UNIVERSITAS ISLAM INDONESIA

The Fundamentals in Machine Learning

Ridho Rahmadi

Universitas Mataram

September 15, 2019

Outline

Motivation

One way to resolve a real-world problem is to make a *possible* solution called a **hypothesis**

Motivation

One way to resolve a real-world problem is to make a *possible* solution called a **hypothesis**

This is, in particular, appropriate for problems for which no *well-established* knowledge exist

Motivation

No *well-established* knowledge exist

Motivation

No *well-established* knowledge exist

- clinical model for a rare disease
- causal model for weather circumstances
- political model for third world countries
- econometric model for ex-socialist countries

Motivation

A hypothesis typically can be of the following forms.

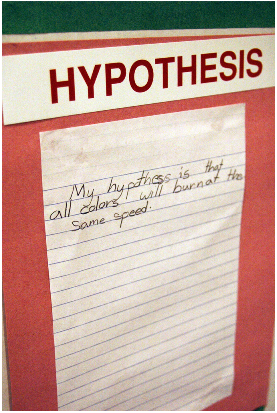
- mathematical formula
- description
- graphical illustration

$$y = \theta^T x + b$$

Motivation

A hypothesis typically can be of the following forms.

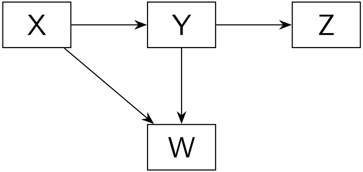
- mathematical formula
- description
- graphical illustration



Motivation

A hypothesis typically can be of the following forms.

- mathematical formula
- description
- graphical illustration



Problem-hypothesis examples

We want to predict whether or not a student passes the exam, based on his/her 7-day study hours prior to the exam.

Problem-hypothesis examples

We want to predict whether or not a student passes the exam, based on his/her 7-day study hours prior to the exam.

- A hypothesis: a student will pass if he/she has studied for at least 15 hours

Problem-hypothesis examples

We want to predict whether or not a student passes the exam, based on his/her 7-day study hours prior to the exam.

- A hypothesis: a student will pass if he/she has studied for at least 15 hours

We want to know if new incoming emails are of spam or not.

Problem-hypothesis examples

We want to predict whether or not a student passes the exam, based on his/her 7-day study hours prior to the exam.

- A hypothesis: a student will pass if he/she has studied for at least 15 hours

We want to know if new incoming emails are of spam or not.

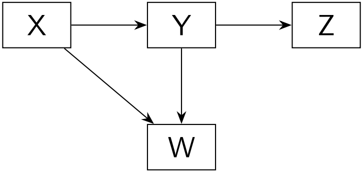
- A hypothesis: if an email contains words "selamat" and "hadiah", then it is a spam.

Hypothesis

The thing is that, there could be **millions of equally plausible** hypotheses to solve a single problem.

Hypothesis

For example, given n variables, with such a hypothesis representation below



we will have $3^{\frac{n(n-1)}{2}}$ equally plausible *models*.

Learning here, is to find
the *best* hypothesis, out of those possibilities.

Learning here, is to find
the *best* hypothesis, out of those possibilities.

Machine learning is about algorithms for learning.

Machine learning is used for **making** hypotheses to our real-world problems

Machine learning is used for **making** hypotheses to our real-world problems

AND/OR

evaluating hypotheses.

Machine learning is used for **making** hypotheses to our real-world problems

AND/OR

evaluating hypotheses.

In order to do those above: we need **data**.

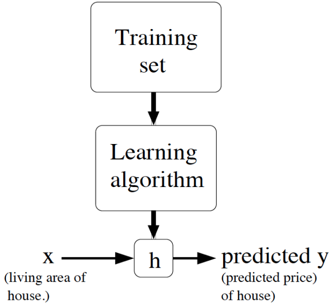
Supervised Learning

Supervised Learning

A learning paradigm that uses data consisting of pairs of input and output variables $(x^{(i)}, y^{(i)})$, to find model.

