Machine Learning

Lecture 8
Regression

Felix Bießmann

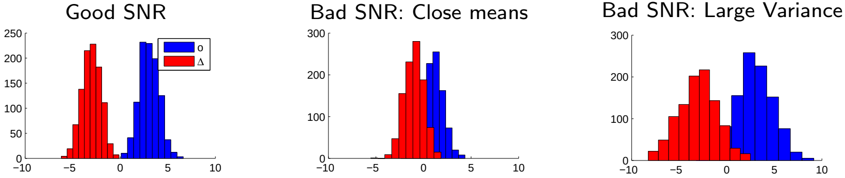
Beuth University & Einstein Center for Digital Future

June 4, 2019

Signal-to-Noise Ratio in Classification Settings

A useful definition of Signal-to-Noise Ratio for classification is

$$\frac{\text{Between Class Variance}}{\text{Within Class Variance}} \tag{1}$$

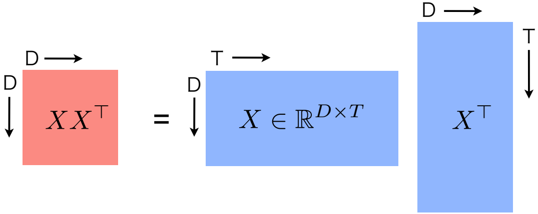


Covariance Matrices

Given T data points $\mathbf{x} \in \mathbb{R}^D$ in a data matrix $\mathbf{X} \in \mathbb{R}^{D \times T}$ the empirical estimate of the **covariance matrix** is defined as

$$1/T \;\; \mathbf{X}\mathbf{X}^\top \tag{2}$$

where we assume centered data, i.e. $\sum_{t=1}^{T} \mathbf{x}_t = 0$.

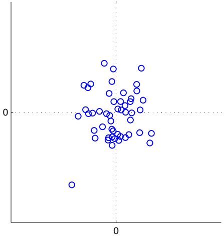


Correlated Data and Linear Mappings

Simulating correlated data can help understanding it

We generate uncorrelated data $\mathbf{x} \in \mathbb{R}^2$ drawn from a normal distribution $\mathbf{x} \sim \mathcal{N}(0, 1)$
We induce *correlations* by a diagonal scaling matrix D and a rotation matrix R

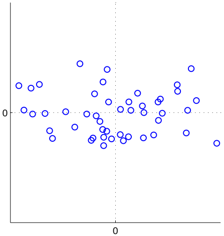
Uncorrelated



$$\mathbf{x} \sim \mathcal{N}(0, 1)$$

$$\mathbf{X}\mathbf{X}^\top = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

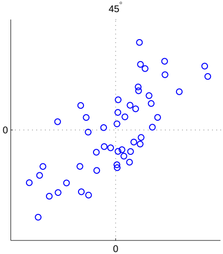
Uncorrelated, scaled



$$\begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{X}$$

$$\mathbf{X}\mathbf{X}^\top = \begin{bmatrix} 9 & 0 \\ 0 & 1 \end{bmatrix}$$

Scaled, rotated by 45°



$$\begin{bmatrix} cos(\phi) & -sin(\phi) \\ sin(\phi) & cos(\phi) \end{bmatrix} \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{X}$$

$$\mathbf{X}\mathbf{X}^\top = \begin{bmatrix} 5 & 4 \\ 4 & 5 \end{bmatrix}$$

Fisher's Linear Discriminant Analysis

$$\underset{w}{\operatorname{argmax}} \frac{\mathbf{w}^\top \mathbf{S}_B \mathbf{w}}{\mathbf{w}^\top \mathbf{S}_W \mathbf{w}} \tag{3}$$

$$\mathbf{S}_B = \underbrace{(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)}_{\text{Distance Class Means}} (\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^\top \tag{4}$$

$$\begin{aligned}\mathbf{S}_W = &\sum_{i \in \mathcal{Y}_{+1}} (\mathbf{x}_i - \boldsymbol{\mu}_+)(\mathbf{x}_i - \boldsymbol{\mu}_+)^\top \\ &+ \sum_{j \in \mathcal{Y}_{-1}} (\mathbf{x}_j - \boldsymbol{\mu}_-) \underbrace{(\mathbf{x}_j - \boldsymbol{\mu}_-)^\top}_{\text{Distance from Class Mean}}\end{aligned} \tag{5}$$

