Practical Issues in Machine Learning
Overfitting and Model selection

Aarti Singh

Machine Learning 10-701/15-781
Feb 3, 2010
True vs. Empirical Risk

**True Risk**: Target performance measure

- **Classification** – Probability of misclassification $P(f(X) ≠ Y)$
- **Regression** – Mean Squared Error $\mathbb{E}[(f(X) - Y)^2]$

Also known as “Generalization Error” – performance on a random test point $(X,Y)$
**True vs. Empirical Risk**

**True Risk**: Target performance measure

Classification – Probability of misclassification  \( P(f(X) \neq Y) \)

Regression – Mean Squared Error  \( \mathbb{E}[(f(X) - Y)^2] \)

Also known as “Generalization Error” – performance on a random test point (X,Y)

**Empirical Risk**: Performance on training data

Classification – Proportion of misclassified examples  \( \frac{1}{n} \sum_{i=1}^{n} 1_{f(X_i) \neq Y_i} \)

Regression – Average Squared Error  \( \frac{1}{n} \sum_{i=1}^{n} (f(X_i) - Y_i)^2 \)
Overfitting

Is the following predictor a good one?

\[ f(x) = \begin{cases} Y_i, & x = X_i \text{ for } i = 1, \ldots, n \\ \text{any value,} & \text{otherwise} \end{cases} \]

What is its empirical risk? (performance on training data)

zero!

What about true risk?

\( > \) zero

Will predict very poorly on new random test point, \textit{Large generalization error}!
Overfitting

If we allow very complicated predictors, we could overfit the training data.

Examples: Classification (0-NN classifier, decision tree with one sample/leaf)

Football player?
- Yes
- No
Overfitting

If we allow very complicated predictors, we could overfit the training data.

Examples: Regression (Polynomial of order $k$ – degree up to $k-1$)  

```
code online
```
If we allow very complicated predictors, we could overfit the training data.

Empirical risk is no longer a good indicator of true risk.
**Behavior of True Risk**

Want predictor based on training data $\hat{f}_n$ to be as good as optimal predictor $f^*$

Excess Risk

$$E[R(\hat{f}_n)] - R^*$$

wrt the distribution of training data

- Why is the risk of $\hat{f}_n$ a random quantity?

$$R(\hat{f}_n) = P_{XY}(\hat{f}_n(X) \neq Y)$$

$$R(\hat{f}_n) = \mathbb{E}_{XY}[(\hat{f}_n(X) - Y)^2]$$

$\hat{f}_n$ depends on random training dataset
Behavior of True Risk

Want predictor based on training data $\hat{f}_n$ to be as good as optimal predictor $f^*$

Excess Risk

$$E \left[ R(\hat{f}_n) \right] - R^* = \left( E[R(\hat{f}_n)] - \inf_{f \in \mathcal{F}} R(f) \right) + \left( \inf_{f \in \mathcal{F}} R(f) - R^* \right)$$

Due to randomness of training data

Due to restriction of model class

Finite sample size + noise
Behavior of True Risk

\[
E \left[ R(\hat{f}_n) \right] - R^* = \underbrace{\left( E[R(\hat{f}_n)] - \inf_{f \in \mathcal{F}} R(f) \right)}_{\text{estimation error}} + \underbrace{\left( \inf_{f \in \mathcal{F}} R(f) - R^* \right)}_{\text{approximation error}}
\]

Complexity of $\mathcal{F}$
Bias – Variance Tradeoff

Regression: \[ Y = f^*(X) + \epsilon \quad \epsilon \sim \mathcal{N}(0, \sigma^2) \]

\[ R^* = \mathbb{E}_{XY}[(f^*(X) - Y)^2] = \mathbb{E}[\epsilon^2] = \sigma^2 \]

\[ \mathbb{E}_D[R(\hat{f}_n)] = \mathbb{E}_{X,Y,D}[(\hat{f}_n(X) - Y)^2] \]

\[ = \mathbb{E}_{X,Y,D}[(\hat{f}_n(X) - \mathbb{E}_D[\hat{f}_n(X)] + \mathbb{E}_D[\hat{f}_n(X)] - Y)^2] \]

\[ = \mathbb{E}_{X,Y,D}[(\hat{f}_n(X) - \mathbb{E}_D[\hat{f}_n(X)])^2 + (\mathbb{E}_D[\hat{f}_n(X)] - Y)^2 \]

\[ + 2(\hat{f}_n(X) - \mathbb{E}_D[\hat{f}_n(X)])(\mathbb{E}_D[\hat{f}_n(X)] - Y) \]

\[ = \mathbb{E}_{X,Y,D}[(\hat{f}_n(X) - \mathbb{E}_D[\hat{f}_n(X)])^2] + \mathbb{E}_{X,Y,D}[(\mathbb{E}_D[\hat{f}_n(X)] - Y)^2] \]

\[ + \mathbb{E}_{X,Y} 2(\mathbb{E}_D[\hat{f}_n(X)] - \mathbb{E}_D[f_n(X)])(\mathbb{E}_D[\hat{f}_n(X)] - Y) \]

