Midterm on 3/14 (Monday after Spring break)
Can opt for take-home

ECE 0402 - Pattern Recognition

Lecture 13

Model selection:

In statistical learning, a **model** is a mathematical representation of a function such as a

- classifier
- regression function
- density
- clustering

(under our control to decide)

So far, we have one (or more) "free parameters" that are not automatically determined by the learning algorithm. And often, the value chosen for these free parameters has a significant impact on the algorithm's output.

The problem of selecting values for these parameters is called model selection.

Examples:

Method	**Parameter**
• polynomial regression	• polynomial degree d
• ridge regression	• regularization parameter λ
• SVMs (MMC, SMC)	• margin violation cost C (slack variables) kernel parameters C try to separate the data vs being robust to outliers
• neural networks	• regularization parameter λ stopping criteria
• k-nearest neighbors—take one point find its k-nearest neighbors and take a vote	• number of clusters k

We want to figure out automatic ways of setting up these parameters.

- We need to select appropriate values for the free parameters
- All we have is the training data
- why are these parameters here to start with?

Cross Validation ☺

- These parameters usually control the balance between **underfitting** and **overfitting**
- Here is a **dilemma**: they were left "free" precisely because we don't want to let the training data influence their selection, as this always leads to overfitting.
  - for example in regression, if we let the training data determine the degree, we will just end up choosing largest degree and end up doing interpolation...

Up until now, we have focused on trying to judge if we are learning via decomposition of the forms

$$\text{true risk} = \text{empirical risk} + \text{excess risk}$$
$$R(h) = \hat{R}(h) + \text{excess risk}$$

We looked at controlling excess risk using VC dimension and regularization. We also looked at:

$$\text{true risk} = \text{bias} + \text{variance}$$

Now we will consider more practical approach: **Validation**.

Almost whole semester we were trying to understand risk, right we said maybe we can break it up into other things that we can kind of control/or analyze...in one way or another...but why didn't we just try a little bit harder to directly estimate what this risk was?

After we select h, why don't we try to estimate

$$R(h) = \mathbb{E}[e(h(X), Y)]$$

directly from data–somehow? Here, "e" is just some function that measure the difference between $h(X)$ and Y. I am being intentionally vague here because we covered both regression and classification–so if we are talking about classification this could be the "probability of error", and it could be the "squared error" if we are talking about regression. This is kind of the reason we introduced Risk at the first place, so we can talk about all these things in one-word.

Okay, no theory here, once we pick h can we somehow estimate its performance?– at least have some prediction what the risk is gonna be...

Validation

Suppose we have this another dataset $(x_1, y_1), ..., (x_k, y_k)$, in addition to our training data

- k additional input-output pairs to our training dataset.

We can use the validation set to form an estimate $\hat{R}_{val}(h)$. A natural way to define this estimate would be:

$$\hat{R}_{val}(h) := \frac{1}{k}\sum_{i=1}^{k} e(h(x_i), y_i)$$

- For classification:

error function (indicator function)

$$\hat{R}_{val}(h) = \frac{1}{k}\sum_{i=1} 1_{h(x_i)\neq y_i}$$

- For regression:

$$\hat{R}_{val}(h) = \frac{1}{k}\sum_{i=1}(h(x_i) - y_i)^2$$

$$\hat{R}_{val}(h) = \frac{1}{k}\sum_{i=1}|h(x_i) - y_i|$$

whatever error function you pick this is easy to estimate

Accuracy of validation

What can we say about the accuracy of $\hat{R}_{val}(h)$?

In the case of classification, $e(h(x_i), y_i) = 1_{h(x_i\neq y_i}(i)$ which is just a Bernoulli random variable

$$\text{Hoeffding}: \ \mathbb{P}[|\hat{R}_{val}(h) - R(h)|] \leq 2e^{-2\epsilon^2 k}$$

decays exponentially fast as k grows – no surprises here. The same story hold for any kind of error metric, the little e function here...More generally, we always have whats the expected value of these RVs – empirical estimates

$$\mathbb{E}[\hat{R}_{val}(h)] = \frac{1}{k}\sum_{i=1}^{k}\mathbb{E}[e(h(x_i), y_i)] = R(h)$$

$$\text{var}[\hat{R}_{val}(h)] = \frac{1}{k^2}\sum_{i=1}^{k}\mathbb{E}var[e(h(x_i), y_i)] = \frac{\sigma^2}{k}$$

so we do it k times, the variance goes down. I am not saying anything sophisticated here, I am just saying if you are trying to estimate something – here you are trying to estimate this mean by taking bunch of things that are randomly distributed as $R(h)$ is its mean, and averaging them, then this is an unbiased estimate of the thing we are trying to get – and the variance of this estimator goes down as you add more samples...