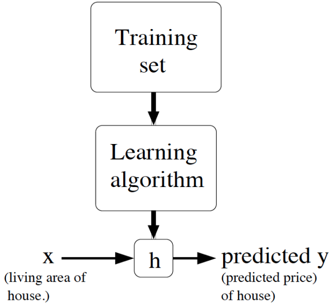
Supervised learning

- $x^{(i)}$ denotes *input* variables or **features** (living area)
- $y^{(i)}$ denotes *output* or **target** variable (price)
- A pair $(x^{(i)}, y^{(i)})$ is called a training example
- A list of m training examples $(x^{(i)}, y^{(i)}); i = 1, \ldots, m$ is called a **training set**



Supervised learning

- We use $\mathcal{X}$ to denote input space and $\mathcal{Y}$ to denote output space
- In this example, $\mathcal{X}, \mathcal{Y} \in \mathbb{R}$
- **Supervised learning**: given a training set, to learn a function $h : \mathcal{X} \to \mathcal{Y}$, so that $h(x)$ is a good predictor for y
- The function h is a **hypothesis** or **model**



Training set

Luas rumah (x)	Harga (y)
2104	400
1600	330
2400	369
1416	232
3000	540
$\vdots$	$\vdots$

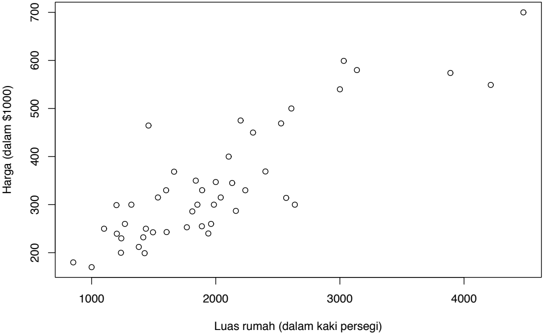
For a regression task

Jam belajar (x)	Lulus (y)
5	Tidak
6	Ya
1	Tidak
7	Ya
4	Tidak
$\vdots$	$\vdots$

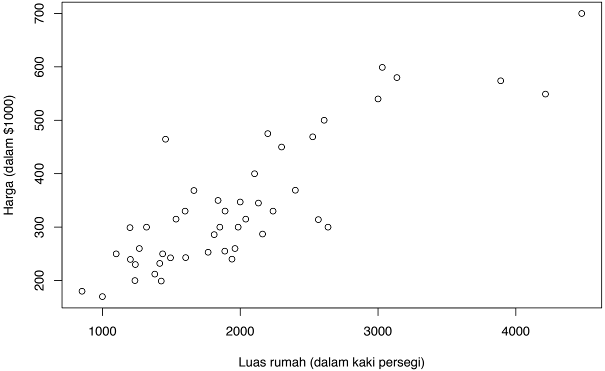
For a classification task

Supervised Learning Example on Linear Regression

Luas rumah	Harga
2104	400
1600	330
2400	369
1416	232
3000	540
⋮	⋮

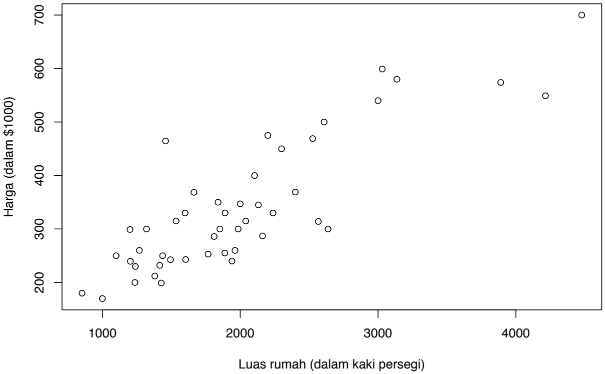


Linear regression



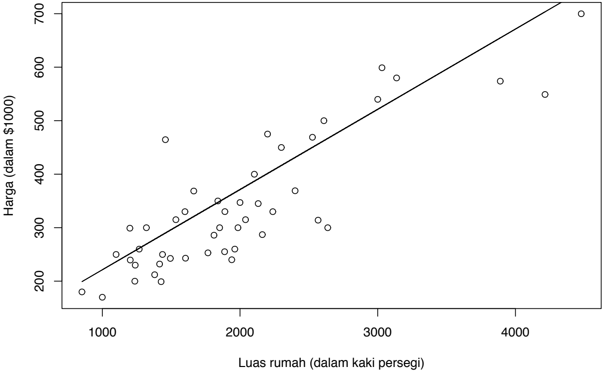
Suppose that we want to predict house price (Harga) based on the training set plotted above.

