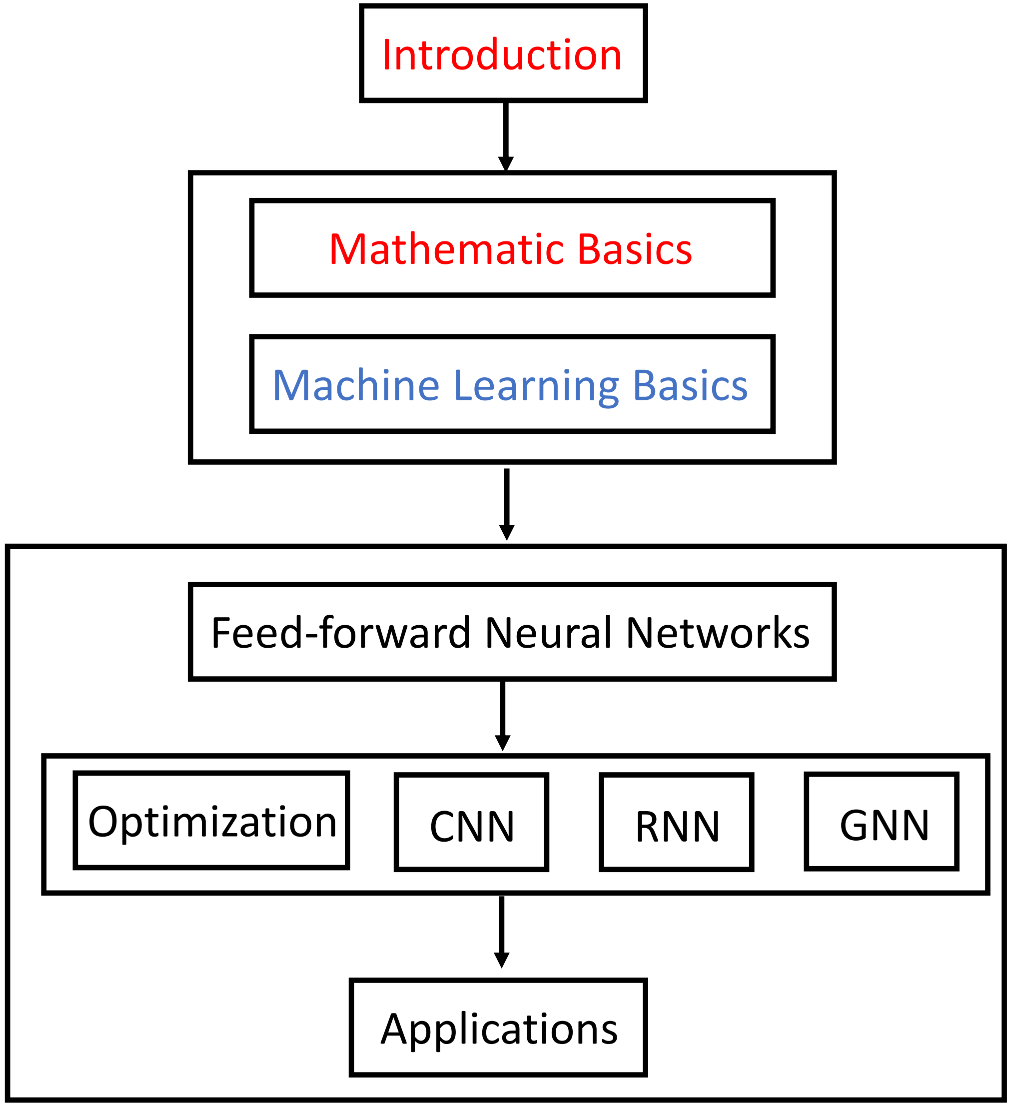
COSI 165B
Deep Learning

Chuxu Zhang
Computer Science Department
Brandeis University

2/3/2021

Structure of This Course



- A machine learning algorithm is an algorithm that is able to learn from data.

