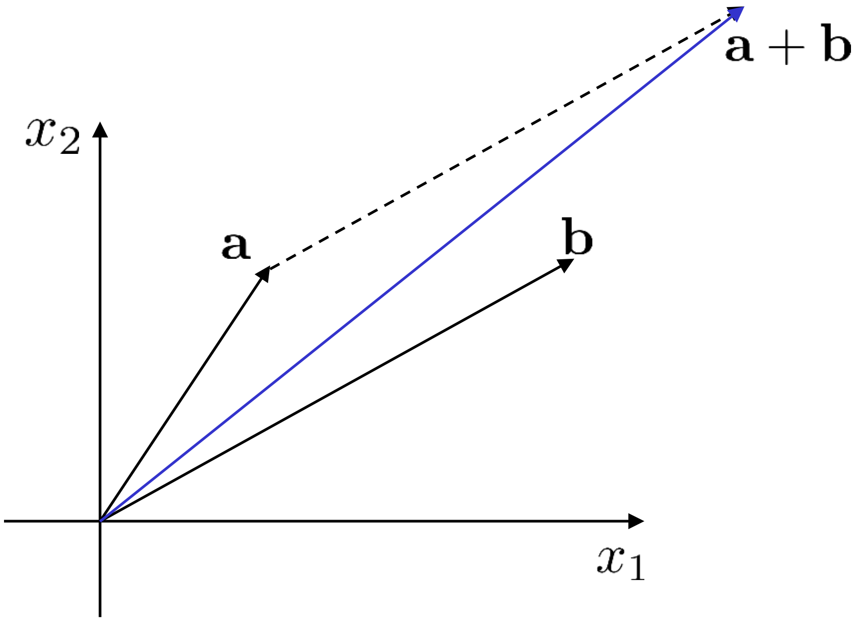
COGS 118A, Spring 2019

Supervised Machine Learning Algorithms

Lecture 2: Vector and Decision Boundary

Vector

Addition:



$$\mathbf{a} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

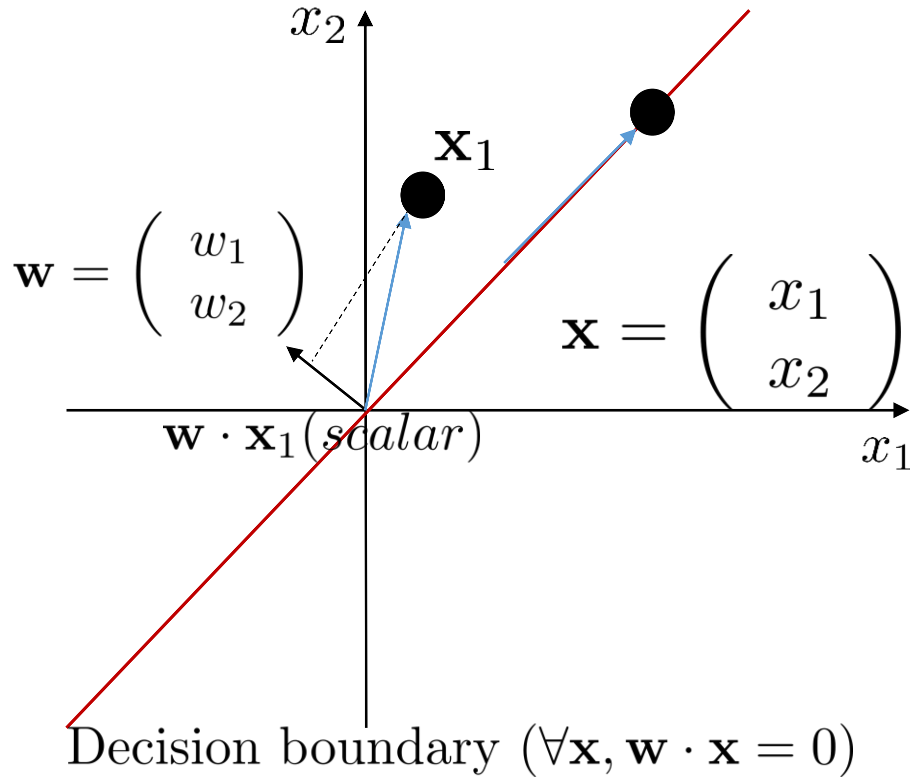
$$\mathbf{a}+\mathbf{b} = \begin{pmatrix} a_1 + b_1 \\ a_2 + b_2 \\ a_3 + b_3 \end{pmatrix}$$

It's still a vector in the same space as a and b.

Visual illustration by 3Blue1Brown

https://www.youtube.com/watch?v=fNk_zzaMoSs

Line and vector



Any point $\mathbf{x}$ on the line satisfies:

$$\mathbf{w}^T \mathbf{x} \equiv < \mathbf{w}, \mathbf{x} > \equiv \mathbf{w} \cdot \mathbf{x} = 0$$

$\mathbf{w}$ is the normal direction of the line

Often: $||\mathbf{w}||_2 = 1$: a unit vector

Line and vector

Understanding the mathematical and physical meaning of vector is very important in machine learning.