In either case, this shows us that

$$\hat{R}_{val}(h) = R(h) \pm O(\frac{1}{\sqrt{k}}) \qquad :O$$

Thus, we can get as accurate an estimate of $R(h)$ using validation set as long as k is large enough. But **"where is this validation set coming from?"**

Remember h is ultimately something we learned from the training data.

This is completely artificial distinction validation and training set–wiseElf

Validation vs training

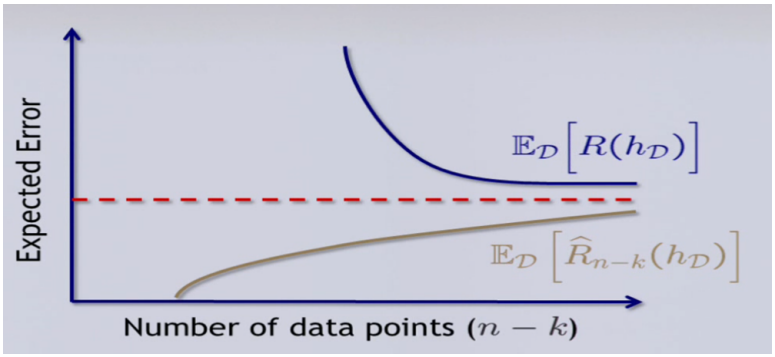
You have to decide and split the data $(x_1, y_1), ..., (x_n, y_n)$ into two sets:

- validation (holdout) set : $(x_1, y_1), ..., (x_k, y_k)$
- training set: $(x_{k+1}, y_{k+1}), ..., (x_n, y_n)$

Validation error is $O(1/\sqrt{k})$:

$$\begin{aligned} \text{Small } k &\implies \text{bad estimate} \\ \text{Large } k &\implies \text{accurate estimate, but of what?} \end{aligned}$$

Large k let's us say that we are very confident that we have selected a terrible h.



What if we decide to be greedy? After using the validation set to estimate the error, re-train on the whole data set.

We use $(x_1, y_1), ..., (x_n, y_n) \implies \hat{h}$:

- training set: $(n-k) \implies h'$
- validation (holdout) set : $k \implies \hat{R}_{val}(h')$

$$\begin{aligned} \text{Small } k &\implies \text{bad estimate of } R(h'), \text{but } R(h') \approx R(\hat{h}) \\ \text{Large } k &\implies \text{accurate estimate of } R(h'), \text{but } R(h') \gg R(\hat{h}) \end{aligned}$$

$\hat{R}_{val}(h')$ is a rough (independent) estimate of how good of a job $\hat{h}$ is doing!

Rule of thumb: Set $k = n/5 \implies$ 20 % of your data

Validation vs testing

We call this "validation", but how is it any different than simply "testing"?

There is a saddle difference! you had this in the first homework, it was a heart disease data (last problem) where you had a test data and training data separate. That might seem like about the same idea, but there is a saddle difference between that and this idea of validation.

Mainly, the whole reason of doing this validation estimates $\hat{R}_{val}$ is not just to know how well we are doing, but actually to inform our algorithm, helps us make choices as part of the learning – we are going to end up using this validation estimates to help us "how to set λ"...and all that "model selection" choices–fixed parameters in our algorithms.

Typically, $\hat{R}_{val}$ is used to make learning choices.

If an estimate of $R(h)$ affects learning, i.e., it impacts which h we choose, than it is no longer a **test** set. It becomes a **validation** set.

The difference:

- a test set is **unbiased**

  this tells us how well we are doing but the moment that we let that leak back into what classification algorithm we use, or how we're designing our h then this becomes a validation set and the estimates it produces will be overly optimistic

- a validation set will have (overly) optimistic **bias**

  optimism in general is not bad, but you don't wanna be overly optimistic because then you might think you are doing much better than you actually are

Example: Suppose we have 2 hypothesis: $h1, h2$ and that they have equal risk (true risk)

$$R(h_1) = R(h_2) = p$$

Of course, we don't get the observe this risk but, we have some validation data that we calculate validation-error. Next suppose that our error estimates for these two hypothesis, denoted by $\hat{R}_{val}(h_1)$ and $\hat{R}_{val}(h_2)$, are distributed according to

$$\hat{R}_{val}(h_i) \sim U[p - \eta, p + \eta]$$

If this is true for both h, we can use these to help us decide which one of these we should use.

