Day 3

Welcome to Day 3
AI Toolkit

Please perform PRE-WORK

1. Access the virtual Lab using link https://html.inspiredvlabs.com Use the username TEKBD142-XX (replace XX with your number) and password

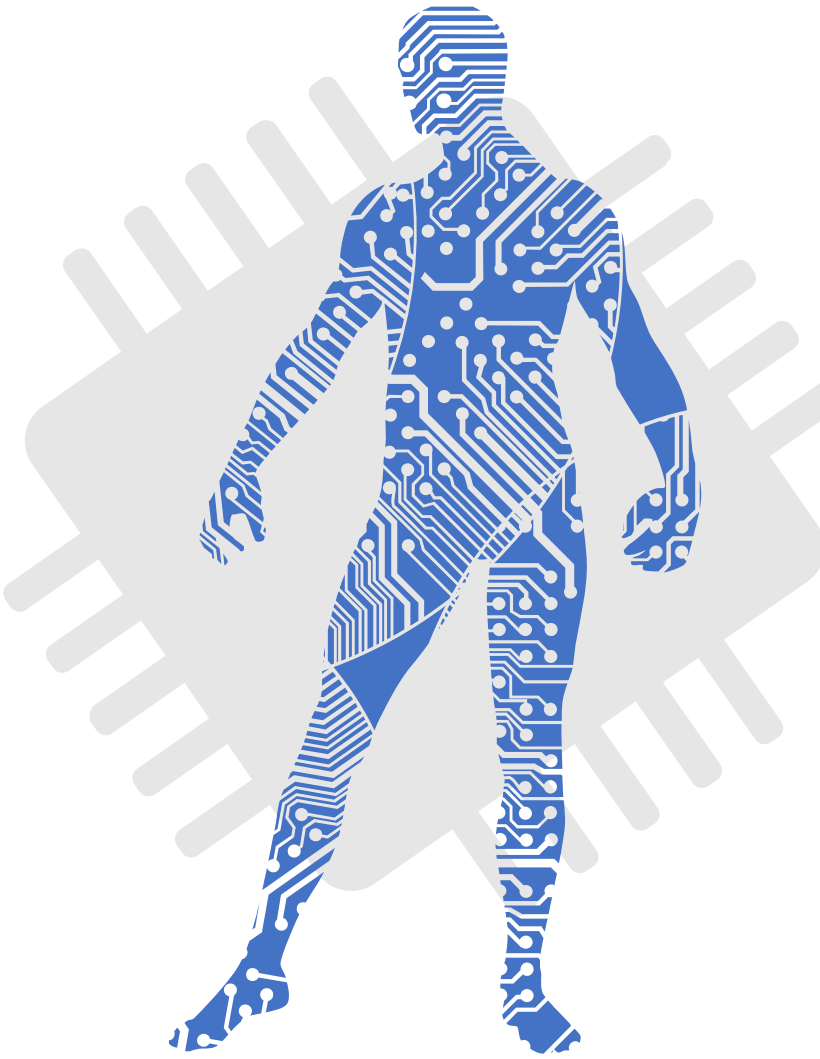
TekBD142!23

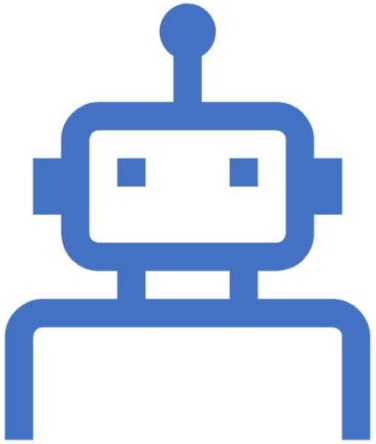
We will be starting soon

Last Name	First Name	Login Id
AHMED	NAZEER	TEKBD142-01
AM	GANESH	TEKBD142-02
BISWAS	SOURAV	TEKBD142-03
CHACKO	THOMAS	TEKBD142-04
JAJOO	SANDESH	TEKBD142-05
MATTOX	CHRISTEL	TEKBD142-06
MAXSON	CRAIG	TEKBD142-07
MURPHY	MATTHEW	TEKBD142-08
PANDIAN	JEYARAJ	TEKBD142-09
PATURI	RAVIKIRAN	TEKBD142-10
RANI	AMITA	TEKBD142-11
SINGLA	SANJEEV	TEKBD142-12
VEERAMASU	BRAHMA RAO	TEKBD142-13

Agenda – Day 3

1. Recap of Day 2
2. Neural Networks – Classification
3. Image Processing (CNN)
4. Recurrent Neural Networks (RNN)
5. Extra Credits
6. Natural Language Processing (NLP)
7. Assignment Solutions
8. Next Steps





Recap Day - 2

- Logistic Regression
- Model and Pipeline Persistence
- Word Cloud
- TensorFlow
- Google’s Colab
- Artificial Neural Networks
- Multiple Hands-on

Neural Networks - Classification

- Open file '**codeSamples/PredictPetFinder**' using Jupyter
- Pet adoption dataset that we used with Word Cloud
- Want to take our Adoption Speed and use it to do binary classification
- If the value is 4 then the pet does not get adopted
- Any other value the pet gets adopted
- Build a neural network to classify

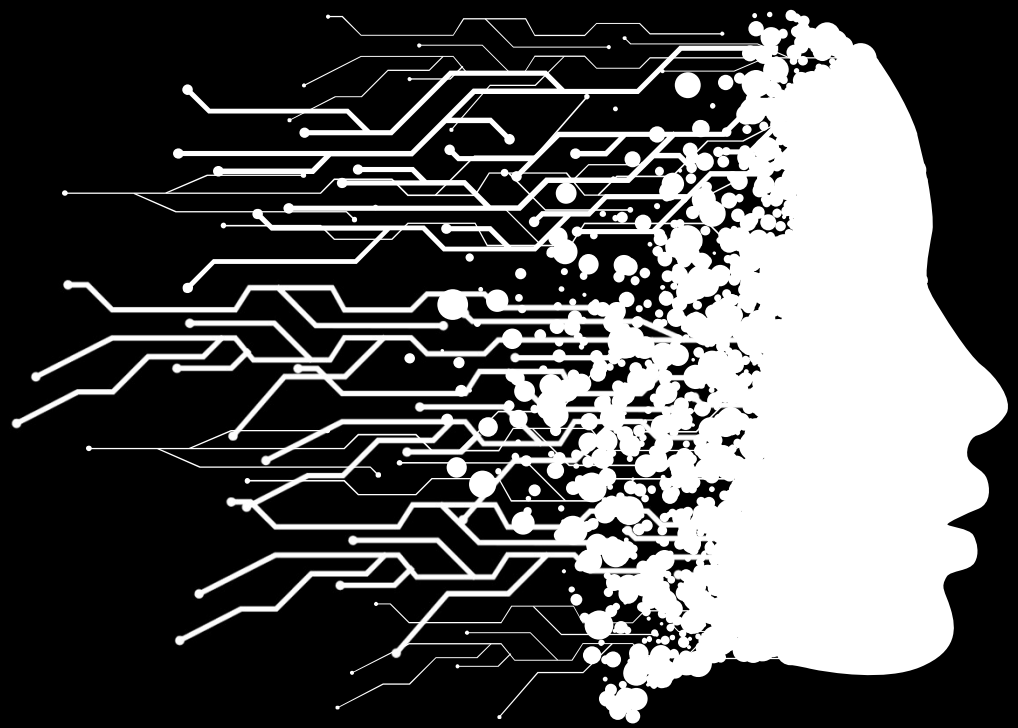


Image Processing

Image – Bunch of Bytes

```
[ [216, 203, 125,  10,  84, 241, 149, 159, 212, 118, 135, 158,  11,  91,  36, 177, 176, 253, 132, 210, 159,  20, 153, 131, 132,  55, 16, 132],
  [184,  34,  95, 225,  60, 218,  49, 193,  93, 119,  68, 133, 195, 104, 248,  18,  18, 136,  90,  71,  81,  41, 233,  53,  46,  87, 86, 243],
  [ 85,  61, 220, 170, 206,  34, 141,  97,  66, 217, 124, 143, 241, 205,  76, 123,  66,  72, 231, 116, 244,  74, 155, 144,  47, 230, 171, 165],
  [156,  87, 181,  90, 160,   2, 184, 112, 108,  62, 223, 153,  93, 244,  83, 187,  83,  18, 134,  28, 121, 244, 202, 176, 228, 233, 76,  13],
  [ 76, 238, 128, 183, 119, 130,  34,  12, 112, 254,  90, 167,  64,  89, 170, 221, 196,  69,  82,  11,  65,  86, 254, 111, 134,   0, 148, 246],
  [105, 178, 254,  31,  32, 133,  57,  40,   6,  85, 115,  56, 132,  64,  35, 119, 158, 182, 106,  77,  84, 106, 164, 230,  54,  42, 55, 130],
  [ 25,  86, 222,  59, 242, 111,  59, 183, 236, 214, 251,   7, 142,  90, 179,  80, 163, 159,  26, 143, 108, 109, 229, 223, 220, 196, 21,  18],
  [ 21,  42, 109, 188,  91,  93, 248, 236, 125,  48, 151,  12, 178,  26, 118, 135,  77,  84, 179, 208, 114, 224,  99, 246,  68,  21, 69,  39],
  [253,  66,  78,  55,  39, 107, 248,  90, 124, 107,  51,  92, 150, 234,  91, 177, 146,  80,   8, 179, 148, 229, 233,  59, 164, 199, 252,  43],
  [ 79,  60,   5,  70,  37, 218,  19,   9,  90,  74, 198, 129,  61, 160, 206,  11,  37, 171,  44, 241, 228, 190, 232,  99,   7, 100, 83, 225],
  [211,  38,  52, 167, 206, 139, 215, 209, 202, 102, 122,  77,  86, 117, 134,  22, 176,  94,  22, 201,   6,  73, 156, 226,  36,   0, 50, 119],
  [159,  24, 197, 215,  16, 243, 177,  13, 108, 211,   6,  97,  75, 214, 121,  92, 154, 109, 213, 163, 123,  20, 190, 174,  89,   6, 136, 164],
  [183, 136, 245, 175, 233,  62, 141, 117, 150,  74, 182, 175,  36, 230,  93, 109, 212,  43,  10,  75, 234, 124,  70, 244, 161,  76, 241, 223],
  [150,   7, 184,  20, 133,  22, 112, 212,  48,  30, 156, 113, 127, 207, 219, 173, 223, 127, 202, 172,  39,  98, 134, 124, 130,  34, 210, 101],
  [101,  77,  87,  37, 152, 112,  34, 106,  30,  23,  79, 214, 245, 152, 129, 243, 109, 213, 170, 190, 220,  25,  76, 205, 135, 227, 225, 165],
  [108, 184, 172, 121,   8,  83, 106, 116, 236,  55,  73, 204,  50,  40, 124, 153, 225, 157,  13,  28, 105,  62, 242, 214,  66, 159, 137,  67],
  [ 14,  75,  26,  47,  74, 205,  46, 219,  27,  18,  79,  28,  49, 224,  85, 214, 180, 106, 183,  87,  18,  64,   7,  61, 125,  87,  38,  98],
  [122, 146,   4,  72, 150, 249,  77,  90,   6, 132, 134, 151, 164,  29,  94, 188, 251, 177,   0, 206, 193, 182, 231,  43,  32,  32,  80, 147],
  [ 26,  39,  76,  12,  35,  81, 103, 233, 204, 136,  82,  28,   5,  68, 229, 197,  52, 215, 224, 117, 101,   4, 154,   4, 205,  50, 251, 114],
  [ 68, 176,  23, 246,  11,  57,  62,  25,  38,  17, 136, 106, 113, 140, 254,  43, 231, 150,  12, 114,  77,   8, 214, 187,  92,  66, 195,  70],
  [ 20, 241, 148, 151,  37,   4,  14, 231, 225,  53, 232, 240, 223,  59, 234, 134, 247, 242, 212,  63, 201,  38,  63, 200, 128, 139, 167, 173],
  [ 80, 244,  33, 111, 143, 127, 168, 237, 189,  63, 126, 181,  92,  91,  14, 211,  21,  26, 253, 109, 174, 100, 138, 138, 221, 204, 29, 230],
  [ 81, 174, 217,  93,  65, 134,   7,  35, 176, 122, 226,  23, 223,  28, 202,   5,  54, 205, 169,  14,  88, 178,  84, 198,  96, 201, 230, 193],
  [215, 168, 125,  92,  70, 151, 183, 210,  36,  32,  19,  51,  42,  64,  19, 146, 183, 246,   0, 184, 236,   7, 226, 118, 113, 241,  85,  89],
  [ 31, 158, 210,  16, 199,  58, 224,   7, 203,  86, 103,  45,  28,  54,  92, 204, 243, 117,  75, 208, 248, 223,  87, 250,  14,  43, 102,  66],
  [ 13, 236, 138,  67, 236, 109, 113,  46, 115,  19, 214, 154, 199, 248,  55, 172, 214, 249, 125, 154, 139, 141, 188,  78, 107, 200, 196,  16],
  [ 65, 150, 158, 254, 114, 177, 120,  15,  65,  58,  79, 171, 118,  32, 250,  81,  27,  85, 128, 146, 144, 234, 139,  26,   6,  68, 133, 205],
  [123,  68, 216,  34, 139,  34,  34, 175, 213,  72,  76,  19,  32, 138, 132, 111, 242, 249, 177,  89,  61,  72, 252,  79,  20, 171, 174, 177] ]
```

Image - Resizing

720 x 960

33 %

216 x 288

```
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  0   0   0   0   1  12   0  11  39 137  37   0 152 147  84   0   0   0
  0   0   1   0   0   0  41 160 250 255 235 162 255 238 206  11  13   0
  0   0   0  16   9   9 150 251  45  21 184 159 154 255 233  40   0   0
 10   0   0   0   0   0 145 146   3  10   0  11 124 253 255 107   0   0
  0   0   3   0   4  15 236 216   0   0  38 109 247 240 169   0  11   0
  1   0   2   0   0   0 253 253  23  62 224 241 255 164   0   5   0   0
  6   0   0   4   0   3 252 250 228 255 255 234 112  28   0   2  17   0
  0   2   1   4   0  21 255 253 251 255 172  31   8   0   1   0   0   0
  0   0   4   0 163 225 251 255 229 120   0   0   0   0   0  11   0   0
  0   0  21 162 255 255 254 255 126   6   0  10  14   6   0   0   9   0
  3  79 242 255 141  66 255 245 189   7   8   0   0   5   0   0   0   0
 26 221 237  98   0  67 251 255 144   0   8   0   0   7   0   0  11   0
125 255 141   0  87 244 255 208   3   0   0  13   0   1   0   1   0   0
145 248 228 116 235 255 141  34   0  11   0   1   0   0   0   1   3   0
 85 237 253 246 255 210  21   1   0   1   0   0   6   2   4   0   0   0
  6  23 112 157 114  32   0   0   0   0   2   0   8   0   7   0   0   0
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
```

Pixel Values (grayscale)

Image Augmentation

- Quite often there might not many images in the dataset
- Doing augmentation helps in adding new images to the data - add variety
- Some of augmentations that can be done on images:
  - Rotate the image by certain angle
  - Width or Height shifting
  - Change image brightness
  - Sheer or Zoom images
  - Convert to grayscale

Image Augmentation

- Open file '**CodeSamples/ImageAugmentation**' using Jupyter
- Example of doing image augmentation such as rotating images, changing width and height
- Converting to Gray Scale

Pixel Normalization: Min - Max

$$\frac{X - min}{\max - min}$$

Pixel values (gray scale) are between 0 (min) and 255 (max)