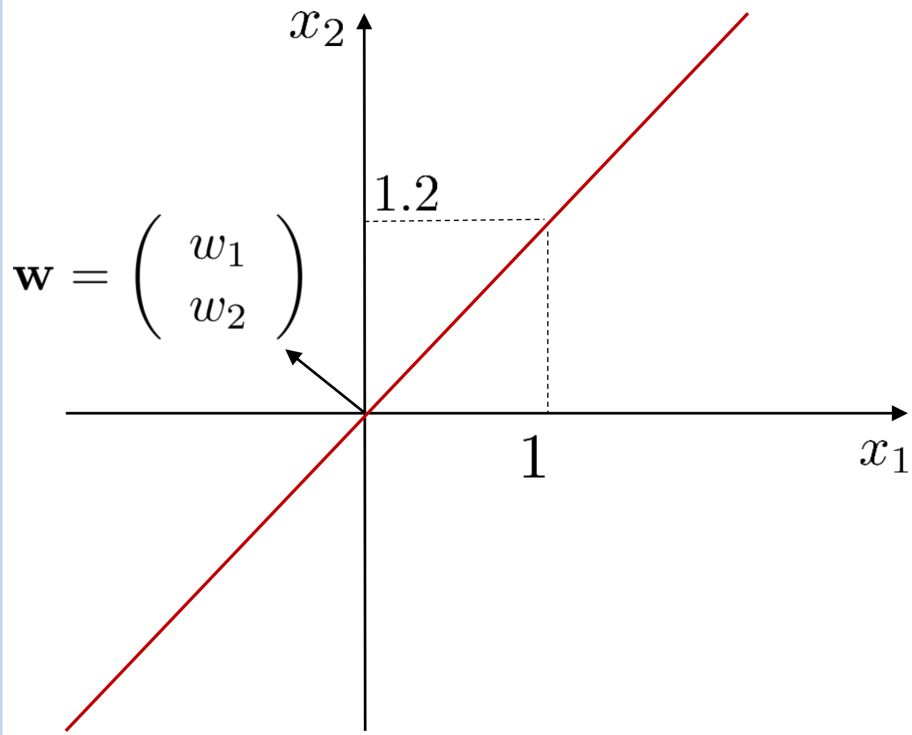
A good resource of visual illustration:

https://www.youtube.com/watch?v=fNk_zzaMoSs&t=3s

Our input data $\mathbf{x}$ and classification model parameter $\mathbf{w}$ are always represeted as (high dimensional) vectors, consisting of a number features.

Line and vector

an example:



$\mathbf{w}$ is the normal direction of the line

Often: $||\mathbf{w}||_2 = 1$: a unit vector

$$x_2 = 1.2 \times x_1$$

$$\mathbf{w} = \begin{pmatrix} -\frac{1.2}{\sqrt{2.44}} \\ \frac{1}{\sqrt{2.44}} \end{pmatrix}$$

Significance of the dot product between two vectors

"Dot product" outputs a scalar value and it is arguably the most important mathematical operation in machine learning.

$$\begin{aligned} &< \mathbf{a}, \mathbf{b} > \equiv \mathbf{a} \cdot \mathbf{b} \equiv \mathbf{a}^T \mathbf{b} \\ &\equiv < \mathbf{b}, \mathbf{a} > \equiv \mathbf{b} \cdot \mathbf{a} \equiv \mathbf{b}^T \mathbf{a} \end{aligned} \qquad \mathbf{a} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

Why?

"Dot product" computes for the magnitude for the projection from one vector to the other, which essentially measures the "similarity" between two vectors.

For two unit vectors (L2 norm being 1), their dot product outputs the largest value 1 when they are well aligned (same), and otherwise 0 when they are orthogonal (different) to each other.

Significance of the dot product between two vectors

	fly?	laying eggs?	weight (lb)
sparrow	yes	yes	0.087
chipmunk	no	no	0.19
bat	yes	no	0.09

Feature representation (one-hot encoded).

$$\text{sparrow} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0.087 \end{pmatrix} \qquad \text{chipmunk} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.19 \end{pmatrix} \qquad \text{bat} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.09 \end{pmatrix}$$

$$\text{sparrow} \cdot \text{chipmunk} = 0.01653$$

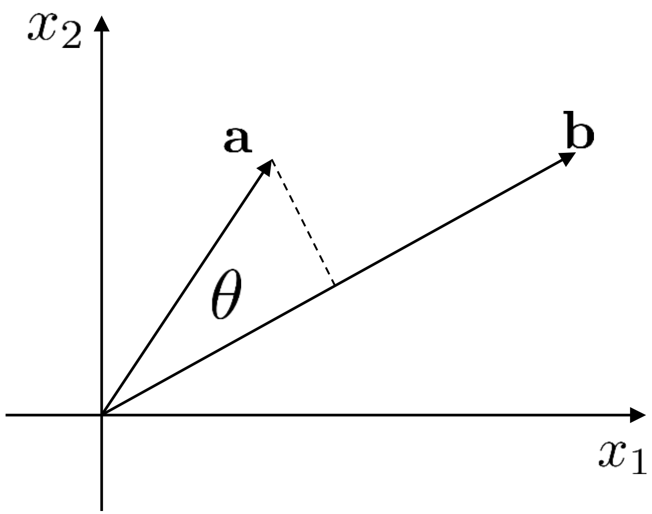
very different!

$$\text{sparrow} \cdot \text{bat} = 1.00783$$

$$\text{chipmunk} \cdot \text{bat} = 1.0171$$

Vector: Projection (inner product)

(one of the most important concepts in machine learning)



$$\mathbf{a} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

$$< \mathbf{a}, \mathbf{b} > \ \equiv \mathbf{a} \cdot \mathbf{b} \ \equiv \mathbf{a}^T \mathbf{b} \ \equiv a_1 b_1 + a_2 b_2 + a_3 b_3$$

It's a scalar!

$$cos(\theta) = \frac{<\mathbf{a},\mathbf{b}>}{||\mathbf{a}||_2 \times ||\mathbf{b}||_2}$$

The "cosine similarity" above can be used to measure the "similarity" between two vectors (data samples) that are not normalized (non-unit).