[Learning] A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

E * T = P

Experience * Task = Performance

Input Data:	Task:	Performance:
• Housing prices • Customer transactions • Clickstream data • Images	• Predict prices • Segment customers • Optimize user flows • Categorize images	• Accurate prices • Coherent groupings • KPI lifts • Correctly sorted images

Machine Learning

❑ Task T

Classification: vector to discrete value $f: \mathbb{R}^n \to \{1, \cdots, k\}$ (image classification, email spam)

Regression: vector to numerical value $f: \mathbb{R}^n \to \mathbb{R}$ (house price prediction, KPI prediction)

Machine translation: sequence to sequence (google translation)

Anomaly detection: unusual or atypical data (credit card fraud detection, system failure)

❑ Performance Measure P

Accuracy: the proportion of examples for which the model produces the correct output

Error rate: the proportion of examples for which the model produces an incorrect output

Mean square error (MSE): measures the average of the squares of the errors $\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$

Mean absolute error (MAE): measures the average of the absolute of errors $\text{MAE} = \frac{\sum_{i=1}^{n}|y_i - x_i|}{n}$

Using a test set of data that is separate from the data used for training

Machine Learning

❑ Experience E

Unsupervised learning: experience a dataset containing many features, then learn useful properties of the structure of this dataset (e.g., user clustering)

Supervised learning: experience a dataset containing features, but each example is also associated with a label or target. (e.g., image classification, house price prediction)

Semi-supervised learning, reinforcement learning, and so on.

Linear Regression

- Task T $f: \mathbb{R}^n \rightarrow \mathbb{R}$ $\quad \hat{y} = \boldsymbol{w}^\top \boldsymbol{x}$
- Measure P $\quad \mathrm{MSE}_{\text{test}} = \frac{1}{m} \sum_i (\hat{\boldsymbol{y}}^{(\text{test})} - \boldsymbol{y}^{(\text{test})})_i^2.$
- Experience (learning)

$$\nabla_{\boldsymbol{w}} \mathrm{MSE}_{\text{train}} = 0$$

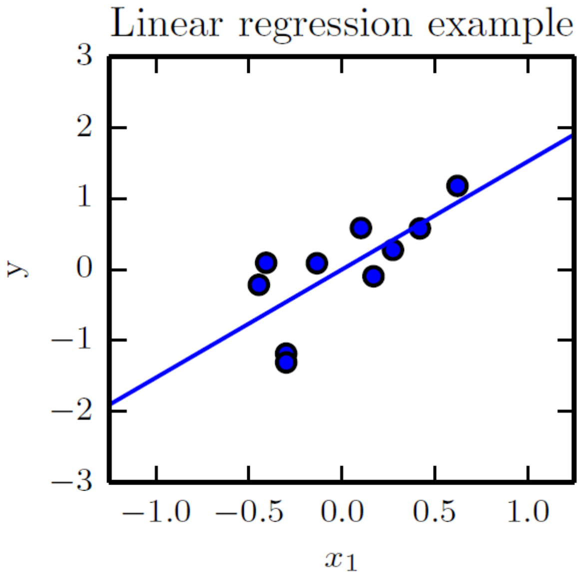
$$\Rightarrow \nabla_{\boldsymbol{w}} \frac{1}{m} ||\hat{\boldsymbol{y}}^{(\text{train})} - \boldsymbol{y}^{(\text{train})}||_2^2 = 0$$

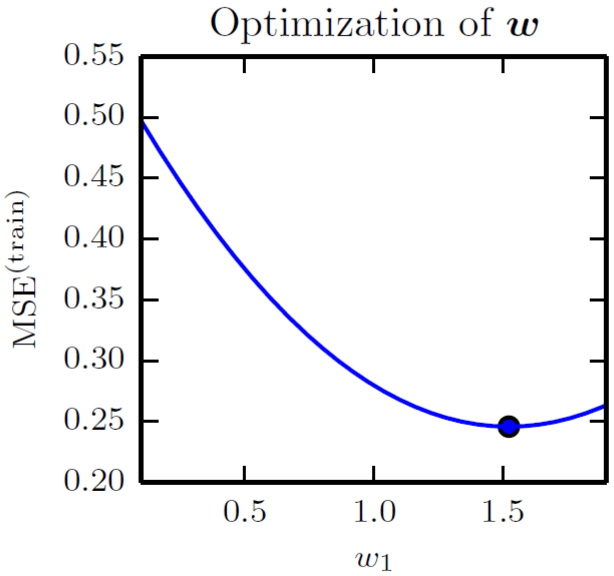
$$\Rightarrow \frac{1}{m} \nabla_{\boldsymbol{w}} ||\boldsymbol{X}^{(\text{train})}\boldsymbol{w} - \boldsymbol{y}^{(\text{train})}||_2^2 = 0$$

$$\Rightarrow \boldsymbol{w} = \left(\boldsymbol{X}^{(\text{train})\top}\boldsymbol{X}^{(\text{train})}\right)^{-1} \boldsymbol{X}^{(\text{train})\top}\boldsymbol{y}^{(\text{train})}$$

house price prediction

Living area (feet^2)	#bedrooms	Price (1000\$s)
2104	3	400
1600	3	330
2400	3	369
1416	2	232
3000	4	540
⋮	⋮	⋮





