UNIVERSITAT DE BARCELONA

Week 6

Course. Introduction to Machine Learning

Theory 6. A gentle introduction to Supervised Learning

Dr. Maria Salamó Llorente
maria.salamo@ub.edu
Dept. Mathematics and Informatics,
Faculty of Mathematics and Informatics,
University of Barcelona (UB)

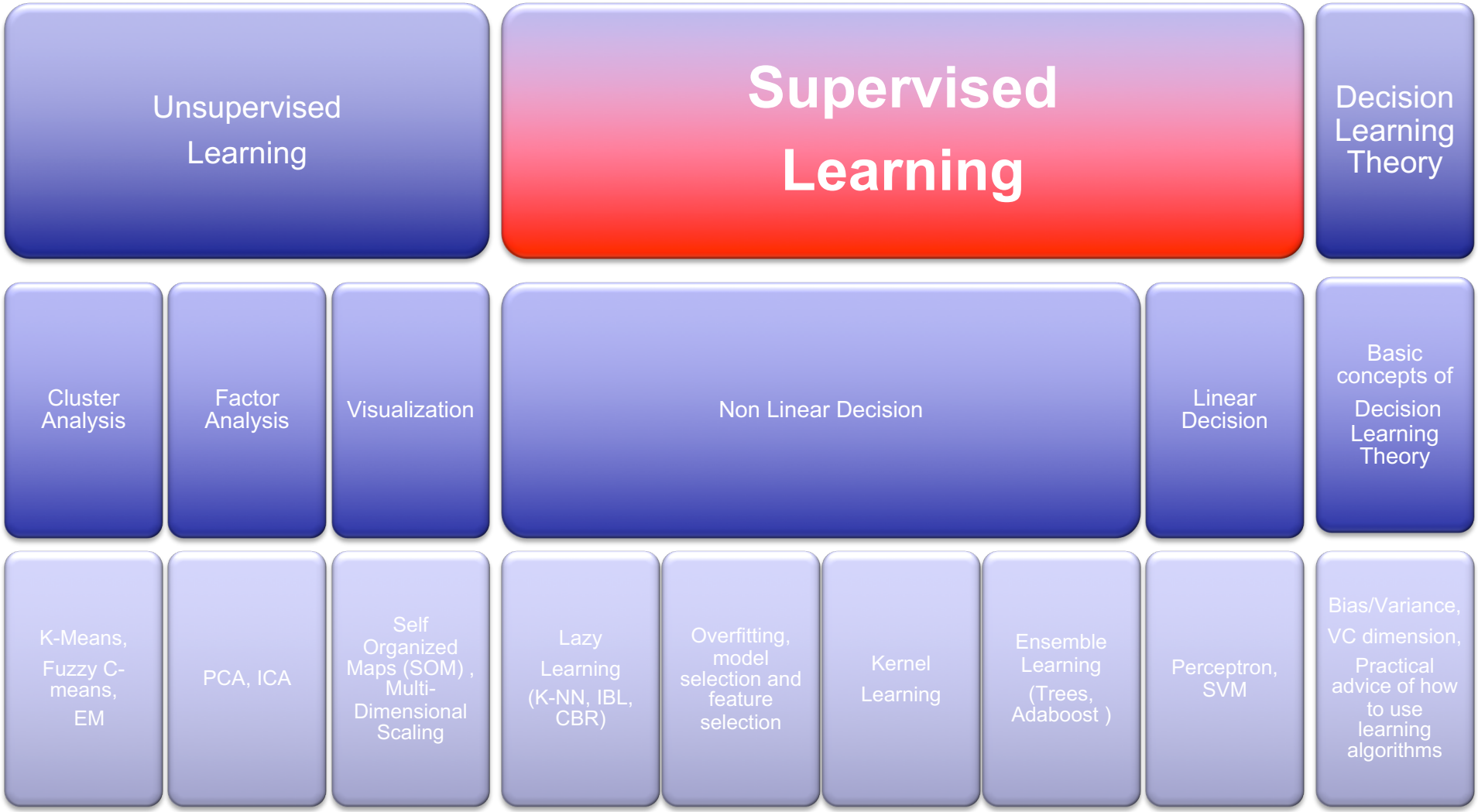
Acknowledgements

This lesson is inspired in the courses of:

- T. Jaakkola, M. Collins, L. Kaelbling, and T. Poggio at MIT
- Andrew Ng at Stanford
- Y. Abu-Mostafa at CalTech
- E. Xing at CMU
- P. Domingos at Georgia

This slides partially come from Oriol Pujol. He was teaching this course during 5 years with me and Machine Learning is one of his passions

Introduction to Machine Learning



Contents

1. What is supervised learning?
2. An introductory example. The linear regression model
3. Traveling across the parameter space. Descent optimization methods
4. Application of the learning model
5. A whirlwind tour on machine learning terms

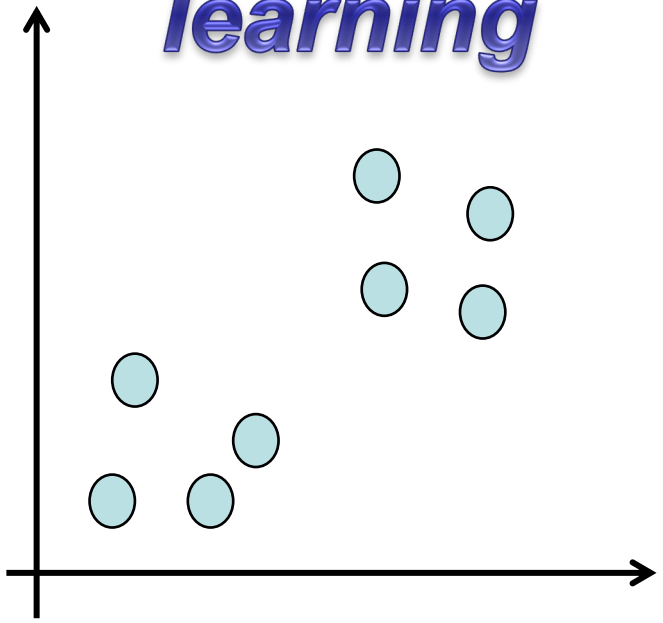
What is supervised learning?

What is supervised learning?

Supervised learning

Inferring a function from labelled training data

Unsupervised learning



Try to find hidden structure in unlabeled data

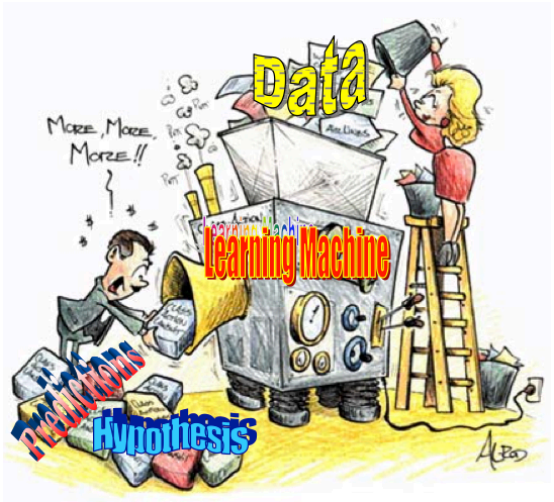
https://youtu.be/bQI5uDxrFfA

What is supervised learning?

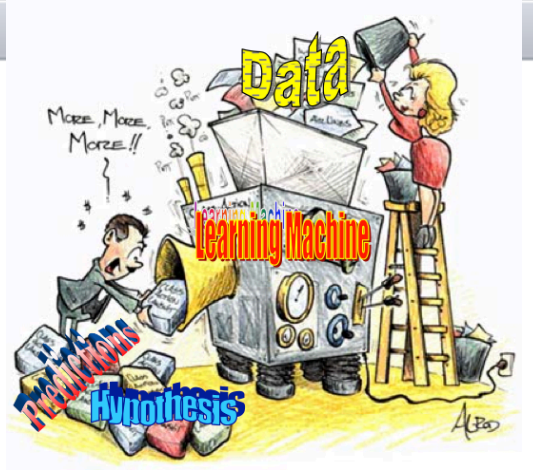
Improve the performance of a software system, based on previous experience

Tasks:

- **prediction**: phone and credit card fraud, medical diagnosis, document retrieval,
- **understanding**: market basks, discovering astronomical features
- **control**: flying helicopters,
  playing backgammon



Key elements for machine learning



Machine learning is used when:

- There is a pattern
- We can not pin it down mathematically
- We have data on it

Which is the most important of the three?

