HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.

ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

Machine Learning

IT3190E

Lecture: Linear Regression

ONE LOVE. ONE FUTURE.

Contents

Regression

- There is an *unknown* function y^* that maps each **x** to a number $y^*(\boldsymbol{x})$
  - In practice, we can collect some pairs: ($\mathbf{x}_i$, y_i), where $y_i = y^*(\boldsymbol{x}_i)$.
  - Each observation of **x** is represented by a vector in an n-dimensional space, e.g., $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{in})^T$. Each dimension represents an *attribute (thuộc tính) or feature (đặc trưng) or variate*.
  - Bold characters denote vectors or matrices.
- **Regression problem:** learn a function y = f(**x**) from a given training set **D** = $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_M, y_M)\}$ such that $y_i \cong f(\boldsymbol{x}_i)$ for every i

Linear regression (Hồi quy tuyến tính)

- Linear model: assume that $y^*(\boldsymbol{x})$ can be well approximated by

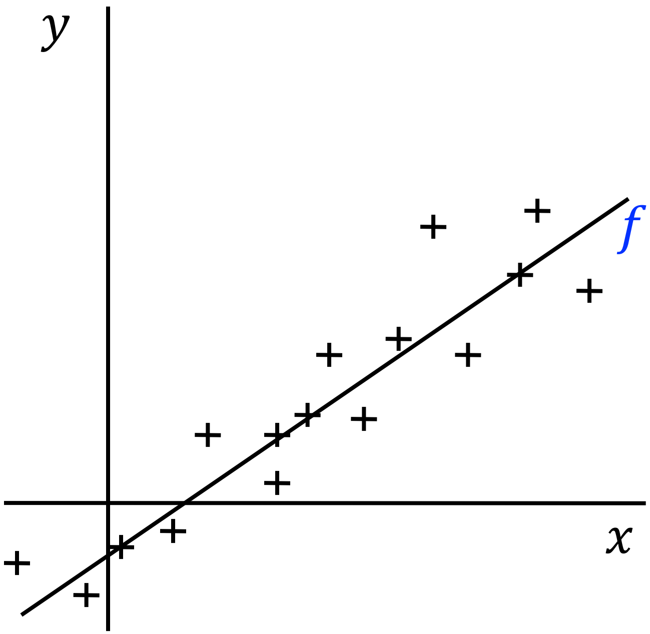
$$f(\mathbf{x},\mathbf{w}) = w_0 + w_1x_1 + \ldots + w_nx_n$$

  - $w_0, w_1, \ldots, w_n$ are the regression coefficients/weights. w_0 sometimes is called "*bias*".
  - In other words, we use a hyperplane to approximate the unknown function.
  - $f(\mathbf{x},\mathbf{w})$ may not be linear in $\mathbf{x}$.
- *Note:* learning a linear model is equivalent to finding the weight vector $\mathbf{w} = (w_0, w_1, \ldots, w_n)^T$.

Linear regression: example

- What is the best function?

x	y
0.13	-0.91
1.02	-0.17
3.17	1.61
-2.76	-3.31
1.44	0.18
5.28	3.36
-1.74	-2.46
7.93	5.56
...	...



Prediction

- For each observation $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T$
  - The *true output*: $y^*(\boldsymbol{x})$ (but unknown for future data)
  - *Prediction* by our linear model:

$$y_x = w_0 + w_1 x_1 + \ldots + w_n x_n$$

  - We often expect $y_x \cong y^*(\boldsymbol{x})$.