Fisher's Linear Discriminant Analysis

Setting the first derivative of eq. 3 to zero, we obtain the generalized eigenvalue equation

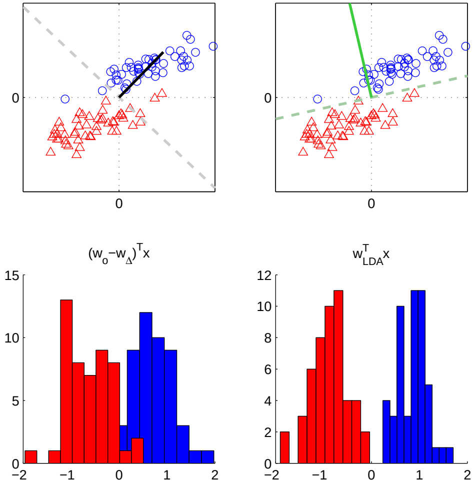
$$\mathbf{S}_B w = \mathbf{S}_W w \lambda \tag{6}$$

Left multiplying with $\mathbf{S}_W^{-1}$ yields

$$\begin{aligned} \mathbf{S}_W^{-1}\mathbf{S}_B w &= \mathbf{S}_W^{-1}\mathbf{S}_W w \lambda \\ \mathbf{S}_W^{-1}(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)\underbrace{(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^\top w}_{\beta} &= w \\ w \propto \mathbf{S}_W^{-1}(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-) \end{aligned} \tag{7}$$

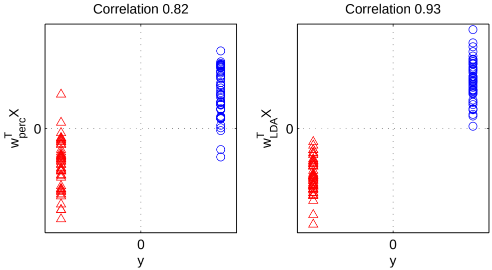
$\rightarrow$ Fisher's LDA first *decorrelates* the data followed by nearest centroid classification

Fisher Linear Discriminant Analysis



Fisher Linear Discriminant Analysis

Another view on LDA:
Maximizing the correlation between class labels $\mathbf{y}$ and $\mathbf{w}^\top \mathbf{X}$:



What if our labels are not $\in \{-1, +1\}$ but $\in \mathbb{R}$?

From Classification to Regression

$y \in \{-1, +1\}$	$y \in \mathbb{R}$
Classification	**Regression**

The most popular and best understood type of regression is **linear regression** using a *least-squares cost function*.

Linear Regression

Target variable $y \in \mathbb{R}$ is modeled as a **linear combination** $w \in \mathbb{R}^N$ of N regressors $\phi(\mathbf{x}) \in \mathbb{R}^N$

$$y = \mathbf{w}^\top \phi(\mathbf{x}) \tag{8}$$

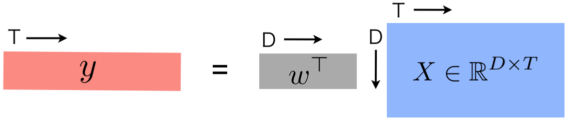
where $\phi(.)$ is one or more (potentially non-linear) function on $\mathbf{x}$.

For the sake of simplicity we assume $\phi(\mathbf{x}) = \mathbf{x}$.

Linear Regression

Let T be the number of samples, so $\mathbf{y} \in \mathbb{R}^{1\times T}$ and $\mathbf{X} \in \mathbb{R}^{N\times T}$. The Linear Regression model in matrix notation then becomes

$$\mathbf{y} = \mathbf{w}^\top \mathbf{X}. \tag{9}$$



Linear Regression

The most popular loss function to optimize w
is the **least-square error** [Gauß, 1809; Legendre, 1805]

$$\mathcal{E}_{lsq}(w) = \sum_{t=1}^{T} (y_t - \mathbf{w}^\top \mathbf{X}_t)^2 \tag{10}$$

C.F. Gauß (1777-1855)

A.M. Legendre (1752-1833)