Cosine similarity

A cosine similarity value of 0 indicates two data samples being, to each other,

A. the least similar.

B. the most similar.

C. the most uncertain.

D. the least uncertain.

Cosine similarity

A cosine similarity value of 0 indicates two data samples (points) being, to each other,

A. the least similar

B. the most similar

C. the most uncertain

D. the least uncertain

Cosine similarity

	fly?	laying eggs?	weight (lb)
sparrow	yes	yes	0.087
chipmunk	no	no	0.19
bat	yes	no	0.09

Feature representation (one-hot encoded).

$$\text{sparrow} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0.087 \end{pmatrix} \qquad \text{chipmunk} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0.19 \end{pmatrix} \qquad \text{bat} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0.09 \end{pmatrix}$$

$$\frac{\text{sparrow} \cdot \text{chipmunk}}{||\text{sparrow}||_2 \times ||\text{chipmunk}||_2} = 0.0082$$

$$\frac{\text{sparrow} \cdot \text{bat}}{||\text{sparrow}||_2 \times ||\text{bat}||_2} = 0.502$$

$$\frac{\text{chipmunk} \cdot \text{bat}}{||\text{chipmunk}||_2 \times ||\text{bat}||_2} = 0.503$$

$\cdot$ refers to the dot product between two vectors;

$||\ ||_2$ refers to the L2 norm of a vector;

$\times$ refers to the multiplication of two scalar values.

Feature scaling is another factor

Now we purposely stretch one particular feature dimension by a large factor. Let's see what will happen.

$$\text{sparrow} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 87 \end{pmatrix} \qquad \text{chipmunk} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 190 \end{pmatrix} \qquad \text{bat} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 90 \end{pmatrix}$$

$$\frac{\text{sparrow}\cdot\text{chipmunk}}{||\text{sparrow}||_2 \times ||\text{chipmunk}||_2} = 0.99984$$

$$\frac{\text{sparrow}\cdot\text{bat}}{||\text{sparrow}||_2 \times ||\text{bat}||_2} = 0.99987$$

$$\frac{\text{chipmunk}\cdot\text{bat}}{||\text{chipmunk}||_2 \times ||\text{bat}||_2} = 0.99990$$

Now, the concept of similarity diminishes.
Conclusion: The relative scaling of the individual features is also important.

In practice, we often normalize the individual features to [0, 1] to make them directly comparable.

Cosine similarity

The semantic interpretation of "dot product" refers to as the similarity (un-normalized) between two vectors (data samples).

The greater the dot product value is, the more "similar" the two data samples are.

The dot product value 0 refers to the least similar two data samples, indicating two vectors that are orthogonal to each other.

The "cosine similarity" can be used to measure the "similarity" (normalized, [0, 1]) between two vectors (data samples).

0 and 1 refer to the least and the most "similar" data samples respectively.

More illustrations about vector operations

https://www.youtube.com/watch?v=fNk_zzaMoSs&t=3s

3Blue1Brown

Basics about data and linear algebra operations

$$S = \{(\mathbf{x}_i, y_i), i = 1..n\} \qquad y_i \in \{-1, +1\}$$

	age	male or female	weight (lb)	height (cm)
$y_1 = -1$ (negative)	$x_{11} = 22$	$x_{12} = M$	$x_{13} = 160$	$x_{14} = 180$
$y_2 = +1$ (positive)	$x_{21} = 51$	$x_{22} = M$	$x_{23} = 190$	$x_{24} = 175$
$y_3 = +1$ (positive)	$x_{31} = 43$	$x_{32} = F$	$x_{33} = 120$	$x_{34} = 165$

$$\Downarrow$$

$$X = \begin{pmatrix} 22 & 1 & 0 & 160 & 180 \\ 51 & 1 & 0 & 190 & 175 \\ 43 & 0 & 1 & 120 & 165 \end{pmatrix} \qquad Y = \begin{pmatrix} -1 \\ +1 \\ +1 \end{pmatrix}$$

$$W = \begin{pmatrix} 0.075 \\ 0 \\ 0 \\ -0.007 \\ -0.008 \end{pmatrix} \qquad \hat{Y} = XW = \begin{pmatrix} -0.91 \\ 1.095 \\ 1.065 \end{pmatrix}$$

We assume a given W.

Matrix multiplication

Vector:

$$A = \begin{pmatrix} a_1 & a_2 & a_3 \end{pmatrix} \qquad B = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

$$AB = \begin{pmatrix} a_1 & a_2 & a_3 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = a_1 b_1 + a_2 b_2 + a_3 b_3$$

$$AB \neq BA$$

$$BA = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \begin{pmatrix} a_1 & a_2 & a_3 \end{pmatrix} = \begin{pmatrix} b_1 a_1 & b_1 a_2 & b_1 a_3 \\ b_2 a_1 & b_2 a_2 & b_2 a_3 \\ b_3 a_1 & b_3 a_2 & b_3 a_3 \end{pmatrix}$$

Matrix multiplication

Matrix:

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{pmatrix} \qquad B = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{pmatrix}$$

$$AB = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{pmatrix}$$

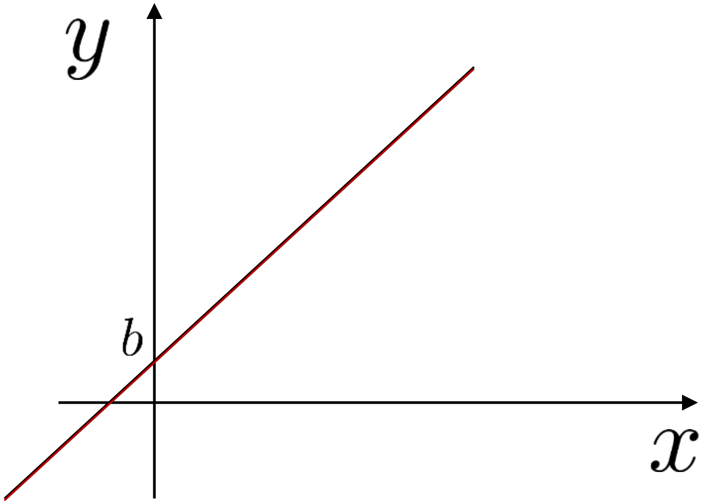
$$= \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} + a_{13}b_{31} & a_{11}b_{12} + a_{12}b_{22} + a_{13}b_{32} \\ a_{21}b_{11} + a_{22}b_{21} + a_{23}b_{31} & a_{21}b_{12} + a_{22}b_{22} + a_{23}b_{32} \end{pmatrix}$$

Calculus

Scalar:

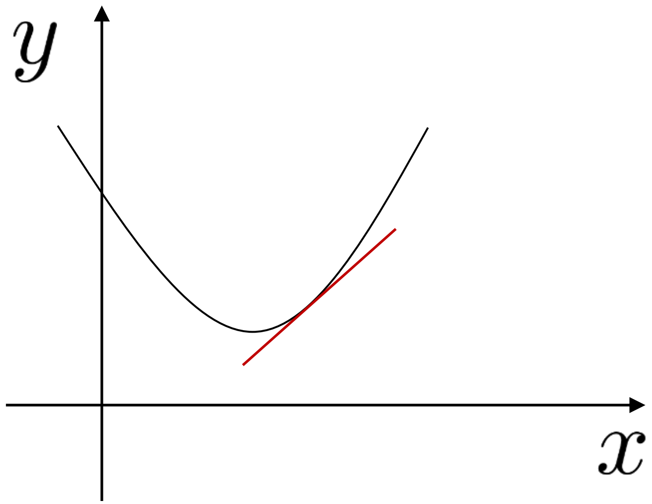
$$y = ax + b$$

$$\frac{dy}{dx} = a$$



$$y = ax^2 + bx + c$$

$$\frac{dy}{dx} = 2ax + b$$



Why Calculus?

Calculus is important in machine learning (ML) is because we are doing “learning”.

In ML, learning refers to the process in which a model is updated/learned to make a better prediction for the data.

The updating/learning process is typically is carried out in an optimization procedure using e.g. the gradient decent algorithm, in which the derivative needs to be computed/attained.

Vector calculus

Vector derivatives

Vector-by-scalar

$$\mathbf{y}(x) = \begin{pmatrix} y_1(x) & y_2(x) & y_3(x) \end{pmatrix}$$

$$\frac{d\mathbf{y}(x)}{dx} = \begin{pmatrix} \frac{dy_1(x)}{dx} & \frac{dy_2(x)}{dx} & \frac{dy_3(x)}{dx} \end{pmatrix}$$

Vector-by-vector

$$\mathbf{y}(\mathbf{x}) = \begin{pmatrix} y_1(\mathbf{x}) & ,\ldots, & y_m(\mathbf{x}) \end{pmatrix}$$

$$\mathbf{x} = \begin{pmatrix} x_1 & ,\ldots, & x_n \end{pmatrix}$$

$$\frac{d\mathbf{y}(\mathbf{x})}{d\mathbf{x}} = \begin{pmatrix} \frac{dy_1(\mathbf{x})}{dx_1} & ,\ldots, & \frac{dy_m(\mathbf{x})}{dx_1} \\ \cdot & \cdot & \cdot \\ \frac{dy_1(\mathbf{x})}{dx_n} & ,\ldots, & \frac{dy_m(\mathbf{x})}{dx_n} \end{pmatrix}$$

Vector calculus

Vector-by-vector

$$A = \begin{pmatrix} a_{11} & ,\ldots, & a_{1m} \\ . & . & . \\ a_{n1} & ,\ldots, & a_{nm} \end{pmatrix}$$

$$\mathbf{x} = \begin{pmatrix} x_1 \\ . \\ x_m \end{pmatrix}$$

$$\frac{\partial A\mathbf{x}}{\partial \mathbf{x}} = A$$

numerator form

$$\frac{\partial \mathbf{x}^T A^T}{\partial \mathbf{x}} = A$$

denominator form

Vector calculus

Identities: vector-by-vector $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$

Condition	Expression	Numerator layout, i.e. by **y** and $\mathbf{x}^T$	Denominator layout, i.e. by $\mathbf{y}^T$ and **x**
a is not a function of **x**	$\frac{\partial \mathbf{a}}{\partial \mathbf{x}} =$	$\mathbf{0}$	
	$\frac{\partial \mathbf{x}}{\partial \mathbf{x}} =$	$\mathbf{I}$	
A is not a function of **x**	$\frac{\partial \mathbf{A}\mathbf{x}}{\partial \mathbf{x}} =$	$\mathbf{A}$	$\mathbf{A}^\top$
A is not a function of **x**	$\frac{\partial \mathbf{x}^\top \mathbf{A}}{\partial \mathbf{x}} =$	$\mathbf{A}^\top$	$\mathbf{A}$
a is not a function of **x**, **u** = **u**(**x**)	$\frac{\partial a\mathbf{u}}{\partial \mathbf{x}} =$	$a\frac{\partial \mathbf{u}}{\partial \mathbf{x}}$	
a = *a*(**x**), **u** = **u**(**x**)	$\frac{\partial a\mathbf{u}}{\partial \mathbf{x}} =$	$a\frac{\partial \mathbf{u}}{\partial \mathbf{x}} + \mathbf{u}\frac{\partial a}{\partial \mathbf{x}}$	$a\frac{\partial \mathbf{u}}{\partial \mathbf{x}} + \frac{\partial a}{\partial \mathbf{x}}\mathbf{u}^\top$
A is not a function of **x**, **u** = **u**(**x**)	$\frac{\partial \mathbf{A}\mathbf{u}}{\partial \mathbf{x}} =$	$\mathbf{A}\frac{\partial \mathbf{u}}{\partial \mathbf{x}}$	$\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\mathbf{A}^\top$
u = **u**(**x**), **v** = **v**(**x**)	$\frac{\partial (\mathbf{u}+\mathbf{v})}{\partial \mathbf{x}} =$	$\frac{\partial \mathbf{u}}{\partial \mathbf{x}} + \frac{\partial \mathbf{v}}{\partial \mathbf{x}}$	
u = **u**(**x**)	$\frac{\partial \mathbf{g}(\mathbf{u})}{\partial \mathbf{x}} =$	$\frac{\partial \mathbf{g}(\mathbf{u})}{\partial \mathbf{u}}\frac{\partial \mathbf{u}}{\partial \mathbf{x}}$	$\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\frac{\partial \mathbf{g}(\mathbf{u})}{\partial \mathbf{u}}$