That's why we also call it **Learning from data**

What is Machine Learning?

Improve the performance of a software system, based on previous experience:

- **prediction**: **supervised learning**: given *(x, y)* pairs, find a mapping from a new *x* to a new *y*, e.g., regression, classification
- **understanding**: **unsupervised learning**: given a set of x, find something interesting or useful about their structure, e.g. density estimation, clustering, dimensionality reduction
- **control**: **reinforcement learning**: given an external system upon which you can exert control action *a* and receive percepts, *p*, a reward signal *r* indicating good performance, find a mapping from *P* →*A* that maximizes some long-term measure of *r*

Generalization in Supervised Learning

We want to find a model that will perform the best on future examples:

- We don't know what the future data will be
- We have **some past data**
- We **hope** that future data will remember past data in a way that while let us use the past data to construct a model that will perform well in the future

Framing the problem

Humans have to do a lot of work, up front, to set up a machine learning problem

- What do you want to predict?
- What kind of data can you get?
- What is the relative costs of different types of errors?
- How does the available data relate to future data?

In order to setup a problem, you first need to think about all of these questions

Examples

- Regression :
  - Given a picture of a person, we have to predict their age on the basis of the given picture
  - The output is continuous
- Classification:
  - Given a patient with a tumor, we have to predict whether the tumor is malignant or benign
  - The output is discrete

An introductory example

An example

Suppose you want to know which grade I may obtain at the end this course

If you know information (students record) about people who passed this course (data *x*), and their performance (label *y*), then, given your own record you may ask the system to predict your grade based on the previous experience and the power of the learning algorithm

An example

Av. math grade	Grade in ML
5.0	4.2
6.2	5.9
7.4	8.1
⋮	⋮

Supervised learning:
Given the "right" answer for each example in de data.

Regression problem:
Predict a real-value output.

Notation

Input (features): $\mathbf{x} \in \mathbf{R}^d$ (student's record)
Output (labels): $y \in \mathbf{R}$ (actual grade in ML)
Data: Examples of inputs and output pairs: $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$

Dataset jargon

We will usually use column-wise notation for each sample.

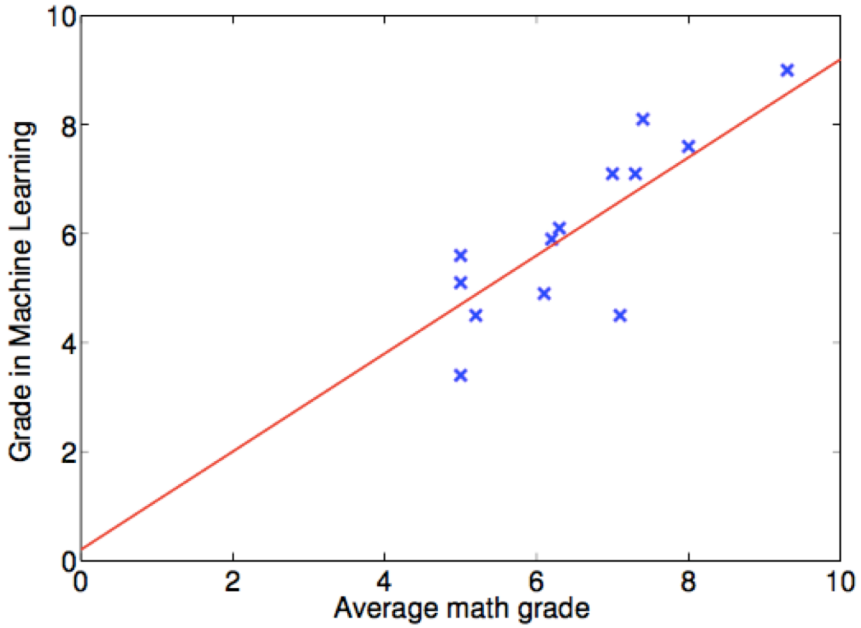
Common jargon:

Rows: features/ attributes/ dimensions.
Columns: instances/ examples/ samples.
The feature to be predicted: target/ outcome/ response/ label/ dependent variable.
The other features: independent variables/ covariates/ predictors/ regressors.

According to the type of data, we can talk about:

- iid (independent identically distributed) vectors
- Time series (dependent vectors)
- Images (matrices)
- Variable-size non-vector data (e.g. strings, trees, graphs, text)
- Objects (e.g. within a relational schema)

An example



Hypothesis: $f : \mathbf{R}^d \rightarrow \mathbf{R}$ A model that maps from the input data to the output label, e.g. $f(x) = 0.9x + 0.5$

Learning algorithm: The process for selecting the most appropriate hypothesis from a hypothesis set $\mathcal{H}$, e.g. the set of linear models $\mathcal{H}(w_0, w_1) \equiv w_1 x + w_0$, where (w_0, w_1) are called parameters.

An example (cont.)

1) The question I would like to answer to the system is

- Given my own record (e.g., 6.4), which is the expected grade in ML I will obtain?

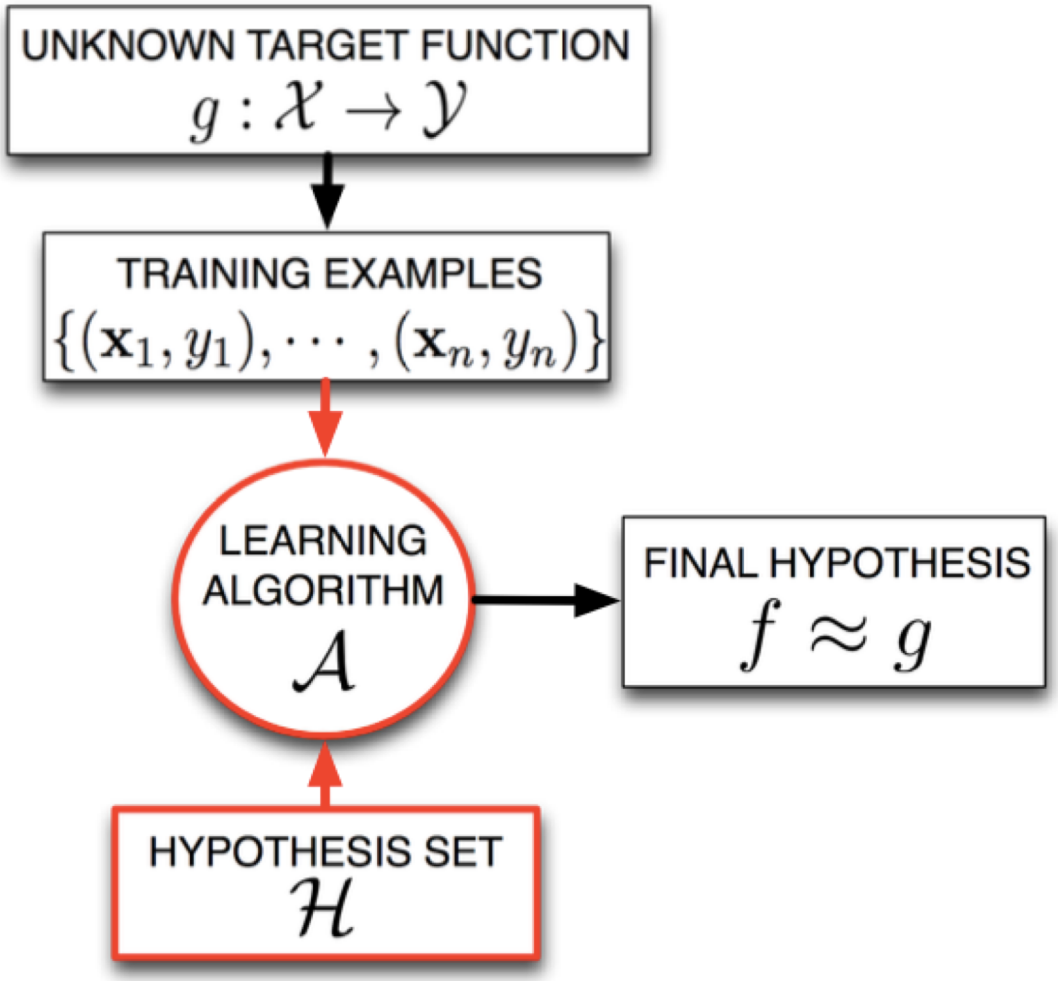
2) So, one possible solution is to model data with a LINEAR MODEL

- And then ask the system which grade I should expect

3) We need to solve the problem

- The selection of a model from a hypothesis space

The supervised classification problem



Elements of the supervised problem

Data:

Training: $\{(\mathbf{x}_1, y_1) \dots (\mathbf{x}_N, y_N)\}; \quad (\mathbf{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$

where

$\mathbf{x}$ is the feature data set.