Notice: Optimal predictor does not have zero error
Bias – Variance Tradeoff

Regression: \( Y = f^*(X) + \epsilon \quad \epsilon \sim \mathcal{N}(0, \sigma^2) \)

Notice: Optimal predictor does not have zero error

\[ R^* = \mathbb{E}_{X,Y}[(f^*(X) - Y)^2] = \mathbb{E}[\epsilon^2] = \sigma^2 \]

\[ \mathbb{E}_D[R(\hat{f}_n)] = \mathbb{E}_{X,Y,D}[(\hat{f}_n(X) - Y)^2] \]

\[ = \mathbb{E}_{X,Y,D}[(\hat{f}_n(X) - \mathbb{E}_D[\hat{f}_n(X)])^2] + \mathbb{E}_{X,Y,D}[(\mathbb{E}_D[\hat{f}_n(X)] - Y)^2] \]

variance - how much does the predictor vary about its mean for different training data points

Now, let's look at the second term:

\[ \mathbb{E}_{X,Y,D}[(\mathbb{E}_D[\hat{f}_n(X)] - Y)^2] = \mathbb{E}_{X,Y}[(\mathbb{E}_D[\hat{f}_n(X)] - Y)^2] \]

Note: this term doesn’t depend on D
Bias – Variance Tradeoff

\[ \mathbb{E}_{X,Y} \left[ (\mathbb{E}_D[\hat{f}_n(X)] - Y)^2 \right] = \mathbb{E}_{X,Y} \left[ (\mathbb{E}_D[\hat{f}_n(X)] - f^*(X) - \epsilon)^2 \right] \\
= \mathbb{E}_{X,Y} \left[ (\mathbb{E}_D[\hat{f}_n(X)] - f^*(X))^2 + \epsilon^2 \\
- 2\epsilon (\mathbb{E}_D[\hat{f}_n(X)] - f^*(X)) \right] \\
= \mathbb{E}_{X,Y} \left[ (\mathbb{E}_D[\hat{f}_n(X)] - f^*(X))^2 \right] + \mathbb{E}_{X,Y} \left[ \epsilon^2 \right] \\
- 2\mathbb{E}_{X,Y} \left[ \epsilon (\mathbb{E}_D[\hat{f}_n(X)] - f^*(X)) \right] \\
0 \text{ since noise is independent and zero mean} \\
= \mathbb{E}_{X,Y} \left[ (\mathbb{E}_D[\hat{f}_n(X)] - f^*(X))^2 \right] + \mathbb{E}_{X,Y} \left[ \epsilon^2 \right] \\
\text{bias}^2 - \text{how much does the predictor on average differ from the optimal predictor} \\
\text{noise variance}
Bias – Variance Tradeoff

Regression: \( Y = f^*(X) + \epsilon \quad \epsilon \sim \mathcal{N}(0, \sigma^2) \)

\[
R^* = \mathbb{E}_{X,Y}[(f^*(X) - Y)^2] = \mathbb{E}[\epsilon^2] = \sigma^2
\]

\[
\mathbb{E}_D[R(\hat{f}_n)] = \mathbb{E}_{X,Y,D}[(\hat{f}_n(X) - Y)^2]
\]
\[
= \mathbb{E}[(\hat{f}_n(X) - \mathbb{E}[\hat{f}_n(X)])^2] + \mathbb{E}[(\mathbb{E}[\hat{f}_n(X)] - f^*(X))^2] + \sigma^2
\]

Excess Risk = \( \mathbb{E}_D[R(\hat{f}_n)] - R^* \) = variance + bias^2

Random component ≡ est err  ≡ approx err

Notice: Optimal predictor does not have zero error
Bias – Variance Tradeoff

3 Independent training datasets

Large bias, Small variance – poor approximation but robust/stable

Small bias, Large variance – good approximation but instable
Examples of Model Spaces

Model Spaces with increasing complexity:

• Nearest-Neighbor classifiers with varying neighborhood sizes $k = 1, 2, 3, \ldots$
  Small neighborhood => Higher complexity