https://en.wikipedia.org/wiki/Matrix_calculus

Matrix calculus

Matrix-by-scalar

$$Y(x) = \left(\begin{array}{ccc} y_{11}(x) & ,\ldots, & y_{1m}(x) \\ . & . & . \\ y_{n1}(x) & ,\ldots, & y_{nm}(x) \end{array} \right)$$

$$\frac{dY(x)}{dx} = \left(\begin{array}{ccc} \frac{dy_{11}(x)}{dx} & ,\ldots., & \frac{dy_{1m}(x)}{dx} \\ \frac{dy_{n1}(x)}{dx} & ,\ldots., & \frac{dy_{nm}(x)}{dx} \end{array} \right)$$

Matrix calculus

Scalar-by-vector

$$A = \begin{pmatrix} a_{11} & ,\ldots, & a_{1n} \\ . & . & . \\ a_{n1} & ,\ldots, & a_{nn} \end{pmatrix}$$

$$\mathbf{x} = \begin{pmatrix} x_1 \\ . \\ x_n \end{pmatrix}$$

$$\frac{\partial \mathbf{x}^T A \mathbf{x}}{\partial \mathbf{x}} = 2\mathbf{x}^T A$$

Matrix calculus

Condition	Expression	Numerator layout, i.e. by $\mathbf{x}^T$; result is row vector	Denominator layout, i.e. by x; result is column vector
a is not a function of **x**	$\frac{\partial(\mathbf{a}\cdot\mathbf{x})}{\partial\mathbf{x}} = \frac{\partial(\mathbf{x}\cdot\mathbf{a})}{\partial\mathbf{x}} =$ $\frac{\partial\mathbf{a}^\top\mathbf{x}}{\partial\mathbf{x}} = \frac{\partial\mathbf{x}^\top\mathbf{a}}{\partial\mathbf{x}} =$	$\mathbf{a}^\top$	$\mathbf{a}$
A is not a function of **x** **b** is not a function of **x**	$\frac{\partial\mathbf{b}^\top\mathbf{A}\mathbf{x}}{\partial\mathbf{x}} =$	$\mathbf{b}^\top\mathbf{A}$	$\mathbf{A}^\top\mathbf{b}$
A is not a function of **x**	$\frac{\partial\mathbf{x}^\top\mathbf{A}\mathbf{x}}{\partial\mathbf{x}} =$	$\mathbf{x}^\top(\mathbf{A}+\mathbf{A}^\top)$	$(\mathbf{A}+\mathbf{A}^\top)\mathbf{x}$
A is not a function of **x** **A** is symmetric	$\frac{\partial\mathbf{x}^\top\mathbf{A}\mathbf{x}}{\partial\mathbf{x}} =$	$2\mathbf{x}^\top\mathbf{A}$	$2\mathbf{A}\mathbf{x}$
A is not a function of **x**	$\frac{\partial^2\mathbf{x}^\top\mathbf{A}\mathbf{x}}{\partial\mathbf{x}^2} =$	$\mathbf{A}+\mathbf{A}^\top$	
A is not a function of **x** **A** is symmetric	$\frac{\partial^2\mathbf{x}^\top\mathbf{A}\mathbf{x}}{\partial\mathbf{x}^2} =$	$2\mathbf{A}$	

https://en.wikipedia.org/wiki/Matrix_calculus

Vector and Matrix Calculus

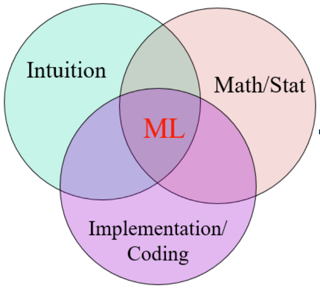
The main reason to study/use vector calculus is to formulate machine learning problems using canonical mathematical representations that can be accepted by the generic machine learning algorithms including neural networks, decision tree, nearest neighborhood, boosting, logistic regression classifier etc.

Using vector representations with vector calculus significantly facilitates the understanding, training, evaluating, scaling up, and transferring of the machine learning algorithms with significantly reduced overhead and customization.

Vector and Matrix Calculus

One way to master the concept of vector representation and calculus is by simplifying (conceptually) the formulation into scalar cases, which can be understood more easily.

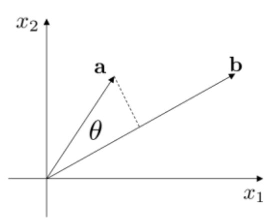
Taking the derivatives w.r.t. scalar and vector gives us a strong leverage in using vector calculus for performing optimization to train the various machine learning algorithms.