Linear Regression

To minimize the least-squares loss function in eq. 10

$$\mathcal{E}_{lsq}(w) = (\mathbf{y} - \mathbf{w}^\top \mathbf{X})^2 = \mathbf{y}\mathbf{y}^\top - 2\mathbf{w}^\top \mathbf{X}\mathbf{y}^\top + \mathbf{w}^\top \mathbf{X}\mathbf{X}^\top \mathbf{w}$$

we compute derivative w.r.t. **w**

Linear Regression

To minimize the least-squares loss function in eq. 10

$$\mathcal{E}_{lsq}(w) = (\mathbf{y} - \mathbf{w}^\top \mathbf{X})^2 = \mathbf{y}\mathbf{y}^\top - 2\mathbf{w}^\top \mathbf{X}\mathbf{y}^\top + \mathbf{w}^\top \mathbf{X}\mathbf{X}^\top \mathbf{w}$$

we compute derivative w.r.t. $\mathbf{w}$

$$\frac{\partial \mathcal{E}_{lsq}(\mathbf{w})}{\partial \mathbf{w}} = -2\mathbf{X}\mathbf{y}^\top + 2\mathbf{X}\mathbf{X}^\top \mathbf{w} \tag{11}$$

Linear Regression

To minimize the least-squares loss function in eq. 10

$$\mathcal{E}_{lsq}(w) = (\mathbf{y} - \mathbf{w}^\top \mathbf{X})^2 = \mathbf{y}\mathbf{y}^\top - 2\mathbf{w}^\top \mathbf{X}\mathbf{y}^\top + \mathbf{w}^\top \mathbf{X}\mathbf{X}^\top \mathbf{w}$$

we compute derivative w.r.t. **w**

$$\frac{\partial \mathcal{E}_{lsq}(\mathbf{w})}{\partial \mathbf{w}} = -2\mathbf{X}\mathbf{y}^\top + 2\mathbf{X}\mathbf{X}^\top \mathbf{w} \tag{11}$$

set it to zero and solve for w

$$\begin{aligned} -2\mathbf{X}\mathbf{y}^\top + 2\mathbf{X}\mathbf{X}^\top \mathbf{w} &= 0 \\ \mathbf{X}\mathbf{X}^\top \mathbf{w} &= \mathbf{X}\mathbf{y}^\top \\ \mathbf{w} &= (\underbrace{\mathbf{X}\mathbf{X}^\top}_{\text{Cov. Mat.}})^{-1}\mathbf{X}\mathbf{y}^\top \end{aligned} \tag{12}$$

Linear Regression for Vector Labels

Prediction of vector-valued labels $\mathbf{y} \in \mathbb{R}^M$
is called **Multiple Linear Regression**:

For a measurement $\mathbf{X} \in \mathbb{R}^{N \times T},\ \mathbf{Y} \in \mathbb{R}^{M \times T}$ the MLR model is

$$\mathbf{Y} = \mathbf{W}^\top \mathbf{X} \tag{13}$$

where $\mathbf{W}^\top \in \mathbb{R}^{M \times N}$ is a **linear mapping** from data to labels.

Linear Regression for Vector Labels

Given Data $\mathbf{X} \in \mathbb{R}^{N \times T}$ and labels $\mathbf{Y} \in \mathbb{R}^{M \times T}$
the error function for multiple linear regression is

$$\mathcal{E}_{MLR}(\mathbf{W}) = \sum_{m=1}^{M} (\mathbf{Y}_m - \mathbf{W}_m^\top \mathbf{X})^2 \tag{14}$$

where $\mathbf{Y}_m$ denotes the m-th output dimension
and $\mathbf{W}_m$ the corresponding weight vector

Linear Regression for Vector Labels

Given Data $\mathbf{X} \in \mathbb{R}^{N \times T}$ and labels $\mathbf{Y} \in \mathbb{R}^{M \times T}$
the error function for multiple linear regression is

$$\mathcal{E}_{MLR}(\mathbf{W}) = \sum_{m=1}^{M} (\mathbf{Y}_m - \mathbf{W}_m^\top \mathbf{X})^2 \tag{14}$$

where $\mathbf{Y}_m$ denotes the m-th output dimension
and $\mathbf{W}_m$ the corresponding weight vector

Eq. 14 is minimized by

$$W = (\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X}\mathbf{Y}^\top \tag{15}$$

Linear Discriminant Analysis and Linear Regression

Remember the solution to linear discriminant analysis

$$\mathbf{w}_{LDA} \propto \mathbf{S}^{-1}(\boldsymbol{\mu}_{+} - \boldsymbol{\mu}_{-})$$