Linear regression



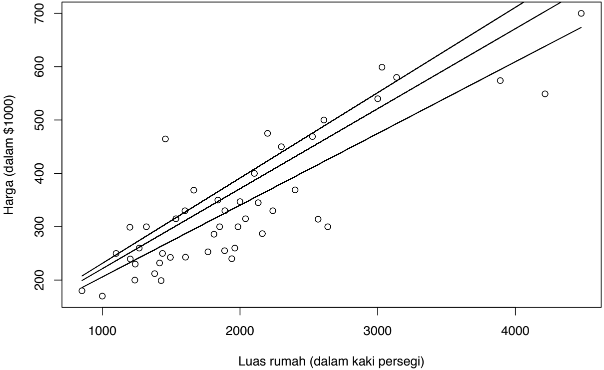
If we are about to *draw* a good hypothesis model h, what would you draw to *best* model the data?

Linear regression



Yes, a straight line seems to fit the data well.

Linear regression



Yes, a straight line seems to fit the data well. But which line? There are infinitely many possible lines (hypotheses, recall the earlier slides).

Linear regression

Which line?

Linear regression

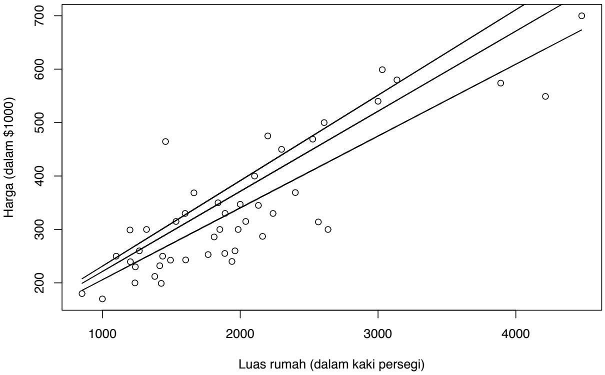
Which line?
Intuitively, we want to get a line that is *approximately going through the middle* of the data distribution.

Linear regression

Which line?
Intuitively, we want to get a line that is *approximately going through the middle* of the data distribution.

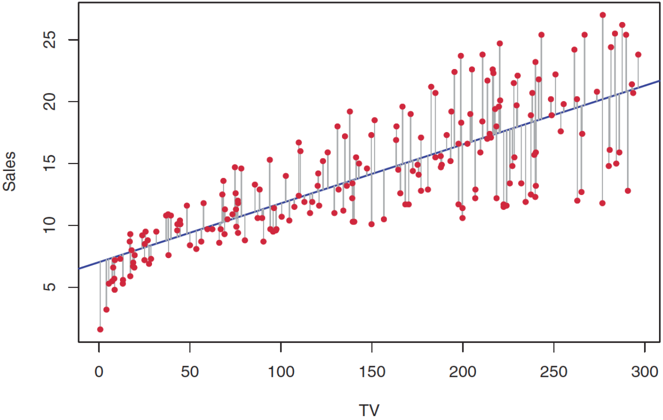
If we think in terms of distance, the best line is the one that is close to every data point.

Linear regression



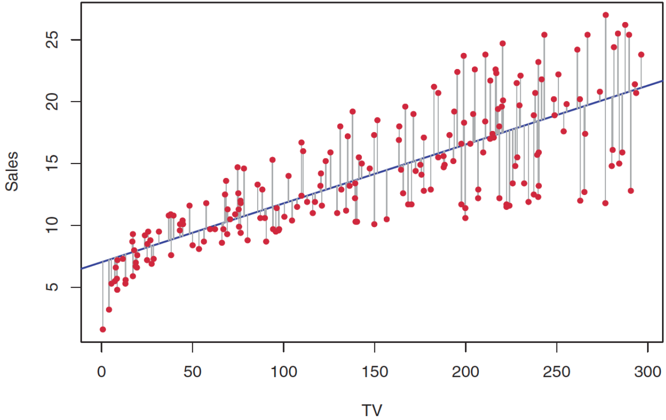
Which line?

Linear regression



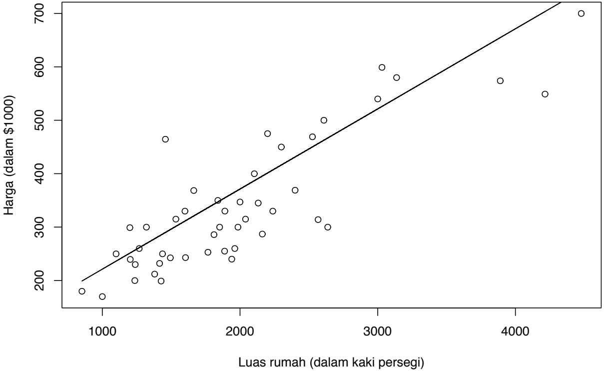
We can think a distance between the line and a data point as an **error**.

