Alpaydin Chapter 2, Mitchell Chapter 7

- Alpaydin slides are in turquoise.
  - Ethem Alpaydin, copyright: The MIT Press, 2010.
  - alpaydin@boun.edu.tr
  - `http://www.cmpe.boun.edu.tr/~ethem/i2ml2e`
- All other slides are based on Mitchell.

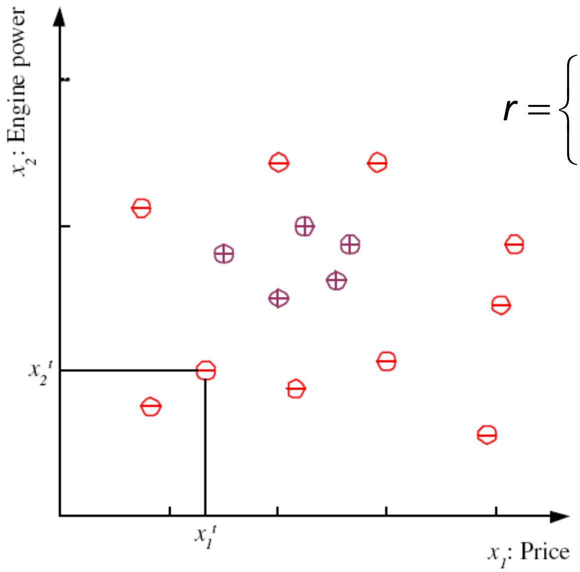
Learning a Class from Examples

- Class C of a “family car”
  - Prediction: Is car *x* a family car?
  - Knowledge extraction: What do people expect from a family car?
- Output:

  Positive (+) and negative (–) examples
- Input representation:

  x_1: price, x_2 : engine power

Training set $\mathcal{X}$

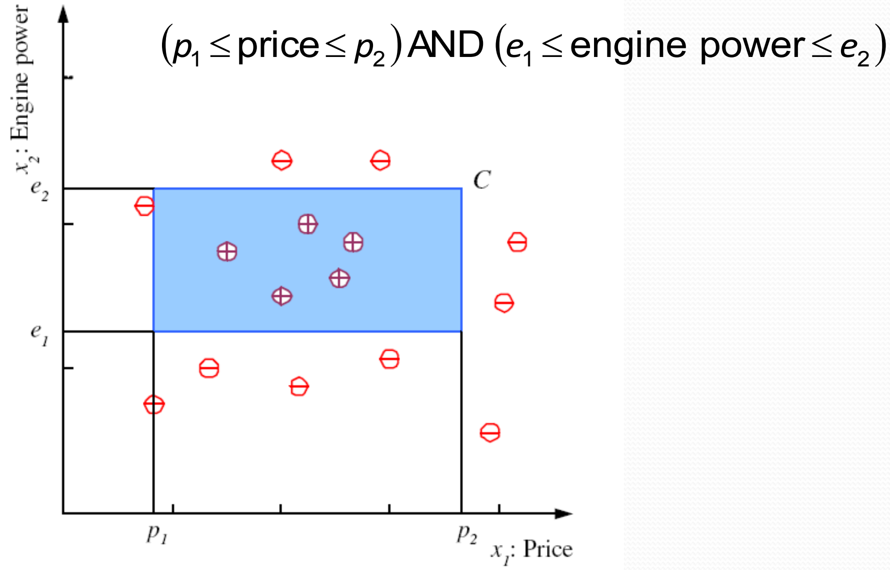


$$\mathcal{X} = \{\mathbf{x}^t, r^t\}_{t=1}^{N}$$

$$r = \begin{cases} 1 \text{ if } \mathbf{x} \text{ is positive} \\ 0 \text{ if } \mathbf{x} \text{ is negative} \end{cases}$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

