hochschule mannheim

Fakultät für Wirtschaftsingenieurwesen

Machine Learning with python

Prof. Dr. Stefan Rist

Content

- Introduction to machine learning
- Linear Classifiers
- Feed Forward Neural Networks
- Convolution Neural Networks

Great online resources

MITx: 6.86x
Machine Learning with Python-From Linear Models to Deep Learning

MIT Introduction to Deep Learning | 6.S191

Alexander Amini • 477K views • 1 month ago

https://www.kaggle.com

kaggle Competitions Datasets Models

Introduction

Definition

Machine learning as a discipline aims to design, understand and apply computer programs that learn from experience (i.e., data) for the purpose of modeling, prediction, or control

MITx: 6.86x

Introduction

When should it be applied?

It is easier to express what one wants in terms of examples rather than to figure out how to solve the problem

e.g. image classification

Image	Category
 . © Neural Information Processing Systems Foundation, Inc.	mushroom
 8. © Geoffrey Hinton, University of Toronto.	cherry
...	...

MITx: 6.86x

Introduction

Image | **Category**

$h($  $\theta) =$ mushroom

. © Neural Information Processing Systems Foundation, Inc.

cherry

8. © Geoffrey Hinton, University of Toronto.

... ...

Rather than solving the problem directly (hard) we automate the process of finding a solution by giving examples

MITx: 6.86x

Introduction

a concrete example

11.© 2020 Sony Pictures Digital Productions Inc.

10. © Twentieth Century Fox.

9. © 2020 The Studios At Paramount.

12.© Disney

© Black Bear Pictures

-1 -1 +1 +1 ?, ?,

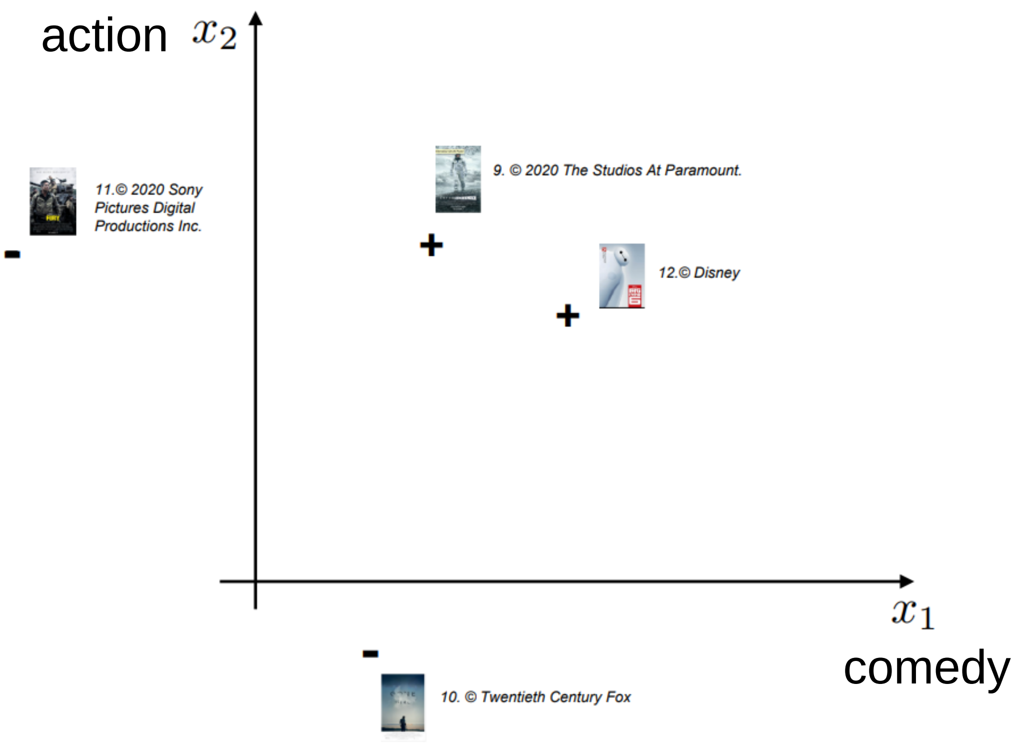
comedy? top lead?

$$x^{(1)} = [1\ 0\ 0\ 1\ 1\ \ldots\ 0]^T \qquad \text{(feature vector)}$$

action? romance? Spielberg?

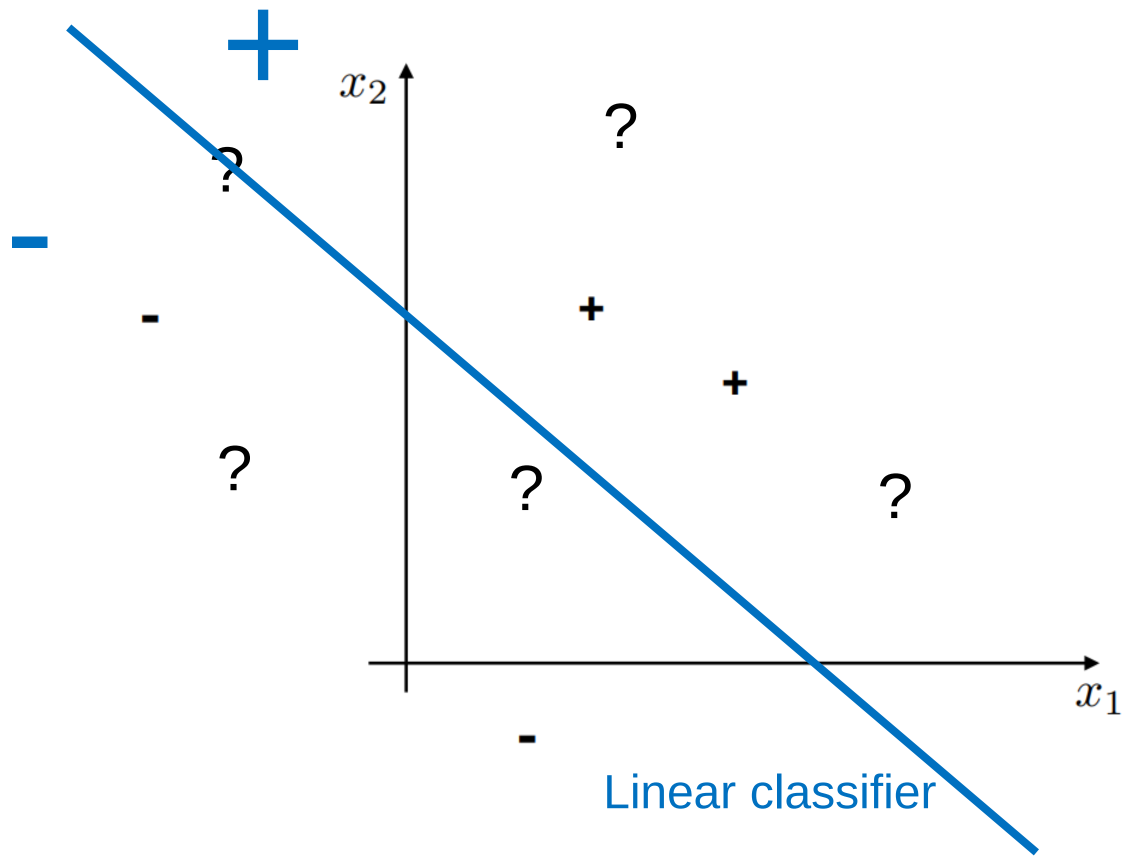
MITx: 6.86x

Introduction



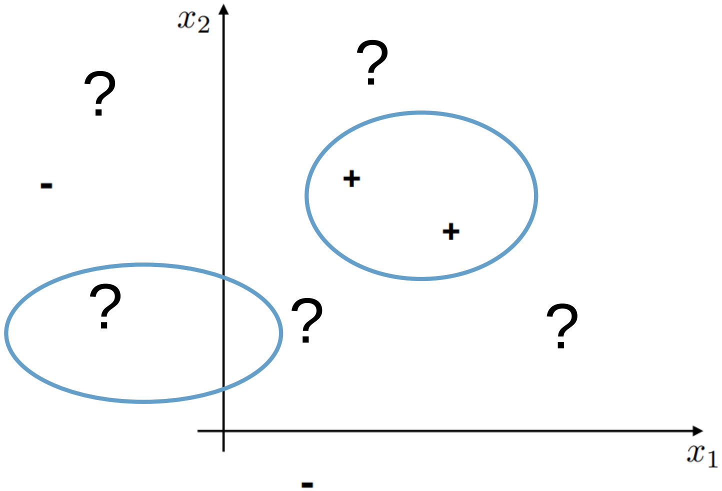
MITx: 6.86x

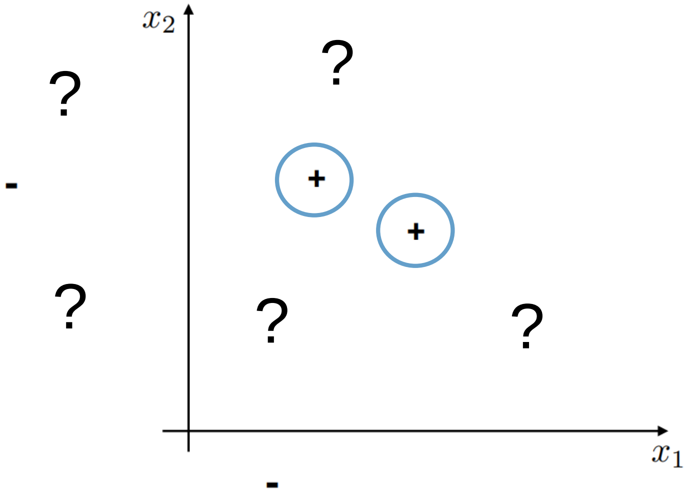
Introduction



MITx: 6.86x

Introduction





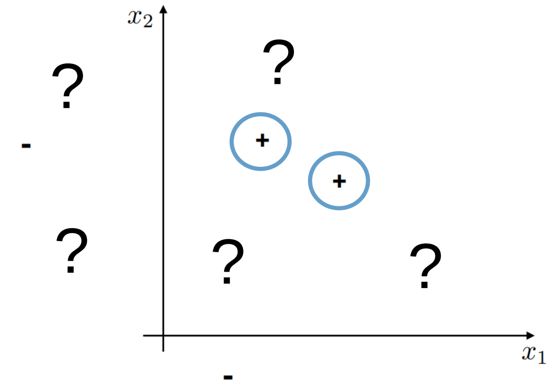
Nonlinear classifier

Can give better results if data is not linearly separabel

Very unlikely to generalize well
(Give good results on unseen data)

