CS 2750 Machine Learning
Lecture 9

Linear regression II

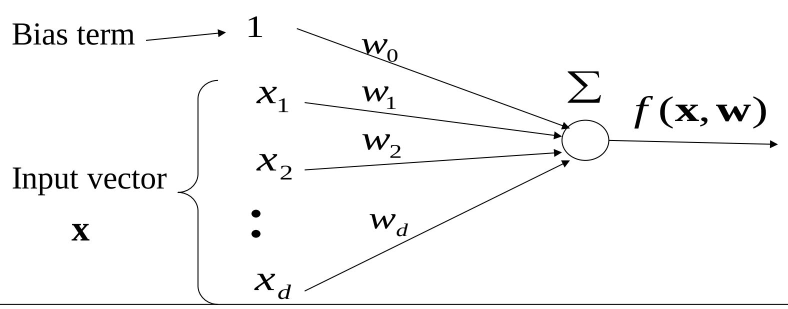
Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Linear regression

- **Function** $f : X \rightarrow Y$
- Y is a linear combination of input components

$$f(\mathbf{x}) = w_0 + w_1 x_1 + w_2 x_2 + \ldots w_d x_d = w_0 + \sum_{j=1}^{d} w_j x_j$$

$w_0, w_1, \ldots w_k$ - **parameters (weights)**



Linear regression. Error.

- **Data:** $D_i = < \mathbf{x}_i, y_i >$
- **Function:** $\mathbf{x}_i \rightarrow f(\mathbf{x}_i)$
- We would like to have $y_i \approx f(\mathbf{x}_i)$ for all $i = 1,..,n$

- **Error function**
  - measures how much our predictions deviate from the desired answers

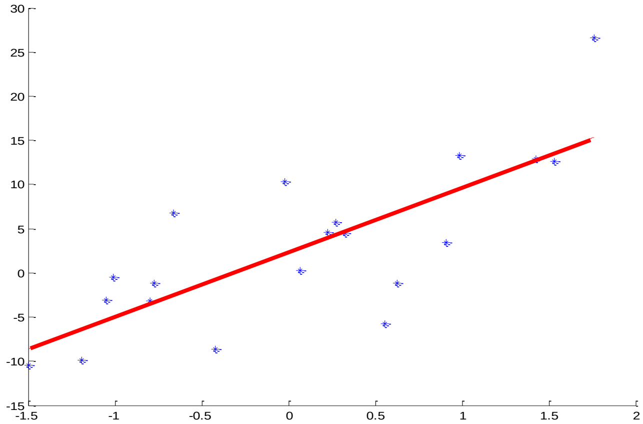
Mean-squared error $J_n = \frac{1}{n} \sum_{i=1,..n} (y_i - f(\mathbf{x}_i))^2$

- **Learning:**

  We want to find the weights minimizing the error !

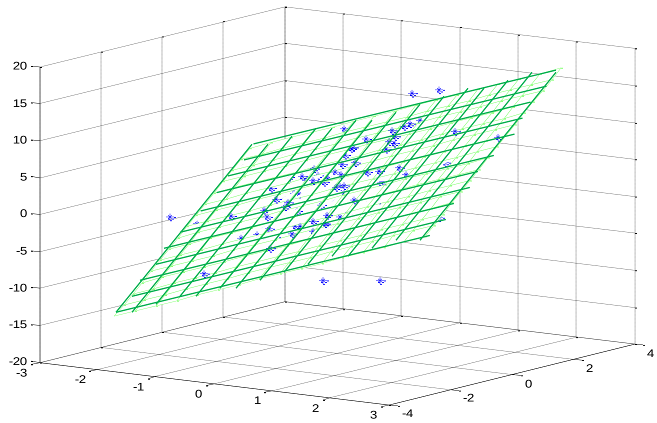
Linear regression. Example

- 1 dimensional input $\mathbf{x} = (x_1)$



Linear regression. Example.