Capacity, Overfitting and Underfitting

Training error $\frac{1}{m^{(\text{train})}} \left\| \boldsymbol{X}^{(\text{train})} \boldsymbol{w} - \boldsymbol{y}^{(\text{train})} \right\|_2^2$

Test error (generalization error) $\frac{1}{m^{(\text{test})}} \left\| \boldsymbol{X}^{(\text{test})} \boldsymbol{w} - \boldsymbol{y}^{(\text{test})} \right\|_2^2$

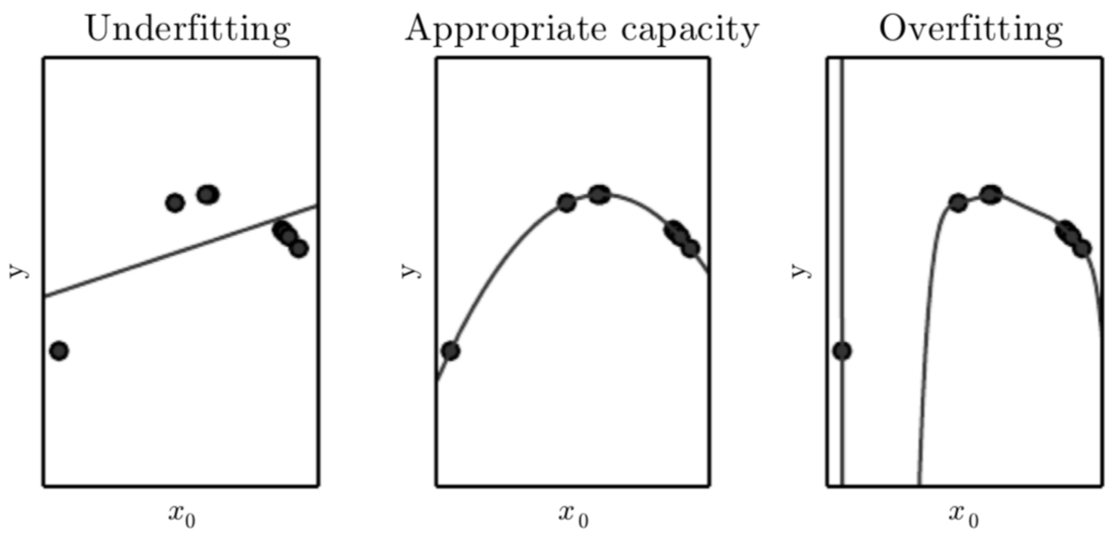
1. Make the training error small.
2. Make the gap between training and test error small.

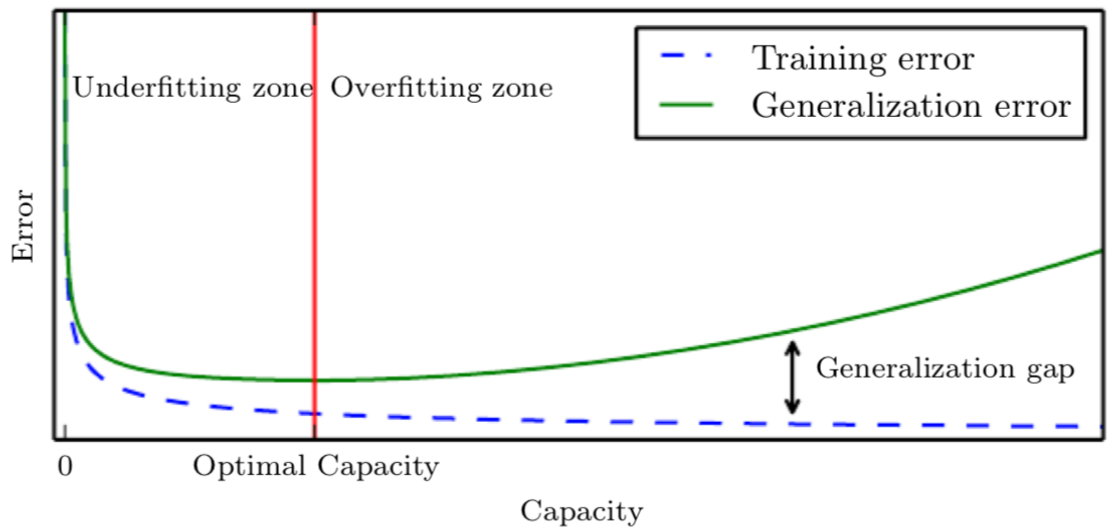
underfitting and *overfitting*. Underfitting occurs when the model is not able to obtain a sufficiently low error value on the training set. Overfitting occurs when the gap between the training error and test error is too large.