- Prediction for a future observation $\mathbf{z} = (z_1, z_2, \ldots, z_n)^T$
  - Use the learned function to make prediction

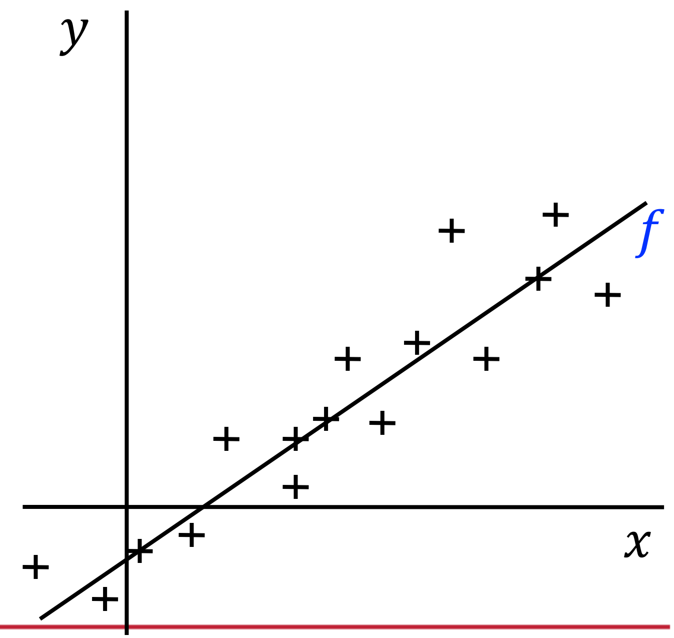
$$f(\mathbf{z},\mathbf{w}) = w_0 + w_1 z_1 + \ldots + w_n z_n$$

Learning a regression function

- Learning goal: *learn a function f* such that its prediction in the future is the best.*
  - Its generalization is the best.
- Difficulty: infinite number of functions

$$\boldsymbol{H} = \{ f(\boldsymbol{x}, \boldsymbol{w}) \colon \boldsymbol{w} = (w_0, w_1, \dots, w_n) \in \mathbb{R}^{n+1} \}$$

  - How can we learn?
  - Is function f better than g?
- Use a measure
  - *Loss function* is often used to guide learning.



Loss function

- The *error/loss* of the prediction for an example **x** = $(x_1, x_2, \ldots, x_n)^T$:

$$r(f, \boldsymbol{x}) = [y^*(\boldsymbol{x}) - f(\boldsymbol{x}, \boldsymbol{w})]^2$$

Cost, risk

- The *expected loss (risk) of f* over the whole space:

$$E = \boldsymbol{E}_x[r(f, \boldsymbol{x})] = \boldsymbol{E}_x[y^*(\boldsymbol{x}) - f(\boldsymbol{x}, \boldsymbol{w})]^2$$

($\mathbf{E}_x$ is the expectation over **x**)

- About the loss/cost: $r(f, \boldsymbol{x})$
  - Square loss is used above. Other loss functions can be used, e.g.
    - *Absolute loss:* $|y^*(\boldsymbol{x}) - f(\boldsymbol{x}, \boldsymbol{w})|$
    - *Hinge loss:* $\max\{0, 1 - y^*(\boldsymbol{x})\, f(\boldsymbol{x}, \boldsymbol{w})\}$
    - …

Loss function

- The goal of learning is to find f* that minimizes the expected loss:

$$f^* = \arg\min_{f \in \boldsymbol{H}} \boldsymbol{E}_x[r(f, \boldsymbol{x})]$$

  - For linear model: **H** is the space of functions of linear form.
- But we cannot work directly with this problem during the learning phase.
  (Why?)

Empirical loss

- We can observe a data set **D** = {($\mathbf{x}_1$, y_1), ($\mathbf{x}_2$, y_2), …, ($\mathbf{x}_M$, y_M)}, and have to learn f from **D**.
- Residual sum of squares:

$$RSS(f) = \sum_{i=1}^{M} \big(y_i - f(\boldsymbol{x}_i, \boldsymbol{w})\big)^2 = \sum_{i=1}^{M} (y_i - w_0 - w_1 x_{i1} - \cdots - w_n x_{in})^2$$

- Empirical loss (lỗi thực nghiệm): $L(f, \mathbf{D}) = \frac{1}{M} RSS(f)$
  - $L(f, \mathbf{D})$ is an approximation of $\mathbf{E}_x[r(\mathbf{x})]$.
- $|L(f, \mathbf{D}) - \boldsymbol{E}_x[r(\boldsymbol{x})]|$ is often known as *generalization error* (lỗi tổng quát hoá) of *f*.
- Many learning algorithms base on this RSS or its variants.