Recap: Vector Calculus

Intuition: Both the input data and the model parameters are represented as vectors in high dimensional spaces. Using vector calculus allows us to apply mathematical operations on the vectors to train a model, make an estimation, and make a prediction using principled and sound mathematical/statistical formulations that can scale.

Math:



$$\mathbf{a} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

$$<\mathbf{a},\mathbf{b}> \; \equiv \mathbf{a}\cdot\mathbf{b} \; \equiv \mathbf{a}^T\mathbf{b} \; \equiv a_1b_1 + a_2b_2 + a_3b_3$$

$$cos(\theta) = \frac{<\mathbf{a},\mathbf{b}>}{||\mathbf{a}||_2 \times ||\mathbf{b}||_2}$$

Summary of the classification problem

$\mathbf{x} = (x_1, x_2, \ldots)$

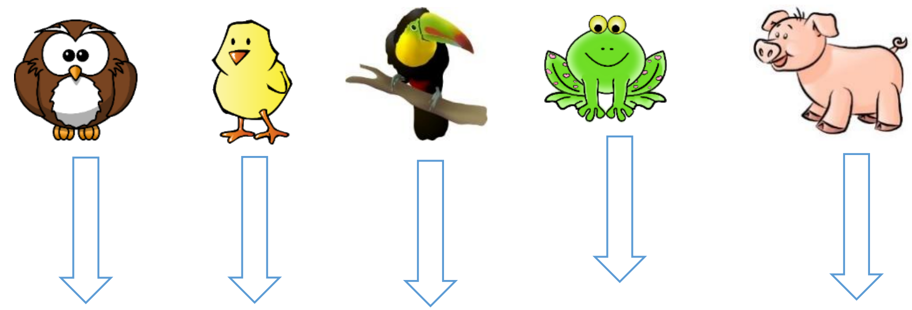
$y = 1(bird)$

x_1: color

x_2 weight

$\ldots$

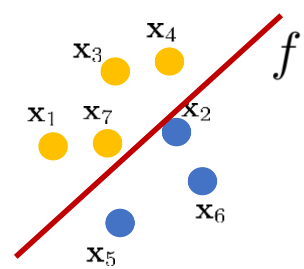
$$S_{training} =$$



$$\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_3, y_3), (\mathbf{x}_4, y_4), (\mathbf{x}_5, y_5)\}$$

Train classifier $f(\mathbf{x}; W)$

W: model parameter



$f(\mathbf{x}; W)$

Basic concepts (supervised)

$$\mathbf{x}_i = (x_{i1}, ..., x_{im}), x_{ij} \in R, \qquad \mathbf{x} \in R^m$$

$$y_i \in R$$

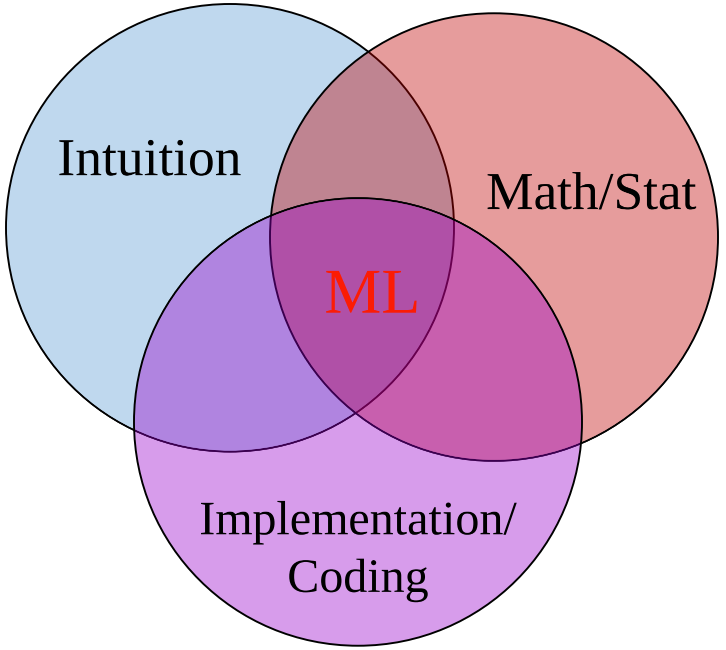
Training (supervised)

$$S_{training} = \{(\mathbf{x}_i, y_i), i = 1..n\}$$

blood presure	age	male or female	weight (lb)	height (cm)
y_1=131	$x_{11} = 22$	$x_{12} = M$	$x_{13} = 160$	$x_{14} = 180$
y_2=150	$x_{21} = 51$	$x_{22} = M$	$x_{23} = 190$	$x_{24} = 175$
y_3=105	$x_{31} = 43$	$x_{32} = F$	$x_{33} = 120$	$x_{34} = 165$

In supervised setting during training, y_i (the solution) to each sample x_i is provided.

A Big Picture



To do well in machine learning:

Intuition + Math/Stat + Implementation/Coding

Now, let's look at machine learning from a more systematic way.

