Validation

Idea: once you pick an hypothesis, use new data to estimate its true error

Assume we have picked a predictor h (e.g., by ERM rule on a $\mathcal{H}_d$).

Let $V = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{m_v}, y_{m_v})$ be a set of m_v *fresh* samples from $\mathcal{D}$ and let $L_V(h) = \frac{1}{m_V} \sum_{i=1}^{m_V} \ell(h, (\mathbf{x}_i, y_i))$

Assume the loss function is in $[0, 1]$. Then by Hoeffding inequality we have the following.

Proposition

For every $\delta \in (0, 1)$, with probability $\geq 1 - \delta$ (over the choice of V) we have

$$|L_V(h) - L_{\mathcal{D}}(h)| \leq \sqrt{\frac{\log(2/\delta)}{2m_v}}$$

Note: possible only because we use *fresh* (new) samples...

In practice:

- we have only 1 dataset
- we split it into 2 parts:
  - training set
  - *hold out* or *validation* set

A similar approach can be used for model selection, i.e. to pick one hypothesis (or class of hypothesis, or value of a parameter) among hypothesis in several classes...

Validation for Model Selection

Assume we have $\mathcal{H} = \cup_{i=1}^{r} \mathcal{H}_i$

Given a training set S, let h_i be the hypothesis obtained by ERM rule from $\mathcal{H}_i$ using S
$\Rightarrow$ how do we pick a final hypothesis from $\{h_1, h_2, \ldots, h_r\}$?

Validation set: $V = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{m_v}, y_{m_v})$ be a set of *fresh* m_v samples from $\mathcal{D}$

$\Rightarrow$ choose final hypothesis (or class or value of the parameter) from $\{h_1, h_2, \ldots, h_r\}$ by ERM over validation set

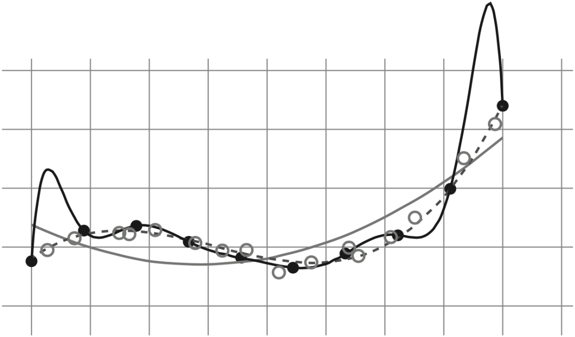
Assume loss function is in $[0, 1]$. Then we have the following.

Proposition

With probability $\geq 1 - \delta$ over the choice of V we have

$$\forall h \in \{h_1, \ldots, h_r\} : |L_{\mathcal{D}}(h) - L_V(h)| \leq \sqrt{\frac{\log(2r/\delta)}{2m_v}}$$

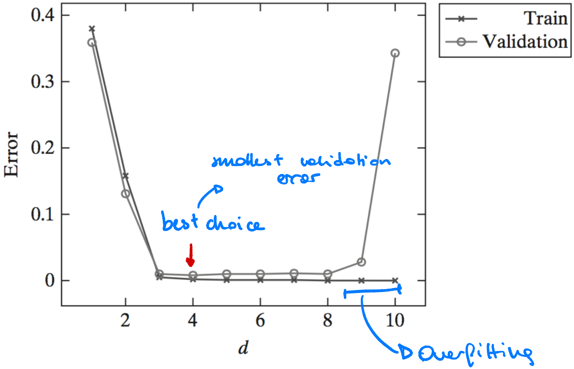
Example



Model-Selection Curve

Shows the training error and validation error as a function of the complexity of the model considered

Example



Training error decreases but validation error increases ⇒ *overfitting*

What if we have one or more parameters with values in $\mathbb{R}$?

1. Start with a rough *grid* of values
2. Plot the corresponding model-selection curve
3. Based on the curve, zoom in to the correct *regime*
4. Restart from 1) with a finer grid

Note: the empirical risk on the validation set *is not* an estimate of the true risk, in particular if r is large (i.e., we choose among many models)!

Question: how can we estimate the true risk after model selection?

