Linear Regression

Spring 2019
Lect-02

What's on Menu Today?

- Introduction to ML
- Classification
- Regression
- Linear Regression
- Logistic Regression
- Reading Material
- Next Lecture outline
- Next Lecture Reading Material

Machine Learning

- **Machine learning** is the subfield of computer science that gives computers the ability to learn without being explicitly programmed (Arthur Samuel, 1959)
- What we do in Machine Learning?
  - Making predictions or
    - decisions from Data.

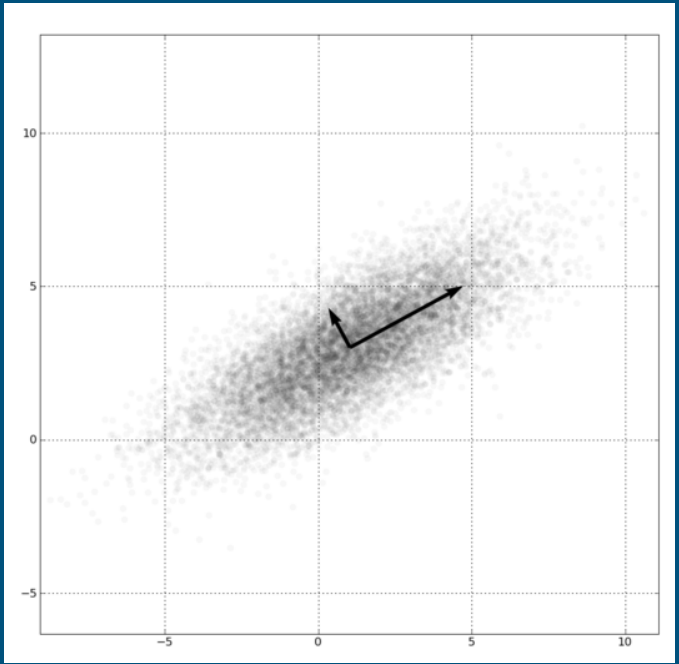
	Supervised Learning	Unsupervised Learning
Discrete	classification or categorization	clustering
Continuous	regression	dimensionality reduction

Dimensionality Reduction

- **PCA**, ICA, LLE, Isomap

- PCA is the most important technique to know. It takes advantage of correlations in data dimensions to produce the best possible lower dimensional representation, according to reconstruction error.

- PCA should be used for dimensionality reduction, not for discovering patterns or making predictions. Don't try to assign semantic meaning to the bases.



Machine Learning Problems

	Supervised Learning	***Unsupervised Learning***
Discrete	classification or categorization	$\boxed{\text{clustering}}$
Continuous	regression	dimensionality reduction

Why do we cluster?

- **Summarizing data**
  - Look at large amounts of data
  - Patch-based compression or denoising
  - Represent a large continuous vector with the cluster number
- **Counting**
  - Histograms of texture, color, SIFT vectors
- **Segmentation**
  - Separate the image into different regions
- **Prediction**
  - Images in the same cluster may have the same labels

How do we cluster?

- K-means
  - Iteratively re-assign points to the nearest cluster center
- Agglomerative clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters
- Mean-shift clustering
  - Estimate modes of pdf
- Spectral clustering
  - Split the nodes in a graph based on assigned links with similarity weights

Clustering for Summarization

Goal: cluster to minimize variance in data given clusters

- Preserve information

Cluster center

Data

$$\mathbf{c}^*, \boldsymbol{\delta}^* = \underset{\mathbf{c}, \boldsymbol{\delta}}{\operatorname{argmin}} \frac{1}{N} \sum_{j}^{N} \sum_{i}^{K} \delta_{ij} \left(\mathbf{c}_i - \mathbf{x}_j \right)^2$$