Linear regression



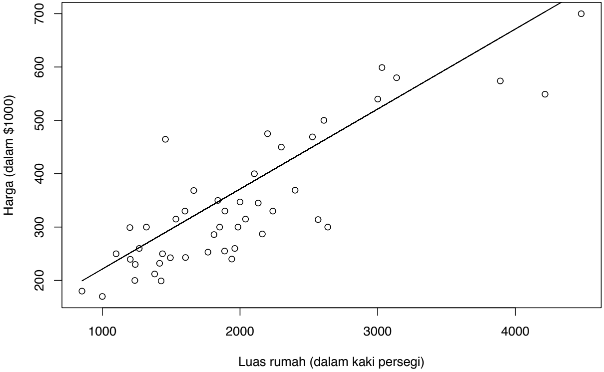
Thus, the *best* line to model our hypothesis is the one that has the smallest accumulated error.

Linear regression



Mathematically, the above straight line can be written by
Harga $= \theta_0 + \theta_1$ Luas.

Linear regression



What are parameters θ_0 and θ_1 in the plot above?

Linear regression

Given that,

$$\text{Harga} = \theta_0 + \theta_1 \text{Luas},$$

linear regression is a procedure to find the best line (hypothesis or model) by *searching* parameters θ_0 and θ_1 that give the *smallest* total error.

Linear regression

More formally, a straight line can be represented by,

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

where θ_i's are parameters or weights, parameterizing the space of linear functions mapping $\mathcal{X} \to \mathcal{Y}$.

To obtain the best line, we minimize the cost function,

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

which is called the **ordinary least squares** regression model.

Gradient descent

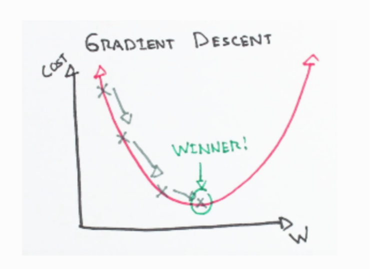
Steps to minimize the cost function $J(\theta)$ using the gradient descent approach:

1. Pick an *initial guess* of θ
2. Repeatedly changes θ to make $J(\theta)$ smaller
3. Until hopefully converge to a value that minimizes $J(\theta)$

Gradient descent

Steps to minimize the cost function $J(\theta)$ using the gradient descent approach:

1. Pick an *initial guess* of θ (or w in the figure)
2. Repeatedly changes θ to make $J(\theta)$ smaller
3. Until hopefully converge to a value that minimizes $J(\theta)$

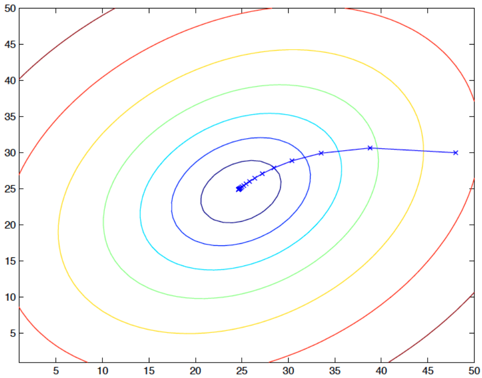


Gradient descent

Gradient descent repeatedly performs the update:

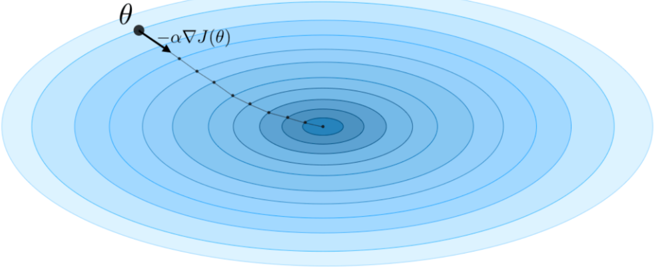
$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta),$$

where α is the learning parameter.



