Lecture 2

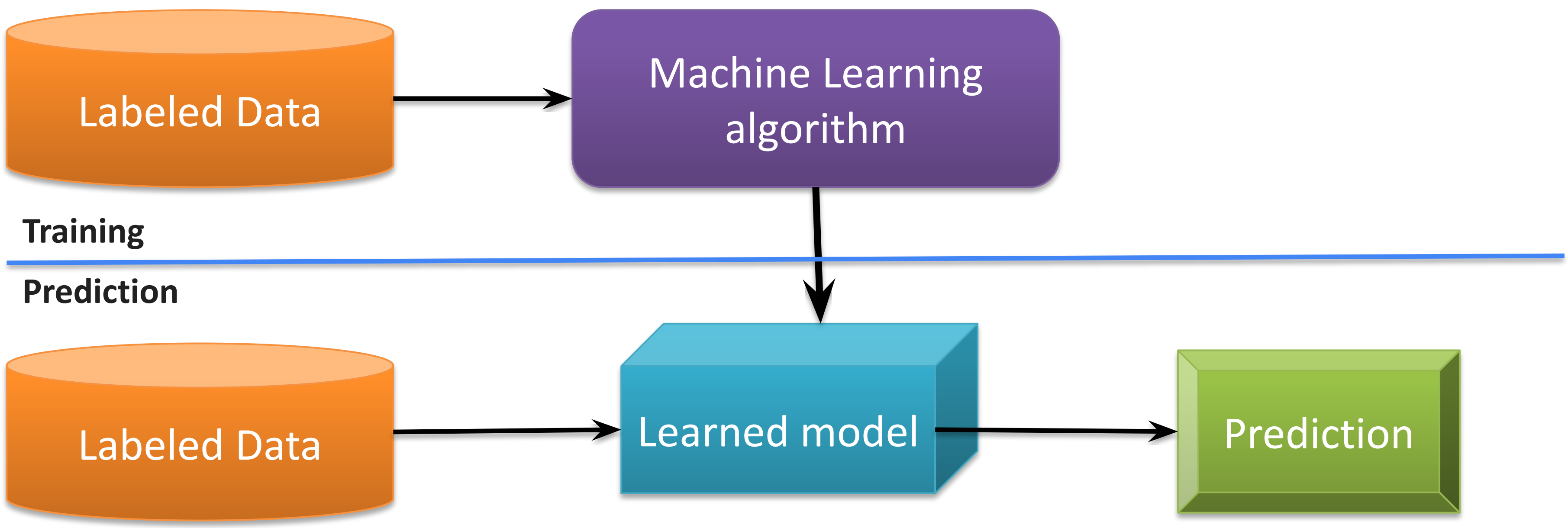
Deep Learning Overview

Lecture Outline

- Machine learning basics
  - Supervised and unsupervised learning
  - Linear and non-linear classification methods
- Introduction to deep learning
- Elements of neural networks (NNs)
  - Activation functions
- Training NNs
  - Gradient descent
  - Regularization methods
- NN architectures
  - Convolutional NNs
  - Recurrent NNs

Machine Learning Basics

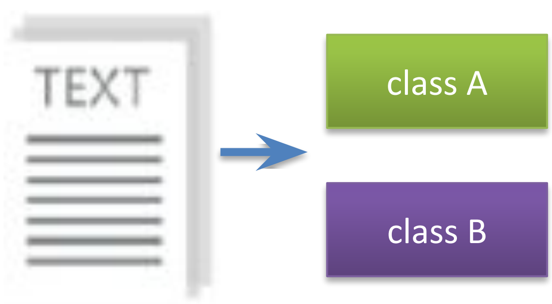
- ***Artificial Intelligence*** is a scientific field concerned with the development of algorithms that allow computers to learn without being explicitly programmed
- ***Machine Learning*** is a branch of Artificial Intelligence, which focuses on methods that learn from data and make predictions on unseen data



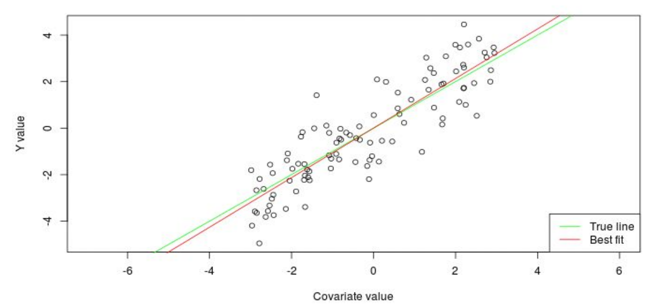
Picture from: Ismini Lourentzou – Introduction to Deep Learning

Machine Learning Types

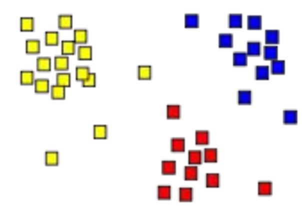
- ***Supervised***: learning with labeled data
  - Example: email classification, image classification
  - Example: regression for predicting real-valued outputs
- ***Unsupervised***: discover patterns in unlabeled data
  - Example: cluster similar data points
- ***Reinforcement learning***: learn to act based on feedback/reward
  - Example: learn to play Go



Classification



Regression



Clustering

Supervised Learning

- *Supervised learning* categories and techniques
  - **Numerical classifier functions**
    - Linear classifier, perceptron, logistic regression, support vector machines (SVM), neural networks
  - **Parametric (probabilistic) functions**
    - Naïve Bayes, Gaussian discriminant analysis (GDA), hidden Markov models (HMM), probabilistic graphical models
  - **Non-parametric (instance-based) functions**
    - *k*-nearest neighbors, kernel regression, kernel density estimation, local regression
  - **Symbolic functions**
    - Decision trees, classification and regression trees (CART)
  - **Aggregation (ensemble) learning**
    - Bagging, boosting (Adaboost), random forest

