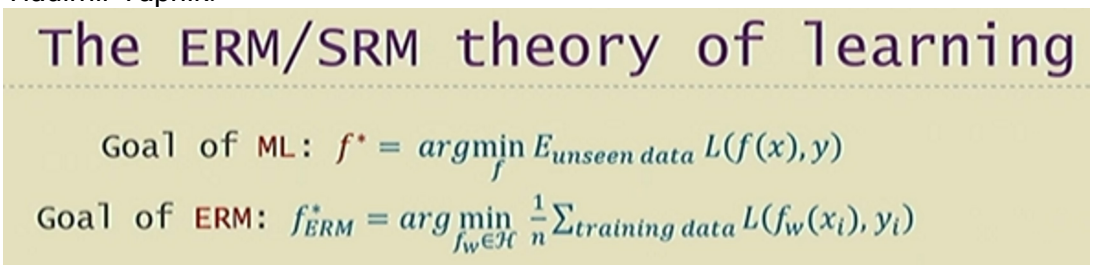
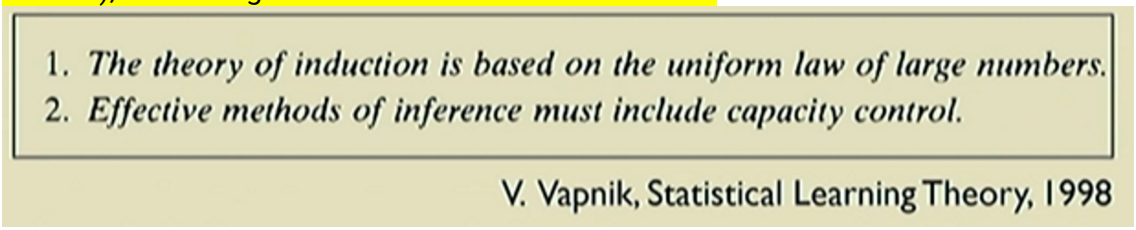
Why it's not overfitting

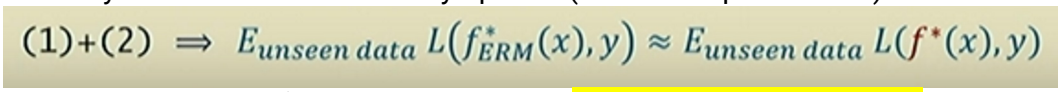
Thursday, March 26, 2020 2:54 PM

3/25
Why it's not overfitting? -- From Classical Statistics to Modern ML: the Lessons of Deep Learning - Mikhail Belkin, From Classical Statistics to Modern ML: the Lessons of Deep Learning, From Classical Statistics to Modern ML: the Lessons of Deep Learning,
https://www.youtube.com/watch?v=5-Kqb80h9rk&feature=youtu.be
OR https://www.youtube.com/watch?v=JS-Bl36aVPs&t=747s

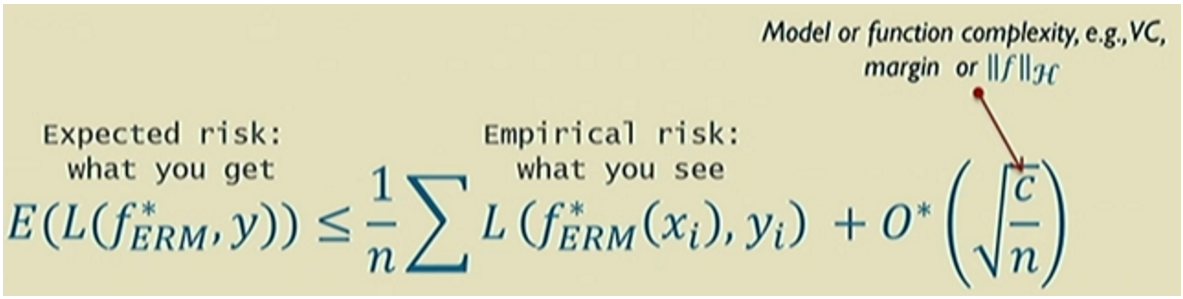
Mikhail (Misha) Belkin, https://scholar.google.com/citations?hl=en&user=Iwd9DdkAAAAJ&view_op=list_works&sortby=pubdate