Gradient descent

$$\boxed{\theta \longleftarrow \theta - \alpha \nabla J(\theta)}$$



The idea behind the learning

1. Pick an initial model or hypothesis $h(\theta)$ (or technically a "guess" of θ value).
2. Compute the corresponding cost function

$$J(\theta) = \sum_{i=1}^{m} L(h_\theta(x^{(i)}), y^{(i)}),$$

   where L is the loss function.
3. Update the model $h(\theta)$ (technically by changing θ that makes $J(\theta)$ smaller; this can be done by using the gradient descent)

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

4. Repeat Steps 2 and 3 until converges

This learning idea is typically used in mostly (parametric) models of machine learning and deep learning.

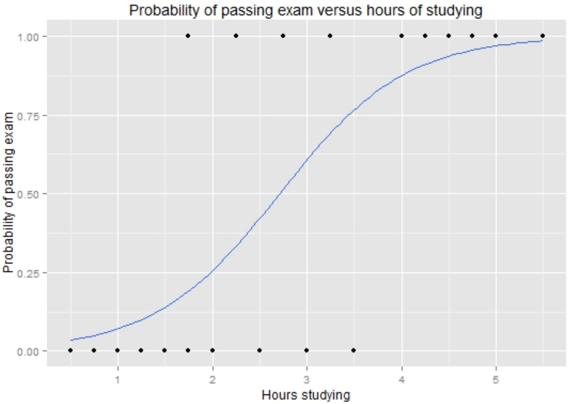
Logistic regression

Recall that we have been so far assuming that y is continuous (e.g., house price). What if y is discrete?

For example, if y indicates whether an email is a spam (1) or not (0), or whether a student passes the exam (1) or not (0).

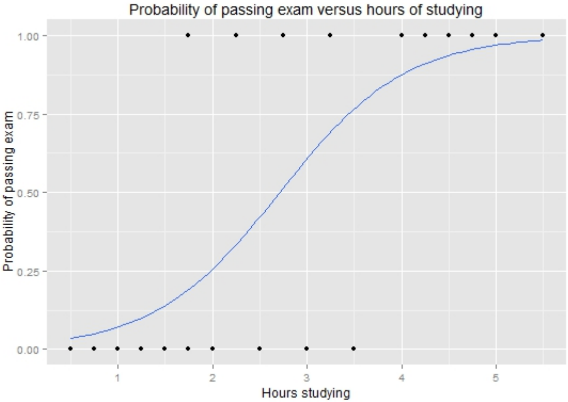
This problem is called **classification**.

Logistic regression



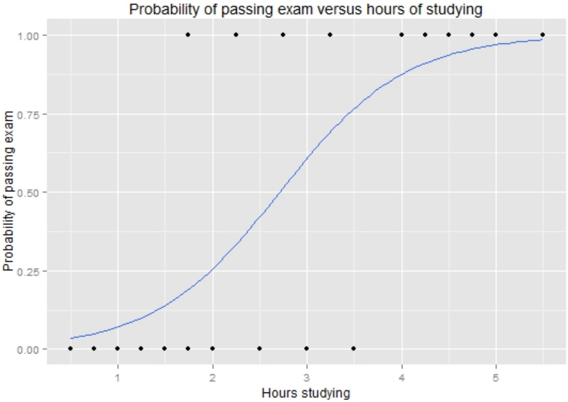
In this example, y denotes the status whether students passing exam (1) or not (0), and x indicates hours that students spent for studying.

Logistic regression



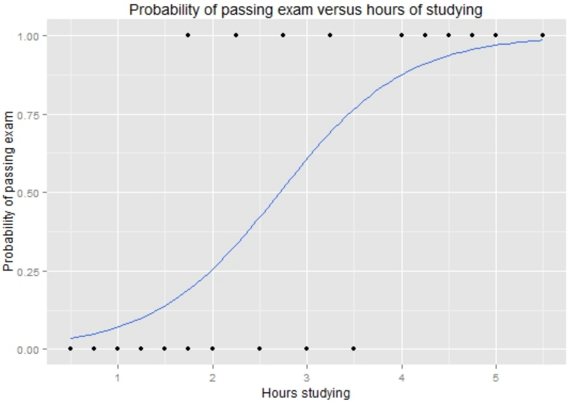
We can see that the straight line is no longer suitable to represent the data here.

Logistic regression



The model indicated by the blue curve represents the data better.

Logistic regression



The blue curve can be represented by a logistic or a sigmoid function.