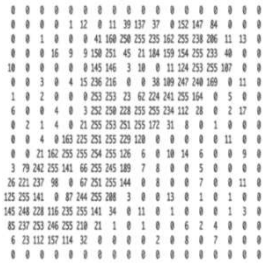
Want values between 0 and 1

$$\frac{X}{255}$$

Hand-Written Digits Recognition

Digit Recognition

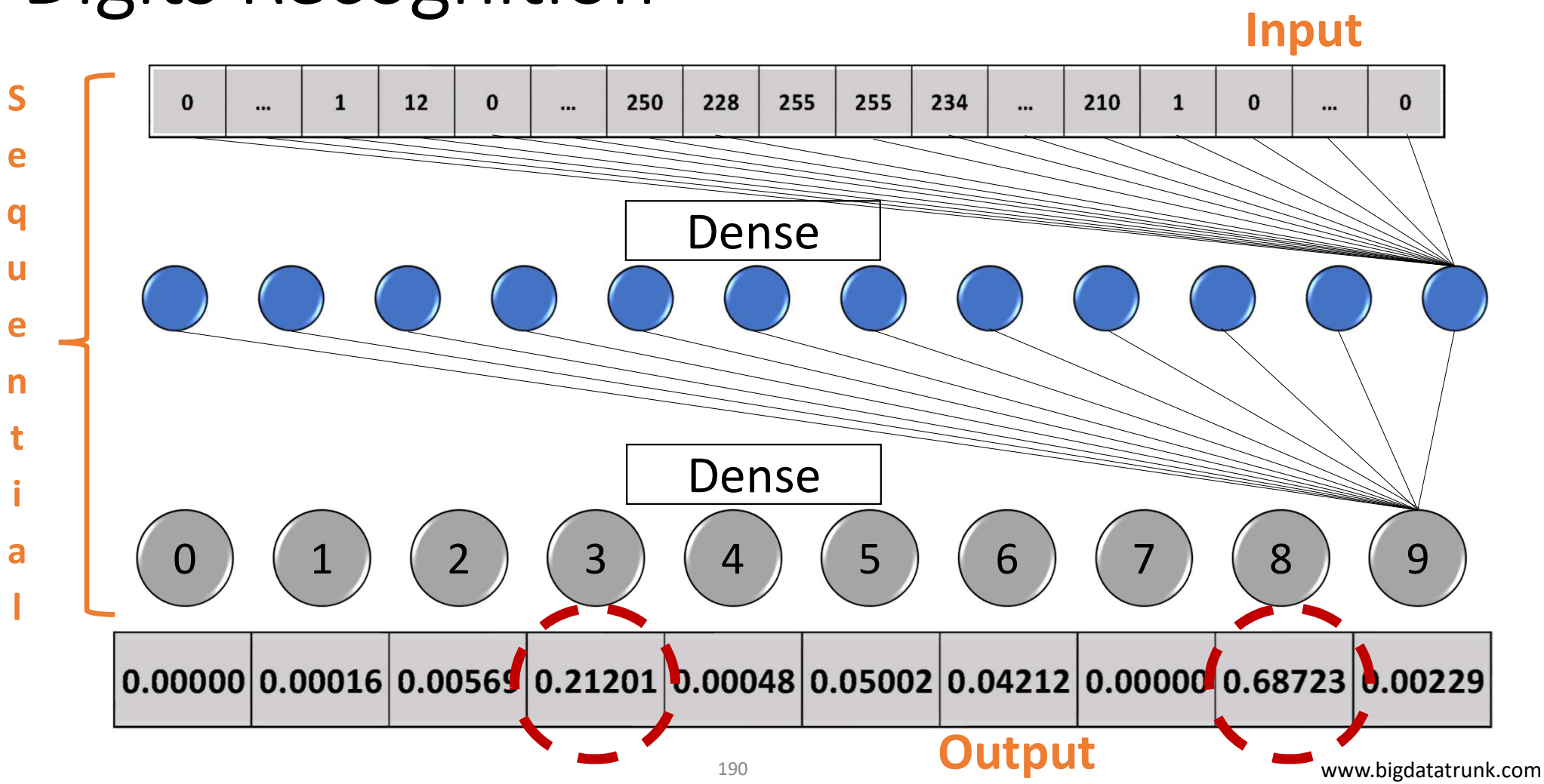
28 x 28



Flatten

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 12, 0, 11, 39, 137, 37, 0, 152, 147, 84, 0, 0, 0, 0,
0, 1, 0, 0, 0, 41, 160, 250, 255, 235, 162, 255, 238, 206, 11, 13, 0, 0, 0, 0, 16, 9, 9, 150, 251, 45, 21, 184, 159, 154, 2
55, 233, 40, 0, 0, 10, 0, 0, 0, 0, 0, 145, 146, 3, 10, 0, 11, 124, 253, 255, 107, 0, 0, 0, 0, 3, 0, 4, 15, 236, 216, 0, 0,
38, 109, 247, 240, 169, 0, 11, 0, 1, 0, 2, 0, 0, 0, 253, 253, 23, 62, 224, 241, 255, 164, 0, 5, 0, 0, 6, 0, 0, 4, 0, 3, 252
, 250, 228, 255, 255, 234, 112, 28, 0, 2, 17, 0, 0, 2, 1, 4, 0, 21, 255, 253, 251, 255, 172, 31, 8, 0, 1, 0, 0, 0, 0, 0, 4,
 0, 163, 225, 251, 255, 229, 120, 0, 0, 0, 0, 0, 11, 0, 0, 0, 0, 21, 162, 255, 255, 254, 255, 126, 6, 0, 10, 14, 6, 0, 0, 9
, 0, 3, 79, 242, 255, 141, 66, 255, 245, 189, 7, 8, 0, 0, 5, 0, 0, 0, 0, 26, 221, 237, 98, 0, 67, 251, 255, 144, 0, 8, 0, 0
, 7, 0, 0, 11, 0, 125, 255, 141, 0, 87, 244, 255, 208, 3, 0, 0, 13, 0, 1, 0, 1, 0, 0, 145, 248, 228, 116, 235, 255, 141, 34
, 0, 11, 0, 1, 0, 0, 0, 1, 3, 0, 85, 237, 253, 246, 255, 210, 21, 1, 0, 1, 0, 0, 6, 2, 4, 0, 0, 0, 6, 23, 112, 157, 114, 32
, 0, 0, 0, 0, 2, 0, 8, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Digits Recognition



www.bigdatatrunk.com

Keras and TensorFlow

```
model = keras.models.Sequential()
model.add(keras.layers.Flatten(input_shape=[28, 28]))
model.add(keras.layers.Dense(128, activation="relu"))
model.add(keras.layers.Dense(10, activation="softmax"))
```

Train model using the training data

```
model.fit(trainingImages, trainingLabels, epoch = 10)
```

Evaluate the performance of the trained model

```
model.evaluate(testImages, testLabels)
```

Make predictions on the test data

```
model.predict(testImages)
```

Hyper-Parameters

Hidden Layers

- Choose 1 or 2 layers

Neurons Per Hidden Layer

- Make first layer large
- Or make all layers with same number of neurons

Learning Rate

- Start at low rate (0.0001)
- Gradually increase

Activation

- Hidden layers: ReLU
- Multi-Class: Softmax
- Binary Class: Sigmoid
- Regression: None

Hyper-Parameters

	Classification	Regression
Optimizer	‘adam’: keras.optimizers.Adam(lr = 0.001) ‘rmsprop’: keras.optimizers.RMSProp(lr = 0.001)	‘sgd’: keras.optimizers.SGD(lr = 0.001) ‘rmsprop’: keras.optimizers.RMSProp(lr = 0.001)
Loss	Binary: ‘binary_crossentropy’ Multi-Class: ‘categorical_crossentropy’ Discrete: ‘sparse_categorical_crossentropy’	‘mse’
Metrics	‘accuracy’	‘mse’, ‘mae’

Neural Networks

- Open file '**CodeSamples/NeuralNetworks**' using Jupyter
- Classify handwritten digits using MNIST Digits Data
- Recognize digits between 0 and 9

Challenges with Image Handling

Image 28 x28

Input Layer: 28 * 28 = 784 Features

Image 1000 x 1000

Input Layer: 1000 * 1000 = 1 million Features

Challenges with Image Handling

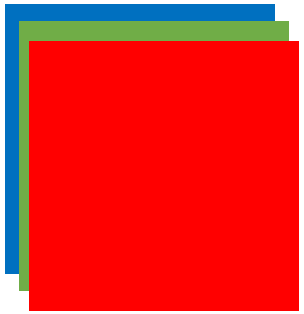


Image 1000 x 1000

Input Layer: 3 * 1000 * 1000 = 3 million Features

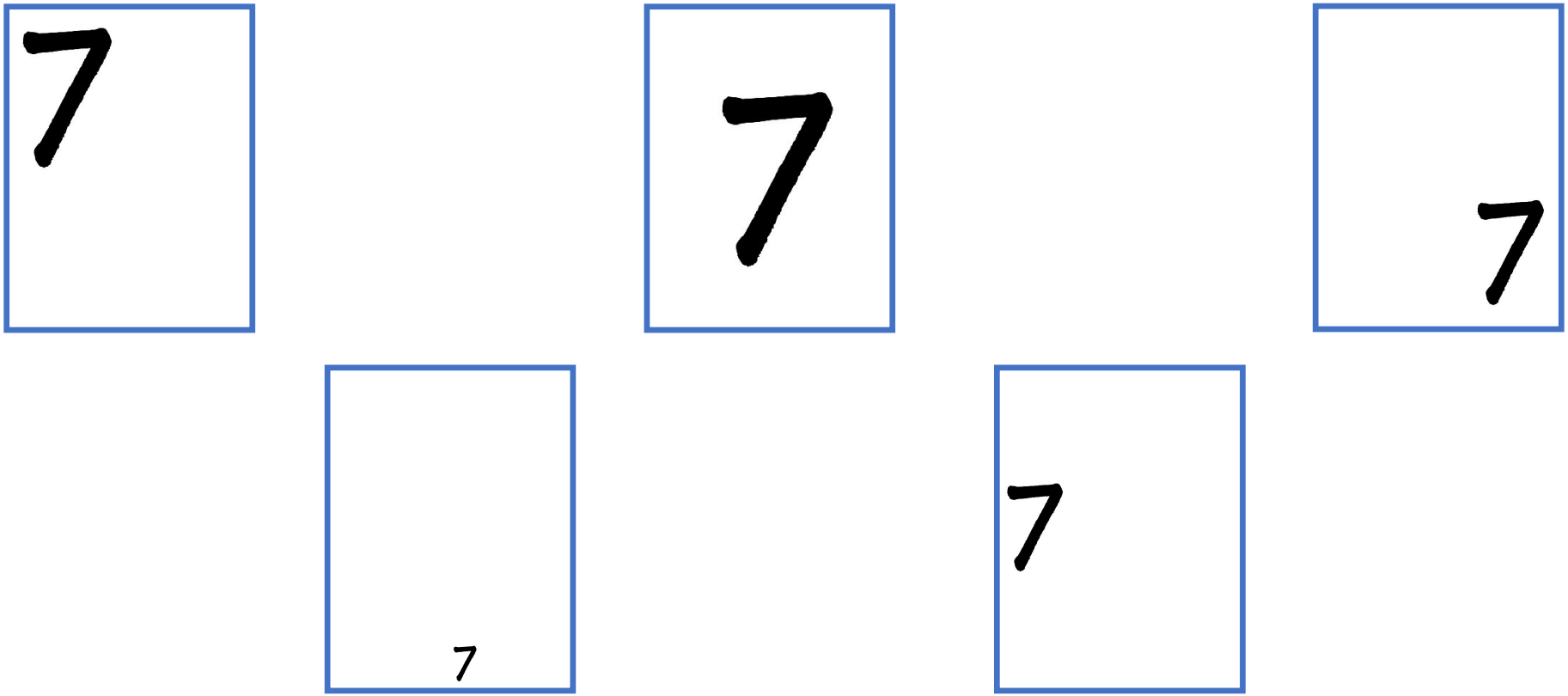
THAT IS HUGE - COMPUTATIONALLY

1 Hidden layer and 1000 Neurons in it:

3 million * 1000 = **3 billion weights**

Challenges with Image Processing

How can neural network recognize the digits?



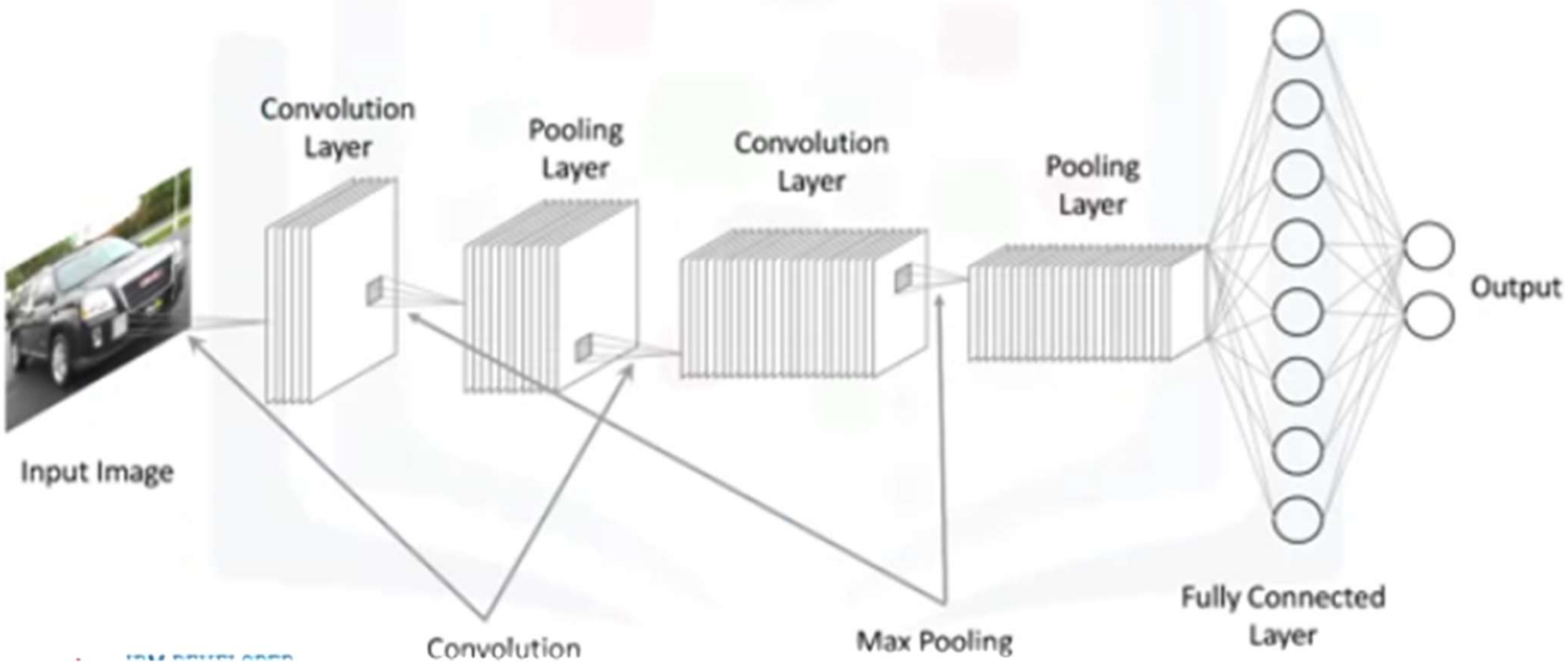
Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNN)

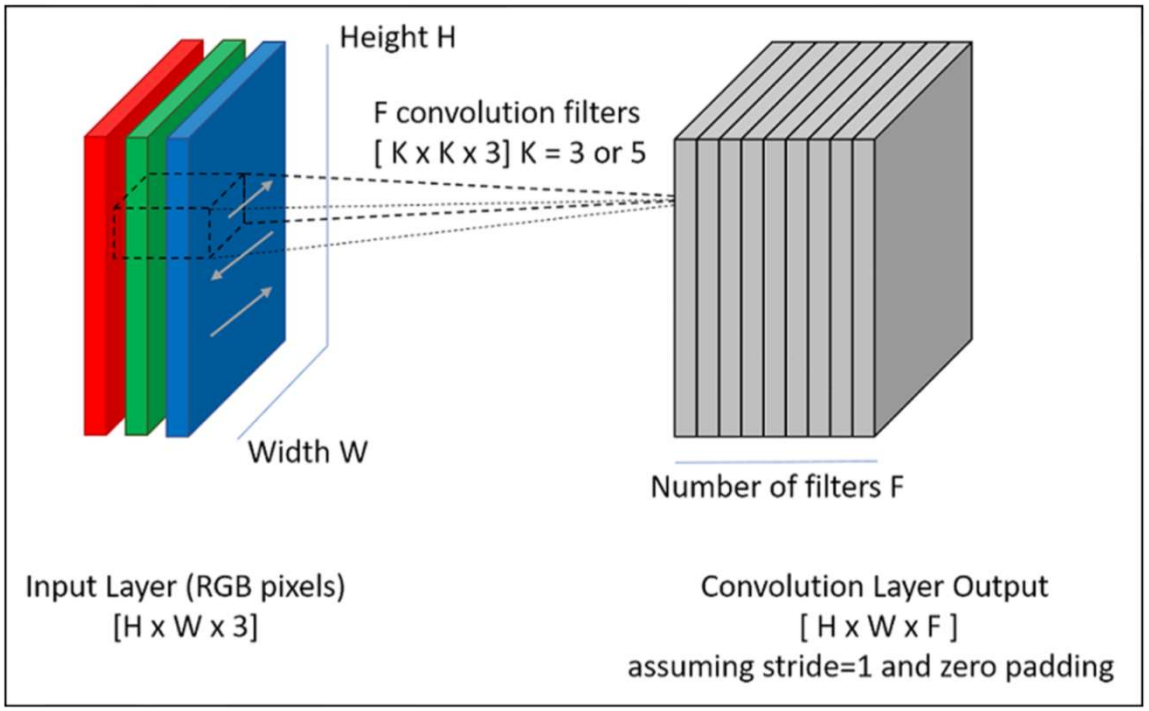
199

- Type of Supervised Deep Learning technique
- Takes images as inputs
- Provides methods to make the training of images efficient
- Helps make the forward propagation step efficient
- Reduces number of features in the images
- Useful in image recognition, object detection and other computer vision applications