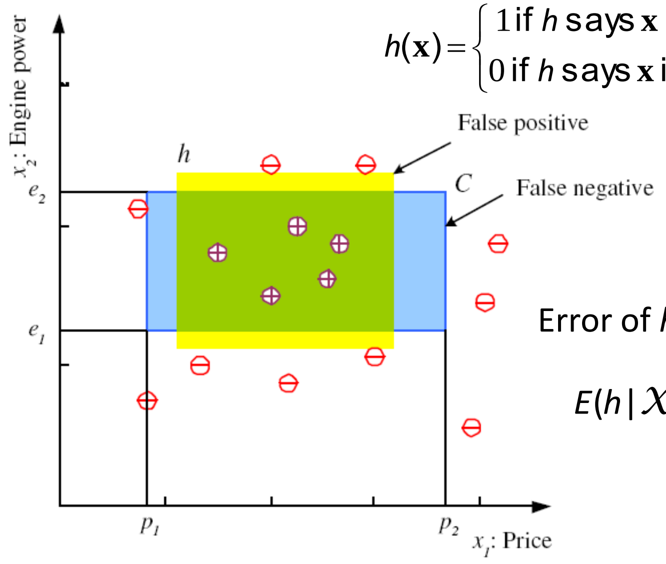
Class C



$$(p_1 \leq \text{price} \leq p_2) \text{ AND } (e_1 \leq \text{engine power} \leq e_2)$$

Hypothesis class $\mathcal{H}$

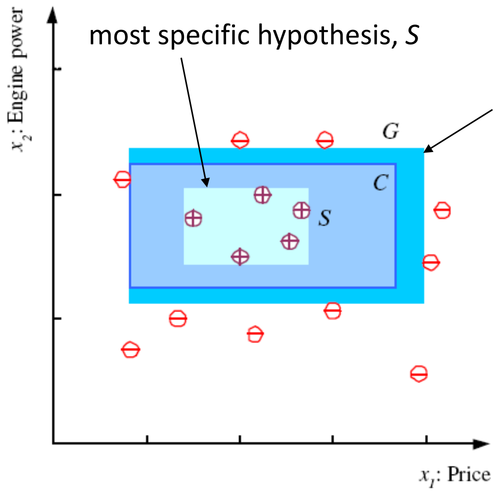
$$h(\mathbf{x}) = \begin{cases} 1 \text{ if } h \text{ says } \mathbf{x} \text{ is positive} \\ 0 \text{ if } h \text{ says } \mathbf{x} \text{ is negative} \end{cases}$$



Error of h on $\mathcal{H}$

$$E(h \mid \mathcal{X}) = \sum_{t=1}^{N} 1\left(h\left(\mathbf{x}^t\right) \neq r^t\right)$$

S, G, and the Version Space



most specific hypothesis, S

most general hypothesis, G

$h \in$ H, between S and G is consistent and make up the version space (Mitchell, 1997)

Computational Learning Theory (from Mitchell Chapter 7)

- Theoretical characterization of the **difficulties** and **capabilities** of learning algorithms.
- Questions:
  - Conditions for successful/unsuccessful learning
  - Conditions of success for particular algorithms
- Two frameworks:
  - Probably Approximately Correct (PAC) framework: classes of hypotheses that can be learned; complexity of hypothesis space and bound on training set size.
  - Mistake bound framework: number of training errors made before correct hypothesis is determined.

Computational Learning Theory

What general laws constrain inductive learning?

We seek theory to relate:

- Probability of successful learning
- Number of training examples
- Complexity of hypothesis space
- Accuracy to which target concept is approximated
- Manner in which training examples presented

Specific Questions

- Sample complexity: How many training examples are needed for a learner to converge?
- Computational complexity: How much computational effort is needed for a learner to converge?
- Mistake bound: How many training examples will the learner misclassify before converging?

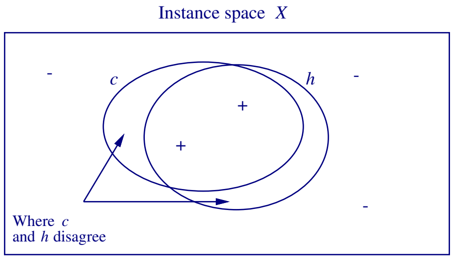
Issues: When to say it was successful? How are inputs acquired?