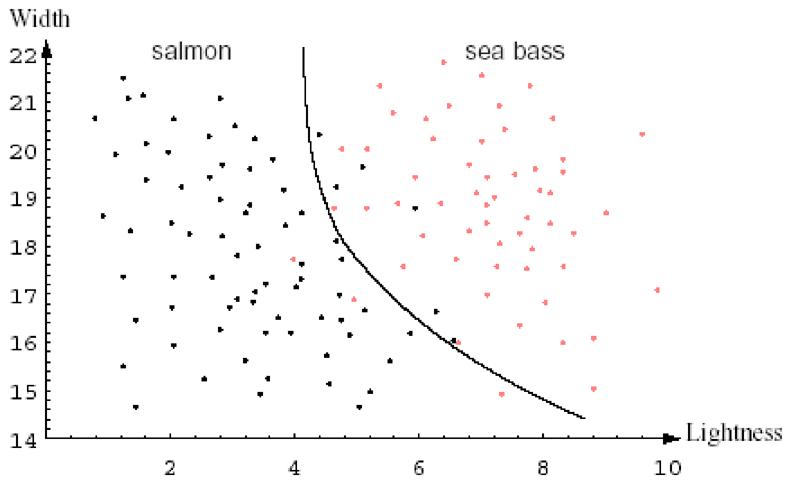
Some Learned Lessons (Pedro Domingos)

Table 1. The three components of learning algorithms.

Representation	Evaluation	Optimization
Instances	Accuracy/Error rate	Combinatorial optimization
K-nearest neighbor	Precision and recall	Greedy search
Support vector machines	Squared error	Beam search
Hyperplanes	Likelihood	Branch-and-bound
Naive Bayes	Posterior probability	Continuous optimization
Logistic regression	Information gain	Unconstrained
Decision trees	K-L divergence	Gradient descent
Sets of rules	Cost/Utility	Conjugate gradient
Propositional rules	Margin	Quasi-Newton methods
Logic programs		Constrained
Neural networks		Linear programming
Graphical models		Quadratic programming
Bayesian networks		
Conditional random fields		

Error

Now, let $f(\mathbf{x};\ \mathbf{W})$ be one classifier which makes the prediction for the label y, we define the error on a set of input as:



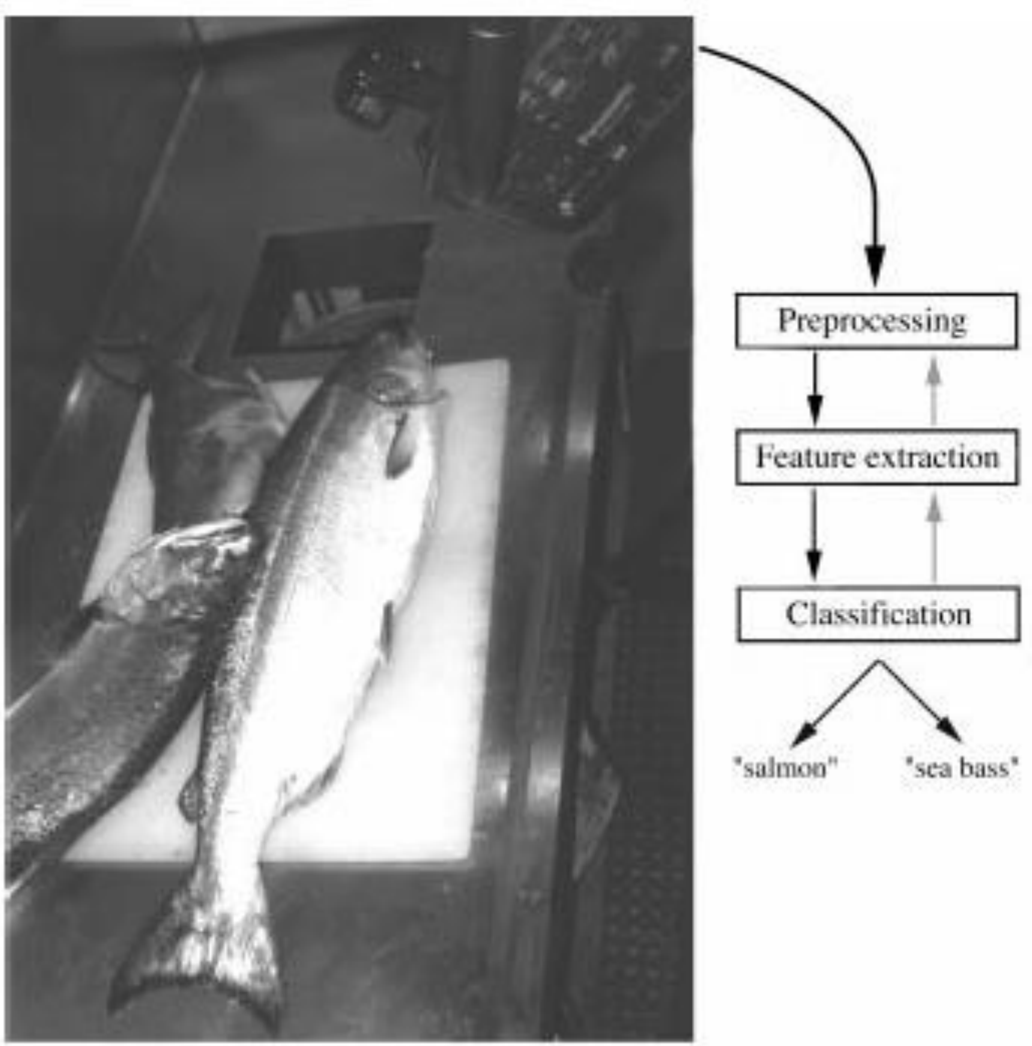
$$S_{training} = \{(\mathbf{x}_i, y_i), i = 1..n\}$$

$$e_{training} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(y_i \neq f(\mathbf{x}_i; W)) \qquad \mathbf{1}(z) = \begin{cases} 1 & if\ z = TRUE \\ 0 & otherwise \end{cases}$$

Error Metrics and Object Functions

- One thing that separates modern machine learning from the efforts in traditional AI is the establishment of benchmarks under widely accepted common evaluation metrics.
- Being able to faithfully compare the performances of different machine learning algorithms/systems significantly propel the advancement of machine learning field.
- In addition, establishing a clear objective function (errors + regularization) to optimize when training machine learning algorithms is a key reason for the success of modern machine/deep learning.

