Statistical Machine Learning

Pier Francesco Palamara
Department of Statistics
University of Oxford

Slide credits and other course material can be found at:
http://www.stats.ox.ac.uk/~palamara/SML_BDI.html

Supervised Learning

Supervised Learning

Unsupervised learning:

- Visualize, summarize and compress data.
- To "extract structure" and postulate hypotheses about data generating process from "unlabelled" observations $x_1, \ldots, x_N$.

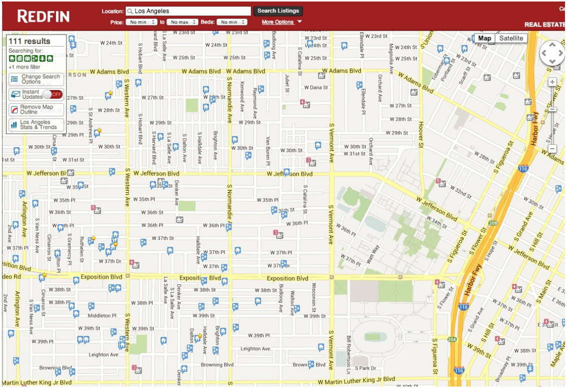
Supervised learning:

- In addition to the observations of X, we have access to their response variables / labels $Y \in \mathcal{Y}$: we observe $\{(x_i, y_i)\}_{i=1}^N$.
- Types of supervised learning:
  - Regression: a numerical value is observed and $\mathcal{Y} = \mathbb{R}$.
  - Classification: discrete responses, e.g. $\mathcal{Y} = \{+1, -1\}$ or $\{1, \ldots, K\}$.

The goal is to accurately predict the response Y on new observations of X, i.e., to **learn a function** $f : \mathbb{R}^p \to \mathcal{Y}$, such that $f(X)$ will be close to the true response Y.

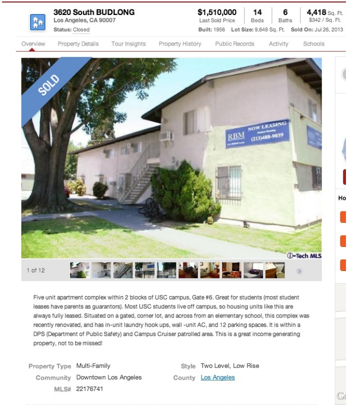
Regression Example: House Price

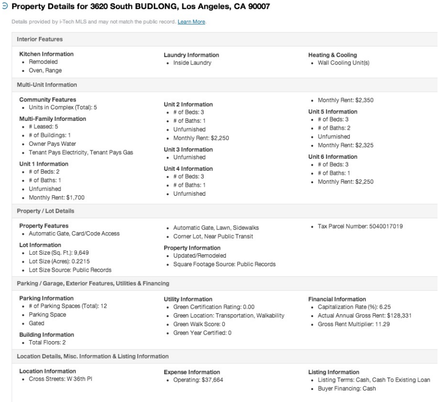
Retrieve historical sales records



Features used to predict

We will use properties of the house, e.g. squared meters, distance from train station, etc.





Goal: predict price of another house given these properties.

Classification Example: Lymphoma

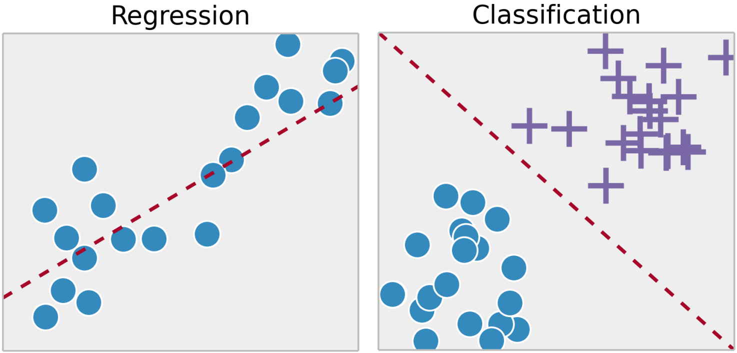
We have gene expression measurements X of $N = 62$ patients for $p = 4026$ genes. For each patient, $Y \in \{0, 1\}$ denotes one of two subtypes of cancer.

```
> str(X)
'data.frame':   62 obs. of  4026 variables:
 $ Gene 1   : num  -0.344 -1.188  0.520 -0.748 -0.868 ...
 $ Gene 2   : num  -0.953 -1.286  0.657 -1.328 -1.330 ...
 $ Gene 3   : num  -0.776 -0.588  0.409 -0.991 -1.517 ...
 $ Gene 4   : num  -0.474 -1.588  0.219  0.978 -1.604 ...
 $ Gene 5   : num  -1.896 -1.960 -1.695 -0.348 -0.595 ...
 $ Gene 6   : num  -2.075 -2.117  0.121 -0.800  0.651 ...
 $ Gene 7   : num  -1.875 -1.818  0.317  0.387  0.041 ...
 $ Gene 8   : num  -1.539 -2.433 -0.337 -0.522 -0.668 ...
 $ Gene 9   : num  -0.604 -0.710 -1.269 -0.832  0.458 ...
 $ Gene 10  : num  -0.218 -0.487 -1.203 -0.919 -0.848 ...
 $ Gene 11  : num  -0.340  1.164  1.023  1.133 -0.541 ...
 $ Gene 12  : num  -0.531  0.488 -0.335  0.496 -0.358 ...

> str(Y)
 num [1:62] 0 0 0 1 0 0 1 0 0 0 ...
```

