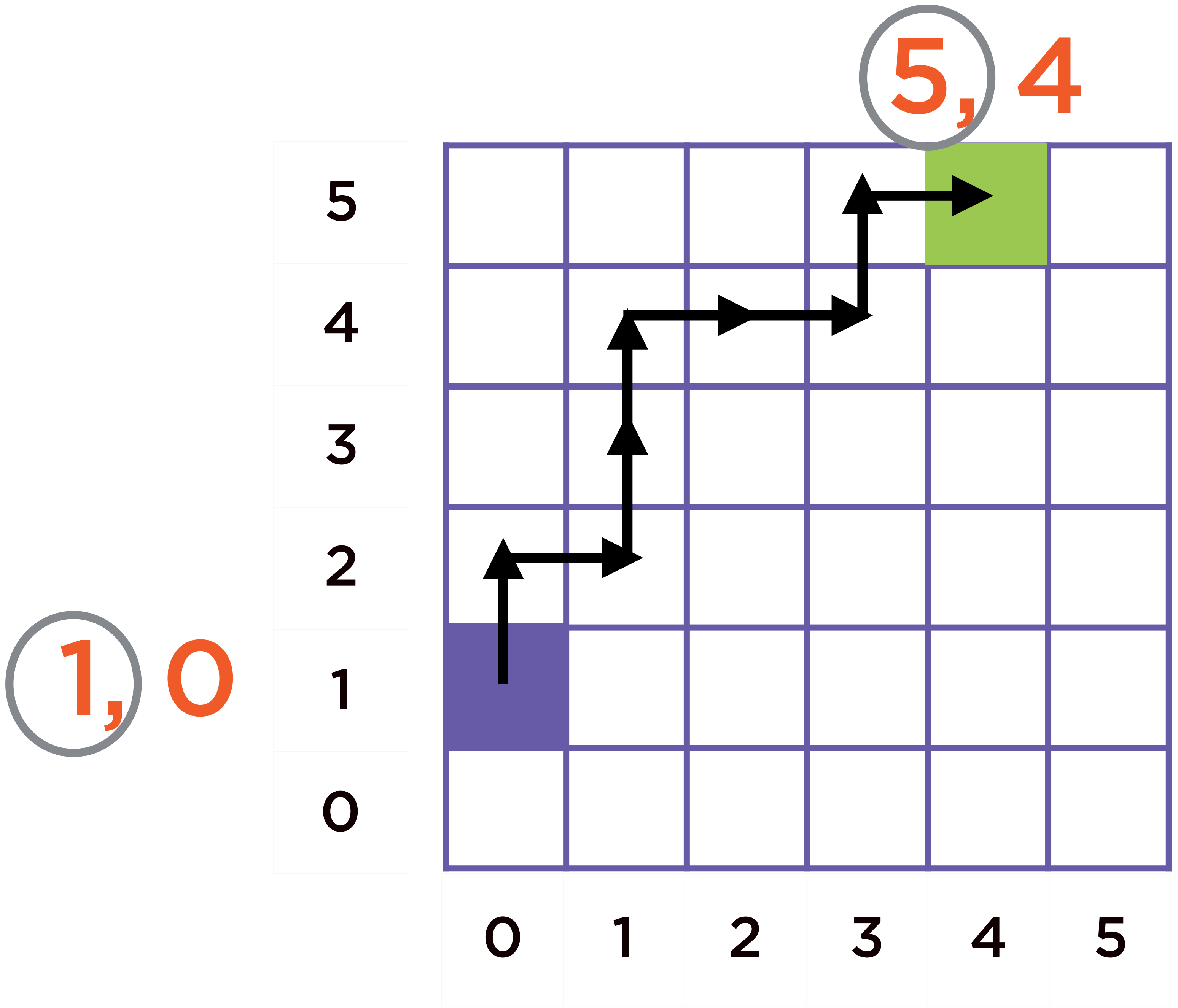
Snake distance

City block distance

Manhattan distance

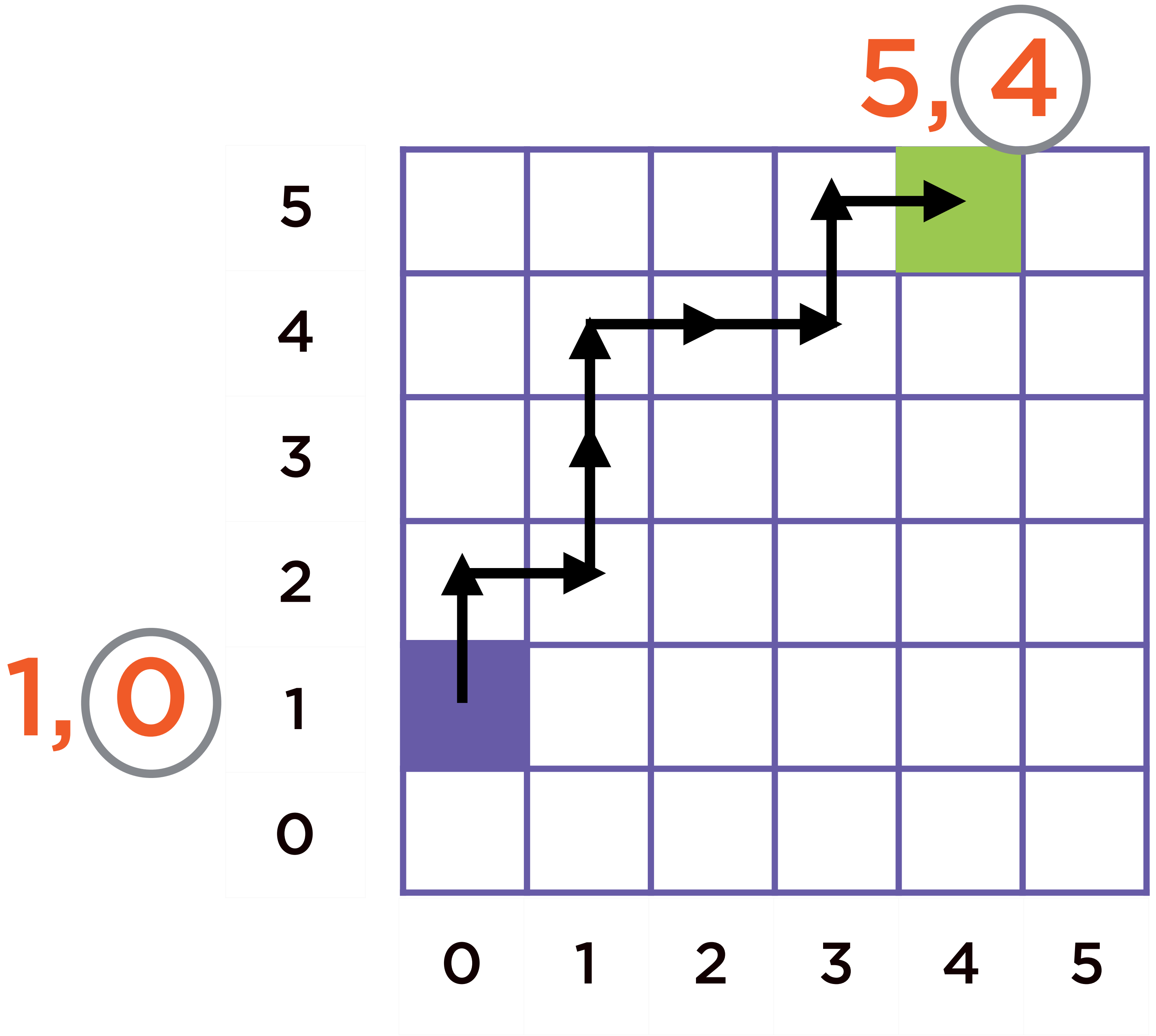


L1 Distance



5-1 = 4

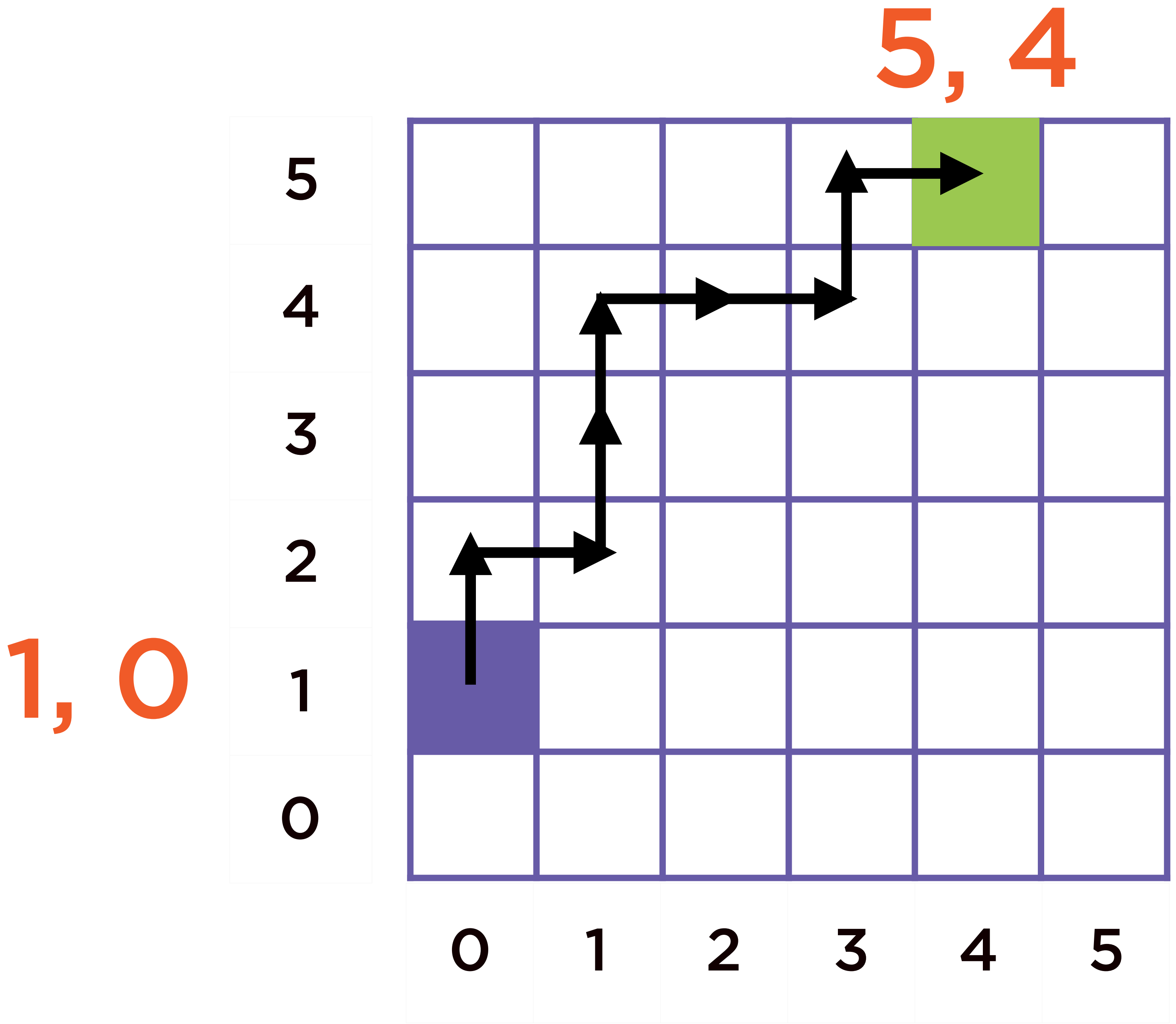
L1 Distance



5-1 = 4

4-0 = 4

L1 Distance



$$5-1 = 4$$

$$4-0 = 4$$

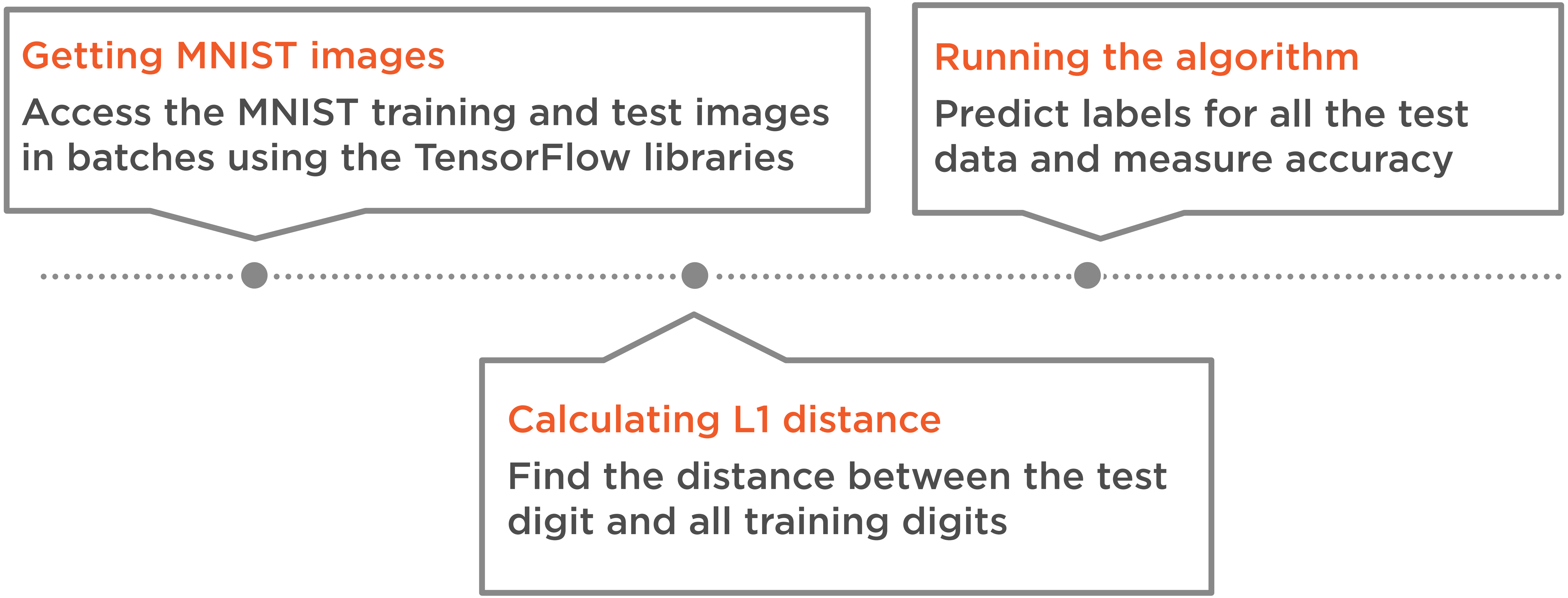
$$= 8$$

Demo

Handwritten image recognition using the k-nearest-neighbors ML algorithm

- Use the L1 distance measure to find the nearest neighbor
- Measure the accuracy of the algorithm on the test data

KNN Implemented in TensorFlow



Getting MNIST images

Access the MNIST training and test images in batches using the TensorFlow libraries