Logistic regression

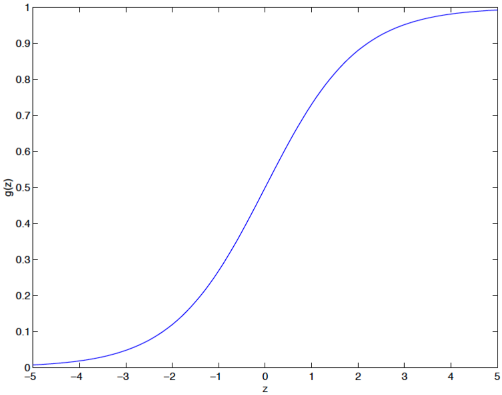
The logistic function reads

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

We predict "1" if $h_\theta(x) \geq 0.5$, i.e., if and only if $\theta^T x \geq 0$.
Let $\hat{y} = h_\theta(x)$, the logistic loss function is given by

$$L(y, \hat{y}) = \log(1 + exp(-y\hat{y})).$$

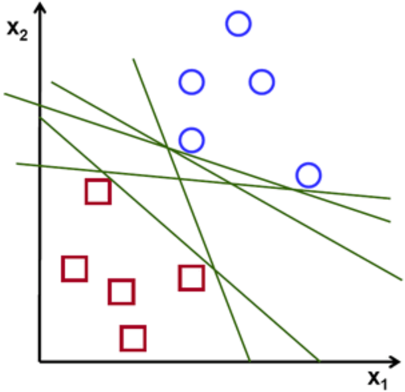
Logistic regression



Note that $g(\theta^T x)$ tends toward 1 as $\theta^T x \to \infty$, and $g(\theta^T x)$ tends toward 0 as $\theta^T x \to -\infty$.

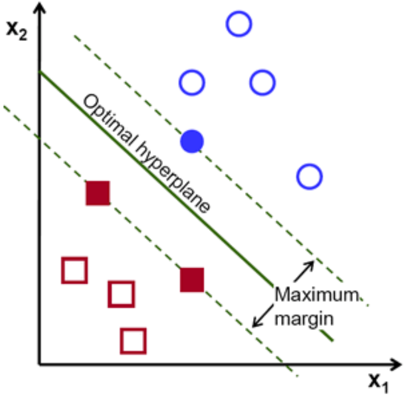
Support vector machine

SVM



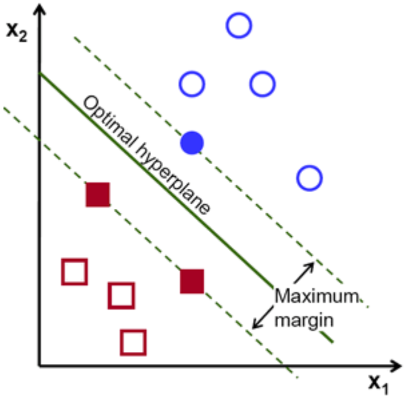
Which line best separates the data points into two classes?

SVM



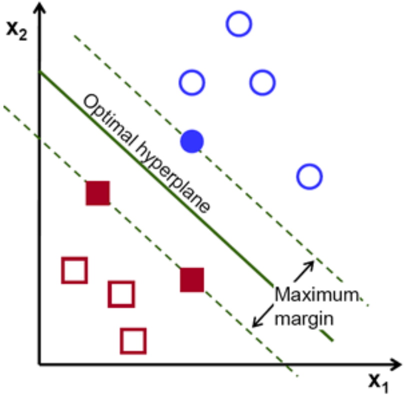
SVM aims to find the optimal line (hyperplane) for classifying the data points. How?

SVM



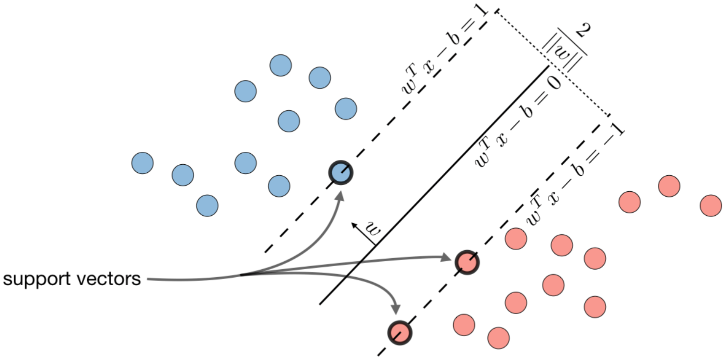
SVM uses **margin** to indicate the distance between a hyperplane to the closest data points (support vectors).