Linear Discriminant Analysis and Linear Regression

Remember the solution to linear discriminant analysis

$$\begin{aligned}\mathbf{w}_{LDA} &\propto \mathbf{S}^{-1}(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)\\ &\propto \mathbf{S}^{-1}(\underbrace{(+1)}_{y}\ \underbrace{(1/N_{+1})}_{\gamma_1} \sum_{i\in\mathcal{Y}_{+1}} \mathbf{x}_i + \underbrace{(-1)}_{y}\ \underbrace{(1/N_{-1})}_{\gamma_2} \sum_{j\in\mathcal{Y}_{-1}} \mathbf{x}_j)\end{aligned}$$

Linear Discriminant Analysis and Linear Regression

Remember the solution to linear discriminant analysis

$$\begin{aligned}\mathbf{w}_{LDA} &\propto \mathbf{S}^{-1}(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)\\ &\propto \mathbf{S}^{-1}(\underbrace{(+1)}_{y}\ \underbrace{(1/N_{+1})}_{\gamma_1} \sum_{i\in\mathcal{Y}_{+1}} \mathbf{x}_i + \ \underbrace{(-1)}_{y}\ \underbrace{(1/N_{-1})}_{\gamma_2} \sum_{j\in\mathcal{Y}_{-1}} \mathbf{x}_j)\\ &\propto \mathbf{S}^{-1}\mathbf{X}\mathbf{y}^\top \text{ assuming } N_{+1} = N_{-1}\end{aligned}$$

Linear Discriminant Analysis and Linear Regression

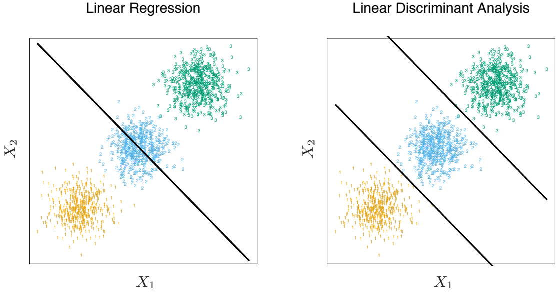
Remember the solution to linear discriminant analysis

$$
\begin{aligned}
\mathbf{w}_{LDA} &\propto \mathbf{S}^{-1}(\boldsymbol{\mu}_{+} - \boldsymbol{\mu}_{-}) \\
&\propto \mathbf{S}^{-1}(\underbrace{(+1)}_{y}\ \underbrace{(1/N_{+1})}_{\gamma_1} \sum_{i \in \mathcal{Y}_{+1}} \mathbf{x}_i + \underbrace{(-1)}_{y}\ \underbrace{(1/N_{-1})}_{\gamma_2} \sum_{j \in \mathcal{Y}_{-1}} \mathbf{x}_j) \\
&\propto \mathbf{S}^{-1}\mathbf{X}\mathbf{y}^{\top} \text{ assuming } N_{+1} = N_{-1}
\end{aligned}
$$

LDA	Linear Regression
$\mathbf{w} \propto \mathbf{S}^{-1}\mathbf{X}\mathbf{y}^{\top}$	$\mathbf{w} = (\mathbf{X}\mathbf{X}^{\top})^{-1}\mathbf{X}\mathbf{y}^{\top}$

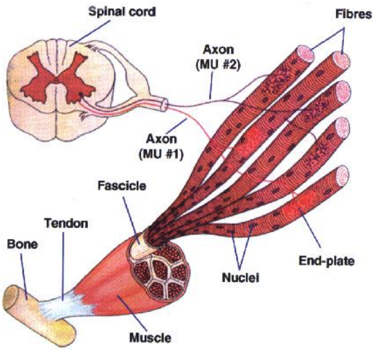
Classification by Linear Regression?

So when coding the class as a boolean multivariate label vector, can we do classification with linear regression?