Methods: ordinary least squares (OLS)

- Given **D**, we find f* that minimizes RSS:

$$f^* = \arg\min_{f \in \boldsymbol{H}} RSS(f) \qquad (1)$$

$$\Leftrightarrow \boldsymbol{w}^* = \arg\min_{\mathbf{w}} \sum_{i=1}^{M} (y_i - w_0 - w_1 x_{i1} - \cdots - w_n x_{in})^2$$

- This method is often known as *ordinary least squares (OLS, bình phương tối thiểu).*
- Find **w*** by taking the gradient of RSS and solving the equation RSS'=0. We have:

$$\boldsymbol{w}^* = \left(\boldsymbol{A}^T \boldsymbol{A}\right)^{-1} \boldsymbol{A}^T \boldsymbol{y}$$

  - Where $\mathbf{A}$ is the data matrix of size $M \times (n+1)$, where the i^{th} row is $\mathbf{A_i} = (1, x_{i1}, x_{i2}, \ldots, x_{in})$; $\mathbf{B}^{-1}$ is the inversion of matrix $\mathbf{B}$; $\mathbf{y} = (\mathrm{y}_1, \mathrm{y}_2, \ldots, \mathrm{y}_\mathrm{M})^\mathrm{T}$.
  - Note: we assume that $\mathbf{A}^T\mathbf{A}$ is invertible (ma trận $\mathbf{A}^T\mathbf{A}$ khả nghịch).

Methods: OLS

- Input: $\mathbf{D} = \{(\mathbf{x}_1, \mathrm{y}_1), (\mathbf{x}_2, \mathrm{y}_2), \dots, (\mathbf{x}_\mathrm{M}, \mathrm{y}_\mathrm{M})\}$
- Output: $\boldsymbol{w}^*$
- Learning: compute

$$\boldsymbol{w}^* = \left(\boldsymbol{A}^T\boldsymbol{A}\right)^{-1}\boldsymbol{A}^T\boldsymbol{y}$$

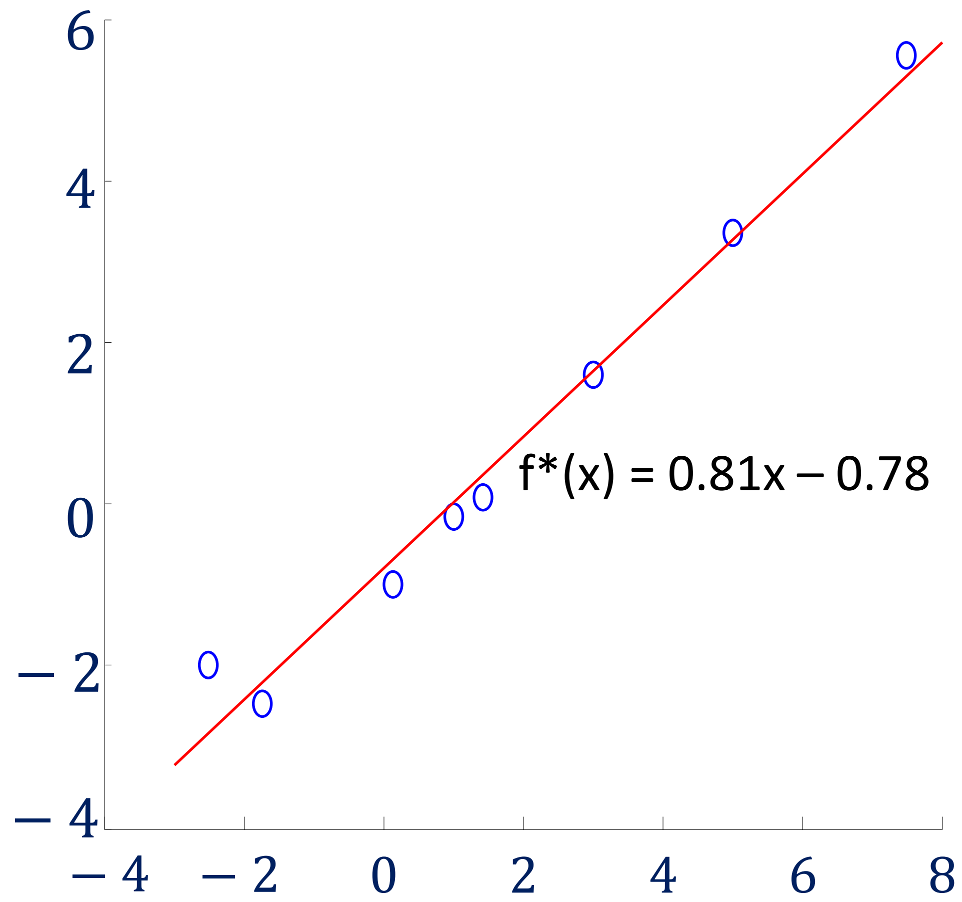
  - Where $\mathbf{A}$ is the data matrix of size $M \times (n+1)$, where the $\mathrm{i^{th}}$ row is $\mathbf{A_i} = (1, x_{i1}, x_{i2}, \dots, x_{in})$; $\mathbf{B}^{-1}$ is the inversion of matrix $\mathbf{B}$;

