# Steps in developing a machine learning application

- Collect Data
  - Collect samples
- Prepare the input data
  - Make sure the data is in useable format
- Analyze the input data
  - Recognize patterns or similarity

# Steps in developing a machine learning application

- Train the algorithm
  - This is where machine learning takes place
  - For supervised learning
    - Feed the algorithm good clean data form the first two steps
    - Extract knowledge and information.
  - For unsupervised learning
    - There’s no training step.

# Steps in developing a machine learning application

- Test the algorithm
  - Test how well it does or how successful it is
- Use it

# Key tasks of machine learning

- Classification
- Regression
- Clustering
- Density estimation

# Modelling Terminology

- Features or attributes – input representation
- Instance or sample or training set
- Training examples(positive and negative examples)
- Test set
- Hypothesis class and Hypothesis

# 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:

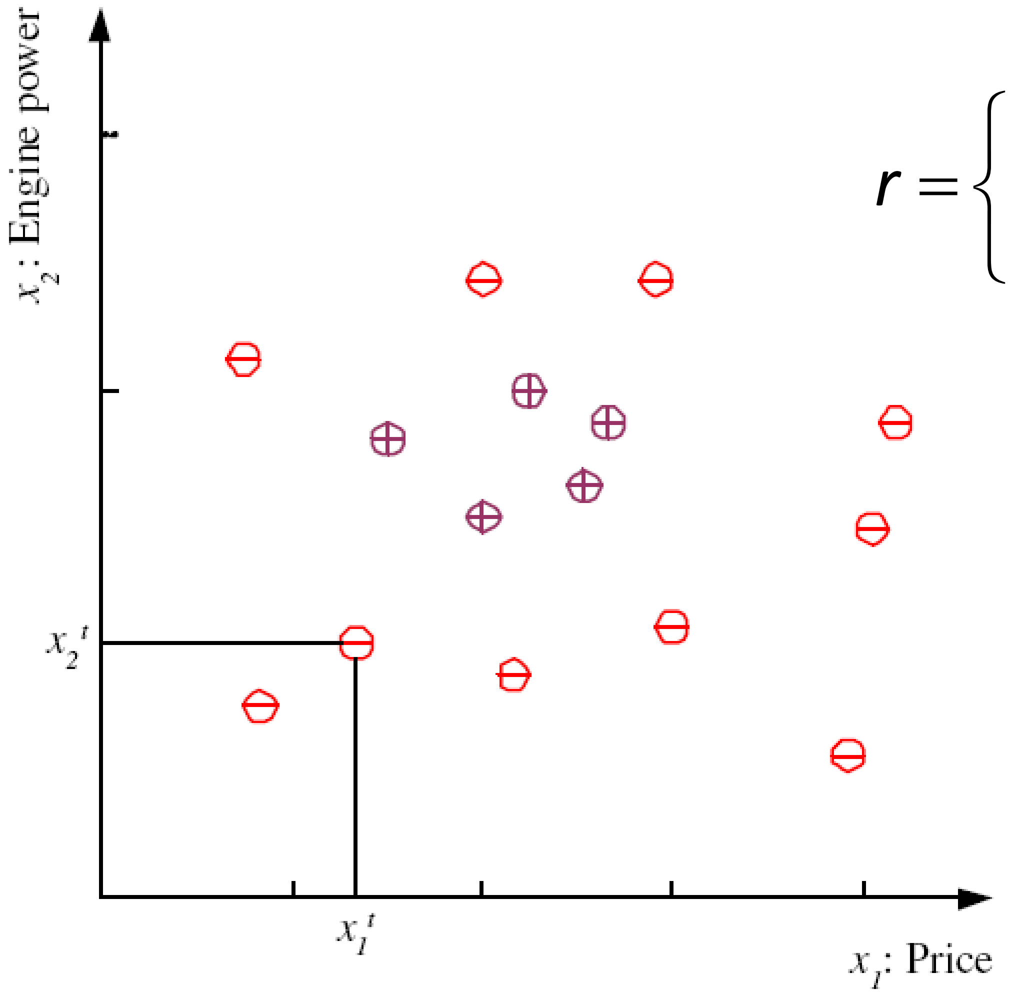
  Positive (+) and negative (–) examples
- Input representation:

  $x_1$: price, $x_2$ : engine power

# Training set $\mathcal{X}$

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

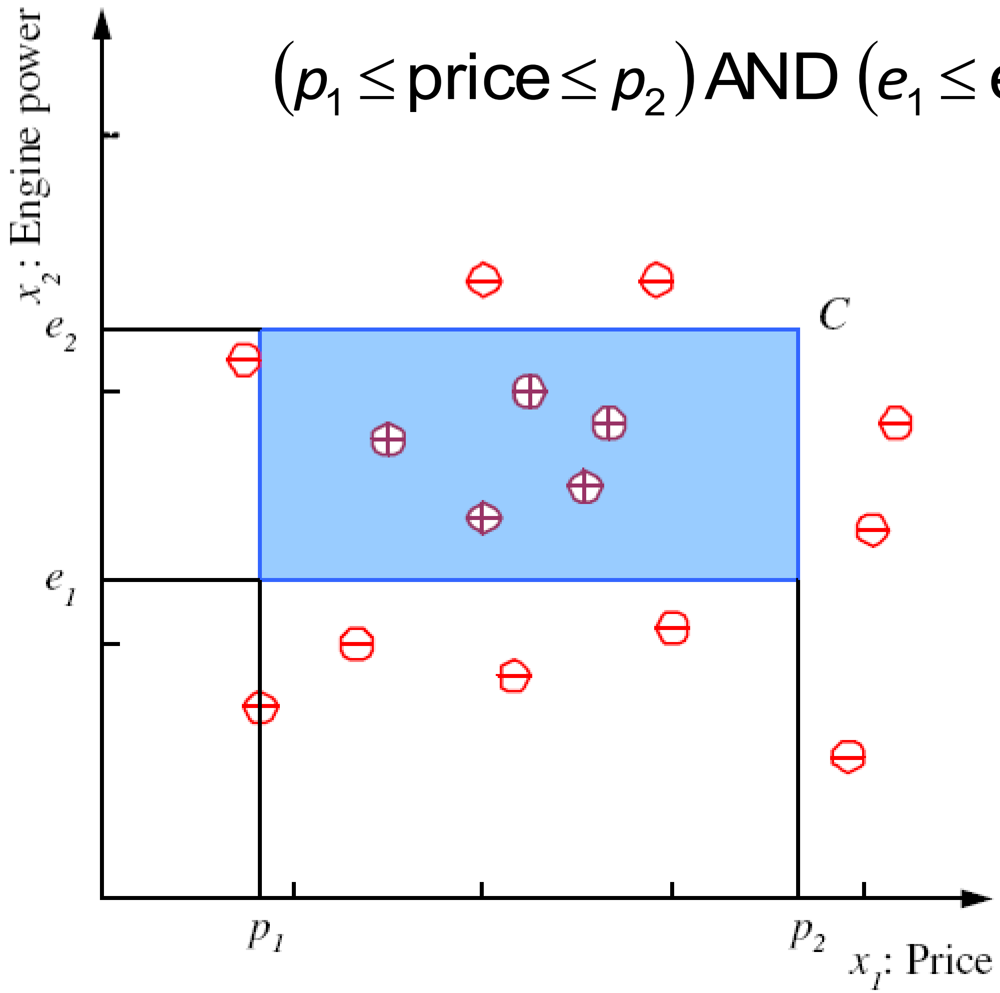
$$r = \begin{cases} 1 \text{ if } \mathbf{x} \text{ is positive} \\ 0 \text{ if } \mathbf{x} \text{ is negative} \end{cases}$$