Goal: predict cancer subtype given gene expressions of a new patient.

Regression VS Classification



Loss function

- Suppose we made a prediction $\hat{Y} = f(X) \in \mathcal{Y}$ after observing X.
- How good is the prediction? We can use a **loss function** $L : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}^+$ to formalize the quality of the prediction.
- Typical loss functions:
  - **Squared loss** for regression

$$L(Y, f(X)) = (f(X) - Y)^2 .$$

  - **Absolute loss** for regression

$$L(Y, f(X)) = |f(X) - Y| .$$

  - **Misclassification loss** (or **0-1 loss**) for classification

$$L(Y, f(X)) = \begin{cases} 0 & f(X) = Y \\ 1 & f(X) \neq Y \end{cases} .$$

  Many other choices are possible, e.g., **weighted misclassification loss**.
- In classification, if estimated probabilities $\hat{p}(k)$ for each class $k \in \mathcal{Y}$ are returned, **log-likelihood loss** (or **log loss**) $L(Y, \hat{p}) = -\log \hat{p}(Y)$ is often used.

Risk

- paired observations $\{(x_i, y_i)\}_{i=1}^N$ viewed as i.i.d. realizations of a random variable (X, Y) on $\mathcal{X} \times \mathcal{Y}$ with joint distribution P_{XY}

Risk

For a given loss function L, the **risk** R of a learned function f is given by the expected loss

$$R(f) = \mathbb{E}_{P_{XY}}\left[L(Y, f(X))\right],$$

where the expectation is with respect to the true (unknown) joint distribution of (X, Y).

- The risk is unknown, but we can compute the **empirical risk**:

$$R_N(f) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)).$$

Hypothesis space and Empirical Risk Minimization

- Hypothesis space $\mathcal{H}$ is the space of functions f under consideration.
- **Inductive bias**: necessary assumptions on “plausible” hypotheses
- Find best function in the space of hypothesis $\mathcal{H}$ minimizing the risk:

$$f_\star = \operatorname*{argmin}_{f \in \mathcal{H}} \mathbb{E}_{X,Y}[L(Y, f(X))]$$

- **Empirical Risk Minimization** (ERM): minimize the empirical risk instead, since we typically do not know $P_{X,Y}$.

$$\hat{f} = \operatorname*{argmin}_{f \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i))$$

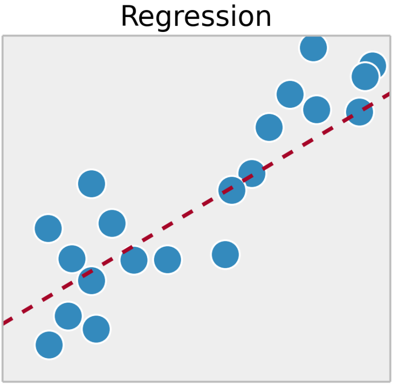
- How complex should we allow functions f to be? If hypothesis space $\mathcal{H}$ is “too large”, ERM will overfit. Function

$$\hat{f}(x) = \begin{cases} y_i & \text{if } x = x_i, \\ 0 & \text{otherwise} \end{cases}$$

will have zero empirical risk, but is useless for generalization, since it has simply “memorized” the dataset.