Whether x_j is assigned to c_i

Slide: Derek Hoiem

K-means algorithm

1. Randomly select K centers

2. Assign each point to nearest center

3. Compute new center (mean) for each cluster

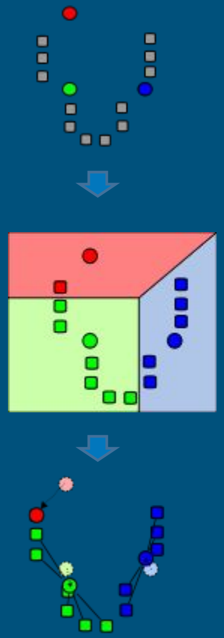
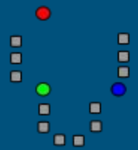


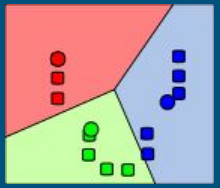
Illustration: http://en.wikipedia.org/wiki/K-means_clustering

K-means algorithm

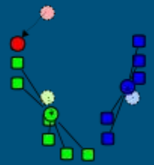
1. Randomly select K centers



2. Assign each point to nearest center



3. Compute new center (mean) for each cluster



Back to 2

Illustration: http://en.wikipedia.org/wiki/K-means_clustering

K-means

1. Initialize cluster centers: $\mathbf{c}^0$; t=0

2. Assign each point to the closest center

$$\boldsymbol{\delta}^t = \underset{\boldsymbol{\delta}}{\operatorname{argmin}} \frac{1}{N} \sum_j^N \sum_i^K \delta_{ij} \left(\mathbf{c}_i^{t-1} - \mathbf{x}_j \right)^2$$

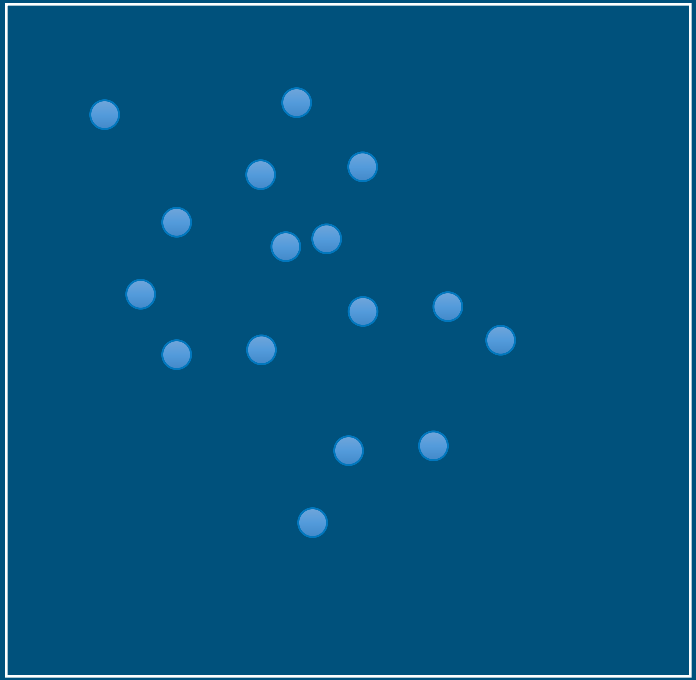
3. Update cluster centers as the mean of the points

$$\mathbf{c}^t = \underset{\mathbf{c}}{\operatorname{argmin}} \frac{1}{N} \sum_j^N \sum_i^K \delta_{ij}^t \left(\mathbf{c}_i - \mathbf{x}_j \right)^2$$