No, for more than 2 classes, this can lead to poor classification

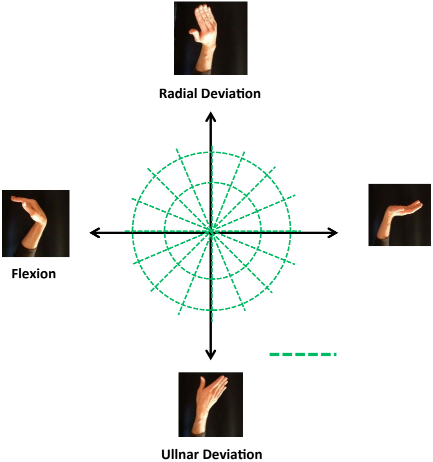
Application Example: Myoelectric Control of Prostheses



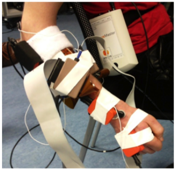
Neurons activate muscles via electric discharges
Electric activity can be measured non-invasively

State-of-the-art hand prosthesis
Only 2 degrees of freedom are controlled (open/close, rotate)
Controlled by muscle activity

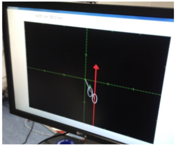
Acquisition of Training Data



Experimental Paradigm

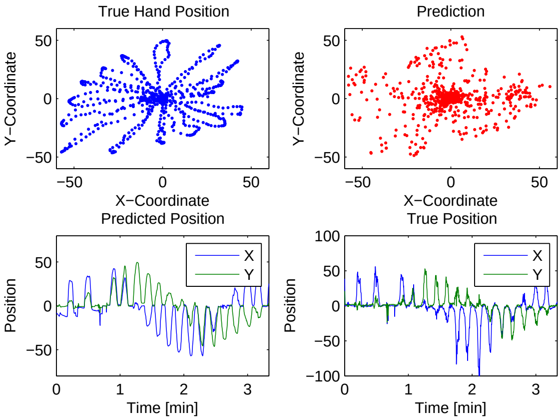


Motion Capture System

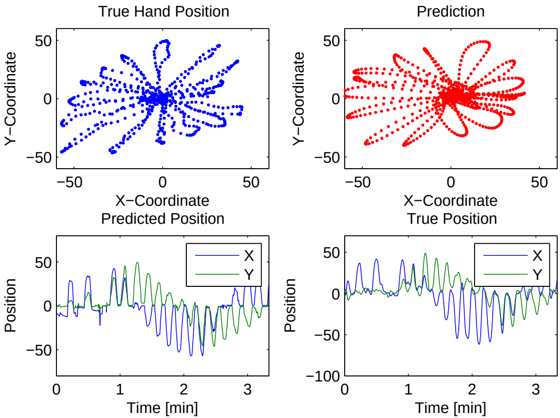


Visual Feedback

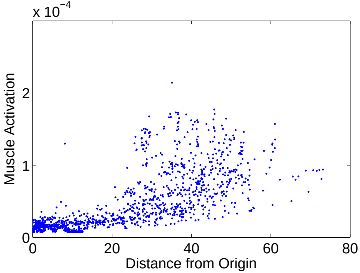
Results Linear Regression



Results Linear Regression - Smoothed

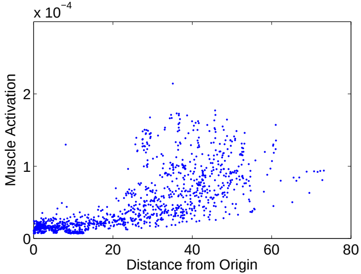


Linear Regression

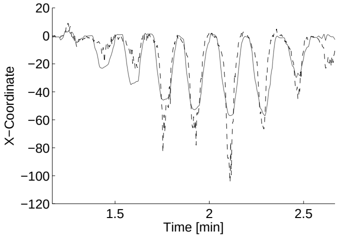


Hand position is a *non-linear* function of muscle activation

Linear Regression



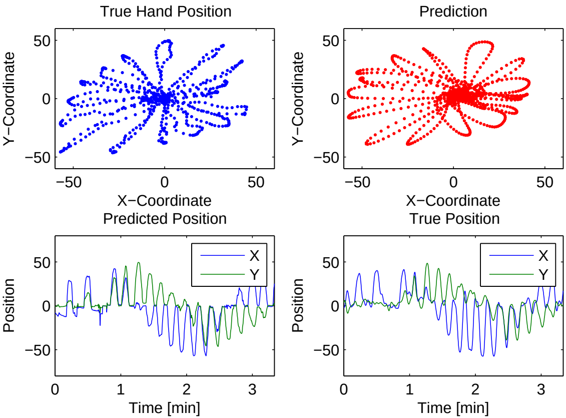
Hand position is a *non-linear* function of muscle activation