$\mathbf{y}$ is the target output / label.

Model:

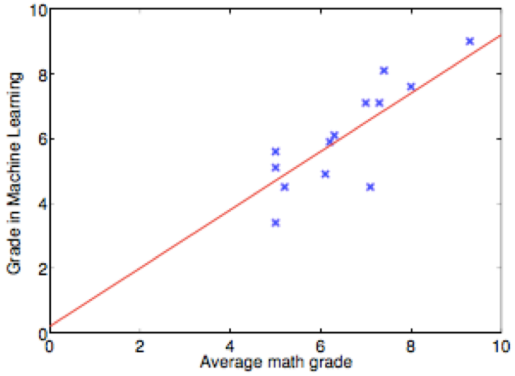
- Hypothesis set: The set of possible / candidate models, $\mathcal{H}$.
- Learning algorithm: The method for selecting the most appropriate model candidate (f) with respect to some quality function (e.g. model accuracy)

Output: The selected model, $f \in \mathcal{H}$.

1st supervised learning model: Linear regression

Univariate linear regression hypotheses set

$$\mathcal{H}(w_0, w_1) \equiv w_1 x + w_0.$$



Learning process idea:

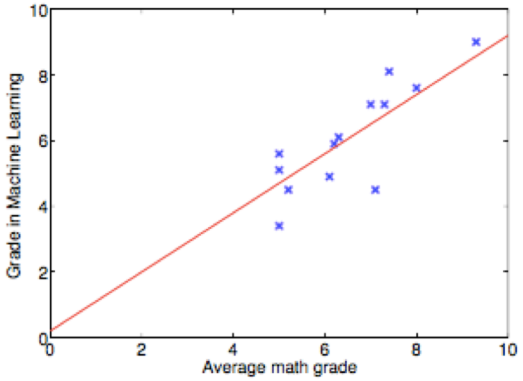
We want to find the model defined by the parameters (w_0, w_1) so that our prediction $f(x_i; w_0, w_1)$ on sample x_i is as close as possible to y_i. This is, the "distance" between $f(x_i; w_0, w_1)$ and y_i is minimum for all elements in the training set.

$$\underset{w_0, w_1}{\text{minimize}} \frac{1}{N} \sum_{i=1}^{N} (f(x_i; w_0, w_1) - y_i)^2$$

1st supervised learning model: Linear regression

Univariate linear regression hypotheses set

$$\mathcal{H}(w_0, w_1) \equiv w_1 x + w_0.$$



The last ingredient:

The cost function/ loss function/ error measure:

$$\underset{w_0, w_1}{\text{minimize}}\ J(w_0, w_1)$$

Putting all the elements together

Hypothesis:

$$f(x, w) = w_1 x + w_0$$

Parameters:

$$w = (w_0, w_1)^T$$

Cost function:

$$\mathcal{J}(w_0, w_1) = \frac{1}{N} \sum_{i=1}^{N} (f(x_i; w_0, w_1) - y_i)^2$$

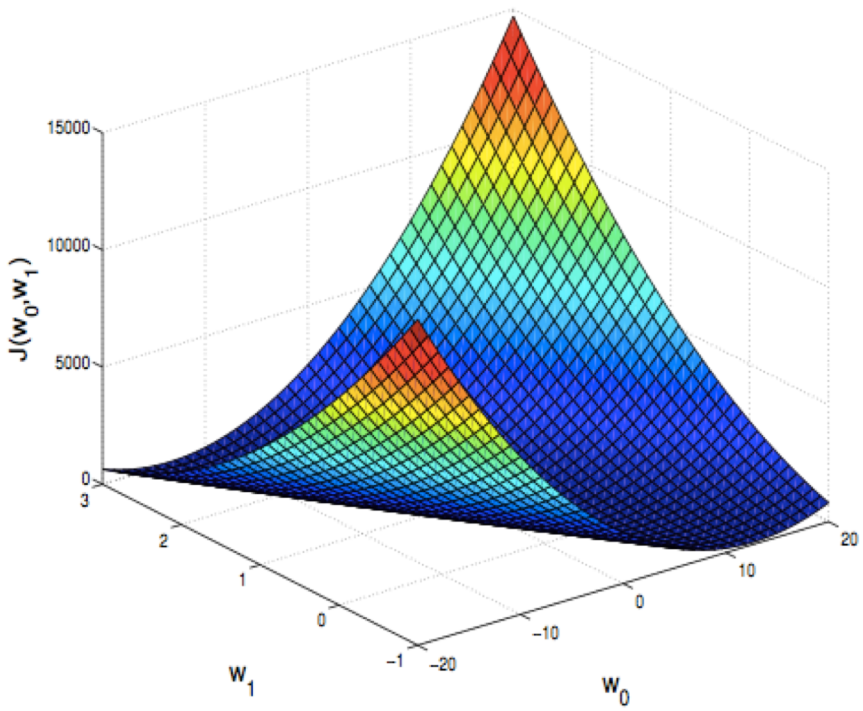
Goal:

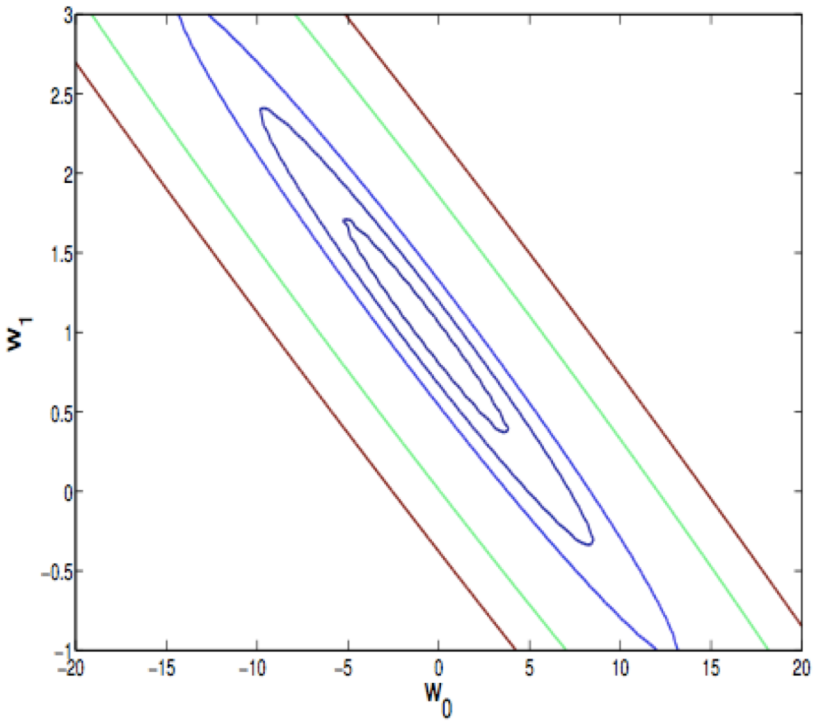
$$\underset{w_0, w_1}{minimize}\ \mathcal{J}(w_0, w_1)$$