4. Repeat 2-3 until no points are re-assigned (t=t+1)

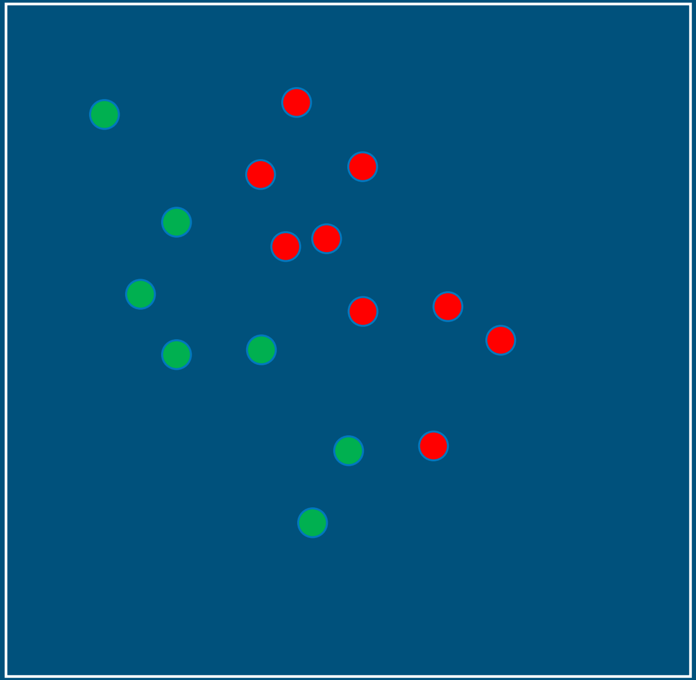
Supervised Learning

- Data consists
  - Input-output pairs
- Input
  - data points
  - features
  - covariates
- Output
  - labels
  - targets
  - variates



Supervised Learning

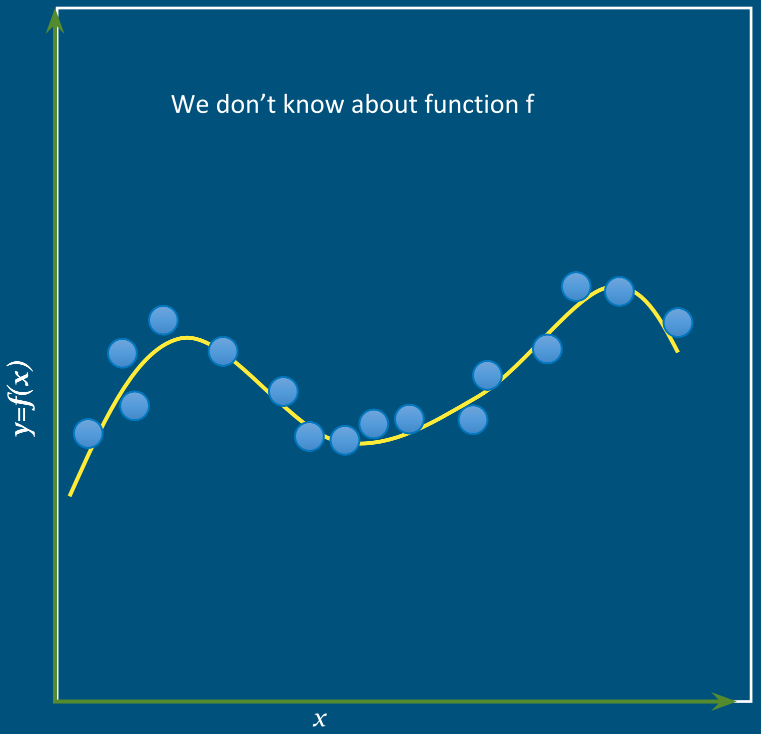
- Data consists
  - Input-output pairs
- Input
  - data points
  - features
- Output
  - labels
  - targets
  - variates