Sample Complexity

How many training examples are sufficient to learn the target concept?

1. If learner proposes instances, as queries to teacher
   - Learner proposes instance x, teacher provides $c(x)$
2. If teacher (who knows c) provides training examples
   - teacher provides sequence of examples of form $\langle x, c(x) \rangle$
3. If some random process (e.g., nature) proposes instances
   - instance x generated randomly, teacher provides $c(x)$

True Error of a Hypothesis



Definition: The **true error** (denoted $error_{\mathcal{D}}(h)$) of hypothesis h with respect to target concept c and distribution $\mathcal{D}$ is the probability that h will misclassify an instance drawn at random according to $\mathcal{D}$.

$$error_{\mathcal{D}}(h) \equiv \Pr_{x \in \mathcal{D}}[c(x) \neq h(x)]$$

Two Notions of Error

Training error of hypothesis h with respect to target concept c

- How often $h(x) \neq c(x)$ over training instances

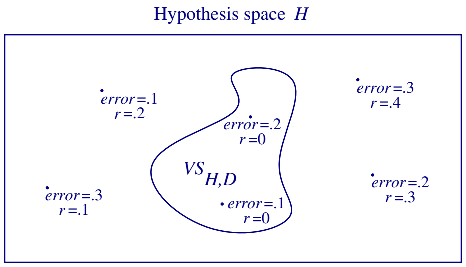
True error of hypothesis h with respect to c

- How often $h(x) \neq c(x)$ over future random instances

Our concern:

- Can we bound the true error of h given the training error of h?
- First consider when training error of h is zero (i.e., $h \in VS_{H,D}$)

Exhausting the Version Space



(r = training error, $error$ = true error)

Definition: The version space $VS_{H,D}$ is said to be ϵ-**exhausted** with respect to c and $\mathcal{D}$, if every hypothesis h in $VS_{H,D}$ has error less than ϵ with respect to c and $\mathcal{D}$.

$$(\forall h \in VS_{H,D})\ error_{\mathcal{D}}(h) < \epsilon$$

How many examples will ϵ-exhaust the VS?

Theorem: [Haussler, 1988].

> If the hypothesis space H is finite, and D is a sequence of $m \geq 1$ independent random examples of some target concept c, then for any $0 \leq \epsilon \leq 1$, the probability that the version space with respect to H and D is not ϵ-exhausted (with respect to c) is less than
>
> $$|H|e^{-\epsilon m}$$

This bounds the probability that any consistent learner will output a hypothesis h with $error(h) \geq \epsilon$

If we want this probability to be below δ

$$|H|e^{-\epsilon m} \leq \delta$$

then

$$m \geq \frac{1}{\epsilon}(\ln |H| + \ln(1/\delta))$$

Proof of ϵ-Exhasting Theorem

Theorem: Prob. of $VS_{H,D}$ **not** being ϵ-exhausted is $\leq |H|e^{-\epsilon m}$.
Proof:

- Let $h_i \in H$ ($i = 1..k$) be those that have true error greater than ϵ wrt c ($k \leq |H|$).
- We fail to ϵ-exhaust the VS iff at least one h_i is consistent with all m sample training instances (note: they have true error greater than ϵ).
- Prob. of a single hypothesis with error $> \epsilon$ is consistent for one random sample is at most $(1-\epsilon)$.
- Prob. of that hypothesis being consistent with m samples is $(1-\epsilon)^m$.
- Prob. of at least one of k hypotheses with error $> \epsilon$ is consistent with m samples is $k(1-\epsilon)^m$.
- Since $k \leq |H|$, and for $0 \leq \epsilon \leq 1$, $(1-\epsilon) \leq e^{-\epsilon}$:

$$k(1-\epsilon)^m \leq |H|(1-\epsilon)^m \leq |H|e^{-\epsilon m}$$

PAC Learning

Consider a class C of possible target concepts defined over a set of instances X of length n, and a learner L using hypothesis space H.