Train-Validation-Test Split

Assume we have $\mathcal{H} = \cup_{i=1}^{r} \mathcal{H}_i$

Idea: instead of splitting data in 2 parts, divide into 3 parts

1. *training set*: used to learn the best model h_i from each $\mathcal{H}_i$
2. *validation set*: used to pick one hypothesis h from $\{h_1, h_2, \ldots, h_r\}$
3. *test set*: used to estimate the true risk $L_{\mathcal{D}}(h)$

⇒ the estimate from the test set has the guarantees provided by the proposition on estimate of $L_{\mathcal{D}}(h)$ for 1 class

Note:

- If you use validation to pick the value of a parameter: you learn the best model for the given value using train + valid. set

- the test set *is not involved* in the choice of h
- if after using the test set to estimate $L_{\mathcal{D}}(h)$ we decide to choose another hypothesis (*because we have seen the estimate of* $L_{\mathcal{D}}(h)$ *from the test set...*)

  ⇒ we cannot use the test set again to estimate $L_{\mathcal{D}}(h)$!

k-Fold Cross Validation

When data is not plentiful, we cannot afford to use a *fresh* validation set $\Rightarrow$ cross validation

$\Rightarrow$ k-fold cross validation:

1. partition (training) set into k *folds* of size m/k
2. for each fold:
   - train on union of other folds
   - estimate error (for learned hypothesis) from the fold
3. estimate of the true error = average of the estimated errors above

Leave-one-out cross validation: $k = m$

Often cross validation is used for model selection

- at the end, the final hypothesis is obtained from training on the entire training set

k-Fold Cross Validation for Model Selection

```
input:
    training set $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
    set of parameter values $\Theta$
    learning algorithm $A$
    integer $k$
partition $S$ into $S_1, S_2, \ldots, S_k$
foreach $\theta \in \Theta$
    for $i = 1 \ldots k$
        $h_{i,\theta} = A(S \setminus S_i; \theta)$
    $\text{error}(\theta) = \frac{1}{k} \sum_{i=1}^{k} L_{S_i}(h_{i,\theta})$
output
  $\theta^{\star} = \text{argmin}_{\theta} [\text{error}(\theta)]$
  $h_{\theta^{\star}} = A(S; \theta^{\star})$
```

What if learning fails?

You use training data S and validation to pick a model h_S...
everything looks good!
But then, on test set results are bad...

What can we do?

Need to understand where the error comes from!

Two cases:

- $L_S(h_s)$ is large
- $L_S(h_s)$ is small

$L_S(h_s)$ is large

Let $h^* \in \arg\min_{h \in \mathcal{H}} L_{\mathcal{D}}(h)$.

Note that:

$$L_S(h_S) = (L_S(h_S) - L_S(h^*)) + (L_S(h^*) - L_{\mathcal{D}}(h^*)) + L_{\mathcal{D}}(h^*)$$

and

- $L_S(h_S) - L_S(h^*) \leqslant 0$
- $L_S(h^*) \approx L_{\mathcal{D}}(h^*)$

Therefore:

$L_S(h_S)$ large $\Rightarrow$ $L_{\mathcal{D}}(h^*)$ is large $\Rightarrow$ approximation error is large

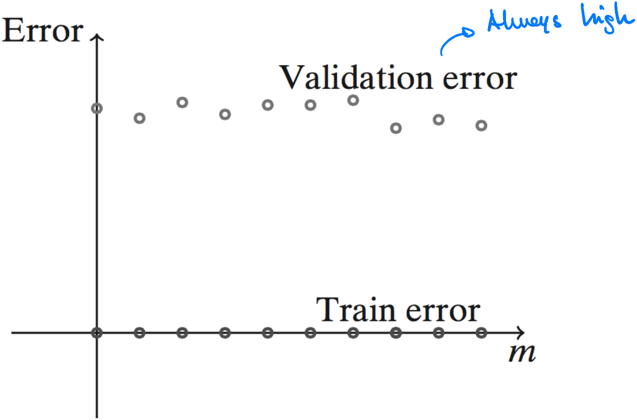
$L_S(h_S)$ is small

Need to understand if $L_{\mathcal{D}}(h^*)$ is large or not!

How?

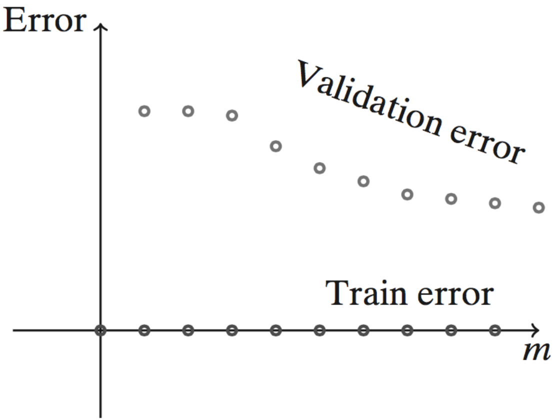
Learning curves: plot of training error and validation error when we run our algorithms on *prefixes of the data of increasing size* m