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

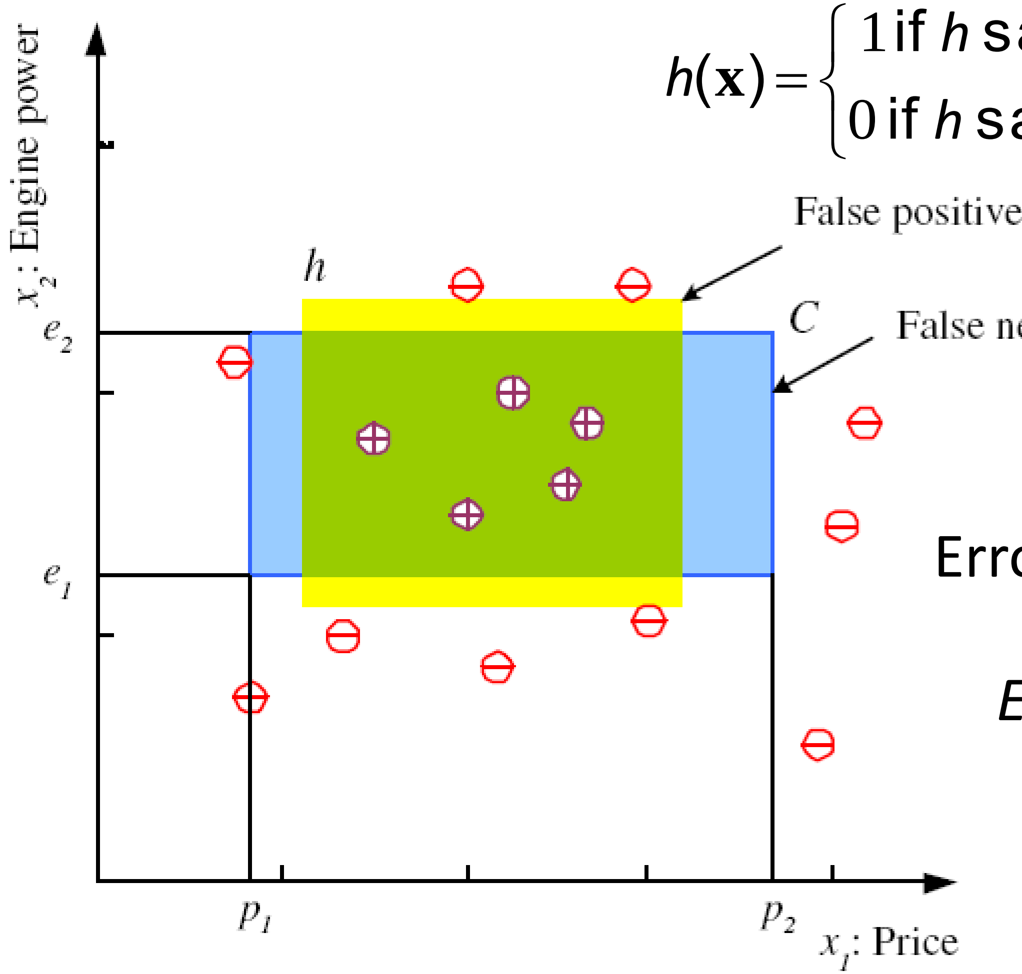
# Class $C$

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



# Hypothesis class $\mathcal{H}$

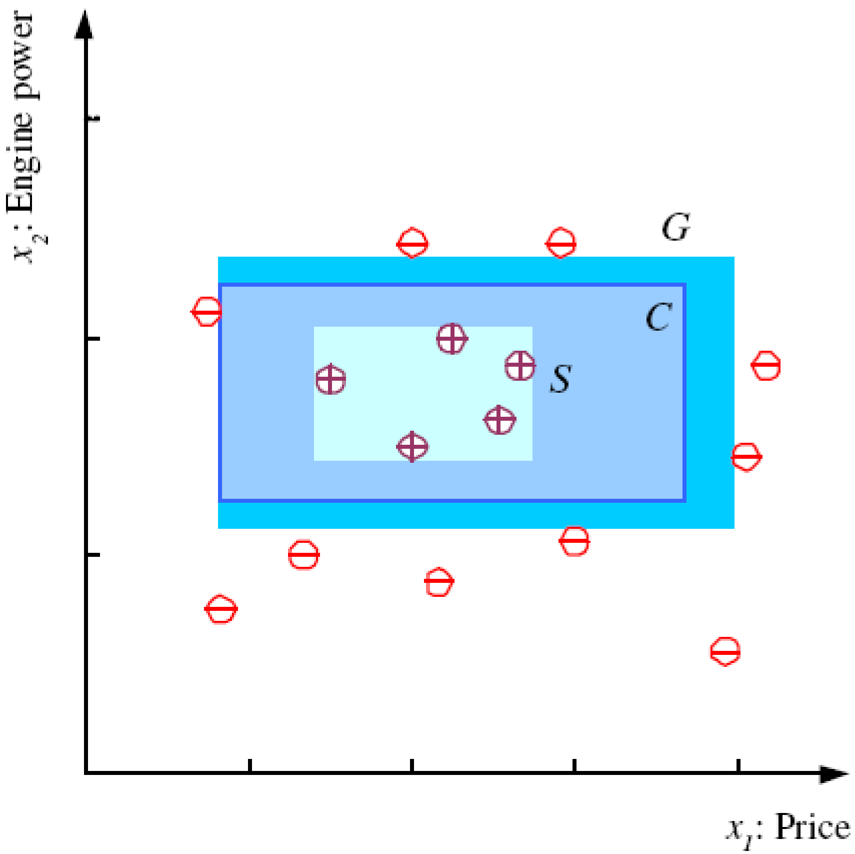
$$h(\mathbf{x}) = \begin{cases} 1 \text{ if } h \text{ says } \mathbf{x} \text{ is positive} \\ 0 \text{ if } h \text{ says } \mathbf{x} \text{ is negative} \end{cases}$$