CNN Architecture



Filter Maps


Height H
F convolution filters
[K x K x 3] K = 3 or 5
Width W
Number of filters F
Input Layer (RGB pixels)
[H x W x 3]
Convolution Layer Output
[H x W x F]
assuming stride=1 and zero padding

Convolutional Layer – Feature Maps

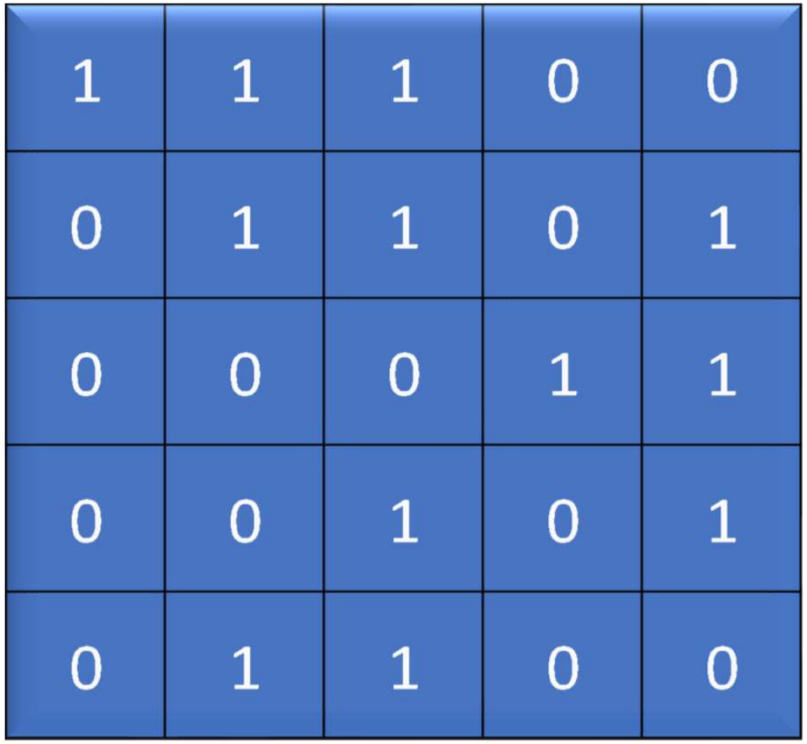
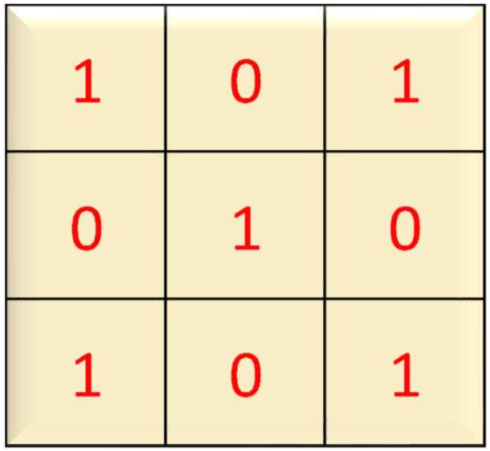
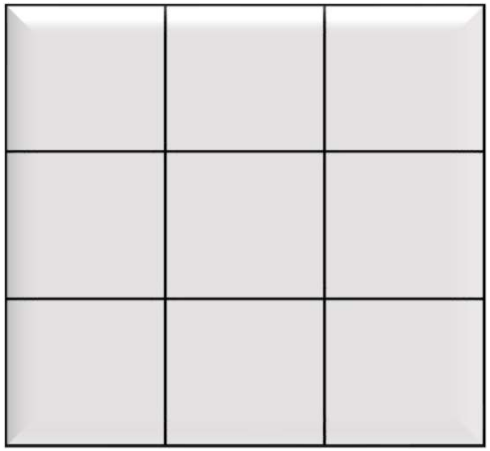


Image (5 x 5)

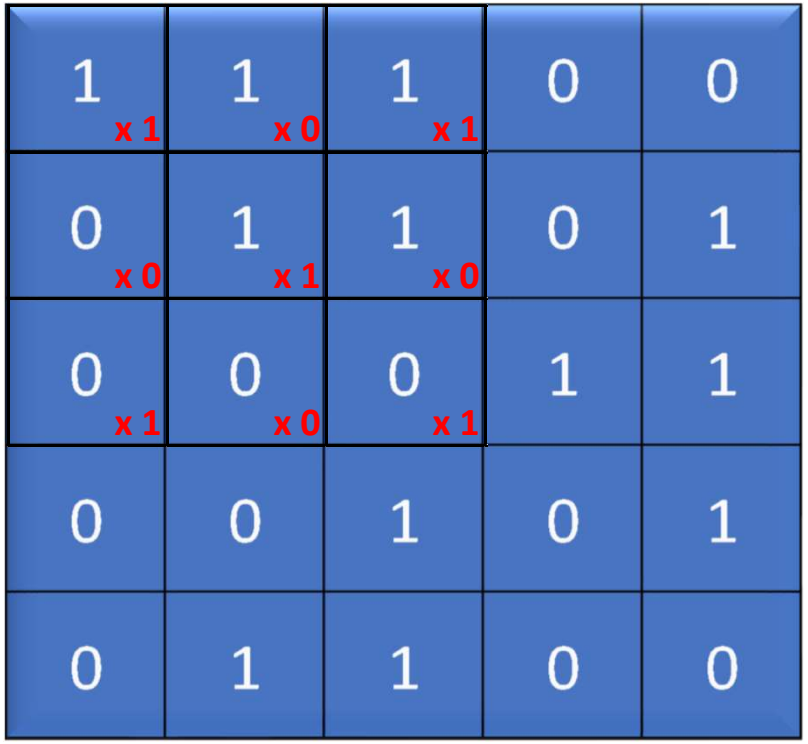


Filter (3 x 3)