Traveling across the parameter space

Descent optimization methods

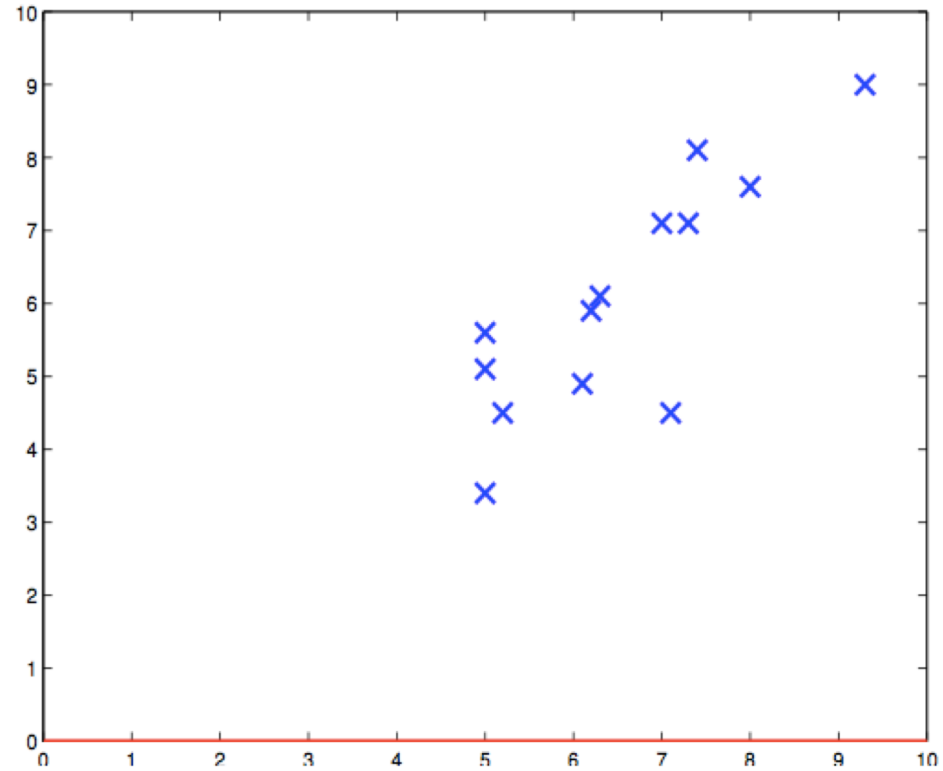
Visualizing the parameter space



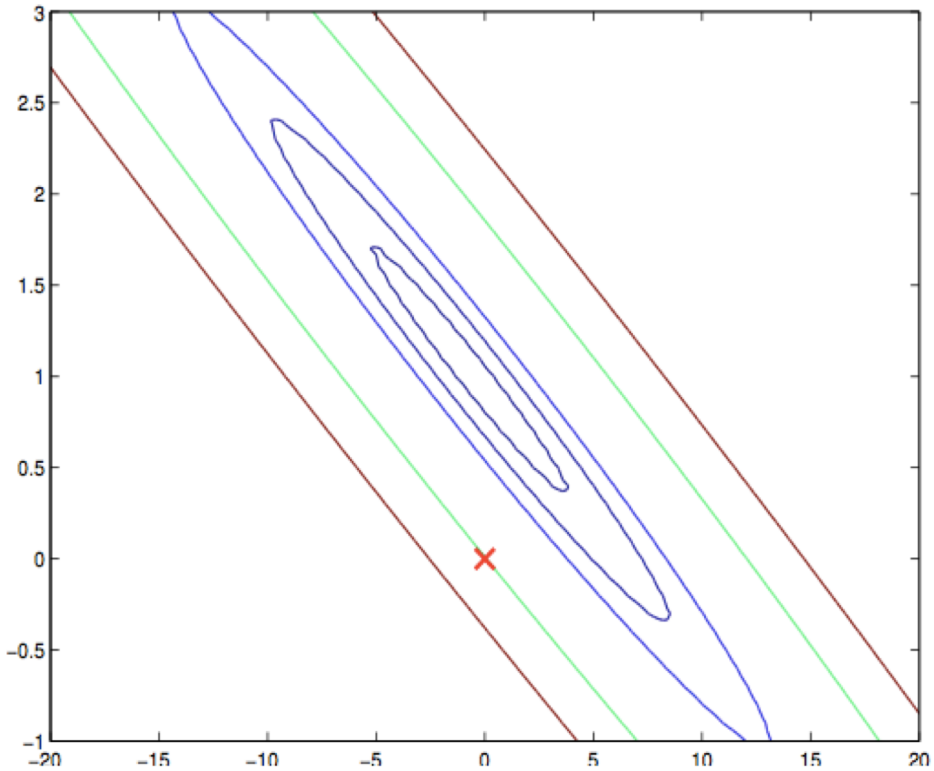


Visualizing the parameter space

$f_{\mathbf{w}}(x)$



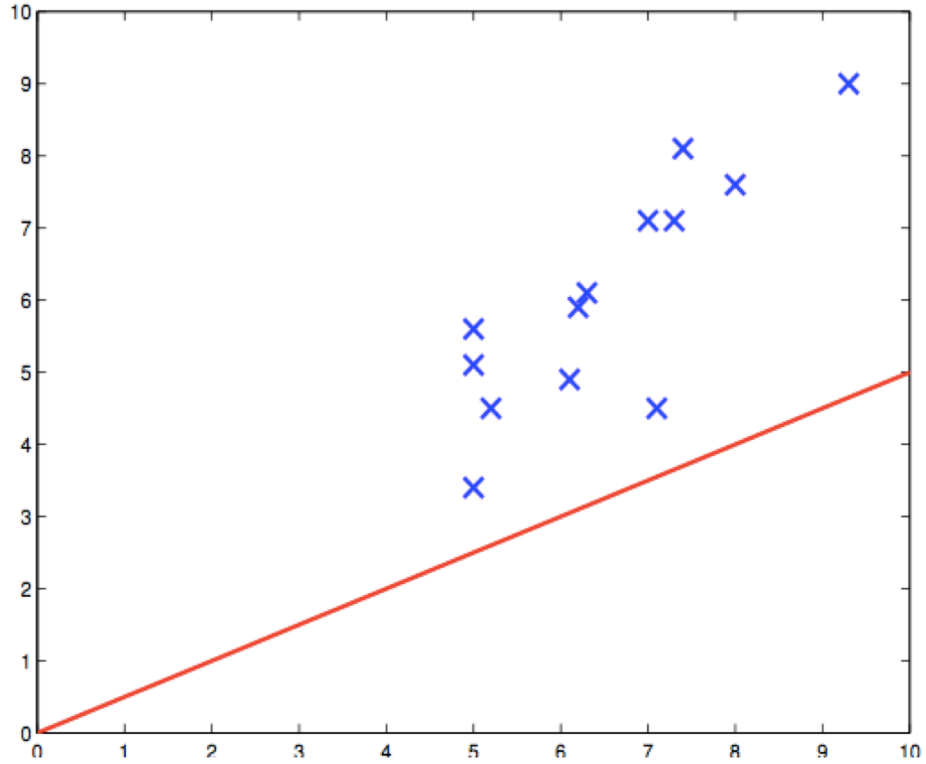
$J(\mathbf{w})$



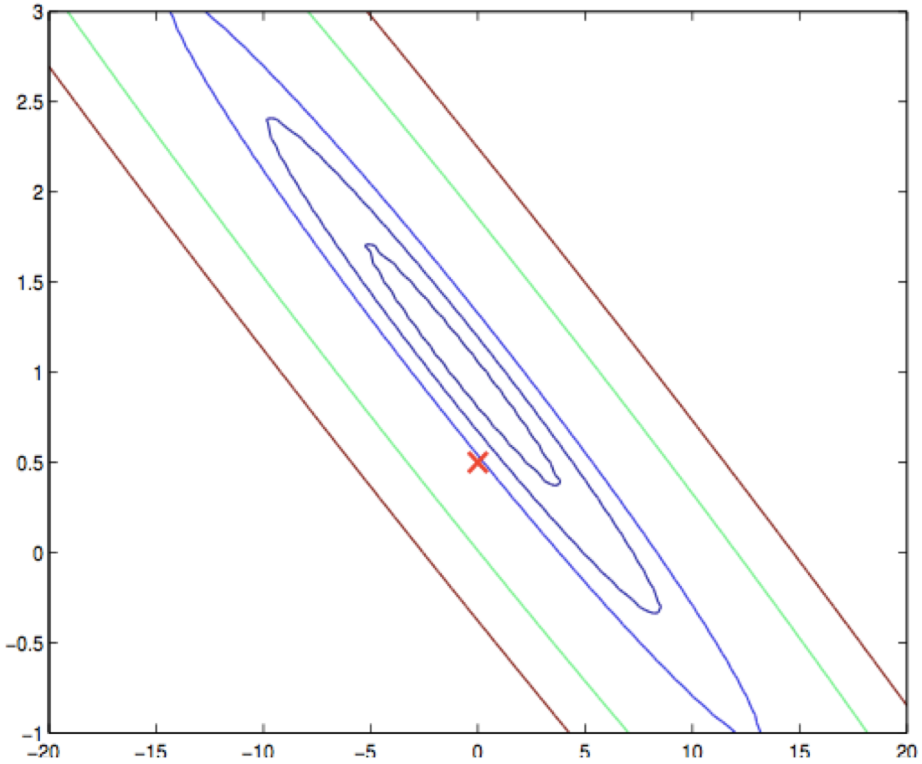
$w_0 = 0, \; w_1 = 0$

Visualizing the parameter space

$f_{\mathbf{w}}(x)$



