Machine Learning

(Học máy – IT3190E)

Khoat Than
School of Information and Communication Technology
Hanoi University of Science and Technology

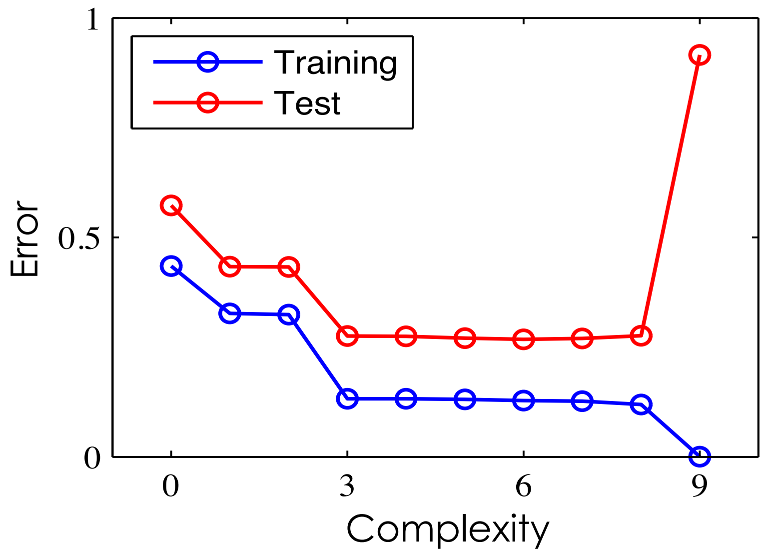
2022

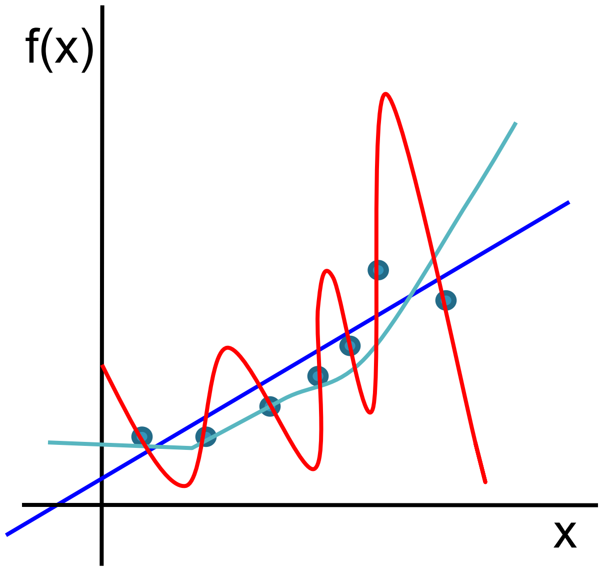
Contents

- Introduction to Machine Learning
- Supervised learning
- Probabilistic modeling
- **Regularization**
- Reinforcement learning
- Practical advice

Revisiting overfiting

- The complexity of the learned function: $y = \hat{f}(x, \boldsymbol{D})$
  - For a given training data **D**: *the more complicated* $\hat{f}$*, the more possibility that* $\hat{f}$ *fits* ***D*** *better.*
  - For a given **D**: there exist many functions that fit **D** perfectly (i.e., no error on **D**).
  - However, those functions might generalize badly.





The Bias-Variance Decomposition

- Consider $y = f(x) + \epsilon$ as the regression function
  - where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ is a Gaussian noise with mean 0 and variance σ^2.
  - ϵ may represent the *noise* due to measurement or data collection.
- Let $\hat{f}(x; \boldsymbol{D})$ be the regressor learned from a training data **D**
- Note:
  - We want that $\hat{f}(x; \boldsymbol{D})$ approximates the truth $f(x)$ well.
  - $\hat{f}(x; \boldsymbol{D})$ is random, according to the randomness when collecting **D**.
- For any x, the error made by $\hat{f}(x; \boldsymbol{D})$ is

$$\mathbb{E}_{D,\epsilon}\left(y(x) - \hat{f}(x; \boldsymbol{D})\right)^2 = \sigma^2 + Bias^2\left(\hat{f}(x; \boldsymbol{D})\right) + Var\left(\hat{f}(x; \boldsymbol{D})\right)$$

