Machine learning: lecture 12

Tommi S. Jaakkola
MIT CSAIL

tommi@csail.mit.edu
Topics

• Complexity and model selection
  – Finite case
  – VC dimension
  – structural risk minimization
Why care about “complexity”?  

- We need a quantitative measure of complexity in order to be able to relate the training error (which we can observe) and the test error (that we’d like to optimize)
Simple case: finite number of classifiers

• Suppose we consider only a finite number of classifiers, $h_1(x), \ldots, h_m(x)$.

• How does the number of classifiers $m$ affect the difference between training and test errors?
Simple case: finite number of classifiers

- Suppose we consider only a finite number of classifiers, \( h_1(x), \ldots, h_m(x) \).
- How does the number of classifiers \( m \) affect the difference between training and test errors?

Let’s start by defining

\[
\hat{E}_n(i) = \frac{1}{n} \sum_{t=1}^{n} \text{Loss}(y_t, h_i(x_t)) = \text{empirical error of } h_i(x)
\]

\[
E(i) = E_{(x,y) \sim P} \{ \text{Loss}(y, h_i(x)) \} = \text{expected error of } h_i(x)
\]
Simple case cont’d

\[ \hat{E}_n(i) = \frac{1}{n} \sum_{t=1}^{n} \text{Loss}(y_t, h_i(x_t)) = \text{empirical error of } h_i(x) \]

\[ \mathcal{E}(i) = E_{(x,y) \sim P} \{ \text{Loss}(y, h_i(x)) \} = \text{expected error of } h_i(x) \]

- If we choose the classifier that minimizes the training error, \( \hat{i}_n = \arg\min_i \hat{E}_n(i) \), then

\[
\begin{align*}
\text{Training error} & = \hat{E}_n(\hat{i}_n) \\
\text{Test error} & = \mathcal{E}(\hat{i}_n)
\end{align*}
\]
Simple case cont’d

\[ \hat{E}_n(i) = \frac{1}{n} \sum_{t=1}^{n} \text{Loss}(y_t, h_i(x_t)) = \text{empirical error of } h_i(x) \]

\[ \mathcal{E}(i) = E_{(x,y) \sim P} \{ \text{Loss}(y, h_i(x)) \} = \text{expected error of } h_i(x) \]

- If we choose the classifier that minimizes the training error, 
  \( \hat{i}_n = \arg \min_i \hat{E}_n(i) \), then

  \[
  \text{Training error} = \hat{E}_n(\hat{i}_n)
  \]

  \[
  \text{Test error} = \mathcal{E}(\hat{i}_n)
  \]

- The training and test errors are necessarily close if

  \[
  |\hat{E}_n(i) - \mathcal{E}(i)| \leq \epsilon, \text{ for all } i = 1, \ldots, m
  \]
Simple case cont’d

- We’d like to evaluate the probability that the training error deviates more than $\epsilon$ from the corresponding test error:

$$P \left( \exists i : |\hat{E}_n(i) - E(i)| > \epsilon \right)$$

where the probability is over the choice of the training set.
Simple case cont’d

- We’d like to evaluate the probability that the training error deviates more than $\epsilon$ from the corresponding test error:

$$P\left(\exists i : |\hat{E}_n(i) - E(i)| > \epsilon\right)$$

where the probability is over the choice of the training set.

By using the fact that $P(A \text{ or } B) \leq P(A) + P(B)$ we get

$$P\left(\exists i : |\hat{E}_n(i) - E(i)| > \epsilon\right) \leq \sum_{i=1}^{m} P\left(|\hat{E}_n(i) - E(i)| > \epsilon\right)$$
\textbf{Simple case cont’d}

- We’d like to evaluate the probability that the training error deviates more than $\epsilon$ from the corresponding test error:

$$P \left( \exists i : |\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon \right)$$

where the probability is over the choice of the training set.