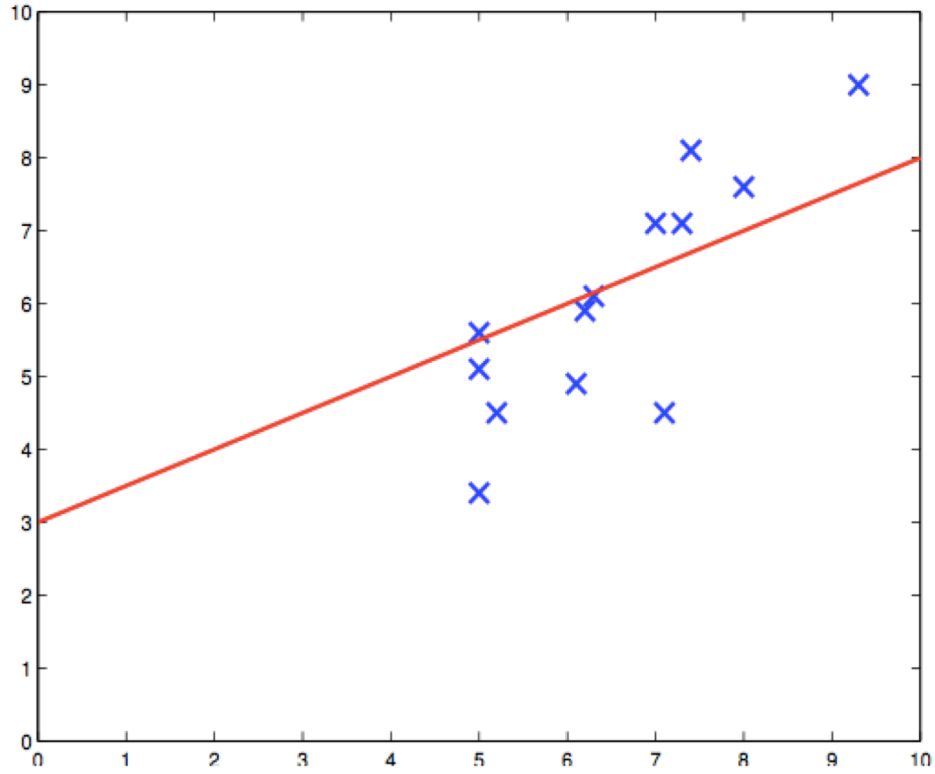
$J(\mathbf{w})$



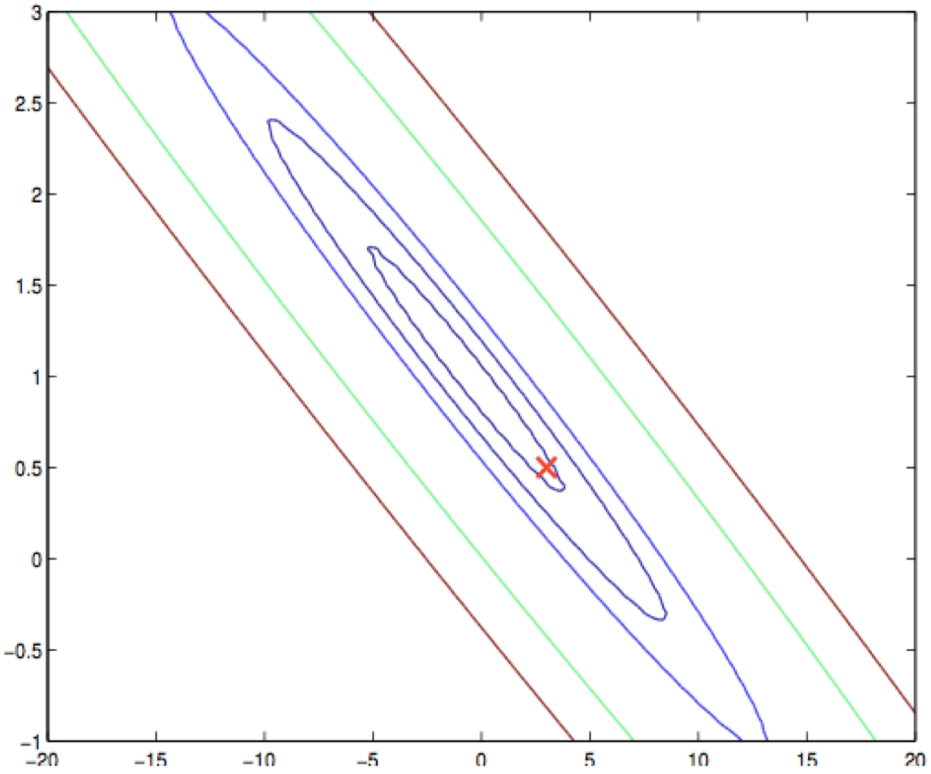
$w_0 = 0, \; w_1 = 0.5$

Visualizing the parameter space

$f_{\mathbf{w}}(x)$



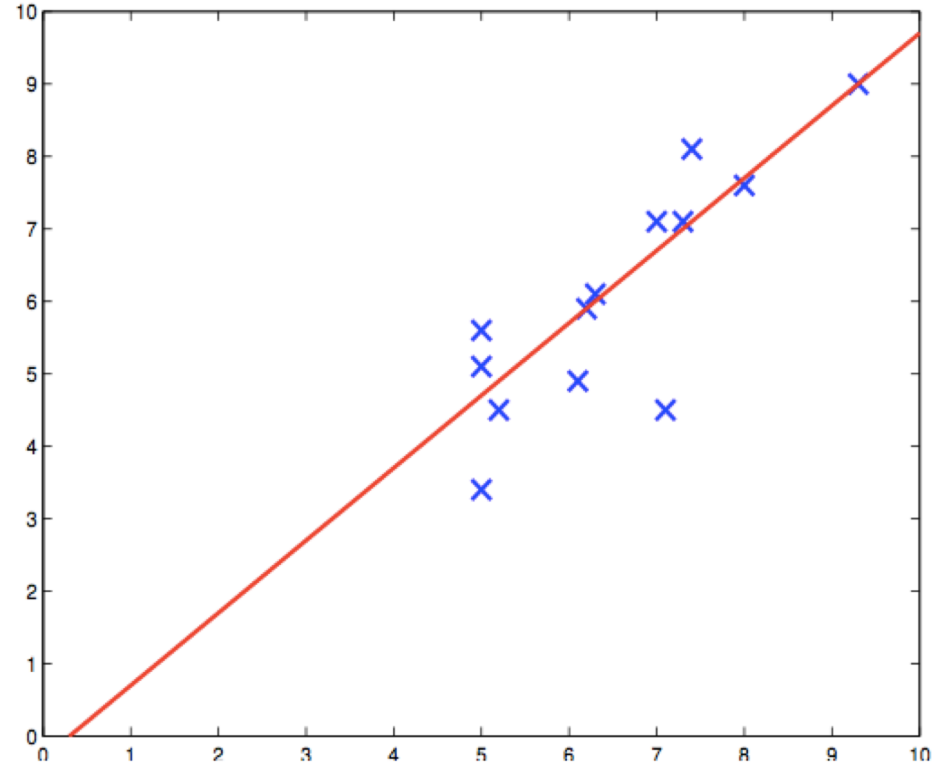
$J(\mathbf{w})$



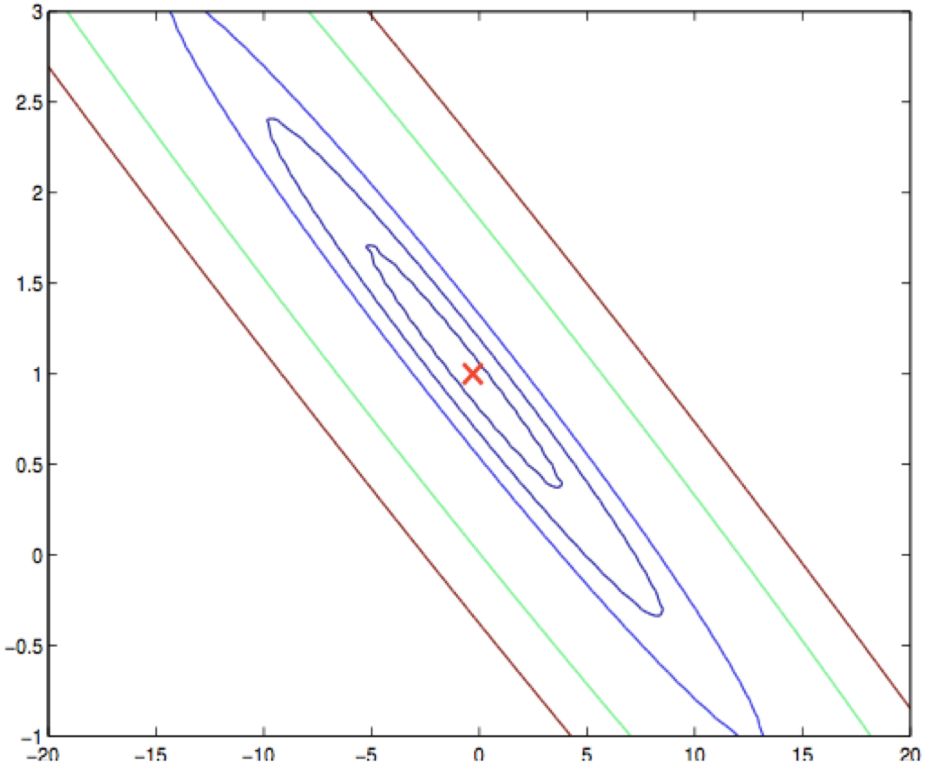
$w_0 = 3, \; w_1 = 0.5$

Visualizing the parameter space

$f_{\mathbf{w}}(x)$



$J(\mathbf{w})$



$w_0 = -0.3, \; w_1 = 1$