Pick $h \in \{h_1, h_2\}$ that minimizes $\hat{R}_{val}(h)$.

- Then we have a problem, we will pick the one that looks better. But then it is easy to argue the empirical risk of the one you picked-on average is going to be less than p. If it was an unbiased estimate, it would be equal to p, but it is usually less, and in expectation definitely less.

One way to see this: If I have 2 uniform random variables, this expectation will be the minimum of these two uniform random RVs–minimum of 2 uniform RVs is not just a uniform random variable over that same range–if you write down the CDF and worked out what it is, you will see.

What is the probability that both $\hat{R}_{val}(h_1), \hat{R}(h_1)$ are bigger than p. This is only going to happen 1-quarter of the time, 75-percent of the time one-or-both of these will actually be smaller than p. So if 75-percent time you are lower than p, it would be weird that expected value was not less than p. It is easy to argue that $\mathbb{E}[\hat{R}_{val}(h)] < p$ optimistic bias (75% of the time, $min(\hat{R}_{val}(h_1), \hat{R}_{val}(h_2)) < p$

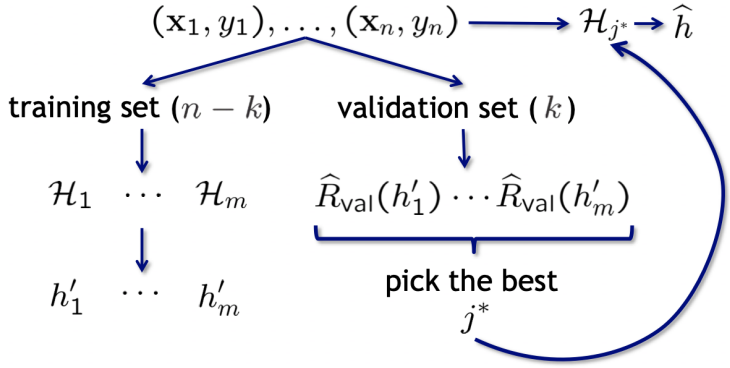
So this is the problem, we are specifically picking h_1 and h_2 based on whether or not they gave us a smaller estimate on our validation error, we are specifically making h depend on "which one of these gives us a smallest value" here, this is gonna become "optimistically biased" estimate of "how well we are performing".

This is all similar idea, what we have been talking about from the beginning of the term, if we look at the data, we could overfit to it. Training error is not super meaningful when we pick our hypothesis with respect to it.

But we could still use it to do model selection.

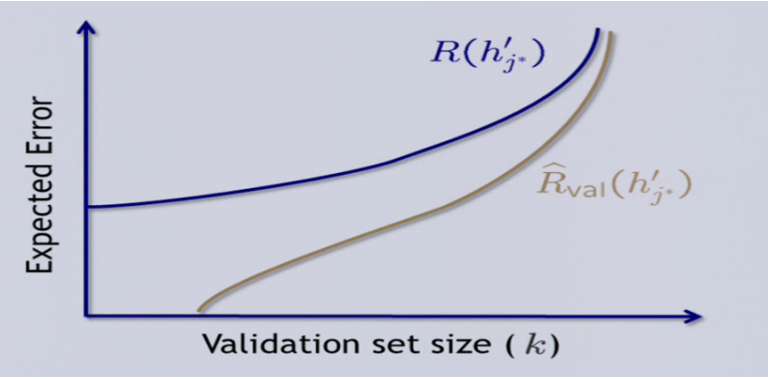
Using validation for model selection

Suppose we have m models: $\mathcal{H}_1, ..., \mathcal{H}_m$



We can't use $\hat{R}_{val}$ to tell how well $\hat{h}$ will perform in practice...biased as we just discussed.."optimistically biased"–usually people just ignore that issue–I know it is not telling me how well h-hat is working, I am just using this to pick one hypothesis..but if you wanna know how well it is actually working and have some unbiased estimate of how well $\hat{h}$is doing, you will need some whole data set separate from all of these...