Case 1



⇒ There is no evidence that the approximation error of $\mathcal{H}$ is good (i.e., that is small)

Case 2



⇒ $\mathcal{H}$ may have a good approximation error but maybe we do not have enough data

Summarizing

Some potential steps to follow if learning fails:

- if you have parameters to tune, plot model-selection curve to make sure they are tuned appropriately
- if training error is excessively large consider:
  - enlarge $\mathcal{H}$
  - change $\mathcal{H}$
  - change feature representation of the data
- if training error is small, use learning curves to understand whether problem is approximation error (or estimation error)
  - if approximation error seems small:
    - get more data
    - reduce complexity of $\mathcal{H}$
  - if approximation error seems large:
    - change $\mathcal{H}$
    - change feature representation of the data

Machine Learning

Regularization and Feature Selection

Fabio Vandin November 13th, 2023

Learning Model

- A: learning algorithm for a machine learning task
- S: m i.i.d. pairs $z_i = (x_i, y_i), i = 1, \ldots, m$, with $z_i \in Z = \mathcal{X} \times Y$, generated from distribution $\mathcal{D}$
  $\Rightarrow$ training set available to A to produce $A(S)$;
- $\mathcal{H}$: the hypothesis (or model) set for A
- loss function: $\ell(h, (x, y))$, $\ell : \mathcal{H} \times Z \to \mathbb{R}^+$
- $L_S(h)$: empirical risk or training error of hypothesis $h \in \mathcal{H}$

$$L_S(h) = \frac{1}{m} \sum_{i=1}^{m} \ell(h, z_i)$$

- $L_{\mathcal{D}}(h)$: true risk or generalization error of hypothesis $h \in \mathcal{H}$:

$$L_{\mathcal{D}}(h) = \mathbb{E}_{z \in \mathcal{D}}[\ell(h, z)]$$

Learning Paradigms

We would like A to produce $A(S)$ such that $L_{\mathcal{D}}(A(S))$ is *small*, or at least close to the smallest generalization error $L_{\mathcal{D}}(h^*)$ achievable by the "best" hypothesis h^* in $\mathcal{H}$:

$$h^* = \arg\min_{h \in \mathcal{H}} L_{\mathcal{D}}(h)$$

We have seen a *learning paradigm*: Empirical Risk Minimization

We will now see another learning paradigm...

Regularized Loss Minimization

Assume h is defined by a vector $\mathbf{w} = (w_1, \dots, w_d)^T \in \mathbb{R}^d$ (e.g., linear models)

Regularization function $R : \mathbb{R}^d \to \mathbb{R}$

Regularized Loss Minimization (RLM): pick h obtained as

$$\arg\min_{\mathbf{w}} \left(L_S(\mathbf{w}) + R(\mathbf{w}) \right)$$

Intuition: $R(\mathbf{w})$ is a "measure of complexity" of hypothesis h defined by $\mathbf{w}$
$\Rightarrow$ regularization balances between low empirical risk and "less complex" hypotheses

We will see some of the most common regularization function

ℓ_1 Regularization

Regularization function: $R(\mathbf{w}) = \lambda||\mathbf{w}||_1$

- $\lambda \in \mathbb{R}, \lambda > 0$
- ℓ_1 norm: $||\mathbf{w}||_1 = \sum_{i=1}^{d} |w_i|$

Therefore the *learning rule* is: pick

$$A(S) = \arg\min_{\mathbf{w}} \left(L_S(\mathbf{w}) + \lambda||\mathbf{w}||_1 \right)$$

Intuition:

- $||\mathbf{w}||_1$ measures the "complexity" of hypothesis defined by $\mathbf{w}$
- λ regulates the tradeoff between the empirical risk ($L_S(\mathbf{w})$) or overfitting and the complexity ($||\mathbf{w}||_1$) of the model we pick

LASSO

Linear regression with squared loss + ℓ_1 regularization ⇒ *LASSO* (*least absolute shrinkage and selection operator*)

LASSO: pick

$$\mathbf{w} = \arg\min_{\mathbf{w}} \lambda ||\mathbf{w}||_1 + \sum_{i=1}^{m} (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2$$

How?