MITx: 6.86x

Introduction



Complexity of classifiers: Number of possible classifiers (number of free parameters)

Ideally we can find a classifier with a small number of parameters that works well on the training set. Then we expect good generalization

Linear classifier

$$\theta_1 x_1 + \theta_2 x_2 + \theta_0 = 0$$

NonLinear classifier

$$\theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + \theta_0 = 0$$

MITx: 6.86x

Introduction

Types of machine learning:

Supervised learning: predictions based on examples with correct behaviour (labeled examples)

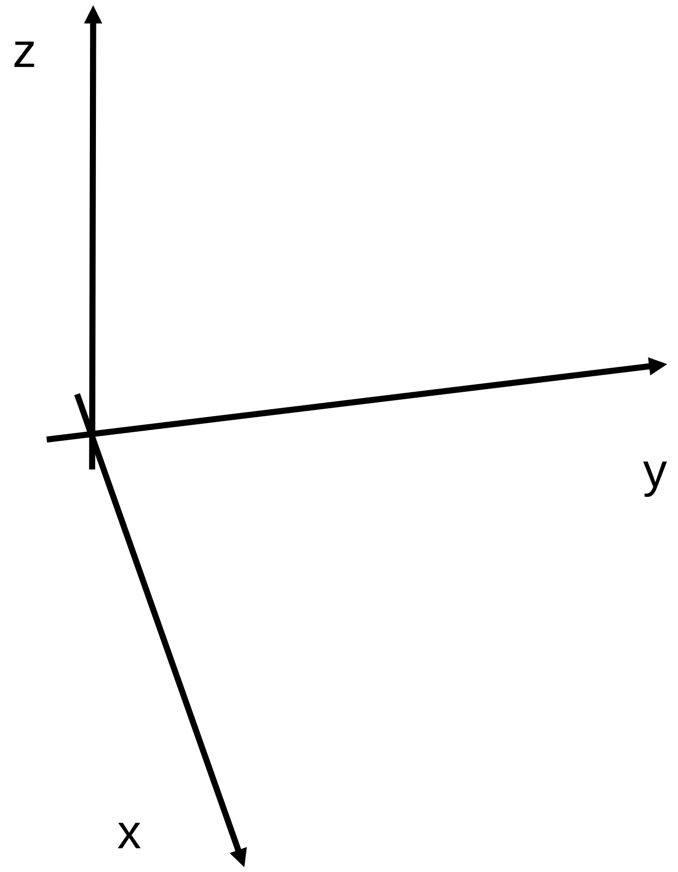
Unsupervised learning: no explicit target, only data -> search for structure

Reinforcement learning: learning to act, not just predict, goal is to optimize the consequences of actions

MITx: 6.86x

Linear classifiers

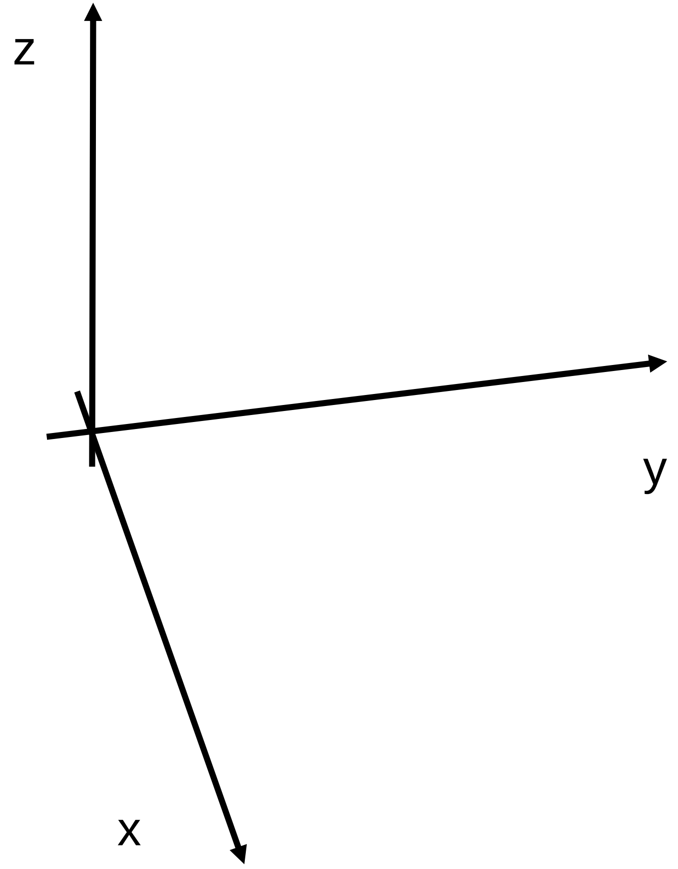
Review Points and vectors



MITx: 6.86x

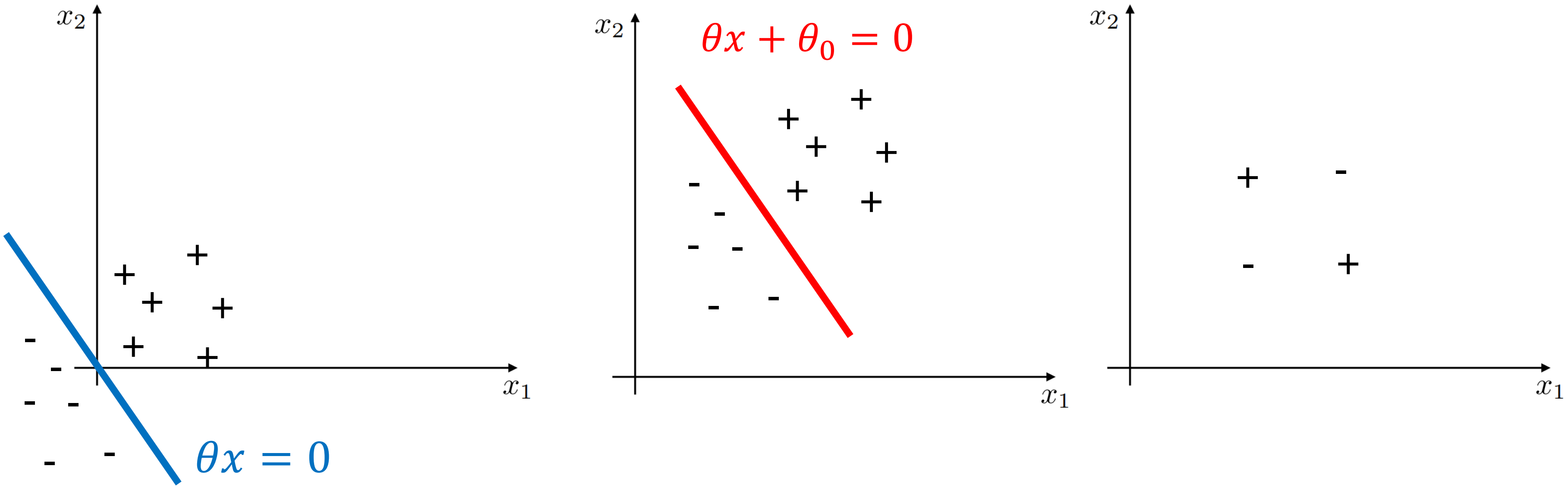
Linear classifiers

Review Planes



MITx: 6.86x

Linear classifiers



Linearly separable with zero offset

Linearly separable with offset

Not linearly separable (in 2D)

MITx: 6.86x

Linear classifiers

Definition:
Training examples $S_n = \{(x^{(i)}, y^{(i)}\}), i = 1, \ldots, n\}$ are *linearly separable* if there exists a parameter vector $\hat{\theta}$ and offset parameter $\hat{\theta}_0$ such that $y^{(i)}(\hat{\theta} \cdot x^{(i)} + \hat{\theta}_0) > 0$ for all $i = 1, \ldots, n$.