Classification

Supervised Learning

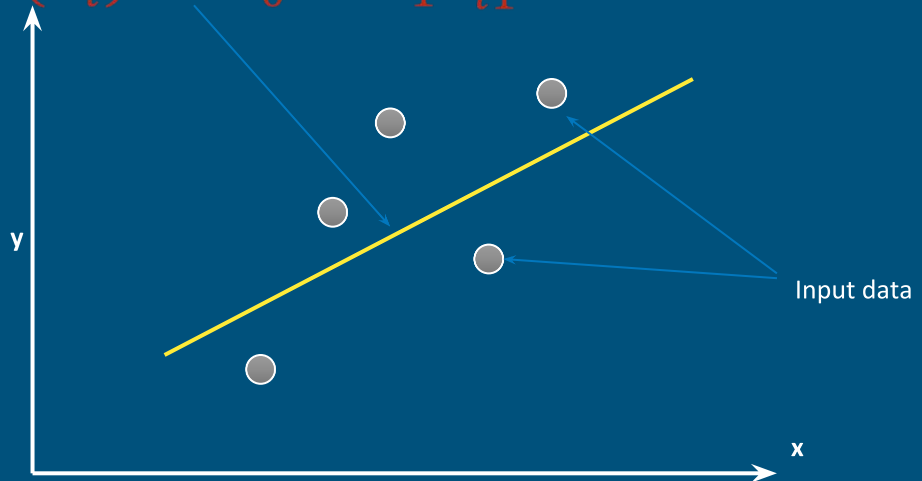
- Data consists
  - Input-output pairs
- Input
  - data points
  - features
  - covairates
- Output
  - labels
  - targets
  - variates



regression

Linear Regression

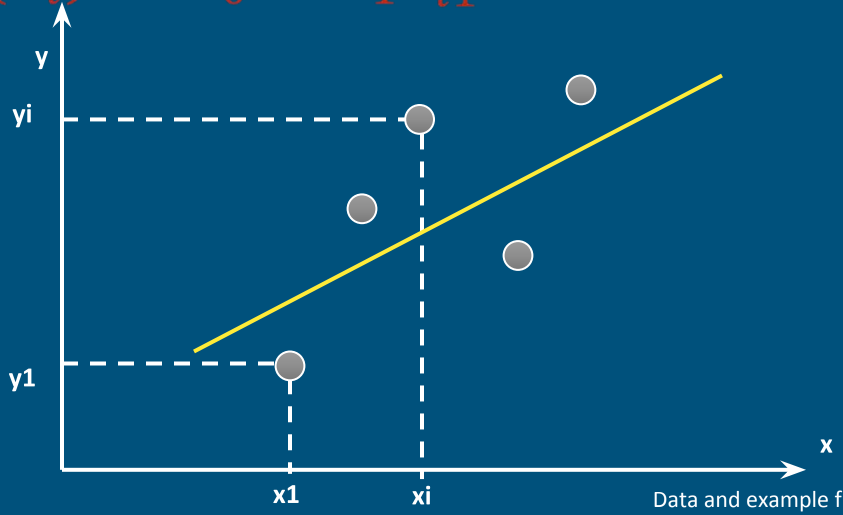
- Lets assume the 'model' is **Linear**
  - $\hat{y}_i = \hat{y}(\boldsymbol{x}_i) = w_0 + w_1 x_{i1} + w_2 x_{i2} + \cdots + w_d x_{id}$
  - If d = 1
    - $\hat{y}_i = \hat{y}(\boldsymbol{x}_i) = w_0 + w_1 x_{i1}$



Data and example from Nando de Fretias's lecture slides

Linear Regression

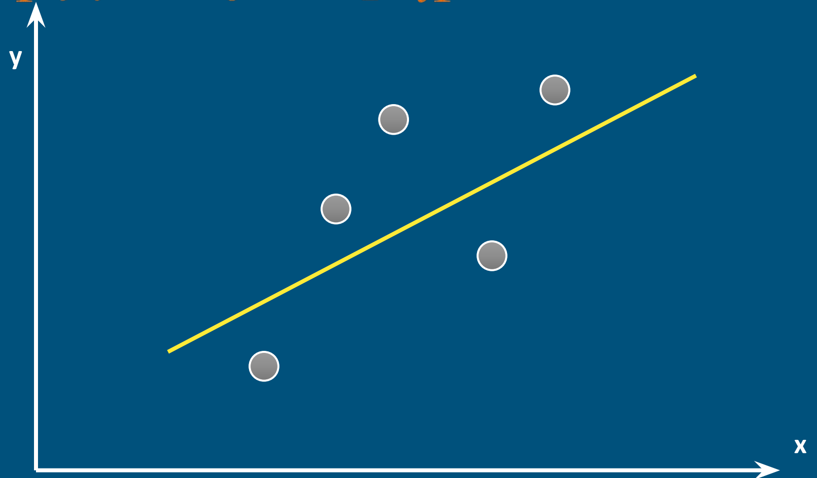
- Lets assume the 'model' is **Linear**
  - $\hat{y}_i = \hat{y}(\boldsymbol{x}_i) = w_0 + w_1 x_{i1} + w_2 x_{i2} + \cdots + w_d x_{id}$
  - If d = 1
    - $\hat{y}_i = \hat{y}(\boldsymbol{x}_i) = w_0 + w_1 x_{i1} = \boldsymbol{w}^t \boldsymbol{x}$