Traveling across the parameter space

Question:

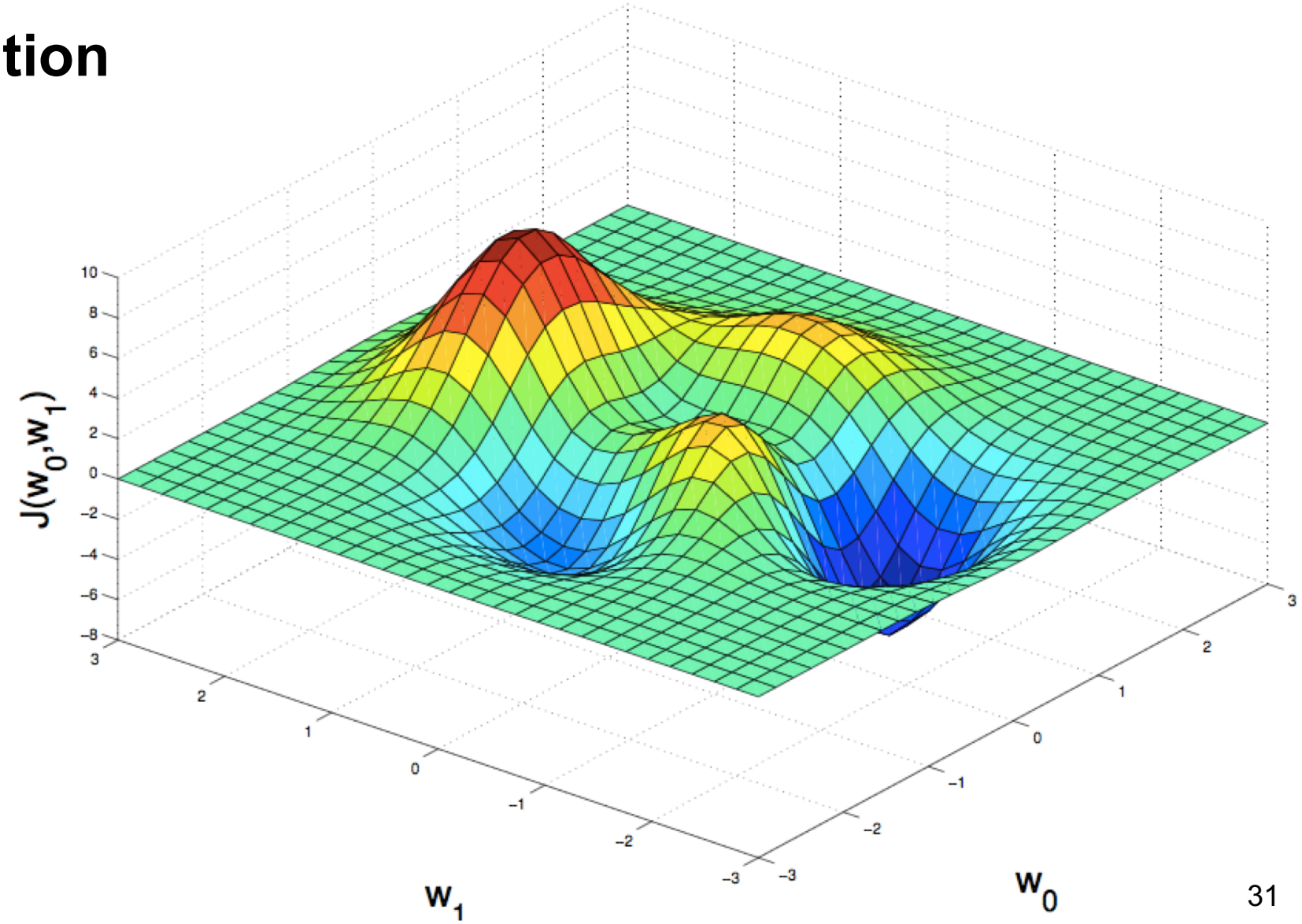
How do we find the minimum in an automated way?

A basic technique:

1. Start with some starting point (guess) $(w_0^{(0)}, w_1^{(0)})$.
2. Change the parameters so that we reduce the cost function $J(w_0^{(k+1)}, w_1^{(k+1)}) < J(w_0^{(k)}, w_1^{(k)})$
3. Repeat (2) until the cost function is not reduced anymore.