An example



Preprocessing
Feature extraction
Classification
"salmon"
"sea bass"

Summary of the problem

Let **x** be the input vector (observation) and y be its label:

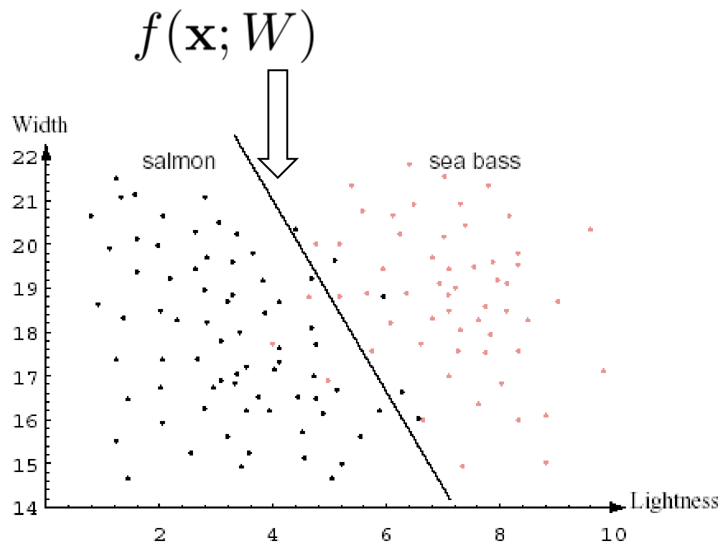
Often, we are given a set of training data

$$S_{training} = \{(\mathbf{x}_i, y_i), i = 1..n\} \qquad \mathbf{x} = (x_1, ..., x_m), x_i \in \mathbb{R}, \quad \mathbf{x} \in \mathbb{R}^m$$

$y \in \{0, 1\}$ $\qquad y = 0$: negative $\qquad y = 1$: positive

We also alternatively use $y \in \{-1, +1\}$ $\qquad y = -1$: negative $\qquad y = +1$: positive

We use the training set to train a classifier $f(\mathbf{x}; W)$.



Summary of the problem

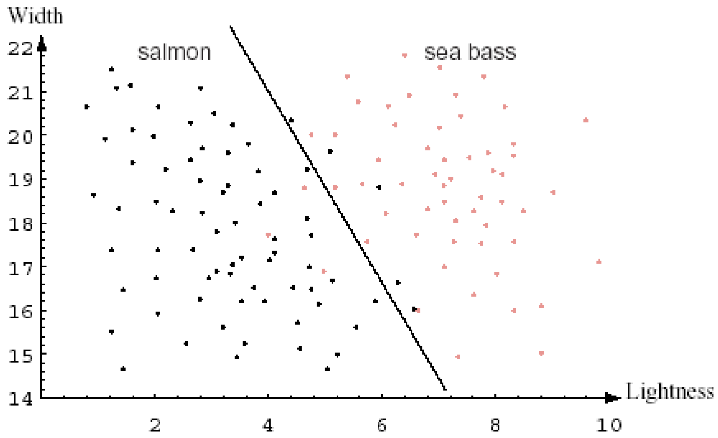
$$S_{training} = \{(\mathbf{x}_i, y_i), i = 1..n\}$$

$\mathbf{x} = (x_1, ..., x_m), x_i \in \mathbb{R}, \quad \mathbf{x} \in \mathbb{R}^m$

$y \in \{0, 1\}$ $\quad y = 0$: negative
$\quad y = 1$: positive

Classifier: $f(\mathbf{x}; W) \in \{0, 1\}$

Model parameter to be learned: W



Training error:

$$e_{training} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(y_i \neq f(\mathbf{x}_i; W))$$

Training errors

If we ignore the overfitting problem for now, nearly all supervised classifiers are learned by minimizing the training error (either implicitly or explicitly)

$$\text{Minimize } e_{training}$$

The definition of training error varies depending on the types of classifier.

Typically:

$$\text{Minimize } \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}(y_i \neq f(\mathbf{x}_i; W))$$

Question?

Given an input set and a choice of classifier:

$$S_{training} = \{(\mathbf{x}_i, y_i), i = 1..n\}$$

Classifier: $f(\mathbf{x}; W) \in \{0, 1\}$

If we try our best, what is the worst possible training error (assuming equal number of positives and negatives for binary classification)?

$$e_{training} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(y_i \neq f(\mathbf{x}_i; W))$$

A. 0
B. 0.25
C. 0.5
D. 0.75
E. 1.0

Question?

Given a input set and a choice of classifier:

$$S_{training} = \{(\mathbf{x}_i, y_i), i = 1..n\}$$

Classifier: $f(\mathbf{x}; W) \in \{0, 1\}$

If we try our best, what is the worst possible training error (assuming equal number of positives and negatives)?

$$e_{training} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}(y_i \neq f(\mathbf{x}_i; W))$$

A. 0
B. 0.25
C. 0.5
D. 0.75
E. 1.0

Question?

$$S_{training} = \{(\mathbf{x}_i, y_i), i = 1..n\}$$

Classifier: $f(\mathbf{x}; W) \in \{0, 1\}$

$$e_{training} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(y_i \neq f(\mathbf{x}_i; W))$$

$e_{training} \leq 0.5$, if we try our best

Why?

If $e_{training} > 0.5$

we simply define $f'(\mathbf{x}_i; W) = 1 - f(\mathbf{x}_i; W)$