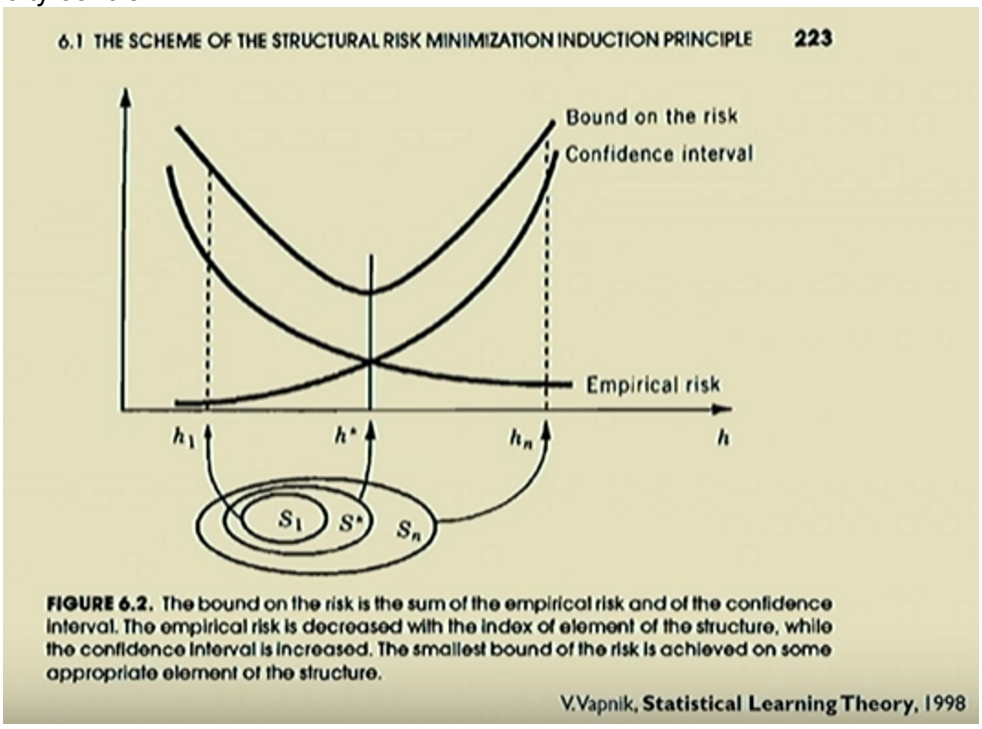
- VC dimension: A measure of the capacity of a space of function that can be learned by a statistical classification algorithm. It's defined as the cardinality of the largest set of points that the algorithm can shatter, which was originally defiend by Vladimir Vapnik.
- 

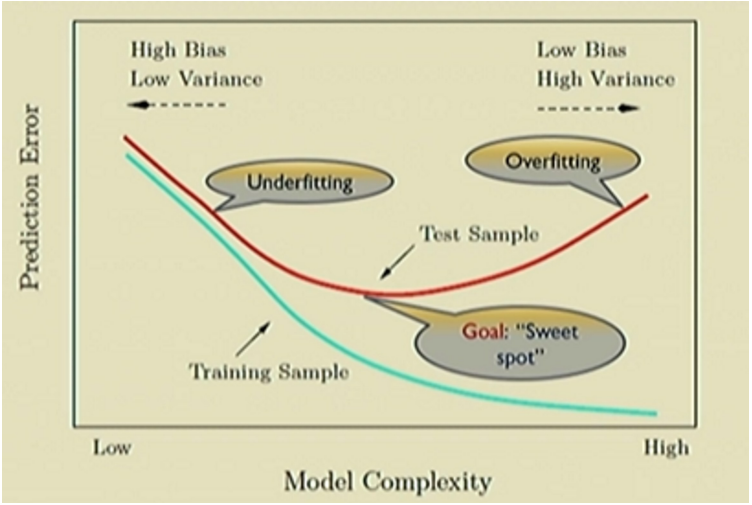
  - ML: the goal of Machine Learning is to find a function that minimize the loss in the future (Or the expected loss over prob dist under some assumption).
  - ERM: the goal of Empirical risk minimization is to find a fucntion over a class of function that minimize the loss over the training data
- How should we build a connection between Two?
  - According to Vapnik: 1) Uniform Law of Large number. 2)Capacity control
    i. **uniform Law of large number** says that the ERM is approximately equal to the expected loss, as the training sample set goes infinitely, so it allow us to use the the empirical loss function to approximate the expected loss function
    ii. **Capacity control:** say there are H (A set of function) contrains a funciton that approximate the goal of Machine Learning function (The target funct f*), so the target function must be with this set H.
    - 

  - You can see your ERM solution is nearly optimal (Close to expected loss)
    - 

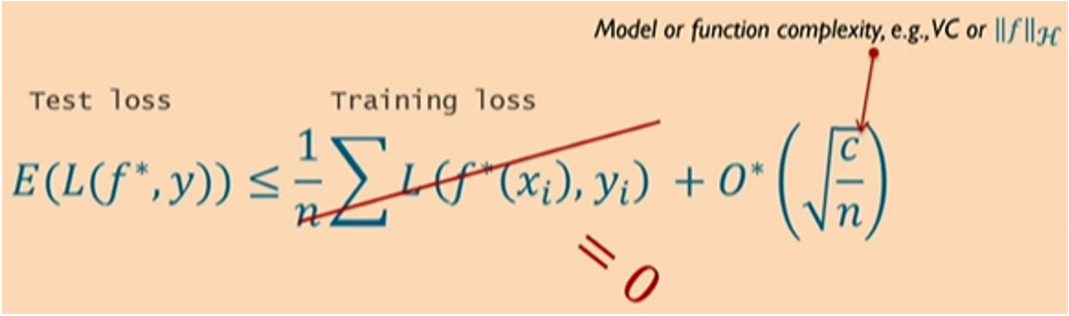
- Uniform laws of large numbers (Aka WYSIWG bounds, "what you see is what you get"), e.g. vc-dim, fat shattering, rademacher, covering number, margin…