Traveling across the parameter space

Intuition



Traveling across the parameter space

With descent methods

Input: given a starting point $x \in$ **dom** f
Output:
while *stopping criterion is not satisfied:* **do**
    (a) Determine a descent direction Δx;
    [(b) *Line search.* Choose a step size $t > 0$];
    (c) *Update.* $x := x + t\Delta x$;
end

Algorithm 1: General descent method

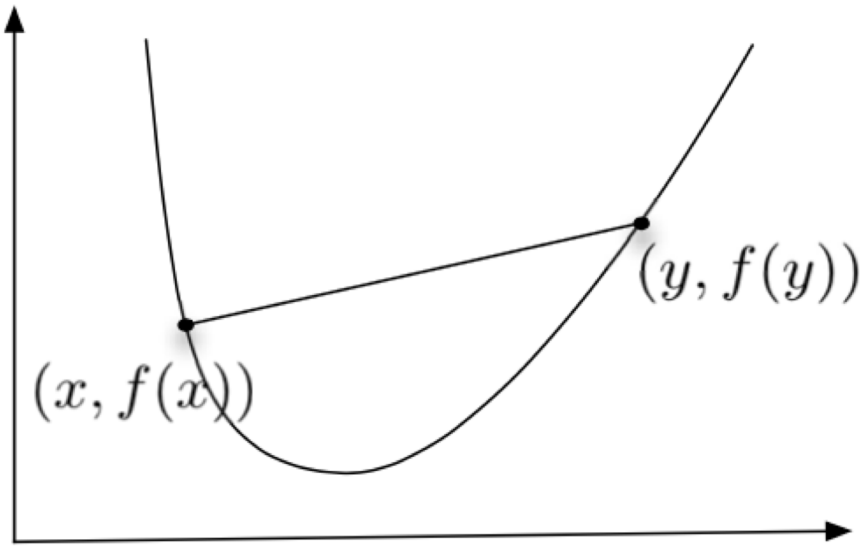
Traveling across the parameter space

With descent methods. Previous Concepts.

Previous concepts

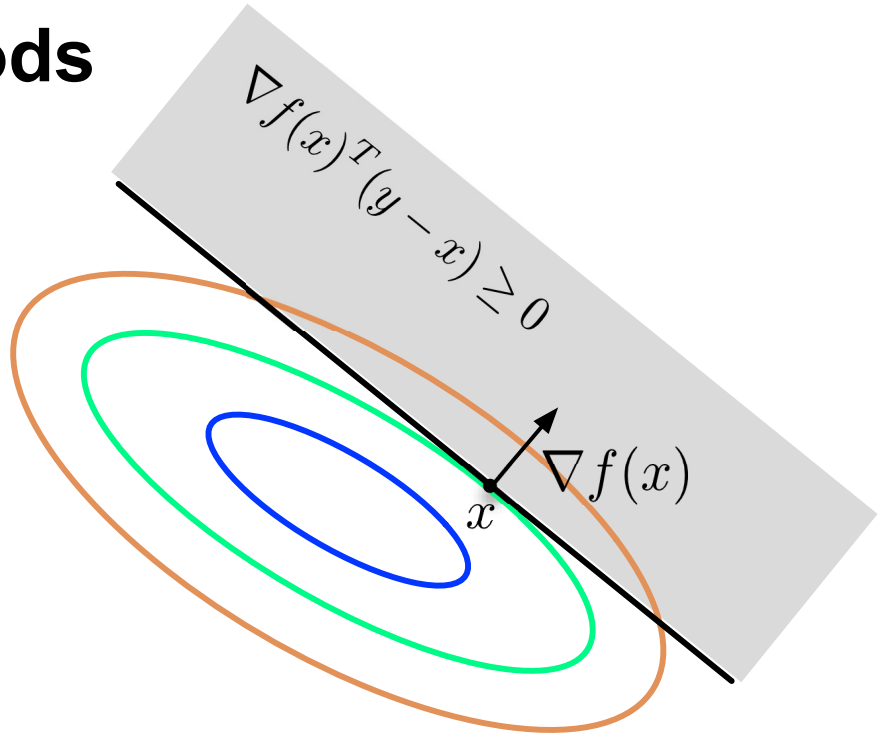
Definition A function $f : \mathbf{R}^n \to \mathbf{R}$ is *convex* if **dom** f is a convex set and if for all $x, y \in$ **dom** f, and θ with $0 \leqslant \theta \leqslant 1$, we have

$$f(\theta x + (1-\theta)y) \leqslant \theta f(x) + (1-\theta)f(y).$$



Traveling across the parameter space

With descent methods



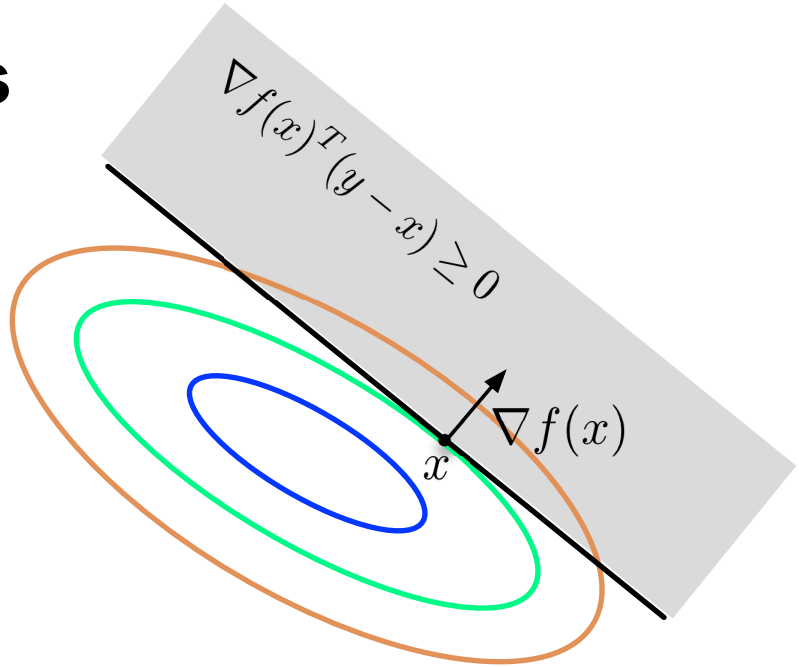
Previous concepts

A differentiable function $f : \mathbf{R}^n \to \mathbf{R}$ is *convex* if and only if **dom** f is convex and

$$f(y) \geqslant f(x) + \nabla f(x)^T(y-x), \quad \forall x, y \in \mathbf{dom}\ f.$$

Traveling across the parameter space

With descent methods



A descent direction must satisfy (necessary but not sufficient):

$$\nabla f(x)^T \Delta x < 0.$$

A natural choice is to select the steepest descent direction given by

$$\Delta x = -\nabla f(x).$$

Traveling across the parameter space

Explanation of the previous slide

- **The gradient of a function is the multi-variate extension of the derivative for a single variable**
- Observe that the optimum is located in the other half space
- Note that not all direction on that space are descent directions
- Observe that we can find a direction with higher value pointing to any point between the green and orange level sets
- Observe that the direction with steepest descent is precisely $\Delta x = -\nabla f(x)$

Gradient/Steepest Descent

Input: given a starting point $x \in$ **dom** f
Output:
while *stopping criterion is not satisfied:* **do**
    (a) Determine a descent direction $\Delta x = -\nabla f(x)$;
    [(b) *Line search.* Choose a step size $t > 0$];
    (c) *Update.* $x := x + t\Delta x$;
end

Algorithm 2: Gradient descent method

Gradient/Steepest Descent

- Gradient descent is an algorithm to minimize any function or parameter
  - It can be used in Linear Regression but is actually used in several ML algorithms
- *t* is the learning rate
  - If *t* is too small, gradient descent will be slow
  - If *t* is too large, gradient descent can overshoot the minimum. It may fail to converge or even diverge
  - There is no need to decrease t over time
  - Gradient descent can converge even if *t* is fixed

Steepest descent implementation

Hypothesis:

$$f(x, w) = w_0 + w_1 x$$

Cost function:

$$\mathcal{J}(w_0, w_1) = \frac{1}{2N} \sum_{i=1}^{N} (f(x_i; w_0, w_1) - y_i)^2$$

Gradient:

$$\nabla J(\mathbf{w}) = \left(\frac{\partial J}{\partial w_0}, \frac{\partial J}{\partial w_1}\right)^T$$

$$\frac{\partial J}{\partial w_0} = \frac{1}{N} \sum_{i=1}^{N} (w_0 + w_1 x_i - y_i) 1$$

$$\frac{\partial J}{\partial w_1} = \frac{1}{N} \sum_{i=1}^{N} (w_0 + w_1 x_i - y_i) x_i$$