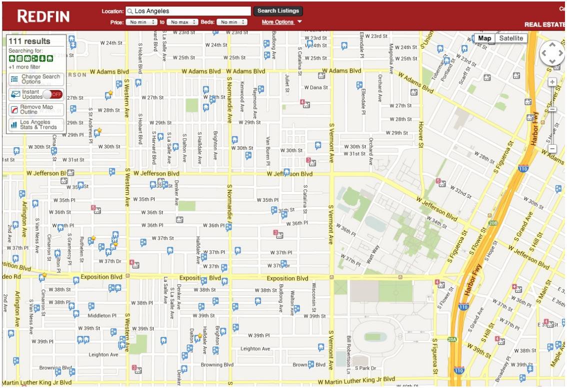
Linear Regression

We will use the framework of linear regression, which should be familiar to you, to illustrate some of the key concepts of supervised learning.



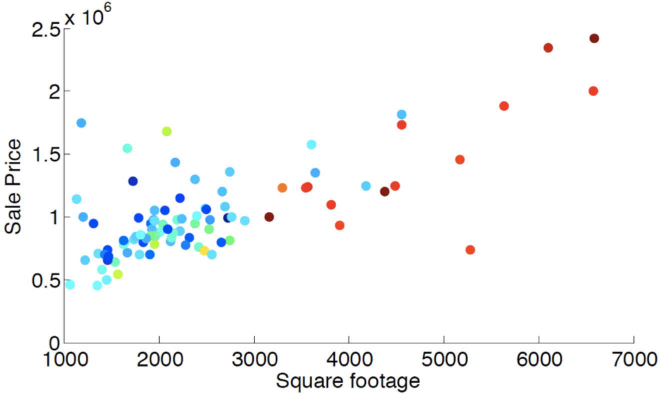
Linear regression: predicting the sale price of a house

We will use the house price example.
(This will be our training data)



Correlation between square footage and sale price

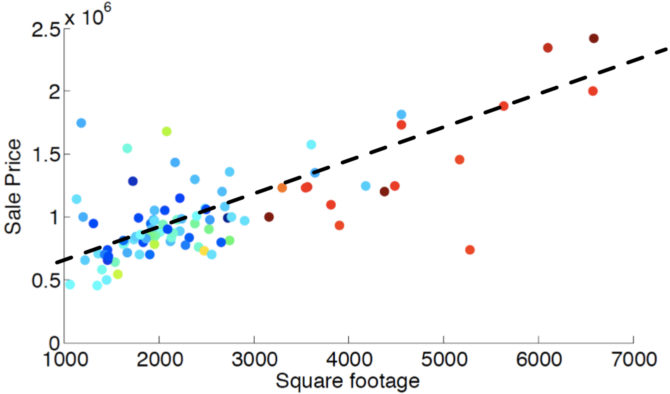
The size of a house is a good predictor of its price.



Note: colors are not important here

Roughly linear relationship

The size of a house is a good predictor of its price.



Sale price $\approx$ price_per_sqft $\times$ square_footage $+$ fixed_expense

Linear regression (ordinary least squares)

Setup

- Input: $\boldsymbol{x} \in \mathbb{R}^{\mathsf{D}}$ (covariates, predictors, features, etc)
- Output: $y \in \mathbb{R}$ (responses, targets, outcomes, outputs, etc)
- **Hypotheses**: $h_{\boldsymbol{\theta},\theta_0} : \boldsymbol{x} \rightarrow y$, with $h_{\boldsymbol{\theta},\theta_0}(\boldsymbol{x}) = \theta_0 + \sum_d \theta_d x_d = \theta_0 + \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}$

  $\boldsymbol{\theta} = [\theta_1 \ \theta_2 \ \cdots \ \theta_{\mathsf{D}}]^{\mathrm{T}}$: **weights**, **parameters**. θ_0 is the intercept (also called bias).
- Training data: $\mathcal{D} = \{(\boldsymbol{x}_n, y_n), n = 1, 2, \ldots, \mathsf{N}\}$
- We will use the **squared loss** (differentiable):

$$\text{(sale price - prediction)}^2 = (y_n - h_{\boldsymbol{\theta}}(\boldsymbol{x}_n))^2$$

  - Could use other loss functions, e.g. **absolute loss**:

$$|\text{sale price - prediction}| = |y_n - h_{\boldsymbol{\theta}}(\boldsymbol{x}_n)|$$

How do we learn parameters?