  - $Bias\left(\hat{f}(x; \boldsymbol{D})\right) = \mathbb{E}_D\left[f(x) - \hat{f}(x; \boldsymbol{D})\right]$
  - $Var\left(\hat{f}(x; \boldsymbol{D})\right) = \mathbb{E}_D\left(\hat{f}(x; \boldsymbol{D}) - \mathbb{E}_D\hat{f}(x; \boldsymbol{D})\right)^2$

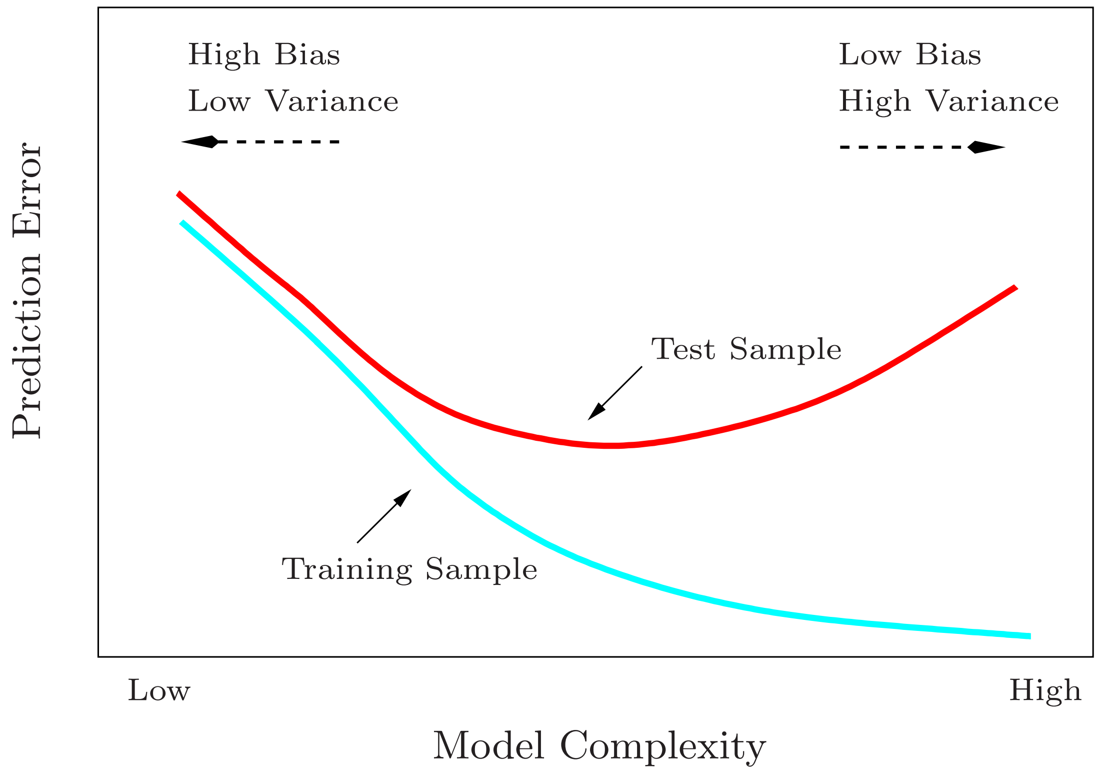
The Bias-Variance Decomposition (2)

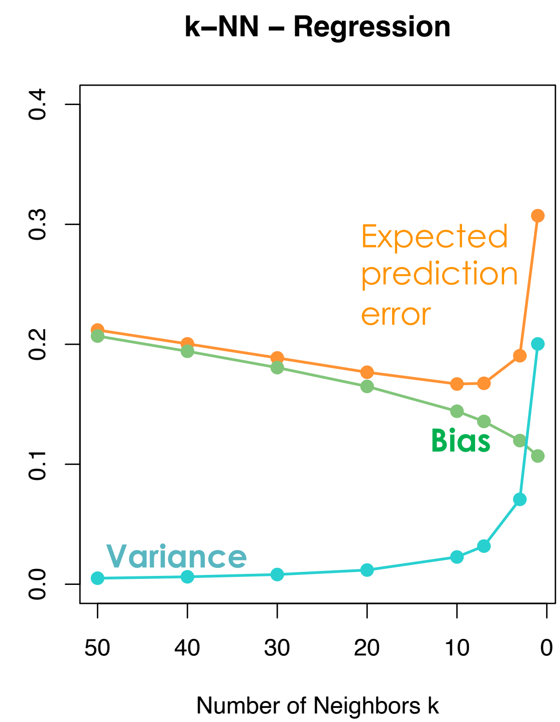
$$
\begin{aligned}
Error(x) &= \sigma^2 + Bias^2\left(\hat{f}(x;\boldsymbol{D})\right) + Var\left(\hat{f}(x;\boldsymbol{D})\right) \\
&= Irreducible\ Error + Bias^2 + Variance
\end{aligned}
$$

