We have \((x, y) \sim D\)

- Training set drawn from \(D\)
- Goal is to discover a hypothesis \(h\)

\[
\min_{h \in C} \quad \mathbb{E}(x, y) \sim D \left[ \text{error}(h(x), y) \right]
\]

Two reasons why \(h\) may do poorly on new examples

- **Bias**: Hypothesis class is not rich enough to explain the data.
  - High training error \(\rightarrow\) Underfitting
  - High test error

- **Variance**: The hypothesis is too tightly aligned to the training set.
  - Low training error \(\rightarrow\) Overfitting
  - High test error

---

![Graph showing bias, variance, and model complexity](image_url)

*Overparametrization*
Regularization: Minimize

\[ J_\lambda(\theta) = J(\theta) + \lambda R(\theta) \]

Old cost \( \downarrow \) hyperparameter \( \frac{1}{2} ||\theta||_2^2 \)

\( ||\theta||_1 \)

Soft SVM: minimize

\[ \frac{1}{2} ||\theta||_2^2 + \text{Hinge Loss}(\theta, s) \]

Choosing \( \lambda \):

- When \( \lambda = 0 \), regularized cost fn is same as old.
- When \( \lambda \) is high, reducing norm of \( \theta \) is of highest priority.

How do we choose hyperparameters, models
Different learning models:
- Due to different values of hyperparameters
- Due to different hypotheses (different degrees of polynomial fits, ...)
- Due to different learning algorithms

**Cross Validation**: Set of training examples \( S \)

\[
S = S_{\text{train}} \cup S_{\text{cv}}
\]

For each model \( M \):
- Train \( M \) on \( S_{\text{train}} \)
- Compute error of \( M \) on \( S_{\text{cv}} \)

Pick the model that has lowest error on \( S_{\text{cv}} \)

\( M^* \)

\( \rightarrow \) Train \( M^* \) on the entire \( S \)

In practice \( S = S_{\text{train}} \cup S_{\text{cv}} \cup S_{\text{test}} \)

- \( S_{\text{train}} \) used to identify the right model & find hypo.
- \( S_{\text{test}} \) used to estimate test error of hypo.

Typically,
- \( S_{\text{train}} 70\% \), \( S_{\text{cv}} 30\% \)
- \( S_{\text{train}} 60\% \), \( S_{\text{cv}} 20\% \), \( S_{\text{test}} 20\% \).
$S \sim 1$ million examples.

**k-fold Cross-Validation** ($k \sim 10$)

$$S = \sum_{i=1}^{k} S_i$$

For each model $M$

For each $i \in 1, \ldots, k$

Train $M$ on $S_i, U \cup S_{i+1}, \ldots, U \cup S_k$

Measure error of $M$ on $S_i$

Average of errors for $S_i$ used as cross-validation set.

Pick $M_*$ that has lowest ave error.

Used a smaller cross-validation set.

Multiple trainings of a given model.

**Leave-One-Out Cross Validation**

**k-fold cross validation where**

$k = n$ ($n$ - size of $S$)

**Feature Selection**

Pick a subset of features that are relevant for a problem.
Forward Search.

- Split $S$ into $S_{train}, S_{cv}$.
- Start $F = \emptyset$
- Repeat
  - For each feature $x_i \notin F$
    - Train algo $FU \Sigma x_i \Sigma$ features
    - Compute error of hypo on $S_{cv}$
  - Pick feature $x_i$ that results in lowest error and add to $F$. 