Generalization

Training error:

same? diff by how much?

$$E_{train} = \frac{1}{n} \underbrace{\sum_{i=1}^{n}}_{\text{training examples}} \overbrace{\text{error}}^{\text{same? diff by how much?}} (\underbrace{f_D(\underline{x}_i)}_{\text{predicted value}}, \underbrace{y_i}_{\text{true label}})$$

Generalization error:

- how well we will do on future data
  - we don't know x_i or y_i of future data
- but we know its range $\{x, y\}$

* $E_{trn} \leq E_{gen}$

$$E_{gen} = \underbrace{\int}_{\text{over all possible } x, y} \overbrace{\text{error}(f_D(\underline{x}), y)}^{\text{error as before}} \underbrace{p(y, \underline{x})}_{\text{how often we expect to see such } x \text{ and } y} d\underline{x}$$

* we can never compute generalization error!

Estimating Gen. Err.

Testing Error

- set aside part of trn data → test set
- learn a predictor w/o this
- predict values for test set, compute error
- gives an estimate of the true gen. err.

* if test set unbiased, $\lim_{n \to \infty} E_{test} = E_{gen}$.

Confidence Interval for future error

- range of errors expected for future test sets.
  - $E_{test} \pm \Delta E$ such that 85% of future test sets fall under that interval
  - E_{test} ↓ unbiased estimate of the true error rate, E
    - ↳ p(system will misclassify a random instance)
- take a random set of n instances, how many misclassified?

example: flip a coin n times. How many heads will we have?
↳ E-biased

Binomial dist w/ $\mu = nE$, $\sigma^2 = nE(1-E)$

$E_{future} = \frac{\# \text{ misclassified}}{n} \sim$ Gaussian, mean E, var $= E(1-E)/n$

confidence interval $= E \pm \sqrt{E(1-E)/n} \cdot \Phi^{-1}\left(\frac{1-p}{2}\right)$

Cross Validation

- conflicting priorities when splitting the dataset
  - estimate future error as accurately as possible
    - large testing set: big n_{test} → tight confidence interval
  - learn classifier as accurately as possible
    - large training set: big n_{train} → better estimates
  - trn and test sets cannot overlap → $n_{test} + n_{train} =$ constant
- Idea: evaluate Train → Test, then Test → Train, average results
  - every point is both training and testing, never at the same time
    - reduces chances of getting an unusual (biased) testing set.
  - 5-fold cross-validation
    - randomly split the data into 5 sets
    - test on each in turn (train on 4 others)
    - average over 5 folds
  - more common: 10 fold.

Leave-one-out

- n-fold cross-validation (n → no. of instances)
  - predict each instance, train on all (n-1) other instances.

Stratification

- keep class labels balanced acc. trn/test sets.
- simple way to guard against unlucky splits.
  - recipe: - randomly split to k parts.
    - assemble i^{th} part from all classes to make i^{th} fold.