- This is known as the **Bias-Variance Decomposition**
  - *Irreducible Error:* cannot be avoided due to noises or uncontrolled factors
  - *Bias:* the average of our estimate differs from the true mean
  - *Variance:* the expected squared deviation of $\hat{f}(x;\boldsymbol{D})$ around its mean

Bias-Variance tradeoff: classical view

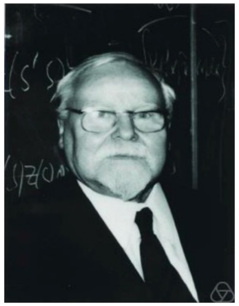
- The more complex the model $\hat{f}(x; \boldsymbol{D})$ is, the more data points it can capture, and the lower the bias can be.
  - However, higher complexity will make the model "move" more to capture the data points, and hence its variance will be larger.



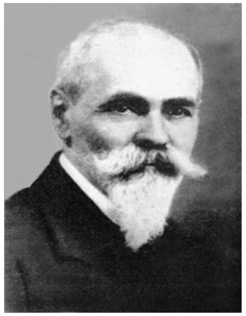


Regularization: introduction

- *Regularization* is now a popular and useful technique in ML.
- It is a technique to exploit further information to
  - Reduce overfitting in ML.
  - Solve ill-posed problems in Maths.
- The further information is often enclosed in a *penalty on the complexity* of $\hat{f}(x, \boldsymbol{D})$.
  - More penalty will be imposed on complex functions.
  - We prefer simpler functions among all that fit well the training data.



Tikhonov, smoothing an ill-posed problem



Zaremba, model complexity minimization

Bayes: priors over parameters

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

Regularization in Ridge regression

- Learning a linear regressor by ordinary least squares (OLS) from a training data $\boldsymbol{D} = \{(x_1, y_1), \dots, (x_M, y_M)\}$ is reduced to the following problem:

$$w^* = \arg\min_{\mathrm{w}} RSS(w, \boldsymbol{D}) + \lambda \|w\|_2^2 = \arg\min_{\mathrm{w}} \sum_{(x_i, y_i) \in \boldsymbol{D}} (y_i - w^T x_i)^2$$

- For Ridge regression, learning is reduced to

$$w^* = \arg\min_{\mathrm{w}} RSS(w, \boldsymbol{D}) + \lambda \|w\|_2^2$$

  - Where λ is a positive constant.
  - The term $\lambda \|w\|_2^2$ plays the role as *limiting the size/complexity of w.*
  - λ allows us to trade off between fitness on **D** and generalization on future observations.

- Ridge regression is a regularized version of OLS.

Regularization: the principle

- We need to learn a function $f(x, w)$ from the training set **D**
  - x is a data example and belongs to **input space**.
  - w is the parameter and often belongs to a ***parameter space*** **W**.
  - $\boldsymbol{F} = \{f(x, w) : w \in \boldsymbol{W}\}$ is the **function space**, parameterized by w.
- For many ML models, the training problem is often reduced to the following optimization:

$$w^* = \arg\min_{w \in \boldsymbol{W}} L(f(x, w), \boldsymbol{D}) \tag{1}$$

  - w sometimes tells the size/complexity of that function.
  - $L(f(x, w), \boldsymbol{D})$ is an *empirical loss/risk* which depends on **D**. This loss shows how well function *f* fits **D**.
- Another view: $f^* = \arg\min_{f \in \boldsymbol{F}} L(f(x, w), \boldsymbol{D})$