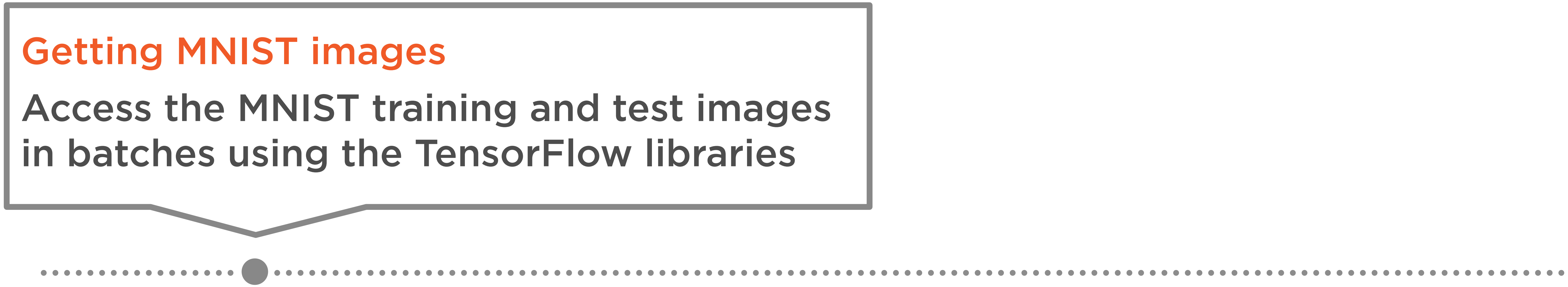
Calculating L1 distance

Find the distance between the test digit and all training digits

Running the algorithm

Predict labels for all the test data and measure accuracy

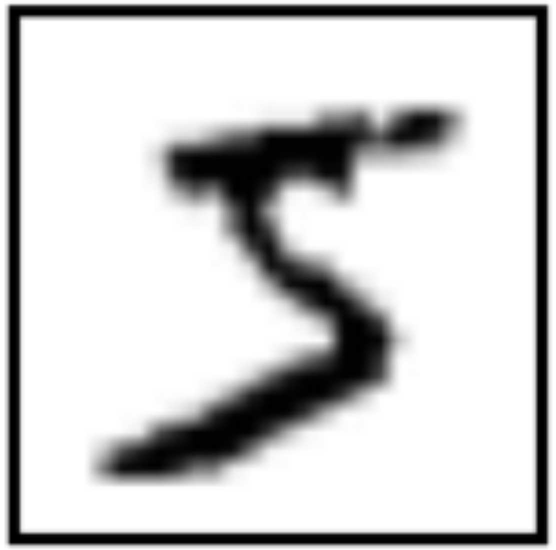
KNN Implemented in TensorFlow



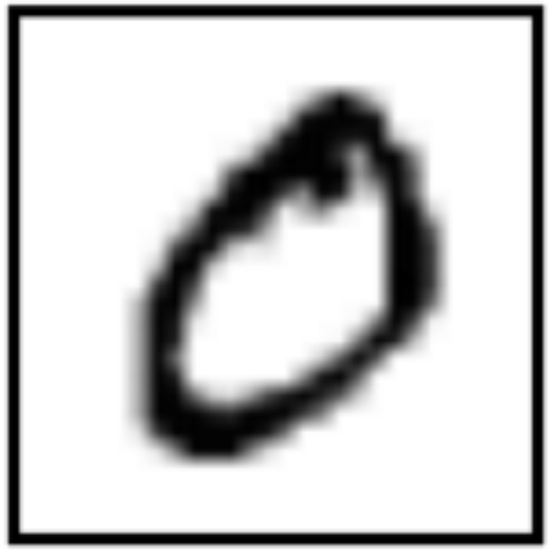
Getting MNIST images

Access the MNIST training and test images in batches using the TensorFlow libraries