> *Definition:* C is **PAC-learnable** by L using H if for all $c \in C$, distributions $\mathcal{D}$ over X, ϵ such that $0 < \epsilon < 1/2$, and δ such that $0 < \delta < 1/2$,
>
> learner L will with probability at least $(1-\delta)$ output a hypothesis $h \in H$ such that $error_{\mathcal{D}}(h) \leq \epsilon$, in time that is polynomial in $1/\epsilon$, $1/\delta$, n and $size(c)$.

Agnostic Learning

So far, we assumed that $c \in H$. What if it is not the case?

Agnostic learning setting: don't assume $c \in H$

- What do we want then?
  - The hypothesis h that makes fewest errors on training data
- What is sample complexity in this case?

$$m \geq \frac{1}{2\epsilon^2}(\ln |H| + \ln(1/\delta))$$

derived from Hoeffding bounds:

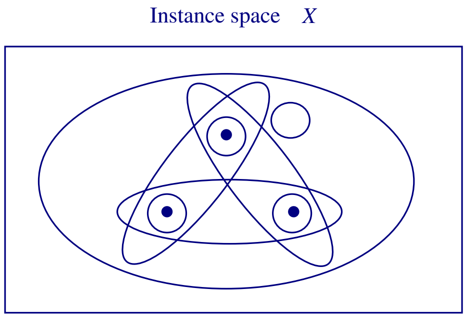
$$Pr[error_{\mathcal{D}}(h) > error_D(h) + \epsilon] \leq e^{-2m\epsilon^2}$$

Shattering a Set of Instances

Definition: a **dichotomy** of a set S is a partition of S into two disjoint subsets.

Definition: a set of instances S is **shattered** by hypothesis space H if and only if for every dichotomy of S there exists some hypothesis in H consistent with this dichotomy.

Three Instances Shattered



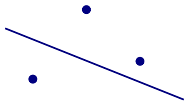
Each closed contour indicates one dichotomy. What kind of hypothesis space H can shatter the instances?

The Vapnik-Chervonenkis Dimension

Definition: The **Vapnik-Chervonenkis dimension**, $VC(H)$, of hypothesis space H defined over instance space X is the size of the largest finite subset of X shattered by H. If arbitrarily large finite sets of X can be shattered by H, then $VC(H) \equiv \infty$.

Note that $|H|$ can be infinite, while $VC(H)$ finite!

VC Dim. of Linear Decision Surfaces



(a)

(b)

- When H is a set of lines, and S a set of points, $VC(H) = 3$.
- (a) can be shattered, but (b) cannot be. However, if at least one subset of size 3 can be shattered, that's fine.
- Set of size 4 cannot be shattered, for any combination of points (think about an XOR-like situation).

VC Dimension: Another Example

$S = \{3.1, 5.7\}$, and hypothesis space includes intervals $a < x < b$.

- Dichotomies: both, none, 3.1, or 5.7.
- Are there intervals that cover all the above dichotomies?

What about $S = x_0, x_1, x_2$ for an arbitrary x_i? (cf. collinear points).

Sample Complexity from VC Dimension

How many randomly drawn examples suffice to ϵ-exhaust $VS_{H,D}$ with probability at least $(1 - \delta)$?

$$m \geq \frac{1}{\epsilon}(4 \log_2(2/\delta) + 8VC(H) \log_2(13/\epsilon))$$

$VC(H)$ is directly related to the sample complexity:

- More expressive H needs more samples.
- More samples needed for H with more tunable parameters.

Mistake Bounds

So far: how many examples needed to learn?

What about: how many mistakes before convergence?

- This is an interesting question because some learning systems may need to start operating while still learning.

Let's consider similar setting to PAC learning:

- Instances drawn at random from X according to distribution $\mathcal{D}$.
- Learner must classify each instance before receiving correct classification from teacher.
- Can we bound the number of mistakes learner makes before converging?