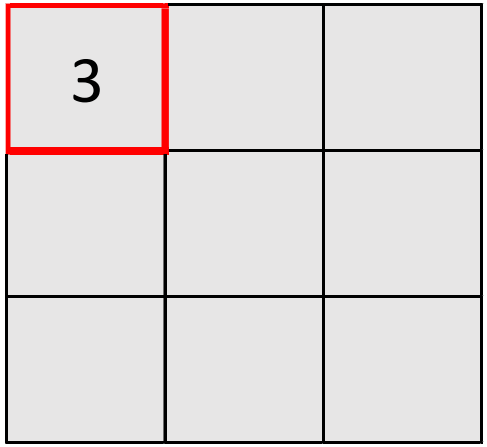


Output
Feature Map

Convolutional Layer – Feature Maps

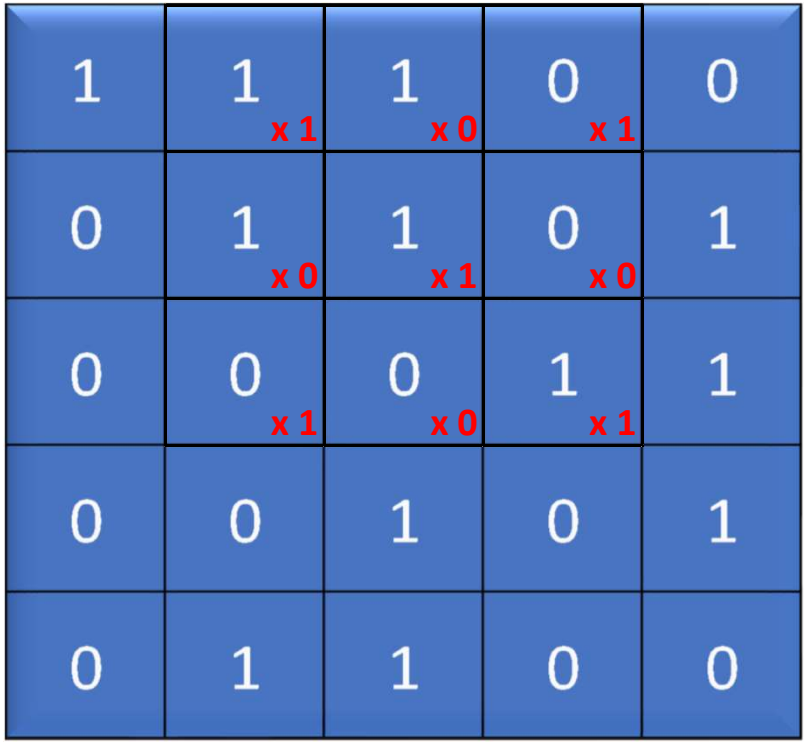


Image

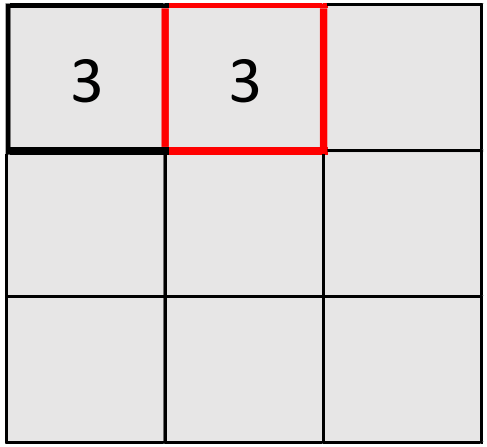


Output
Feature Map

Convolutional Layer- Feature Maps

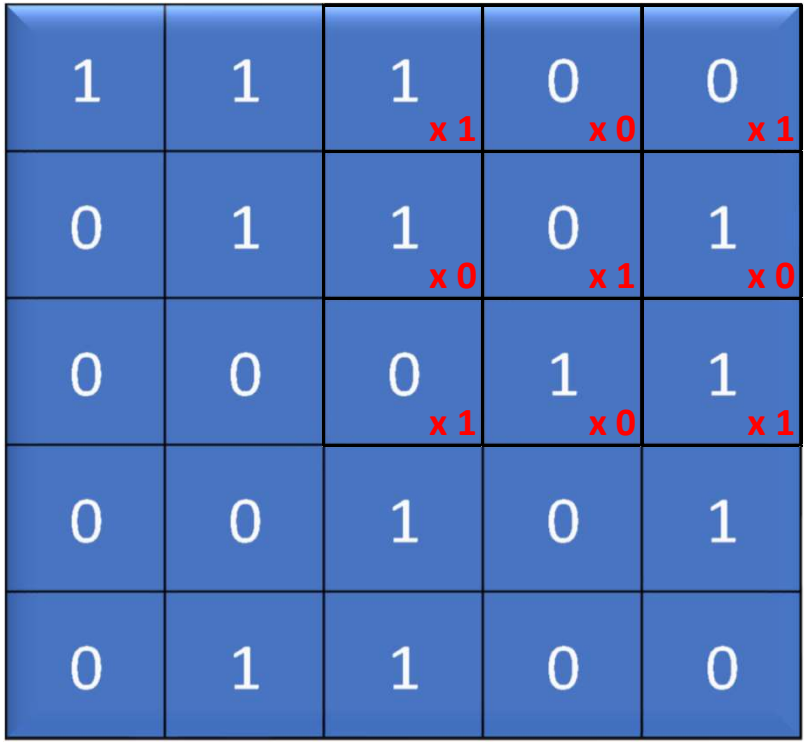


Image

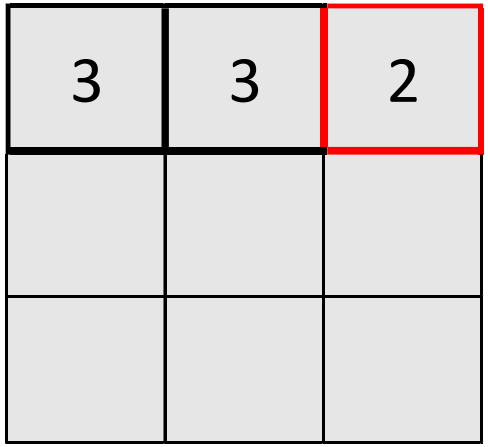


Output
Feature Map

Convolutional Layer – Feature Maps

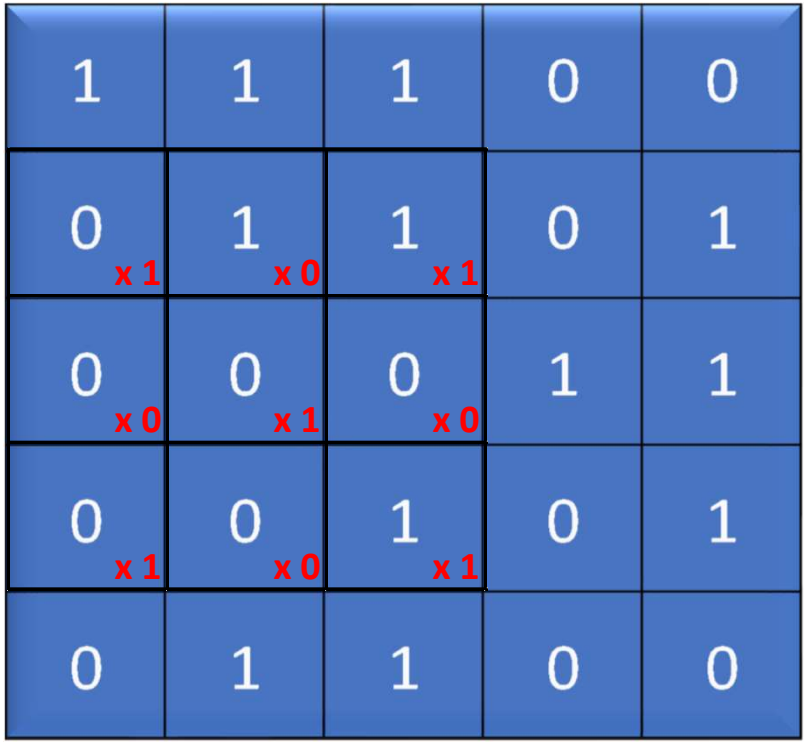


Image

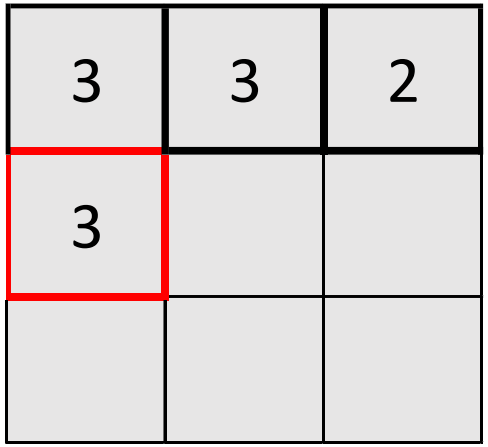


Output
Feature Map

Convolutional Layer – Feature Maps

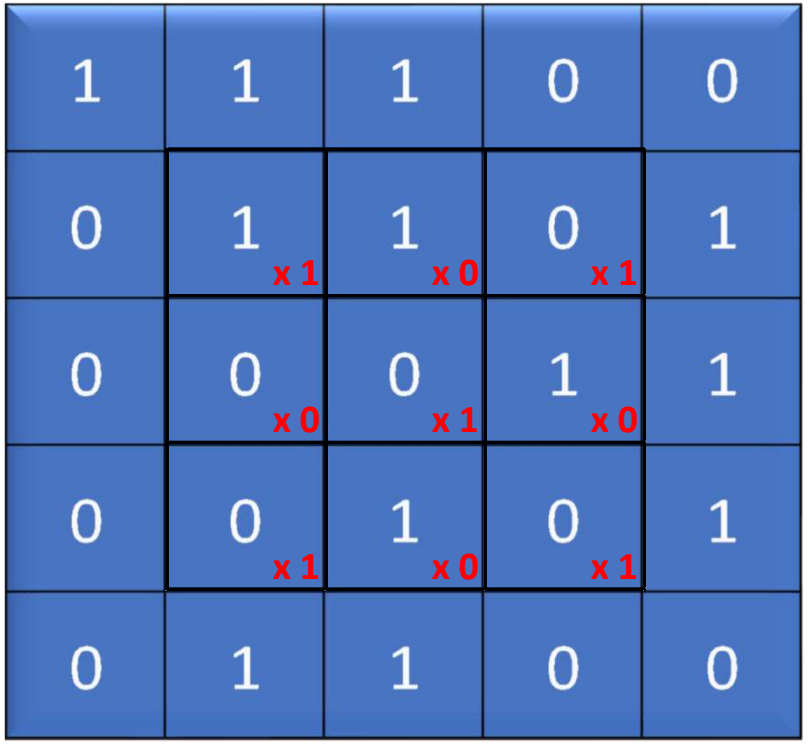


Image

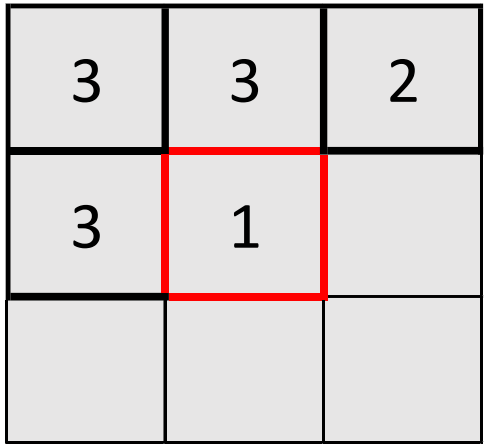


Output
Feature Map

Convolutional Layer – Feature Maps

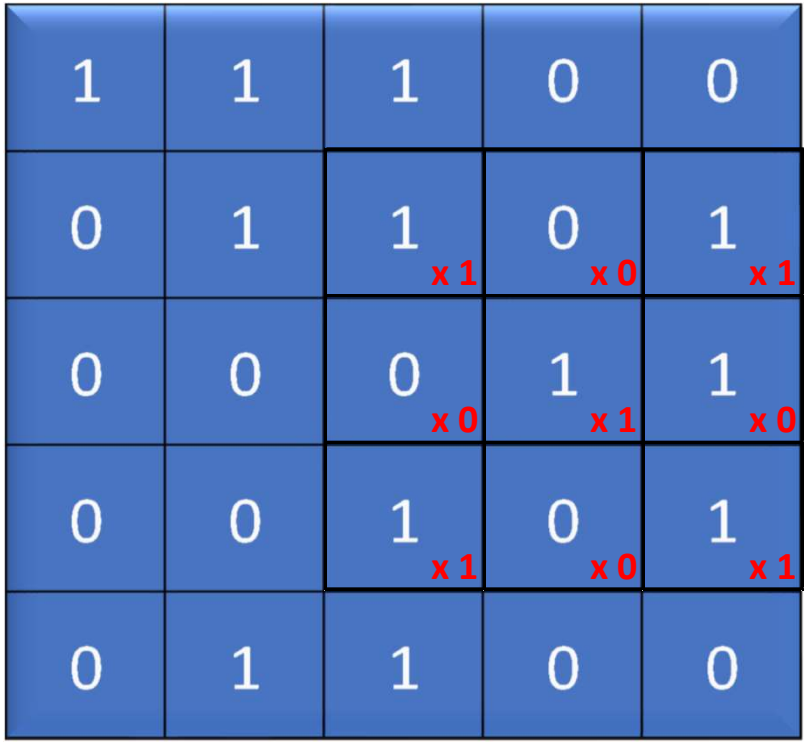


Image

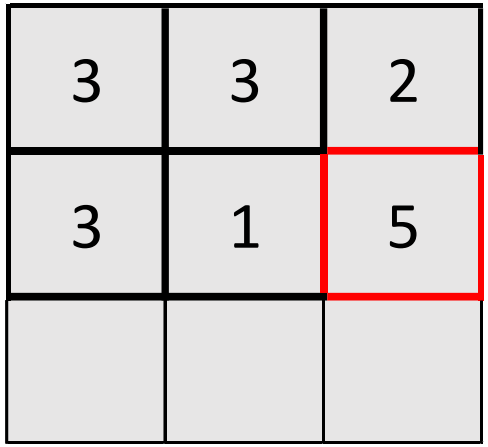


Output
Feature Map

Convolutional Layer – Feature Maps

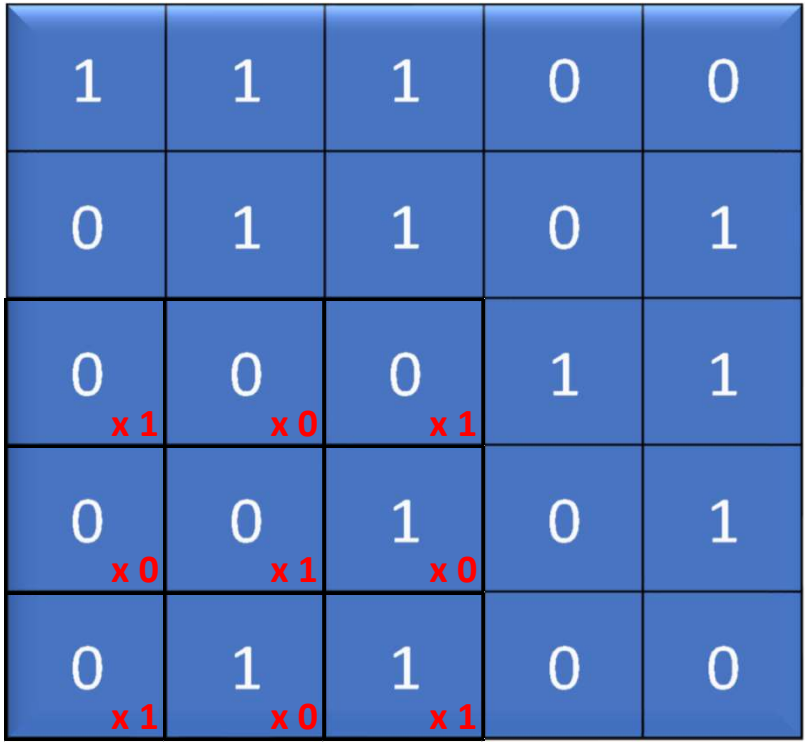


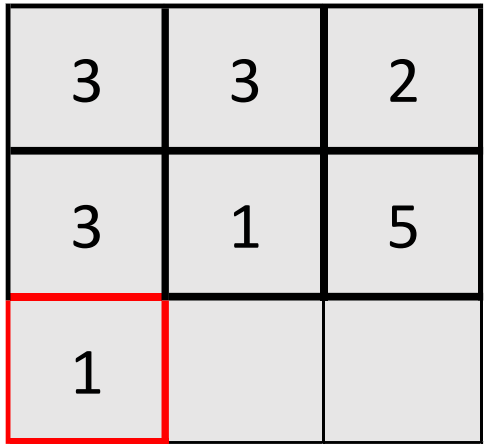
Image



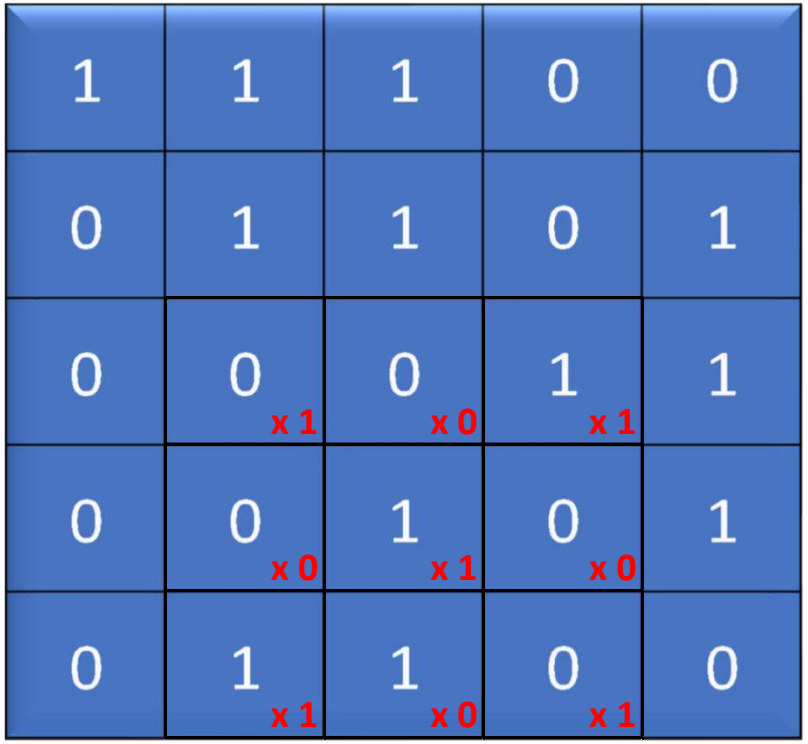
Output
Feature Map

Convolutional Layer – Feature Maps

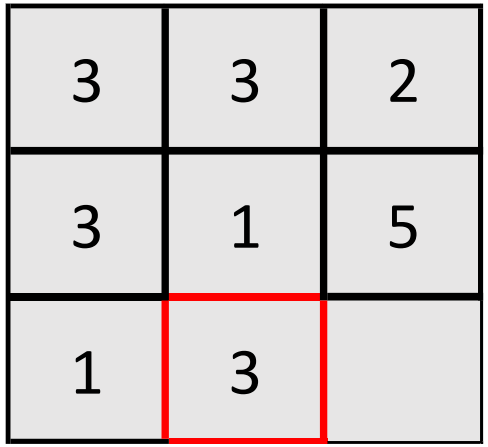




Convolutional Layer – Feature Maps

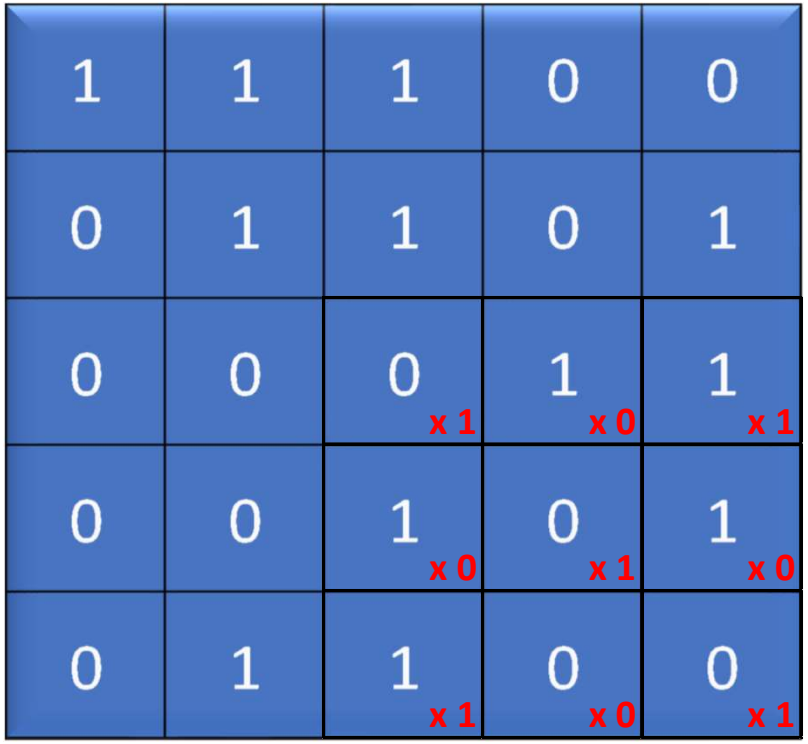


Image

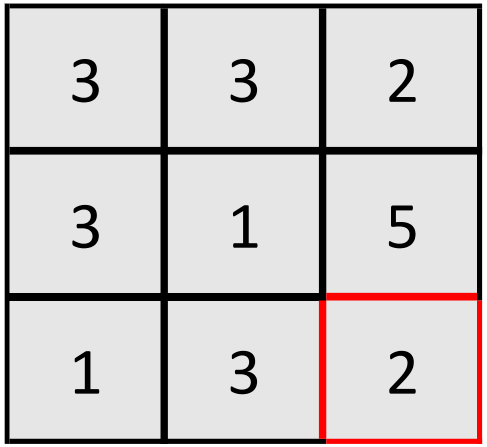


Output
Feature Map

Convolutional Layer – Feature Maps



Image



Output
Feature Map

Convolution

Filters

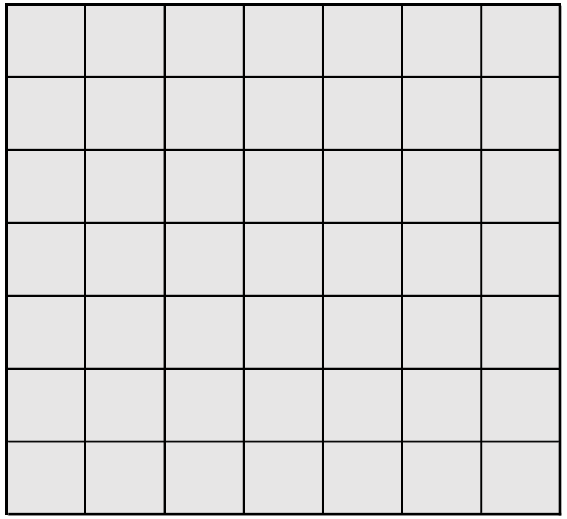
1	-1	-1
-1	1	-1
-1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1

Image

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Feature Map



Convolution

Filters

1	-1	-1
-1	1	-1
-1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1

Image

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Feature Map

7						

Convolution

Filters

1	-1	-1
-1	1	-1
-1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1

Image

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Feature Map

7						
	9					

Convolution

Filters

1	-1	-1
-1	1	-1
-1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1

Image

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Feature Map

7						
	9					
		9				

Convolution

Filters

1	-1	-1
-1	1	-1
-1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1

Image

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Feature Map

7						
	9					
		9				
			5			

Convolution

Filters

1	-1	-1
-1	1	-1
-1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1

Image

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Feature Map

7						
	9					
		9				
			5			
				9		

Convolution

Filters

1	-1	-1
-1	1	-1
-1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1

Image

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Feature Map

7						
	9					
		9				
			5			
				9		
					9	

Convolution

Filters

1	-1	-1
-1	1	-1
-1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1

Image

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Feature Map

7						
	9					
		9				
			5			
				9		
					9	
						7

Convolution

Filters

1	-1	-1
-1	1	-1
-1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1

Image

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Feature Map

7	-1	1	3	5	-1	3
-1	9	-1	3	-1	1	-1
1	-1	9	-3	1	-1	5
3	3	-3	5	-3	3	3
5	-1	1	-3	9	-1	1
-1	1	-1	3	-1	9	-1
3	-1	3	1	1	-1	7

Convolution

Filters

1	-1	-1
-1	1	-1
-1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1

Image

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Feature Map

7	-1	1	3	5	-1	3
-1	9	-1	3	-1	1	-1
1	-1	9	-3	1	-1	5
3	3	-3	5	-3	3	3
5	-1	1	-3	9	-1	1
-1	1	-1	3	-1	9	-1
3	-1	3	1	1	-1	7

www.bigdatatrunk.com

Convolution

Filters

1	-1	-1
-1	1	-1
-1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1

Image

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Feature Map

						7

7	-1	1	3	5	-1	3
-1	9	-1	3	-1	1	-1
1	-1	9	-3	1	-1	5
3	3	-3	5	-3	3	3
5	-1	1	-3	9	-1	1
-1	1	-1	3	-1	9	-1
3	-1	3	1	1	-1	7

www.bigdatatrunk.com

Convolution

Filters

1	-1	-1
-1	1	-1
-1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1

Image

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Feature Map

						7
					9	

7	-1	1	3	5	-1	3
-1	9	-1	3	-1	1	-1
1	-1	9	-3	1	-1	5
3	3	-3	5	-3	3	3
5	-1	1	-3	9	-1	1
-1	1	-1	3	-1	9	-1
3	-1	3	1	1	-1	7

www.bigdatatrunk.com

Convolution

Filters

1	-1	-1
-1	1	-1
-1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1

Image

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Feature Map

						7
					9	
				9		
			5			
		9				
	9					
7						

7	-1	1	3	5	-1	3
-1	9	-1	3	-1	1	-1
1	-1	9	-3	1	-1	5
3	3	-3	5	-3	3	3
5	-1	1	-3	9	-1	1
-1	1	-1	3	-1	9	-1
3	-1	3	1	1	-1	7

www.bigdatatrunk.com

Convolution

Filters

1	-1	-1
-1	1	-1
-1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1

Image

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Feature Map

3	-1	5	3	1	-1	7
-1	1	-1	3	-1	9	-1
5	-1	1	-3	9	-1	1
3	3	-3	5	-3	3	3
1	-1	9	-3	1	-1	5
-1	9	-1	3	-1	1	-1
7	-1	1	3	1	-1	3

7	-1	1	3	5	-1	3
-1	9	-1	3	-1	1	-1
1	-1	9	-3	1	-1	5
3	3	-3	5	-3	3	3
5	-1	1	-3	9	-1	1
-1	1	-1	3	-1	9	-1
3	-1	3	1	1	-1	7

www.bigdatatrunk.com

Pooling

226

- Pooling layer is used to reduce the spatial dimensions of the data propagating through the network
- There are 2 types of pooling options:
  1. Max Pooling – keep the maximum value in the filter matrix
  2. Average Pooling – take an average value in the filter matrix
- Max pooling provides mechanism that helps in recognizing objects in the image even if the object does not resemble the original object

Pooling

7	-1	1	3	5	-1	3
-1	9	-1	3	-1	1	-1
1	-1	9	-3	1	-1	5
3	3	-3	5	-3	3	3
5	-1	1	-3	9	-1	1
-1	1	-1	3	-1	9	-1
3	-1	3	1	1	-1	7

3	-1	5	3	1	-1	7
-1	1	-1	3	-1	9	-1
5	-1	1	-3	9	-1	1
3	3	-3	5	-3	3	3
1	-1	9	-3	1	-1	5
-1	9	-1	3	-1	1	-1
7	-1	1	3	1	-1	3

9		

Pooling

7	-1	1	3	5	-1	3
-1	9	-1	3	-1	1	-1
1	-1	9	-3	1	-1	5
3	3	-3	5	-3	3	3
5	-1	1	-3	9	-1	1
-1	1	-1	3	-1	9	-1
3	-1	3	1	1	-1	7