Slide credit: Y-Fan Chang – An Overview of Machine Learning

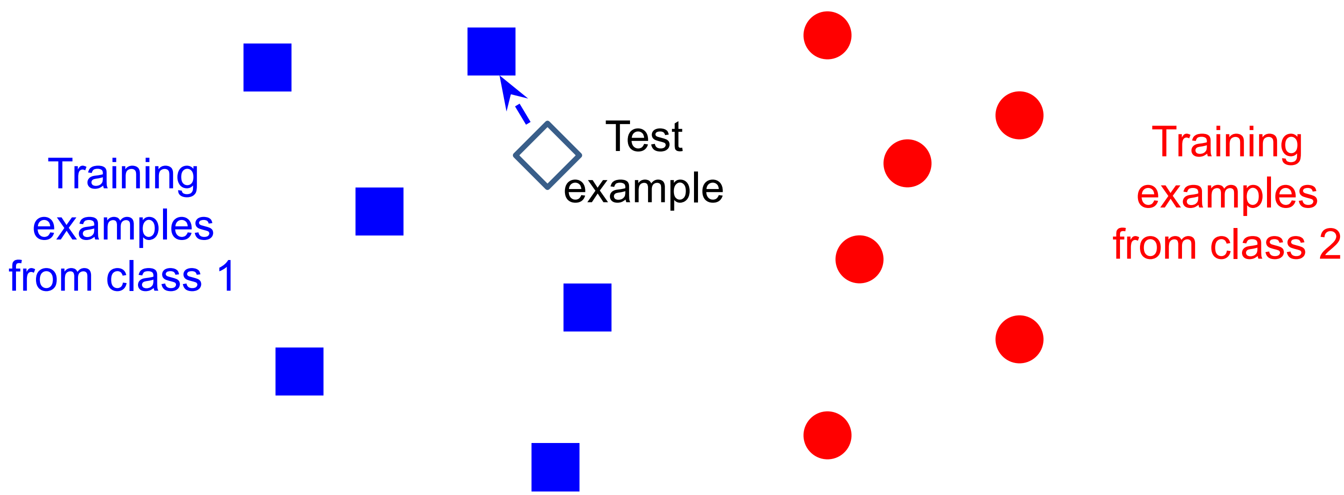
Unsupervised Learning

- ***Unsupervised learning*** categories and techniques
  - **Clustering**
    - *k*-means clustering
    - Mean-shift clustering
    - Spectral clustering
  - **Density estimation**
    - Gaussian mixture model (GMM)
    - Graphical models
  - **Dimensionality reduction**
    - Principal component analysis (PCA)
    - Factor analysis

Slide credit: Y-Fan Chang – An Overview of Machine Learning

Nearest Neighbor Classifier

- ***Nearest Neighbor*** – for each test data point, assign the class label of the nearest training data point
  - Adopt a distance function to find the nearest neighbor
    - Calculate the distance to each data point in the training set, and assign the class of the nearest data point (minimum distance)
  - It does not require learning a set of weights



Picture from: James Hays – Machine Learning Overview

Nearest Neighbor Classifier

- For image classification, the distance between all pixels is calculated (e.g., using ℓ_1 norm, or ℓ_2 norm)
  - Accuracy on CIFAR-10: 38.6%
- Disadvantages:
  - The classifier must remember all training data and store it for future comparisons with the test data
  - Classifying a test image is expensive since it requires a comparison to all training images

test image

56	32	10	18
90	23	128	133
24	26	178	200
2	0	255	220

-

training image

10	20	24	17
8	10	89	100
12	16	178	170
4	32	233	112

=

pixel-wise absolute value differences

46	12	14	1
82	13	39	33
12	10	0	30
2	32	22	108

→ 456

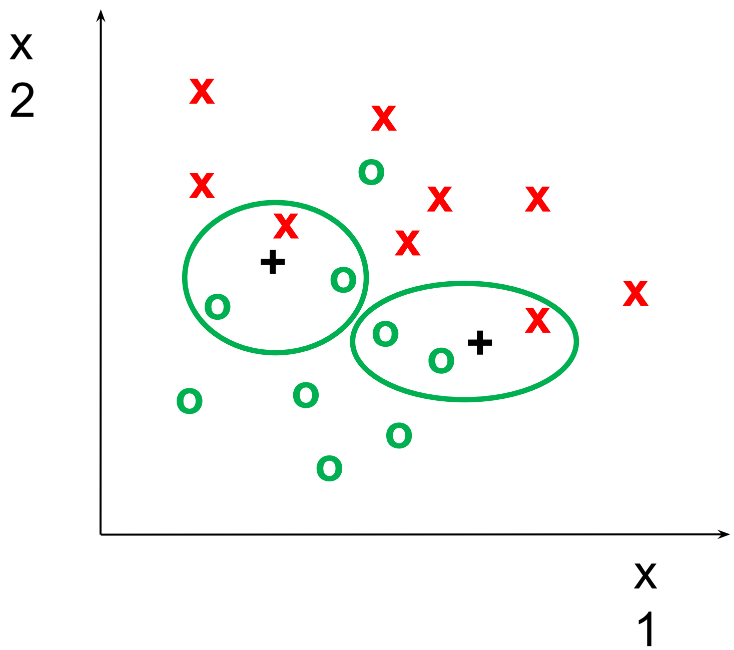
ℓ_1 norm
(Manhattan distance)

$$d_1(I_1, I_2) = \Sigma_p \, |I_1^p - I_2^p|$$

Picture from: https://cs231n.github.io/classification/

k-Nearest Neighbors Classifier

- ***k-Nearest Neighbors*** approach considers multiple neighboring data points to classify a test data point
  - E.g., 3-nearest neighbors
    - The test example in the figure is the + mark
    - The class of the test example is obtained by voting (based on the distance to the 3 closest points)



Picture from: James Hays – Machine Learning Overview

Linear Classifier

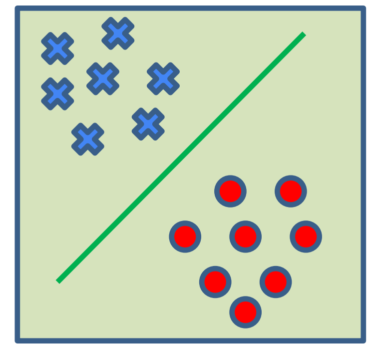
- ***Linear classifier***
  - Find a linear function f of the inputs x_i that separates the classes