By using the fact that $P(A \text{ or } B) \leq P(A) + P(B)$ we get

$$P \left( \exists i : |\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon \right) \leq \sum_{i=1}^{m} P \left( |\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon \right)$$

$$\leq \sum_{i=1}^{m} 2 \exp \left( -2n\epsilon^2 \right) \quad \text{(Chernoff)}$$
Simple case cont’d

- We’d like to evaluate the probability that the training error deviates more than $\epsilon$ from the corresponding test error:

$$P \left( \exists i : |\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon \right)$$

where the probability is over the choice of the training set. By using the fact that $P(A \text{ or } B) \leq P(A) + P(B)$ we get

$$P \left( \exists i : |\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon \right) \leq \sum_{i=1}^{m} P \left( |\hat{\mathcal{E}}_n(i) - \mathcal{E}(i)| > \epsilon \right)$$

$$\leq \sum_{i=1}^{m} 2 \exp(-2n\epsilon^2) \quad \text{(Chernoff)}$$

$$= m \cdot 2 \exp(-2n\epsilon^2)$$
We’d like to evaluate the probability that the training error deviates more than $\epsilon$ from the corresponding test error:

$$P \left( \exists i : \left| \hat{E}_n(i) - E(i) \right| > \epsilon \right)$$

where the probability is over the choice of the training set. By using the fact that $P(A \text{ or } B) \leq P(A) + P(B)$ we get

$$P \left( \exists i : \left| \hat{E}_n(i) - E(i) \right| > \epsilon \right) \leq \sum_{i=1}^{m} P \left( \left| \hat{E}_n(i) - E(i) \right| > \epsilon \right)$$

$$\leq \sum_{i=1}^{m} 2 \exp(-2n\epsilon^2) \quad \text{(Chernoff)}$$

$$= m \cdot 2 \exp(-2n\epsilon^2) = \delta$$

where $(1 - \delta)$ is our “confidence” that the errors are close.
Simple case cont’d

• We can restate our result in terms of a bound on the expected error of any classifier in our set.

\[ m \cdot 2 \exp(-2n\epsilon^2) = \delta, \text{ or } \epsilon = \sqrt{\frac{1}{2n}(\log(2m) + \log(1/\delta))} \]

**Theorem:** With probability at least \(1 - \delta\) over the choice of the training set, for all \(i = 1, \ldots, m\)

\[ \mathcal{E}(i) \leq \hat{\mathcal{E}}_n(i) + \epsilon(n, m, \delta) \]

where \(\epsilon = \epsilon(n, m, \delta)\) given above is a “complexity penalty”. 

Tommi Jaakkola, MIT CSAIL
Simple case cont’d

• We can restate our result in terms of a bound on the expected error of any classifier in our set.

\[ m \cdot 2 \exp(-2n\epsilon^2) = \delta, \text{ or } \epsilon = \sqrt{\frac{1}{2n} \left( \log(2m) + \log\left(\frac{1}{\delta}\right) \right)} \]

**Theorem:** With probability at least \( 1 - \delta \) over the choice of the training set, for all \( i = 1, \ldots, m \)

\[ \mathcal{E}(i) \leq \hat{\mathcal{E}}_n(i) + \epsilon(n, m, \delta) \]

where \( \epsilon = \epsilon(n, m, \delta) \) given above is a “complexity penalty”.

• The complexity penalty
  – is an increasing function of \( m \)
  – increases as \( \delta \) decreases
  – decreases as a function of \( n \)
Measures of complexity

• “Complexity” is a measure of a set of classifiers, not any specific (fixed) classifier

• Many possible measures
  – degrees of freedom
  – description length
  – Vapnik-Chervonenkis (VC) dimension

etc.
VC-dimension: preliminaries

- **A set of classifiers $\mathcal{F}$:**
  
  For example, this could be the set of all possible linear separators, where $h \in \mathcal{F}$ means that

  \[ h(x) = \text{sign} \left( w_0 + w^T x \right) \]

  for some values of the parameters $w, w_0$. 
VC-dimension: preliminaries

- **Complexity:** how many different ways can we label $n$ training points $\{x_1, \ldots, x_n\}$ with classifiers $h \in F$?

  In other words, how many distinct binary vectors

  $$[h(x_1) \ h(x_2) \ \ldots \ h(x_n)]$$

  do we get by trying each $h \in F$ in turn?

  $$\begin{bmatrix} -1 & 1 & \ldots & 1 \end{bmatrix} h_1$$

  $$\begin{bmatrix} 1 & -1 & \ldots & 1 \end{bmatrix} h_2$$

  $$\ldots$$
VC-dimension: shattering

- A set of classifiers $F$ shatters $n$ points $\{x_1, \ldots, x_n\}$ if
  \[
  [h(x_1) \ h(x_2) \ \ldots \ h(x_n)], \ h \in F
  \]
generates all $2^n$ distinct labelings.