3	-1	5	3	1	-1	7
-1	1	-1	3	-1	9	-1
5	-1	1	-3	9	-1	1
3	3	-3	5	-3	3	3
1	-1	9	-3	1	-1	5
-1	9	-1	3	-1	1	-1
7	-1	1	3	1	-1	3

9	3	

Pooling

7	-1	1	3	5	-1	3
-1	9	-1	3	-1	1	-1
1	-1	9	-3	1	-1	5
3	3	-3	5	-3	3	3
5	-1	1	-3	9	-1	1
-1	1	-1	3	-1	9	-1
3	-1	3	1	1	-1	7

3	-1	5	3	1	-1	7
-1	1	-1	3	-1	9	-1
5	-1	1	-3	9	-1	1
3	3	-3	5	-3	3	3
1	-1	9	-3	1	-1	5
-1	9	-1	3	-1	1	-1
7	-1	1	3	1	-1	3

9	3	5

Pooling

7	-1	1	3	5	-1	3
-1	9	-1	3	-1	1	-1
1	-1	9	-3	1	-1	5
3	3	-3	5	-3	3	3
5	-1	1	-3	9	-1	1
-1	1	-1	3	-1	9	-1
3	-1	3	1	1	-1	7

3	-1	5	3	1	-1	7
-1	1	-1	3	-1	9	-1
5	-1	1	-3	9	-1	1
3	3	-3	5	-3	3	3
1	-1	9	-3	1	-1	5
-1	9	-1	3	-1	1	-1
7	-1	1	3	1	-1	3

9	3	5
3		

Pooling

7	-1	1	3	5	-1	3
-1	9	-1	3	-1	1	-1
1	-1	9	-3	1	-1	5
3	3	-3	5	-3	3	3
5	-1	1	-3	9	-1	1
-1	1	-1	3	-1	9	-1
3	-1	3	1	1	-1	7

3	-1	5	3	1	-1	7
-1	1	-1	3	-1	9	-1
5	-1	1	-3	9	-1	1
3	3	-3	5	-3	3	3
1	-1	9	-3	1	-1	5
-1	9	-1	3	-1	1	-1
7	-1	1	3	1	-1	3

9	3	5
3	9	

Pooling

7	-1	1	3	5	-1	3
-1	9	-1	3	-1	1	-1
1	-1	9	-3	1	-1	5
3	3	-3	5	-3	3	3
5	-1	1	-3	9	-1	1
-1	1	-1	3	-1	9	-1
3	-1	3	1	1	-1	7

9	3	5
3	9	3

3	-1	5	3	1	-1	7
-1	1	-1	3	-1	9	-1
5	-1	1	-3	9	-1	1
3	3	-3	5	-3	3	3
1	-1	9	-3	1	-1	5
-1	9	-1	3	-1	1	-1
7	-1	1	3	1	-1	3

Pooling

7	-1	1	3	5	-1	3
-1	9	-1	3	-1	1	-1
1	-1	9	-3	1	-1	5
3	3	-3	5	-3	3	3
5	-1	1	-3	9	-1	1
-1	1	-1	3	-1	9	-1
3	-1	3	1	1	-1	7

3	-1	5	3	1	-1	7
-1	1	-1	3	-1	9	-1
5	-1	1	-3	9	-1	1
3	3	-3	5	-3	3	3
1	-1	9	-3	1	-1	5
-1	9	-1	3	-1	1	-1
7	-1	1	3	1	-1	3

9	3	5
3	9	3
5		

Pooling

7	-1	1	3	5	-1	3
-1	9	-1	3	-1	1	-1
1	-1	9	-3	1	-1	5
3	3	-3	5	-3	3	3
5	-1	1	-3	9	-1	1
-1	1	-1	3	-1	9	-1
3	-1	3	1	1	-1	7

3	-1	5	3	1	-1	7
-1	1	-1	3	-1	9	-1
5	-1	1	-3	9	-1	1
3	3	-3	5	-3	3	3
1	-1	9	-3	1	-1	5
-1	9	-1	3	-1	1	-1
7	-1	1	3	1	-1	3

9	3	5
3	9	3
5	3	

Pooling

7	-1	1	3	5	-1	3
-1	9	-1	3	-1	1	-1
1	-1	9	-3	1	-1	5
3	3	-3	5	-3	3	3
5	-1	1	-3	9	-1	1
-1	1	-1	3	-1	9	-1
3	-1	3	1	1	-1	7

3	-1	5	3	1	-1	7
-1	1	-1	3	-1	9	-1
5	-1	1	-3	9	-1	1
3	3	-3	5	-3	3	3
1	-1	9	-3	1	-1	5
-1	9	-1	3	-1	1	-1
7	-1	1	3	1	-1	3

9	3	5
3	9	3
5	3	9

Pooling

7	-1	1	3	5	-1	3
-1	9	-1	3	-1	1	-1
1	-1	9	-3	1	-1	5
3	3	-3	5	-3	3	3
5	-1	1	-3	9	-1	1
-1	1	-1	3	-1	9	-1
3	-1	3	1	1	-1	7

3	-1	5	3	1	-1	7
-1	1	-1	3	-1	9	-1
5	-1	1	-3	9	-1	1
3	3	-3	5	-3	3	3
1	-1	9	-3	1	-1	5
-1	9	-1	3	-1	1	-1
7	-1	1	3	1	-1	3

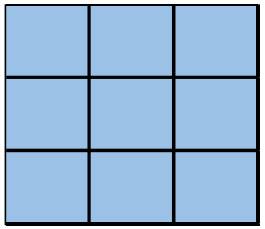
9	3	5
3	9	3
5	3	9

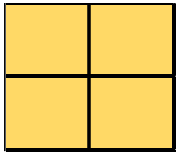
5	3	9
3	9	3
9	3	5

www.bigdatatrunk.com

Zero Padding

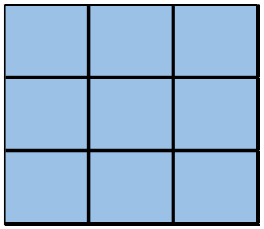
0.3	0.5	0.9	1.0
0.2	0.5	0.2	0.8
0.1	0.7	0.9	1.0
0.6	0.3	0.5	0.2

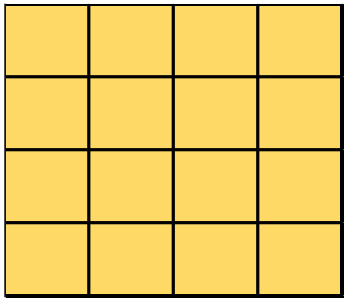




Input: 4 x 4
Filter: 3 x 3
Output: (In – Flt + 1) x (In – Flt + 1)
(4 – 3 + 1) x (4 – 3 + 1)
2 x 2

0	0	0	0	0	0
0	0.3	0.5	0.9	1.0	0
0	0.2	0.5	0.2	0.8	0
0	0.1	0.7	0.9	1.0	0
0	0.6	0.3	0.5	0.2	0
0	0	0	0	0	0





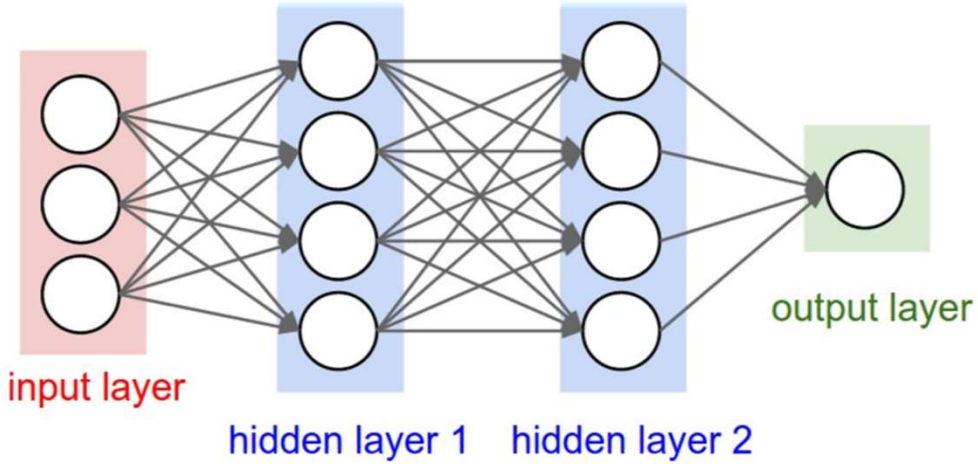
Input: 4 x 4
Filter: 3 x 3
Output: 4 x 4

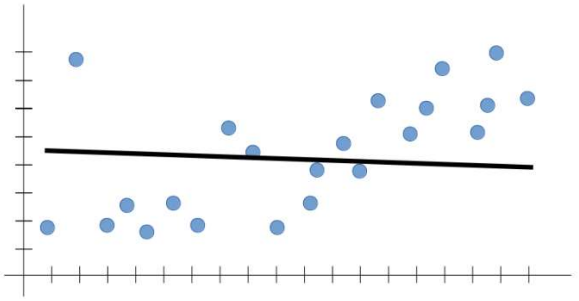
Zero Padding

- Zero padding allows to preserve the original input size
- Can be specified per convolutional layer – need one or not
- Adds a vector of zeros around the image – not mandatory that there is only one layer, can be 2-3 determined by the library
- Specifying zero padding in Keras Layer (e.g. Conv2D):
  - **padding = "valid"** – means apply no zero padding
  - **padding = "same"** – means apply padding to make the output size of the image same as the input

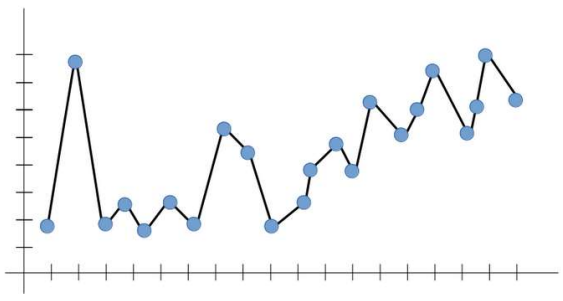
Fully Connected Layer

- Here output from the last Convolutional or Pooling layer are flattened
- Every node in the current layer is connected to every node in the next layer
- The last layer (output layer) consists on n nodes (depends on number of outcomes – in the digit recognition it is 10 representing digits 0 to 9)

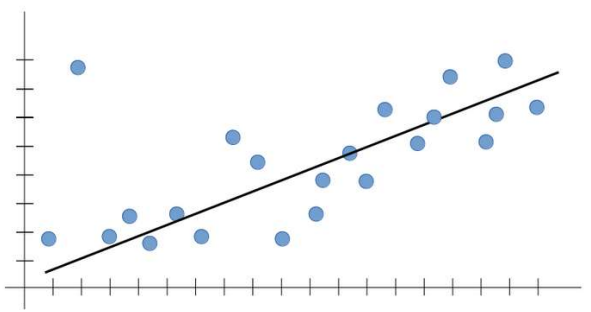




Under fitting (High Bias)



Over fitting (High Variance)



Bias-Variance Trade off

Under and Over Fitting

Convolutional Neural Networks

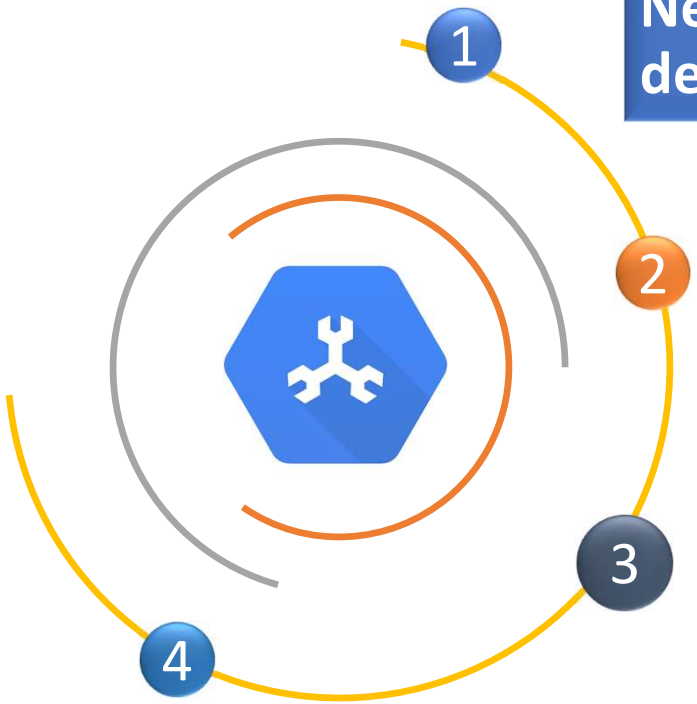
- Open file '**CodeSamples/CNN**' using Jupyter
- Classify handwritten digits using MNIST Digits Data
- Recognize digits between 0 and 9

Recurrent Neural Networks

Short Comings of Artificial Neural Networks

- Traditional Neural Networks assume that each data point is independent of other data points (inputs are analyzed in isolation)
- Does not handle sequential or “temporal” data
  - Sentences
  - Stock Prices trend
  - Time Series
  - Gene Sequences
- In sequential data the length of the input sequence can vary from sample to sample
- In sentences, need to understand the dependencies between words to get the full context of the sentence

Recurrent Neural Networks



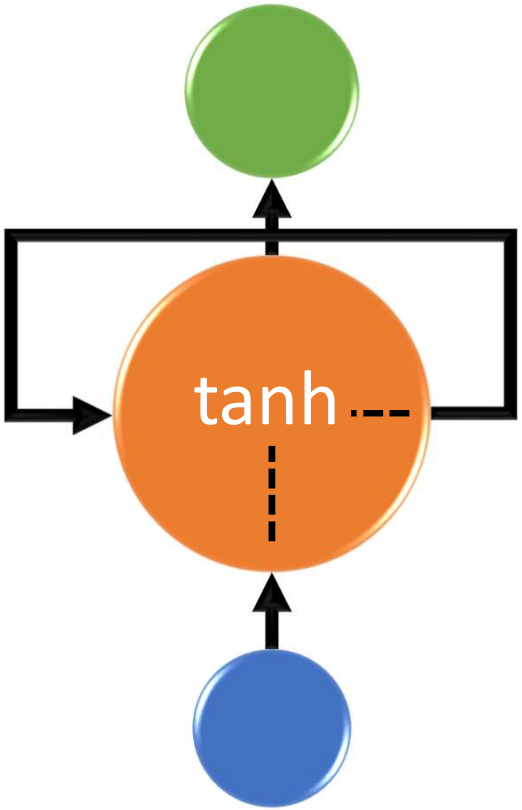
Neural Networks capture the temporal dependencies in data

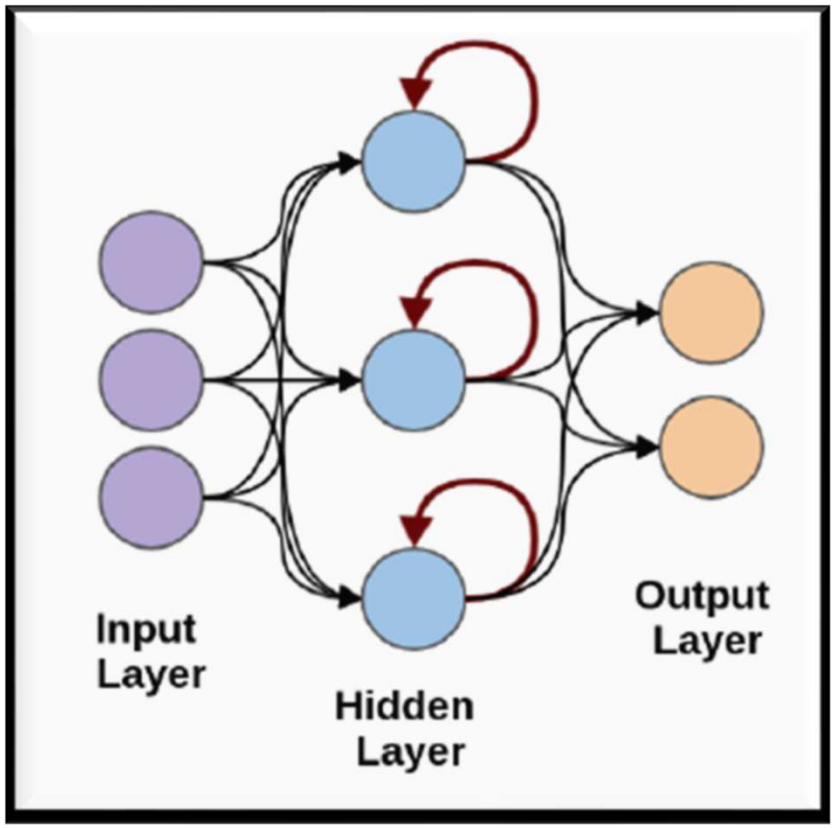
Each neuron in RNN uses its internal memory to maintain information about the previous input

Maintaining such state information is critical during language processing because in order to predict the next word, need to know which words came before it