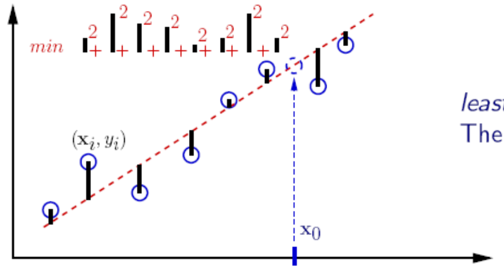
Minimize prediction error on training data

- Hypothesis:

$$y = h_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 x$$

- We chose to minimize the squared loss. Empirical risk:

$$R_N(\boldsymbol{\theta}) = \frac{1}{N} \sum_{n=1}^{N} \left(y_n - h_{\boldsymbol{\theta}}(\boldsymbol{x}_n)\right)^2$$

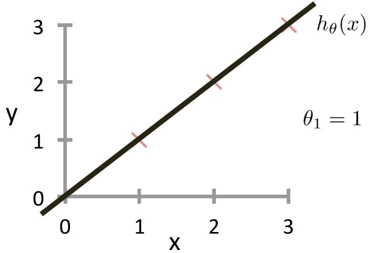


least squares (LSQ)
The fitted line is used as a predictor

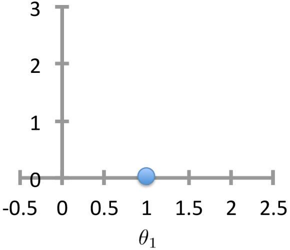
Intuiton behind the squared loss

Assume $x \in \mathbb{R}$

$h_\theta(x)$

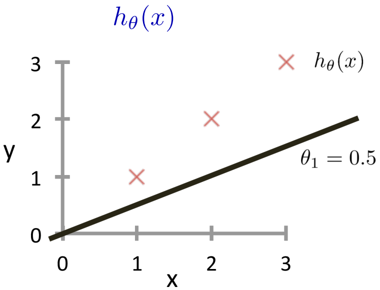


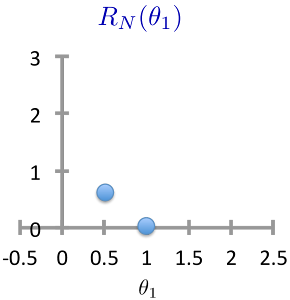
$R_N(\theta_1)$



Intuiton behind the squared loss

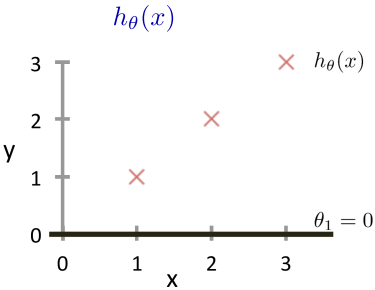
Assume $x \in \mathbb{R}$

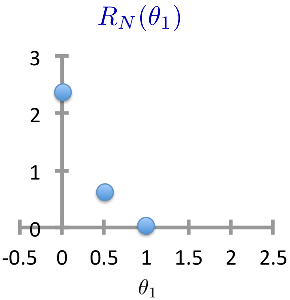




Intuiton behind the squared loss

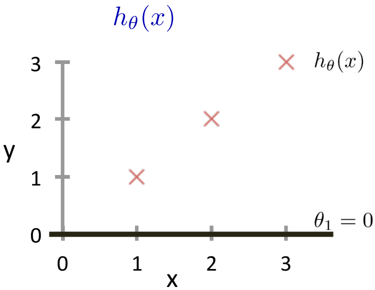
Assume $x \in \mathbb{R}$

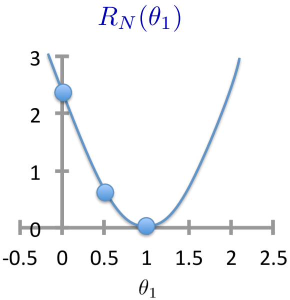




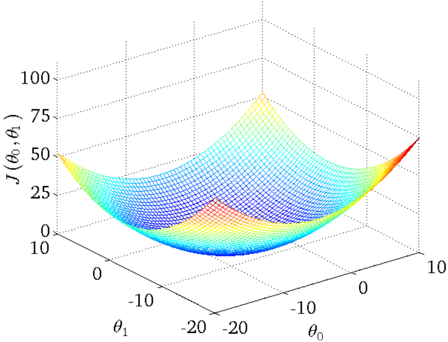
Intuiton behind the squared loss

Assume $x \in \mathbb{R}$





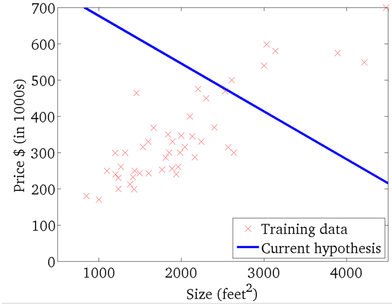
Intuiton behind the squared loss

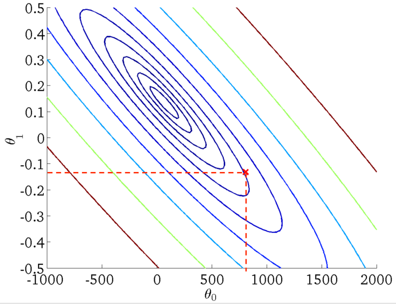


Intuiton behind the squared loss

$h_\theta(x)$

$R_N(\theta_0, \theta_1)$

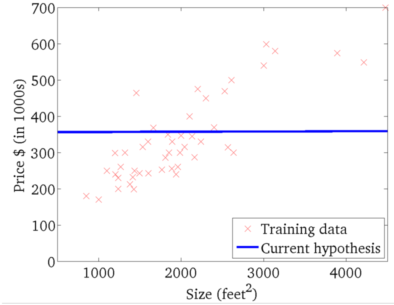


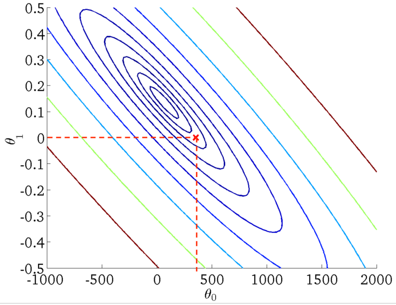


Intuiton behind the squared loss

$h_\theta(x)$

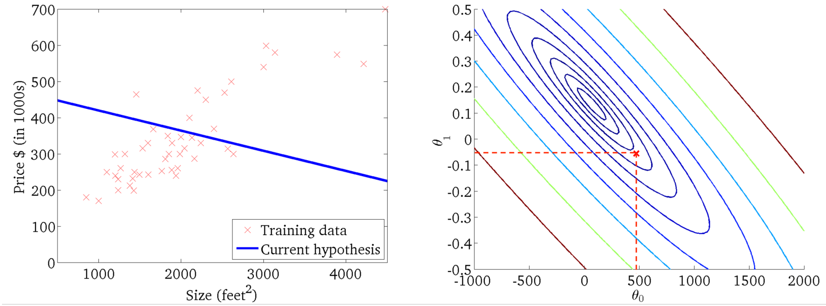
$R_N(\theta_0, \theta_1)$





Intuiton behind the squared loss

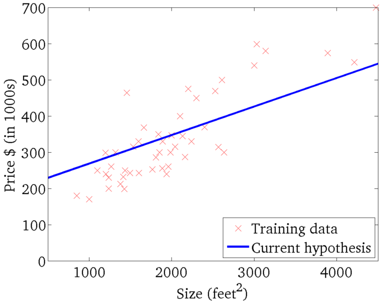