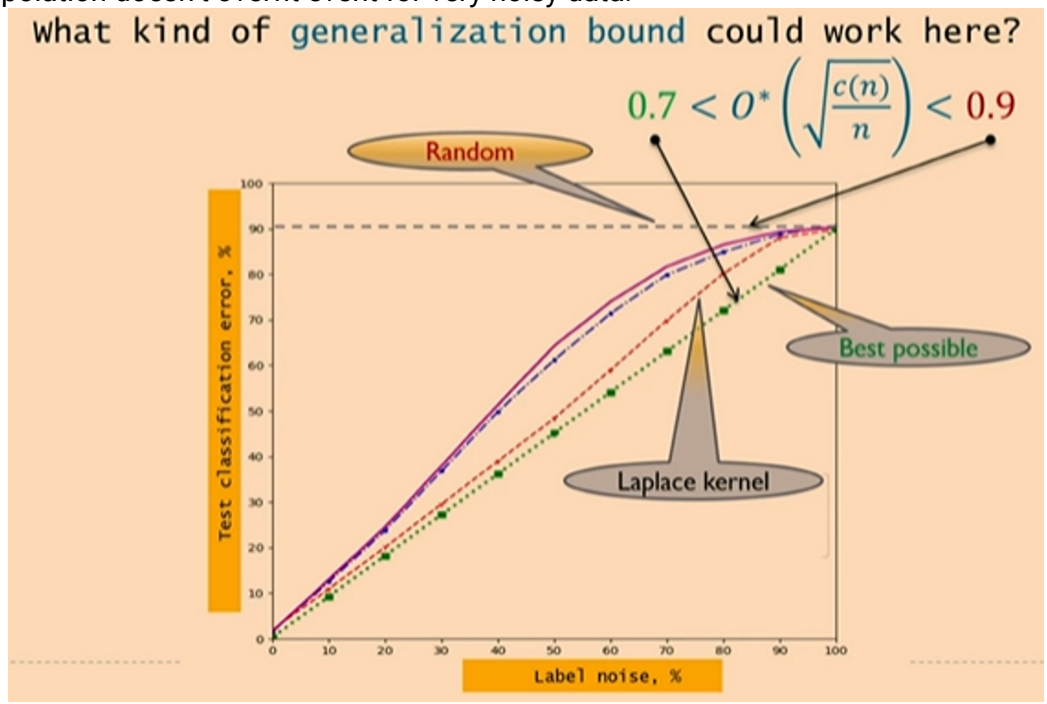
- 

  - In the left: the expected risk, (what you get, in the future), this loss never can goes zero, for the fact you have a lot of randomization in your test data.
  - In the right: 1)the Empirical risk (What you see while you are training the data) <== Defind by Law of large number, can possibly get to zero+ 2) capacity term (e.g. VC, margin)
  - ==> SO basically the "capacity control" defined our generalizatio bound ==> How well our model could possibly get trained.
- Capacity control
  - 

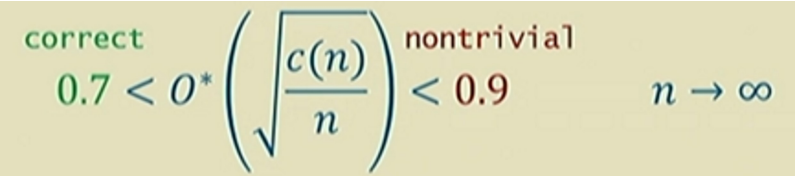
  - U-shaped generalization curve
    - 

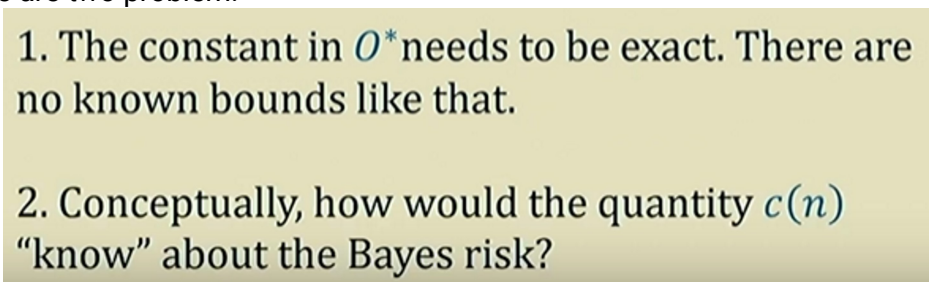
      - In the left side is underfitting, and right side is overfitting, and our