Generalization: ability to perform well on previously unobserved inputs
Capacity: ability to fit a wide variety of functions.
Control overfit or underfit: by altering model capacity
Low capacity: may struggle to fit the training set, underfitting
High capacity: memorizing properties of the training set that do not serve them well on the test set, overfitting

Capacity, Overfitting and Underfitting

- Polynomial regression



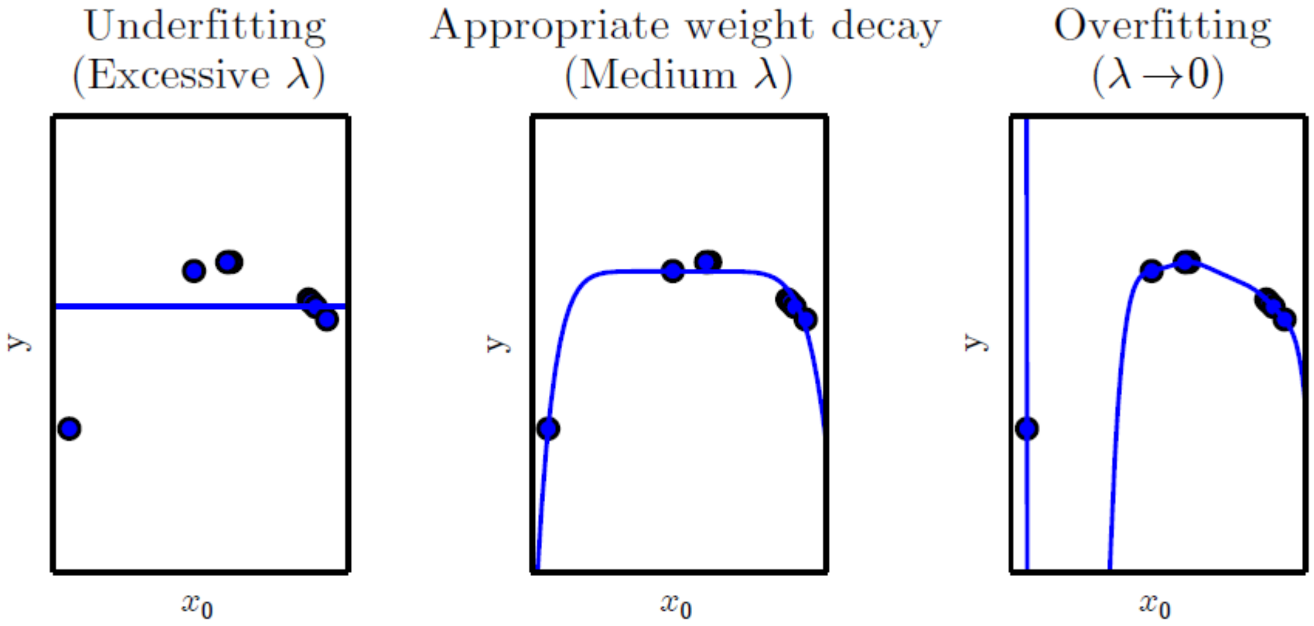


(left) linear function: underfitting. It cannot capture the curvature that is present in the data.
(middle) quadratic function: fit to the data, generalizes well to unseen points, no significant amount of overfitting or underfitting
(right) polynomial of degree 9: fit to the data perfectly while overfitting.

Underfitting: training error and generalization error are both high.
As we increase capacity, training error decreases, but the gap between training and generalization error increases.
Overfitting: the size of this gap outweighs the decrease in training error, where capacity is too large, above the optimal capacity.

❑ Regularization

$$J(w) = \mathrm{MSE}_{\text{train}} + \lambda w^\top w,$$



Larger λ, smaller weights, underfitting

Smaller λ, larger weights, overfitting

A choice of weights that make a tradeoff between underfitting and overfitting

Regularization: we make a learning algorithm that is intended to reduce generalization error but not training error

Hyperparameters and Validation Set

❑ Hyperparameters

Polynomial regression: the degree of the polynomial, which acts as a capacity hyperparameter. The λ value (regularization) is another example of a hyperparameter.