$h_\theta(x)$ $R_N(\theta_0, \theta_1)$

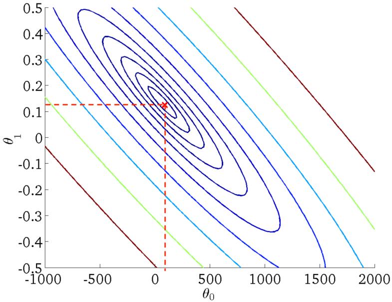


Intuiton behind the squared loss

$h_\theta(x)$

$R_N(\theta_0, \theta_1)$





A simple case: $\boldsymbol{x}$ is just one-dimensional (D=1)

Squared loss

(dropping the $1/N$ for simplicity)

$$R_N(\boldsymbol{\theta}) = \sum_n [y_n - h_{\boldsymbol{\theta}}(\boldsymbol{x}_n)]^2 = \sum_n [y_n - (\theta_0 + \theta_1 x_n)]^2$$

Analytical solution

For linear regression, the minimization can be done in closed form.

Identify stationary points by taking derivative with respect to parameters and setting to zero

$$\frac{\partial R_N(\boldsymbol{\theta})}{\partial \theta_0} = 0 \Rightarrow -2 \sum_n [y_n - (\theta_0 + \theta_1 x_n)] = 0$$

$$\frac{\partial R_N(\boldsymbol{\theta})}{\partial \theta_1} = 0 \Rightarrow -2 \sum_n [y_n - (\theta_0 + \theta_1 x_n)] x_n = 0$$

$$\frac{\partial R_N(\boldsymbol{\theta})}{\partial \theta_0} = 0 \Rightarrow -2\sum_n [y_n - (\theta_0 + \theta_1 x_n)] = 0$$

$$\frac{\partial R_N(\boldsymbol{\theta})}{\partial \theta_1} = 0 \Rightarrow -2\sum_n [y_n - (\theta_0 + \theta_1 x_n)] x_n = 0$$

Simplify these expressions to get "Normal Equations"

$$\sum y_n = N\theta_0 + \theta_1 \sum x_n$$

$$\sum x_n y_n = \theta_0 \sum x_n + \theta_1 \sum x_n^2$$

We have two equations and two unknowns. Solving we get:

$$\theta_1 = \frac{\sum (x_n - \bar{x})(y_n - \bar{y})}{\sum (x_i - \bar{x})^2} \qquad \text{and} \qquad \theta_0 = \bar{y} - \theta_1 \bar{x}$$

where $\bar{x} = \frac{1}{n}\sum_n x_n$ and $\bar{y} = \frac{1}{n}\sum_n y_n$.