Notes:

- no closed form solution!
- ℓ_1 norm is a convex function and squared loss is convex ⇒ problem can be solved efficiently! (true for every convex loss function)

LASSO and Sparse Solutions: Example

(Equivalent) one dimensional regression problem with squared loss:

$$\arg\min_{w\in\mathbb{R}} \left(\frac{1}{2m} \sum_{i=1}^{m} (x_i w - y_i)^2 + \lambda |w| \right)$$

Is equivalent to:

$$\arg\min_{w\in\mathbb{R}} \left(\frac{1}{2} \left(\frac{1}{m} \sum_{i=1}^{m} x_i^2 \right) w^2 - \left(\frac{1}{m} \sum_{i=1}^{m} x_i y_i \right) w + \lambda |w| \right)$$

Assume for simplicity that $\frac{1}{m}\sum_{i=1}^{m} x_i^2 = 1$, and let $\sum_{i=1}^{m} x_i y_i = \langle \mathbf{x}, \mathbf{y} \rangle$.

Then the optimal solution is

$$w = \text{sign}(\langle \mathbf{x}, \mathbf{y} \rangle)[\langle \mathbf{x}, \mathbf{y} \rangle / m - \lambda]_+$$

where $[a]_+ =^{(def)} \max\{a, 0\}$.

Tikhonov regularization

Regularization function: $R(\mathbf{w}) = \lambda ||\mathbf{w}||^2$

- $\lambda \in \mathbb{R}, \lambda > 0$
- ℓ_2 norm: $||\mathbf{w}||^2 = \sum_{i=1}^{d} w_i^2$

Therefore the *learning rule* is: pick

$$A(S) = \arg\min_{\mathbf{w}} \left(L_S(\mathbf{w}) + \lambda ||\mathbf{w}||^2 \right)$$

Intuition:

- $||\mathbf{w}||^2$ measures the "complexity" of hypothesis defined by $\mathbf{w}$
- λ regulates the tradeoff between the empirical risk ($L_S(\mathbf{w})$) or overfitting and the complexity ($||\mathbf{w}||^2$) of the model we pick

Ridge Regression

Linear regression with squared loss + Tikhonov regularization
$\Rightarrow$ *ridge regression*

Linear regression with squared loss:

- **given**: training set $S = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m))$, with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$
- **want**: $\mathbf{w}$ which minimizes empirical risk:

$$\mathbf{w} = \arg\min_{\mathbf{w}} \frac{1}{m} \sum_{i=1}^{m} (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2$$

equivalently, find $\mathbf{w}$ which minimizes the *residual sum of squares* $RSS(\mathbf{w})$

$$\mathbf{w} = \arg\min_{\mathbf{w}} RSS(\mathbf{w}) = \arg\min_{\mathbf{w}} \sum_{i=1}^{m} (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2$$

Linear regression: pick

$$\mathbf{w} = \arg\min_{\mathbf{w}} RSS(\mathbf{w}) = \arg\min_{\mathbf{w}} \sum_{i=1}^{m} (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2$$

Ridge regression: pick

$$\mathbf{w} = \arg\min_{\mathbf{w}} \left(\lambda ||\mathbf{w}||^2 + \sum_{i=1}^{m} (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2 \right)$$

RSS: Matrix Form

Let

$$\mathbf{X} = \begin{bmatrix} \cdots & \mathbf{x}_1 & \cdots \\ \cdots & \mathbf{x}_2 & \cdots \\ \cdots & \vdots & \cdots \\ \cdots & \mathbf{x}_m & \cdots \end{bmatrix}$$

$\mathbf{X}$: *design matrix*

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

$\Rightarrow$ we have that RSS is

$$\sum_{i=1}^{m} (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2 = (\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w})$$

Ridge Regression: Matrix Form

Linear regression: pick

$$\arg\min_{\mathbf{w}} (\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w})$$

Ridge regression: pick

$$\arg\min_{\mathbf{w}} \left(\lambda ||\mathbf{w}||^2 + (\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w}) \right)$$

Want to find $\mathbf{w}$ which minimizes $f(\mathbf{w}) = \lambda||\mathbf{w}||^2 + (\mathbf{y} - \mathbf{X}\mathbf{w})^T(\mathbf{y} - \mathbf{X}\mathbf{w})$.

How?

Compute gradient $\frac{\partial f(\mathbf{w})}{\partial \mathbf{w}}$ of objective function w.r.t $\mathbf{w}$ and compare it to 0.

$$\frac{\partial f(\mathbf{w})}{\partial \mathbf{w}} = 2\lambda\mathbf{w} - 2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\mathbf{w})$$

Then we need to find $\mathbf{w}$ such that

$$2\lambda\mathbf{w} - 2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\mathbf{w}) = 0$$