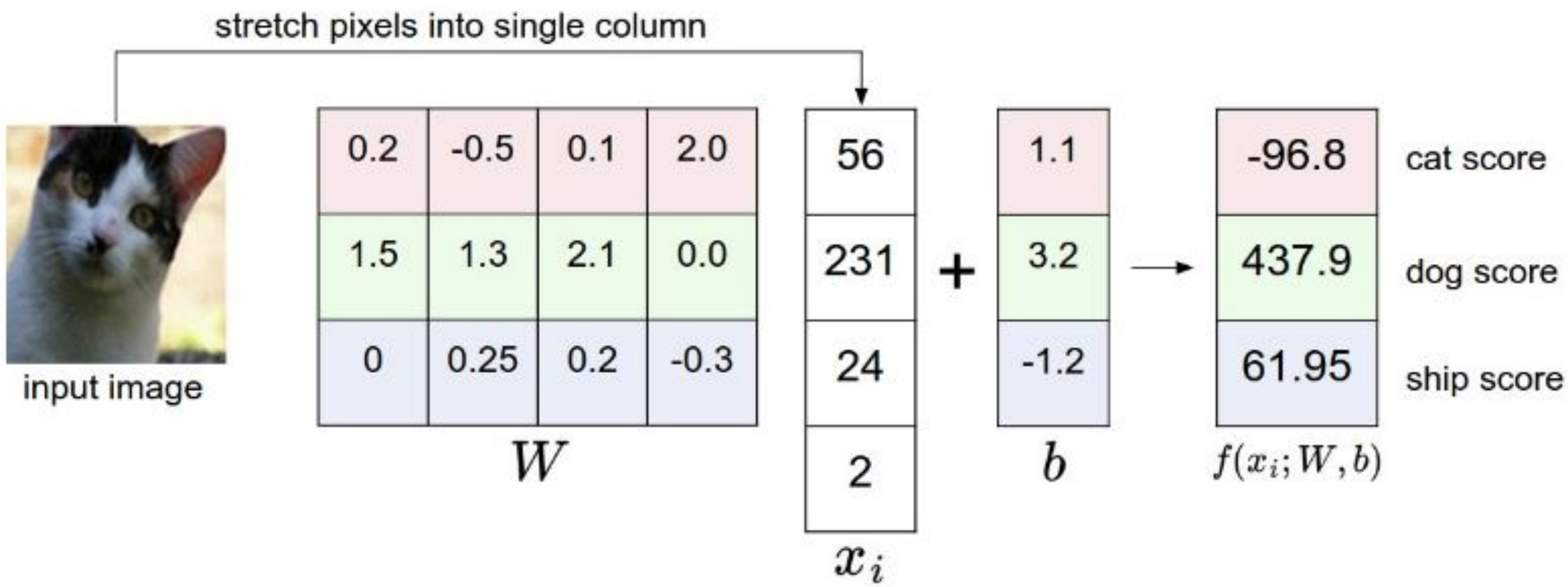
$$f(x_i, W, b) = Wx_i + b$$

  - Use pairs of inputs and labels to find the weights matrix W and the bias vector b
    - The weights and biases are the parameters of the function f
  - Several methods have been used to find the optimal set of parameters of a linear classifier
    - A common method of choice is the ***Perceptron*** algorithm, where the parameters are updated until a minimal error is reached (single layer, does not use backpropagation)
  - Linear classifier is a simple approach, but it is a building block of advanced classification algorithms, such as SVM and neural networks
    - Earlier multi-layer neural networks were referred to as multi-layer perceptrons (MLPs)

Linear Classifier

- The decision boundary is linear
  - A straight line in 2D, a flat plane in 3D, a hyperplane in 3D and higher dimensional space
- Example: classify an input image
  - The selected parameters in this example are not good, because the predicted cat score is low





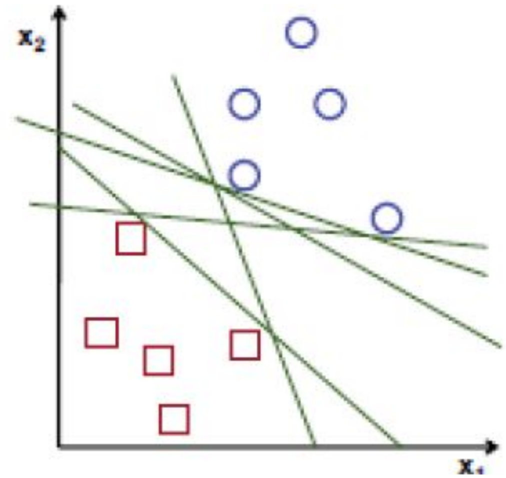
Picture from: https://cs231n.github.io/classification/

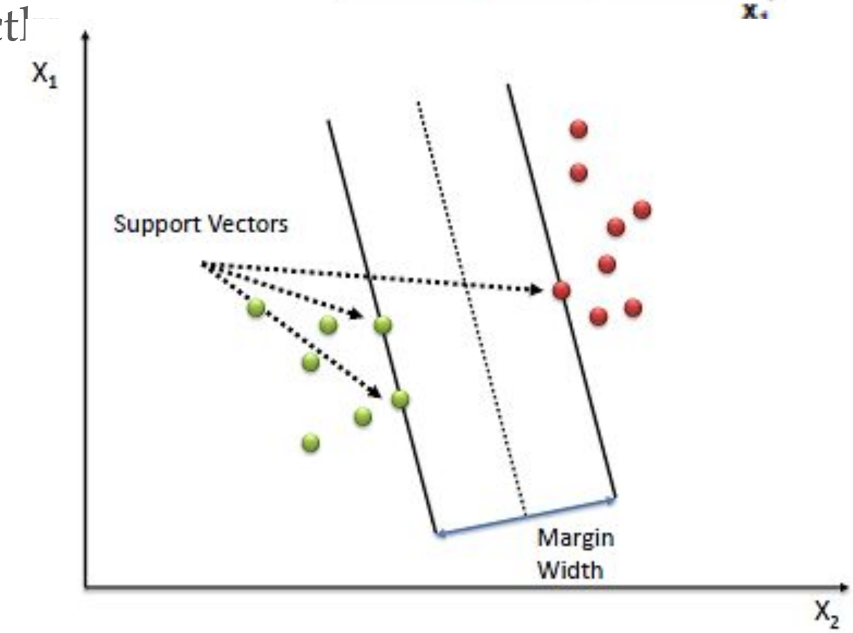
Support Vector Machines

- ***Support vector machines (SVM)***
  - How to find the best decision boundary?
    - All lines in the figure correctly separate the 2 classes
    - The line that is farthest from all training examples will have better generalization capabilities
  - SVM solves an optimization problem:
    - First, identify a decision boundary that correctl classifies the examples
    - Next, increase the geometric margin between the boundary and all examples
  - The data points that define the maximum margin width are called support vectors
  - Find W and b by solving:

$$\min \frac{1}{2}\|w\|^2$$

$$s.t.\ y_i(w \cdot x_i + b) \geq 1,\ \ \forall x_i$$



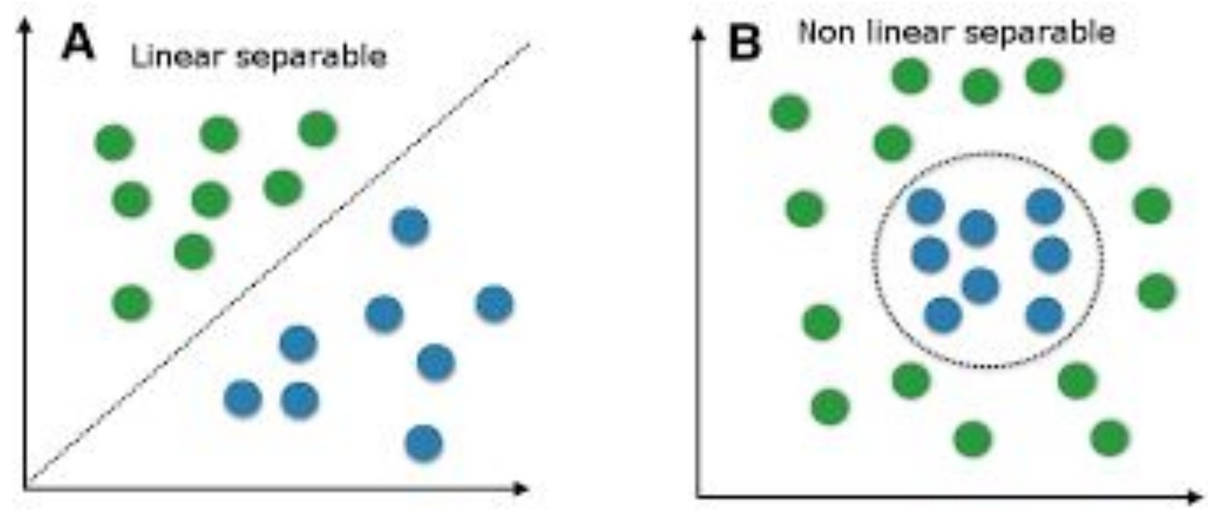


Linear vs Non-linear Techniques

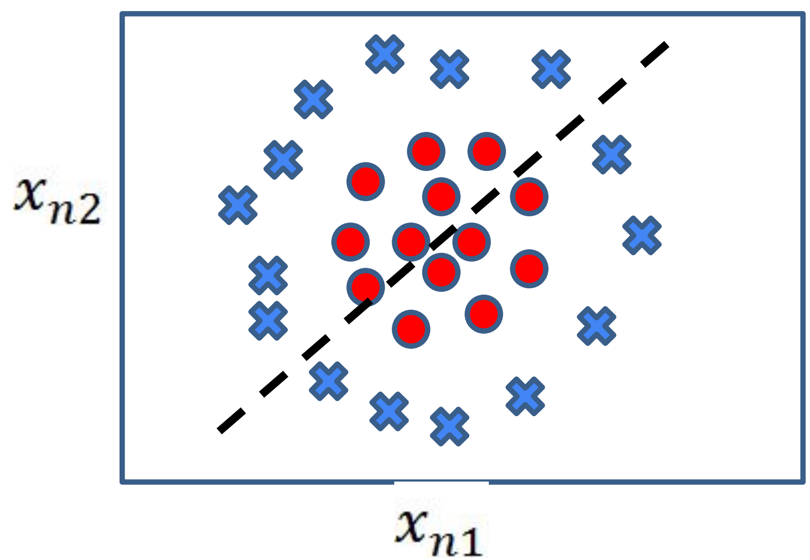
- Linear classification techniques
  - Linear classifier
  - Perceptron
  - Logistic regression
  - Linear SVM
  - Naïve Bayes
- Non-linear classification techniques
  - *k*-nearest neighbors
  - Non-linear SVM
  - Neural networks
  - Decision trees
  - Random forest

Linear vs Non-linear Techniques

- For some tasks, input data can be linearly separable, and linear classifiers can be suitably applied



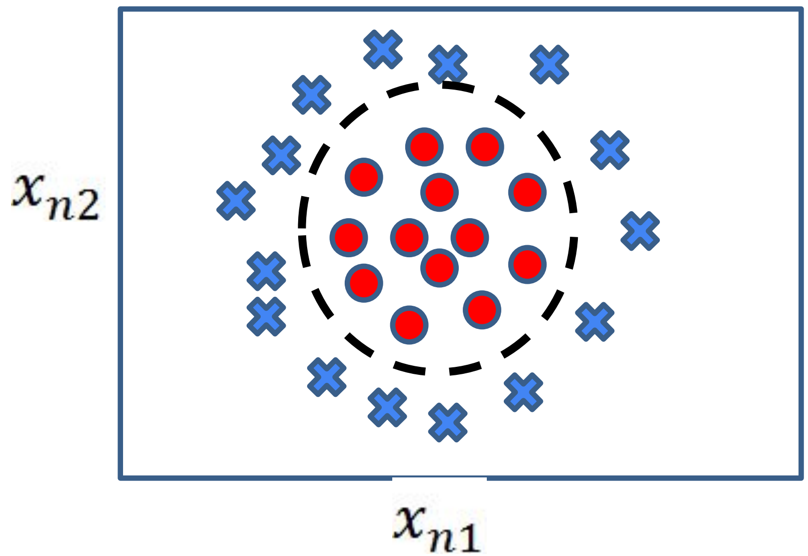
- For other tasks, linear classifiers may have difficulties to produce adequate decision boundaries



Picture from: Y-Fan Chang – An Overview of Machine Learning

Non-linear Techniques

- Non-linear classification
  - Features z_i are obtained as non-linear functions of the inputs x_i
  - It results in non-linear decision boundaries
  - Can deal with non-linearly separable data



Inputs: $x_i = [x_{n1} \quad x_{n2}]$

Features: $z_i = [x_{n1} \quad x_{n2} \quad x_{n1} \cdot x_{n2} \quad x_{n1}^2 \quad x_{n2}^2]$