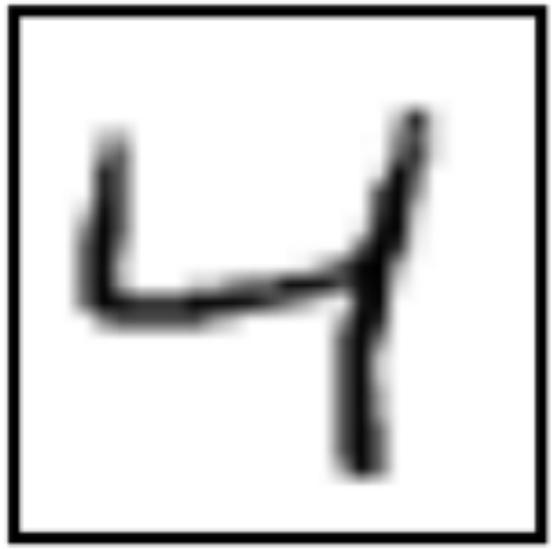
MNIST Dataset



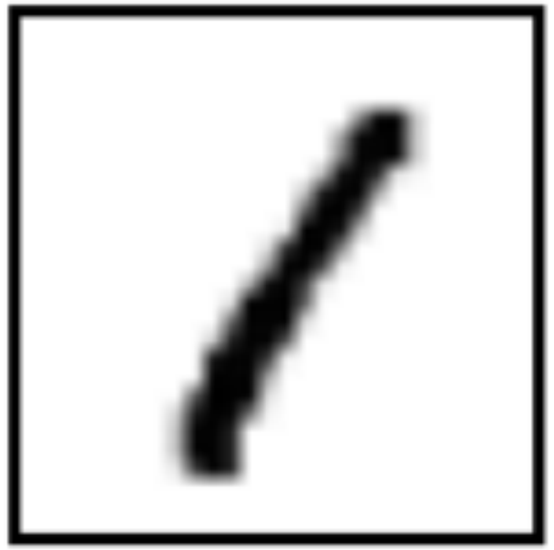
5



0

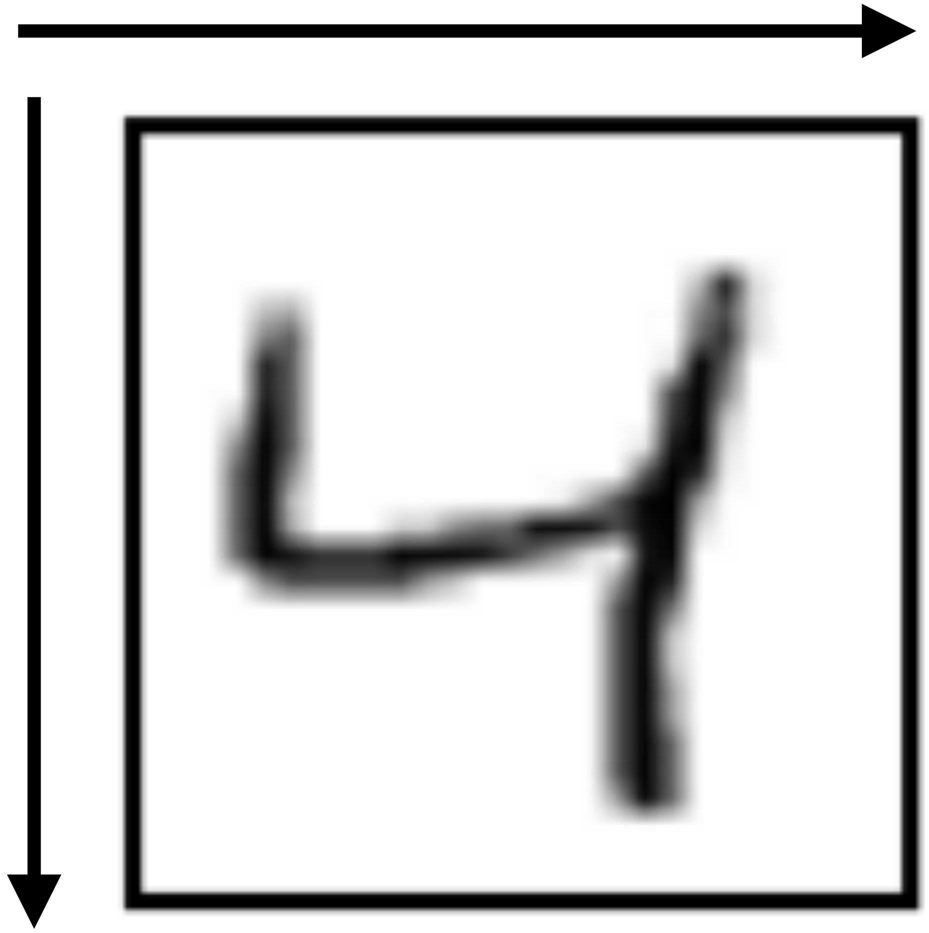


4



1

MNIST Dataset

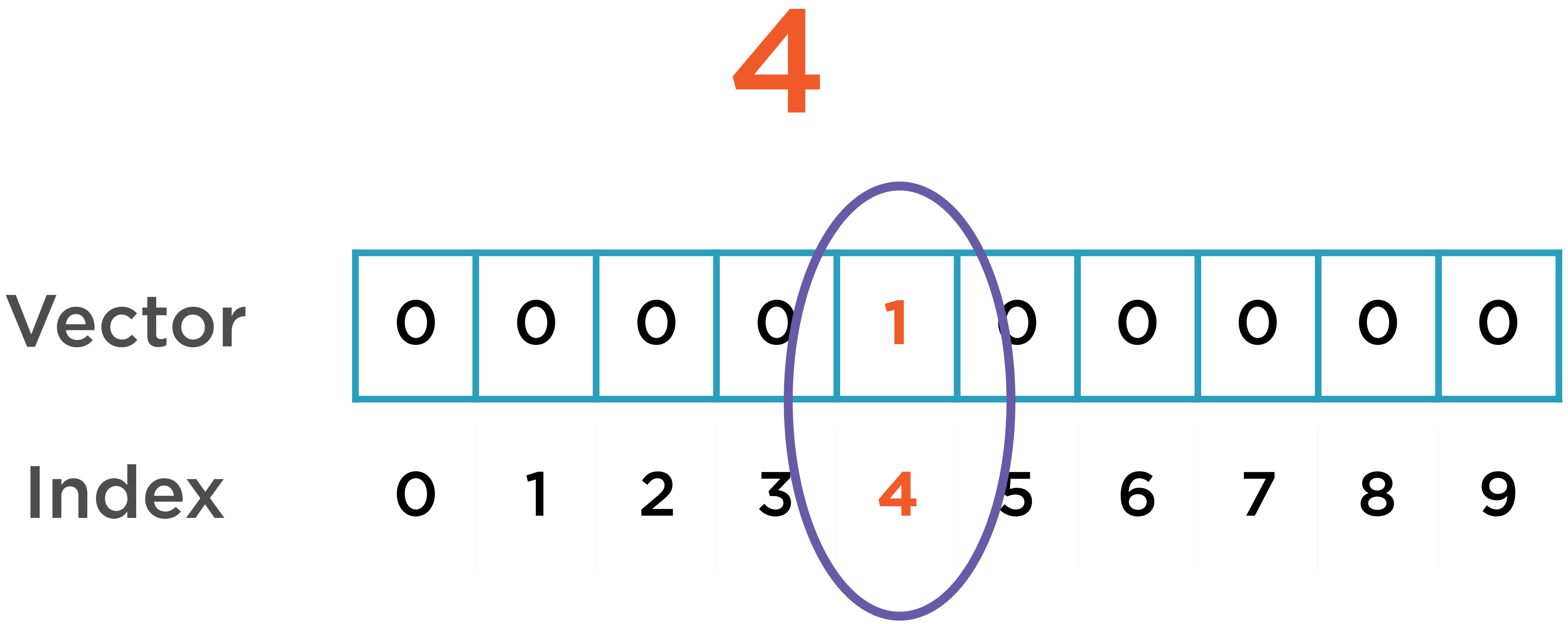


4

Every image is standardized to be of size 28x28

= 784 pixels

Representing Labels



4

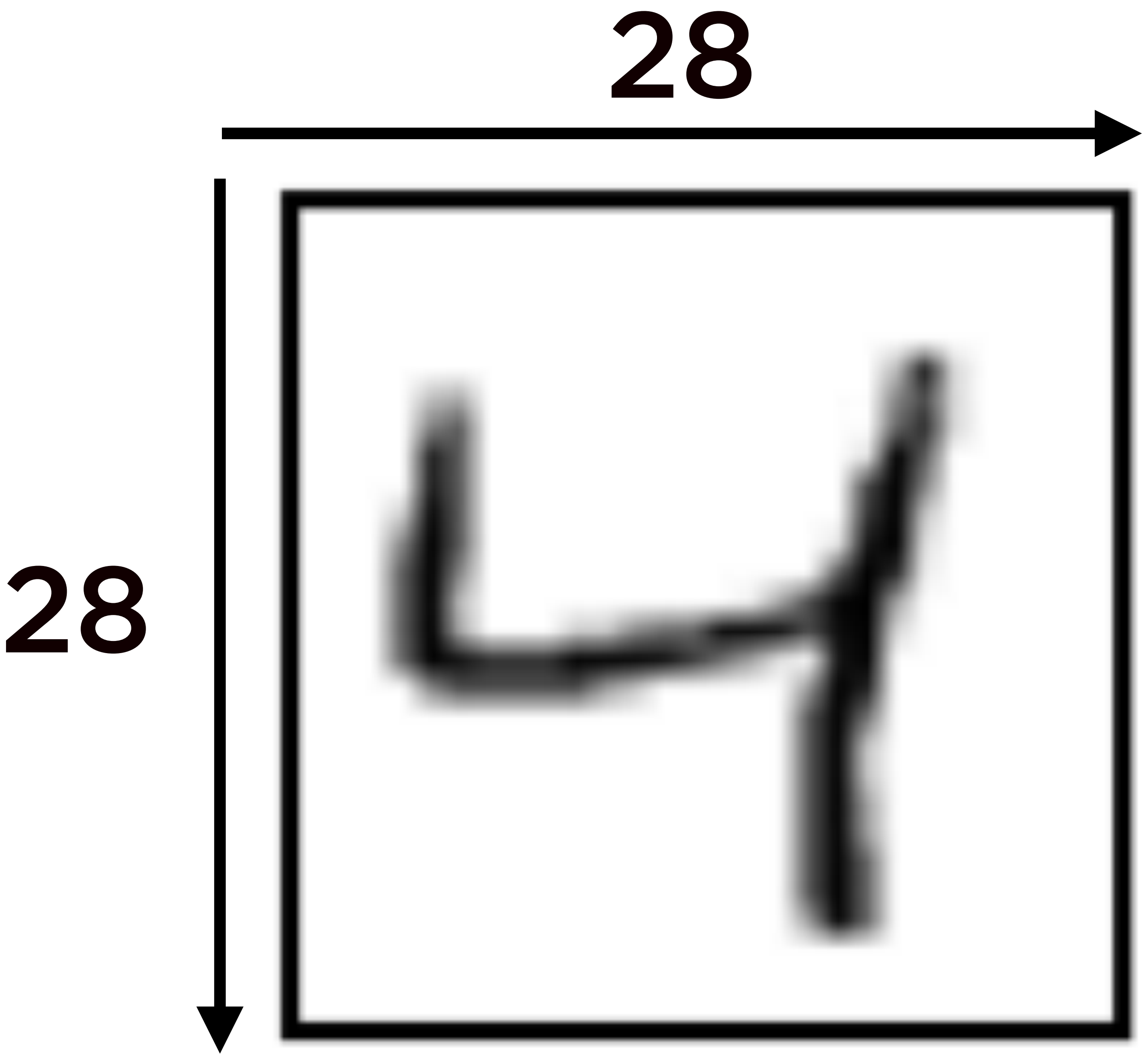
Vector	0	0	0	0	1	0	0	0	0	0
Index	0	1	2	3	4	5	6	7	8	9