Training Error:

$$E(h) = \frac{1}{n}\sum_{i=1}^{n}[\![h(x^i) \neq y^i]\!] = \frac{1}{n}\sum_{i=1}^{n}[\![y^i \cdot (\theta \cdot x + \theta_0) < 0]\!]$$

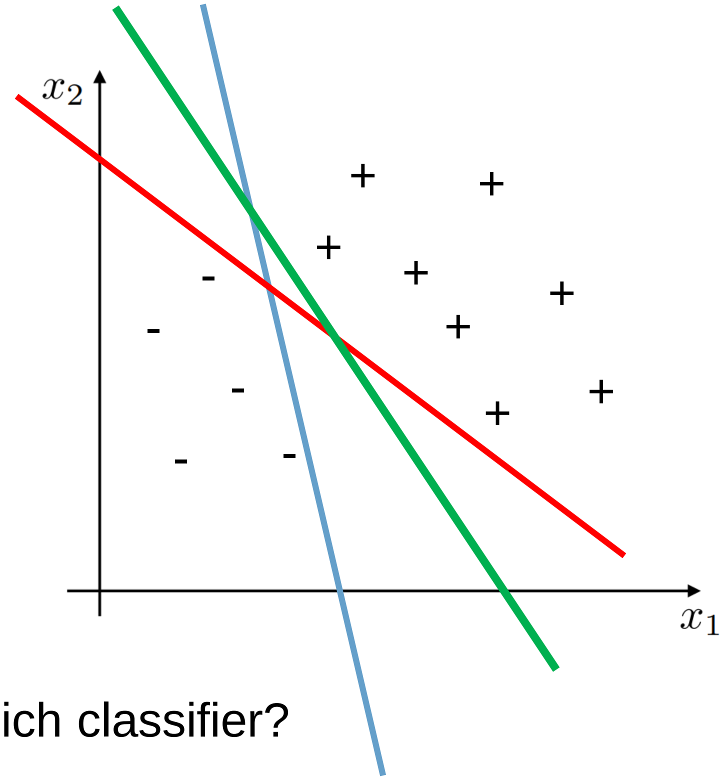
$$[\![True]\!] = 1$$

$$[\![False]\!] = 0$$

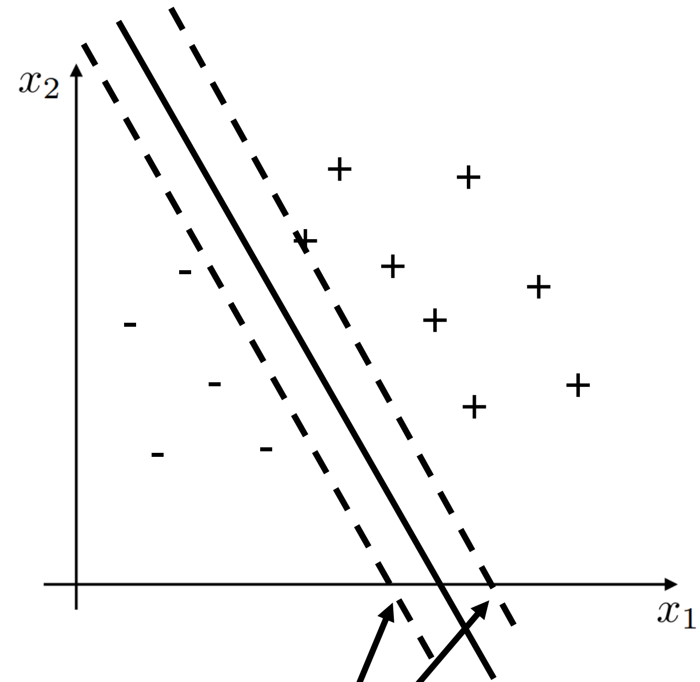
MITx: 6.86x

Linear classifiers

Hinge Loss, Margin Boundaries and Regularization



which classifier?

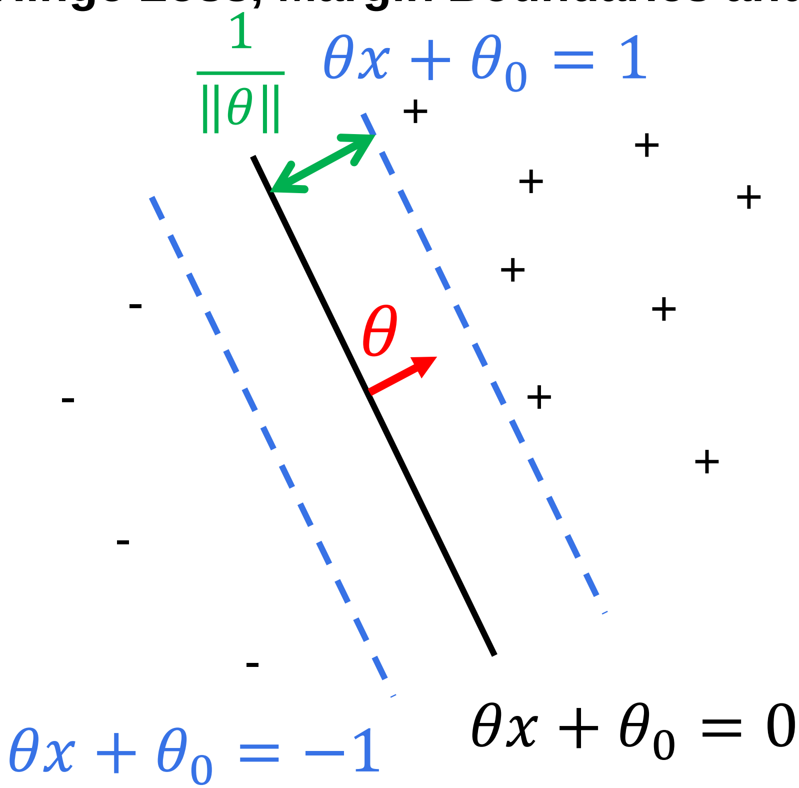


margin boundaries

MITx: 6.86x

Linear classifiers

Hinge Loss, Margin Boundaries and Regularization

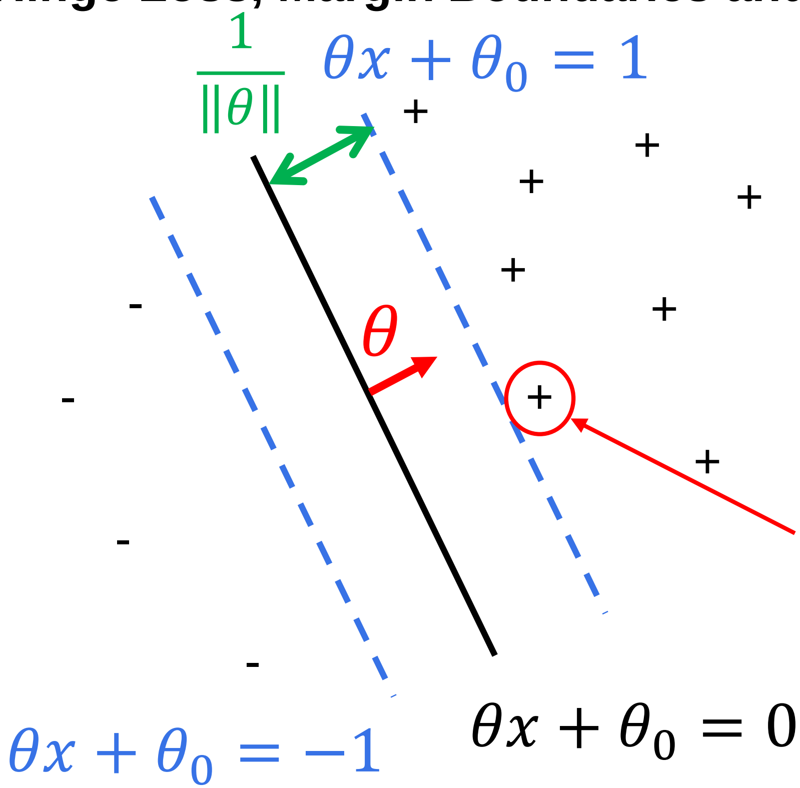


decision boundary	$\theta x + \theta_0 = 0$
margin boundary	$\theta x + \theta_0 = \pm 1$

MITx: 6.86x

Linear classifiers

Hinge Loss, Margin Boundaries and Regularization



support vector
SVM (support vector machine)

Linear separable case:

minimize $\frac{1}{2}||\theta||$

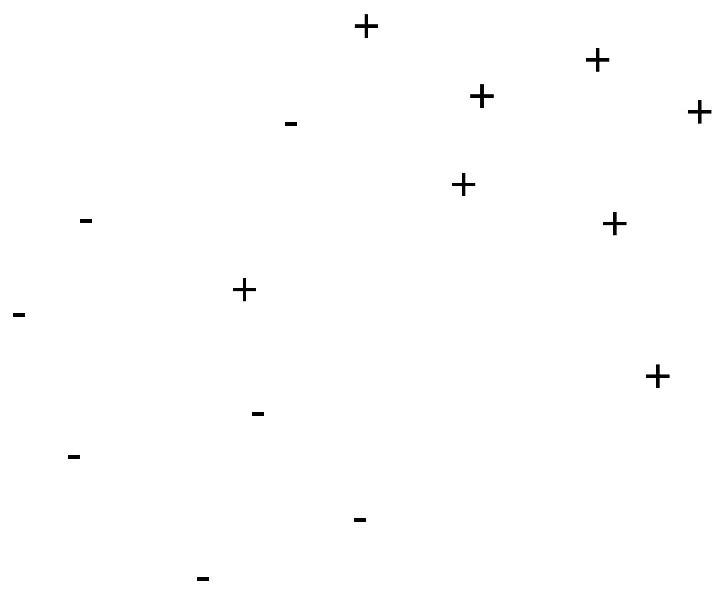
$y^{(i)}\left(\theta x^{(i)} + \theta_0\right) \geq 1$ for all i=1…n