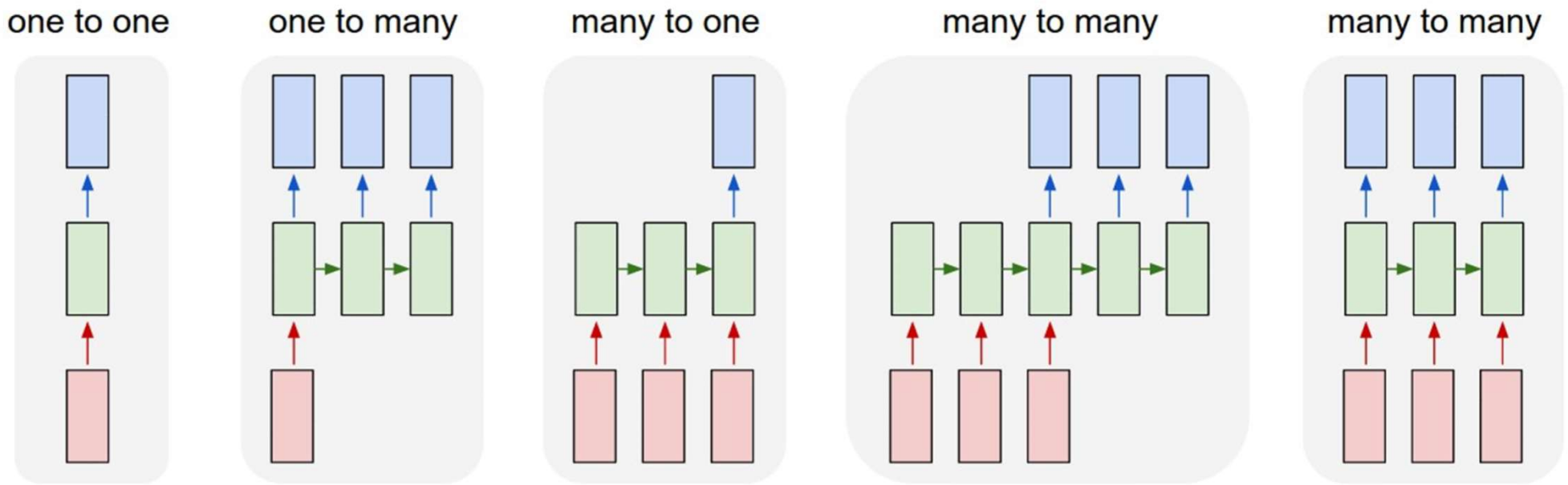
When reading a sentence, we pick up the context of the word based on the words that preceded it

Recurrent Neural Network





Types of RNNs



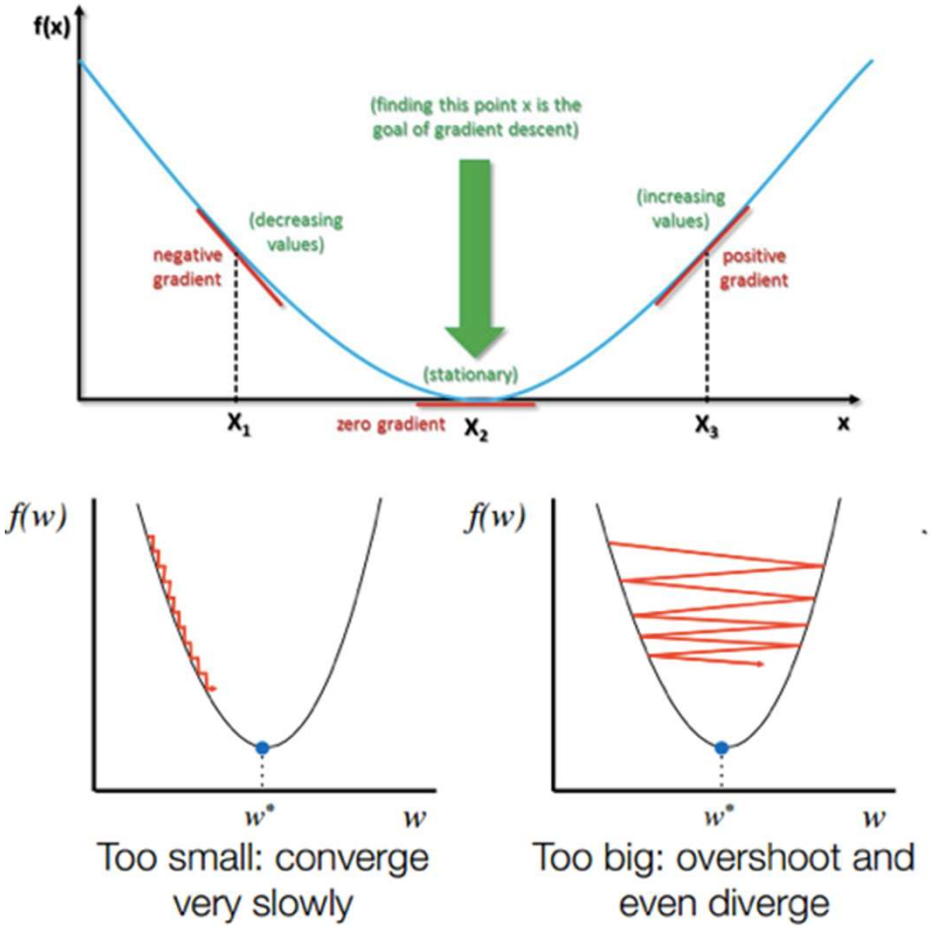
Gradient Problems

Gradient of the loss with respect to weights

- Calculated during the back-propagation phase of learning

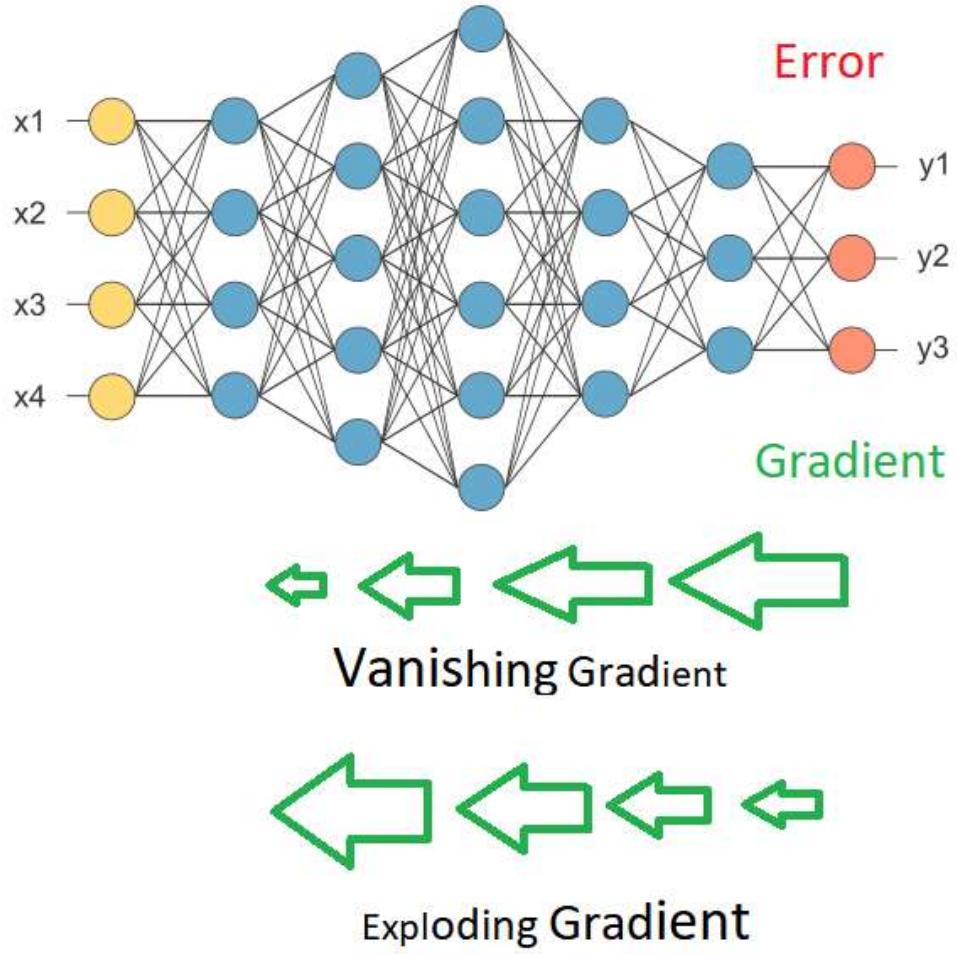
- Objective is to update the weights for each neuron in the network

Backward propagation issues

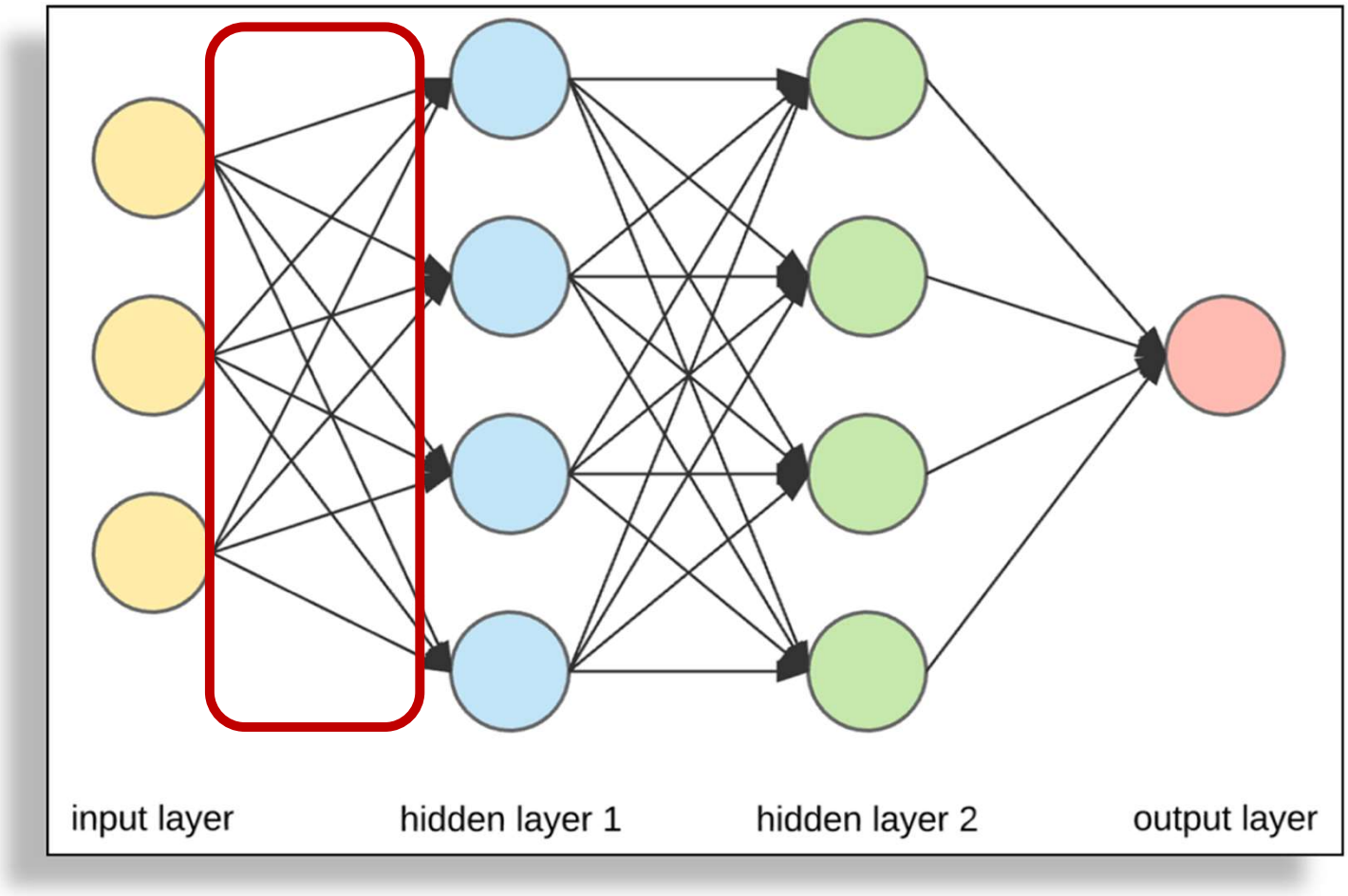


www.bigdatatrunk.com

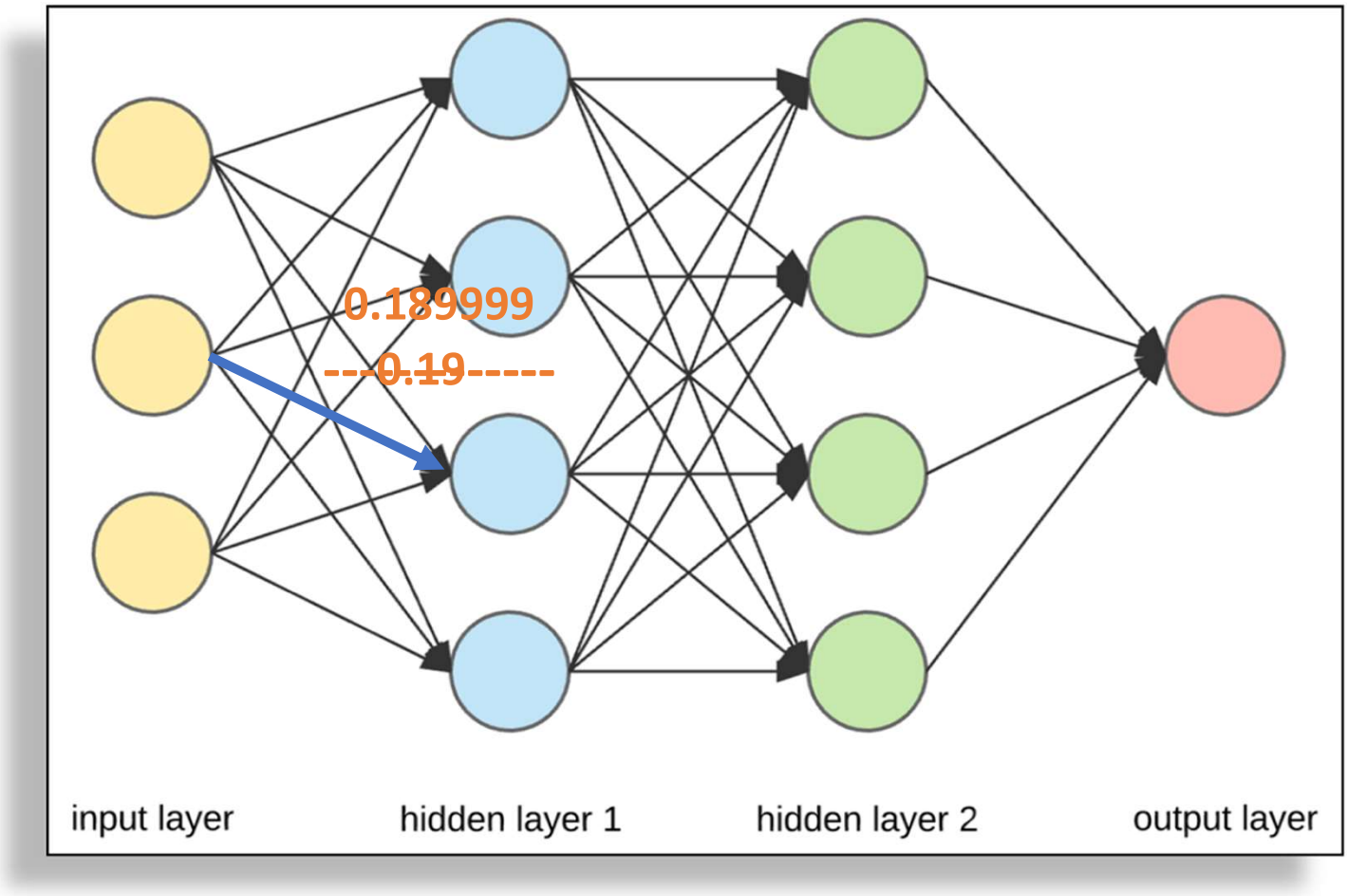
Vanishing and Exploding Gradients



Vanishing Gradient Problem



Vanishing Gradient Problem



www.bigdatatrunk.com

Exploding Gradient Problem

- Exploding Gradient Problem in the opposite of the Vanishing Gradient Problem – rather than the gradient that vanishes it explodes
- What if the initial layers had bigger input number (greater than 1)
- So when such input is multiplied by number > 1, this will result in a number much greater than 1
- Resulting in larger gradient – thus exploding in size
- The updated weights will also be larger (0.19 → 1.8765)

How to fix the Gradient Issues?

