TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC MÁY TÍNH

CS231. Nhập môn Thị giác máy tính

Machine Learning Image Classification

Mai Tiến Dũng

Content

1. Introduction to image classification
2. Image classification with KNN
3. Image feature

1. Introduction to Image Classification

- What is image classification
- Challenges of image classification

What is image classification

- Process of taking an image or picture

Cat

98%

Use of image classification

Organize Photo albums on smart devices

Augment Medical professionals

Identify images around Self-driving cars

Targets:

$y = 0$

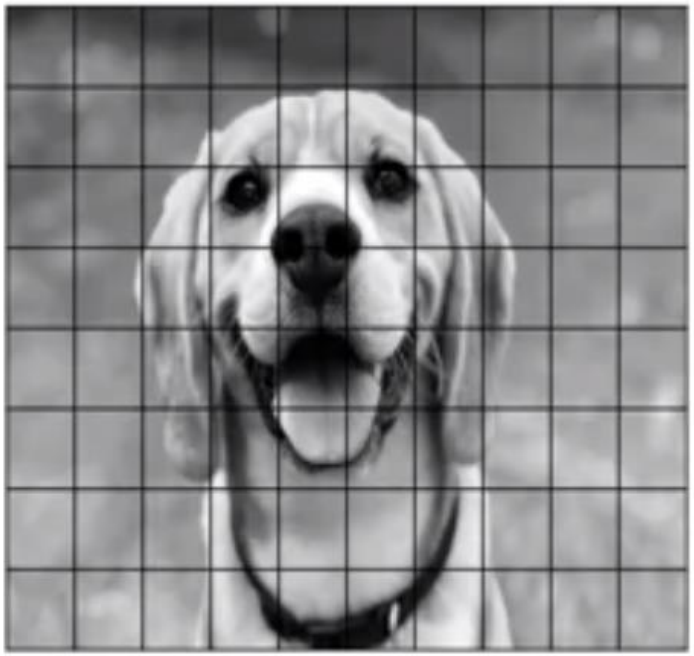
$y = \text{“}cat\text{”}$

$y = 1$

$y = \text{“}dog\text{”}$

$$X = \begin{bmatrix} 2 & 3 & .. & 40 \\ 3 & 10 & .. & 50 \\ 2 & 3 & .. & 40 \\ 5 & 34 & .. & 24 \\ 3 & 10 & .. & 50 \\ 2 & 3 & .. & 40 \\ 5 & 34 & .. & 24 \\ 3 & 10 & .. & 50 \\ 5 & 34 & .. & 24 \end{bmatrix}$$

SKILLS NETWORK



$$X = \begin{bmatrix} 2 & 3 & .. & 40 \\ 3 & 10 & .. & 50 \\ 5 & 34 & .. & 24 \end{bmatrix}$$

$\mathbf{x}_1, y_1 = 1$

$\mathbf{x}_3, y_3 = 0$

$\mathbf{x}_5, y_5 = 0$

$\mathbf{x}_2, y_2 = 1$

$\mathbf{x}_4, y_4 = 1$

$\mathbf{x}_6, y_6 = 0$

$\mathbf{x}_7, y_7 = 0$

$y = 0$ 

$y = 1$ 

$y = 2$ 

$y = 3$ 

$y = 4$ 

$y = 5$ 

$y = 6$ 

$y = 7$ 

$y = 8$ 

$y = 9$ 


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

def classify(X):

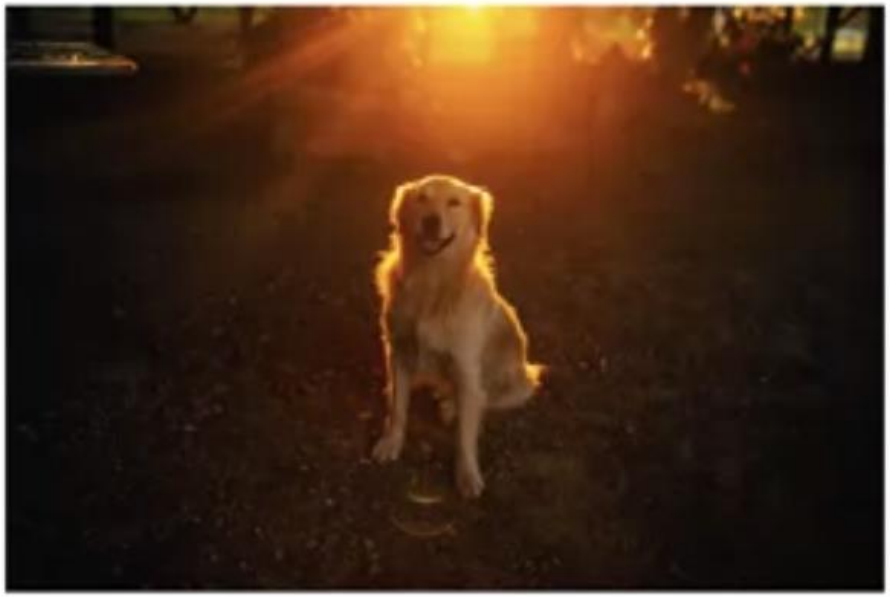
Classifier

Cat

Challenges of image classification

Challenges: change in viewpoint

Challenges: Illumination



Challenges: deformation

Challenges: occlusion

Challenges: background clutter

Image Classification with KNN

Image Classification with KNN

What is k-NN?

- k-NN means k-Nearest Neighbor
- Simplest classification algorithm
- Uses the most common and nearest classes to find closes match

Image Classification with KNN

What is k-NN?

- k-NN means k-Nearest Neighbor
- Simplest classification algorithm
- Uses the most common and nearest classes to find closes match

Image Classification with KNN

Is it a Cat or a Dog?

$y = 0$

$y = 1$

Image Classification with KNN

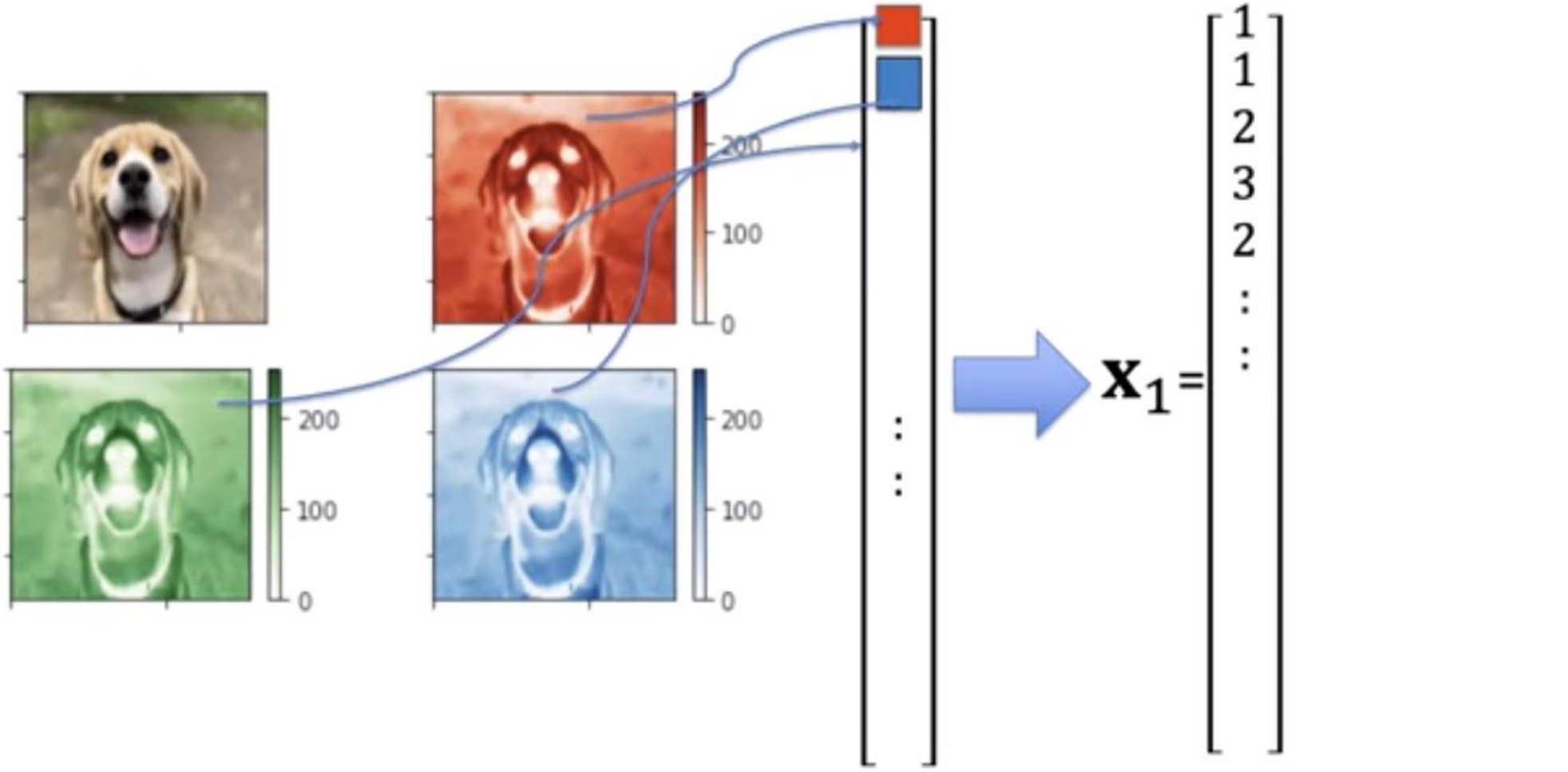
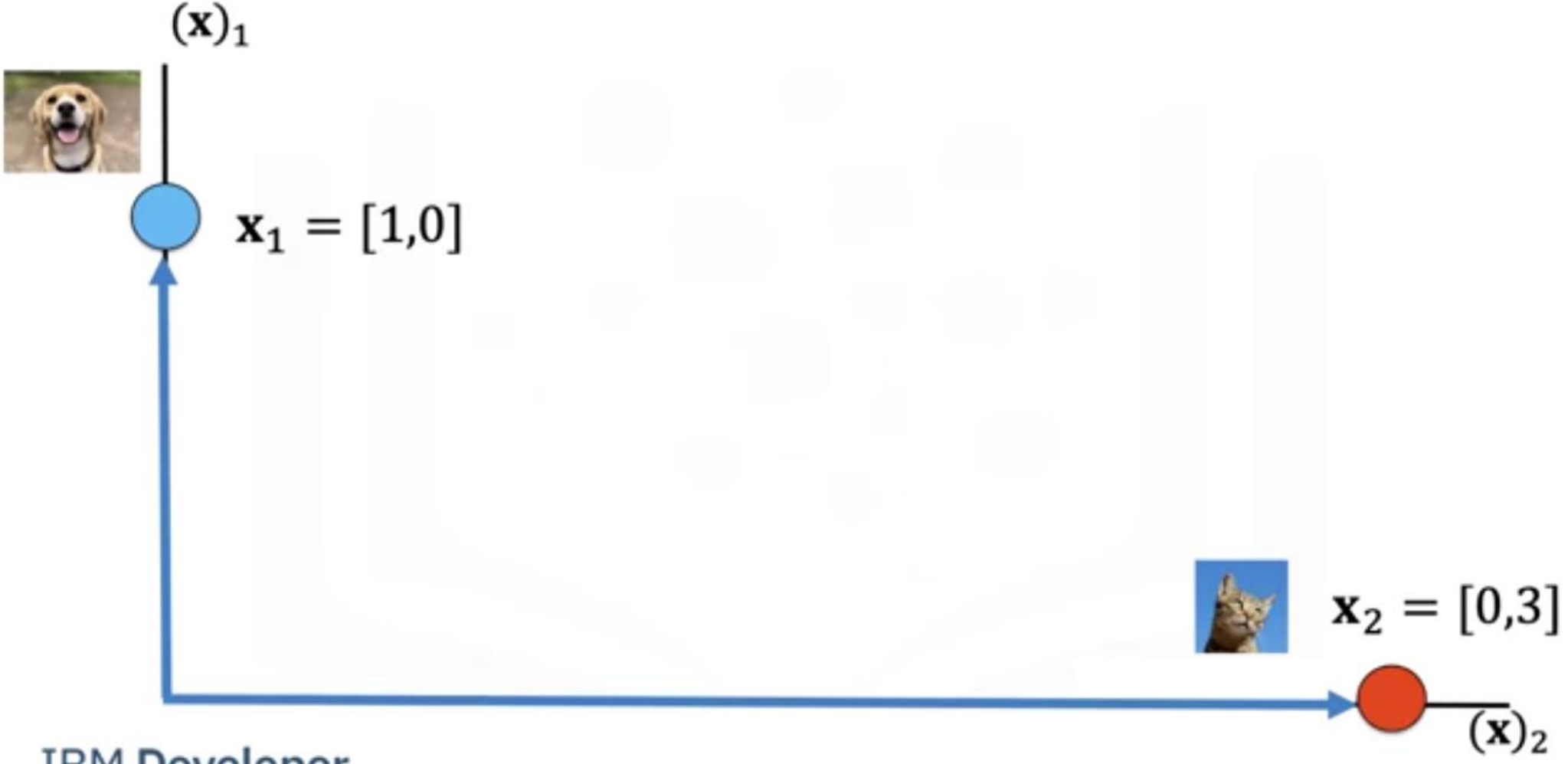