- 2 dimensional input $\mathbf{x} = (x_1, x_2)$



Solving linear regression

- The optimal set of weights satisfies:

$$\nabla_{\mathbf{w}}(J_n(\mathbf{w})) = -\frac{2}{n}\sum_{i=1}^{n}(y_i - \mathbf{w}^T\mathbf{x}_i)\mathbf{x}_i = \overline{\mathbf{0}}$$

Leads to a **system of linear equations (SLE)** with *d+1* unknowns of the form

$$\mathbf{A}\mathbf{w} = \mathbf{b}$$

$$w_0 \sum_{i=1}^{n} x_{i,0}x_{i,j} + w_1 \sum_{i=1}^{n} x_{i,1}x_{i,j} + \ldots + w_j \sum_{i=1}^{n} x_{i,j}x_{i,j} + \ldots + w_d \sum_{i=1}^{n} x_{i,d}x_{i,j} = \sum_{i=1}^{n} y_i x_{i,j}$$

Solution to SLE: $\boxed{\mathbf{w} = \mathbf{A}^{-1}\mathbf{b}}$

Assuming **X** is an *nxd* data matrix with rows corresponding to examples and columns to inputs, and ***y*** is nx1 vector of outputs, then $\mathbf{w} = (\mathbf{X}^{\mathbf{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathbf{T}}\mathbf{y}$

Gradient descent solution

Goal: the weight optimization in the linear regression model

$$J_n = Error(\mathbf{w}) = \frac{1}{n} \sum_{i=1,..n} (y_i - f(\mathbf{x}_i, \mathbf{w}))^2$$

An alternative to SLE solution:

- **Gradient descent**

  Idea:

  – Adjust weights in the direction that improves the Error

  – The gradient tells us what is the right direction

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \nabla_{\mathbf{w}} Error_i(\mathbf{w})$$

$\alpha > 0$ - a **learning rate** (scales the gradient changes)

Batch vs Online regression algorithm

- The error function defined on the complete dataset *D*

$$J_n = Error(\mathbf{w}) = \frac{1}{n} \sum_{i=1,..n} (y_i - f(\mathbf{x}_i, \mathbf{w}))^2$$

- We say we are learning the model in **the batch mode**:
  – All examples are available at the time of learning
  – Weights are optimizes with respect to all training examples

- An alternative is to learn the model in **the online mode**
  – Examples are arriving sequentially
  – Model weights are updated after every example
  – If needed examples seen can be forgotten

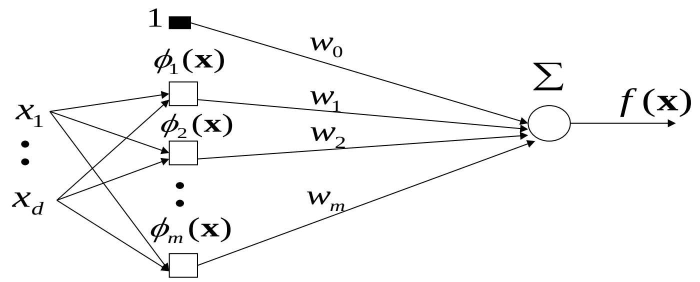
-

Extensions of simple linear model

Replace inputs to linear units with m **feature (basis) functions** to model **nonlinearities**

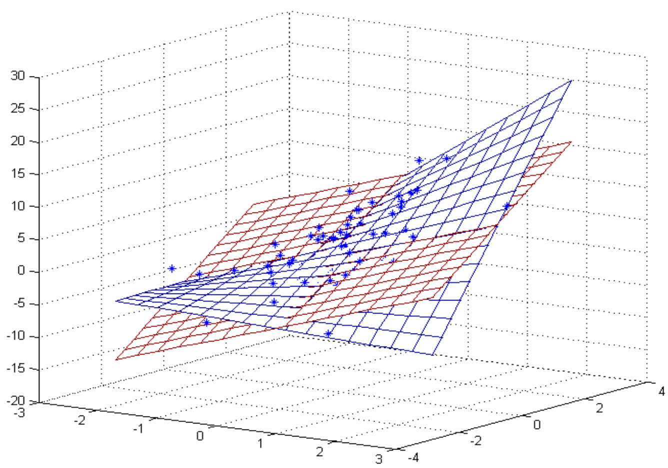
$$f(\mathbf{x}) = w_0 + \sum_{j=1}^{m} w_j \phi_j(\mathbf{x})$$