    $\mathbf{y} = (\mathrm{y}_1, \mathrm{y}_2, \dots, \mathrm{y}_\mathrm{M})^\mathrm{T}$.
  - Note: we assume that $\boldsymbol{A}^T\boldsymbol{A}$ is invertible.

- Prediction for a new **x**: $y_x = w_0^* + w_1^* x_1 + \cdots + w_n^* x_n$

Methods: OLS example

x	y
0.13	−1
1.02	−0.17
3	1.61
−2.5	−2
1.44	0.1
5	3.36
−1.74	−2.46
7.5	5.56



Methods: limitations of OLS

- OLS cannot work if $\mathbf{A}^T\mathbf{A}$ is not invertible
  - If some columns (attributes/features) of **A** are dependent, then **A** will be singular and therefore $\mathbf{A}^T\mathbf{A}$ is not invertible.
    (Nếu một vài cột của A phụ thuộc tuyến tính thì A sẽ không khả nghịch)
- OLS requires considerable computation due to the need of computing a matrix inversion.
  - Intractable for the very high dimensional problems.
- OLS likely tends to overfitting, because the learning phase just focuses on minimizing the error of the training data.

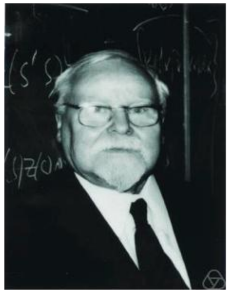
Methods: Ridge

- Given **D** = {($\mathbf{x}_1$, y_1), ($\mathbf{x}_2$, y_2), …, ($\mathbf{x}_M$, y_M)}, we solve for:

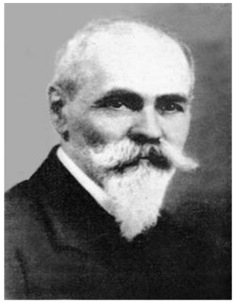
$$f^* = \arg\min_{f \in \boldsymbol{H}} RSS(f) + \lambda \|\boldsymbol{w}\|_2^2$$

$$\Leftrightarrow \boldsymbol{w}^* = \arg\min_{\mathbf{w}} \sum_{i=1}^{M} (y_i - \boldsymbol{A}_i \boldsymbol{w})^2 + \lambda \sum_{j=0}^{n} w_j^2 \quad (2)$$

  - Where λ is a regularization constant ($\lambda > 0$), $\|\boldsymbol{w}\|_2$ is the L^2 norm.



Tikhonov, smoothing an ill-posed problem



Zaremba, model complexity minimization

Bayes: priors over parameters

Andrew Ng: need no maths, but it prevents overfitting!