• Decision Trees with depth $k$ or with $k$ leaves
  Higher depth/ More # leaves => Higher complexity

• Regression with polynomials of order $k = 0, 1, 2, \ldots$
  Higher degree => Higher complexity

• Kernel Regression with bandwidth $h$
  Small bandwidth => Higher complexity

How can we select the right complexity model?
Model Selection

Setup:

Model Classes \( \{ \mathcal{F}_\lambda \}_{\lambda \in \Lambda} \) of increasing complexity \( \mathcal{F}_1 \prec \mathcal{F}_2 \prec \ldots \)

\[
\min_{\lambda} \min_{f \in \mathcal{F}_\lambda} J(f, \lambda)
\]

We can select the right complexity model in a data-driven/adaptive way:

- Cross-validation
- Method of Sieves
- Structural Risk Minimization
- Complexity Regularization
- Information Criteria - Minimum Description Length, AIC, BIC
Hold-out method

We would like to pick the model that has smallest generalization error.

Can judge generalization error by using an independent sample of data.

**Hold-out procedure:**

1. Split into two sets: 
   - Training dataset: \( D_T = \{ X_i, Y_i \}_{i=1}^m \)
   - Validation dataset: \( D_V = \{ X_i, Y_i \}_{i=m+1}^n \)

2. Use \( D_T \) for training a predictor from each model class:

\[
\hat{f}_\lambda = \arg \min_{f \in F_\lambda} \hat{R}_T(f)
\]

Evaluated on training dataset \( D_T \)
Hold-out method

3) Use $D_v$ to select the model class which has smallest empirical error on $D_v$

$$\hat{\lambda} = \arg\min_{\lambda \in \Lambda} \hat{R}_V(\hat{f}_\lambda)$$

Evaluated on validation dataset $D_v$

4) Hold-out predictor

$$\hat{f} = \hat{f}_{\hat{\lambda}}$$

**Intuition:** Small error on one set of data will not imply small error on a randomly sub-sampled second set of data

Ensures method is “stable”
Hold-out method

Drawbacks:

- May not have enough data to afford setting one subset aside for getting a sense of generalization abilities
- Validation error may be misleading (bad estimate of generalization error) if we get an “unfortunate” split

Limitations of hold-out can be overcome by a family of random sub-sampling methods at the expense of more computation.
Cross-validation

K-fold cross-validation

Create K-fold partition of the dataset.
Form K hold-out predictors, each time using one partition as validation and rest K-1 as training datasets.
Final predictor is average/majority vote over the K hold-out estimates.

![Diagram showing K-fold cross-validation]

- Run 1
  - Training
  - Validation
  \[ \Rightarrow \hat{f}_1 \]

- Run 2
  - Training
  - Validation
  \[ \Rightarrow \hat{f}_2 \]

- Run K
  - Training
  - Validation
  \[ \Rightarrow \hat{f}_K \]
Cross-validation

Leave-one-out (LOO) cross-validation

Special case of K-fold with K=n partitions
Equivalently, train on n-1 samples and validate on only one sample per run for n runs

![Diagram showing LOO cross-validation]

- Run 1: Training $\Rightarrow \hat{f}_1$
- Run 2: Training $\Rightarrow \hat{f}_2$
- Run K: Training $\Rightarrow \hat{f}_K$
Cross-validation

Random subsampling

Randomly subsample a fixed fraction $\alpha n$ ($0 < \alpha < 1$) of the dataset for validation.
Form hold-out predictor with remaining data as training data.
Repeat K times
Final predictor is average/majority vote over the K hold-out estimates.

```
Run 1  Run 2  Run K
\[ \Rightarrow \hat{f}_1 \]
\[ \Rightarrow \hat{f}_2 \]
\[ \Rightarrow \hat{f}_K \]
```

```
- | \[ \text{training} \] | \[ \text{validation} \] 
```

Total number of examples
Estimating generalization error

Generalization error \( E_D[R(\hat{f}_n)] \)

Hold-out \( \equiv \) 1-fold:
Error estimate = \( \hat{R}_V(\hat{f}_T) \)

K-fold/LOO/random sub-sampling:
Error estimate = \( \frac{1}{K} \sum_{k=1}^{K} \hat{R}_{V_k}(\hat{f}_{T_k}) \)

We want to estimate the error of a predictor based on \( n \) data points.

If \( K \) is large (close to \( n \)), bias of error estimate is small since each training set has close to \( n \) data points.

However, variance of error estimate is high since each validation set has fewer data points and \( \hat{R}_{V_k} \) might deviate a lot from the mean.