Can not determined by using training data. Why: overfitting

❑ Validation Set

Construct the validation set from the training data

Split the training data into two disjoint subsets: training set, validation set
One for model parameter optimization, the other for hyperparameter selection

80% training data for training, 20% for validation.

Maximum Likelihood Estimation

Linear Regression: Probabilistic Interpretation

$$h(x) = \sum_{i=0}^{d} \theta_i x_i = \theta^T x,$$

real data are generated with noise

$$y^{(i)} = \theta^T x^{(i)} + \epsilon^{(i)},$$

$$p(\epsilon^{(i)}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\epsilon^{(i)})^2}{2\sigma^2}\right)$$

conditional probability

$$p(y^{(i)}|x^{(i)};\theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right)$$

Likelihood with independent assumption

$$\begin{aligned} L(\theta) &= \prod_{i=1}^{n} p(y^{(i)} \mid x^{(i)}; \theta) \\ &= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right) \end{aligned}$$

$$\begin{aligned} \ell(\theta) &= \log L(\theta) \\ &= \log \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right) \\ &= \sum_{i=1}^{n} \log \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right) \\ &= n \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{\sigma^2} \cdot \frac{1}{2} \sum_{i=1}^{n} (y^{(i)} - \theta^T x^{(i)})^2. \end{aligned}$$

$$\frac{1}{2} \sum_{i=1}^{n} (y^{(i)} - \theta^T x^{(i)})^2,$$ (MSE Loss)

Others

- ❑ Estimators, Bias and Variance

- ❑ Bayesian Statistics

Deep Learning, Ian Goodfellow and Yoshua Bengio and Aaron Courville, MIT Press
https://www.deeplearningbook.org

Supervised Learning Algorithms

- Learning algorithms that learn to associate some input with some output, given a training set of examples of inputs x and outputs y.

- Linear regression

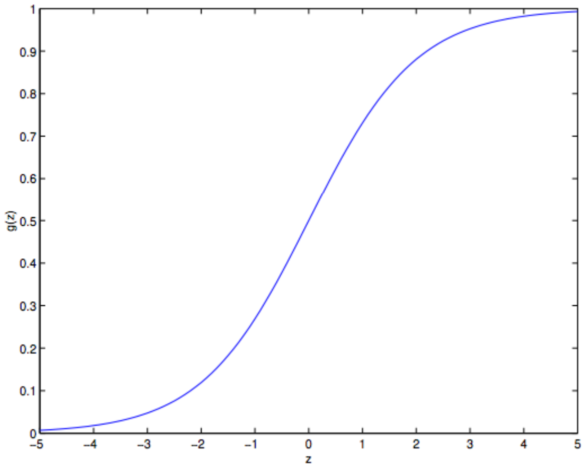
$$h(x) = \sum_{i=0}^{d} \theta_i x_i = \theta^T x,$$

Living area (feet^2)	#bedrooms	Price (1000\$s)
2104	3	400
1600	3	330
2400	3	369
1416	2	232
3000	4	540
⋮	⋮	⋮

- Logistic regression (numerical value 0 - 1)

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}},$$

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



Optimization

❑ Normal Equation

$$
\begin{aligned}
\frac{1}{2}(X\theta - \vec{y})^T(X\theta - \vec{y}) &= \frac{1}{2}\sum_{i=1}^{n}(h_\theta(x^{(i)}) - y^{(i)})^2 \\
&= J(\theta)
\end{aligned}
$$

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta \frac{1}{2}(X\theta - \vec{y})^T(X\theta - \vec{y}) \\
&= \frac{1}{2}\nabla_\theta \left((X\theta)^T X\theta - (X\theta)^T\vec{y} - \vec{y}^T(X\theta) + \vec{y}^T\vec{y}\right) \\
&= \frac{1}{2}\nabla_\theta \left(\theta^T(X^TX)\theta - \vec{y}^T(X\theta) - \vec{y}^T(X\theta)\right) \\
&= \frac{1}{2}\nabla_\theta \left(\theta^T(X^TX)\theta - 2(X^T\vec{y})^T\theta\right) \\
&= \frac{1}{2}\left(2X^TX\theta - 2X^T\vec{y}\right) \\
&= X^TX\theta - X^T\vec{y}
\end{aligned}
$$

$$\theta = (X^TX)^{-1}X^T\vec{y}.$$

Gradient Descent

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{n}(h_\theta(x^{(i)}) - y^{(i)})^2.$$