Data and example from Nando de Fretias's lecture slides

Linear Regression

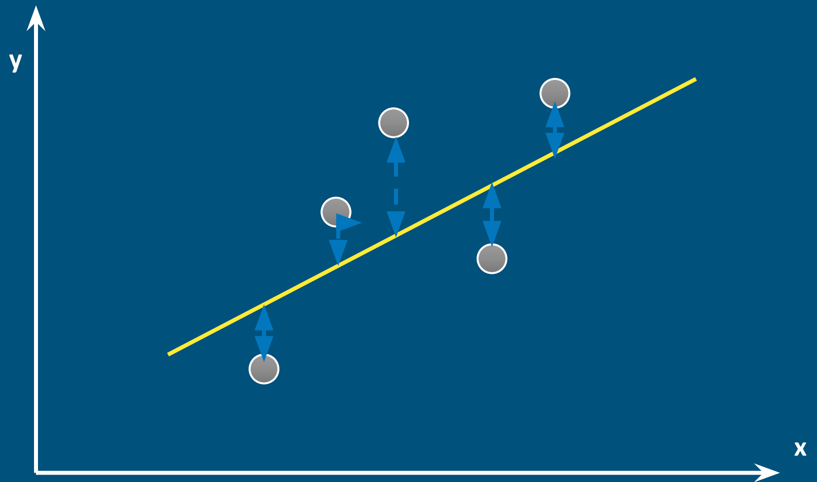
- Given any ‘w’ we want to calculate error
- Lets define **error function/loss/objective function**
  - $J(\mathbf{w}) = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
  - $J(\mathbf{w}) = \sum_{i=1}^{n}(y_i - w_0 - w_1 x_{i1})^2$



Data and example from Nando de Fretias’s lecture slides

Linear Regression

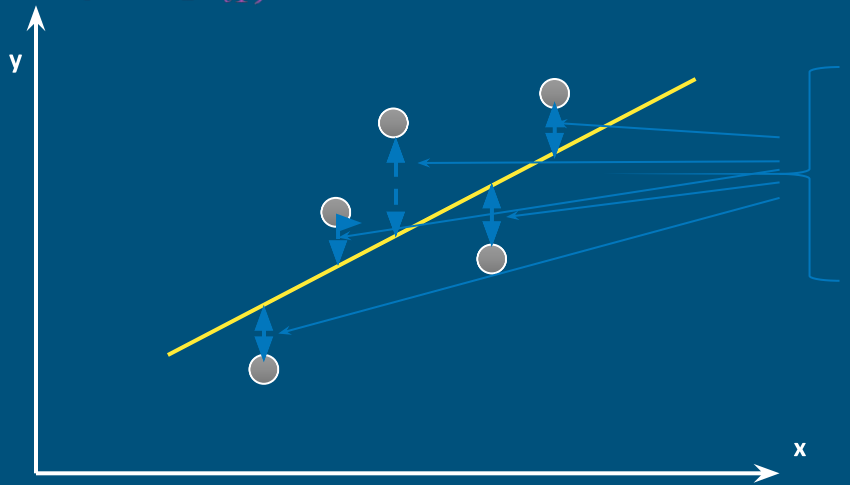
- Given any ‘w’ we want to calculate error
- Lets define **error function/loss/objective function**
  - $J(w) = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
  - $J(w) = \sum_{i=1}^{n}\left(y_i - w_0 - w_1 x_{i1}\right)^2$