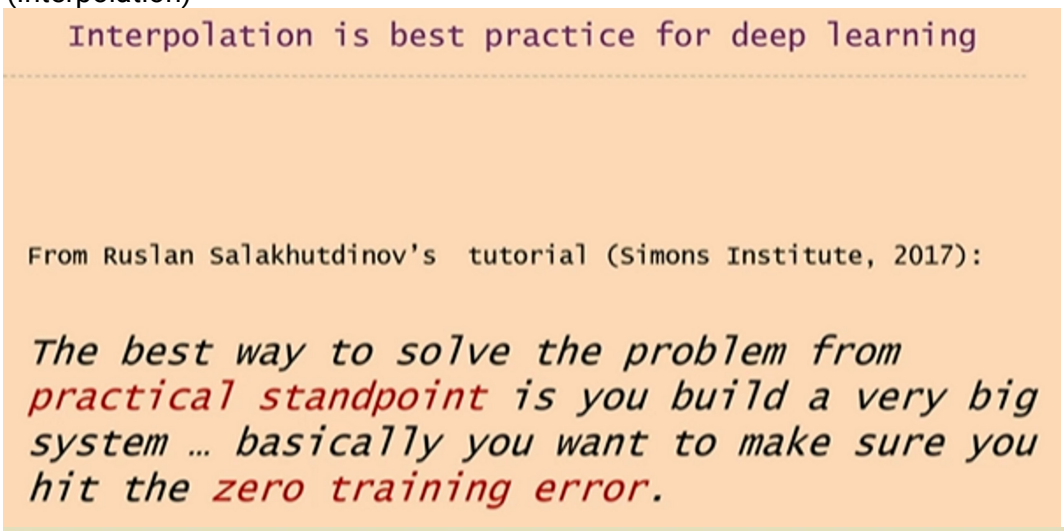
goal is to find this "sweet spot"
    - However, a model with **zero training error** is overfitting to the training data and will typically generalize poorly.
    - zero training error is also known as **Interpolation**, in mathmatical term!
- Does interpolation overfit?
    - There is a paper, prove that you can interpolation and still have a good test result!
    - 

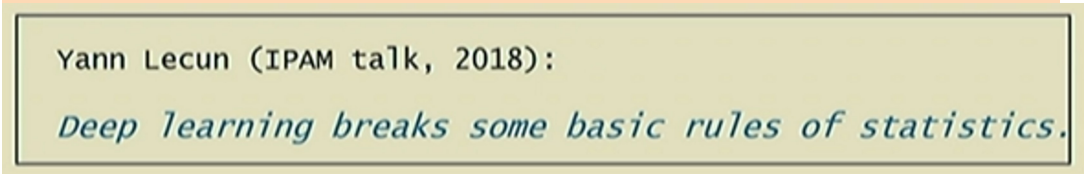
        - Review again:
            - In the left: the expected risk, (what you get, in the future), this loss never can goes zero, for the fact you have a lot of randomization in your test data.
            - In the right: 1)the Empirical risk (What you see while you are training the data) <== Defind by Law of large number, can possibly get to zero+ 2)capacity term (e.g. VC, margin)
            - ==> SO basically the "capacity control" defined our generalizatio bound ==> How well our model could possibly get trained.
- Interpolation doesn't overfit event for very noisy data:
    - 

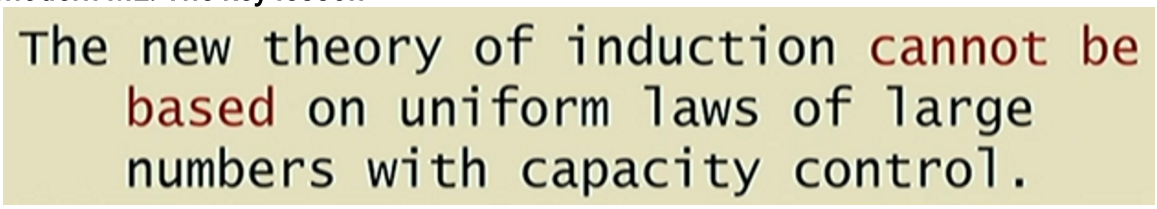
        - Randomization: take 10% of the data and assign random labels
        - Green line: 50%, the best you can possibly do, based on Bayes optimal classifier.
        - Random guess: is 90%, because the dataset has 10 classes.
        - Red line: a kernel machine train to have zero loss on the noisy data. However, the result doesn't seem to overfit.