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

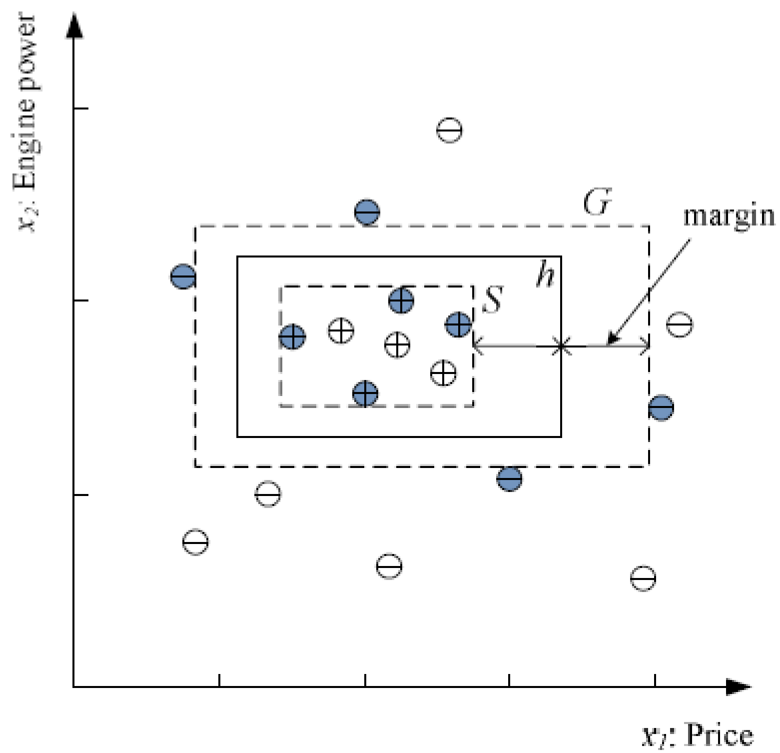



most general hypothesis, *G*

*h* ∈ H, between *S* and *G* is consistent
and make up the
version space
(Mitchell, 1997)

# Margin

- Choose *h* with largest margin



# Probability and Inference

- Result of tossing a coin is $\in$ {Heads,Tails}
- Random var $X \in \{1,0\}$

  Bernoulli: $P\{X=1\} = p_o^X (1 - p_o)^{(1-X)}$
- Sample: $\boldsymbol{X} = \{x^t\}_{t=1}^N$

  Estimation: $p_o = \#\{\text{Heads}\}/\#\{\text{Tosses}\} = \sum_t x^t / N$
- Prediction of next toss:

  Heads if $p_o > ½$, Tails otherwise

# Probably Approximately Correct (PAC) Learning

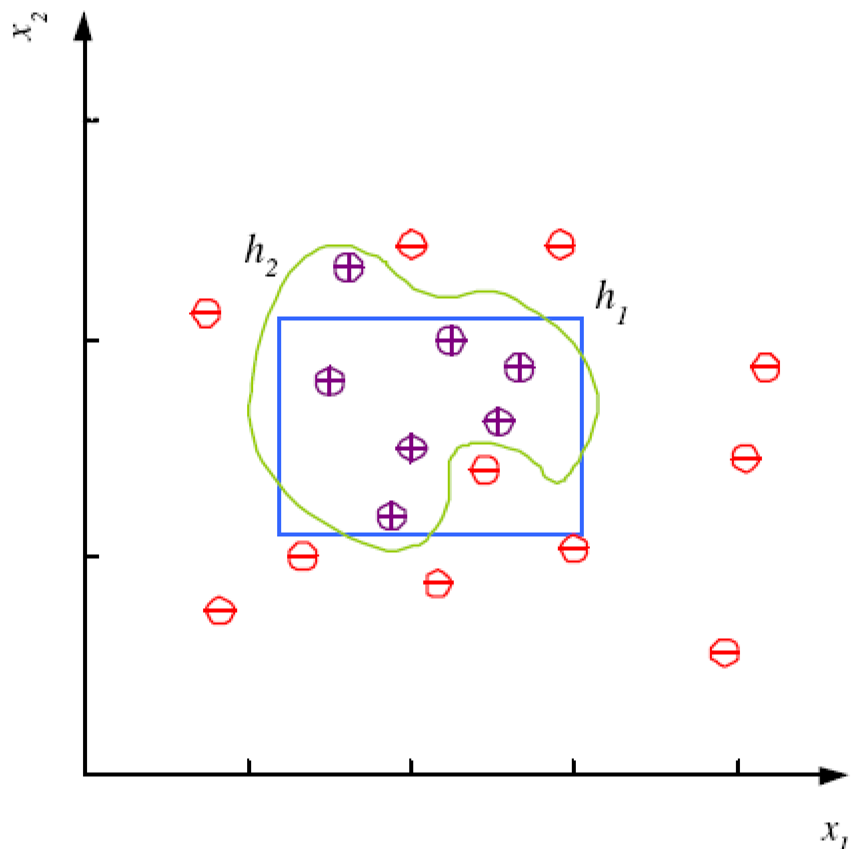
- Given
  - Class C
  - Examples drawn from some unknown but fixed probability distribution p(x)
- To find
  - The number of examples N such that with probability at least $1 - \delta$, the hypothesis *h* has error at most $\varepsilon$, for arbitrary $0 < \delta <= ½$.
- $1 - \delta$ -> confidence probability and $\varepsilon$ -> error probability.

# Noise

- Noise is any unwanted anomaly in the data.
- Due to this,
  - class may be more difficult to learn
  - Zero error may be infeasible with a simple hypothesis class.
- Reasons for the above
  - Imprecision in recording the input attributes
  - Error in labelling the data points – teacher niose
  - Hidden or latent attributes that may be un observeable.

# Noise and model complexity

- There is no simple boundary between +ve and –ve instances.
- Simple hypothesis is a rectangle defining the corners.
- But if we take this hypothesis we will not get zero misclassification error. For this some error should be allowed.
- Another option of hypothesis is the arbitrary closed form.



# Noise and model complexity

- Simpler hypothesis is better 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 – Simpler explanations are more plausible and any unnecessary complexity should be shaved off.

# Modelling Terminology ctd.

- Ill-posed problem : Data is not sufficient to find a unique solution
- Inductive bias : Extra assumptions that we take to have a unique solution with the data we have.
- Model selection : Choosing the right bias or in other words choosing between different $\mathcal{H}$
- Generalization: How well a model performs on new data

# Modelling Terminology ctd.

- For best generalization, match complexity of $\mathcal{H}$ with the complexity of the function $f$ underlying the 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 $E \uparrow$

# Association Rules

- Association rule: $X \rightarrow Y$
- *People who buy/click/visit/enjoy X are also likely to buy/click/visit/enjoy Y.*
- A rule implies association, not necessarily causation.

# Association measures

- Support ($X \rightarrow Y$):

$$P(X,Y) = \frac{\#\{\text{customers who bought } X \text{ and } Y\}}{\#\{\text{customers}\}}$$

- Confidence ($X \rightarrow Y$):

$$P(Y \mid X) = \frac{P(X,Y)}{P(X)}$$

$$= \frac{\#\{\text{customers who bought } X \text{ and } Y\}}{\#\{\text{customers who bought } X\}}$$

# Association measures

- Lift ($X \rightarrow Y$):

$$= \frac{P(X,Y)}{P(X)P(Y)} = \frac{P(Y \mid X)}{P(Y)}$$

# Apriori algorithm (Agrawal et al., 1996)

- For (X,Y,Z), a 3-item set, to be frequent (have enough support), (X,Y), (X,Z), and (Y,Z) should be frequent.
- If (X,Y) is not frequent, none of its supersets can be frequent.
- Once we find the frequent *k*-item sets, we convert them to rules: X, Y $\rightarrow$ Z, ...

  and X $\rightarrow$ Y, Z, ...

# Sample algorithms

| Supervised learning tasks | |
|---|---|
| k-Nearest Neighbors | Linear |
| Naive Bayes | Locally weighted linear |
| Support vector machines | Ridge |
| Decision trees | Lasso |
| **Unsupervised learning tasks** | |
| k-Means | Expectation maximization |
| DBSCAN | Parzen window |

**Table 1.2 Common algorithms used to perform classification, regression, clustering, and density estimation tasks**