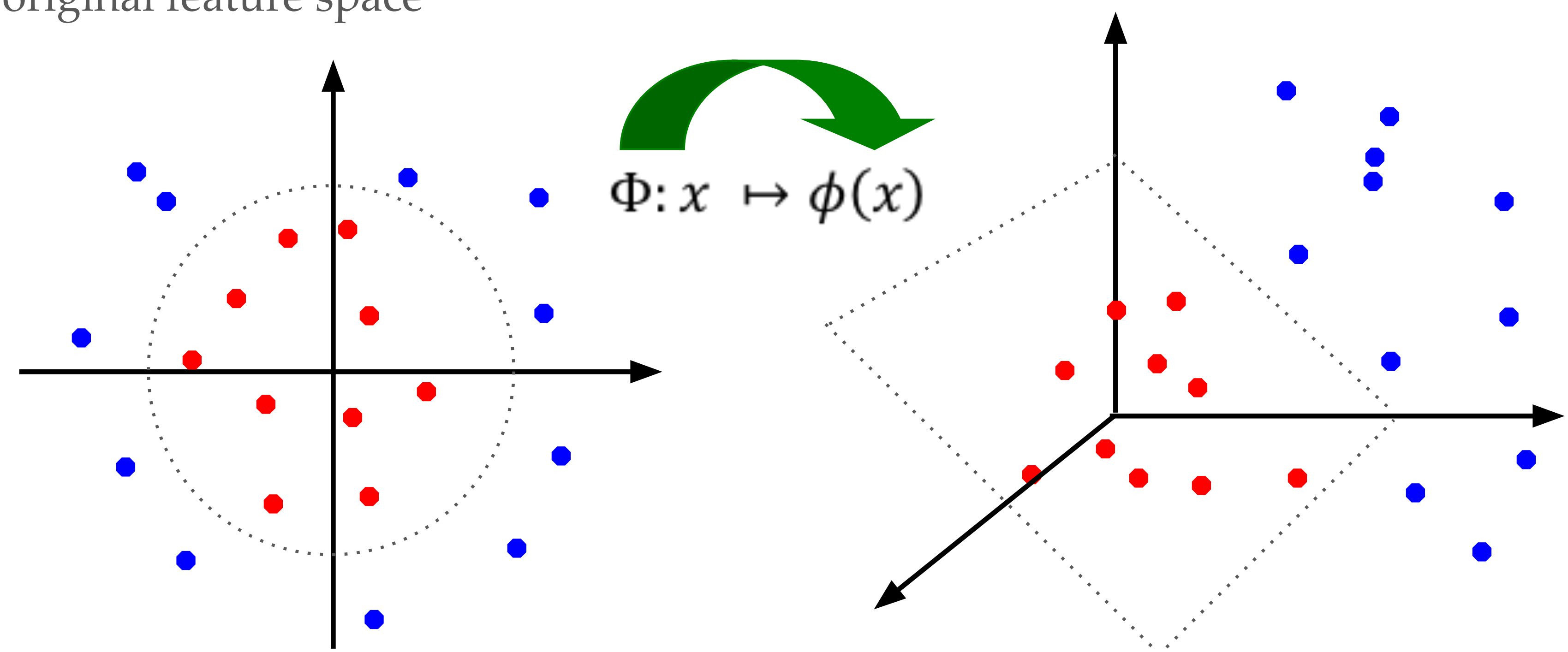
Outputs: $f(x_i, W, b) = Wz_i + b$

Picture from: Y-Fan Chang – An Overview of Machine Learning

Non-linear Support Vector Machines

Linear vs Non-linear Techniques

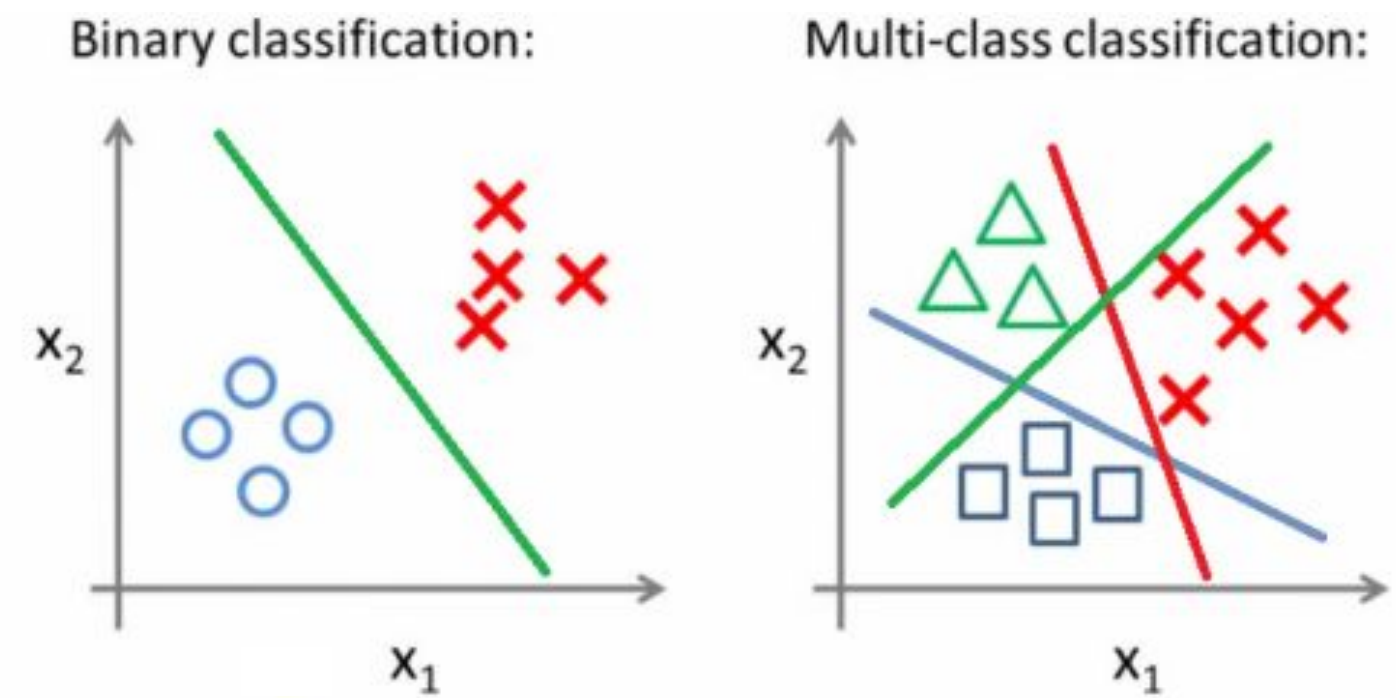
- ***Non-linear SVM***
  - The original input space is mapped to a higher-dimensional feature space where the training set is linearly separable
  - Define a non-linear kernel function to calculate a non-linear decision boundary in the original feature space