Steepest descent implementation

Input: given a starting point $\mathbf{w} = (-10, -10),\ t = 0.001, \tau = 1000$

Output: model $\mathbf{w}$

for $k = 1 : \tau$ **do**

$$w_0^{(k+1)} := w_0^{(k)} - t\frac{1}{N}\sum_{i=1}^{N}(w_0^{(k)} + w_1^{(k)}x_i - y_i)1;$$

$$w_1^{(k+1)} := w_1^{(k)} - t\frac{1}{N}\sum_{i=1}^{N}(w_0^{(k)} + w_1^{(k)}x_i - y_i)x_i;$$

end

Steepest decent implementation

In matrix notation

Hypothesis:

$$f(x; w) = \tilde{\mathbf{x}}^T \mathbf{w}$$

Cost function :

$$J(\mathbf{w}) = \frac{1}{2N}(\tilde{\mathbf{X}}^T \mathbf{w} - \mathbf{y})^T (\tilde{\mathbf{X}}^T \mathbf{w} - \mathbf{y})$$

Gradient :

$$\nabla J(\mathbf{w}) = \frac{1}{N}\tilde{\mathbf{X}}(\tilde{\mathbf{X}}^T \mathbf{w} - \mathbf{y})$$

What else?

You can see another example with these concepts in:

TO READ:

A Gentle Introduction to Supervised Machine Learning

- Read Sections 2, 3, and 4
- The remaining of the sections are optional

Application of the learning model

Application of the learning model

Training

We are given a dataset $\mathcal{D}$ which we use to learn our model parameters, e.g. the parameters of the linear regressor.

Exploitation

Apply the learned model to new data and **hope** it predicts correctly.

A 1st evaluation of the learning model

But … we want to know aproximately how well it will perform during the exploitation step! What to do?

The simplest evaluation strategy: Simulate exploitation data.

We are given a dataset $\mathcal{D}$ and it is divided in two sets

$$\mathcal{D} = \mathcal{D}_{train} \cup \mathcal{D}_{test}$$

Training

Use $\mathcal{D}_{train}$ to learn the model.

Evaluation

Use $\mathcal{D}_{test}$ to compute the performance of the method.

Exploitation

Apply the model to new data and **hope** it predicts correctly.

A 1st evaluation of the learning model

Suppose that we have two different models. We want to select the one that has a better performance during the exploitation step. But … we don't have exploitation data to evaluate on? What to do?

Simulate exploitation data.

We are given a dataset $\mathcal{D}$ and it is divided in two sets

$$\mathcal{D} = \mathcal{D}_{train} \cup \mathcal{D}_{validation}$$

Training

Use $\mathcal{D}_{train}$ to learn both models.

Evaluation

Use $\mathcal{D}_{validation}$ to decide which of the two models better adapts to our problem.

Apply the selected model to new data and **hope** it predicts correctly.

A 1st evaluation of the learning model

But … I want to know the expected performance of the best method!!! What to do?

Simulate exploitation data.

We are given a dataset $\mathcal{D}$ and it is divided in three sets

$$\mathcal{D} = \mathcal{D}_{train} \cup \mathcal{D}_{validation} \cup \mathcal{D}_{test}$$

Training and model selection

Use $\mathcal{D}_{train}$ to learn both models. Use $\mathcal{D}_{validation}$ to decide which of the two models better adapts to our problem.

Evaluation

Use $\mathcal{D}_{test}$ to compute the performance of the selected method.

Apply the selected model to new data and **hope** it predicts correctly.

A whirlwind tour on machine learning terms

Basic concepts of Machine Learning

- **How do I learn a simple Gaussian?** Probability, random variables, distributions; estimation, convergence and asymptotics, confidence interval
- **How do I learn a mixture of Gaussians (MoG)?** Likelihood, Expectation-Maximization (EM); generalization, model selection, cross-validation; k-means, Hidden Markov odels (HMM)
- **How do I learn any density?** Parametric vs. non-Parametric estimation, Sobolev and other functional spaces; l-2 error; Kernel density estimation (KDE), optimal kernel, KDE theory

Basic concepts of Machine Learning

- **How do I predict a continuous variable (regression)?** Linear regression, regularization, ridge regression, and LASSO; local linear regression; conditional density estimation.
- **How do I predict a discrete variable (classification)?** Bayes classifier, naive Bayes, generative vs. discriminative; perceptron, weight decay, linear support vector machine; nearest neighbor classifier and theory

Basic concepts of Machine Learning

General theory and model frameworks of Machine Learning

- **Which loss function should I use?** Maximum likelihood estimation theory; l-2 estimation; Bayessian estimation; minimax and decision theory, Bayesianism vs frequentism
- **Which model should I use?** AIC and BIC; Vapnik-Chervonenskis theory; cross-validation theory; bootstrapping; Probably Approximately Correct (PAC) theory; Hoeffding-derived bounds.

Basic concepts of Machine Learning

General theory and model frameworks of Machine Learning

- **How can I learn fancier (combined) models?** Ensemble learning theory; boosting; bagging; stacking.
- **How can I learn fancier (nonlinear) models?** Generalized linear models, logistic regression; Kolmogorov theorem, generalized additive models; kernelization, reproducing kernel Hilbert spaces, non-linear SVM, Gaussian process regression
- **How can I learn fancier (compositional) models?** Recursive models, decision trees, hierarchical clustering; neural networks, back propagation, deep belief networks; graphical models, mixtures of HMMs, conditional random fields, max-margin Markov networks; log-linear models; grammars

Basic concepts of Machine Learning

Further common machine learning problems and solutions

- **How do I reduce or relate features?** Feature selection vs dimensionality reduction, wrapper methods for feature selection; causality vs correlation, partial correlation, Bayes net structure learning
- **How do I create new features?** principal component analysis (PCA), independent component analysis (ICA), multidimensional scaling (MDS), manifold learning, supervised dimensionality reduction, metric learning
- **How do I reduce or relate the data?** Clustering, bi-clustering, constrained clustering; association rules and market basket analysis; ranking/ordinal regression; link analysis; relational data

Basic concepts of Machine Learning

Further common machine learning problems and solutions

- **How do I treat time series?** ARMA; Kalman filter and stat-space models, particle filter; functional data analysis; change-point detection; cross-validation for time series
- **How do I treat non-ideal data?** covariate shift; class imbalance; missing data, irregularly sampled data, measurement errors; anomaly detection, robustness

Basic concepts of Machine Learning

General computational frameworks for machine learning

- **How do I optimize the parameters?** Unconstrained vs constrained/Convex optimization, derivative-free methods, first- and second-order methods, backfitting; natural gradient; bound optimization and EM
- **How do I optimize linear functions?** computational linear algebra, matrix inversion for regression, singular value decomposition (SVD) for dimensionality reduction
- **How do I optimize with constraints?** Convexity, Lagrange multipliers, Karush-Kuhn-Tucker conditions, interior point methods, SMO algorithm for SVM

Basic concepts of Machine Learning

General computational frameworks for machine learning

- **How do I evaluate deeply-nested sums?** Exact graphical model inference, variational bounds on sums, approximate graphical model inference, expectation propagation
- **How do I evaluate large sums and searches?** Generalized N-body problems (GNP), hierarchical data structures, nearest neighbor search, fast multiple method; Monte Carlo integration, Markov Chain Monte Carlo, Monte Carlo SVD
- **How do I treat even larger problems?** Parallel/distributed EM, parallel/distributed GNP; stochastic subgradient methods, online learning

Real-world application of machine learning

- **How do I apply all this in the real world?**
  Overview of the parts of the ML, choosing between the methods to use for each task, prior knowledge and assumptions; exploratory data analysis and information visualization; evaluation and interpretation, using confidence intervals and hypothesis test, ROC curves; where the research problems in ML are

UNIVERSITAT DE BARCELONA

Week 6

Course. Introduction to Machine Learning

Theory 6. A gentle introduction to Supervised Learning

Dr. Maria Salamó Llorente
maria.salamo@ub.edu
Dept. Mathematics and Informatics,
Faculty of Mathematics and Informatics,
University of Barcelona (UB)