$\phi_j(\mathbf{x})$ - an arbitrary function of $\mathbf{x}$



Original input → **New input** → **Linear model**

Non-linear model

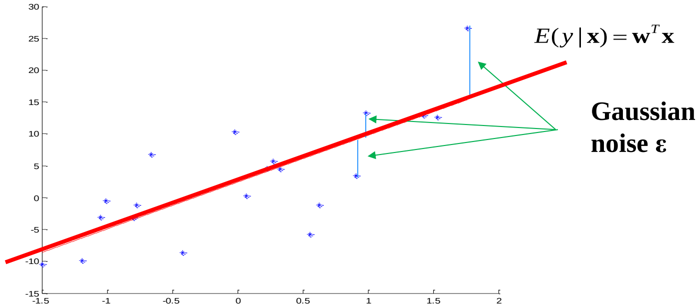


Statistical model of regression

A statistical model of linear regression:

$$y = \mathbf{w}^T \mathbf{x} + \varepsilon \qquad\qquad \varepsilon \sim \mathrm{N}(0, \sigma^2)$$

ε is a random noise, represents things we cannot capture with $\mathbf{w}^T \mathbf{x}$



$$y \sim N(\mathbf{w}^T \mathbf{x}, \sigma^2)$$

Regularized linear regression

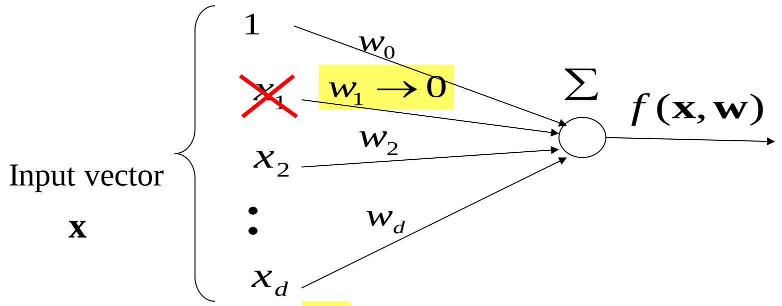
- If the number of parameters is large relative to the number of data points used to train the model, we face the threat of overfitting (generalization error of the model goes up)
- The prediction accuracy can be often improved by setting some coefficients to zero
  - Increases the bias, reduces the variance of estimates
- **Solutions:**
  - **Subset selection**
  - **Ridge regression**
  - **Lasso regression**
  - **Principal component regression**

Regularization: motivation

- If the model is too complex and can cause overfitting, its prediction accuracy can be improved by **removing some inputs from the model = setting their coefficients to zero**

$$f(\mathbf{x}) = w_0 x_0 + w_1 x_1 + w_2 x_2 + w_3 x_3 + \ldots w_d x_d = \mathbf{w}^T \mathbf{x}$$

$w_0, w_1, \ldots w_k$ - **parameters (weights)**



$$f(\mathbf{x}) = w_0 x_0 + 0 x_1 + w_2 x_2 + w_3 x_3 + \ldots w_d x_d = \mathbf{w}^T \mathbf{x}$$

Ridge regression

Question: how to force the weights to 0 ?

- Error function for the standard least squares estimates:

$$J_n(\mathbf{w}) = \frac{1}{n} \sum_{i=1,..n} (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

- **We seek:** $$\mathbf{w}^* = \arg\min_{\mathbf{w}} \frac{1}{n} \sum_{i=1,..n} (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

- **Ridge regression:**

$$J_n(\mathbf{w}) = \frac{1}{n} \sum_{i=1,..n} (y_i - \mathbf{w}^T \mathbf{x}_i)^2 + \lambda \|\mathbf{w}\|_{L2}^2$$

- Where $\|\mathbf{w}\|_{L2}^2 = \sum_{i=0}^{d} w_i^2 = \mathbf{w}^T \mathbf{w}$ and $\lambda \geq 0$

- What does the new objective function do?