- training
- validation

Run 1
\( \Rightarrow \hat{f}_1 \)

Run 2
\( \Rightarrow \hat{f}_2 \)

Run K
\( \Rightarrow \hat{f}_K \)
Practical Issues in Cross-validation

How to decide the values for $K$ and $\alpha$?

- **Large $K$**
  + The bias of the error estimate will be small
  - The variance of the error estimate will be large
  - The computational time will be very large as well (many experiments)

- **Small $K$**
  + The number of experiments and, therefore, computation time are reduced
  + The variance of the error estimate will be small
  - The bias of the error estimate will be large

In practice, the choice of the number of folds depends on the size of the dataset:

For large datasets, even 3-Fold Cross Validation will be quite accurate
For very sparse datasets, we may have to use leave-one-out in order to train on as many examples as possible

- A common choice is $K=10$ and $\alpha = 0.1$
Occam’s Razor

William of Ockham (1285-1349) *Principle of Parsimony:*

“One should not increase, beyond what is necessary, the number of entities required to explain anything.”

Alternatively, seek the simplest explanation.

Penalize complex models based on

- Prior information (bias)
- Information Criterion (MDL, AIC, BIC)
Importance of Domain knowledge

Distribution of photon arrivals

Compton Gamma-Ray Observatory Burst and Transient Source Experiment (BATSE)

Oil Spill Contamination
Method of Sieves

Consider a sequence of models whose complexity grows with # training data, \( n \)

\[
\mathcal{F}_1 < \mathcal{F}_2 < \ldots \mathcal{F}_n < \ldots
\]

\[
\hat{f}_n = \arg \min_{f \in \mathcal{F}_n} \hat{R}_n(f)
\]

**Why does optimal complexity depend on # training data?**

Consider kernel regression in \( d \)-dimensions: complexity \( \equiv \) bandwidth \( h \)

Large \( h \) – average more data points, reduce noise

Lower variance \( \propto \frac{1}{nh^d} = \# \) pts in h-ball

Small \( h \) – less smoothing, more accurate fit

Lower bias \( \propto h^\alpha \rightarrow \) Smoothness of target function
Consider a sequence of models whose complexity grows with \( \# \) training data, \( n \)

\[ \mathcal{F}_1 < \mathcal{F}_2 < \ldots \mathcal{F}_n < \ldots \]

\[ \hat{f}_n = \arg \min_{f \in \mathcal{F}_n} \hat{R}_n(f) \]

**Why does optimal complexity depend on \( \# \) training data?**

Consider kernel regression in \( d \)-dimensions: complexity \( \equiv \) bandwidth \( h \)

Bias-variance tradeoff:

\[ \text{Bias}^2 + \text{Variance} \propto h^{2\alpha} + \frac{1}{nh^d} \]

If smoothness \( \alpha \) is known, we can choose bandwidth \( h \) as:

\[ h \asymp n^{-\frac{2\alpha}{2\alpha + d}} \]

How to choose scaling constant? **Cross-validation**
Structural Risk Minimization

Penalize models using bound on deviation of true and empirical risks.

\[ \hat{f}_n = \arg\min_{f \in \mathcal{F}} \left\{ \hat{R}_n(f) + C(f) \right\} \]

With high probability, \[ |R(f) - \hat{R}_n(f)| \leq C(f) \quad \forall f \in \mathcal{F} \]

Bound on deviation from true risk

Concentration bounds (later)

High prob Upper bound on true risk

C(f) - large for complex models
Structural Risk Minimization

Penalize models using bound on deviation of true and empirical risks.

\[ \hat{f}_n = \arg \min_{f \in \mathcal{F}} \left\{ \hat{R}_n(f) + C(f) \right\} \]

With high probability,

\[ |R(f) - \hat{R}_n(f)| \leq C(f) \quad \forall f \in \mathcal{F} \]

Concentration bounds (later)

\[ R(\hat{f}_n) \leq \hat{R}_n(\hat{f}_n) + C(\hat{f}_n) = \min_{f \in \mathcal{F}} \left\{ \hat{R}_n(f) + C(f) \right\} \]

\[ \leq \min_{f \in \mathcal{F}} \{ R(f) + 2C(f) \} \]

\[ R(\hat{f}_n) - R^* \leq \min_{f \in \mathcal{F}} \{ R(f) - R^* + 2C(f) \} \]

approx err est err
Structural Risk Minimization

Penalize models using bound on deviation of true and empirical risks.

\[ \hat{f}_n = \arg \min_{f \in \mathcal{F}} \left\{ \hat{R}_n(f) + C(f) \right\} \]

How does structural risk minimization help in kernel regression?