Image Classification with KNN



IBM Developer

Image Classification with KNN

$$d'(\mathbf{x}_1,\mathbf{x}_2) = \|\mathbf{x}_1 - \mathbf{x}_2\|$$

$(\mathbf{x})_1$

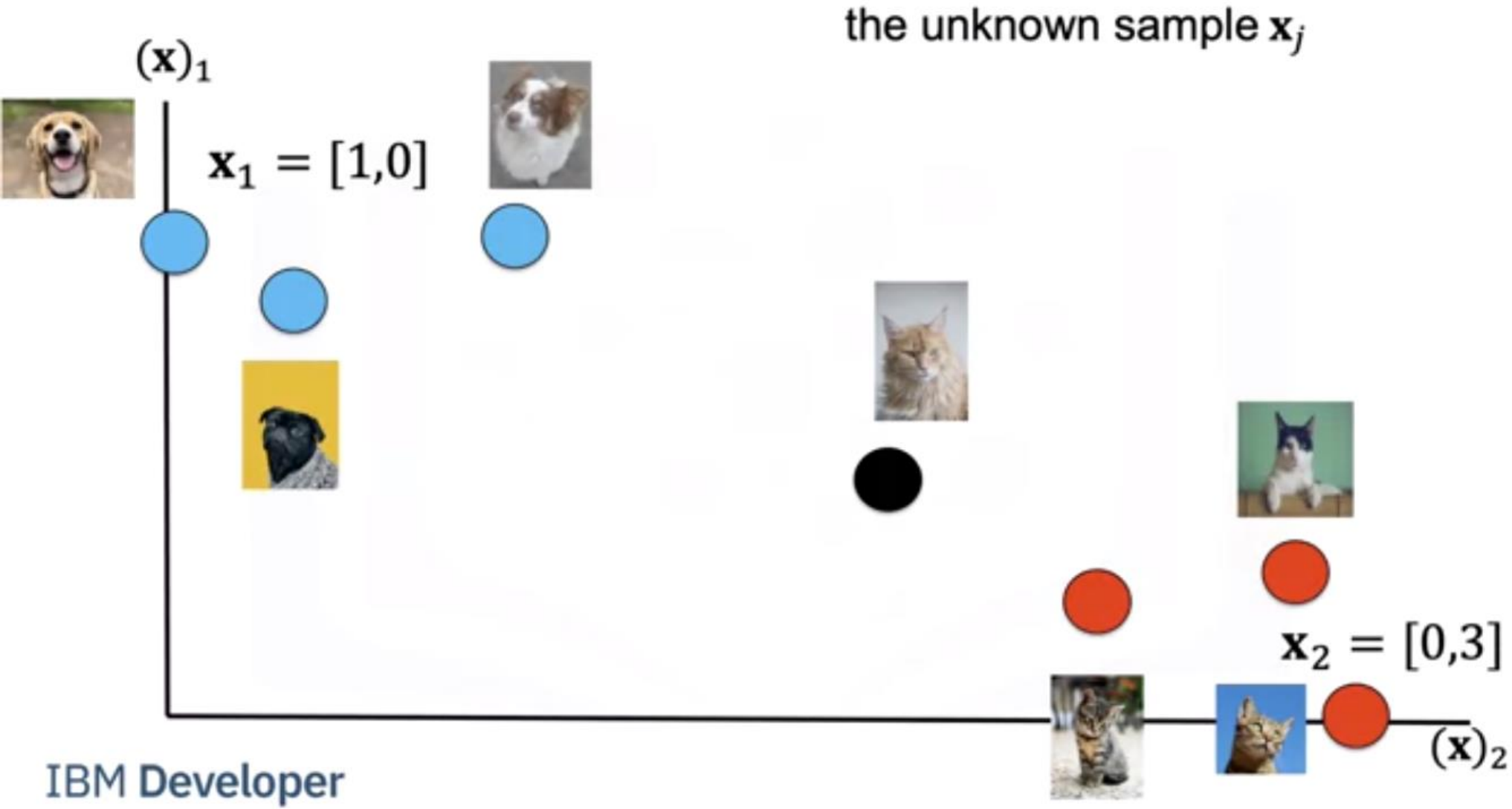
$\mathbf{x}_1 = [1,0]$

$\mathbf{x}_2 = [0,3]$

$(\mathbf{x})_2$

IBM Developer

Image Classification with KNN



IBM Developer

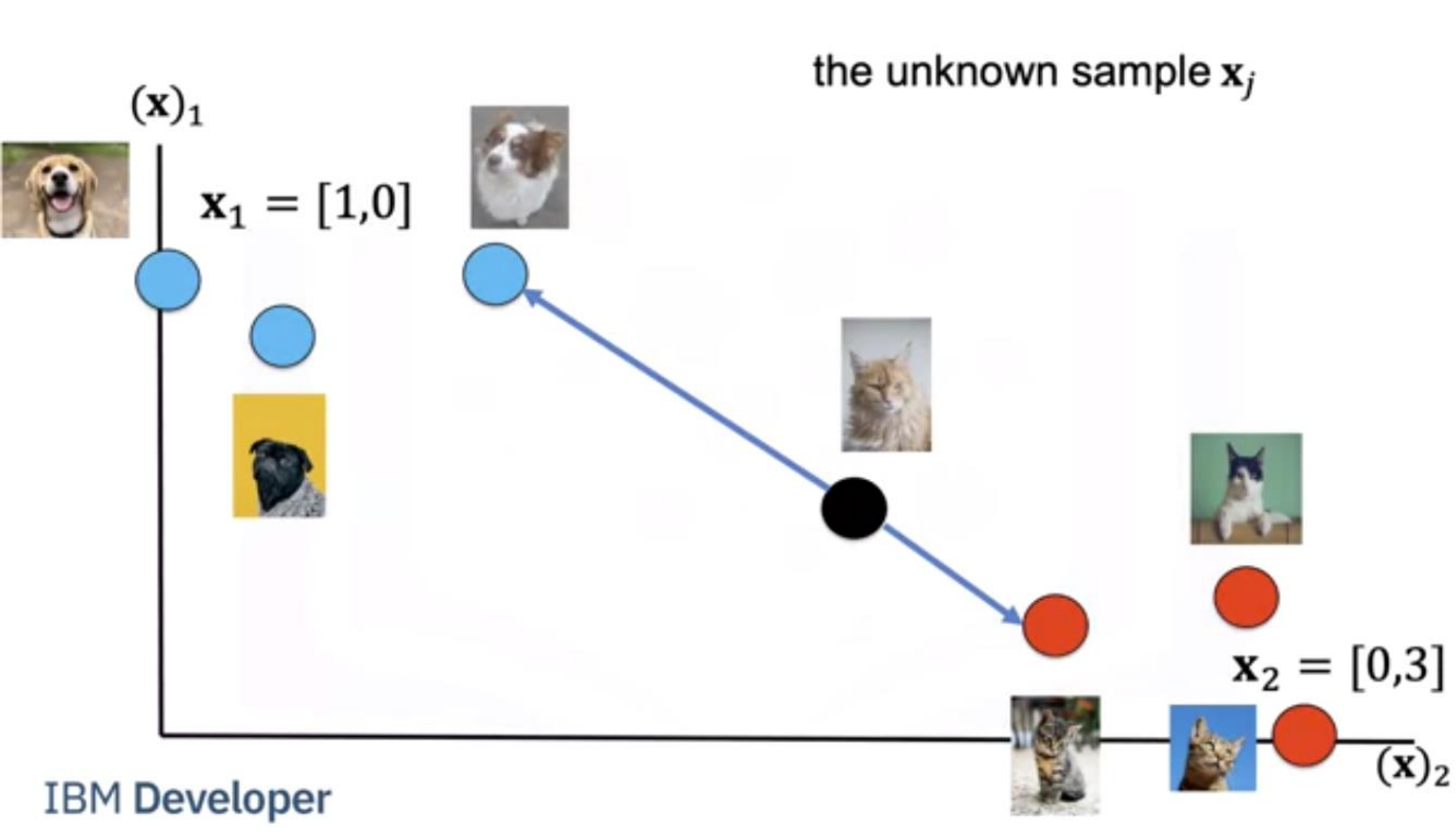

the unknown sample $\mathbf{x}_j$
$(\mathbf{x})_1$
$\mathbf{x}_1 = [1,0]$
$\mathbf{x}_2 = [0,3]$
$(\mathbf{x})_2$

Image Classification with KNN

n	y	$d(\mathbf{x}_n, \mathbf{x}_j)$
1 		
2 		
3 		
4 		
5 		
6 		

the unknown sample

Image Classification with KNN

n	y	$d(\mathbf{x}_n, \mathbf{x}_i)$
1 		
2 		
3 		
4 		
5 		
6 		

$\hat{y} =$

the unknown sample

Image Classification with KNN

n	y	$d(\mathbf{x}_n, \mathbf{x}_i)$
1 		1.0
2 		1.5
3 		1.7
4 		2.0
5 		2.5
6 		3.0

the unknown sample

Image Classification with KNN

n	y	$d(\mathbf{x}_n, \mathbf{x}_j)$
1 		1.0
2 		1.5
3 		1.7
4 		2.0
5 		2.5
6 		3.0