Why is minimizing R_N sensible?

Probabilistic interpretation

- Noisy observation model

$$Y = \theta_0 + \theta_1 X + \eta$$

  where $\eta \sim \mathcal{N}(0, \sigma^2)$ is a Gaussian random variable

- Likelihood of one training sample (x_n, y_n)

$$p(y_n | x_n; \boldsymbol{\theta}) = \mathcal{N}(\theta_0 + \theta_1 x_n, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{[y_n - (\theta_0 + \theta_1 x_n)]^2}{2\sigma^2}}$$

Probabilistic interpretation (cont'd)

Log-likelihood of the training data $\mathcal{D}$ (assuming i.i.d)

$$
\begin{aligned}
\mathcal{LL}(\boldsymbol{\theta}) &= \log P(\mathcal{D}) \\
&= \log \prod_{n=1}^{\mathsf{N}} p(y_n|x_n) = \sum_n \log p(y_n|x_n) \\
&= \sum_n \left\{ -\frac{[y_n - (\theta_0 + \theta_1 x_n)]^2}{2\sigma^2} - \log \sqrt{2\pi}\sigma \right\} \\
&= -\frac{1}{2\sigma^2} \sum_n [y_n - (\theta_0 + \theta_1 x_n)]^2 - \frac{\mathsf{N}}{2} \log \sigma^2 - \mathsf{N} \log \sqrt{2\pi} \\
&= -\frac{1}{2} \left\{ \frac{1}{\sigma^2} \sum_n [y_n - (\theta_0 + \theta_1 x_n)]^2 + \mathsf{N} \log \sigma^2 \right\} + \mathsf{const}
\end{aligned}
$$

What is the relationship between minimizing R_N and maximizing the log-likelihood?

Maximum likelihood estimation

Estimating σ, θ_0 and θ_1 can be done in two steps

- Maximize over θ_0 and θ_1

$$\max \ \log P(\mathcal{D}) \Leftrightarrow \min \sum_n [y_n - (\theta_0 + \theta_1 x_n)]^2 \leftarrow \text{That is } R_N(\boldsymbol{\theta})!$$

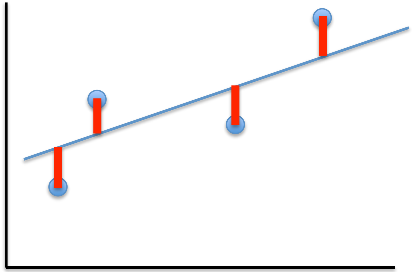
- Maximize over $s = \sigma^2$ (we could estimate σ directly)

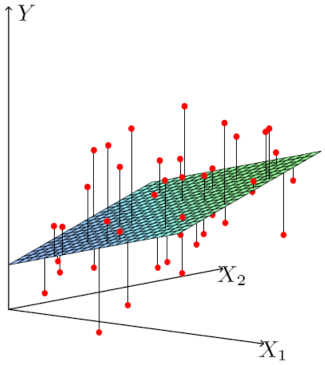
$$\log P(\mathcal{D}) = -\frac{1}{2}\left\{\frac{1}{\sigma^2}\sum_n [y_n - (\theta_0 + \theta_1 x_n)]^2 + \mathsf{N} \log \sigma^2\right\} + \text{const}$$

$$\frac{\partial \log P(\mathcal{D})}{\partial s} = -\frac{1}{2}\left\{-\frac{1}{s^2}\sum_n [y_n - (\theta_0 + \theta_1 x_n)]^2 + \mathsf{N}\frac{1}{s}\right\} = 0$$

$$\rightarrow \sigma^{*2} = s^* = \frac{1}{\mathsf{N}}\sum_n [y_n - (\theta_0 + \theta_1 x_n)]^2$$