Let \( C(f) \propto \frac{1}{nh^d} \) \quad \forall f \in \mathcal{F}_h \)

With high prob. \( R(\hat{f}_n) - R^* \leq \min_{f \in \mathcal{F}} \{ R(f) - R^* + 2C(f) \} \)

\[ \leq \min_{h} \min_{f \in \mathcal{F}_h} \{ R(f) - R^* + 2C(f) \} \]

\[ \propto \min_{h} \left\{ h^{2\alpha} + \frac{1}{nh^d} \right\} \]

Error automatically corresponds to best \( h \)
Deviation bounds are typically pretty loose, for small sample sizes. In practice,

\[ \hat{f}_n = \arg \min_{f \in \mathcal{F}} \left\{ \hat{R}_n(f) + \lambda C(f) \right\} \]

Choose by cross-validation!

**Problem:** Identify flood plain from noisy satellite images

- Noiseless image
- Noisy image
- True Flood plain (elevation level > x)
Structural Risk Minimization

Deviation bounds are typically pretty loose, for small sample sizes. In practice,

\[ \hat{f}_n = \arg \min_{f \in \mathcal{F}} \left\{ \hat{R}_n(f) + \lambda C(f) \right\} \]

Choose by cross-validation!

**Problem:** Identify flood plain from noisy satellite images

- **True Flood plain** (elevation level > x)
- **Zero penalty**
- **CV penalty**
- **Theoretical penalty**
Complexity Regularization

Penalize complex models using \textbf{prior knowledge}.

\[
\hat{f}_n = \arg \min_{f \in \mathcal{F}} \left\{ \hat{R}_n(f) + C(f) \right\}
\]

Bayesian viewpoint:

prior probability of \( f \) \( \equiv e^{-C(f)} \)

cost is small if \( f \) is highly probable, cost is large if \( f \) is improbable

ERM (empirical risk minimization) over a restricted class \( \mathcal{F} \), e.g. linear classifiers,
\( \equiv \) uniform prior on \( f \in \mathcal{F} \), zero probability for other predictors

\[
\hat{f}^L_n = \arg \min_{f \in \mathcal{F}_L} \hat{R}_n(f)
\]
Penalize complex models using **prior knowledge**.

\[
\hat{f}_n = \arg \min_{f \in \mathcal{F}} \left\{ \hat{R}_n(f) + C(f) \right\}
\]

**Cost of model** (log prior)

Examples: MAP estimators
Regularized Linear Regression - Ridge Regression, Lasso

\[
\hat{\theta}_{MAP} = \arg \max_{\theta} \log p(D|\theta) + \log p(\theta)
\]

\[
\hat{\beta}_{MAP} = \arg \min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda \|eta\|
\]

Penalize models based on some norm of regression coefficients

How to choose tuning parameter \( \lambda \)? **Cross-validation**
Information Criteria

Penalize complex models based on their information content.

\[ \hat{f}_n = \arg \min_{f \in \mathcal{F}} \left\{ \hat{R}_n(f) + C(f) \right\} \]

MDL (Minimum Description Length)

Example: Binary Decision trees

\[ \mathcal{F}^T = \bigcup_{k \geq 1} \mathcal{F}^T_k \]

prefix encode each element \( f \) of \( \mathcal{F}^T \)

\[ C(f) = 3k - 1 \text{ bits} \]

\( k \) leaves \( \Rightarrow \) \( 2k - 1 \) nodes

\( 2k - 1 \) bits to encode tree structure

+ \( k \) bits to encode label of each leaf (0/1)

5 leaves \( \Rightarrow \) 9 bits to encode structure
Information Criteria

Penalize complex models based on their information content.

\[ \hat{f}_n = \arg \min_{f \in \mathcal{F}} \left\{ \hat{R}_n(f) + C(f) \right\} \]

MDL (Minimum Description Length)

Other Information Criteria:

**AIC (Akaike IC)** \( C(f) = \# \text{ parameters} \)

Allows \# parameters to be infinite as \# training data \( n \) become large

**BIC (Bayesian IC)** \( C(f) = \# \text{ parameters} \times \log n \)

Penalizes complex models more heavily – limits complexity of models as \# training data \( n \) become large
Summary

True and Empirical Risk

Over-fitting

Approx err vs Estimation err, Bias vs Variance tradeoff

Model Selection

- Hold-out, K-fold cross-validation
- Method of Sieves
- Structural Risk Minimization
- Complexity Regularization
- Information Criteria – MDL, AIC, BIC