We select the model $\mathcal{H}_{j^*}$ using validation set. $\hat{R}_{val}(h'_{j*})$ is a biased estimate of $R(h'_{j^*})$ (and $R(h_{j^*})$).



The bias

the curve is going up because as k grows we have smaller amounts of data to work with... We have seen stuff like this before...For m models $\mathcal{H}_1, ..., \mathcal{H}_m$, we use a data set size of k to pick the model that best out of $\{h'_1, ..., h'_m\}$. Back to Hoeffding!

$$R_{val}(h'_{j*}) \leq \hat{R}_{val}(h'_{j*}) + O(\sqrt{\frac{log\ m}{k}})$$

validation error will always going to be a good estimate of the true risk as long as size of validation set is big enough–how big does it need to be?–it needs to be bigger than log of

the number of models we are considering... Or, if the $\mathcal{H}_j$ correspond to a few continuous parameters we can use the VC approach to argue.

So, k, the size of the validation dataset has to be big compared to the m – all of that VC analysis continues to be useful here

if you have fix m, k needs to be bigger than logarithm of that, in order this to be small

$$R_{val}(h'_{j*}) \leq \hat{R}_{val}(h'_{j*}) + O(\sqrt{\frac{\#\text{ of parameters}}{k}})$$

We have now discussed three different kinds of estimates of risk $R(h) : \hat{R}_{train}(h), \hat{R}_{test}(h), \hat{R}_{val}(h)$. These three estimates have different degrees of "contamination" that manifests itself as a (deceptively) optimistic bias. They are all trying to give us an estimate of true risk:

- training set: totally contaminated
- test set: totally clean
- validation set: slightly contaminated

Validation dilemma

We would like to argue

$$\begin{array}{ccc} R(h) \approx R(h') & \approx & \hat{R}_{val}(h') \\ \uparrow & & \uparrow \\ \textbf{small } k & & \textbf{large } k \end{array}$$

All we need to do is to make k simultaneously very small and very large :)

Can we do this? Yes! this is actually rare things you can actually do and not pay a penalty!

Cross-validation

Leave one out: Let's make k small, so let's set $k = 1$!

$$\mathcal{D}_j = (x_1, y_1), ..., \cancel{(x_j, y_j)}, ..., (x_n, y_n)$$

Select a hypothesis h'_j using the data set $\mathcal{D}_j$.

Validation error $\hat{R}_{val}(h'_j) = e(h'_j(x_j), y_j) := e_j$

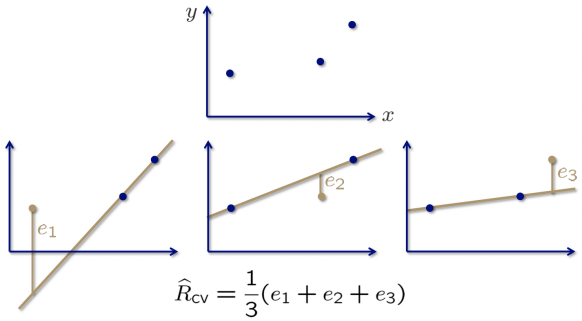
We set k to be too small, so this is a terrible estimate

Repeat this for all possible choices of j and average! Go through his process n times by leaving each one of these our one time.

$$\hat{R}_{CV} = \frac{1}{n}\sum_{i=1}^{n} e_j$$

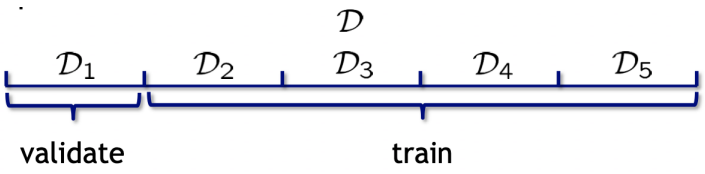
this is called the **leave one our cross validation estimate**

Example: Suppose we are fitting a line to 3 data points.



The downside here is if you have a lot of points, this becomes computationally demanding. Because you have to train your classifier that many times.

k-fold cross validation: Train k times on $n - \frac{n}{k}$ points each.



5-fold

Common choices are $k = 5, 10$. This is one of the work-horses of practical machine learning. there is always parameters, and you don't know how to fit them, this is the principal and robust way to do it.

Notice k changed meaning in the last part

Next time: bootstrap ...