Linear regression when $\boldsymbol{x}$ is D-dimensional





Linear regression when $\boldsymbol{x}$ is D-dimensional

$R_N(\boldsymbol{\theta})$ in matrix form

$$R_N(\boldsymbol{\theta}) = \sum_n [y_n - (\theta_0 + \sum_d \theta_d x_{nd})]^2 = \sum_n [y_n - \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{x}_n]^2$$

where we have redefined some variables (by augmenting)

$$\boldsymbol{x} \leftarrow [1\ x_1\ x_2\ \ldots\ x_{\mathsf{D}}]^{\mathrm{T}}, \quad \boldsymbol{\theta} \leftarrow [\theta_0\ \theta_1\ \theta_2\ \ldots\ \theta_{\mathsf{D}}]^{\mathrm{T}}$$

which leads to

$$\begin{aligned}
R_N(\boldsymbol{\theta}) &= \sum_n (y_n - \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{x}_n)(y_n - \boldsymbol{x}_n^{\mathrm{T}} \boldsymbol{\theta}) \\
&= \sum_n \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{x}_n \boldsymbol{x}_n^{\mathrm{T}} \boldsymbol{\theta} - 2 y_n \boldsymbol{x}_n^{\mathrm{T}} \boldsymbol{\theta} + \text{const.} \\
&= \left\{ \boldsymbol{\theta}^{\mathrm{T}} \left(\sum_n \boldsymbol{x}_n \boldsymbol{x}_n^{\mathrm{T}} \right) \boldsymbol{\theta} - 2 \left(\sum_n y_n \boldsymbol{x}_n^{\mathrm{T}} \right) \boldsymbol{\theta} \right\} + \text{const.}
\end{aligned}$$

$R_N(\boldsymbol{\theta})$ in new notations

Design matrix and target vector

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{x}_1^{\mathrm{T}} \\ \boldsymbol{x}_2^{\mathrm{T}} \\ \vdots \\ \boldsymbol{x}_{\mathsf{N}}^{\mathrm{T}} \end{pmatrix} \in \mathbb{R}^{\mathsf{N}\times(D+1)}, \quad \boldsymbol{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{\mathsf{N}} \end{pmatrix}$$

Compact expression

$$R_N(\boldsymbol{\theta}) = ||\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y}||_2^2 = \left\{\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{\theta} - 2\left(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}\right)^{\mathrm{T}}\boldsymbol{\theta}\right\} + \text{const}$$

Solution in matrix form

Compact expression

$$R_N(\boldsymbol{\theta}) = ||\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y}||_2^2 = \left\{\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{\theta} - 2\left(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}\right)^{\mathrm{T}}\boldsymbol{\theta}\right\} + \text{const}$$

Gradients of Linear and Quadratic Functions

- $\nabla_{\boldsymbol{x}}\boldsymbol{b}^\top\boldsymbol{x} = \boldsymbol{b}$
- $\nabla_{\boldsymbol{x}}\boldsymbol{x}^\top\boldsymbol{A}\boldsymbol{x} = 2\boldsymbol{A}\boldsymbol{x}$ (symmetric $\boldsymbol{A}$)

Normal equation

$$\nabla_{\boldsymbol{\theta}}R_N(\boldsymbol{\theta}) \propto \boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{X}^{\mathrm{T}}\boldsymbol{y} = 0$$

This leads to the linear regression solution[1]

$$\boldsymbol{\theta} = \left(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}$$

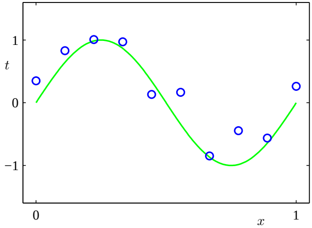
[1] Also see PRML book, Section 3.1.2 for a geometric interpretation.

Mini-Summary

- Linear regression is the linear combination of features $f : \boldsymbol{x} \to y$, with $f(\boldsymbol{x}) = \theta_0 + \sum_d \theta_d x_d = \theta_0 + \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{x}$
- If we minimize residual sum of squares as our learning objective, we get a closed-form solution of parameters
- Probabilistic interpretation: maximum likelihood if assuming residual is Gaussian distributed
- D-dimensional case leads to compact expressions in matrix form.

Nonlinear basis functions

Can we learn non-linear functions?



We can use a nonlinear mapping

$$\boldsymbol{\phi}(\boldsymbol{x}) : \boldsymbol{x} \in \mathbb{R}^D \rightarrow \boldsymbol{z} \in \mathbb{R}^M$$

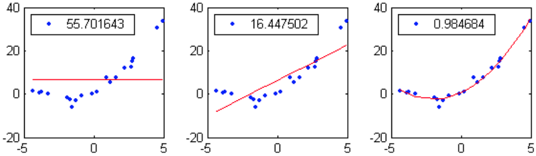
where M is the dimensionality of the new feature/input $\boldsymbol{z}$ (or $\boldsymbol{\phi}(\boldsymbol{x})$). Note that M could be either greater than D or less than or the same.