$\hat{y} =$ 

the unknown sample

Image Classification with KNN

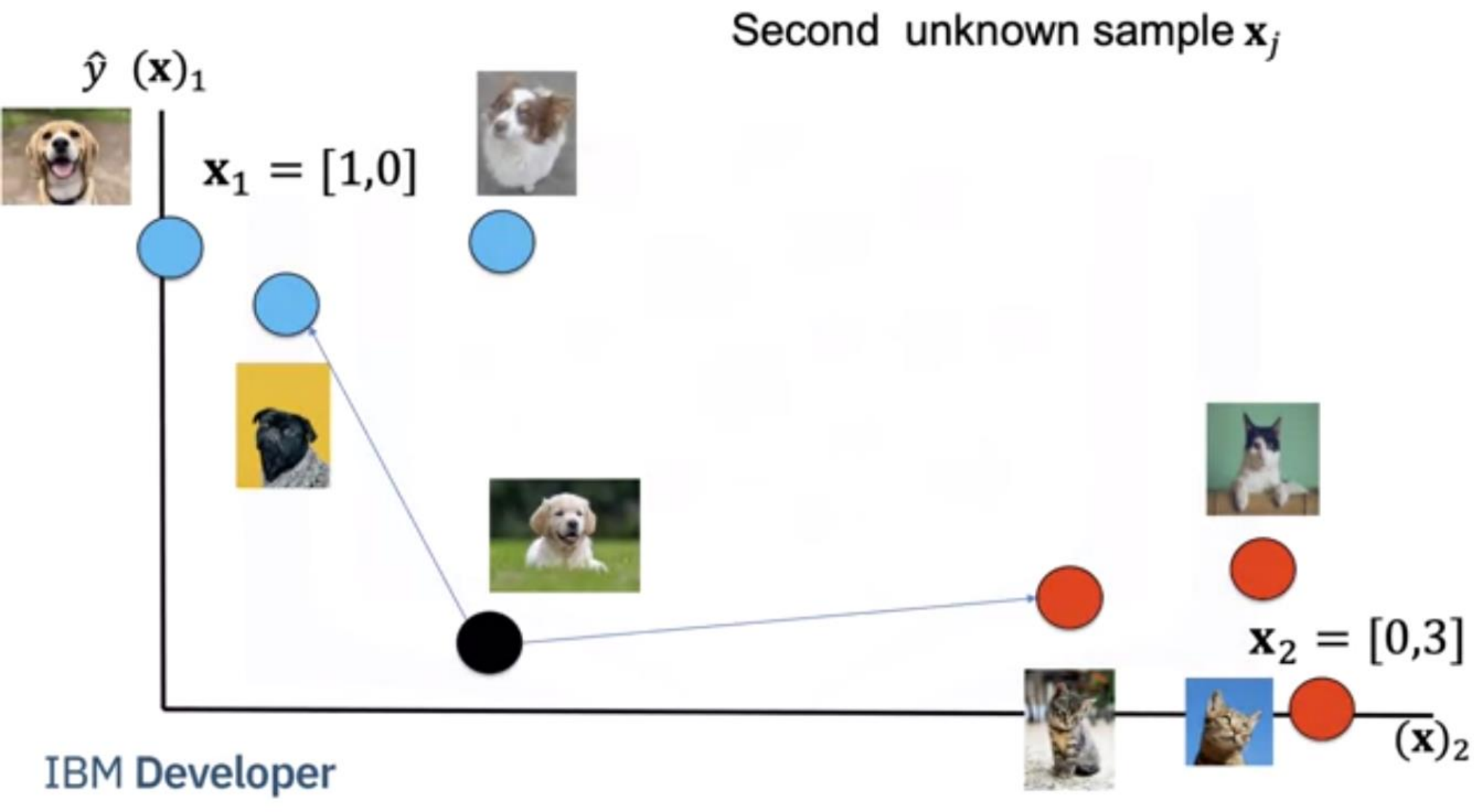


Image Classification with KNN

Training/Testing Sets

- Split dataset into:

Image Classification with KNN

Training/Testing Sets

Data:

- Split dataset into:
  - training set (70%),
  - testing set (30%)
- Build the model with a training set.

Image Classification with KNN

Training/Testing Sets

Data:

- Split dataset into:
  - training set (70%),
  - testing set (30%)
- Build the model with a training set.
- Use testing set to assess the performance.
- When we have completed testing our model we should use all the data

Image Classification with KNN

Accuracy

- What is the accuracy of a classifier?
  - the number of samples that have been predicted correctly divided by the total number of samples

Sample: n	1	2	3	4
y	1	0	1	0
$\hat{y}$	1	0	0	1
$Correct$				

Image Classification with KNN

Accuracy

- What is the accuracy of a classifier?
  - the number of samples that have been predicted correctly divided by the total number of samples

Samples: n	1	2	3	4
y	1	0	1	0
$\hat{y}$	1	0	0	1
$Correct$	1	1	0	0

$$= \frac{1}{4}(1+1+0+0) = 0.5$$

Image Classification with KNN

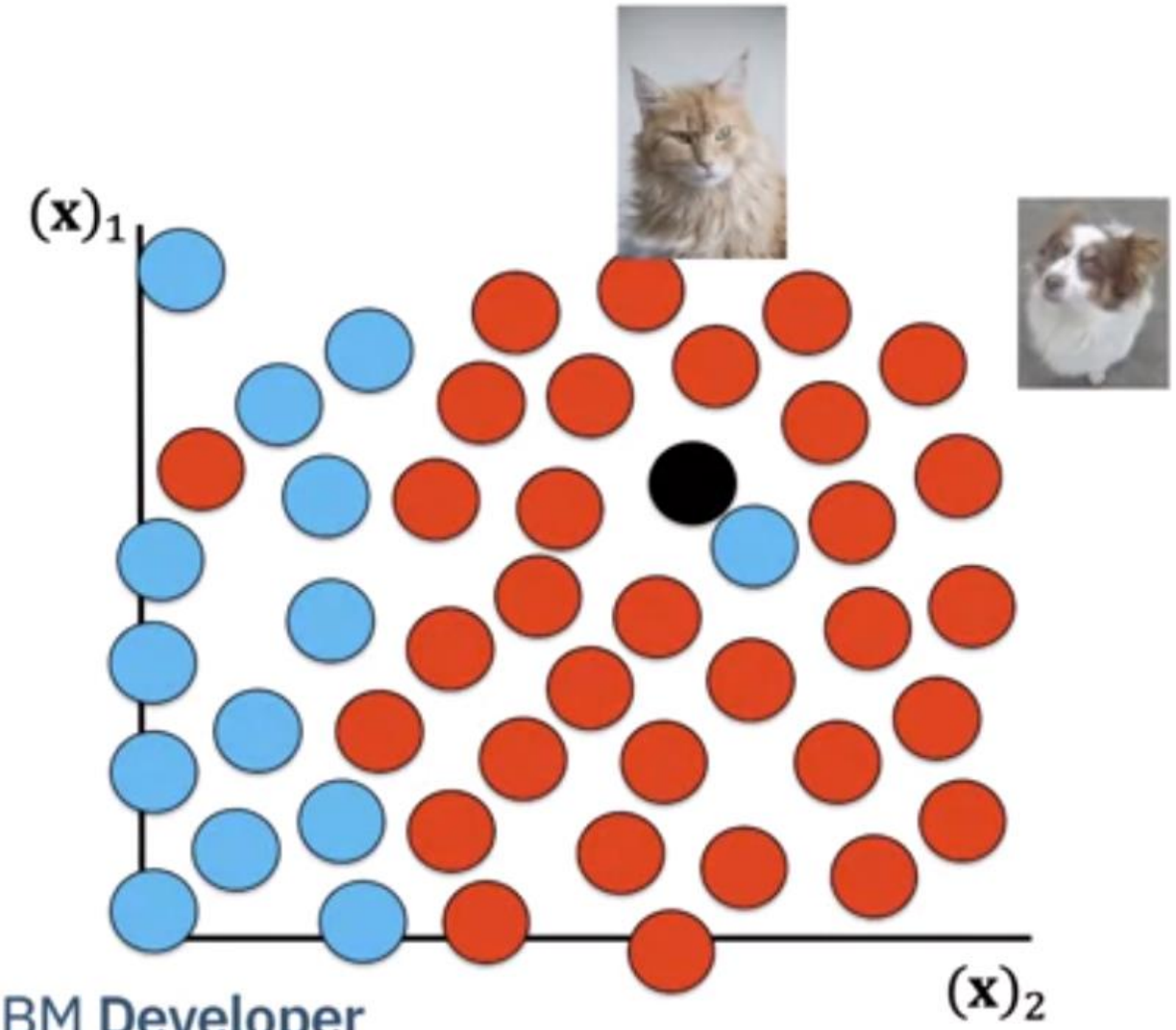


Image Classification with KNN

n	y	$d(\mathbf{x}_n, \mathbf{x}_j)$
1 		1.0
2 		1.5
3 		1.7
4 		2.0
5 		2.5
6 		3.0

$$\hat{y} =$$

Image Classification with KNN

n	y	$d(\mathbf{x}_n, \mathbf{x}_j)$
1 		1.0
2 		1.5
3 		1.7
4 		2.0
5 		2.5
6 		3.0