Regularization: the principle

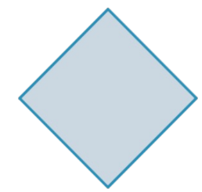
- Adding a penalty to (1), we consider

$$w^* = \arg\min_{w \in \boldsymbol{W}} L(f(x, w), \boldsymbol{D}) + \lambda g(w) \qquad (2)$$

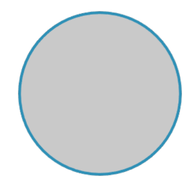
  - Where $\lambda > 0$ is called *the regularization/penalty constant.*
  - $g(w)$ measures the complexity of w. ($g(w) \geq 0$)
- $L(f(x, w), \boldsymbol{D})$ measures the goodness of function *f* on **D**.
- The penalty (regularization) term: $\lambda g(w)$
  - Allows to trade off the fitness on **D** and the generalization.
  - The greater λ, the heavier penalty, implying that $g(w)$ should be smaller.
  - In practice, λ should be neither too small nor too large.
    (λ không nên quá lớn hoặc quá bé trong thực tế)

Regularization: popular types

- $g(w)$ often relates to some norms when w is an n-dimensional vector.
  - L_0-norm: $||w||_0$ counts the number of non-zeros in w.
  - L_1-norm: $\|w\|_1 = \sum_{i=1}^{n} |w|$



  - L_2-norm: $\|w\|_2^2 = \sum_{i=1}^{n} w_i^2$



  - L_p-norm: $\|w\|_p = \sqrt[p]{|w_1|^p + \ldots + |w_n|^p}$

Regularization in Ridge regression

- Ridge regression can be derived from OLS by adding a penalty term into the objective function when learning.
- Learning a regressor in Ridge is reduced to

$$w^* = \arg\min_{\mathbf{w}} RSS(w, \boldsymbol{D}) + \lambda \|w\|_2^2$$

  - Where λ is a positive constant.
  - The term $\lambda \|w\|_2^2$ plays the role as regularization.
  - Large λ reduces the size of w.

Regularization in Lasso

- Lasso [Tibshirani, 1996] is a variant of OLS for linear regression by using L_1 to do regularization.
- Learning a linear regressor is reduced to

$$w^* = \arg\min_{w} RSS(w, \boldsymbol{D}) + \lambda ||w||_1$$

  - Where λ is a positive constant.
  - $\lambda ||w||_1$ is the regularization term. Large λ reduces the size of w.
- Regularization here amounts to imposing a Laplace distribution (as prior) over each w_i, with density function:

$$p(w_i|\lambda) = \frac{\lambda}{2} e^{-\lambda |w_i|}$$

  - The larger λ, the more possibility that $w_i = 0$.

Regularization in SVM

- Learning a classifier in SVM is reduced to the following problem:
  - Minimize $$\frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2}$$
  - Conditioned on $$y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1, \quad \forall i = 1..r$$
- In the cases of noises/errors, learning is reduced to
  - Minimize $$\frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2} + C \sum_{i=1}^{r} \xi_i$$
  - Conditioned on $$\begin{cases} y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1 - \xi_i, & \forall i = 1..r \\ \xi_i \geq 0, & \forall i = 1..r \end{cases}$$
- $C(\xi_1 + \ ... + \xi_r)$ is *the regularization term*.

Some other regularizations

- **Dropout:** (by Hilton and his colleagues, 2012)
  - At each iteration of the training process, randomly drop out some parts and just update the other parts of our model.
- **Batch normalization** [Ioffe & Szegedy, 2015]
  - Normalize the inputs at each neuron of a neural network
  - Reduce input variance, easier training, faster convergence
- **Data augmentation**
  - Produce different versions of an example in the training set, by adding simple noises, translation, rotation, cropping, …
  - Those versions are added to the training data set
- **Early stopping**
  - Stop training early to avoid overtraining & reduce overfitting

Regularization: MAP role

- Under some conditions, we can view regularization as

$$w^* = \arg\min_{w \in \boldsymbol{W}} L(f(x,w), \boldsymbol{D}) + \lambda g(w)$$

Likelihood Prior

  - Where **D** is a sample from a probability distribution whose log likelihood is $-L(f(x,w), \boldsymbol{D})$.
  - w is a random variable and follows the prior with density

$$p(w) \propto \exp(-\lambda g(w))$$

- Then $$w^* = \arg\max_{w \in \boldsymbol{W}} \{-L(f(x,w), \boldsymbol{D}) - \lambda g(w)\}$$

$$w^* = \arg\max_{w \in \boldsymbol{W}} \log \Pr(\boldsymbol{D}|w) + \log \Pr(w) = \arg\max_{w \in \boldsymbol{W}} \log \Pr(w|\boldsymbol{D})$$

- As a result, regularization in fact helps us to learn an MAP solution w*.