Nonlinear basis functions

Can we learn non-linear functions?
We can use a nonlinear mapping

$$\boldsymbol{\phi}(\boldsymbol{x}) : \boldsymbol{x} \in \mathbb{R}^D \rightarrow \boldsymbol{z} \in \mathbb{R}^M$$

For instance, we could use polynomials of increasing order, $\phi_k(\boldsymbol{x}_i) = \boldsymbol{x}_i^k$



P = 0 P = 1 P = 2

With the new features, we can apply our learning techniques to minimize our errors on the transformed training data

- for linear methods, prediction is still based on $\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x})$

Regression with nonlinear basis functions

Residual sum squares

$$\sum_n [\boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\phi}(\boldsymbol{x}_n) - y_n]^2$$

where $\boldsymbol{\theta} \in \mathbb{R}^M$, the same dimensionality as the transformed features $\phi(x)$.

The linear regression solution can be formulated with the new design matrix

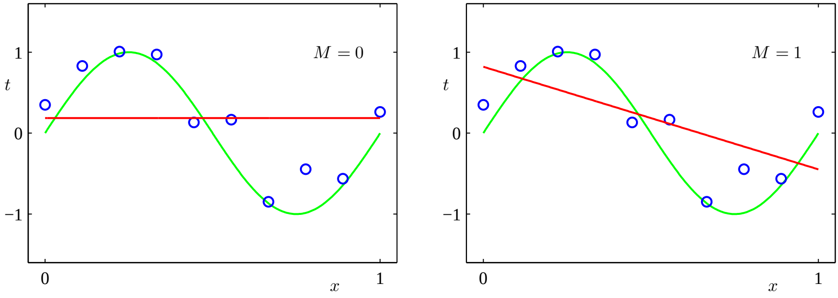
$$\boldsymbol{\Phi} = \begin{pmatrix} \boldsymbol{\phi}(\boldsymbol{x}_1)^{\mathrm{T}} \\ \boldsymbol{\phi}(\boldsymbol{x}_2)^{\mathrm{T}} \\ \vdots \\ \boldsymbol{\phi}(\boldsymbol{x}_N)^{\mathrm{T}} \end{pmatrix} \in \mathbb{R}^{N \times M}, \quad \boldsymbol{\theta}^{\mathsf{LMS}} = \left(\boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi} \right)^{-1} \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{y}$$

Regression with nonlinear basis functions

Polynomial basis functions

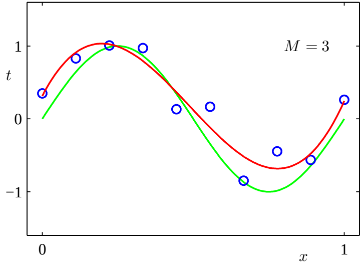
$$\phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^M \end{bmatrix} \Rightarrow f(x) = \theta_0 + \sum_{m=1}^{M} \theta_m x^m$$

Fitting samples from a sine function: **underfitting** as $f(x)$ is too simple

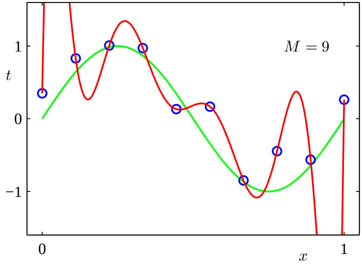


Adding high-order terms

M=3



M=9: **overfitting**



More complex features lead to better results on the training data, but potentially worse results on new data, e.g., test data!

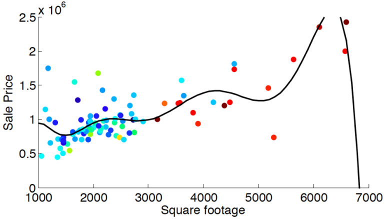
Overfitting

Parameters for higher-order polynomials are very large

	$M = 0$	$M = 1$	$M = 3$	$M = 9$
θ_0	0.19	0.82	0.31	0.35
θ_1		-1.27	7.99	232.37
θ_2			-25.43	-5321.83
θ_3			17.37	48568.31
θ_4				-231639.30
θ_5				640042.26
θ_6				-1061800.52
θ_7				1042400.18
θ_8				-557682.99
θ_9				125201.43

Overfitting can be quite disastrous

Fitting the housing price data with $M = 7$



Note that the price would go to zero (or negative) if you buy bigger ones!
This is called poor generalization/overfitting.