Representing Labels

5

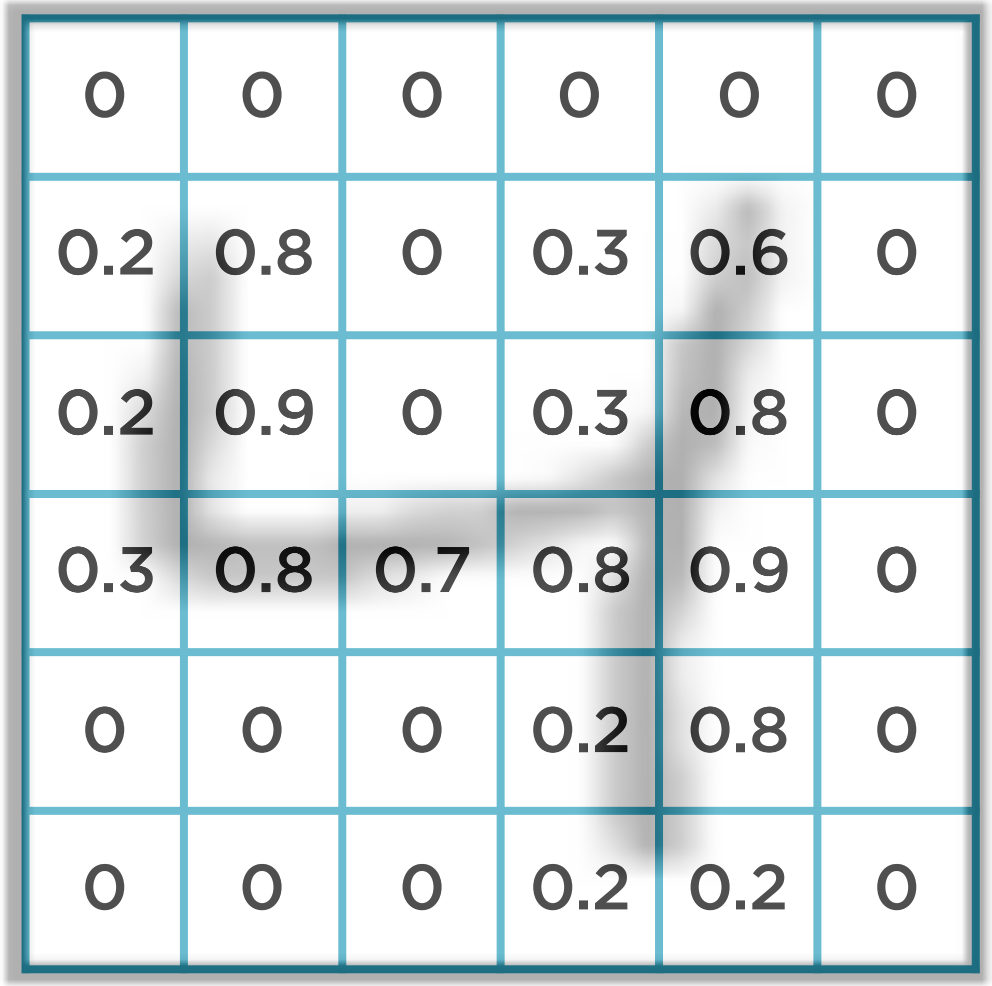
Vector	0	0	0	0	0	1	0	0	0	0
Index	0	1	2	3	4	5	6	7	8	9

Representing Images



= 784 pixels

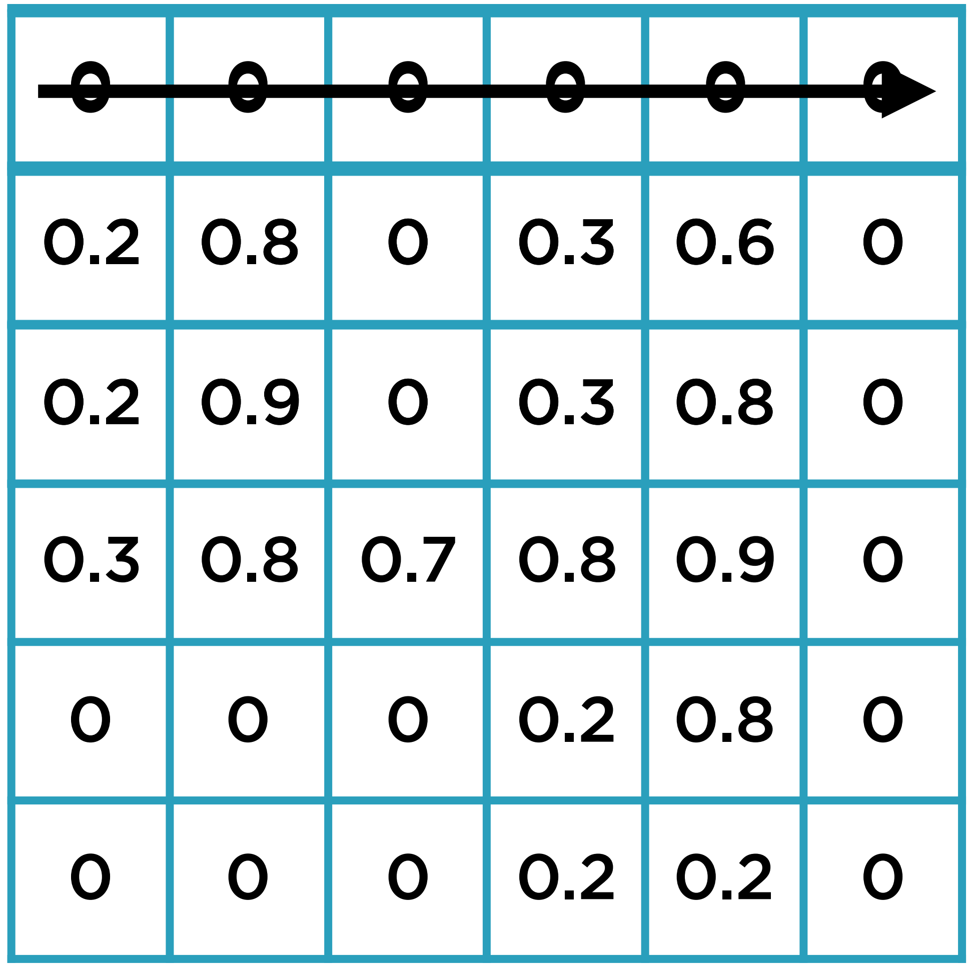
Representing Images



0	0	0	0	0	0
0.2	0.8	0	0.3	0.6	0
0.2	0.9	0	0.3	0.8	0
0.3	0.8	0.7	0.8	0.9	0
0	0	0	0.2	0.8	0
0	0	0	0.2	0.2	0