Methods: Ridge

- Problem (2) is equivalent to the following:

$$w^* = \arg\min_{\boldsymbol{w}} \sum_{i=1}^{M} (y_i - \boldsymbol{A}_i \boldsymbol{w})^2 \qquad \text{Subject to } \sum_{j=0}^{n} w_j^2 \leq t \tag{3}$$

  - for some constant *t*.

- The **regularization/penalty** term: $\lambda ||\boldsymbol{w}||_2^2$
  - Limits the magnitute/size of **w*** (i.e., reduces the search space for f*).
  - Helps us to trade off between *the fitting of f on* ***D*** and *its generalization* on future observations.

Methods: Ridge

- We solve for **w*** by taking the gradient of the objective function in (2), and then zeroing it. Therefore we obtain:

$$\boldsymbol{w}^* = \left(\boldsymbol{A}^T\boldsymbol{A} + \lambda\boldsymbol{I}_{n+1}\right)^{-1}\boldsymbol{A}^T\boldsymbol{y}$$

  - Where **A** is the data matrix of size $M \times (n+1)$, where the i^{th} row is $\mathbf{A}_i = (1, x_{i1}, x_{i2}, \ldots, x_{in})$; $\mathbf{y} = (y_1, y_2, \ldots, y_M)^T$; $\mathbf{I}_{n+1}$ is the identity matrix of size $n+1$.
- Compared with OLS, Ridge can
  - Avoid the cases of singularity, unlike OLS. Hence Ridge always works.
  - Reduce overfitting.
  - Increase the error for the training set.
- Note: *the predictiveness of Ridge depends heavily on the choice of λ.*

Methods: Ridge

- Input: **D** = {($\mathbf{x}_1$, y_1), ($\mathbf{x}_2$, y_2), …, ($\mathbf{x}_M$, y_M)} and $\lambda > 0$
- Output: **w***
- Learning: compute

$$\boldsymbol{w}^* = \left(\boldsymbol{A}^T\boldsymbol{A} + \lambda\boldsymbol{I}_{n+1}\right)^{-1}\boldsymbol{A}^T\boldsymbol{y}$$

- Prediction for a new **x**:

$$y_x = w_0^* + w_1^* x_1 + \cdots + w_n^* x_n$$

- *Note:* to avoid some negative effects of the magnitude of y on covariates **x**, one should remove w_0 from the penalty term in (2). In this case, the solution of **w*** should be modified slightly.