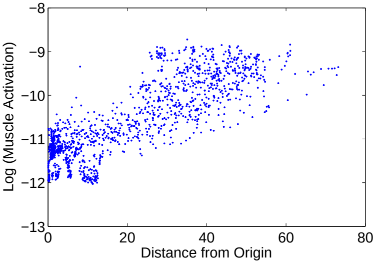
Weak muscle activation
→ True Handposition (dashed) **over**estimated (grey)

Strong muscle activation
→ True Handposition **under**estimated

Results Linear Regression - Smoothed and Log Features

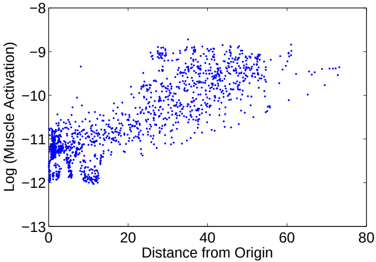


Linear(ized) Regression

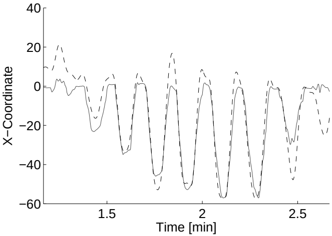


Hand position is *almost linearly* related to log of muscle activation

Linear(ized) Regression



Hand position is *almost linearly* related to log of muscle activation



Weak muscle activation
→ Handposition *less* **under**estimated

Strong muscle activation
→ Handposition *less* **over**estimated

Regularization

Often it is important to **control the complexity** the solution w.

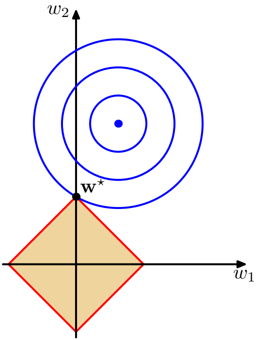
This is done by constraining the $\mathcal{L}_p$-norm of w.

So we add to the least-squares error minimization the constraint

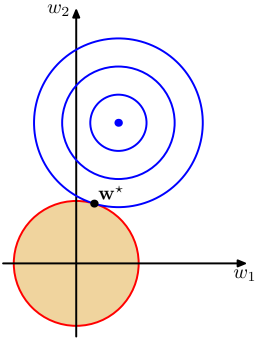
$$||\mathbf{w}||_p = \lambda \tag{16}$$

Regularization

What does this mean geometrically?



$$||\mathbf{w}||_1 = \lambda$$

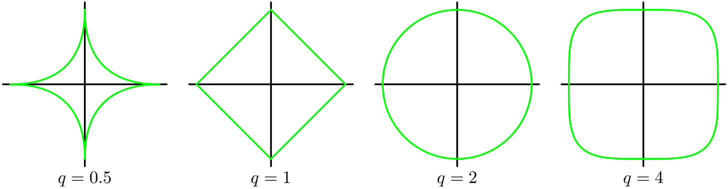


$$||\mathbf{w}||_2 = \lambda$$

The least squares error $\mathcal{E}(\mathbf{w})$ is the same on the blue circles
The $\mathcal{L}_p$-norm of $\mathbf{w}$ is indicated by the red line
The optimal $\mathbf{w}$ is on the intersection of the constraint and the error

Other Regularizers

Other norms than $\mathcal{L}_1$-norm and $\mathcal{L}_2$-norm are



Why $\mathcal{L}_1$-norm and $\mathcal{L}_2$-norm?

- The $\mathcal{L}_2$-norm is analytically tractable:

$$\frac{\partial \|\mathbf{w}\|_2^2}{\partial \mathbf{w}} = \frac{\partial \sqrt{\mathbf{w}^\top \mathbf{w}}^2}{\partial \mathbf{w}} = 2\mathbf{w}$$

$\rightarrow$ Very popular and known as Tikhonov Regularization, Weight Decay, Shrinkage, Ridge Regression, ...

- The $\mathcal{L}_1$-norm is not differentiable (at 0)
- But it has the nice property of leading to **sparse solutions**

$\rightarrow$ Known as: Lasso, Sparse [insert any method here], ...