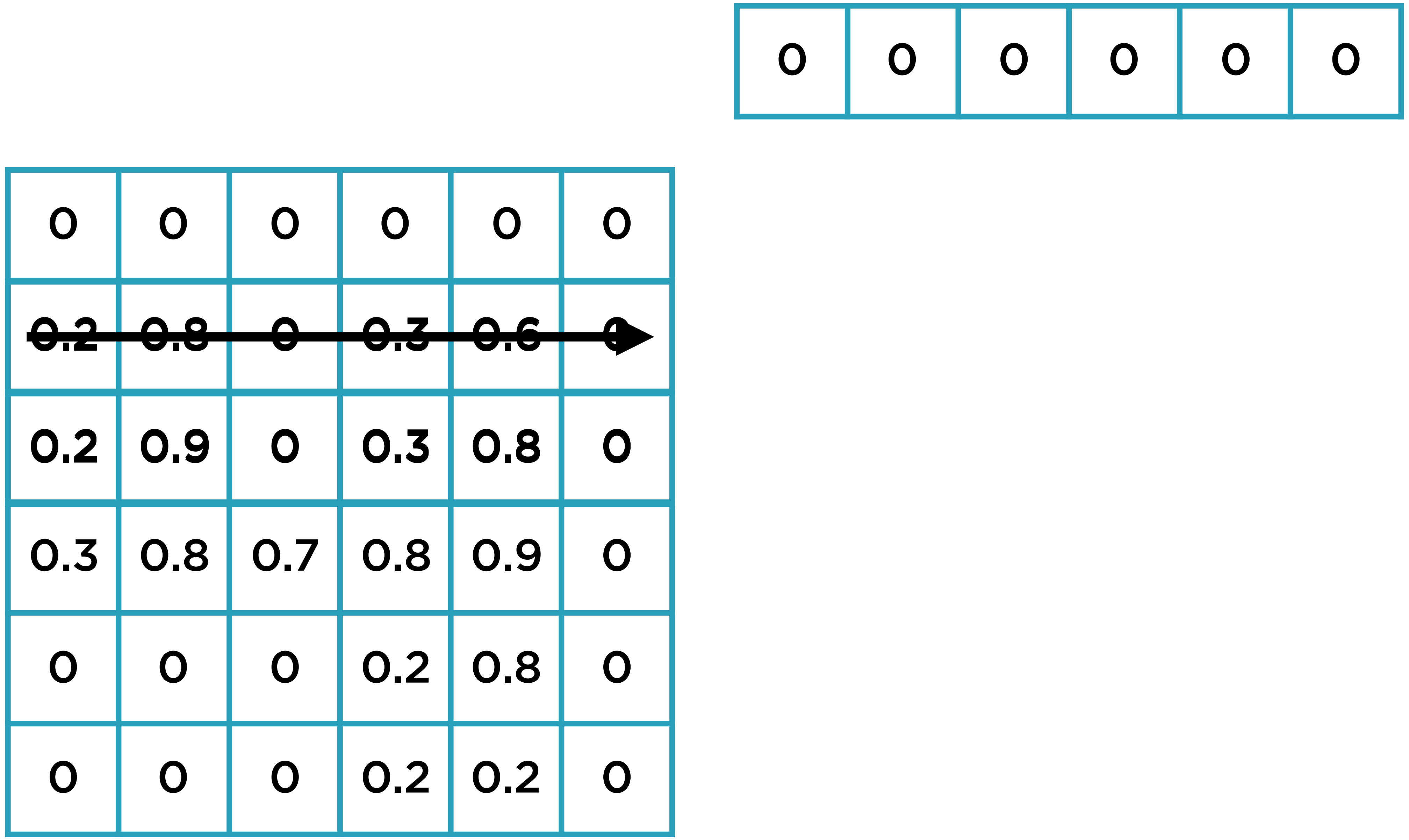
= 784 pixels

Representing Images



0	0	0	0	0	0
0.2	0.8	0	0.3	0.6	0
0.2	0.9	0	0.3	0.8	0
0.3	0.8	0.7	0.8	0.9	0
0	0	0	0.2	0.8	0
0	0	0	0.2	0.2	0

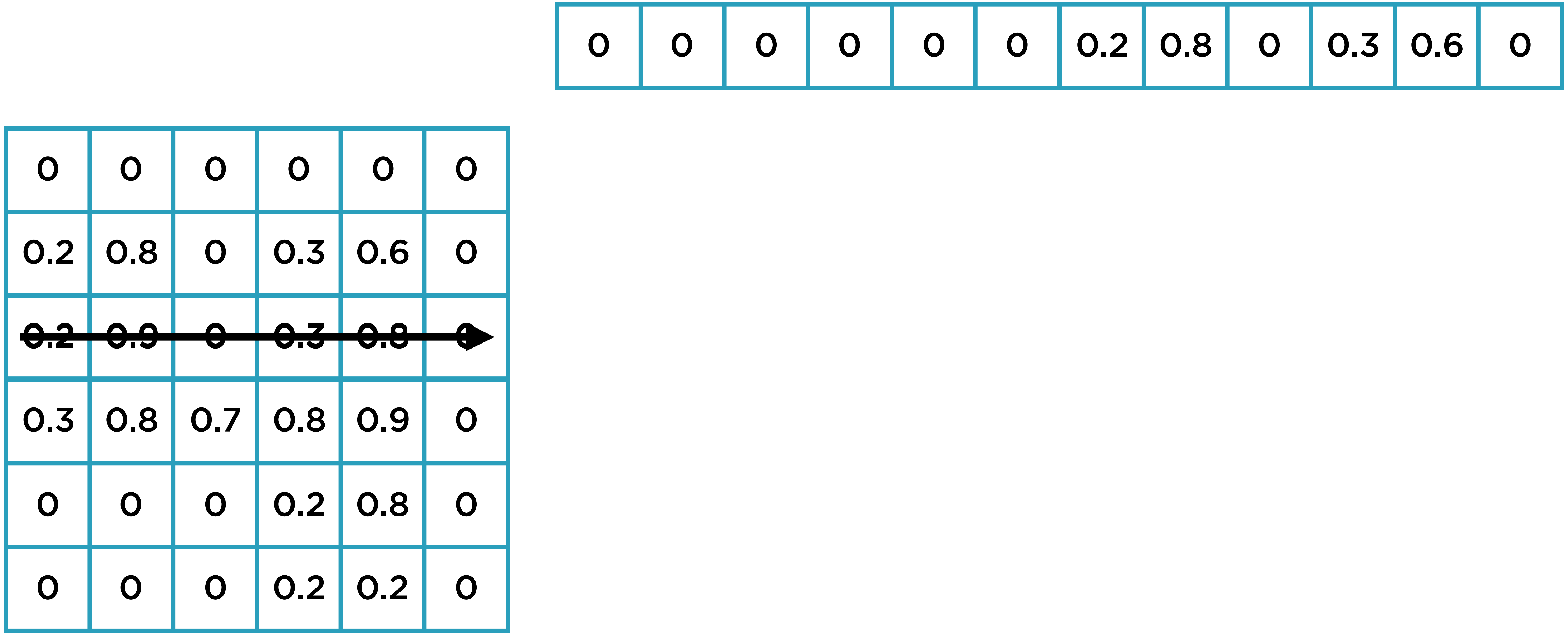
Representing Images



0	0	0	0	0	0

0	0	0	0	0	0
0.2	0.8	0	0.3	0.6	0
0.2	0.9	0	0.3	0.8	0
0.3	0.8	0.7	0.8	0.9	0
0	0	0	0.2	0.8	0
0	0	0	0.2	0.2	0

Representing Images



0	0	0	0	0	0	0.2	0.8	0	0.3	0.6	0

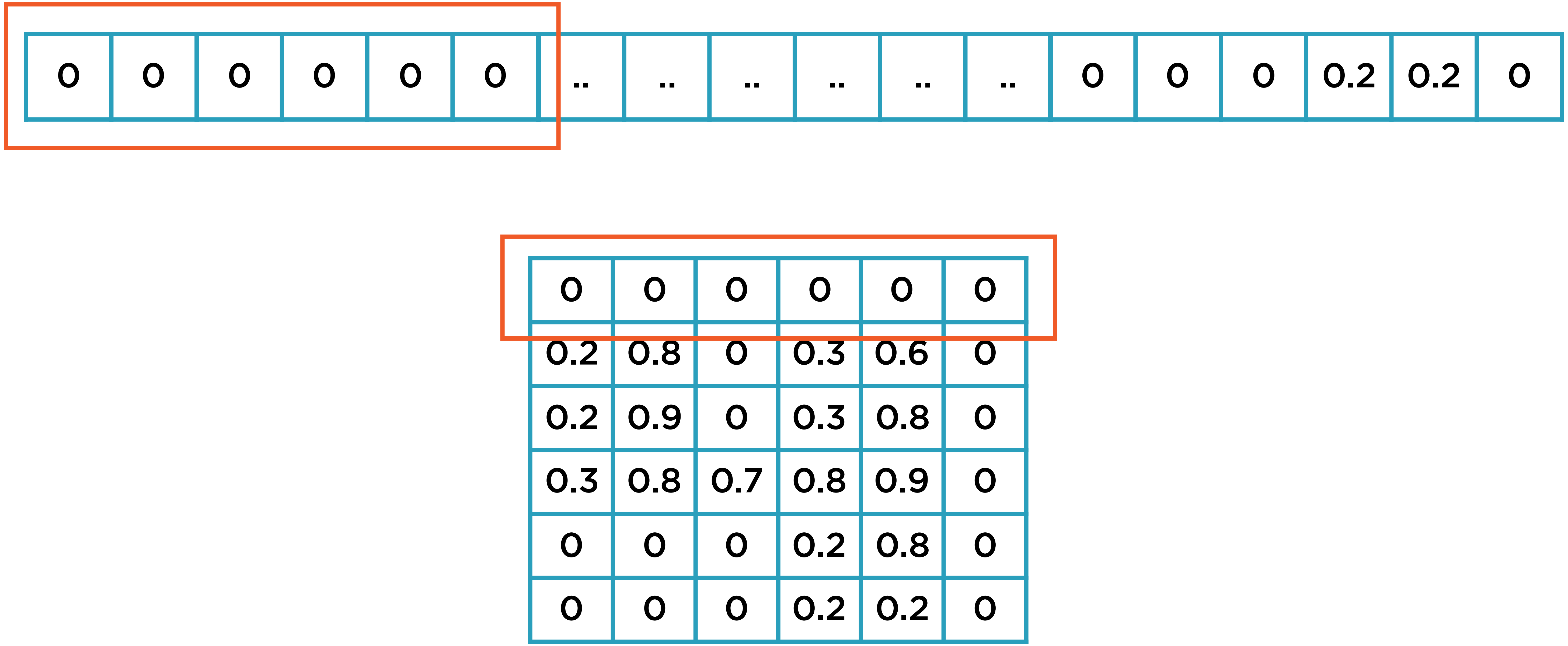
0	0	0	0	0	0
0.2	0.8	0	0.3	0.6	0
0.2	0.9	0	0.3	0.8	0
0.3	0.8	0.7	0.8	0.9	0
0	0	0	0.2	0.8	0
0	0	0	0.2	0.2	0