- why bounds fail?
  - 

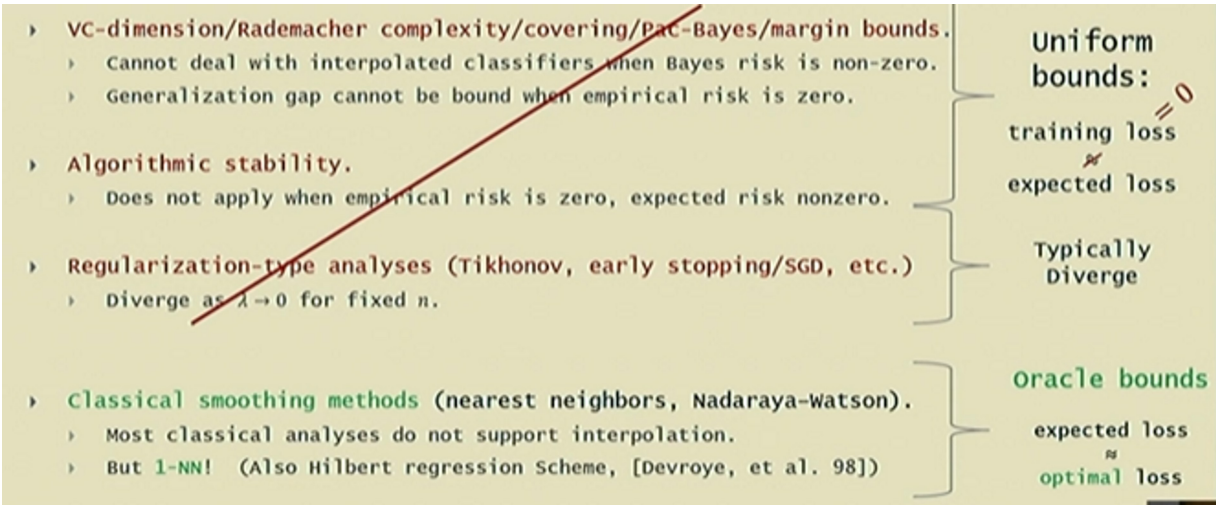
    - There are two problem:
      - 

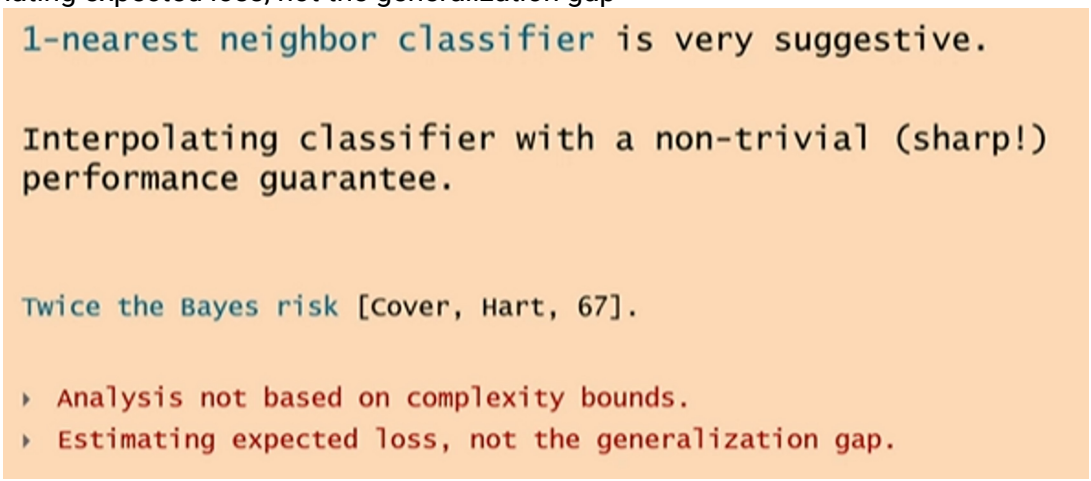
- Interpolation is best practice for deep learning:
  - The best way to solve the problem from practical standpoint is you build a very big system …. Basically you want to make sure you hit the zero training error (Interpolation)
    - 

    - 

- **The modern ML: The key lesson**
  - 

- So the generalization theory will be helpful for understanding interpolation?
  - What theoretical analyses do we have?

- 

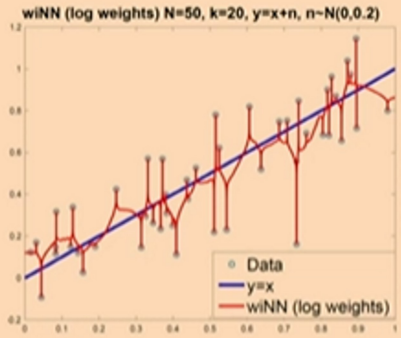
- 1-NN
  - Analysis doesn't based on complexity bounds
  - Estimating expected loss, not the generalization gap
    - 

- Could we do better than 1-NN? ==> Yes
  - Interpolated k-NN schemes, the data is randomly generated from a linear function plus some noise
  - The scheme, the red line model that we approximated, the more data you have the model will get better, and this result will get better in high dimenion. So, in this 1-D example you might feel it's terrible, but it's getting better in high dimension.