Regularization: MAP in Ridge

- Consider the Gaussian regression model:
  - w follows a Gaussian prior: N(w|0, $\sigma^2\rho^2$).
  - Variable f = y – w^Tx follows the Gaussian distribution N(f|0,ρ^2,w) with mean 0 and variance ρ^2, and conditioned on w.
- Then the MAP estimation of f from the training data **D** is

$$w^* = \operatorname{argmax}_w \log \Pr(w \mid D) = \operatorname{argmax}_w \log\left[\Pr(D \mid w) * \Pr(w)\right]$$

$$= \operatorname{argmin}_w \sum_{(x_i,y_i)} \frac{1}{2\rho^2}\left(y_i - w^T x_i\right)^2 + \frac{1}{2\sigma^2\rho^2} w^T w - \text{constant}$$

$$= \operatorname{argmin}_w \sum_{(x_i,y_i)} \left(y_i - w^T x_i\right)^2 + \frac{1}{\sigma^2} w^T w$$

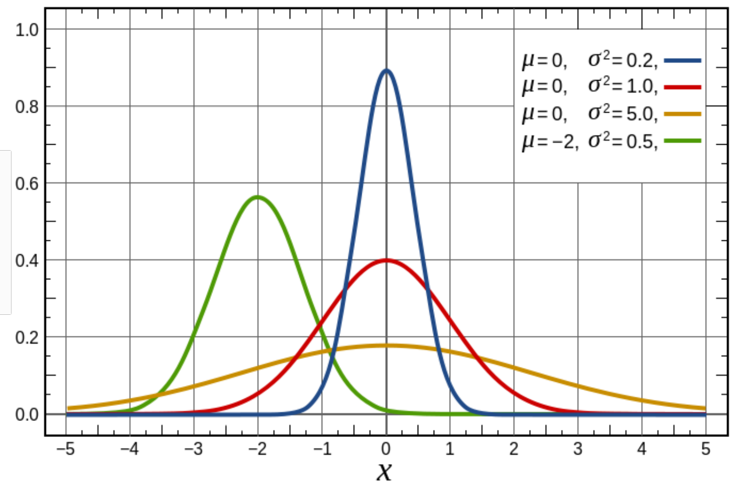
Ridge regression @@

- *Regularization using L_2 with penalty constant $\lambda = \sigma^{-2}$.*

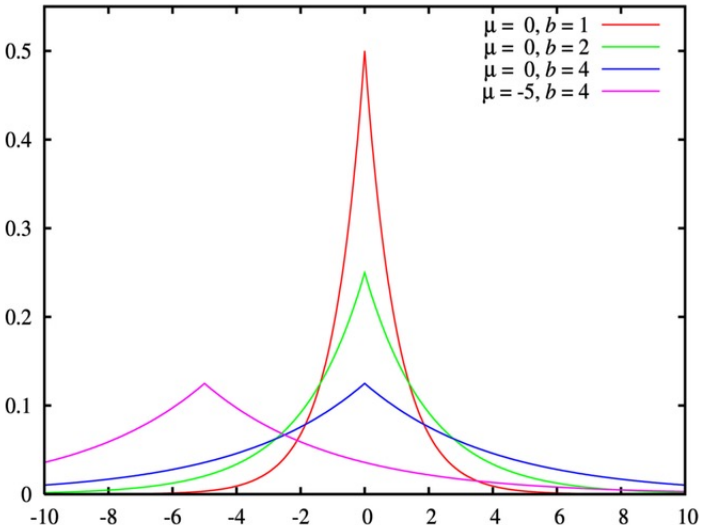
Regularization: MAP in Ridge & Lasso

- The regularization constant in Ridge: $\lambda = \sigma^{-2}$
- The regularization constant in Lasso: $\lambda = b^{-1}$
- Gaussian (left) and Laplace distribution (right)