$\hat{y} =$ 

Image Classification with KNN

Image Classification with KNN

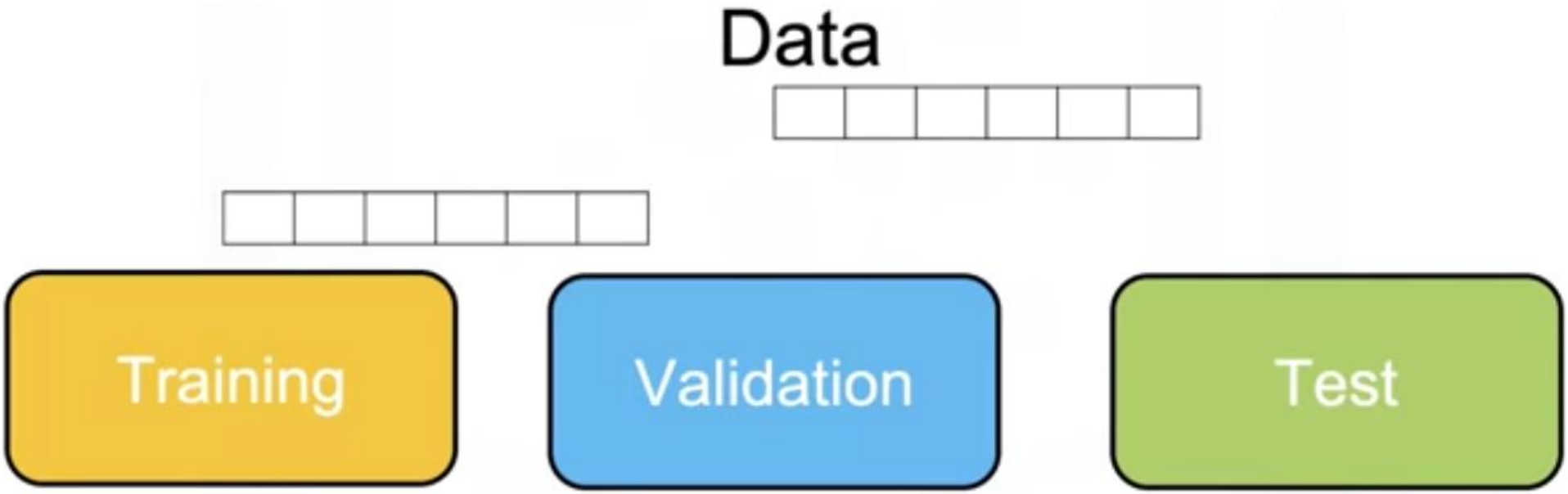


Image Classification with KNN

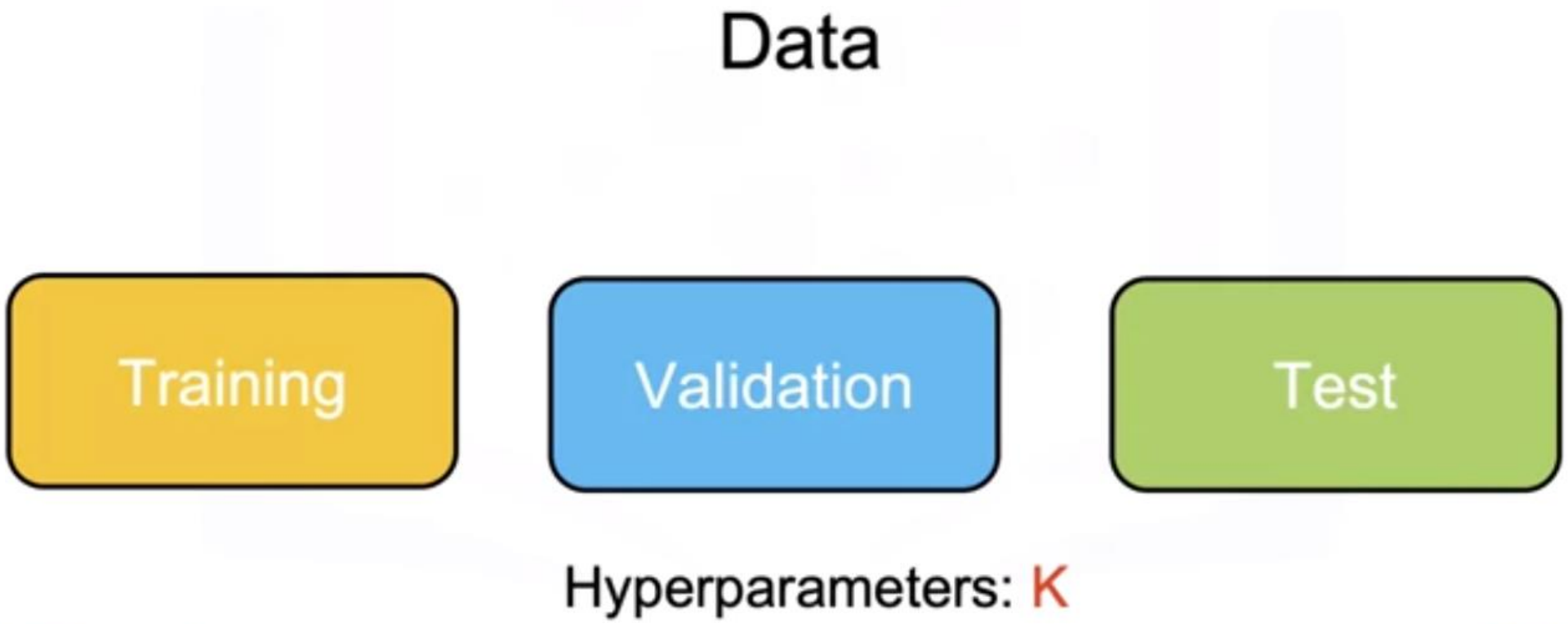


Image Classification with KNN

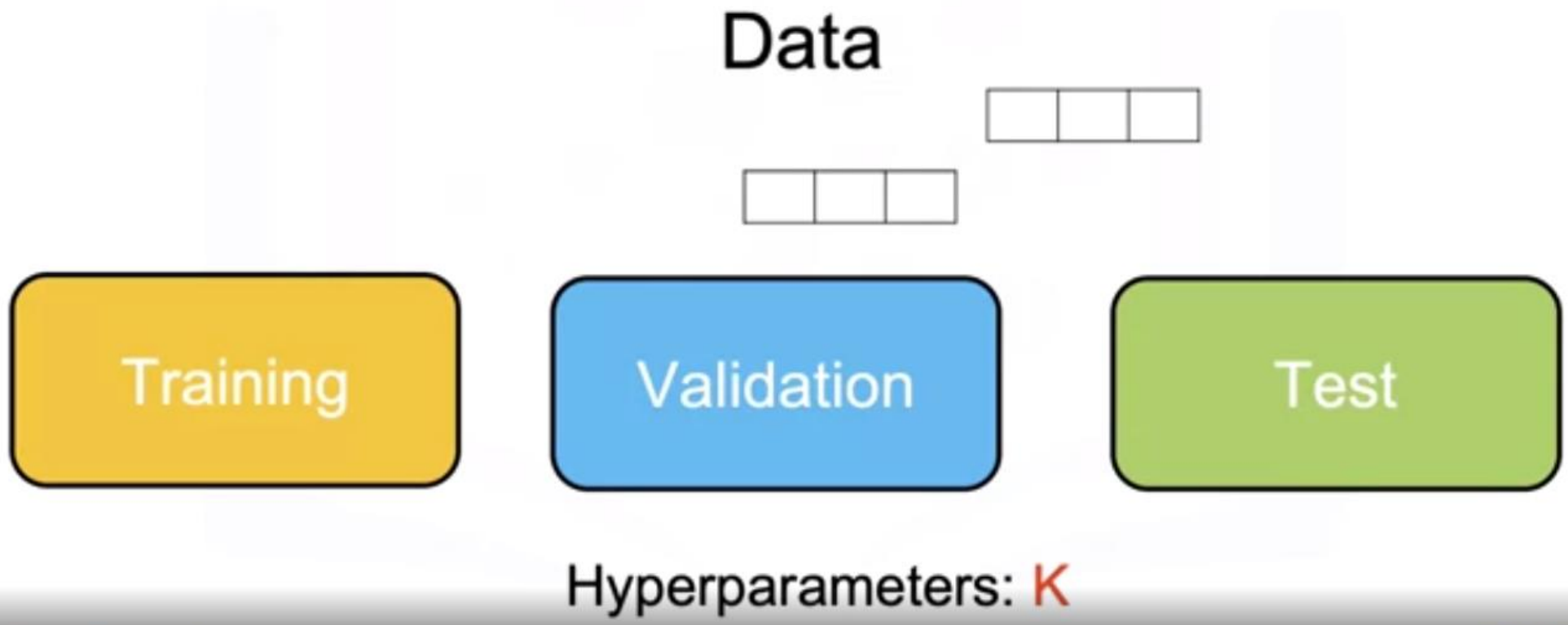


Image Classification with KNN

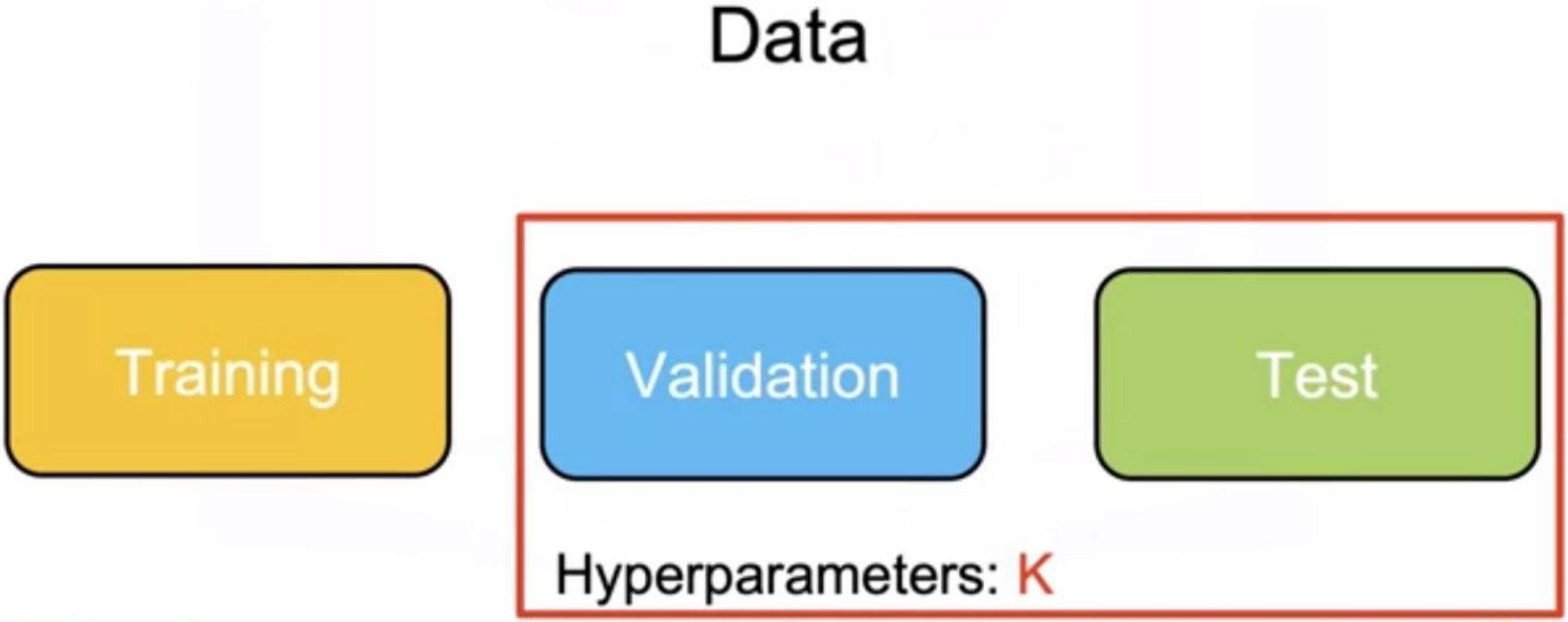
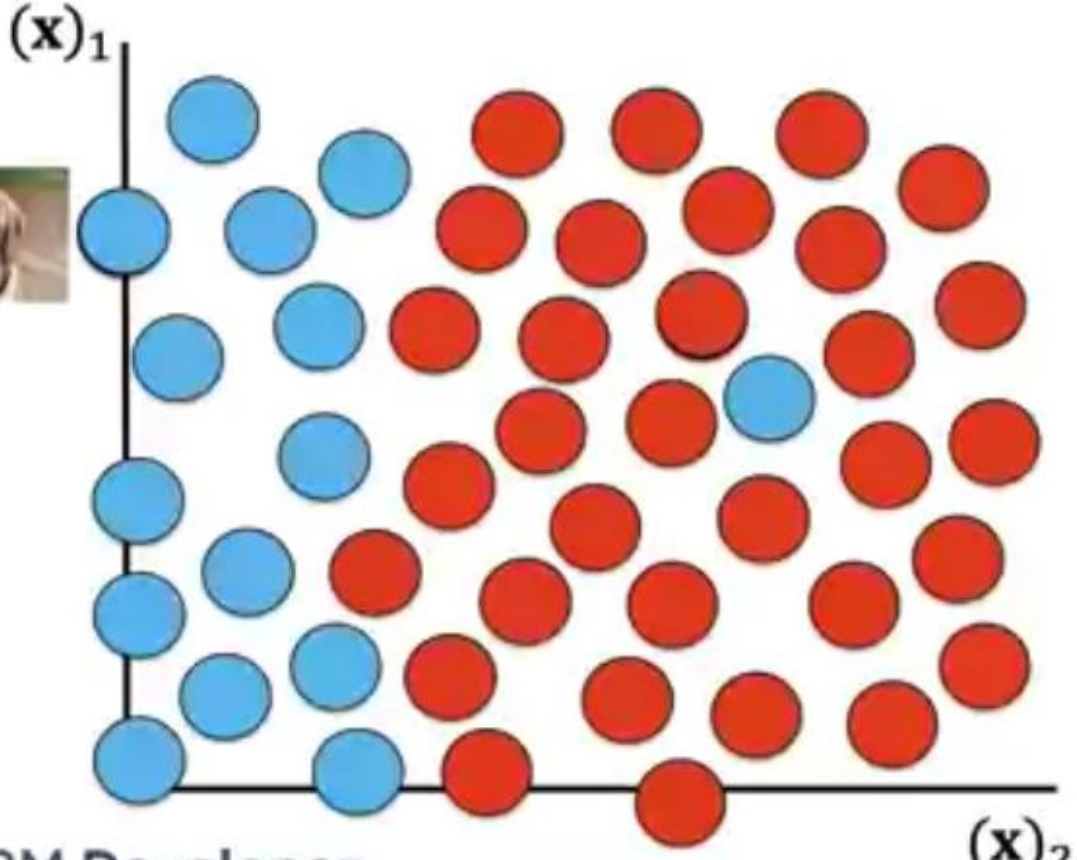