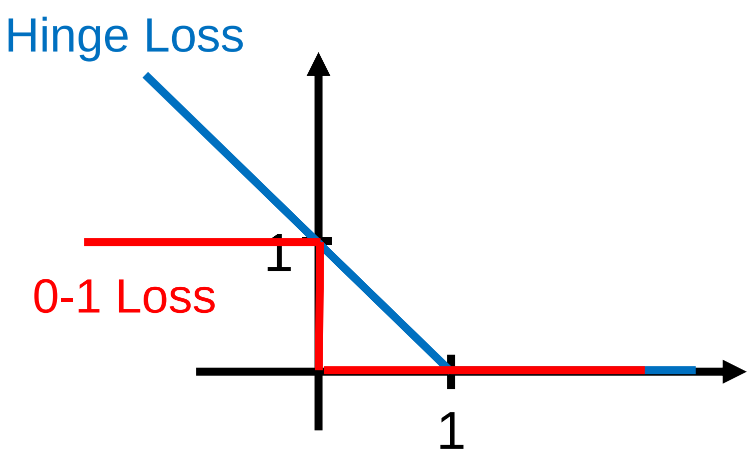
MITx: 6.86x

Linear classifiers

Hinge Loss, Margin Boundaries and Regularization

non separable case





$$\mathrm{Loss}_h\left(y^{(i)}(\theta \cdot x^{(i)} + \theta_0)\right) =$$

MITx: 6.86x

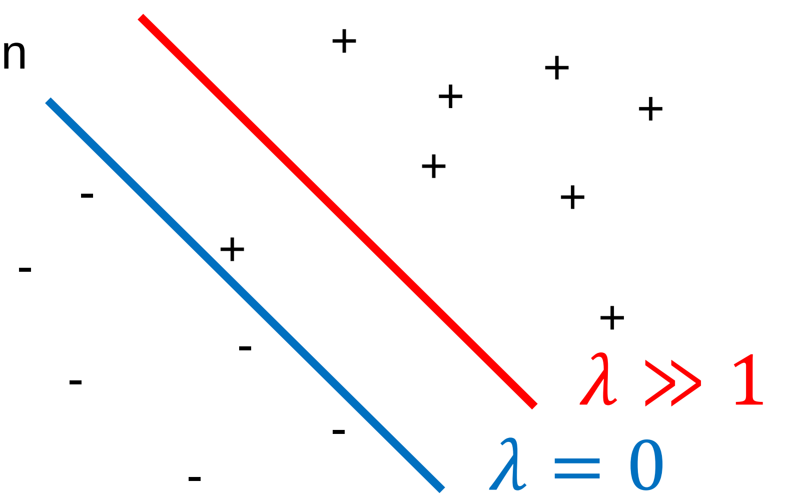
Linear classifiers

Hinge Loss, Margin Boundaries and Regularization

objective function

$$J(\theta, \theta_0) = \underbrace{\frac{1}{n}\sum_{i=1}^{n} \mathrm{Loss}_h\left(y^{(i)}(\theta \cdot x^{(i)} + \theta_0)\right)}_{\text{Loss}} + \underbrace{\frac{\lambda}{2}\|\theta\|^2}_{\text{Regularization}}$$

effect of regularization



MITx: 6.86x

Linear classifiers

Hinge Loss, Margin Boundaries and Regularization

objective function

$$J(\theta, \theta_0) = \underbrace{\frac{1}{n} \sum_{i=1}^{n} \text{Loss}_h \big(y^{(i)} (\theta \cdot x^{(i)} + \theta_0) \big)}_{\text{Loss}} + \underbrace{\frac{\lambda}{2} \|\theta\|^2}_{\text{Regularization}}$$

Loss: Try to minimize errors on training data

Regularization: Try to get a model that generalizes well.
-> Good performance on unseen data

MITx: 6.86x

Linear classifiers

Hinge Loss, Margin Boundaries and Regularization

objective function
$$J(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n} \text{Loss}_h\left(y^{(i)}(\theta \cdot x^{(i)} + \theta_0)\right) + \frac{\lambda}{2}\|\theta\|^2$$

MITx: 6.86x

Linear classifiers

Example: Iris Dataset

The ***Iris* flower data set** or **Fisher's *Iris* data set** is a multivariate data set used and made famous by the British statistician and biologist Ronald Fisher in his 1936 paper *The use of multiple measurements in taxonomic problems* as an example of linear discriminant analysis.[1] It is sometimes called **Anderson's *Iris* data set** because Edgar Anderson collected the data to quantify the morphologic variation of *Iris* flowers of three related species.[2] Two of the three species were collected in the Gaspé Peninsula "all from the same pasture, and picked on the same day and measured at the same time by the same person with the same apparatus".[3]

The data set consists of 50 samples from each of three species of *Iris* (*Iris setosa*, *Iris virginica* and *Iris versicolor*). Four features were measured from each sample: the length and the width of the sepals and petals, in centimeters. Based on the combination of these four features, Fisher developed a linear discriminant model to distinguish the species from each other. Fisher's paper was published in the Annals of Eugenics and includes discussion of the contained techniques' applications to the field of phrenology.[1]

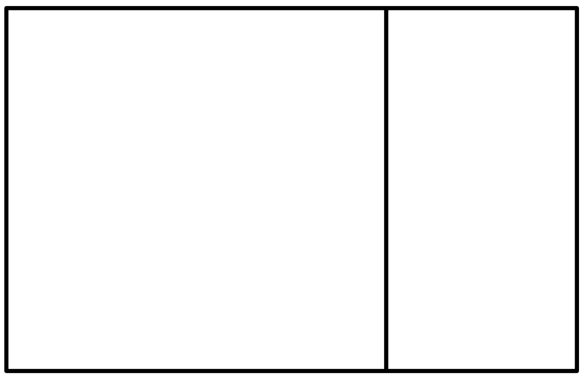
wikipedia

Linear classifiers

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} \mathrm{Loss}_h\left(y^{(i)}(\theta \cdot x^{(i)} + \theta_0)\right) + \frac{\lambda}{2}\|\theta\|^2$$

Validation

data for training



use to fit

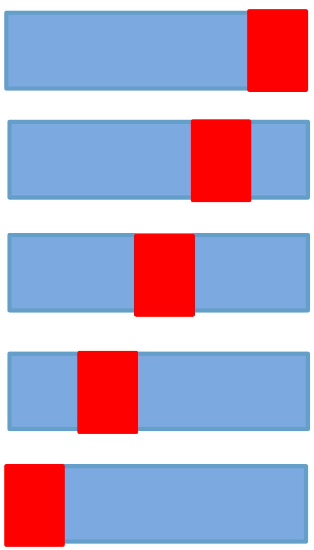
θ, θ_0

use for validation
get optimal

λ

Cross validation
(change splitting and repeat trying)

$\lambda_1, \lambda_2, \dots \lambda_n$



$A_1(\lambda_k)$

$A_2(\lambda_k)$

$A_3(\lambda_k)$

$A_4(\lambda_k)$

$A_5(\lambda_k)$

$$A(\lambda_k) = \frac{1}{5}\sum_m A_m(\lambda_k)$$

Repeat for each λ, and choose the one with the best accuracy

MITx: 6.86x

Linear classifiers

Gradient descent

$$J(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} \mathrm{Loss}_h\left(y^{(i)}(\theta \cdot x^{(i)} + \theta_0)\right) + \frac{\lambda}{2} \|\theta\|^2$$

MITx: 6.86x

Linear classifiers

Stochastic Gradient Descent (SGD)

$$J(\theta) = \frac{1}{n} \sum_{i=1}^{n} \left[\text{Loss}_h(y^{(i)} \theta \cdot x^{(i)}) + \frac{\lambda}{2} \|\theta\|^2 \right]$$

$$\text{Select } i \in \{1, \ldots, n\} \text{ at random}$$

$$\theta \leftarrow \theta - \eta_t \nabla_\theta \left[\text{Loss}_h(y^{(i)} \theta \cdot x^{(i)}) + \frac{\lambda}{2} \|\theta\|^2 \right]$$

Use Batches:
Compromise, use more than 1 Datapoint to calculate a gradient descent step but not all. Batchsize is then a hyperparameter for training.