Representing Images

0	0	0	0	0	0	0.2	0.8	0	0.3	0.6	0	0.2	0.9	0	0.3	0.8	0

0	0	0	0	0	0
0.2	0.8	0	0.3	0.6	0
0.2	0.9	0	0.3	0.8	0
0.3	0.8	0.7	0.8	0.9	0
0	0	0	0.2	0.8	0
0	0	0	0.2	0.2	0

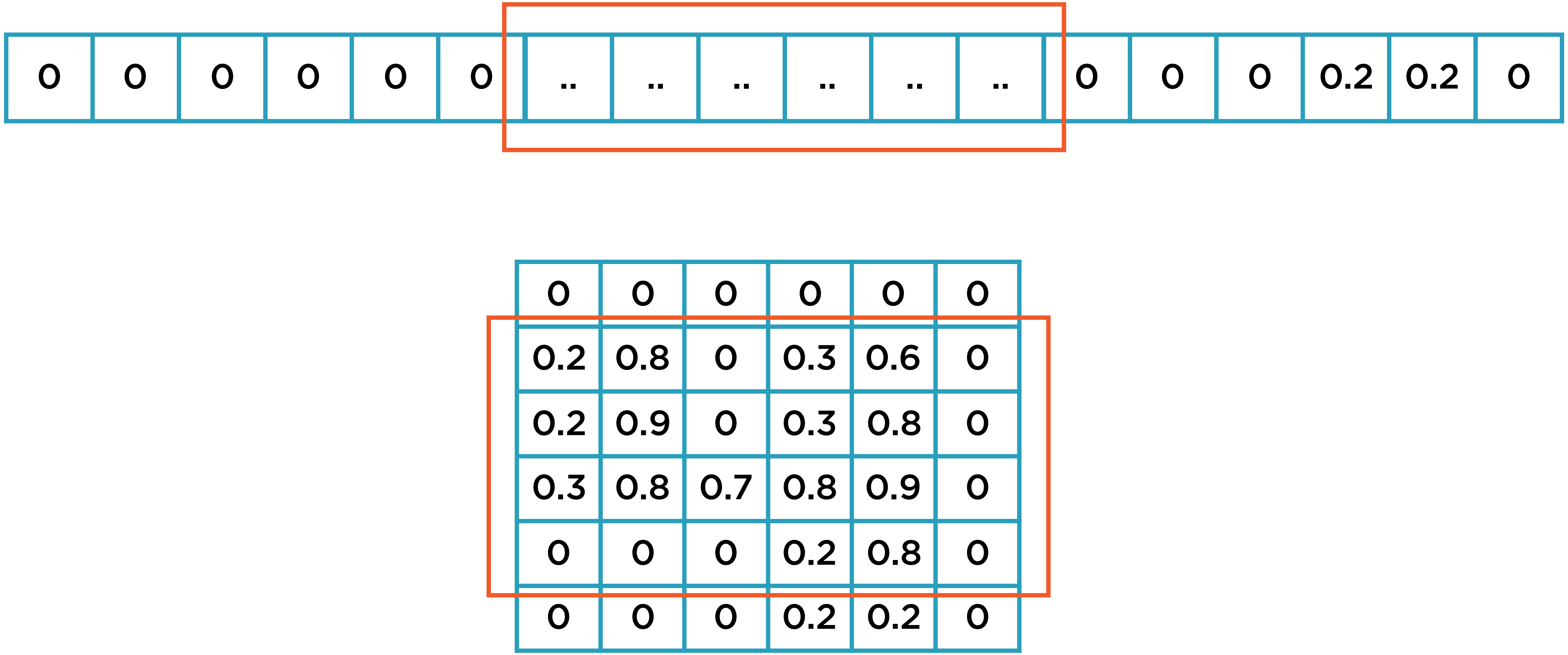
Representing Images



0	0	0	0	0	0	..	..	..	..	..	..	0	0	0	0.2	0.2	0

0	0	0	0	0	0
0.2	0.8	0	0.3	0.6	0
0.2	0.9	0	0.3	0.8	0
0.3	0.8	0.7	0.8	0.9	0
0	0	0	0.2	0.8	0
0	0	0	0.2	0.2	0

Representing Images



0	0	0	0	0	0	..	..	..	..	..	..	0	0	0	0.2	0.2	0

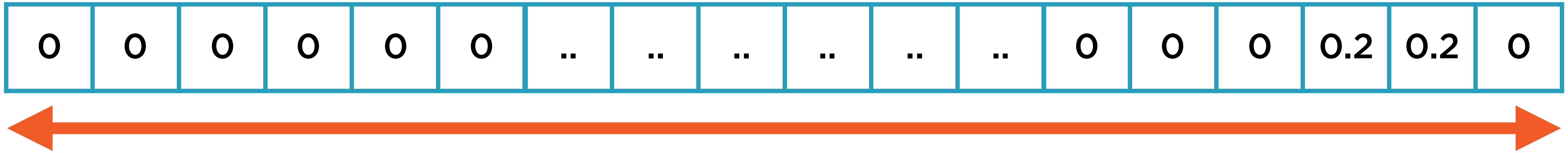
0	0	0	0	0	0
0.2	0.8	0	0.3	0.6	0
0.2	0.9	0	0.3	0.8	0
0.3	0.8	0.7	0.8	0.9	0
0	0	0	0.2	0.8	0
0	0	0	0.2	0.2	0

Representing Images

0	0	0	0	0	0	..	..	..	..	..	..	0	0	0	0.2	0.2	0

0	0	0	0	0	0
0.2	0.8	0	0.3	0.6	0
0.2	0.9	0	0.3	0.8	0
0.3	0.8	0.7	0.8	0.9	0
0	0	0	0.2	0.8	0
0	0	0	0.2	0.2	0