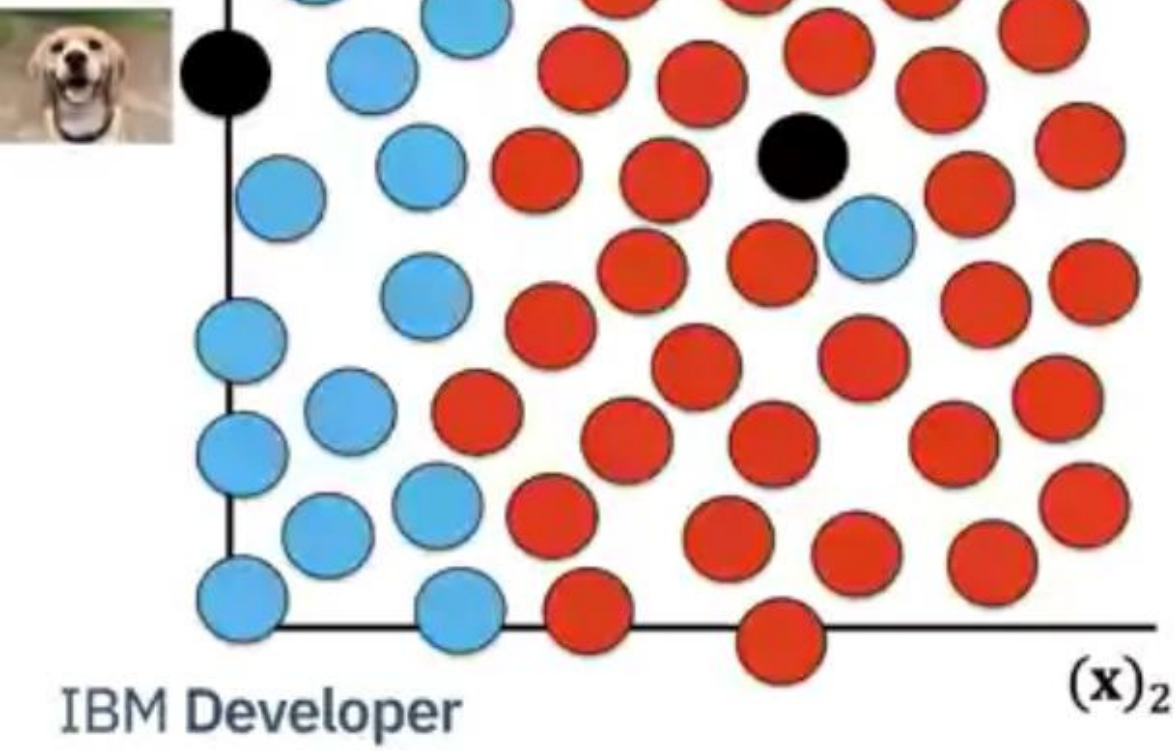
Image Classification with KNN



Validation Data

$\hat{y}$ \ y			Accuracy
K=1			
K=3			

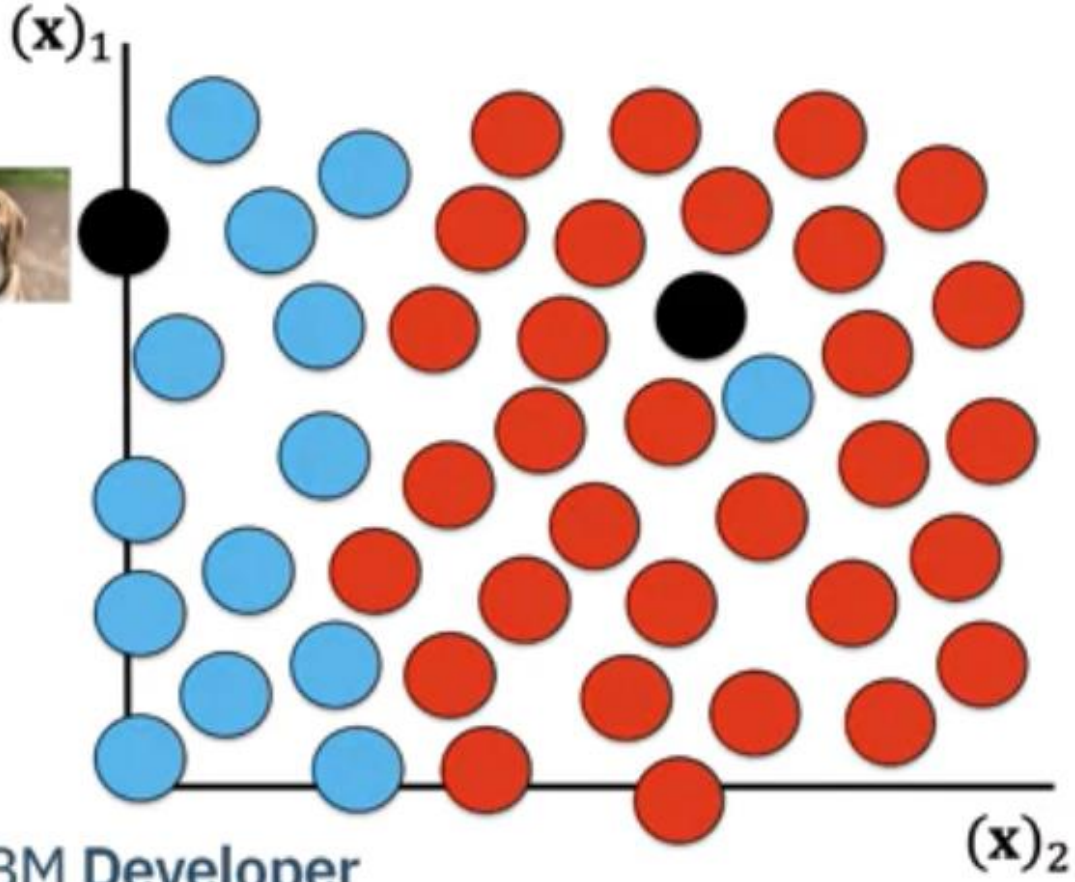
Image Classification with KNN



Validation Data

$\hat{y}$ \ y			Accuracy
K=1			
K=3			

Image Classification with KNN



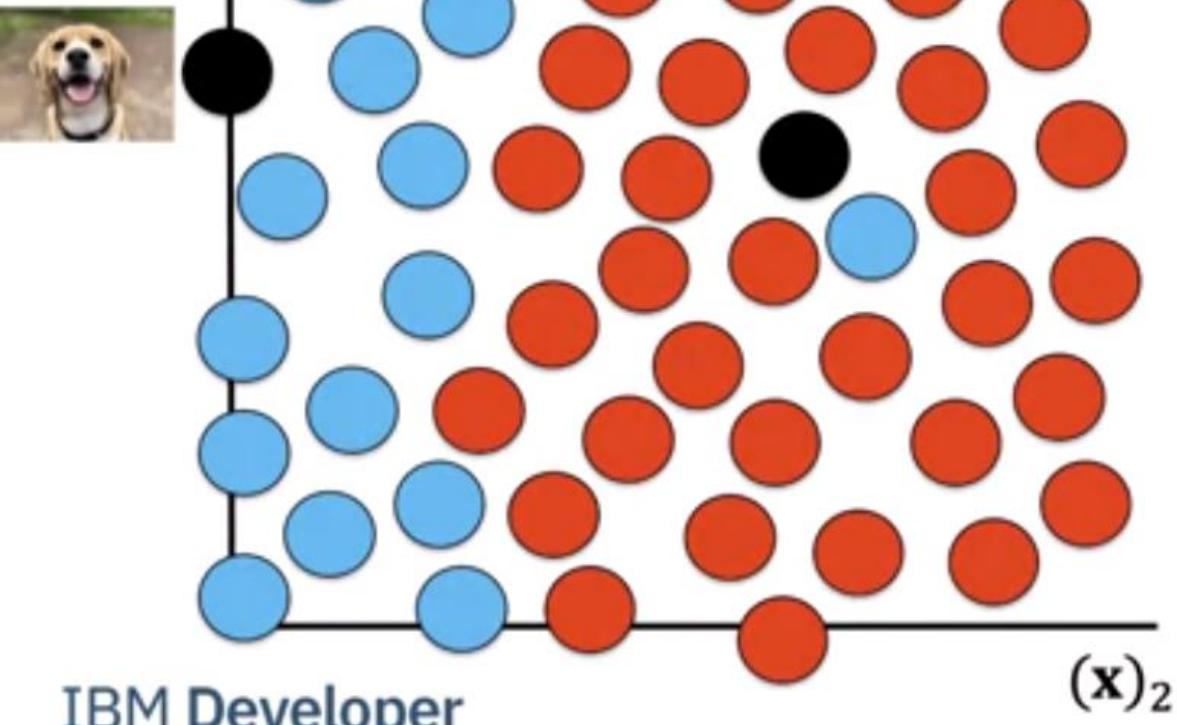
Validation Data

$\hat{y}$ \ y			Accuracy
K=1			50%
K=3			

IBM Developer

SKILLS NETWORK

Image Classification with KNN



Validation Data

$\hat{y}$ \ y			Accuracy
K=1			50%
K=3			100%

Image Classification with KNN

n	y	$d(x_n, x_i)$
1 		1.0
2 		1.5
3 		1.7
4 		2.0
5 		2.5
6 		3.0

Image Classification with KNN

Multiclass :

$y = 0$

$y = 1$

$y = 2$

Image Classification with KNN

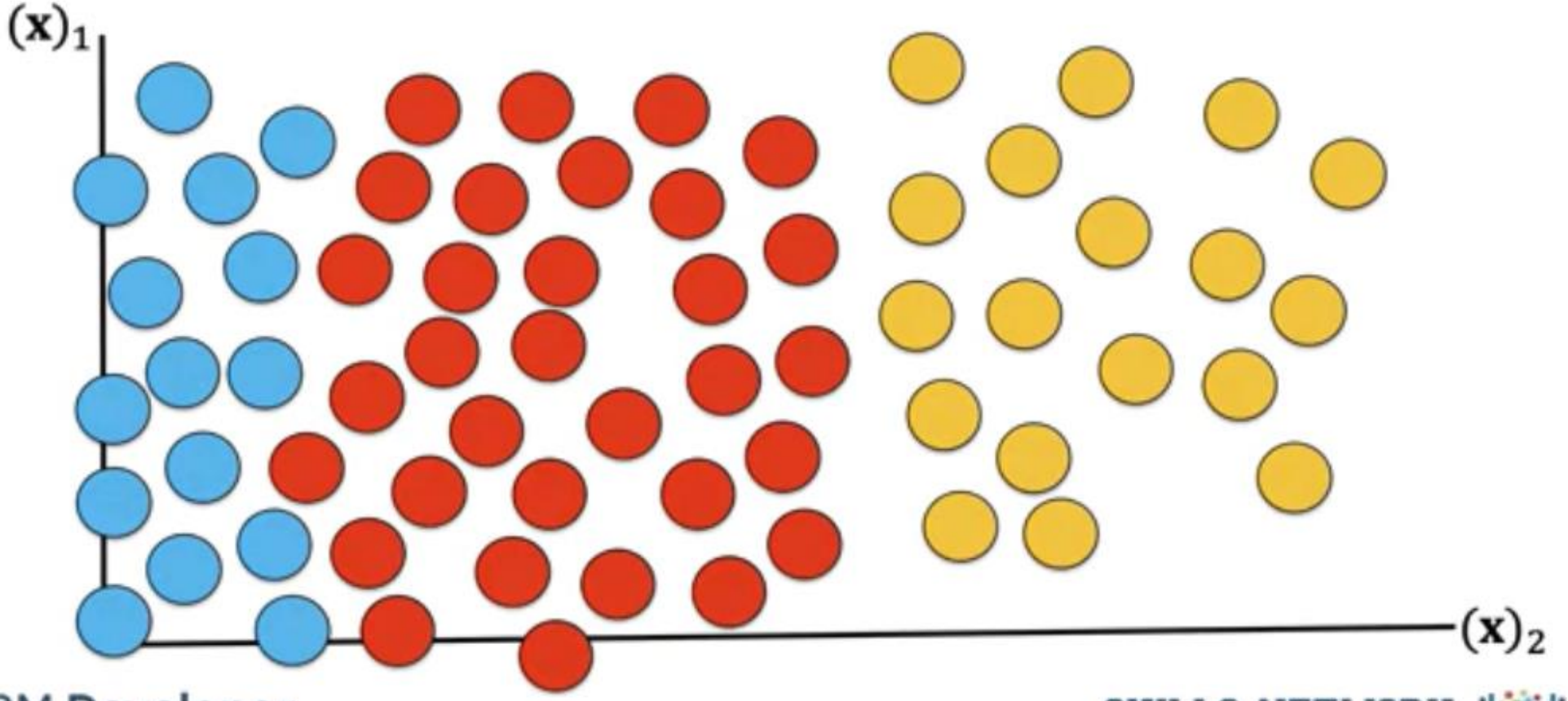
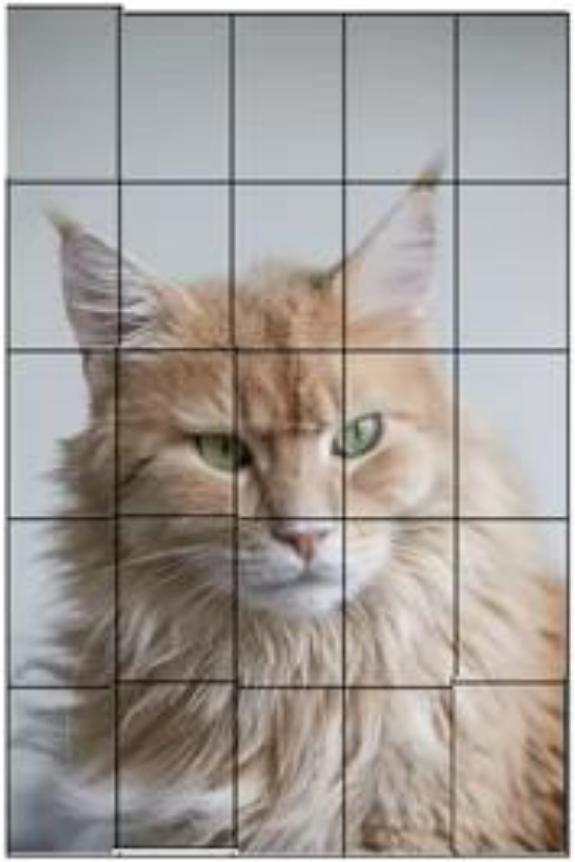