- $f(x) = \frac{\sum y_i k(x_i,x)}{\sum k(x_i,x)}$

  $k(x_i,x) = \frac{1}{||x-x_i||^\alpha}$, $k(x_i,x) = -\log||x-x_i||$

  (cf. Shepard's interpolation)

  - Theorem:

    Weighted (interpolated) k-nn schemes with certain singular kernels are consistent (converge to Bayes optimal) for classification in any dimension.

    Moreover, statistically (minimax) optimal for regression in any dimension.

    [B., Hsu, Mitra, NeuriPS 18] [B., Rakhlin, Tsybakov, AIStats 19]

  - Review
    - This talk so far:

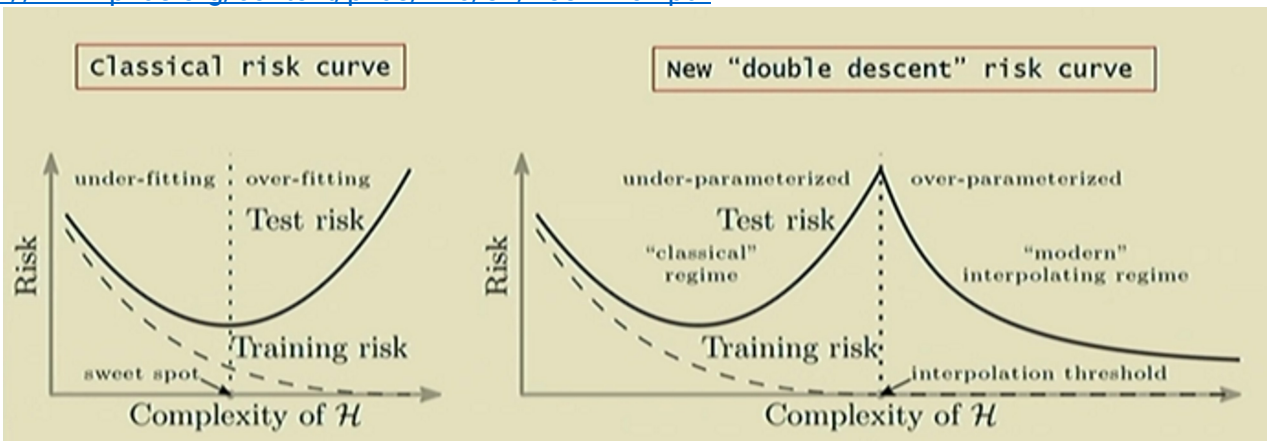
      A. Empirical effectiveness of interpolation.
      B. Theory of interpolation cannot be based on uniform bounds.
      C. Statistical validity of interpolating nearest neighbor methods.

      Yet, there is a mismatch between A and C.

      Methods we analyze are different from those used in practice.

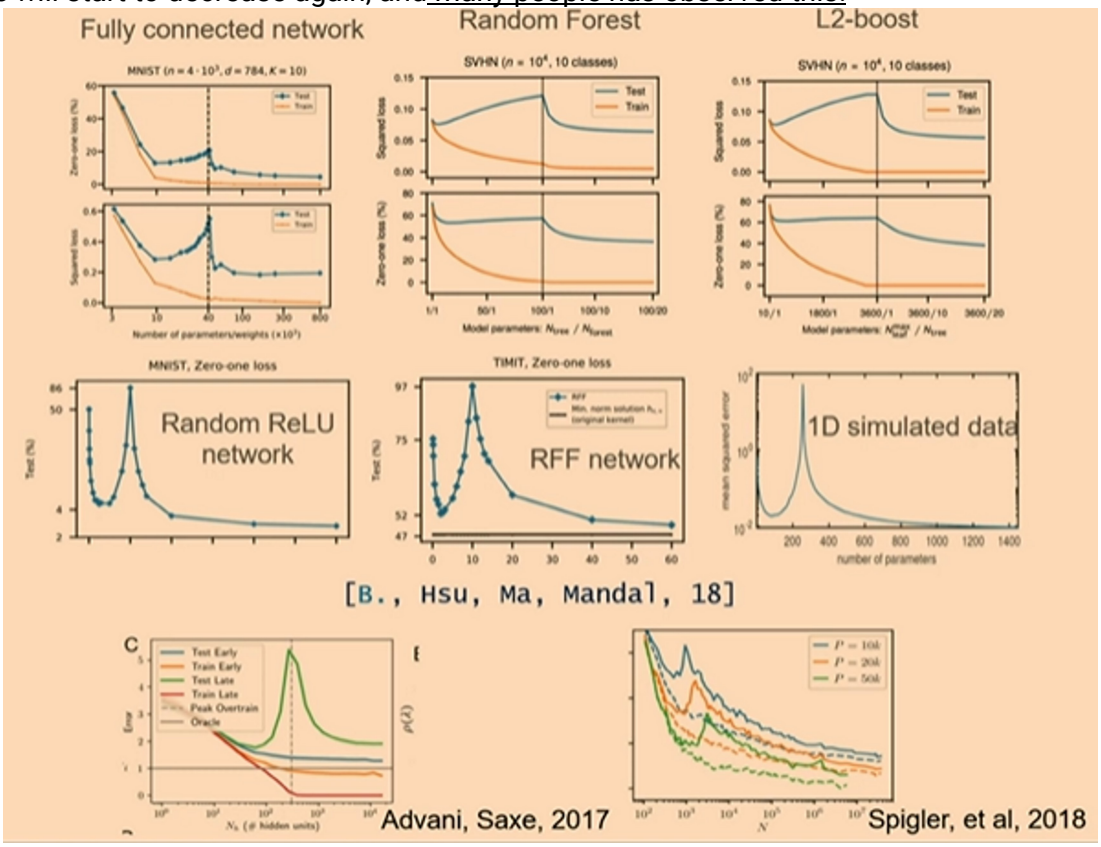
    - The New question we want to consider now:
      1. Dependence of generalization on model complexity?
      2. What is the role of optimization?