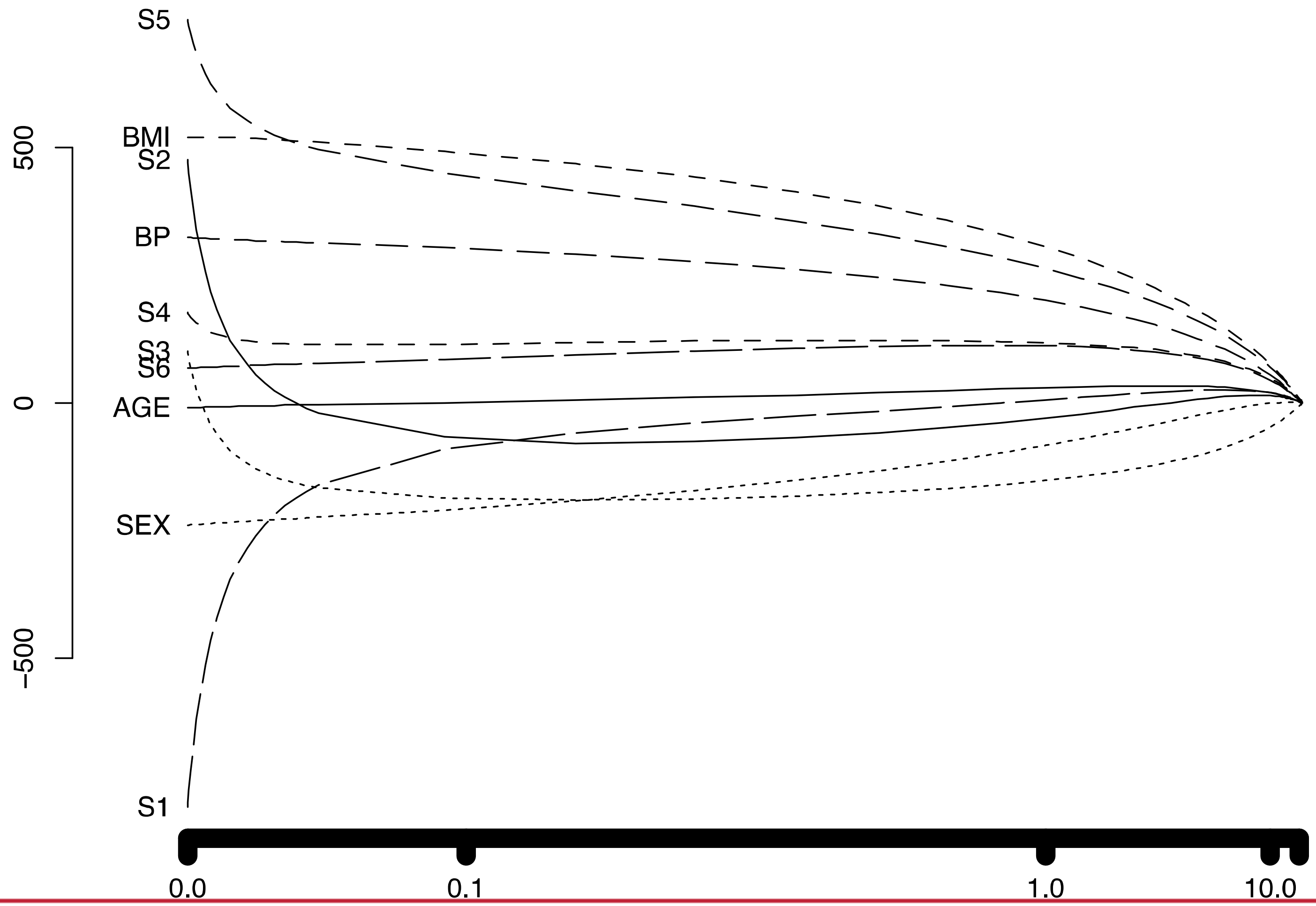
An example of using Ridge and OLS

- The training set **D** contains 67 observations on prostate cancer, each was represented with 8 attributes. Ridge and OLS were learned from **D**, and then predicted 30 new observations.

w	Ordinary Least Squares	Ridge
0	2.465	2.452
lcavol	0.680	0.420
lweight	0.263	0.238
age	−0.141	−0.046
lbph	0.210	0.162
svi	0.305	0.227
lcp	−0.288	0.000
gleason	−0.021	0.040
pgg45	0.267	0.133
Test RSS	**0.521**	**0.492**

Effects of λ in Ridge regression

- **W*** = (w_0, S1, S2, S3, S4, S5, S6, AGE, SEX, BMI, BP) changes as the regularization constant λ changes.



LASSO

- Ridge regression use L^2 norm for regularization:

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \sum_{i=1}^{M} (y_i - \boldsymbol{A}_i \boldsymbol{w})^2 \text{, subject to } \sum_{j=0}^{n} w_j^2 \leq t \quad (3)$$

- Replacing L^2 by L^1 norm will result in LASSO:

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \sum_{i=1}^{M} (y_i - \boldsymbol{A}_i \boldsymbol{w})^2$$

$$\text{Subject to } \sum_{j=0}^{n} |w_j| \leq t$$

- Equivalently:

$$w^* = \arg\min_{\boldsymbol{w}} \sum_{i=1}^{M} (y_i - \boldsymbol{A}_i \boldsymbol{w})^2 + \lambda \|\boldsymbol{w}\|_1 \quad (4)$$

- This problem is non-differentiable → the training algorithm should be more complex than Ridge.