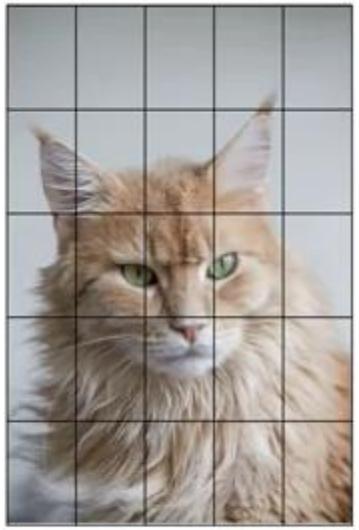
Image Features

Image Features



$$x = \begin{bmatrix} \\ \vdots \\ \\ \\ \vdots \\ \end{bmatrix}$$

Image Features



IBM Developer

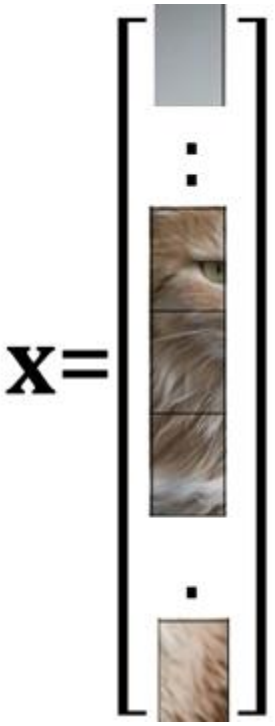
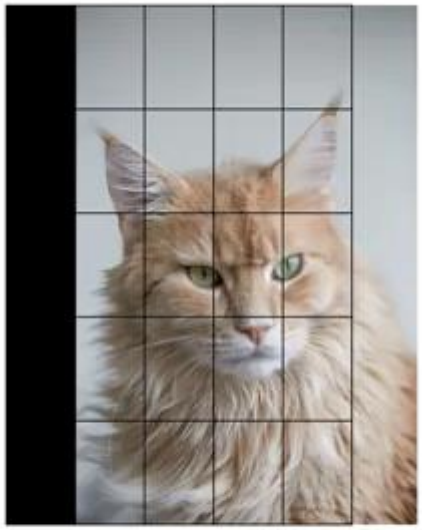


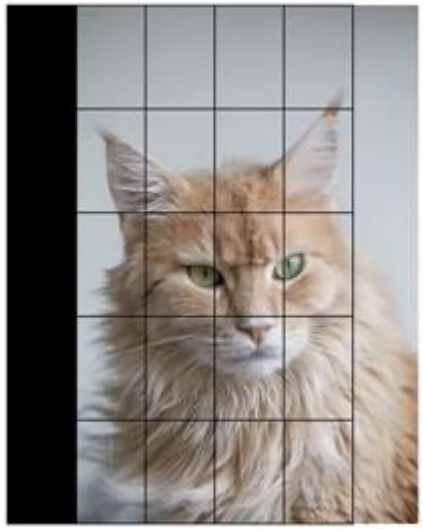
Image Features



IBM Developer

$$x = \begin{bmatrix} \\ \vdots \\ \\ \vdots \\ \\ \end{bmatrix}$$

Image Features



IBM Developer

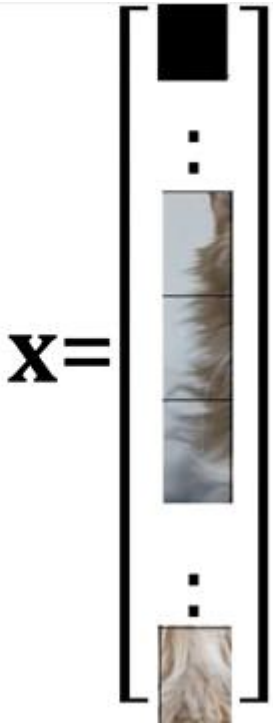


Image Features

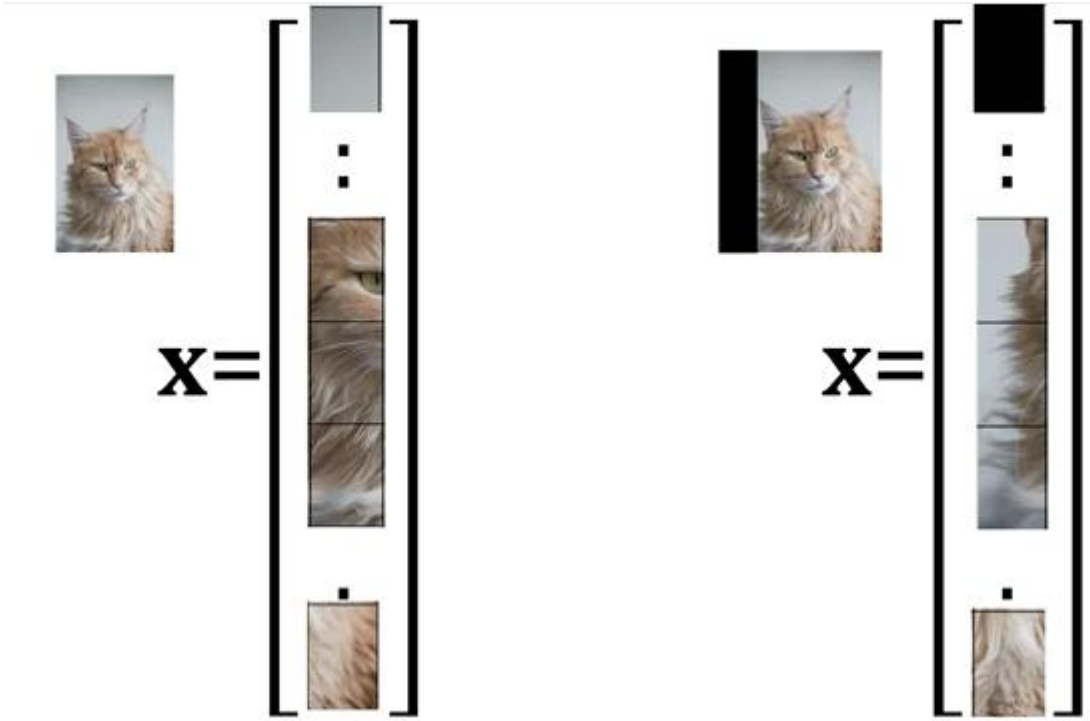
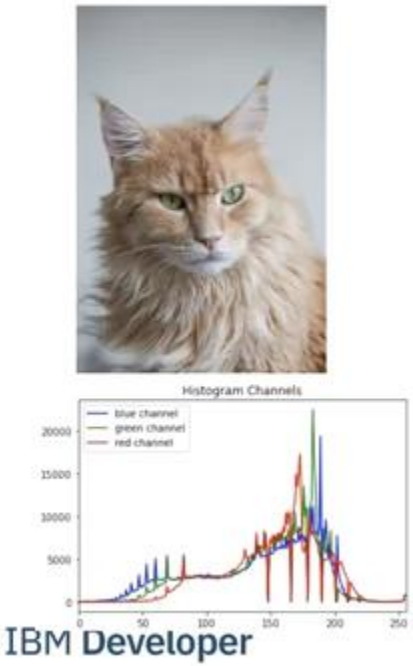


Image Features



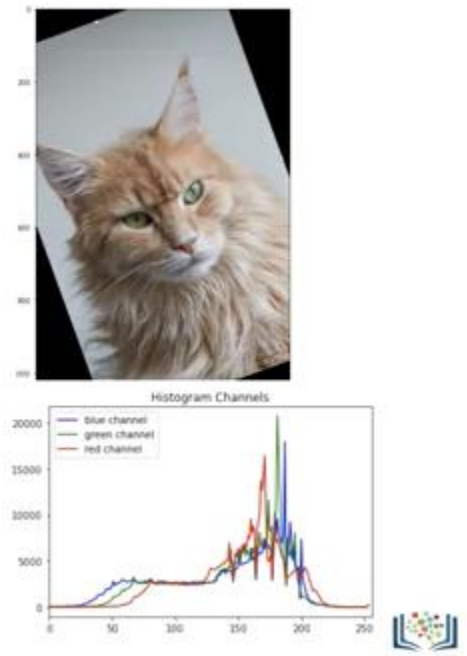
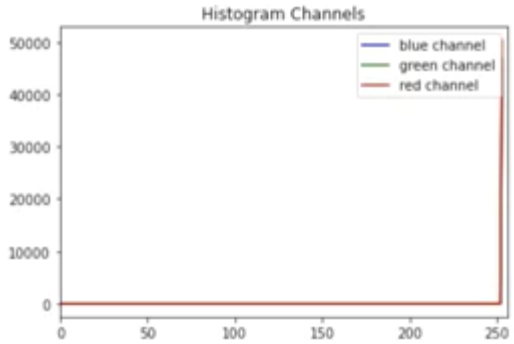


Image Features



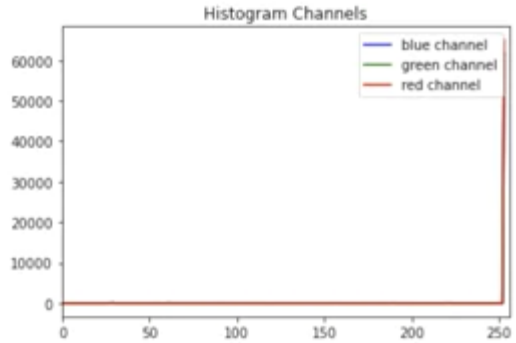
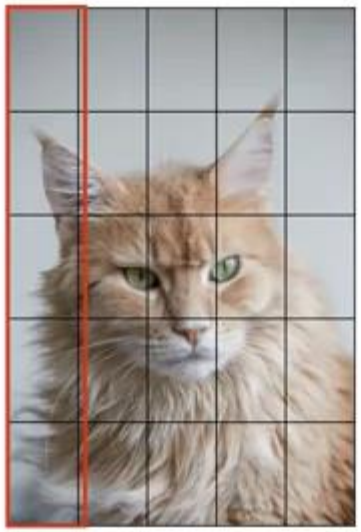