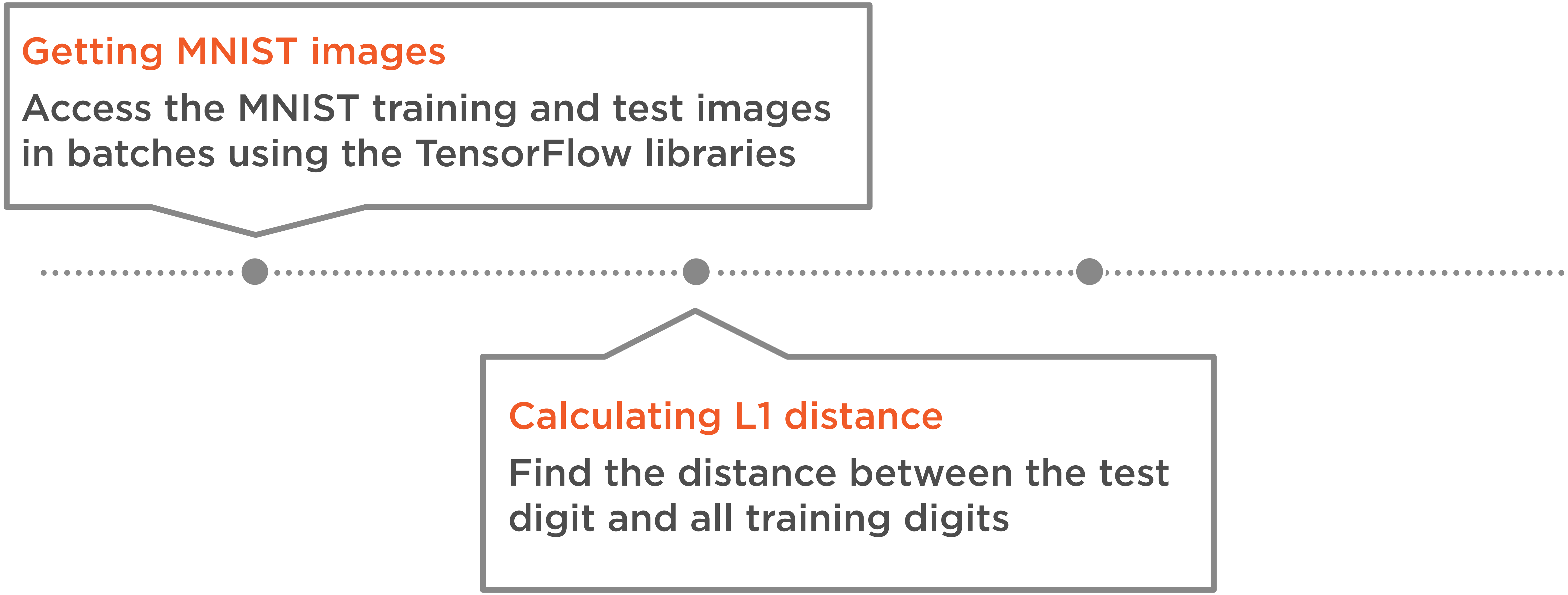
Representing Images



0	0	0	0	0	0	..	..	..	..	..	..	0	0	0	0.2	0.2	0

= 784 pixels

KNN Implemented in TensorFlow



Getting MNIST images

Access the MNIST training and test images in batches using the TensorFlow libraries

Calculating L1 distance

Find the distance between the test digit and all training digits

L1 Distance

Training

0.2	0.8	0	0.3	0.6	0

0.2	0.8	0	0.3	0.6	0

Test

0	0.6	0.2	0.3	0.3	0.1

tf.negative()

0	-0.6	-0.2	-0.3	-0.3	-0.1

tf.add()

0.2	0.2	-0.2	0	0.3	-0.1

L1 Distance

tf.add()

0.2	0.2	-0.2	0	0.3	-0.1

tf.abs()

0.2	0.2	0.2	0	0.3	0.1

tf.reduce_sum()

1.0

tf.add()

0	0.2	0.3	0.6	-0.3	-0.1

tf.abs()

0	0.2	0.3	0.6	0.3	0.1

tf.reduce_sum()

1.5

tf.add()

-0.2	0.4	-0.2	0	-0.3	-0.1

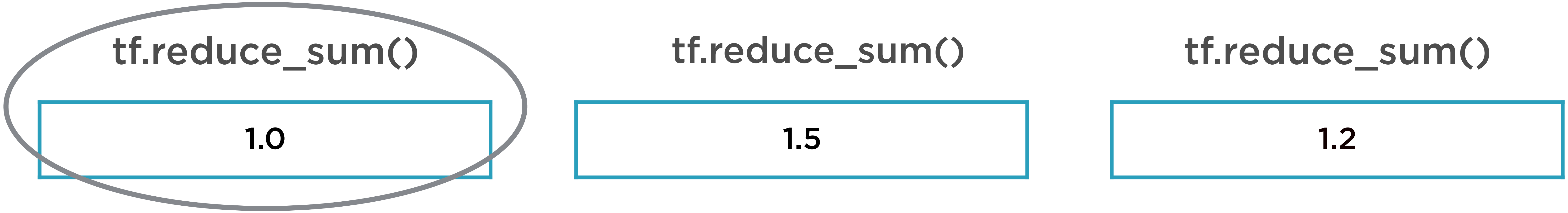
tf.abs()

0.2	0.4	0.2	0	0.3	0.1

tf.reduce_sum()

1.2

L1 Distance



tf.reduce_sum()

1.0

tf.reduce_sum()

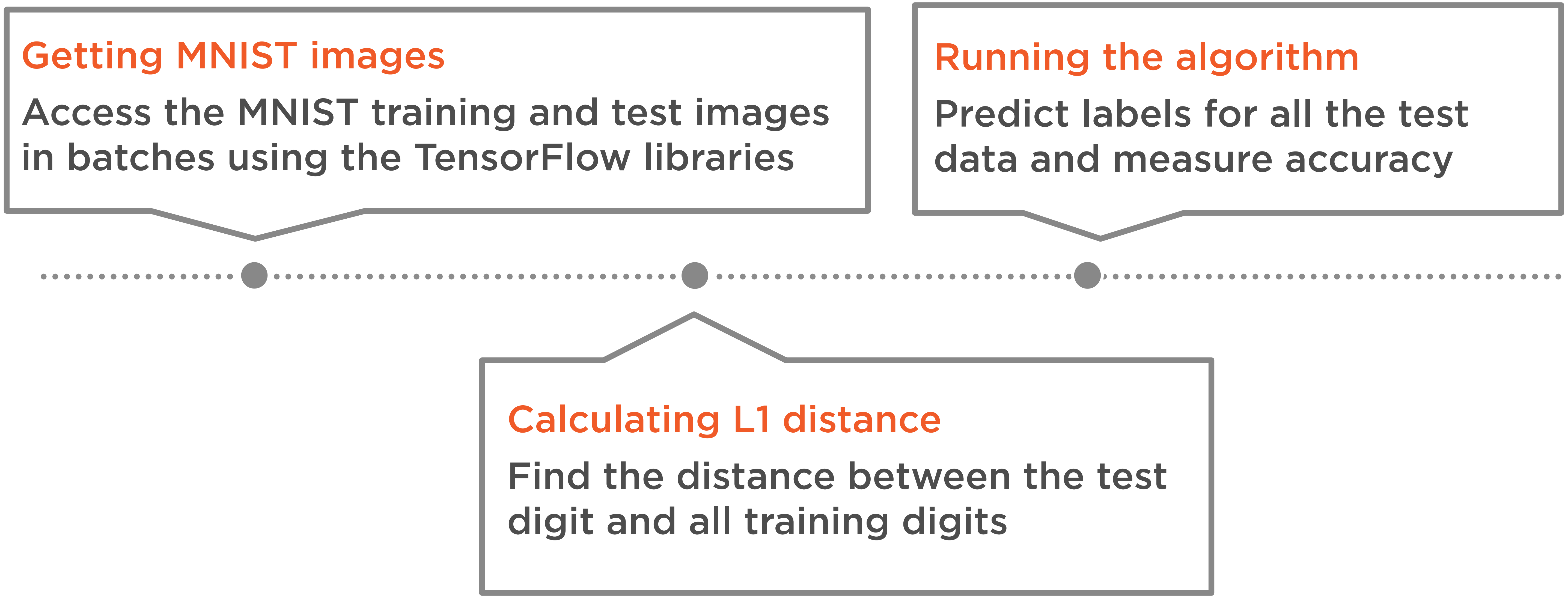
1.5

tf.reduce_sum()

1.2

index = 0

KNN Implemented in TensorFlow



Getting MNIST images

Access the MNIST training and test images in batches using the TensorFlow libraries

Calculating L1 distance

Find the distance between the test digit and all training digits

Running the algorithm

Predict labels for all the test data and measure accuracy

Representing Labels

4

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

np.argmax()

Summary

Familiar with the MNIST handwritten digit dataset

Understood the logic behind the K-nearest-neighbors algorithm

Implemented K-nearest-neighbors using L1 distance to identify handwritten digits from 0 to 9