Picture from: James Hays – Machine Learning Overview

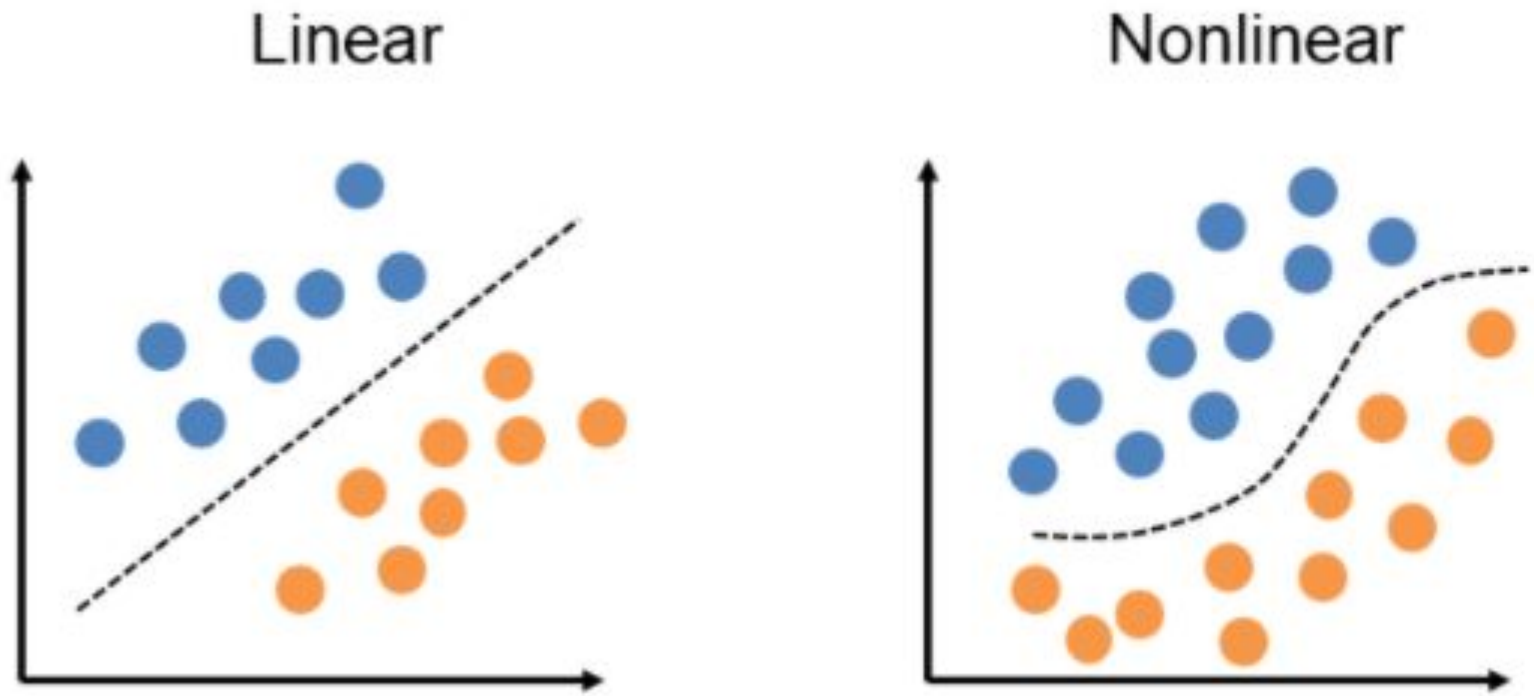
Binary vs Multi-class Classification

- A classification problem with only 2 classes is referred to as ***binary classification***
  - The output labels are 0 or 1
  - E.g., benign or malignant tumor, spam or no-spam email
- A problem with 3 or more classes is referred to as ***multi-class classification***



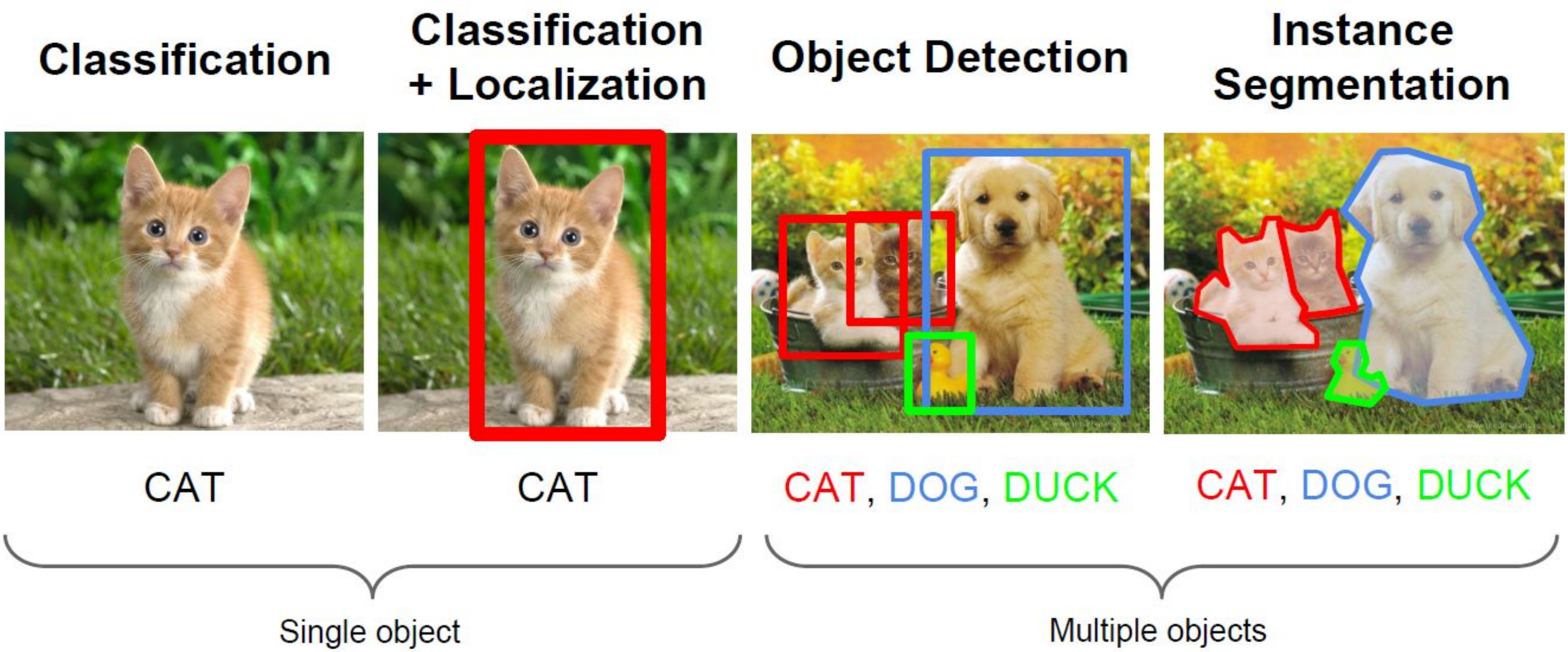
Binary vs Multi-class Classification

- Both the binary and multi-class classification problems can be linearly or non-linearly separated
  - Figure: linearly and non-linearly separated data for binary classification problem



Computer Vision Tasks

- Computer vision has been the primary area of interest for ML
- The tasks include: classification, localization, object detection, instance segmentation