- "Double descent" risk curve: Reconciling modern machine-learning practice and the classical bias–variance trade-off, https://www.pnas.org/content/pnas/116/32/15849.full.pdf

  - Complexity of H: can be thinks as the number of parameters
  - What is the training end in Classical curve: It ends when our training get to error, or below certain tolerance
  - The complexity of model can be choose arbitraily, you can just add more and more neurons, and can continuely grow the model, to have more

complicated model. As the model become more complicated, the zag-zig line will start to converge to something that is smooth, and getting better!

- It's true that, if we keep growing the complexity of model, the test risk curve will start to decrease again, and many people has observed this!
  - 

- Let's getting more granularity:
  - Looking at the Random Fourier network, which also decribed in following paper. The result that we get is as the h_n(x) gets to infinitely, as you increase the number of features, this actually converged to certain kernel machine, which is some sort of functional minimum-norm solution!

- Random Fourier networks

  Random Fourier Features networks [Rahimi, Recht, NIPS 2007]

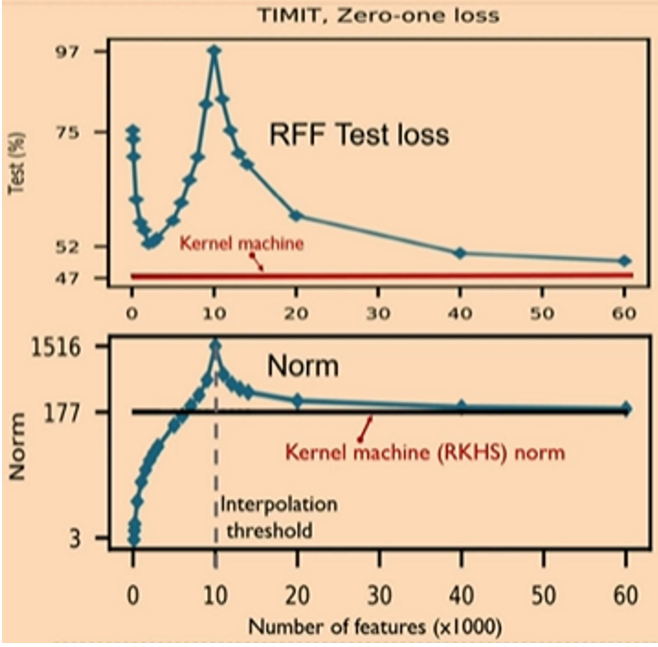
  $$h_{n,N}(x) = \sum_{j=1}^{N} \alpha_j \, e^{i\pi\langle w_j, x\rangle}$$

  Neural network with one hidden layer, *cos* non-linearity, fixed first layer weights. Hidden layer of size N. Data size n.

  Key property:

  $$\lim_{N\to\infty} h_{n,N}(x) = \text{kernel machine}$$

  - Let look at this more closely:
    - So more feature is better approximation to minimum norm solution:
  - What is the mechanism?



$N \to \infty$ -- infinite neural net

=

kernel machine

- ERM and Interpolation:
  - We choose a baseline function, and try to minimize the norm over the subspace of function, which fits the constraints exactly! However, we never actually do this explicitly, at least not at neural networks. The minimization of the norm is hidden somehow, within the dynamics of SGD.