$$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta).$$

$$\begin{aligned}
\frac{\partial}{\partial\theta_j}J(\theta) &= \frac{\partial}{\partial\theta_j}\frac{1}{2}(h_\theta(x) - y)^2 \\
&= 2\cdot\frac{1}{2}(h_\theta(x) - y)\cdot\frac{\partial}{\partial\theta_j}(h_\theta(x) - y) \\
&= (h_\theta(x) - y)\cdot\frac{\partial}{\partial\theta_j}\left(\sum_{i=0}^{d}\theta_i x_i - y\right) \\
&= (h_\theta(x) - y)\,x_j
\end{aligned}$$

$$\theta_j := \theta_j + \alpha\left(y^{(i)} - h_\theta(x^{(i)})\right)x_j^{(i)}.$$

Take repeated steps in the opposite direction of the gradient, the direction of steepest descent

Batch gradient descent

Repeat until convergence {

$$\theta_j := \theta_j + \alpha\sum_{i=1}^{n}\left(y^{(i)} - h_\theta(x^{(i)})\right)x_j^{(i)},\ (\text{for every } j)$$

}

Stochastic Gradient Descent

Stochastic gradient descent

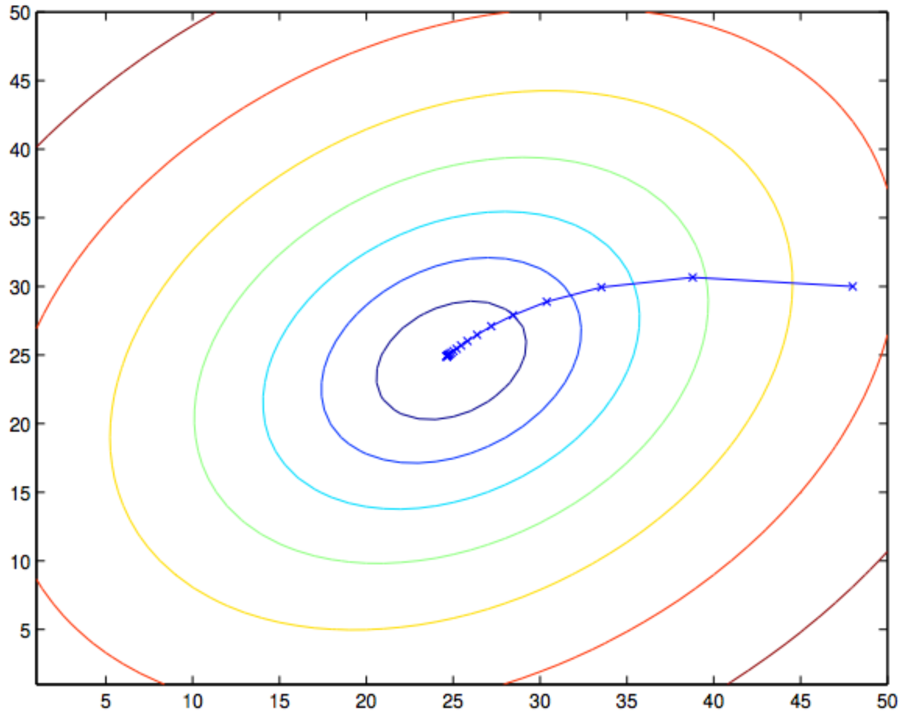
Loop {

  for $i = 1$ to n, {

$$\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)}) \right) x_j^{(i)}, \quad (\text{for every } j)$$

  }

}



contours of a quadratic function
trajectory taken by gradient descent

Supervised Learning Algorithms

- ❑ Softmax Regression
- ❑ Perceptron Learning
- ❑ Support Vector Machine
- ❑ And so on

Machine Learning | Andrew Ng: YouTube, Coursera

Unsupervised Learning Algorithms

- Extract information from a distribution that do not require human labor to annotate examples.

- Find the “best” representation of the data.

- Looking for a representation that preserves as much information about data as possible while obeying some penalty or constraint aimed at keeping the representation simpler or more accessible than data itself.

- Low-dimensional representations: compress as much information about x as possible in a smaller representation

- Sparse representations: representation whose entries are mostly zeroes for most inputs

- Independent representations: dimensions of the representation are statistically independent.

k-means clustering

In the clustering problem, we are given a training set $\{x^{(1)}, \ldots, x^{(n)}\}$, and want to group the data into a few cohesive "clusters." Here, $x^{(i)} \in \mathbb{R}^d$ as usual; but no labels $y^{(i)}$ are given. So, this is an unsupervised learning problem.