LASSO: regularization role

- The regularization types lead to different domains for **w**.
- LASSO often produces **sparse** solutions, i.e., many components of **w** are zero.
  - Shinkage and selection at the same time

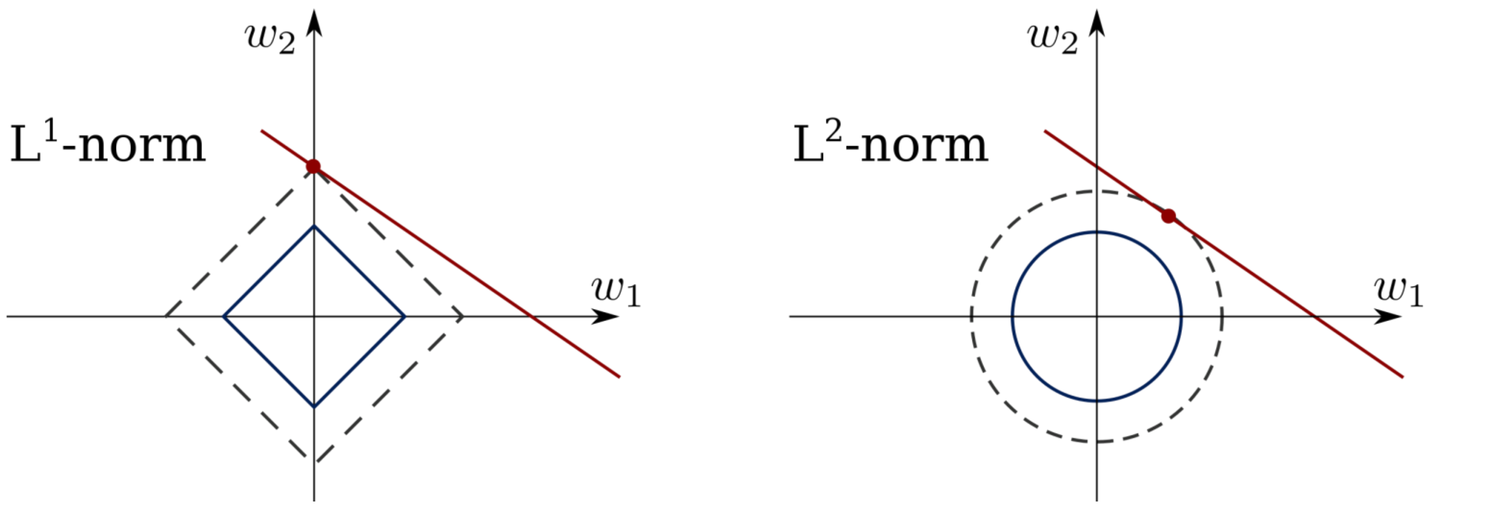


Figure by Nicoguaro - Own work, CC BY 4.0,
https://commons.wikimedia.org/w/index.php?curid=58258966

OLS, Ridge, and LASSO

- The training set **D** contains 67 observations on prostate cancer, each was represented with 8 attributes. OLS, Ridge, and LASSO were trained from **D**, and then predicted 30 new observations.

w	Ordinary Least Squares	Ridge	LASSO
0	2.465	2.452	2.468
lcavol	0.680	0.420	0.533
lweight	0.263	0.238	0.169
age	−0.141	−0.046	
lbph	0.210	0.162	0.002
svi	0.305	0.227	0.094
lcp	−0.288	0.000	
gleason	−0.021	0.040	
pgg45	0.267	0.133	
Test RSS	**0.521**	**0.492**	**0.479**

Some weights are 0
→ some attributes may not be important

References

- Hesterberg, T., Choi, N. H., Meier, L., & Fraley, C. (2008). Least angle and L1 penalized regression: A review. *Statistics Surveys.*
- Trevor Hastie, Robert Tibshirani, Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2009.
- Tibshirani, Robert (1996). Regression Shrinkage and Selection via the lasso. *Journal of the Royal Statistical Society. Series B (methodological)*. Wiley. 58 (1): 267–88.

THANK YOU !

hust.edu.vn fb.com/dhbkhn