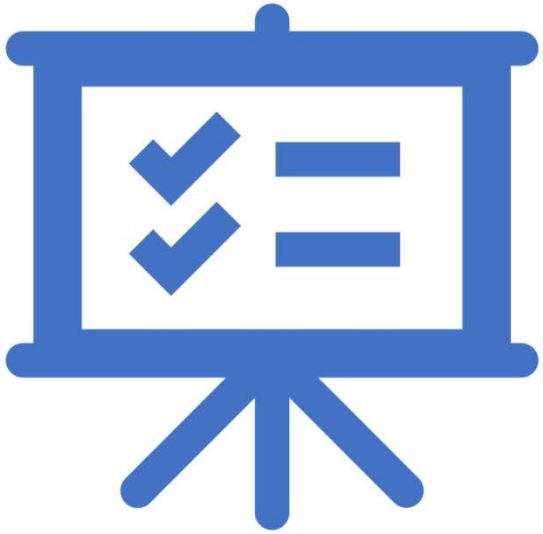
Exploding Gradients

- ✓ Instead of taking the data from first timestamp, use few timestamps to get the training going
- ✓ Use RMSprop optimizer to adjust the learning rate
- ✓ Set a threshold for the gradient if it goes above certain value

Vanishing Gradients

- ✓ Use different network architectures such as LSTM, GRU that are designed to handle such problems
- ✓ Use ReLU activation which takes negative values as zero and positive as one or greater, thereby helping in calculating gradient
- ✓ Use RMSprop to clip the gradient if it is going above certain threshold

Extra Credits



Extra Credits

- There are 4 datasets:
  1. Credit Card Application
  2. Churn Modelling
  3. Heart Disease
  4. Breast Cancer
- Your Work
  - Pick anyone of these datasets
  - Apply the techniques that you have learnt related to Feature Cleaning, Feature selection, etc.
  - Choose 1 or more algorithms and make predictions
  - Tomorrow afternoon we will review it with the class
- Description of datasets in following slides

Financial Customer Churn Data

- Dataset – Customer information with a financial institution
- If doing classification with SVM then it can be applied to most of the datasets where logistic regression is used
- Dataset has following features:

Dataset - Churn_Modelling.csv

RowNumber: Dataset row number
CustomerId: Customer Id
Surname: Last name of the person
CreditScore: Credit Score of the person
Geography: Country of residence
Gender: Person's Gender
AGE: Age of the person
Tenure: How long has the person owned the card
Balance: Outstanding balance
NumOfProducts: Number of products owned by the person with company
HasCrCard: Person has credit card
IsActiveMember: Is the person active member of the company
EstimatedSalary: Estimated salary of the person
Exited: Did the person stay or leave

Credit Card Application

- Based on 15 features (not named), predict whether a new customer's credit card application should be approved
- Predicting a "class" – 1 or 0

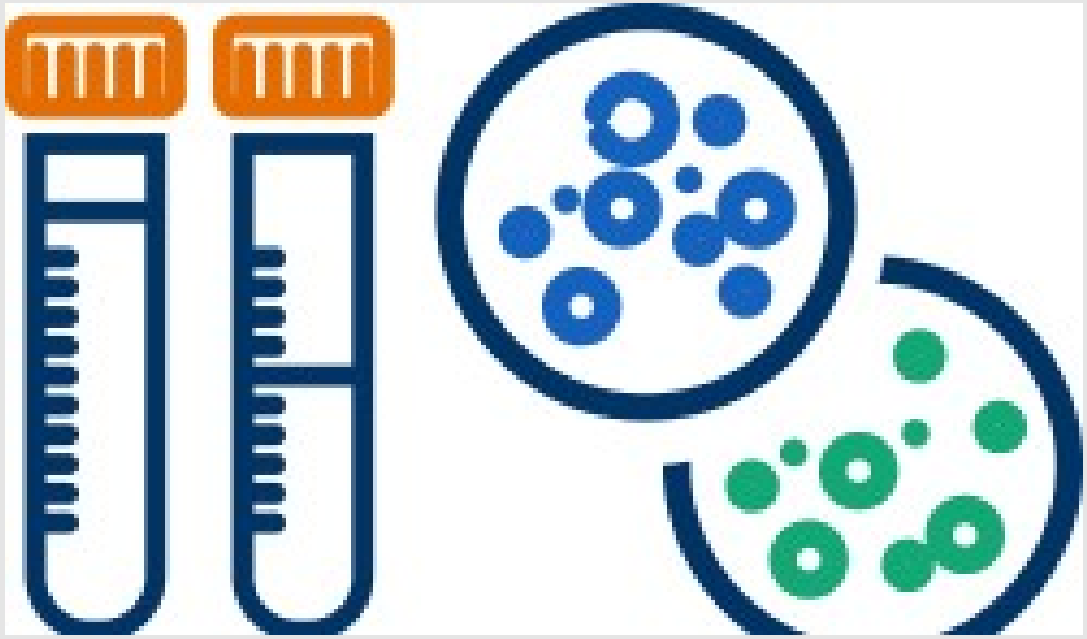
Dataset – Credit_Card_Applications.csv

Heart Disease

Dataset – Heart.csv

- Based on 13 features predict if a patient will have heart disease
- Features include age, gender, chest pain, resting blood pressure, cholesterol, fasting blood sugar, resting ECG, max heart rate, exercise induced angina, etc.
- Predicting a “target” – 1 or 0

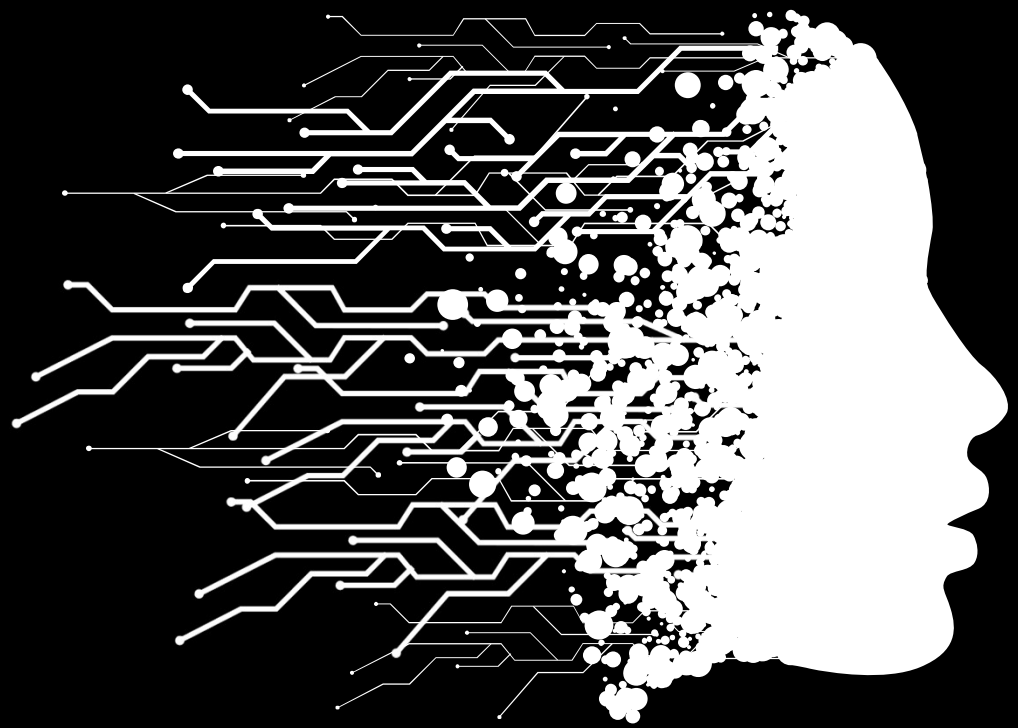
Breast Cancer



- Dataset includes a number of features related to tissue samples collected
- 2 Classes: benign and malignant (values 2 and 4 respectively)
- Attributes:
  - Sample code number
  - Clump Thickness,
  - Uniformity of Cell Size,
  - Uniformity of Cell Shape,
  - Marginal Adhesion,
  - Single Epithelial Cell Size,
  - Bare Nuclei,
  - Bland Chromatin,
  - Normal Nucleoli
  - Mitoses

Dataset:

breast-cancer-wisconsin.csv



Natural Language Processing (NLP)

NLP Use cases

Sentiment Recognition

We enjoyed the food and the service was great.

We had to wait for an excessive amount of time and our order was wrong.

© www.bigdatatrunk.com

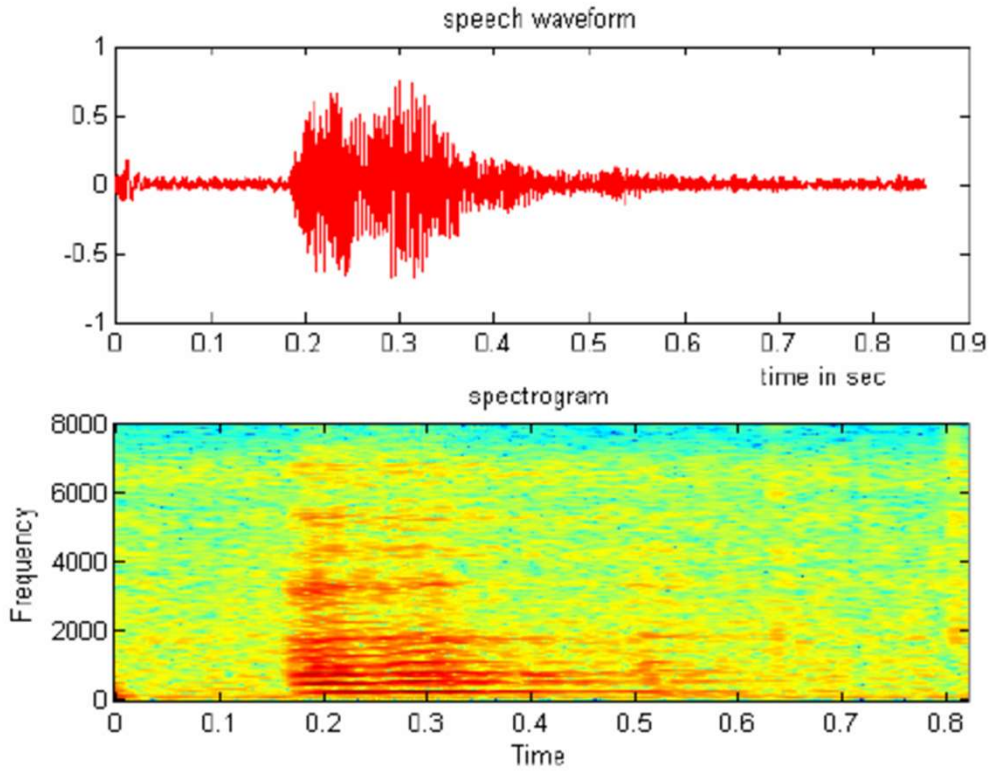
Machine Translation

A bird flies over the water

Machine Learning

Un oiseau vole au-dessus de l'eau

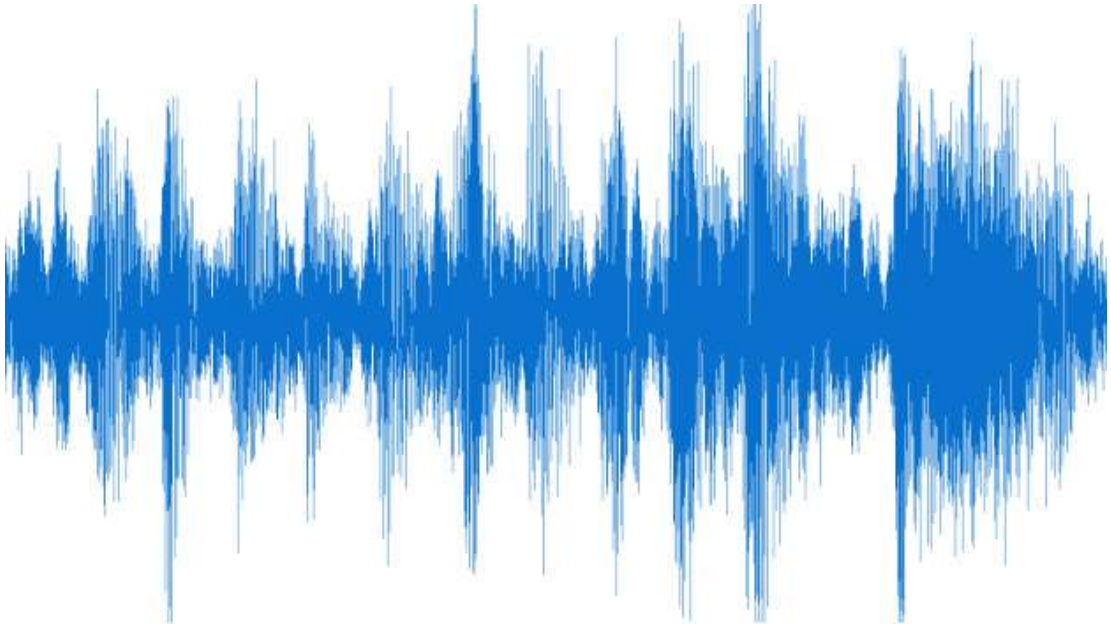
Speech Recognition (Speech to Text)



© www.bigdatatrunk.com

Speech Synthesis (Text to Speech)

Becoming Human



NLP Processing

Encoding Words

Sentiment Classification

"A touching movie full of emotions with wonderful acting. Worth seeing a second time!"

Pre-Process

A touching movie full of emotions with wonderful acting
Worth seeing a second time

© www.bigdatatrunk.com

Encoding Words

Pre-Processed

A touching movie full of emotions with wonderful acting
Worth seeing a second time

Lowercase

a touching movie full of emotions with wonderful acting
worth seeing a second time

© www.bigdatatrunk.com

Encoding Words

Lowercase

a touching movie full of emotions with wonderful acting worth seeing a second time

Tokenization

[‘a’, ‘touching’, ‘movie’, ‘full’, ‘of’, ‘emotions’, ‘with’, ‘wonderful’, ‘acting’, ‘worth’, ‘seeing’, ‘a’, ‘second’, ‘time’]

© www.bigdatatrunk.com

Encoding Words

Tokenized

['a', 'touching', 'movie', 'full', 'of', 'emotions', 'with', 'wonderful', 'acting', 'worth', 'seeing', 'a', 'second', 'time']

Stop words removal

['touching', 'movie', 'full', 'emotions', 'wonderful', 'acting', 'worth', 'seeing', 'second', 'time']

© www.bigdatatrunk.com

Encoding Words

Stop words removed

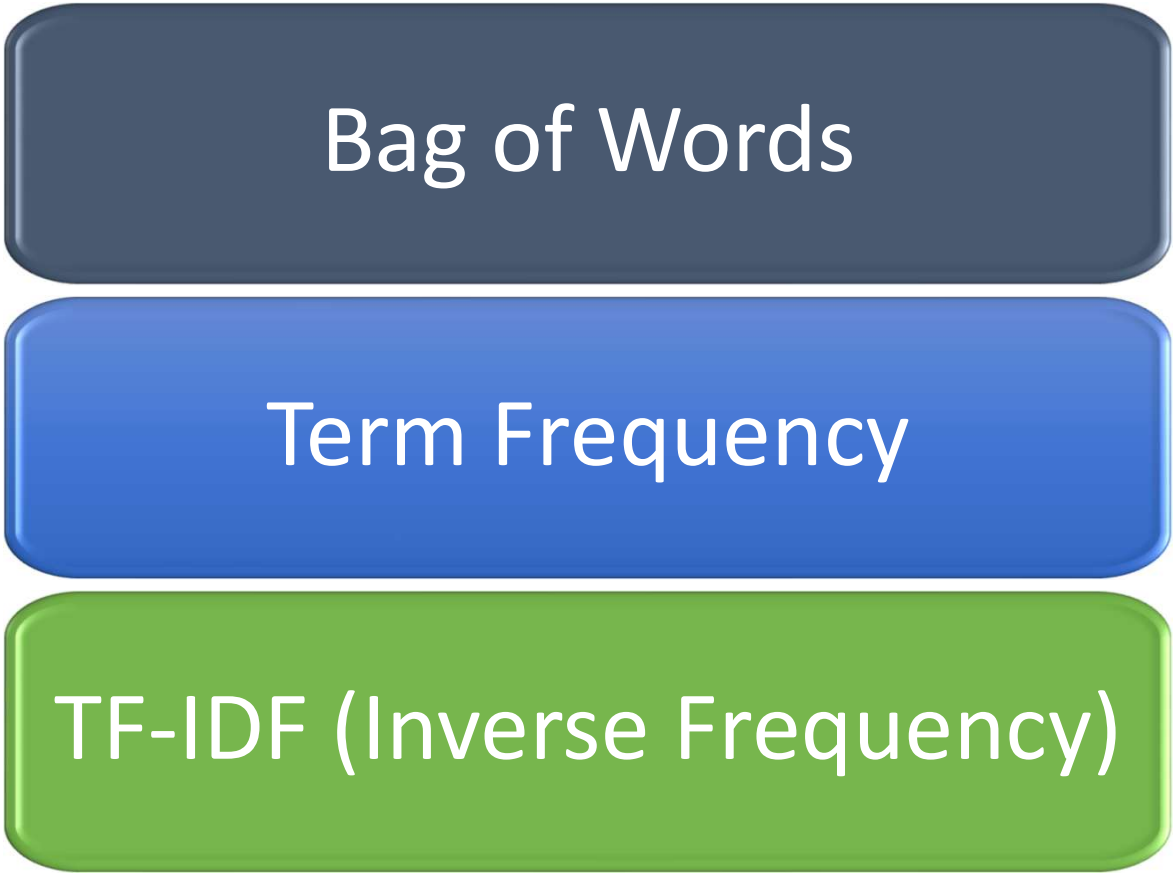
['touching', 'movie', 'full', 'emotions', 'wonderful', 'acting', 'worth', 'seeing', 'second', 'time']