Mistake Bounds: Halving Algorithm

Consider the Halving Algorithm:

- Learn concept using version space *Candidate-Elimination* or *List-Then-Eliminate* algorithm (no need to know details about these algorithms).
- Classify new instances by majority vote of version space members.

How many mistakes before converging to correct h?

- ... in worst case?
- ... in best case?

Mistake Bound of Halving Algorithm

- Start with version space = H.
- Mistake is made when more than half of the $h \in H$ misclassified.
- In that case, at most half of $h \in VS$ will be eliminated.
- That is, each **mistake** reduces the VS by half.
- Initially $|VS| = |H|$, and each mistake halves the VS, so it takes $\log_2 |H|$ mistakes to reduce $|VS|$ to 1.
- Actual worst-case bound is $\lfloor \log_2 |H| \rfloor$.

Optimal Mistake Bounds

Let $M_A(C)$ be the max number of mistakes made by algorithm A to learn concepts in C. (maximum over all possible $c \in C$, and all possible training sequences)

$$M_A(C) \equiv \max_{c \in C} M_A(c)$$

Definition: Let C be an arbitrary non-empty concept class. The **optimal mistake bound** for C, denoted $Opt(C)$, is the minimum over all possible learning algorithms A of $M_A(C)$.

$$Opt(C) \equiv \min_{A \in learning\ algorithms} M_A(C)$$

Mistake Bounds and VC Dimension

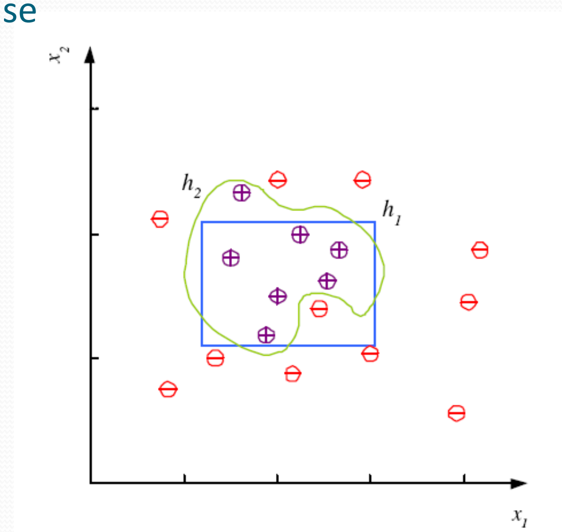
Littlestone (1987) showed:

$$VC(C) \leq Opt(C) \leq M_{Halving}(C) \leq \log_2(|C|)$$

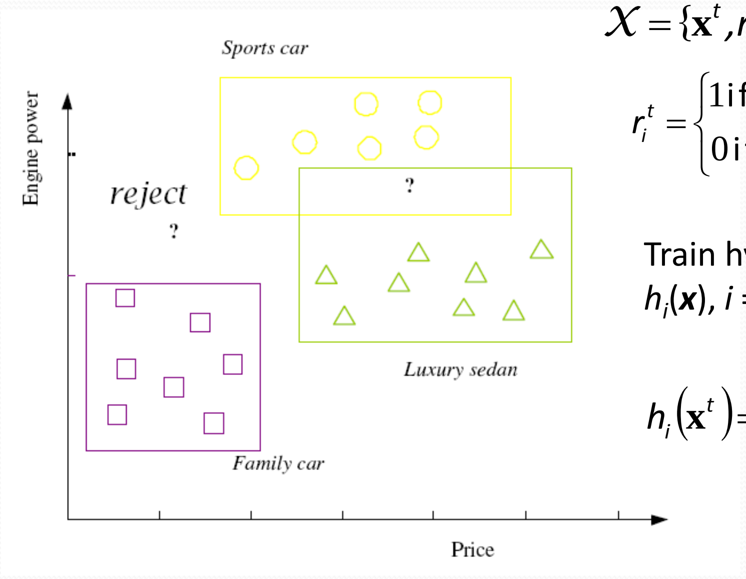
Noise and Model Complexity

Use the simpler one because

- Simpler to use (lower computational complexity)
- Easier to train (lower space complexity)
- Easier to explain (more interpretable)
- Generalizes better (lower variance - Occam's razor)