SVM



The optimal hyperplane is the one that maximizes the margin, i.e., maximum distance between data points of both classes.

SVM



SVM

We define the classifier in SVM via

$$h(x) = g(w^T b).$$

Here, $g(w^T x + b) = +1$ if $w^T x + b \geq 0$, and $g(w^T x + b) = -1$ otherwise.

SVM

The optimal margin classifier h is the one which (w, b) are the solution of the following constrained optimization problem.

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}||w||^2 \\ \text{subject to} \quad & y^{(i)}(w^T x^{(i)} + b) \geq 1,\ i = 1, \ldots, n. \end{aligned}$$

This is the learning procedure of SVM, which is basically in the same spirit of the learning procedure we have described previously, except that here we apply a linear constraint.

SVM

SVM uses the hinge loss function that reads

$$L(y, \hat{y}) = [1 - y\hat{y}]_+ = \max(0, 1 - y\hat{y}).$$

SVM: an extended version of the objective function

$$\min_{\mathbf{w}} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_i \max(0, 1 - y_i\mathbf{w}^T\mathbf{x}_i)$$

Regularization term:
- Maximize the margin
- Imposes a preference over the hypothesis space and pushes for better generalization
- Can be replaced with other regularization terms which impose other preferences

Empirical Loss:
- Hinge loss
- Penalizes weight vectors that make mistakes
- Can be replaced with other loss functions which impose other preferences

A **hyper-parameter** that controls the tradeoff between a large margin and a small hinge-loss

SVM

The objective function

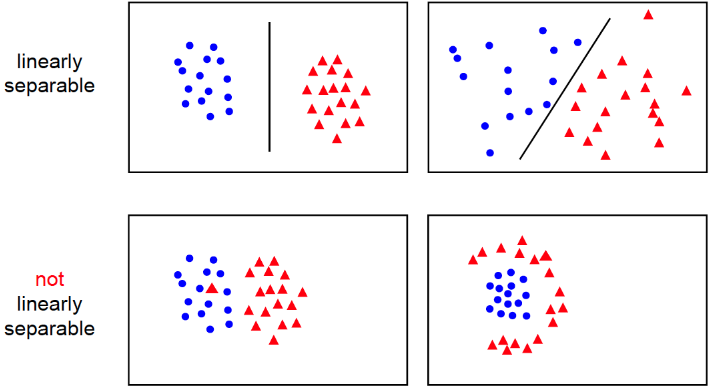
$$\min_w \frac{1}{2} w^T w + C \sum_i \max(0, 1 - y^{(i)} \hat{y}^{(i)})$$

is equivalent to an unconstrained optimization and can be solved with the gradient descent, by minimizing rephrasing it via a cost function

$$J(w) = \frac{1}{2} w^T w + C \sum_i \max(0, 1 - y^{(i)} \hat{y}^{(i)}).$$

Recall the learning paradigm we have discussed.

SVM



We have discussed the case of linearly separable data. What if not?

SVM

In the case of non linearly separable data, SVM transforms the data into a higher dimension space via a feature mapping

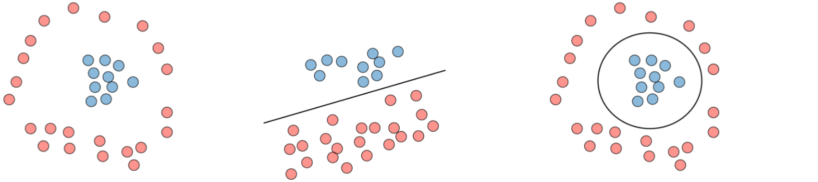
$$K(x, z) = \phi(x)^T \phi(z).$$

Typically the kernel K is defined by

$$K(x, z) = \exp\left(\frac{||x - z||^2}{2\sigma^2}\right),$$

or called a Gaussian kernel. Note that z here is used to distinguish data points, e.g., $K(x_i, x_j)$.

SVM



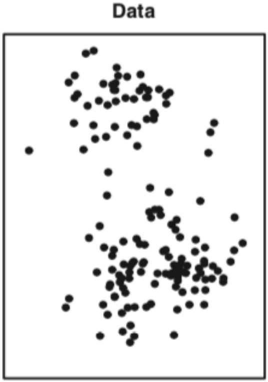
Non-linear separability → Use of a kernel mapping ϕ → Decision boundary in the original space

Unsupervised learning

Unsupervised Learning

- We **ONLY** have $x^{(i)}$
- We **do not have** *output* or **target** variable $y^{(i)}$
- Thus, our training set becomes $\{x^{(1)}, \ldots, x^{(m)}\}$

Here, we are not interested in prediction or classification (as we don't have the associated target variable $y^{(i)}$).