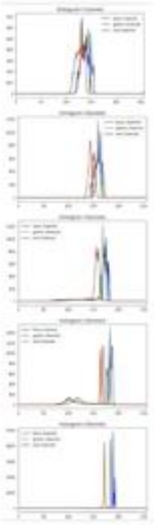
Image Features

IBM Developer

SKILLS NETWORK

Image Features

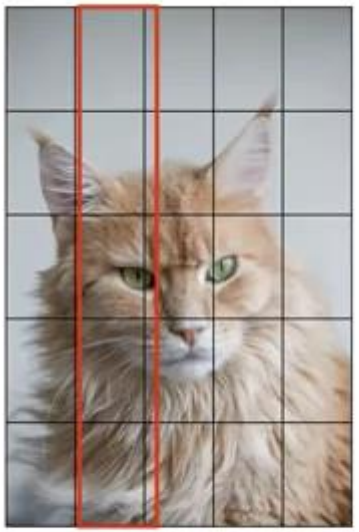


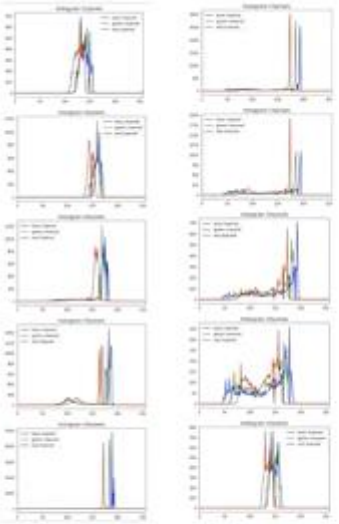


IBM Developer

SKILLS NETWORK

Image Features

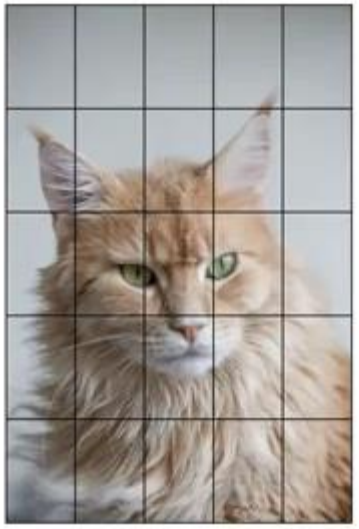


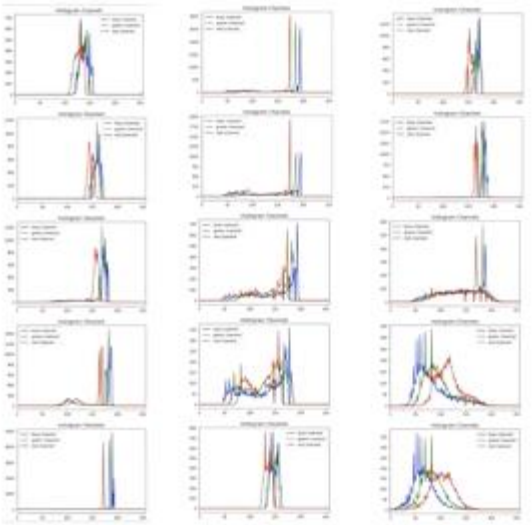


IBM Developer

SKILLS NETWORK

Image Features





IBM Developer

SKILLS NETWORK

Image Features

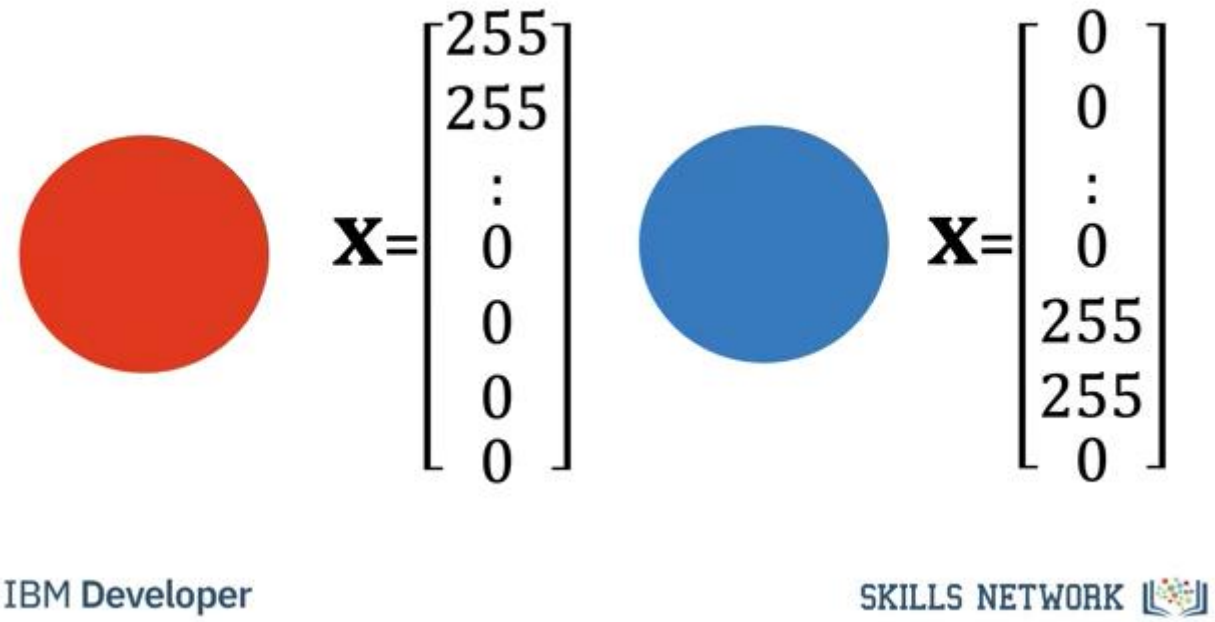
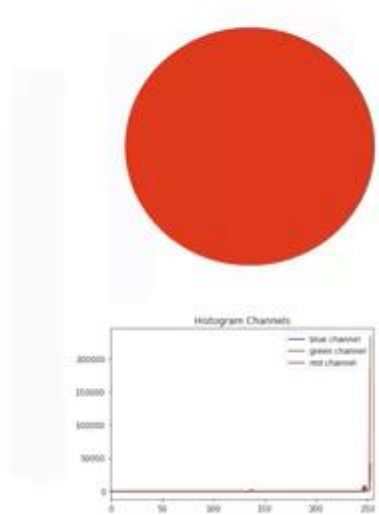
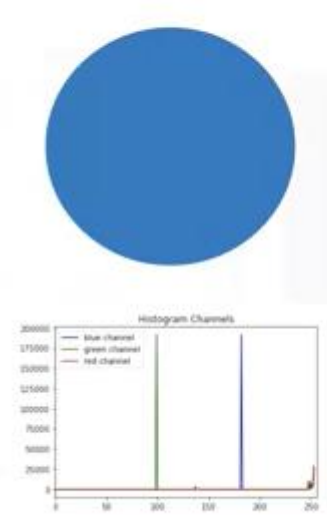


Image Features

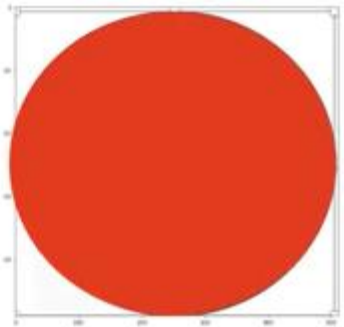


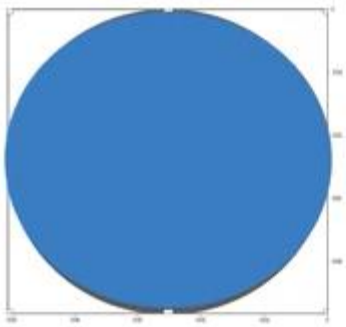


IBM Developer

SKILLS NETWORK

Image Features

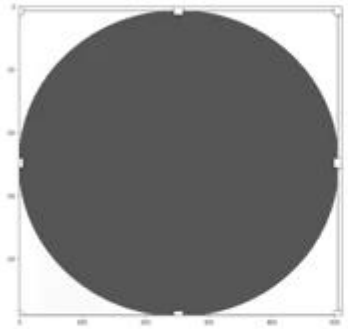




IBM Developer

SKILLS NETWORK

Image Features



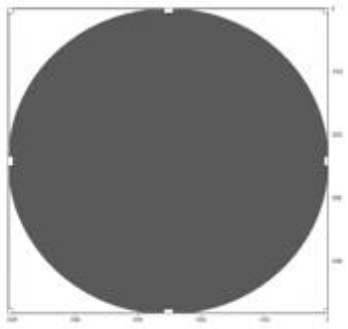
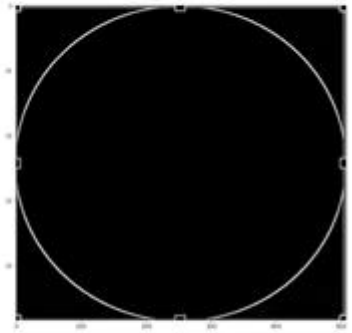
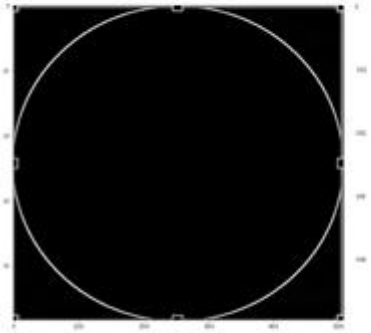


Image Features





IBM Developer

SKILLS NETWORK

Image Features

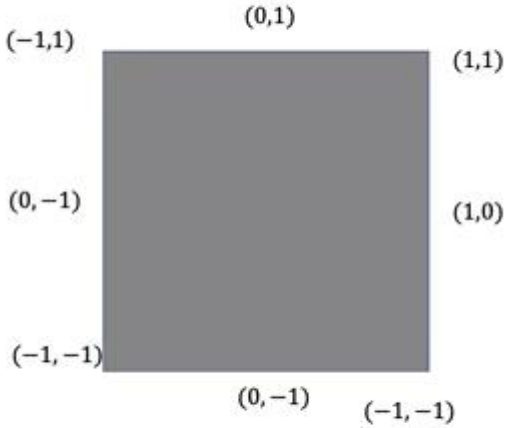
What is H.O.G.