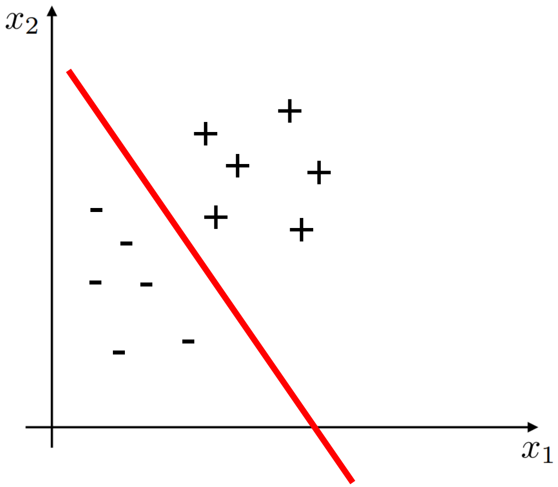
MITx: 6.86x

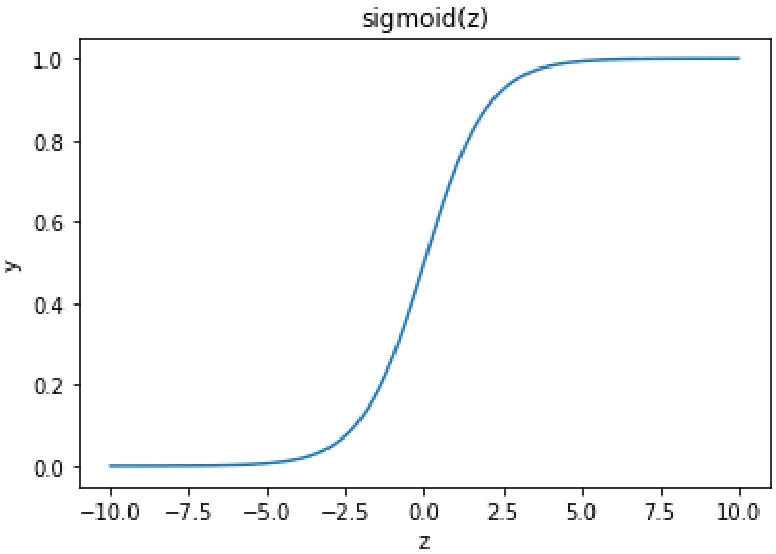
Linear classifiers

Logistic Regression (Softmax)

$$sigmoid(z) = \frac{1}{1 - e^{-z}}$$

$$z = \theta x + \theta_0$$



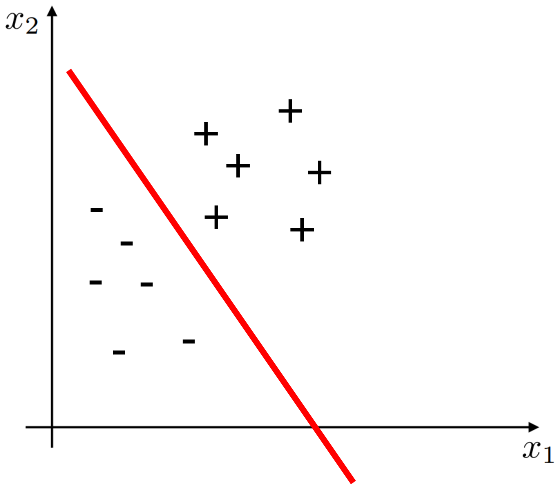


MITx: 6.86x

Linear classifiers

Logistic Regression (Softmax)

Find θ, θ_0 such that Liklihood for observed data is maximized



$$\max L = \prod_{j=1}^{n} p_j = \prod_{j=1}^{n} \mathrm{sig}(\theta \cdot x^{(j)} + \theta_0)$$

Same as minimization of neg. LogLiklihood

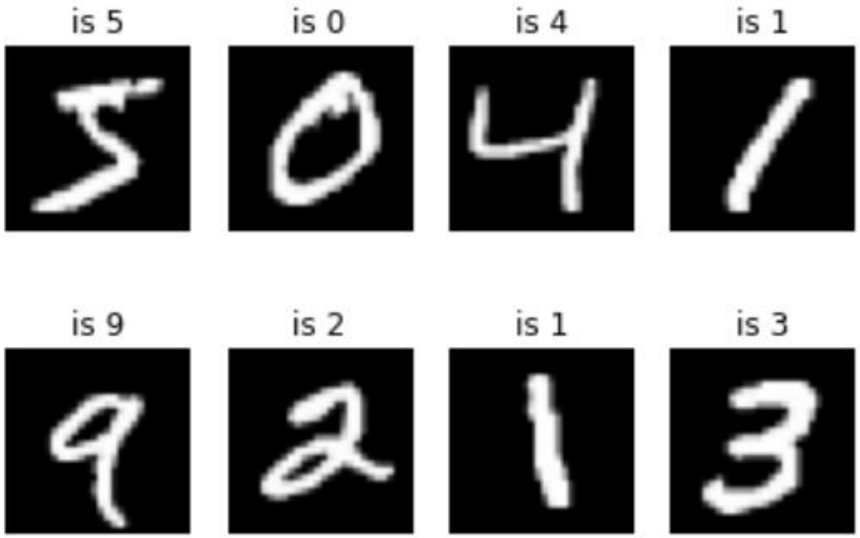
$$J(\theta) \;=\; \sum_{j=1}^{n} -\log\left(\mathrm{sig}(\theta \cdot x^{(j)} + \theta_0)\right) + \frac{\lambda}{2}||\theta||^2$$

Regularization

MITx: 6.86x

Linear classifiers

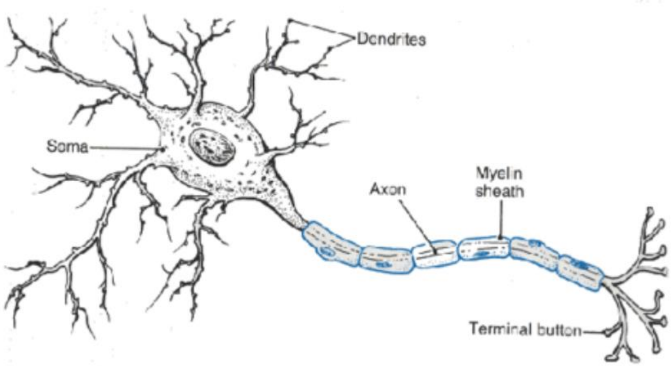
Example: Digit recognition



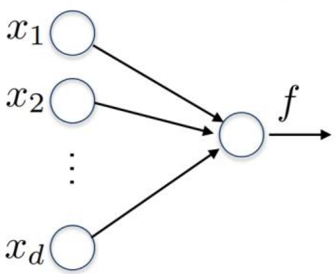
Try to classifiy handwritten numbers

Feed Forward Neural Networks

Introduction

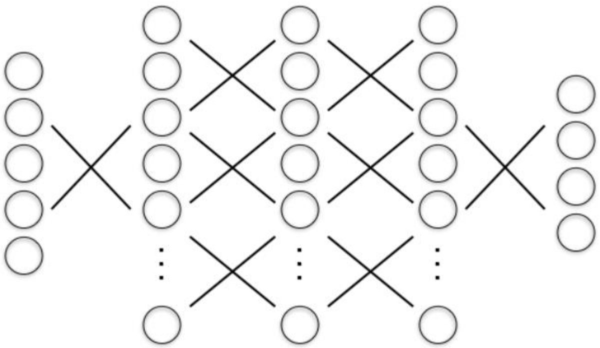


(c) Michael DeBellis



(e.g., a linear classifier)

2. Image on Wikimedia by Users: Ramón Santiago y Cajal.



MITx: 6.86x

Feed Forward Neural Networks

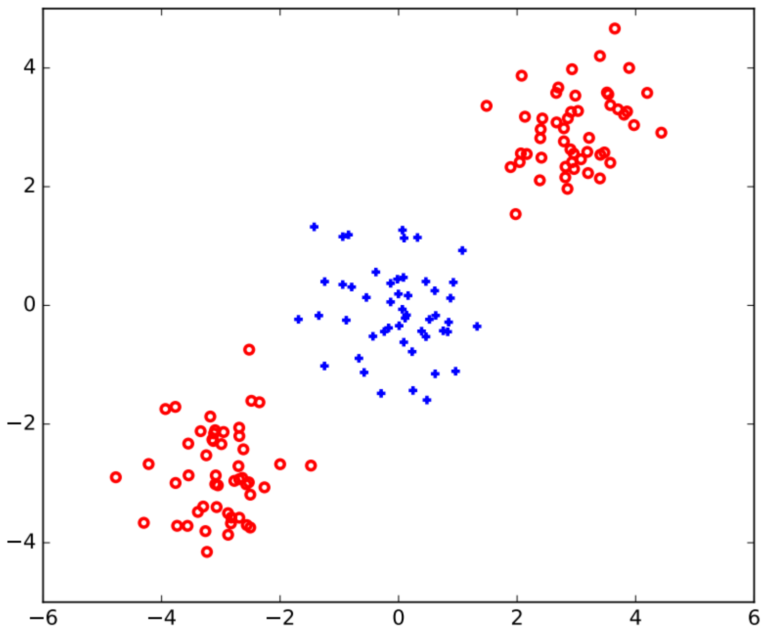
Reasons for success:

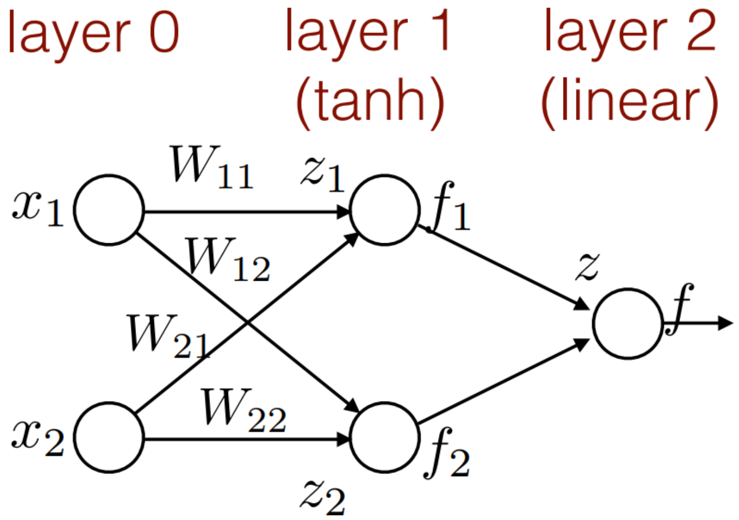
- **Reason #1:** lots of data
  - many significant problems can only be solved at scale
- **Reason #2:** computational resources (esp. GPUs)
  - platforms/systems that support running deep (machine) learning algorithms at scale
- **Reason #3:** large models are easier to train
  - large models can be successfully estimated with simple gradient based learning algorithms
- **Reason #4:** flexible neural “lego pieces”
  - common representations, diversity of architectural choices

MITx: 6.86x

Feed Forward Neural Networks

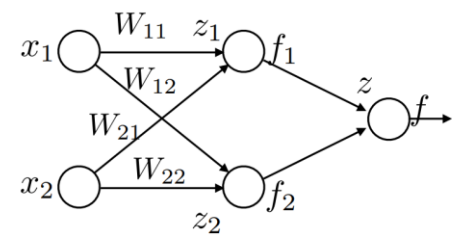
Simple model:





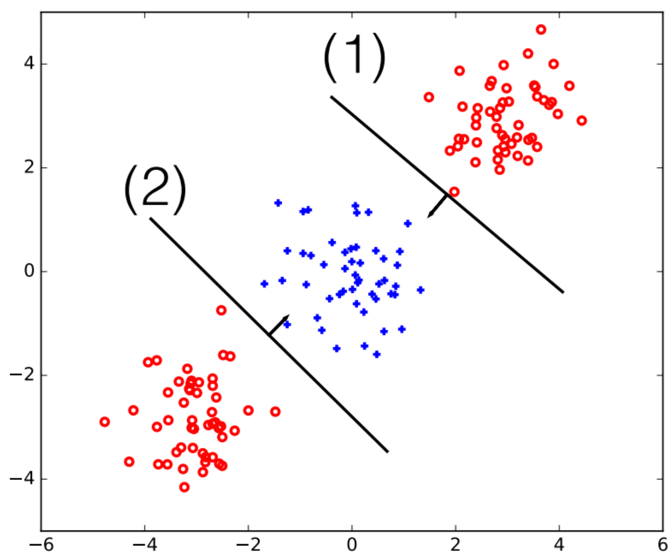
MITx: 6.86x

Feed Forward Neural Networks

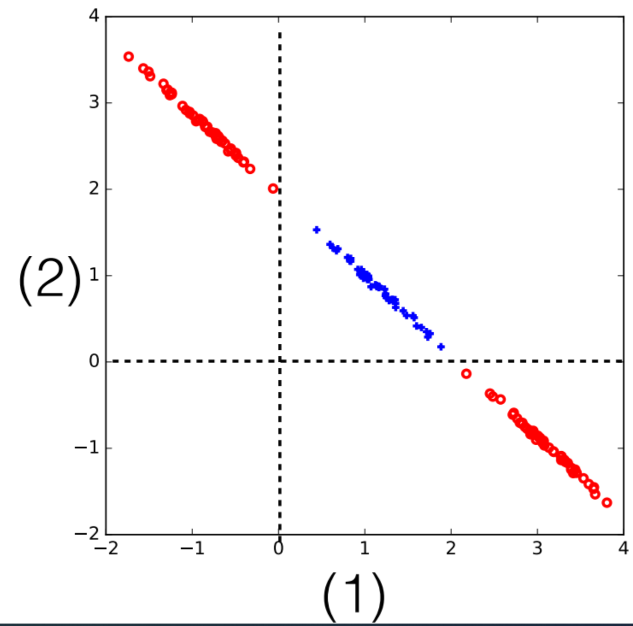


Simple model:

Hidden layer units



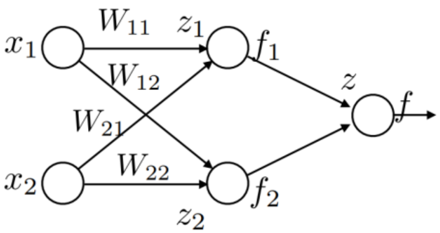
Linear activation



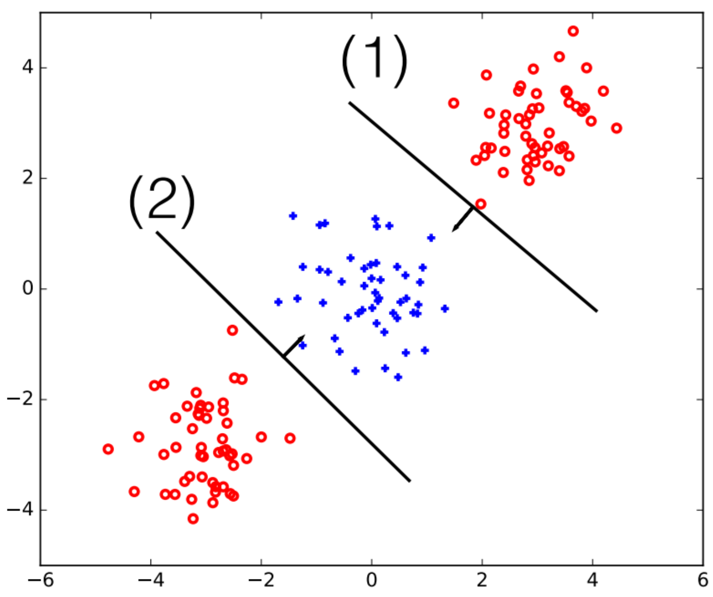
MITx: 6.86x

Feed Forward Neural Networks

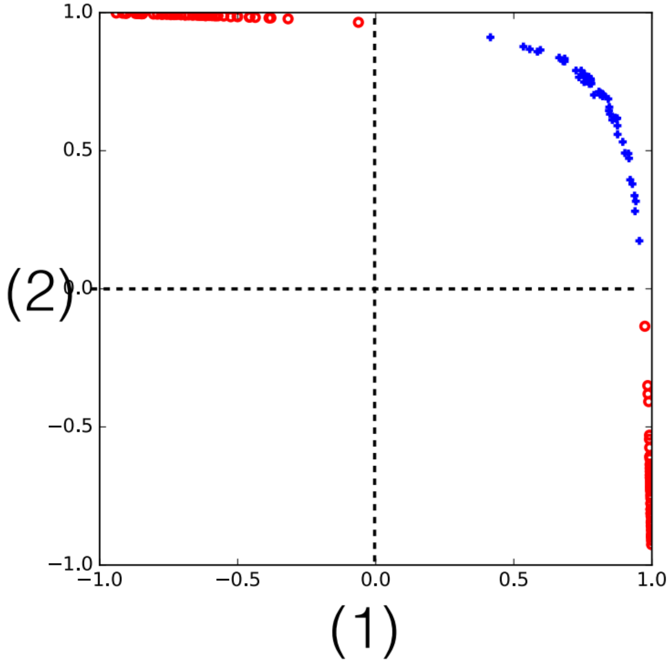
Simple model:



Hidden layer units

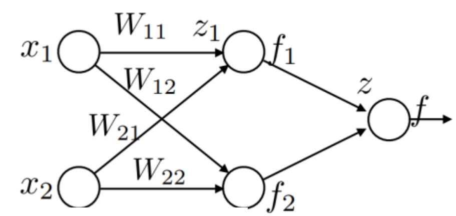


tanh activation

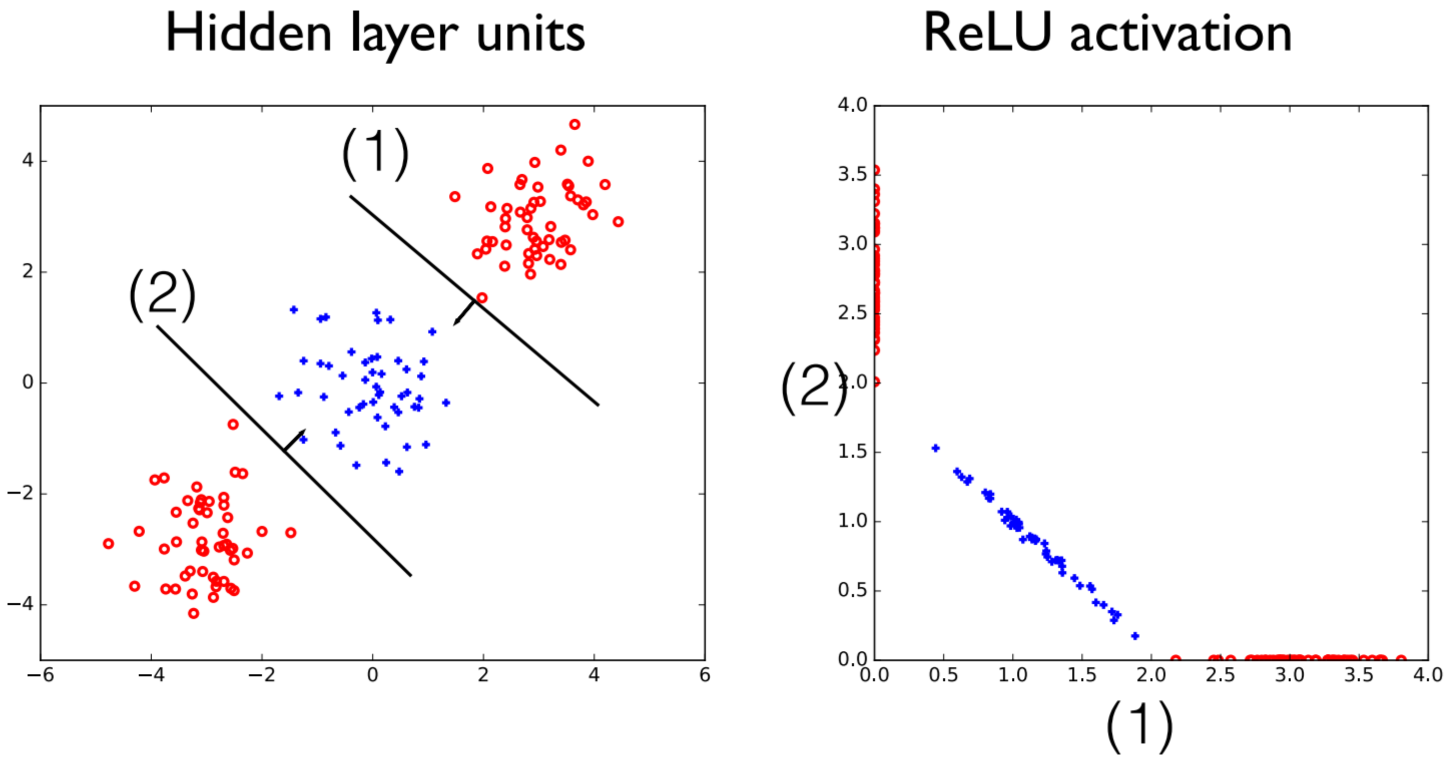


MITx: 6.86x

Feed Forward Neural Networks



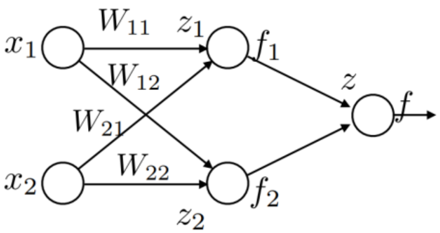
Simple model:

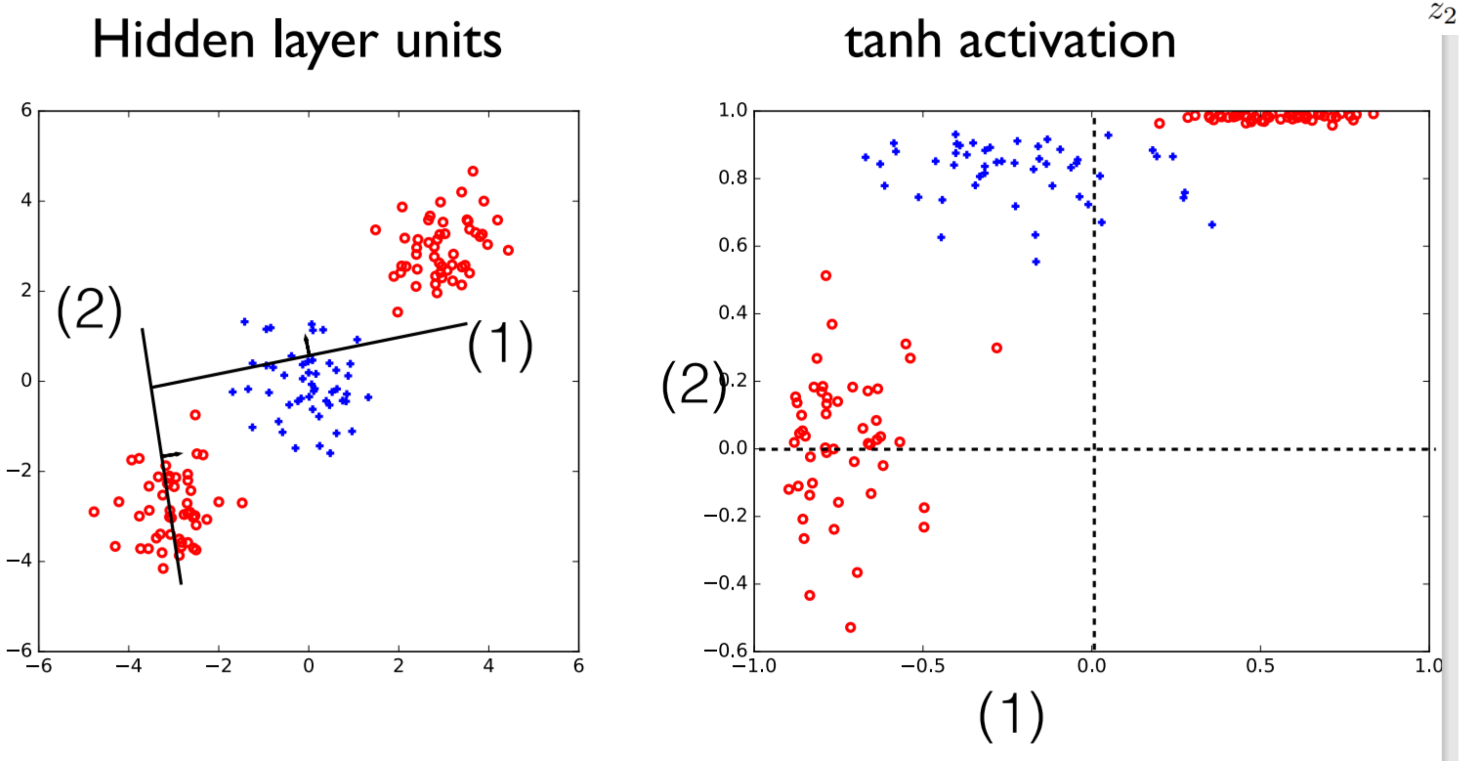


MITx: 6.86x

Feed Forward Neural Networks

Simple model:



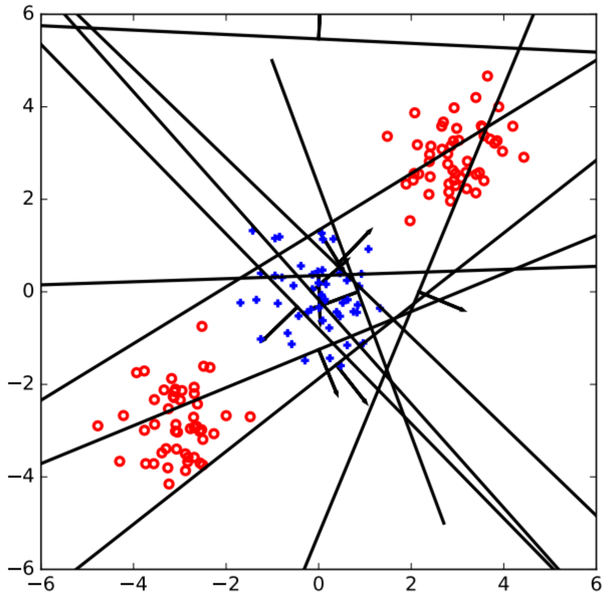


MITx: 6.86x

Feed Forward Neural Networks

Simple model:

Hidden layer units



(10 randomly chosen units)

Are the points
linearly separable
in the resulting
10 dimensional space?

YES!