The k-means clustering algorithm is as follows:

1. Initialize **cluster centroids** $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^d$ randomly.
2. Repeat until convergence: {

   For every i, set

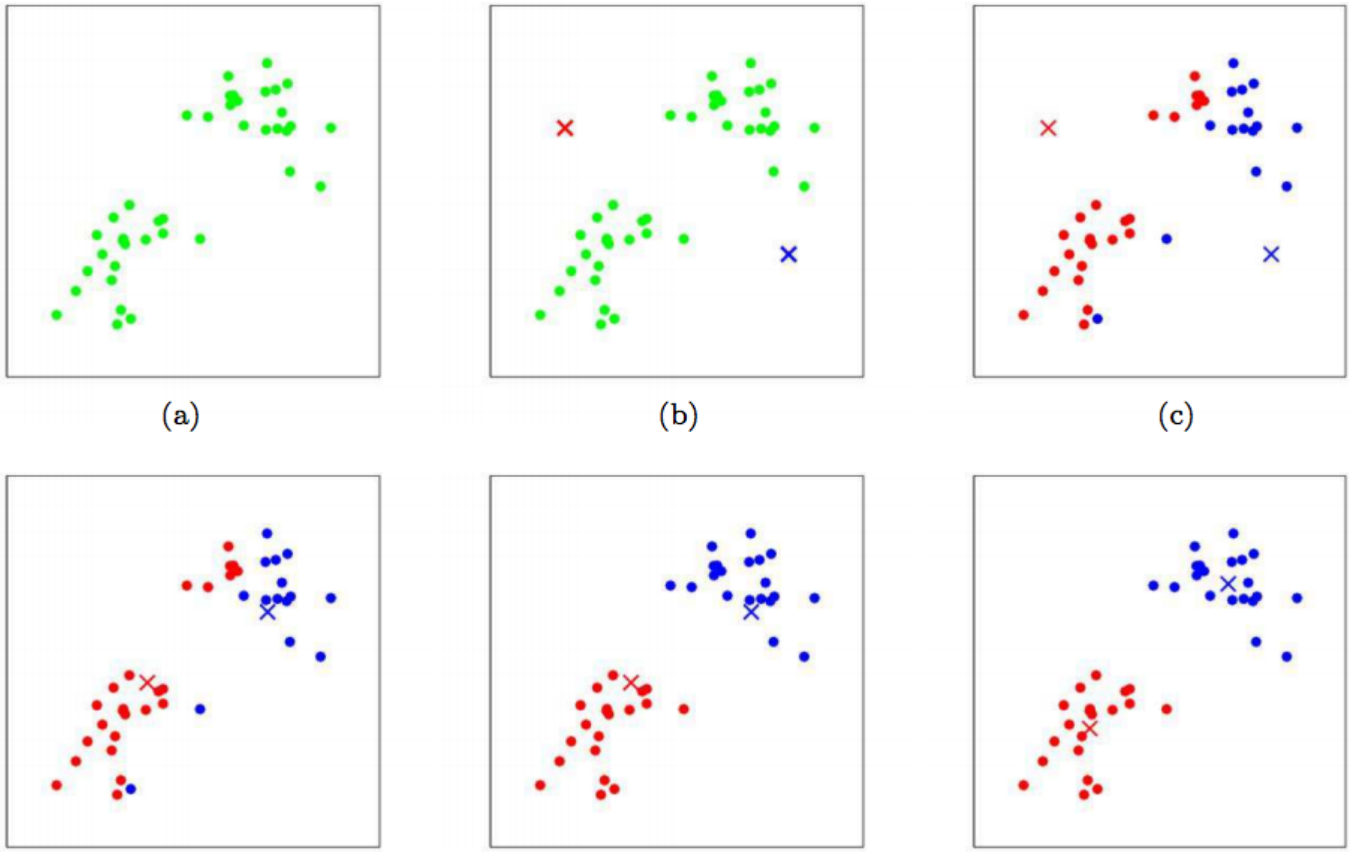
$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2.$$

   For each j, set

$$\mu_j := \frac{\sum_{i=1}^n 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^n 1\{c^{(i)} = j\}}.$$

   }

k-means clustering


(a) (b) (c) (d) (e) (f)