- Example: linear decision boundaries shatter (any) 3 points in 2D

  ![Diagrams](https://example.com/diagrams.png)

  but not any 4 points...
VC-dimension: shattering cont’d

• We cannot shatter 4 points in 2D with linear separators
  For example, the following labeling

  \[
  + x x
  \]
  \[
  x x
  \]
  \[
  - x +
  \]

cannot be produced with any linear separator

• More generally: the set of all $d$-dimensional linear separators can shatter exactly $d + 1$ points

• **Definition:** The VC-dimension of a set of classifiers $F$ is the number of points $F$ can shatter
Learning and VC-dimension

- We don’t really learn anything until after we have more than \( d_{VC} \) training examples.

- The number of labelings that the set of classifiers can generate over \( n \) points increases sub-exponentially only after \( n > d_{VC} \) (in this case \( d_{VC} = 100 \)).
Learning and VC-dimension

- Let $d_{VC}$ be the VC-dimension of our set of classifiers $F$.

**Theorem:** With probability at least $1 - \delta$ over the choice of the training set, for all $h \in F$

$$\mathcal{E}(h) \leq \hat{\mathcal{E}}_n(h) + \epsilon(n, d_{VC}, \delta)$$

where

$$\epsilon(n, d_{VC}, \delta) = \sqrt{\frac{d_{VC}(\log(2n/d_{VC}) + 1) + \log(1/(4\delta))}{n}}$$
Complexity and margin

- The number of possible labelings of points with large margin can be dramatically less than the (basic) VC-dimension.

- The set of separating hyperplanes which attain margin $\gamma$ or better for examples within a sphere of radius $R$ has VC-dimension bounded by $d_{VC}(\gamma) \leq R^2 / \gamma^2$. 
Model selection

- We try to find the model with the best balance of complexity and the fit to the training data
- Ideally, we would select a model from a nested sequence of models of increasing complexity (VC-dimension)

\[
\begin{align*}
\text{Model 1} & \quad d_1 \\
\text{Model 2} & \quad d_2 \\
\text{Model 3} & \quad d_3 \\
\end{align*}
\]

where \( d_1 \leq d_2 \leq d_3 \leq \ldots \)

- The model selection criterion is: find the model class that achieves the lowest upper bound on the expected loss

\[
\text{Expected error} \leq \text{Training error} + \text{Complexity penalty}
\]
Structural risk minimization cont’d

- We choose the model class $F_i$ that minimizes the upper bound on the expected error:

$$\mathcal{E}(\hat{h}_i) \leq \mathcal{E}_n(\hat{h}_i) + \sqrt{\frac{d_i (\log(2n/d_i) + 1) + \log(1/(4\delta))}{n}}$$

where $\hat{h}_i$ is the best classifier from $F_i$ selected on the basis of the training set.

![Graph showing the relationship between VC dimension, bound, complexity penalty, and training error.](image)
Example

- Models of increasing complexity
  
  Model 1  \[ K(x_1, x_2) = (1 + (x_1^T x_2)) \]
  
  Model 2  \[ K(x_1, x_2) = (1 + (x_1^T x_2))^2 \]
  
  Model 3  \[ K(x_1, x_2) = (1 + (x_1^T x_2))^3 \]
  
  . . .

- These are nested, i.e.,

  \[ F_1 \subseteq F_2 \subseteq F_3 \subseteq \ldots \]

  where \( F_k \) refers to the set of possible decision boundaries that the model \( k \) can represent.
Structural risk minimization: example

- Linear
- 2nd order polynomial
- 4th order polynomial
- 8th order polynomial
Structural risk minimization: example cont’d

- Number of training examples $n = 50$, confidence parameter $\delta = 0.05$.

<table>
<thead>
<tr>
<th>Model</th>
<th>$d_{VC}$</th>
<th>Empirical fit</th>
<th>$\epsilon(n, d_{VC}, \delta)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st order</td>
<td>3</td>
<td>0.06</td>
<td>0.5501</td>
</tr>
<tr>
<td>2nd order</td>
<td>6</td>
<td>0.06</td>
<td>0.6999</td>
</tr>
<tr>
<td>4th order</td>
<td>15</td>
<td>0.04</td>
<td>0.9494</td>
</tr>
<tr>
<td>8th order</td>
<td>45</td>
<td>0.02</td>
<td>1.2849</td>
</tr>
</tbody>
</table>

- Structural risk minimization would select the simplest (linear) model in this case.