Multiple Classes, C_i i=1,...,K



$$\mathcal{X} = \{\mathbf{x}^t, r^t\}_{t=1}^N$$

$$r_i^t = \begin{cases} 1 \text{ if } \mathbf{x}^t \in C_i \\ 0 \text{ if } \mathbf{x}^t \in C_j, j \neq i \end{cases}$$

Train hypotheses $h_i(\mathbf{x})$, i =1,...,K:

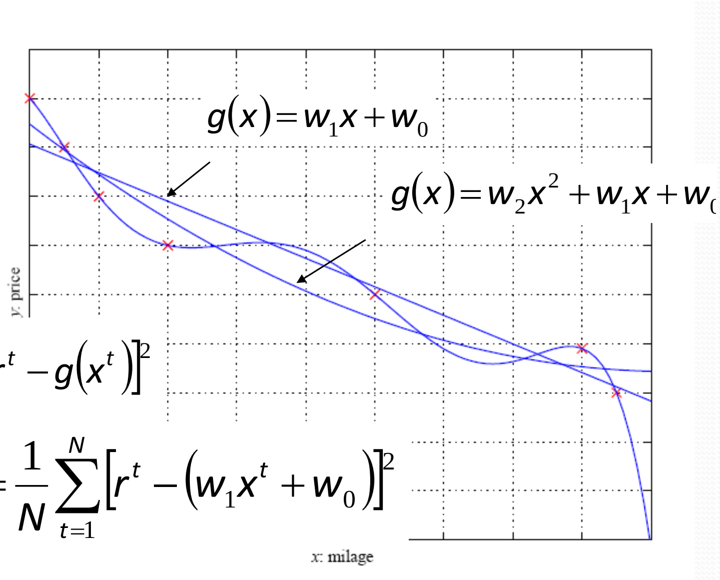
$$h_i(\mathbf{x}^t) = \begin{cases} 1 \text{ if } \mathbf{x}^t \in C_i \\ 0 \text{ if } \mathbf{x}^t \in C_j, j \neq \end{cases}$$

Regression

$$\mathcal{X} = \{x^t, r^t\}_{t=1}^N$$

$$r^t \in \Re$$

$$r^t = f(x^t) + \varepsilon$$



$$E(g \mid \mathcal{X}) = \frac{1}{N} \sum_{t=1}^N \left[r^t - g(x^t)\right]^2$$

$$E(w_1, w_0 \mid \mathcal{X}) = \frac{1}{N} \sum_{t=1}^N \left[r^t - (w_1 x^t + w_0)\right]^2$$

Model Selection & Generalization

- Learning is an ill-posed problem; data is not sufficient to find a unique solution
- The need for inductive bias, assumptions about $\mathcal{H}$
- Generalization: How well a model performs on new data
- Overfitting: $\mathcal{H}$ more complex than C or f
- Underfitting: $\mathcal{H}$ less complex than C or f

Triple Trade-Off

- There is a trade-off between three factors (Dietterich, 2003):
  1. Complexity of $\mathcal{H}$, $c(\mathcal{H})$,
  2. Training set size, N,
  3. Generalization error, E, on new data
- As $N\uparrow$, $E\downarrow$
- As $c(\mathcal{H})\uparrow$, first $E\downarrow$ and then $E\uparrow$

Cross-Validation

- To estimate generalization error, we need data unseen during training. We split the data as
  - Training set (50%)
  - Validation set (25%)
  - Test (publication) set (25%)
- Resampling when there is few data

Dimensions of a Supervised Learner

1. Model: $g(\mathbf{x}|\theta)$
2. Loss function: $$E(\theta|\mathcal{X}) = \sum_t L\left(r^t, g\left(\mathbf{x}^t|\theta\right)\right)$$
3. Optimization procedure:

$$\theta^* = \arg\min_\theta E(\theta|\mathcal{X})$$