Lemmatization

['touch', 'movie', 'full', 'emotion', 'wonder', 'act', 'worth', 'see', 'second', 'time']

© www.bigdatatrunk.com

Vectorization



Text Classification – Movie Review

- Build a Neural Network for training (build layers)
  - Embedding layer: Takes the integer-encoded reviews and looks up embedding vector for each word-index
  - GlobalAveragePooling layer: Take average of each sequence dimension
  - Add couple of ReLU activation layers
  - Finally add a sigmoid layer (that will output probability of the sentiment – positive or negative)
- Explore the embedding – check weights, write files to visualize the embeddings
- Plot trained model performance
- Evaluate the model with the test dataset
- Write couple of reviews and make predictions

Text Processing: Neural Networks

- Open file '**CodeSamples/TextClassification**' using Jupyter
- Take Movies Dataset and build model to predict positive or negative review

TensorFlow Projector

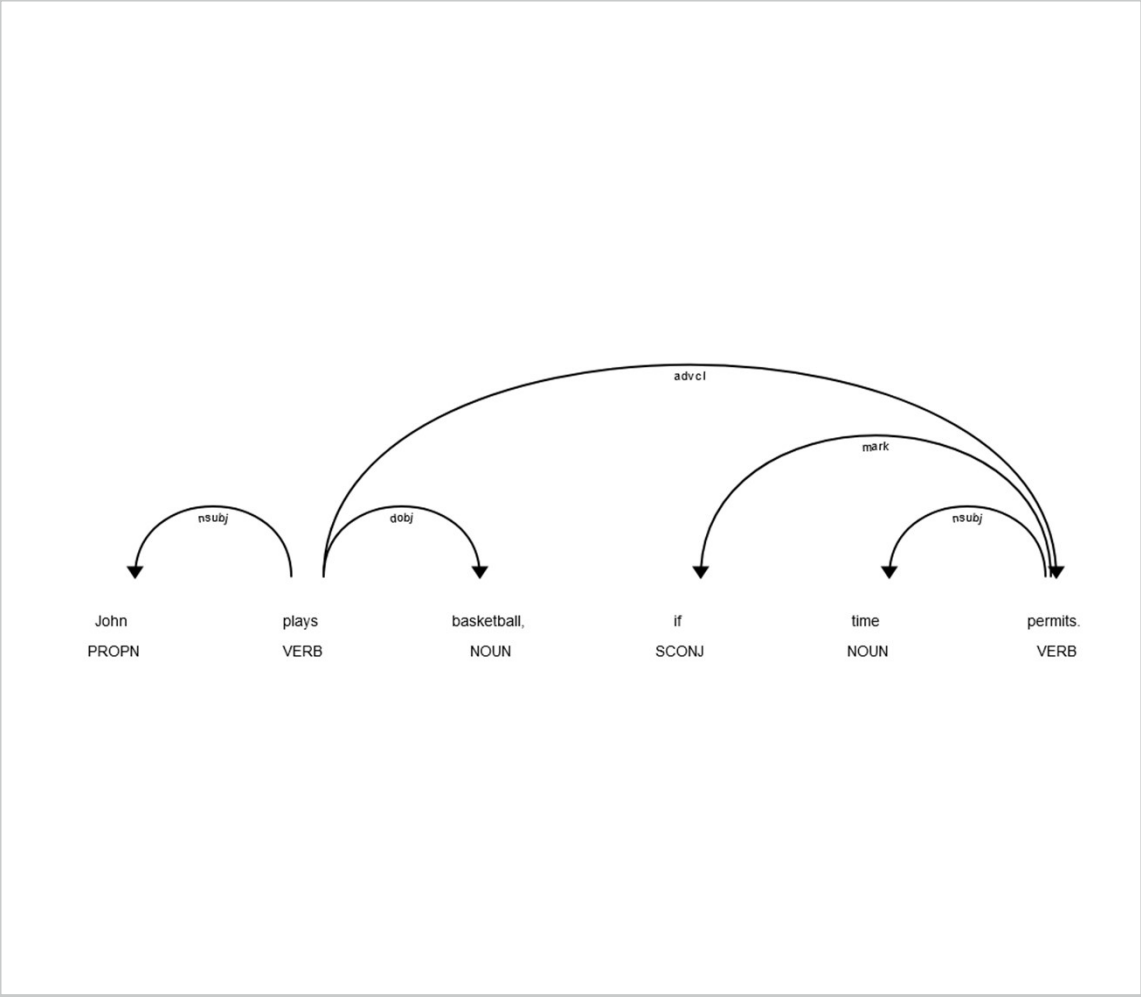
- Open link ‘https://projector.tensorflow.org/’
- Iris dataset
- MNIST with images
- Generated by Text Classification

Popular NLP Tools and Libraries

Name	Spark NLP	spaCy	NLTK	CoreNLP
Sentence detection	Yes	Yes	Yes	Yes
Tokenization	Yes	Yes	Yes	Yes
Stemming	Yes	Yes	Yes	Yes
Lemmatization	Yes	Yes	Yes	Yes
POS tagger	Yes	Yes	Yes	Yes
NER	Yes	Yes	Yes	Yes
Dependency parse	Yes	Yes	Yes	Yes
Text matcher	Yes	Yes	No	Yes
Date matcher	Yes	No	No	Yes
Chunking	Yes	Yes	Yes	Yes
Spell checker	Yes	No	No	No
Sentiment detector	Yes	No	No	Yes
Pretrained models	Yes	Yes	Yes	Yes
Training models	Yes	Yes	Yes	Yes

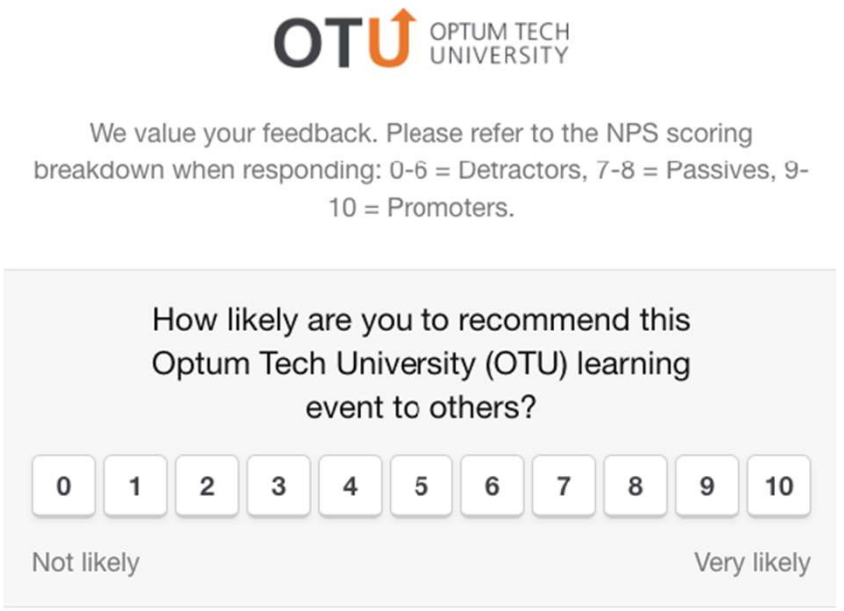
spaCy

- Natural language processing library
- Fast – written using Cython
  - Cython is a superset of python that compiles to C code, thus providing better performance
  - Numpy is a wrapper around C libraries
- spaCy supports following functionality:
  - Tokenizer
  - Creating word vectors
  - Syntactic Parser
  - Named Entity Recognition



Survey

- The Optum Tech University (OTU) team would like your feedback regarding your learning experience in this class. Input from participants about their experience helps to continually enhance the value and effectiveness of courses like this.
- Thank you!



spaCy Examples

- Open file '**CodeSamples/SpacyExamples**' using Jupyter
- Different examples of using the spaCy library

spaCy Examples

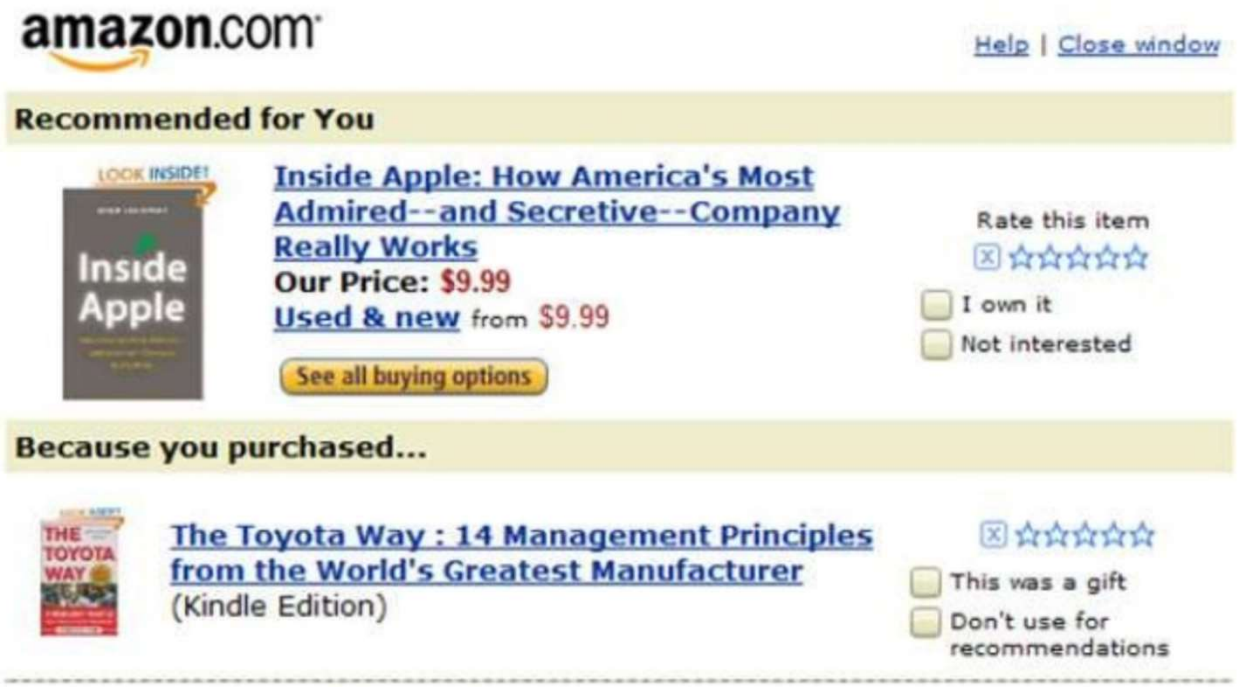
- Open file '**CodeSamples/Spacy-SentimentAnalysis**' using Jupyter
- Will use spaCy to preprocess the text
- Build a pipeline to clean the text, vectorize it
- Apply logistic regression to classify the text (positive or negative)

Recommendation Systems

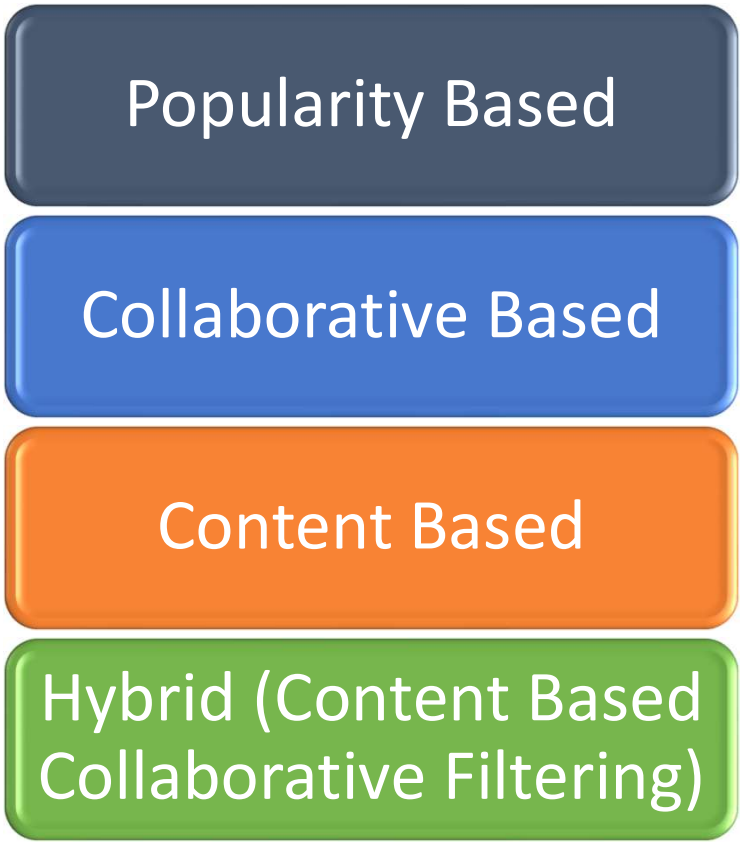
www.bigdatatrunk.com

Recommendation System

Example

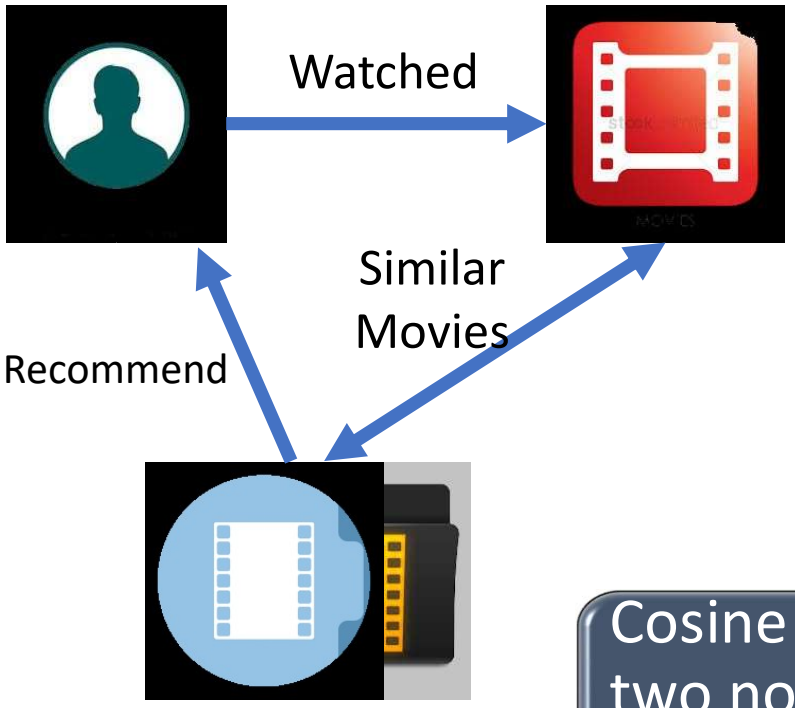

amazon.com
Help | Close window
Recommended for You
LOOK INSIDE!
Inside Apple
Inside Apple: How America's Most Admired--and Secretive--Company Really Works
Our Price: $9.99
Used & new from $9.99
See all buying options
Rate this item
I own it
Not interested
Because you purchased...
THE TOYOTA WAY
The Toyota Way : 14 Management Principles from the World's Greatest Manufacturer
(Kindle Edition)
This was a gift
Don't use for recommendations

Types of Recommendation Systems



www.bigdatatrunk.com

Content based filtering



User ratings – movies liked and dis-liked (Profile Vector – X)

Item information – movie genre, cast, length, etc. (Item Vector – Y)

$$\cos(\theta) = \frac{\sum_i X_i Y_i}{\sqrt{\sum_i X_i^2}\sqrt{\sum_i Y_i^2}}$$

Cosine Similarity – measure of similarity between two non-zero vectors, movies are sorted in descending order based on these cosine values

Content Based

- Open file '**CodeSamples/CosineSimilarity**' using Jupyter
- Movie recommendation using keywords, cast, director and genres

Assignment

- Open file ‘Assignment/CRM-CustomerChurn’ using Jupyter
- Implementation requirements are defined in the notebook

Assignment

- Open file ‘Assignment/CNN-Fashion’ using Jupyter
- Implementation requirements are defined in the notebook

Dataset Sources

- UCI – Machine Learning Repository https://archive.ics.uci.edu/ml/index.php
- Kaggle - https://www.kaggle.com/
- Google - https://toolbox.google.com/datasetsearch
- Amazon – https://aws.amazon.com/opendata/public-datasets/
- US Government - https://www.data.gov/

Next steps

- Make a custom plan for yourself to continue this journey
- Improve some of your skills (Stats, Python, ML, Domain, etc.)
- Take some additional courses
- Try a Kaggle.com competition
- Work on a personal project to improve understanding

Glossary

Machine Learning Glossary

https://developers.google.com/machine-learning/glossary

Thank You