Random initial cluster centroids

assign each training example to the closest cluster centroid

move each cluster centroid to the mean of the points assigned to it

Unsupervised Learning Algorithms

- Principal Components Analysis

- EM Algorithm

- Gaussian mixture model

- And so on

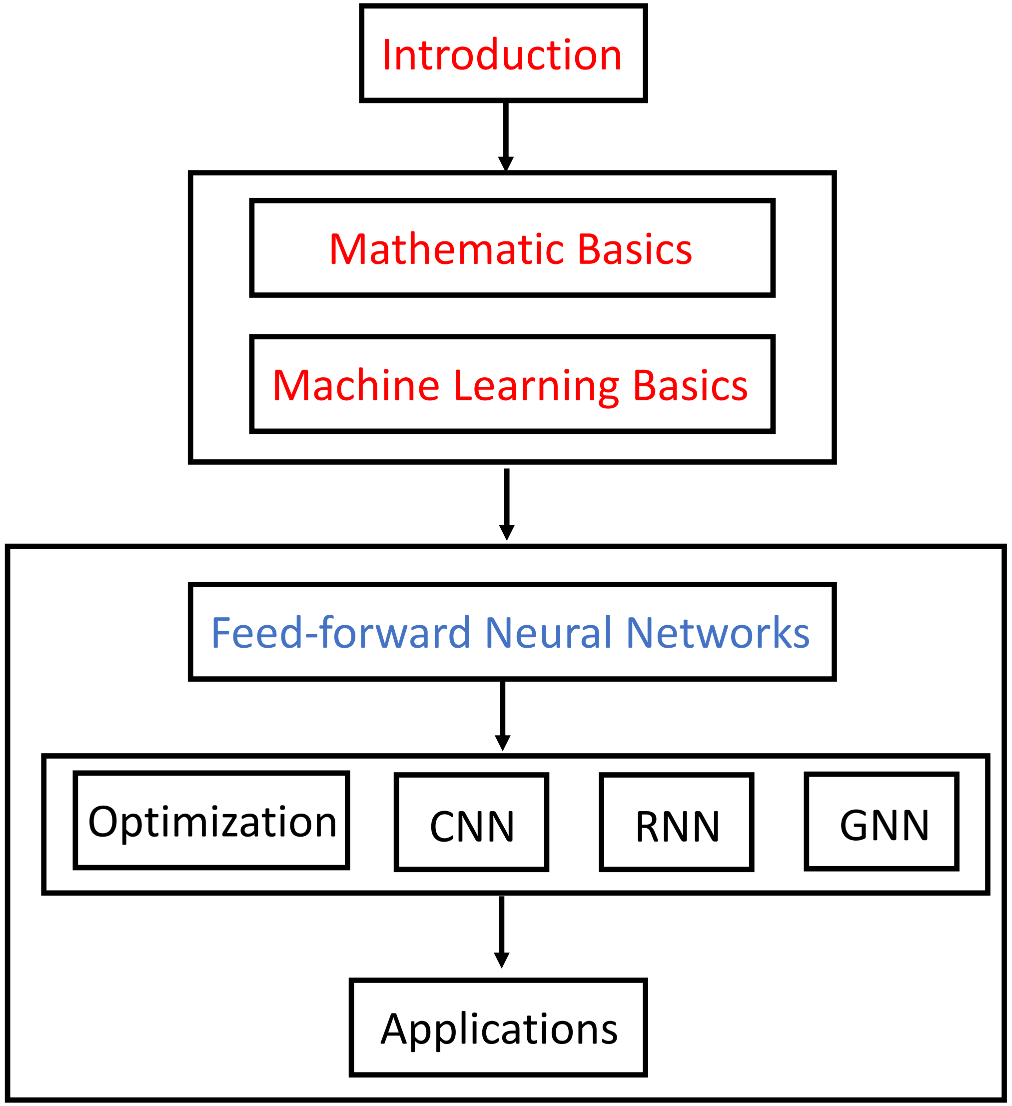
Machine Learning | Andrew Ng: YouTube, Coursera

Build Machine Learning Algorithms

- Dataset and Task (e.g., housing price prediction)
- Model (e.g., linear regression) $$h(x) = \sum_{i=0}^{d} \theta_i x_i = \theta^T x,$$
- Objective function (e.g., mean square error) $$\frac{1}{2} \sum_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)})^2$$
- Optimization Strategy (e.g., gradient descent)

$$\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)}) \right) x_j^{(i)}, \quad \text{(for every } j\text{)}$$

Structure of This Course



Q & A