Ridge regression

- **Standard regression:**

$$J_n(\mathbf{w}) = \frac{1}{n}\sum_{i=1,..n}(y_i - \mathbf{w}^T\mathbf{x}_i)^2$$

- **Ridge regression:**

$$J_n(\mathbf{w}) = \frac{1}{n}\sum_{i=1,..n}(y_i - \mathbf{w}^T\mathbf{x}_i)^2 + \lambda\|\mathbf{w}\|_{L2}^2$$

$$\|\mathbf{w}\|_{L2}{}^2 = \sum_{i=0}^{d} w_i^2 = \mathbf{w}^T\mathbf{w}$$

- penalizes non-zero weights with the cost proportional to λ (a **shrinkage coefficient**)
- If an input attribute x_j has a small effect on improving the error function it is "shut down" by the penalty term
- Inclusion of a shrinkage penalty is often referred to as **regularization.**

(ridge regression is related to Tikhonov regularization**)**

Regularized linear regression

How to solve the least squares problem if the error function is enriched by the regularization term $\lambda\|\mathbf{w}\|^2$?

Answer: The solution to the optimal set of weights **w** is obtained again by solving a set of linear equation.

Standard linear regression:

$$\nabla_{\mathbf{w}}(J_n(\mathbf{w})) = -\frac{2}{n}\sum_{i=1}^{n}(y_i - \mathbf{w}^T\mathbf{x}_i)\mathbf{x}_i = \bar{\mathbf{0}}$$

Solution: $\mathbf{w}^* = (\mathbf{X}^\mathbf{T}\mathbf{X})^{-1}\mathbf{X}^\mathbf{T}\mathbf{y}$

where **X** is an *nxd* matrix with rows corresponding to examples and columns to inputs, y is nx1 vector of outputs

Regularized linear regression:

$$\mathbf{w}^* = (\lambda\mathbf{I} + \mathbf{X}^\mathbf{T}\mathbf{X})^{-1}\mathbf{X}^\mathbf{T}\mathbf{y}$$

Regularized linear regression

Ridge regularization is also related to the **Bayesian regression with the Gaussian prior**

Idea:

Statistical model for linear regression:

$$y = \mathbf{w}^T \mathbf{x} + \varepsilon \qquad \varepsilon \sim \mathrm{N}(0, \sigma^2)$$

$$p(y \mid x) = N(\mathbf{w}^T \mathbf{x}, \sigma)$$

Add a prior on w $p(\mathbf{w}) = N(\mathbf{0}, I\gamma)$

Posterior on w: $p(\mathbf{w} \mid \mathbf{X}, \mathbf{Y}, \gamma)$ **Gaussian**

- The objective function for the regularized least squares corresponds to the problem of finding the mode of the posterior . $p(\mathbf{w} \mid \mathbf{X}, \mathbf{Y}, \gamma)$, where $\lambda \approx \gamma$

Lasso regression

- **Standard regression:**

$$J_n(\mathbf{w}) = \frac{1}{n} \sum_{i=1,..n} (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

- **Lasso regression/regularization:**

$$J_n(\mathbf{w}) = \frac{1}{n} \sum_{i=1,..n} (y_i - \mathbf{w}^T \mathbf{x}_i)^2 + \lambda \|\mathbf{w}\|_{L1}$$

- $\|\mathbf{w}\|_{L1} = \sum_{i=0}^{d} | w_i |$ penalizes non-zero weights with the cost proportional to λ.
- L1 is more aggressive pushing the weights to 0 compared to L2
- The objective function corresponds to the mode of the posterior in the Bayesian regression when the prior on **w** is modeled using a Laplace distribution