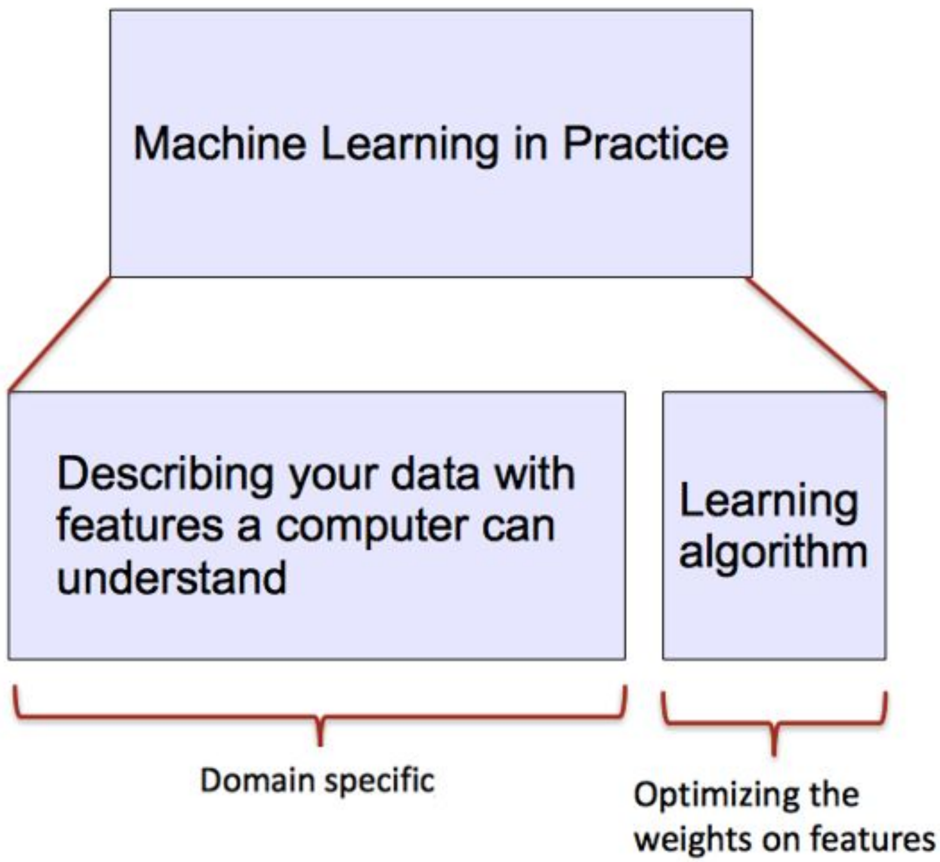
Picture from: Fie-Fei Li, Andrej Karpathy, Justin Johnson – Understanding and Visualizing CNNs

No-Free-Lunch Theorem

- Wolpert (2002) - The Supervised Learning No-Free-Lunch Theorems
- The derived classification models for supervised learning are simplifications of the reality
  - The simplifications are based on certain assumptions
  - The assumptions fail in some situations
    - E.g., due to inability to perfectly estimate ML model parameters from limited data
- In summary, ***No-Free-Lunch Theorem*** states:
  - **No single classifier works the best for all possible problems**
  - Since we need to make assumptions to generalize

ML vs. Deep Learning

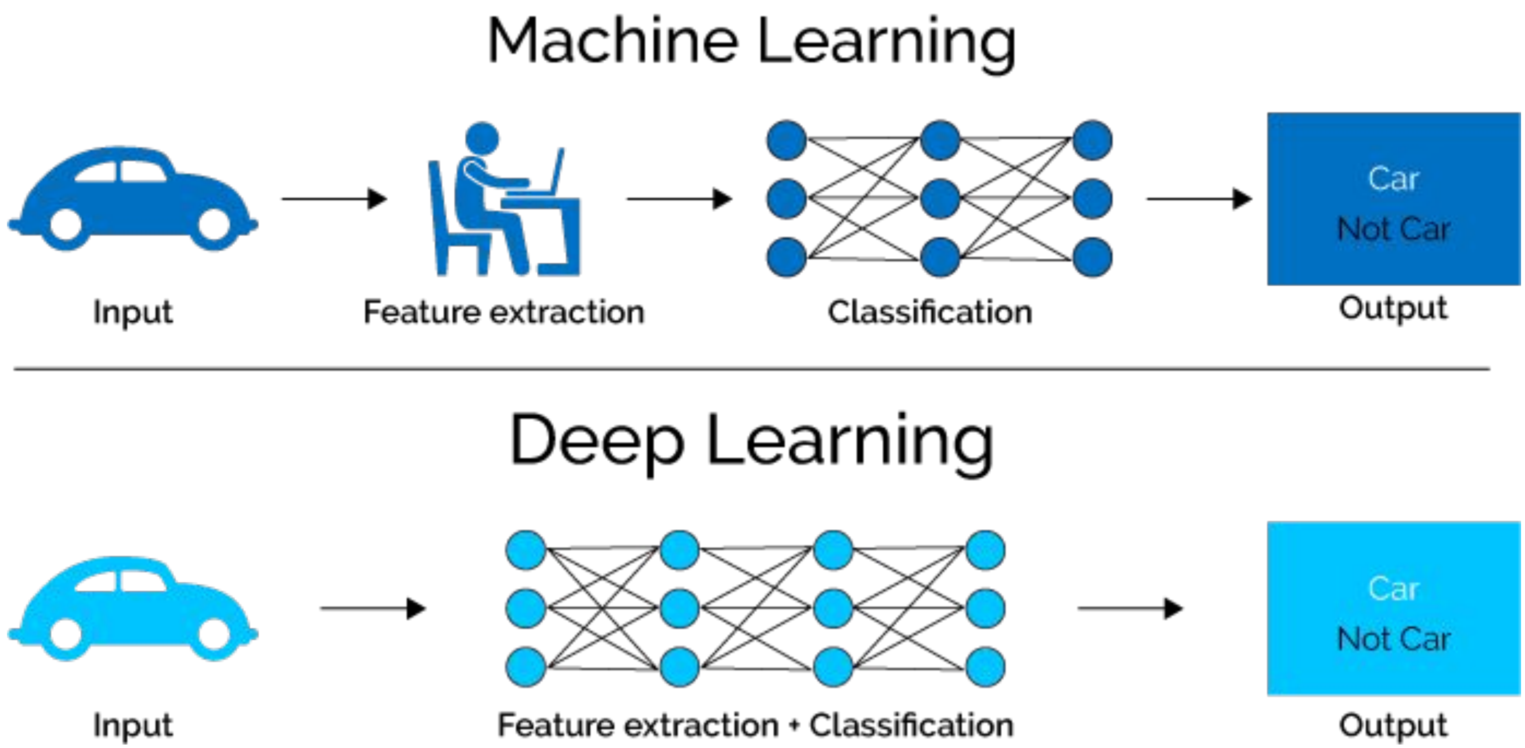
- Conventional machine learning methods rely on human-designed feature representations
  - ML becomes just optimizing weights to best make a final prediction