$\mathcal{L}_2$-norm Regularization: Ridge Regression

For the euklidian norm $p = 2$ the error function is then

$$\mathcal{E}_{RR}(\mathbf{w}) = (\mathbf{y} - \mathbf{w}^\top \mathbf{X})^2 + \lambda ||\mathbf{w}||_2^2 \tag{17}$$

Ridge Regression

Computing the derivative w.r.t. w yields

$$\frac{\partial \mathcal{E}_{RR}(\mathbf{w})}{\partial \mathbf{w}} = -2\mathbf{x}\mathbf{y}^\top + 2\mathbf{X}\mathbf{X}^\top\mathbf{w} + \lambda 2\mathbf{w}. \tag{18}$$

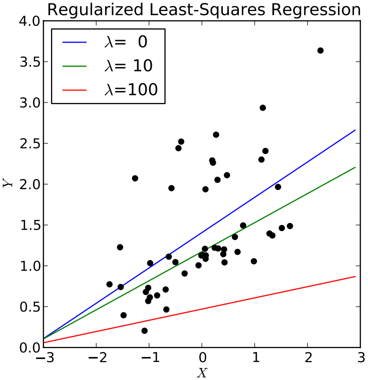
Setting eq. 18 to zero and rearranging terms the optimal **w** is

$$\begin{aligned} 2\mathbf{X}\mathbf{X}^\top\mathbf{w} + \lambda 2\mathbf{w} =& 2\mathbf{X}\mathbf{y}^\top \\ (\mathbf{X}\mathbf{X}^\top + \lambda\mathbf{I})\mathbf{w} =& \mathbf{X}\mathbf{y}^\top \\ \mathbf{w} =& (\mathbf{X}\mathbf{X}^\top + \lambda\mathbf{I})^{-1}\mathbf{X}\mathbf{y}^\top \end{aligned} \tag{19}$$

[Hoerl and Kennar, 1970; Tychonoff, 1943]

Effect of λ on $\mathbf{w}$

Increasing λ will *shrink* coefficients of $\mathbf{w}$ to zero



(Multi-)Linear (Ridge) Regression Algorithm

Algorithm 1 (Multi-)Linear (Ridge) Regression

Require: $\mathbf{x}_i \in \mathbb{R}^U,\ \mathbf{y}_i \in \mathbb{R}^V$, ridge λ
Ensure: Weight matrix $\mathbf{W}$ for linear mapping of $\mathbb{R}^U \to \mathbb{R}^V$
1: Include offset parameters (row vector of N ones)
2: $\mathbf{X} = \begin{bmatrix} \mathbb{1} \\ \mathbf{X} \end{bmatrix}$
3: $\mathbf{W} = (\mathbf{X}\mathbf{X}^\top + \lambda \mathbf{I})^{-1}\mathbf{X}\mathbf{Y}^\top$

$\mathcal{L}_1$-norm Regularization: The Lasso

Choosing the norm $p = 1$ for regression is called the **Lasso** [Tibshirani, 1996]

The error function is

$$\mathcal{E}_{RR}(\mathbf{w}) = (\mathbf{y} - \mathbf{w}^\top \mathbf{X})^2 + \lambda ||\mathbf{w}||_1 \tag{20}$$

There is no closed form solution for this.

Lasso with First Order Stochastic Gradient Descent

Algorithm 2 Lasso

Require: $\mathbf{x}_i, \ldots, \mathbf{x}_N \in \mathbb{R}^D$, $\mathbf{y}_i, \ldots, \mathbf{y}_N \in \mathbb{R}^1$, maximum $\mathcal{L}_1$ norm λ, step size η

Ensure: Weight vector $\mathbf{w}$ for linear mapping of $\mathbb{R}^D \rightarrow \mathbb{R}^1$

Include offset parameters (row vector of N ones)

$\mathbf{X} = \begin{bmatrix} \mathbb{1} \\ \mathbf{X} \end{bmatrix}$

$\#$ Initialize $\mathbf{w} = \mathbf{I}/D$

for $i = 1$ to N_{it} **do**

  $\#$ Draw random data point $\mathbf{x}_i$ and label y_i

  $\mathbf{v} = \mathbf{w}_-$

  $\mathbf{u} = \mathbf{w}_+$

  $v_d \leftarrow \max\left(v_d - \eta/i(\lambda + (y_i - \mathbf{w}^\top \mathbf{x}_i)\mathbf{x}_{i,d}), 0\right)$