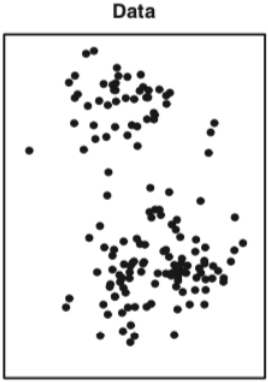


Unsupervised Learning

In unsupervised learning, we are interested in to discover *interesting* things from the data set $\{x^{(1)}, \ldots, x^{(m)}\}$.

Clustering

- Clustering aims to find *subgroups* or *clusters* in a data set
- The idea: partitioning data into distinct groups
  - observations within each group are quite **similar**
  - observations in different groups are quite **different**

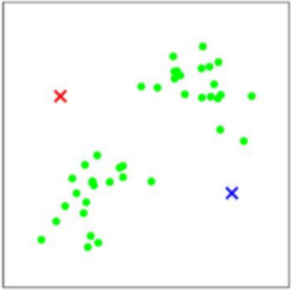


K-Means

1. Initialize cluster centroids
2. Repeat until convergence (no change)
   2.1 Assign each ith observation to the closest cluster centroid
   2.2 For each cluster, move the centroid to the mean of observations belong to the cluster

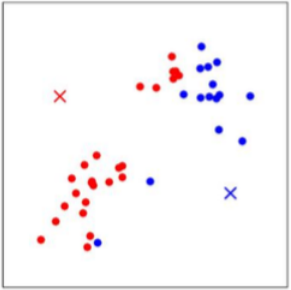
K-Means

1. Initialize cluster centroids
2. Repeat until convergence (no change)
    2.1 Assign each ith observation to the closest cluster centroid
    2.2 For each cluster, move the centroid to the mean of observations belong to the cluster



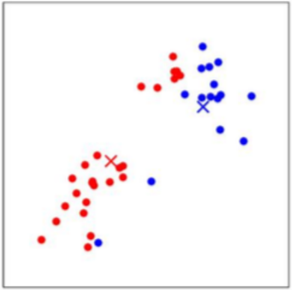
K-Means

1. Initialize cluster centroids
2. Repeat until convergence (no change)
   2.1 Assign each ith observation to the closest cluster centroid
   2.2 For each cluster, move the centroid to the mean of observations belong to the cluster



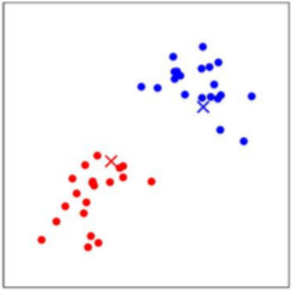
K-Means

1. Initialize cluster centroids
2. Repeat until convergence (no change)
    2.1 Assign each ith observation to the closest cluster centroid
    2.2 For each cluster, move the centroid to the mean of observations belong to the cluster



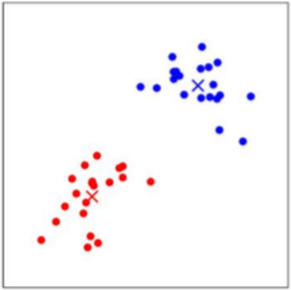
K-Means

1. Initialize cluster centroids
2. Repeat until convergence (no change)
    2.1 Assign each ith observation to the closest cluster centroid
    2.2 For each cluster, move the centroid to the mean of observations belong to the cluster



K-Means

1. Initialize cluster centroids
2. Repeat until convergence (no change)
   2.1 Assign each ith observation to the closest cluster centroid
   2.2 For each cluster, move the centroid to the mean of observations belong to the cluster



K-Means

1. Initialize cluster centroids

$$\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}$$

2. Repeat until convergence (no change)
   2.1 Assign each ith observation to the closest cluster centroid

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

   2.2 For each cluster, move the centroid to the mean of observations belong to the cluster

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

Sources

- Andrew Ng's machine learning materials
- An Introduction to statistical learning, James et al.
- https://stanford.edu/ shervine/teaching/cs-229/cheatsheet-deep-learning
- http://cs231n.github.io/convolutional-networks/
- https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53
- https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47