MITx: 6.86x

Feed Forward Neural Networks

Summary:

- Units in neural networks are linear classifiers, just with different output non-linearity
- The units in feed-forward neural networks are arranged in layers (input, hidden,..., output)
- By learning the parameters associated with the hidden layer units, we learn how to represent examples (as hidden layer activations)
- The representations in neural networks are learned directly to facilitate the end-to-end task
- A simple classifier (output unit) suffices to solve complex classification tasks if it operates on the hidden layer representations

MITx: 6.86x

Feed Forward Neural Networks

Convolution Neural Networks (cnn)

Digit recognition so far:

Each pixel is a featur for our learning algorithm

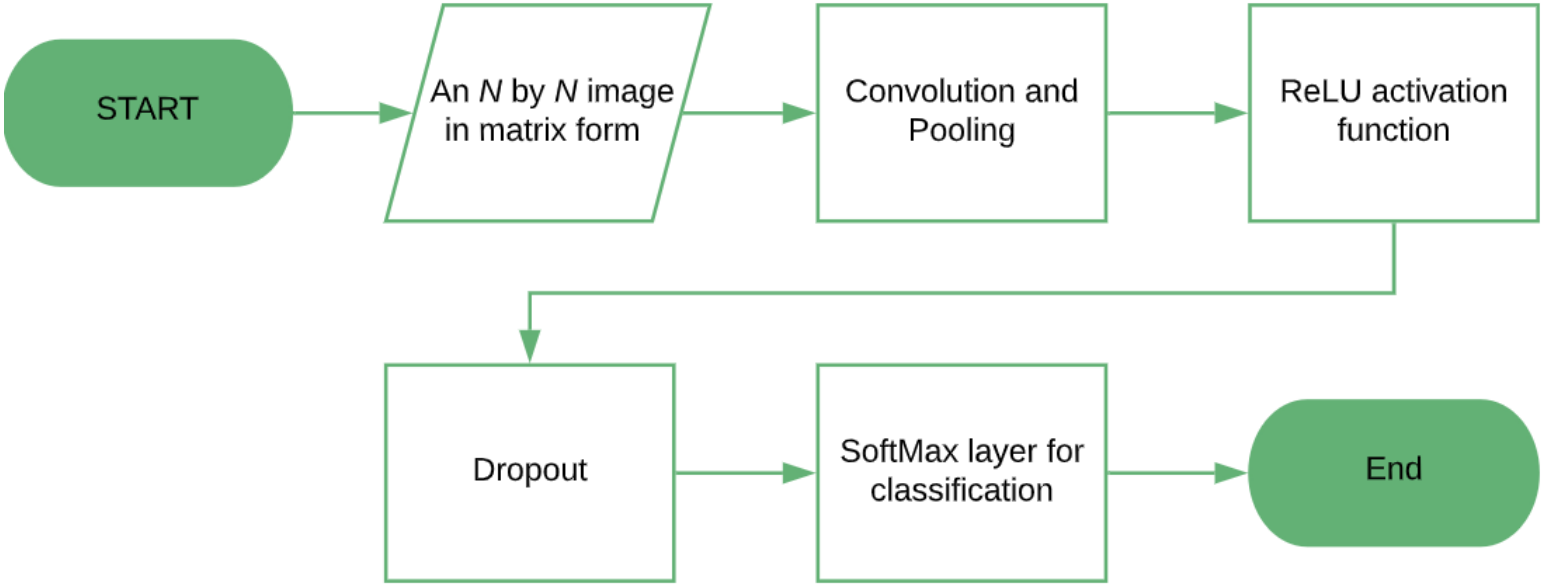
If we shift the whole number the algorithm would not
Recognize the number any more since now different
Pixels are black and white.

We neet a method that recognizes the features (numbers)
In the image independent on their position

MITx: 6.86x

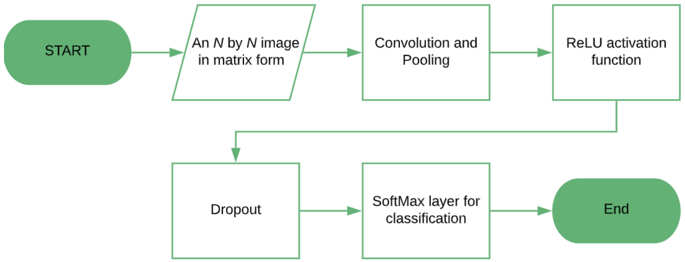
Feed Forward Neural Networks

Convolution Neural Networks (cnn)



[How ReLU and Dropout Layers Work in CNNs | Baeldung on Computer Science]

Feed Forward Neural Networks



Convolution Neural Networks (cnn)

Convolution layer:

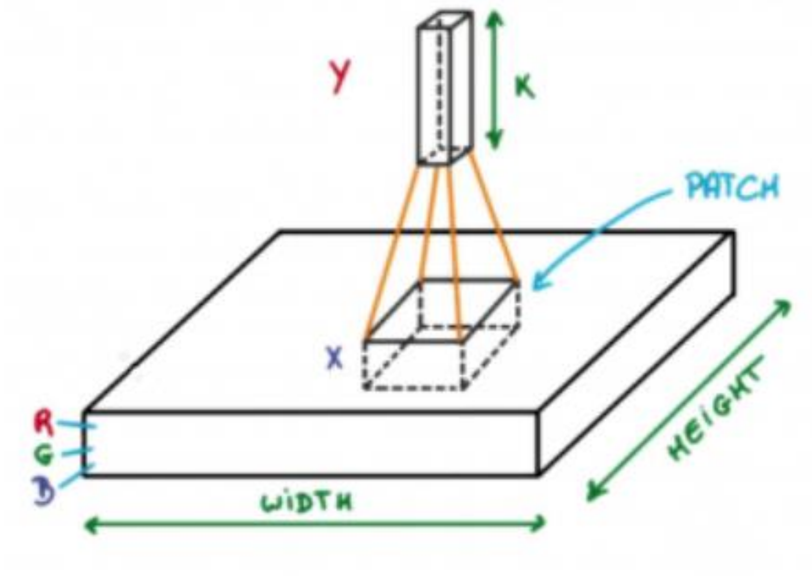


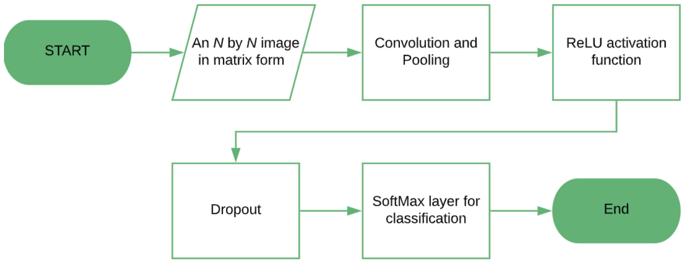
Image source: Deep Learning Udacity

- Learns features such as lines, edges, circles independent on position in image
- stable against pixel noise

[Introduction to Convolution Neural Network - GeeksforGeeks]

Feed Forward Neural Networks

Convolution Neural Networks (cnn)



MaxPool layer:

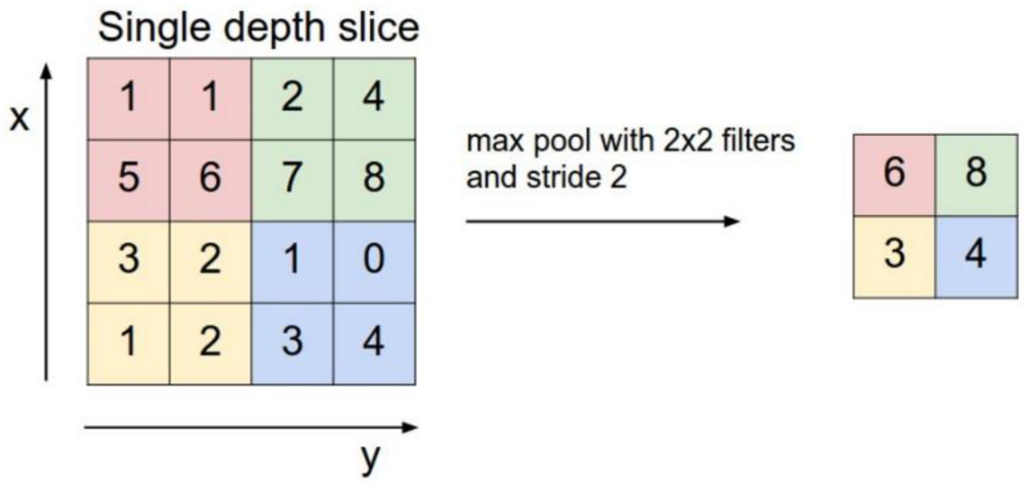


Image source: cs231n.stanford.edu

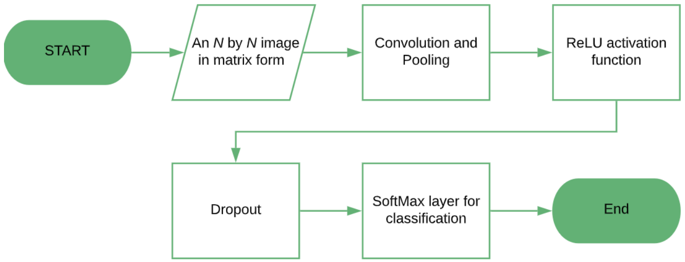
- Reduce dimesions -> speedup
- Avoid overfitting

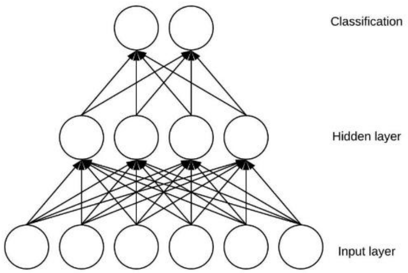
[Introduction to Convolution Neural Network - GeeksforGeeks]

Feed Forward Neural Networks

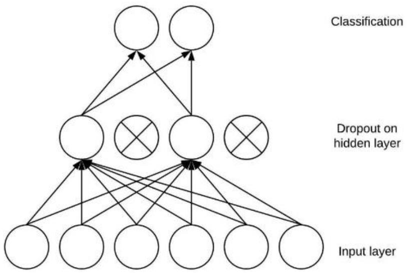
Convolution Neural Networks (cnn)

Dropout layer:





Without Dropout



With Dropout

- Avoid to high effect of features in first batches
- Avoid overfitting

[How ReLU and Dropout Layers Work in CNNs | Baeldung on Computer Science]()