- H.O.G. stands for Histogram of Oriented Gradients
- H.O.G uses the gradient orientation of the localized regions of an image
- H.O.G. generates a histogram for each localized region

IBM Developer

SKILLS NETWORK

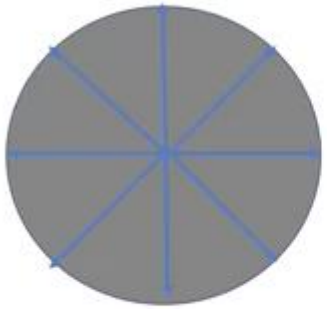
Image Features

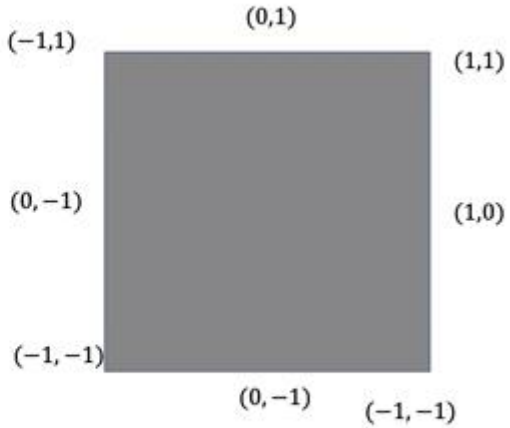


IBM Developer

SKILLS NETWORK

Image Features

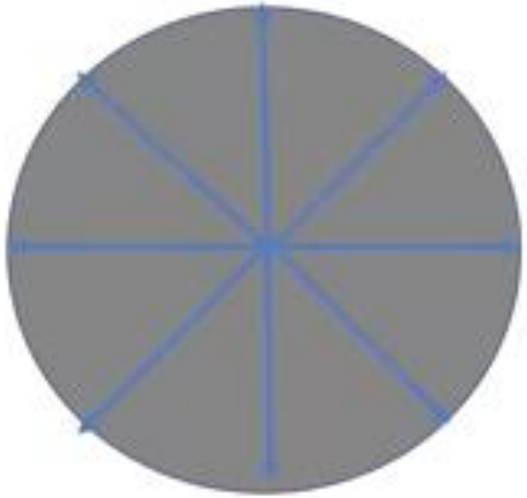


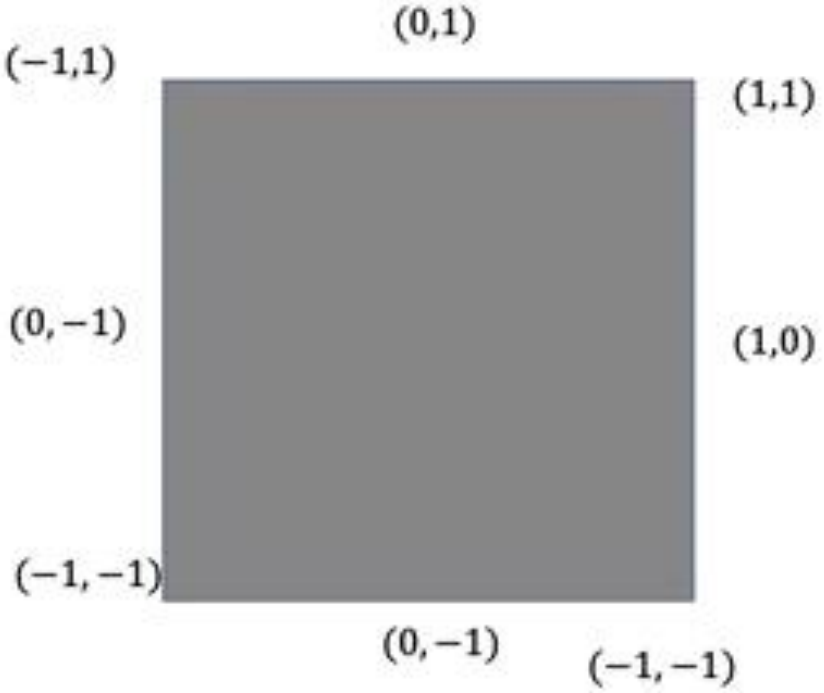


IBM Developer

SKILLS NETWORK

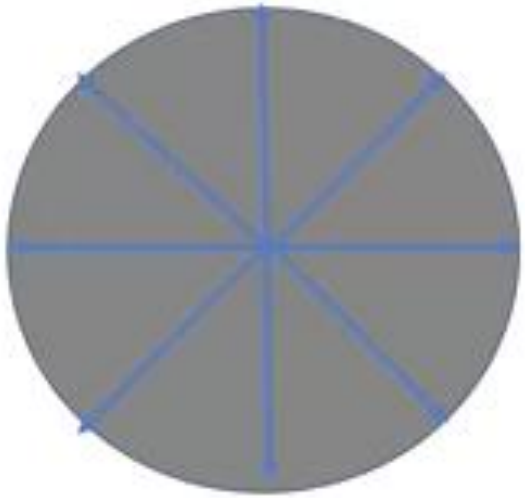
Image Features

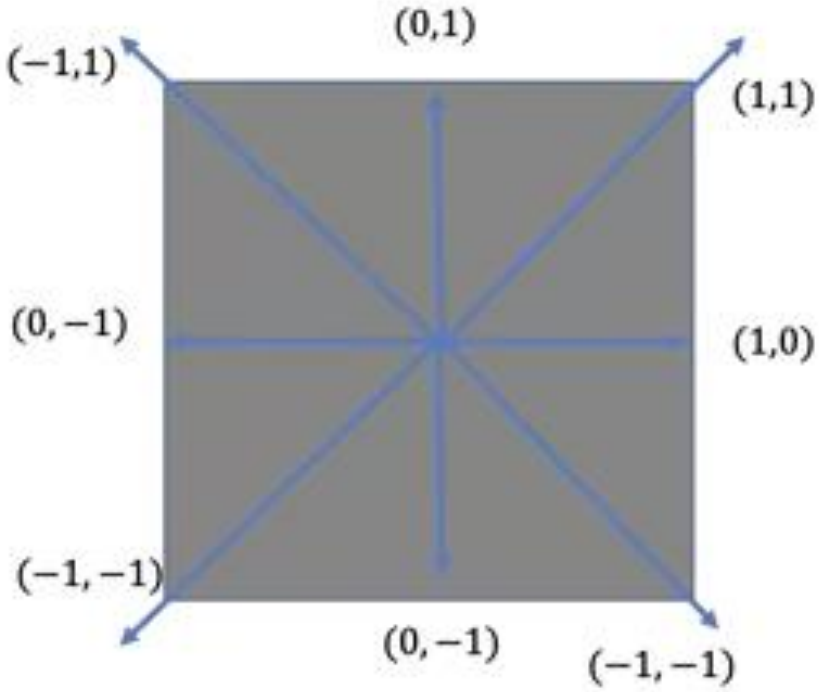




	1	1	1	1	1	1	1
θ	0	45	90	135	180	225	270

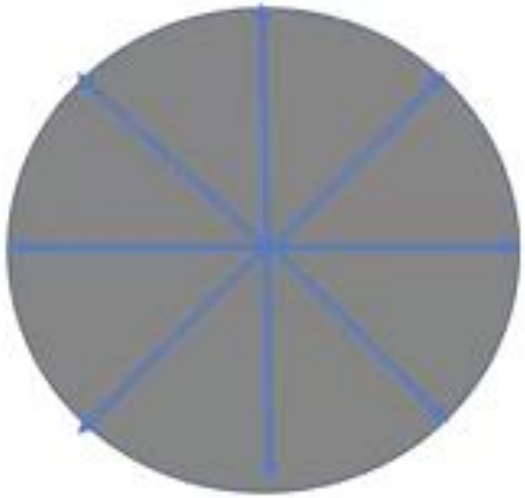
Image Features



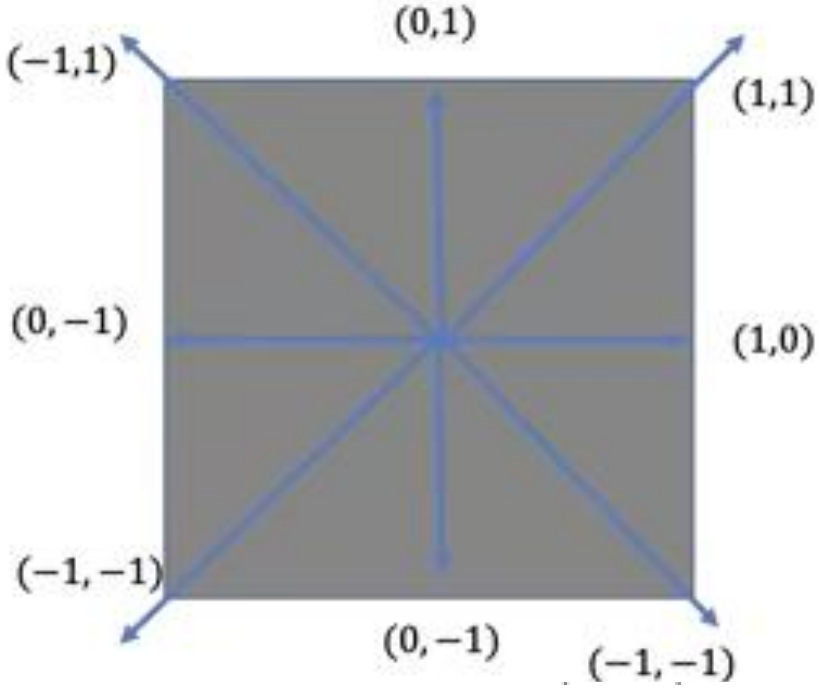


	1	1	1	1	1	1	1
θ	0	45	90	135	180	225	270

Image Features



	1	1	1	1	1	1	1
θ	0	45	90	135	180	225	270



	1	$\sqrt{2}$	1	$\sqrt{2}$	1	$\sqrt{2}$	1
θ	0	45	90	135	180	225	270

Image Features

H.O.G.

Image Features

H.O.G.

$$\|G\| = \sqrt{G_x{}^2 + G_y{}^2}$$

$$\theta = \arctan\left(\frac{G_y}{G_x}\right)$$

Image Features

H.O.G.

$$\|G\| = \sqrt{G_x{}^2 + G_y{}^2}$$

$$\theta = \arctan\left(\frac{G_y}{G_x}\right)$$

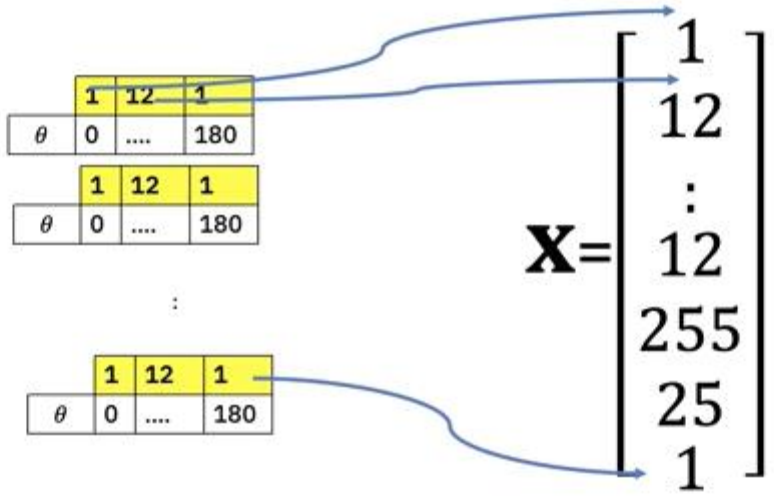
	1	12	1
θ	0	....	180

	1	12	1
θ	0	....	180

:

	1	12	1
θ	0	....	180

Image Features



IBM Developer

SKILLS NETWORK

Tài liệu tham khảo

- https://www.coursera.org/learn/introduction-computer-vision-watson-opencv

Thực hành

Yêu cầu

- Phân lớp ảnh: dogs vs cats
- Phương pháp: KNN
- Features: histogram
- Color: grayscale

Project structure

CS231.

```
├──Dataset
│      └──dogscats_small
│             ├──train
│             └──test
├──Baitap3
│      └──MSSV.ipynb
```

Tính histogram cho ảnh

```
#Hàm tính histogram

def TinhHist(pathfilename):
    img = cv.imread(pathfilename,0)
    hist = cv.calcHist([img],[0],None,
                       [256],[0,256])
    size = img.shape[0]*img.shape[1]
    hist = hist / size
    return hist
```

Code: gợi ý

```
def ReadData(path_to_files):
  for file_name in os.listdir(path_to_files):
    # Lấy label của ảnh
    # -> Thêm vào danh sách labels
    # Tính đặc trưng của ảnh
    # -> Thêm vào danh sách features
  return features, labels
```

Code: gợi ý

- `os.listdir(path_to_files)` → đọc tất cả file trong thư mục path_to_files

- `Sorted (list_files)` → sắp xếp danh sách list_files theo thứ tự tăng dần

Code: gợi ý

```
from sklearn.neighbors import KNeighborsClassifier

---
label = imagePath.split(os.path.sep)[-1].split(".")[0]

----
features = np.array(features)
labels = np.array(labels)

---
model = KNeighborsClassifier(n_neighbors=1)
model.fit(features, labels)
acc = model.score(test_features, test_labels)
```