  $u_d \leftarrow \max\left(u_d - \eta/i(\lambda - (y_i - \mathbf{w}^\top \mathbf{x}_i)\mathbf{x}_{i,d}), 0\right)$

  $\mathbf{w} = \mathbf{u} - \mathbf{v}$

end for

taken from [Bottou, 2010]

Random Kitchen Sinks

- What if the dependencies between features $\phi(\mathbf{x})$ and labels y are non-linear and we don't know the non-linearity?
- We can use algorithms that learn a non-linear function
  - Multilayer Perceptrons
  - Kernel Methods (next lectures)
- Use Linear Regression and **random basis functions** [Rahimi and Recht, 2008]
- This trick is called *Random Kitchen Sinks*

`http://www.keysduplicated.com/~ali/random-features/`

Random Kitchen Sinks

Algorithm 3 Random Kitchen Sinks

Require: $\mathbf{x}_i, \ldots, \mathbf{x}_N \in \mathbb{R}^D$, $\mathbf{y}_i, \ldots, \mathbf{y}_N \in \mathbb{R}^1$, regularizer λ, number of features F

Ensure: Weight vector $\mathbf{a}$, random basis $\mathbf{W}$

1: # Training

2: $\mathbf{W} \in \mathbb{R}^{D \times F} \sim \mathcal{N}(0, 1)$

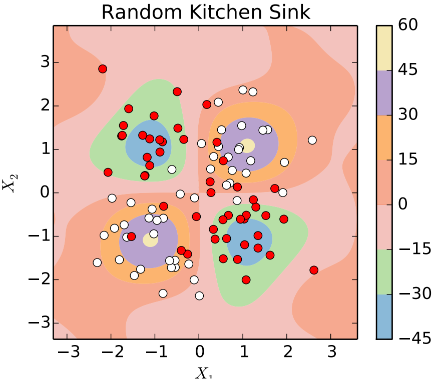
3: $\mathbf{Z} = e^{i\mathbf{W}^\top \mathbf{X}}$

4: $\mathbf{a} = (\mathbf{I}\lambda + \mathbf{Z}\mathbf{Z}^\top)^{-1}\mathbf{Z}\mathbf{y}^\top$

5: # Testing

6: $\hat{\mathbf{y}} = \mathbf{a}^\top e^{i\mathbf{W}^\top \mathbf{X}}$

Random Projections for Solving XOR



Summary

Linear Regression

- is a generic framework for prediction
- straightforwardly extends to vector labels
- can be made more robust by constraining $\|\mathbf{w}\|_p$
  - $p = 1$: Sparse solutions, no closed form solution
  - $p = 2$: Analytically tractable
- Non-linear dependencies:
  - Explicitly model non-linearity (if possible)
  - Random Projections

References

C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer US, 2007.

L. Bottou. Large-scale machine learning with stochastic gradient descent. In Y. Lechevallier and G. Saporta, editors, *Proceedings of the 19th International Conference on Computational Statistics (COMPSTAT'2010)*, pages 177–187, Paris, France, 2010. Springer.

C. F. Gauß. Theoria motus corporum coelestium in sectionibus conicis solem ambientium. *Göttingen*, 1809.

T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. 2003.

A. E. Hoerl and R. W. Kennar. Ridge regression: Applications to nonorthogonal problems. *Technometrics*, 12(1):69–82, 1970.

A.-M. Legendre. *Nouvelles méthodes pour la détermination des orbites des comètes*, chapter Sur la methode des moindres quarres. Firmin Didot, http://imgbase-scd-ulp.u-strasbg.fr/displayimage.php?pos=-141297, 1805.

K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. Adaptive Computation and Machine Learning. The MIT Press, 1 edition, 2012. ISBN 0262018020,9780262018029.

A. Rahimi and B. Recht. Weighted Sums of Random Kitchen Sinks: Replacing minimization with randomization in learning. In D. Koller, D. Schuurmans, Y. Bengio, L. Bottou, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *NIPS*, pages 1313–1320. MIT Press, 2008.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1996.

A. N. Tychonoff. On the stability of inverse problems. *Doklady Akademii Nauk SSSR*, 39(5):195–198, 1943.