- Classical ERM:

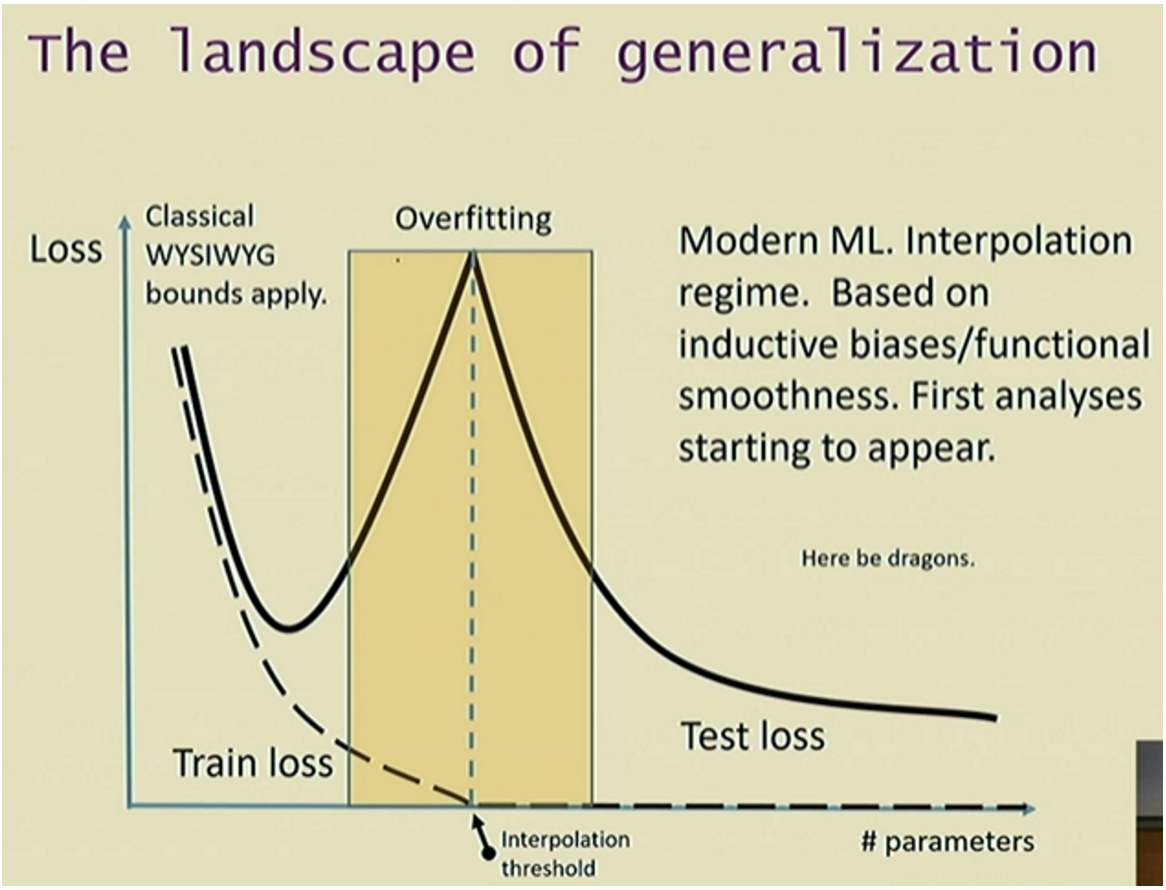
$$f^*_{ERM} = arg\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{training\ data} L(f(x_i), y_i)$$

- Modern ML/interpolation:

$$f^*_{int} = arg \min_{\substack{f \in \mathcal{H} \\ \forall_i f(x_i) = y_i}} ||f||$$

(Norm hidden within the dynamics of SGD)

- Framework for modern ML
  - Think of form of Occam's razor: which is based on inductive bias, Maximum smoothness subject to interpolating(or exactly fitting) the data!
  - There are three way to increase the smoothness:
    1. Explicitly: minimum functional norm solution
       1) Exact: Kernel machine
       2) Approximate: RFF, ReLU features
    2. Implicit: SGD/opimization (Neural Networks)
    3. Averaging (Bagging, L2-Boost)
- [35:08]New understanding of overfitting
  - Previous: while the training loss is low, we must be overfitting, and therefore we should decrease the number of parameter, such introduce regularization,
  - Now: there are two way to avoid the overfitting. 1)The classical way is to reduce the number of parameter. 2)The modern ML is to increase the number of parameter, moving to the right. It's counterintuitive, but

- 

- Classical Optimization(Under-parametrized)
  - Many local minima
  - SGD(Fixed step size) doesn't converge
- Modern Optimization(Interpolation/over-parametrized)
  - 1. Every local minimum is global (for networks wide enough) [Li, Ding, Sun, 18], [Yu, Chen, 95]

    2. Local methods converge to global optima [Kawaguchi, 16] [Soheil, et al, 16] [Bartlett, et al, 17] [Soltanolkotabi, et al, 17, 18] [Du, et al, 19] …

    3. Small batch SGD (fixed step size) converges as fast as GD per iteration. [Ma, Bassily, B., ICML 18] [Bassily, Ma, B., 18]