$$f(x,\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



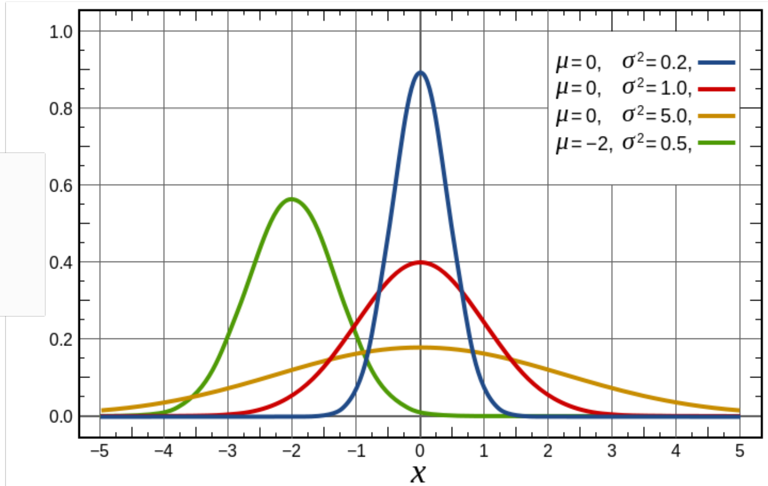
$$f(x|\mu,b) = \frac{1}{2b} \exp\left(-\frac{|x-\mu|}{b}\right)$$



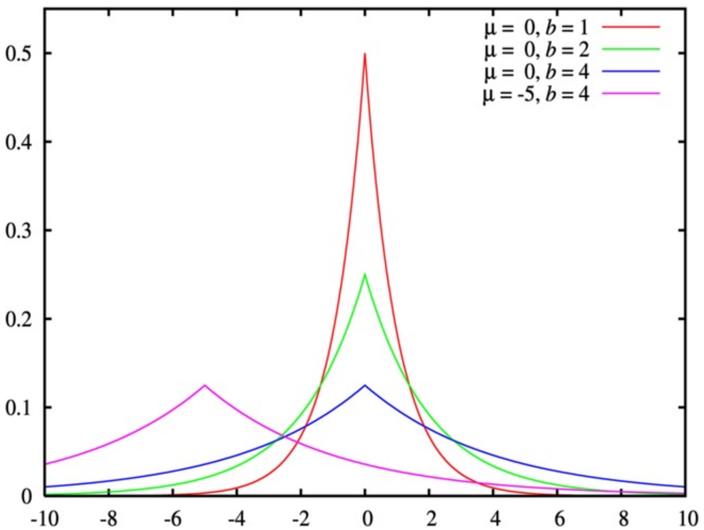
Regularization: limiting the search space

- The regularization constant in Ridge: $\lambda = \sigma^{-2}$
- The regularization constant in Lasso: $\lambda = b^{-1}$
- *The larger λ, the higher probability that x occurs around 0.*

$$f(x,\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



$$f(x|\mu,b) = \frac{1}{2b}\exp\left(-\frac{|x-\mu|}{b}\right)$$



Regularization: limiting the search space

- The regularized problem:

$$w^* = \arg\min_{w \in \boldsymbol{W}} L(f(x, w), \boldsymbol{D}) + \lambda g(w) \qquad (2)$$

- A result from the optimization literature shows that (2) is equivalent to the following:

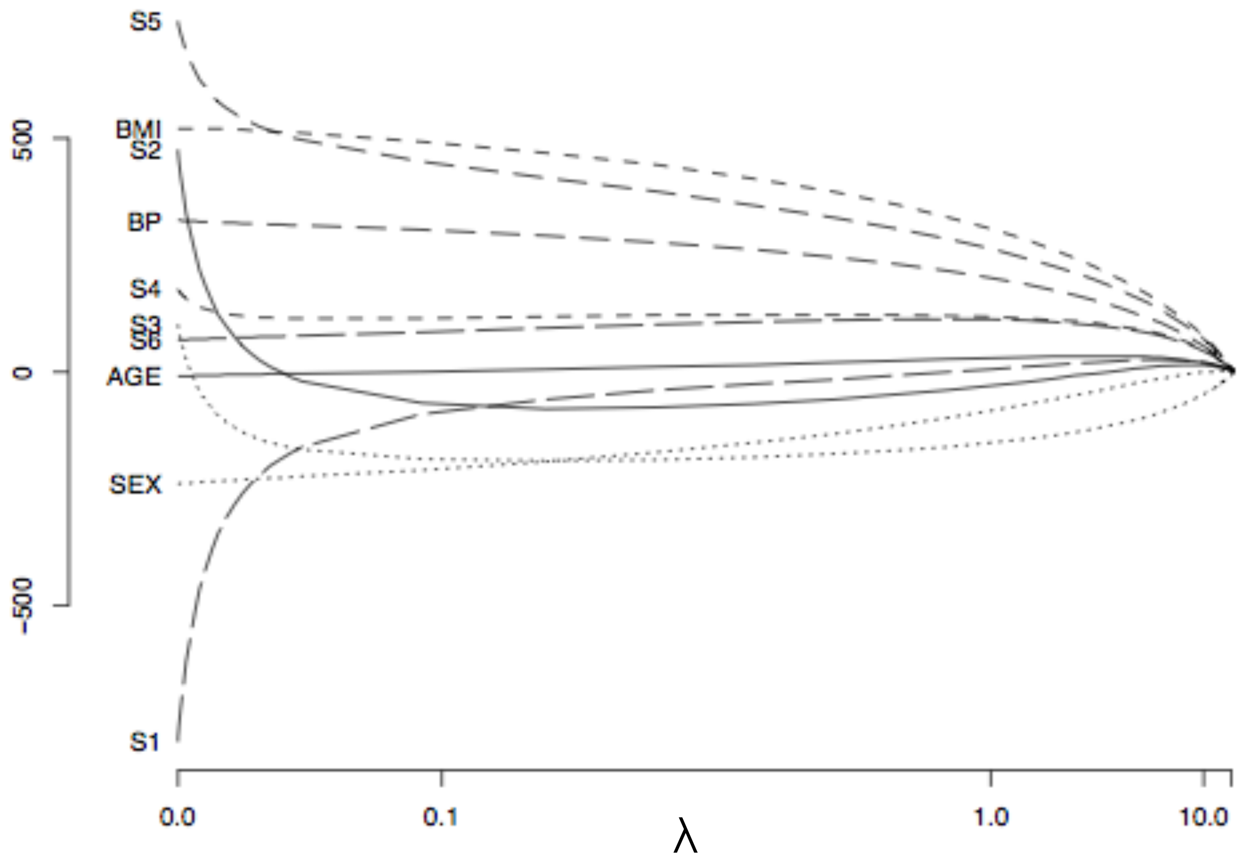
$$w^* = \arg\min_{w \in \boldsymbol{W}} L(f(x, w), \boldsymbol{D}) \quad \text{such that} \quad g(w) \leq s \qquad (3)$$

  - For some constant s.

- *Note that the constraint of g(w) ≤ s plays the role as limiting the search space of w.*

Regularization: effects of λ

- Vector **w*** = (w_0, s1, s2, s3, s4, s5, s6, Age, Sex, BMI, BP) changes when λ changes in Ridge regression.
  - **w*** goes to 0 as λ increases.



Regularization: practical effectiveness

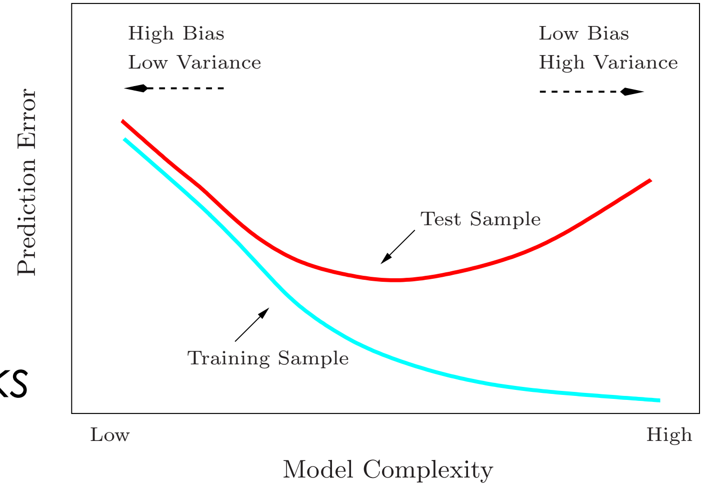
- Ridge regression was under investigation on a prostate dataset with 67 observations.
  - Performance was measured by RMSE (root mean square errors) and Correlation coefficient.