Picture from: Ismini Lourentzou – Introduction to Deep Learning

ML vs. Deep Learning

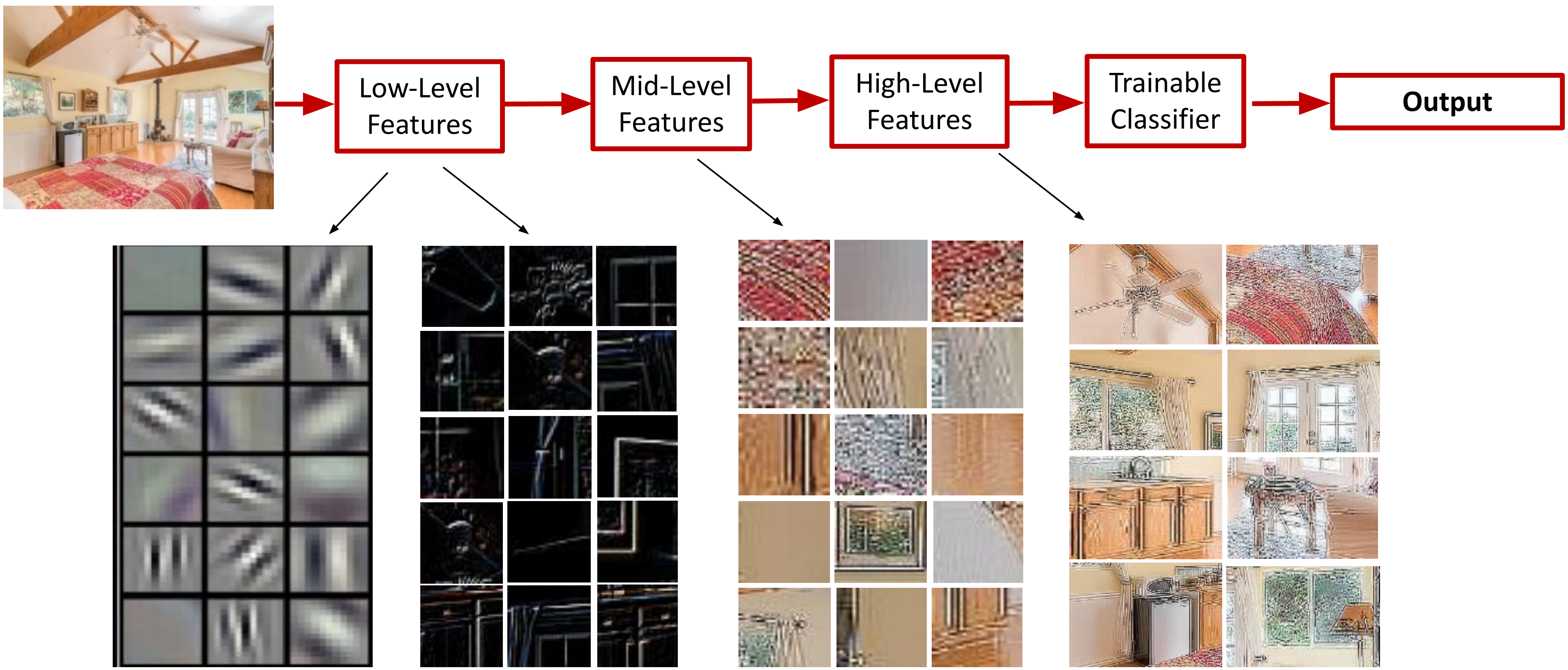
- ***Deep learning*** (DL) is a machine learning subfield that uses multiple layers for learning data representations
  - DL is exceptionally effective at learning patterns



Picture from: https://www.xenonstack.com/blog/static/public/uploads/media/machine-learning-vs-deep-learning.png

ML vs. Deep Learning

- DL applies a multi-layer process for learning rich hierarchical features (i.e., data representations)
  - Input image pixels → Edges → Textures → Parts → Objects



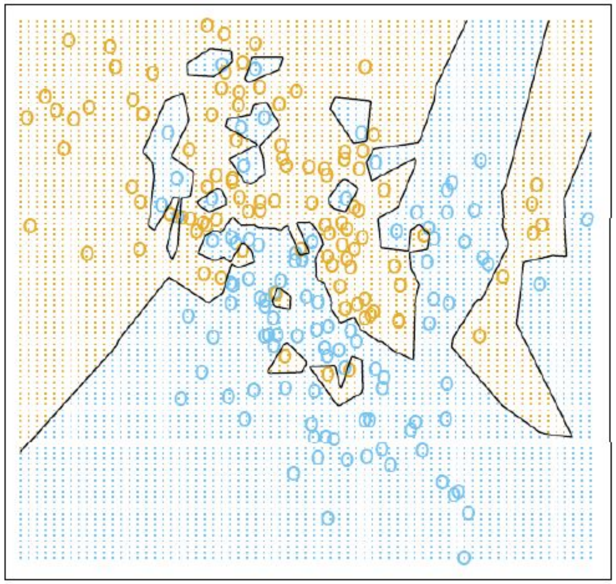
Slide credit: Param Vir Singh – Deep Learning

Why is DL Useful?

- DL provides a flexible, learnable framework for representing visual, text, linguistic information
  - Can learn in supervised and unsupervised manner
- DL represents an effective end-to-end learning system
- Requires large amounts of training data
- Since about 2010, DL has outperformed other ML techniques
  - First in vision and speech, then NLP, and other applications

Representational Power

- NNs with at least one hidden layer are universal approximators
  - Given any continuous function $h(x)$ and some $\epsilon > 0$, there exists a NN with one hidden layer (and with a reasonable choice of non-linearity) described with the function $f(x)$, such that $\forall x,\ |h(x) - f(x)| < \epsilon$
  - I.e., NN can approximate any arbitrary complex continuous function
- NNs use nonlinear mapping of the inputs x to the outputs $f(x)$ to compute complex decision boundaries
- But then, why use deeper NNs?
  - The fact that deep NNs work better is an empirical observation
  - Mathematically, deep NNs have the same representational power as a one-layer NN



- End