Data and example from Nando de Fretias’s lecture slides

Linear Regression

- Given any 'w' we want to calculate error
- Lets define **error function/loss/objective function**
  - $J(\boldsymbol{w}) = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
  - $J(\boldsymbol{w}) = \sum_{i=1}^{n}\left(y_i - w_0 - w_1 x_{i1}\right)^2$



We want to choose '**w**' such that these distances are decreased

Data and example from Nando de Fretias's lecture slides

Line Fitting: Least Squared Error Solution

$$E = \sum_i \left(m x_i + c - y_i \right)^2$$

$$\frac{\partial E}{\partial m} = \sum_i \left(m x_i + c - y_i \right) x_i = 0$$

$$\frac{\partial E}{\partial c} = \sum_i \left(m x_i + c - y_i \right) = 0$$

$$\begin{bmatrix} \sum_i x_i^2 & \sum_i x_i \\ \sum_i x_i & \sum_i 1 \end{bmatrix} \begin{bmatrix} m \\ c \end{bmatrix} = \begin{bmatrix} \sum_i x_i y_i \\ \sum_i y_i \end{bmatrix}$$

x	y
1.3	5.7
2.4	7.3
3.4	10.5
4.6	11.8
5.3	13.9
6.6	16.3
6.4	15.3
8.0	17.9
8.9	20.8
9.2	20.9

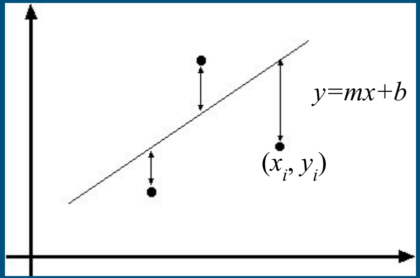
$$\begin{pmatrix} 380.63 & 56.1 \\ 56.1 & 10 \end{pmatrix} \begin{pmatrix} m \\ c \end{pmatrix} = \begin{pmatrix} 914.68 \\ 140.4 \end{pmatrix}$$

Solution: m = 1.9274 c = 3.227

Linear Regression: Least Square Error Solution

Model

- Data: $(x_1, y_1), \ldots, (x_n, y_n)$
- Line equation: $y_i = m x_i + b$
- Find (m, b) to minimize



Error Function

$$\boxed{E = \sum_{i=1}^{n} (y_i - m x_i - b)^2}$$

$$E = \sum_{i=1}^{n} \left(\begin{bmatrix} x_i & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} - y_i \right)^2 = \left\| \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} - \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \right\|^2 = \|\mathbf{Ap} - \mathbf{y}\|^2$$

$$= \mathbf{y}^T \mathbf{y} - 2(\mathbf{Ap})^T \mathbf{y} + (\mathbf{Ap})^T (\mathbf{Ap})$$

$$\frac{dE}{dp} = 2\mathbf{A}^T \mathbf{Ap} - 2\mathbf{A}^T \mathbf{y} = 0$$

$$\mathbf{A}^T \mathbf{Ap} = \mathbf{A}^T \mathbf{y} \Rightarrow \mathbf{p} = \left(\mathbf{A}^T \mathbf{A}\right)^{-1} \mathbf{A}^T \mathbf{y}$$

Matlab: `p = A \ y;`

Modified from S. Lazebnik

Administrative Stuff

Administrative Issues

- Course Outline
- Course Website
- Zero tolerance Plagiarism policy
- Assignments
- Quizzes
- Exams
  - Mid-term
  - Final term

Administrative Issues

- We MIGHT OR MIGHT-NOT share Slides
  - Take Notes
  - Share notes
- We Will Provide
  - Reading Material (with concise pointers)
  - Links to the video Lectures that are helpful
  - Reference Material
  -
  - Office Hours

Assigned Readings

- Deep Learning, Nature’s Paper
- Some interesting blogs?

Reference and Reading Material for Next Class

- Neural Networks And Deep learning, Chapter 1

http://neuralnetworksanddeeplearning.com/chap1.html