λ	0.1	1	10	100	1000	10000
RMSE	**0.74**	**0.74**	**0.74**	0.84	1.08	1.16
Correlation coeficient	0.77	0.77	**0.78**	0.76	0.74	0.73

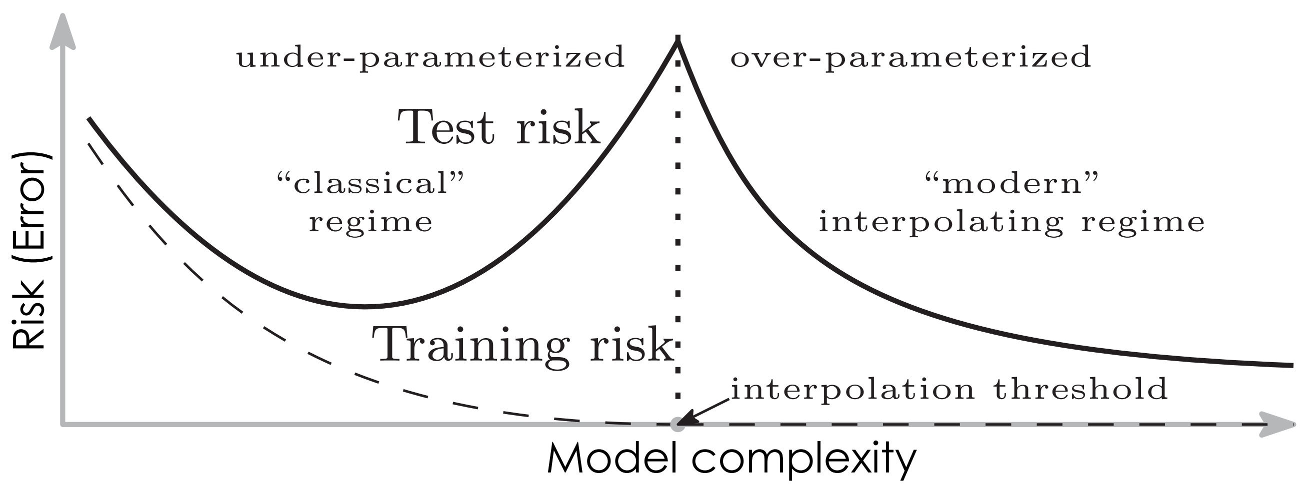
  - Too high or too low values of λ often result in bad predictions.
  - Why??

Bias-Variance tradeoff: revisit

- Classical view:
  More complex model $\hat{f}(x; \boldsymbol{D})$
  - Lower bias, higher variance



- Modern phenomenon:
  - *Very rich models such as neural networks are trained to **exactly fit** the data, but often obtain **high accuracy** on test data* [Belkin et al., 2019; Zhang et al., 2021]
  - $Bias \cong 0$
  - GPT-3, ResNets, VGG, StyleGAN, DALLE-3, …
- **Why???**



Regularization: summary

- Advantages:
  - Avoid overfitting.
  - Limit the search space of the function to be learned.
  - Reduce bad effects from noises or errors in observations.
  - Might model data better. As an example, L_1 often work well with data/model which are inherently sparse.
- Limitations:
  - Consume time to select a good regularization constant.
  - Might pose some difficulties to design an efficient algorithm.

References

- Belkin, M., Hsu, D., Ma, S., & Mandal, S. (2019). Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, *116*(32), 15849-15854.
- Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning* (pp. 448-456).
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, *25*, 1097-1105.
- Tibshirani, R (1996). *Regression shrinkage and selection via the Lasso*. Journal of the Royal Statistical Society, vol. 58(1), pp. 267-288.
- Trevor Hastie, Robert Tibshirani, Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2009.
- Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2021). Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, *64*(3), 107-115.