CHAPTER 2:

Supervised Learning

Learning a Class from Examples

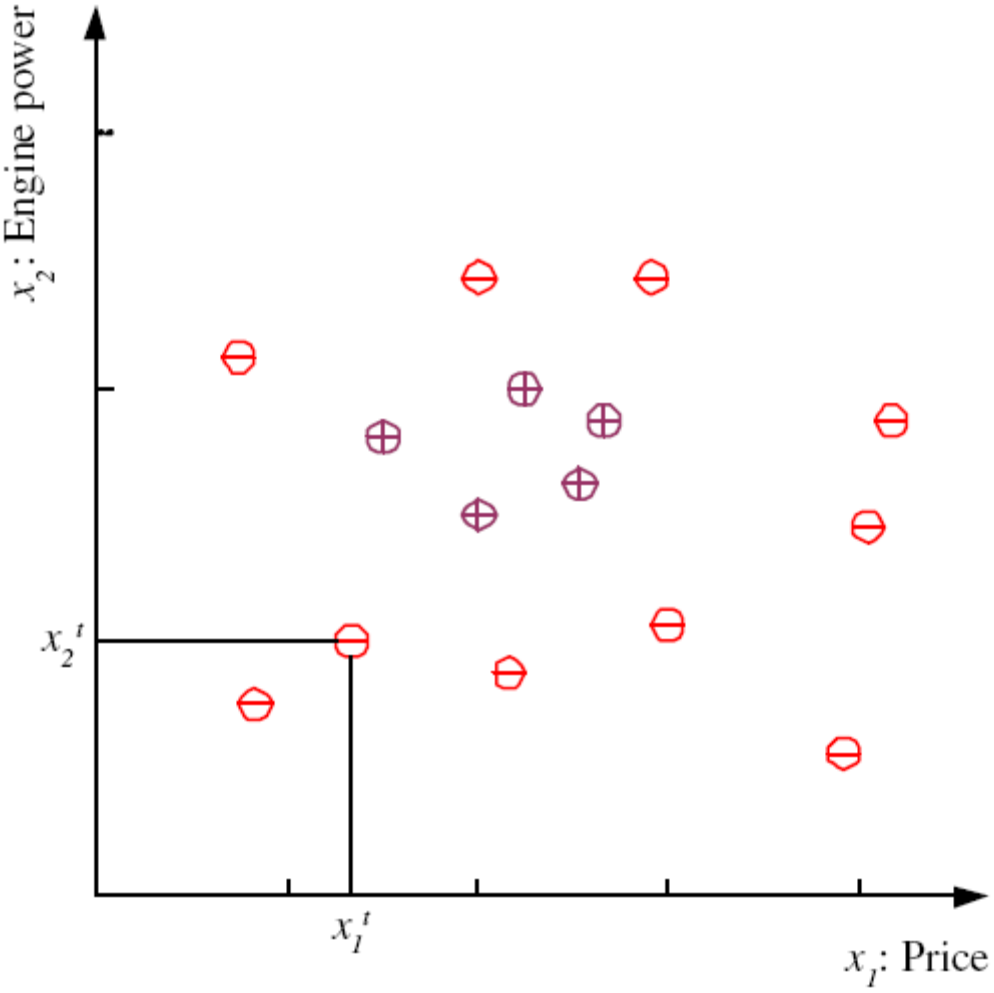
- Class C of a "family car"
  - Prediction: Is car *x* a family car?
  - Knowledge extraction: What do people expect from a family car?
- Output:

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

  x_1: price, x_2 : engine power

Training set $\mathcal{X}$

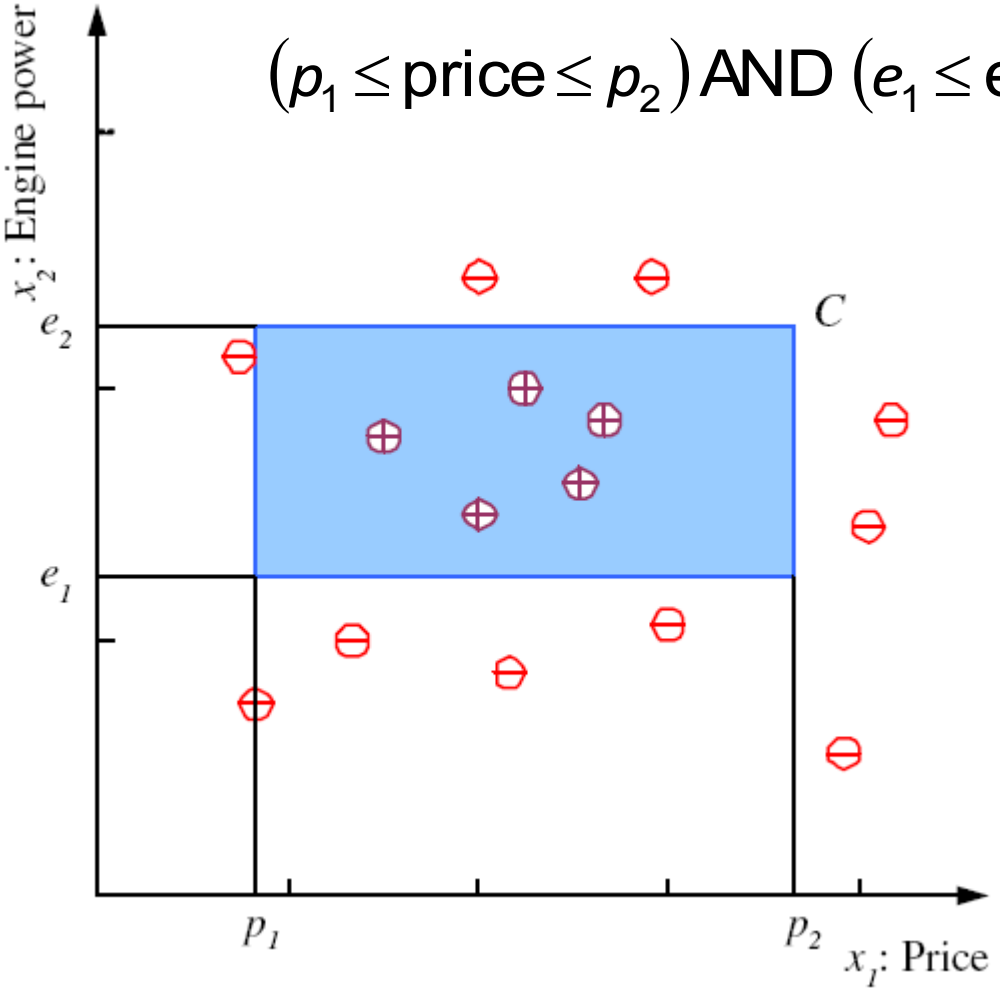
$$\mathcal{X} = \{\mathbf{x}^t, r^t\}_{t=1}^N$$



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

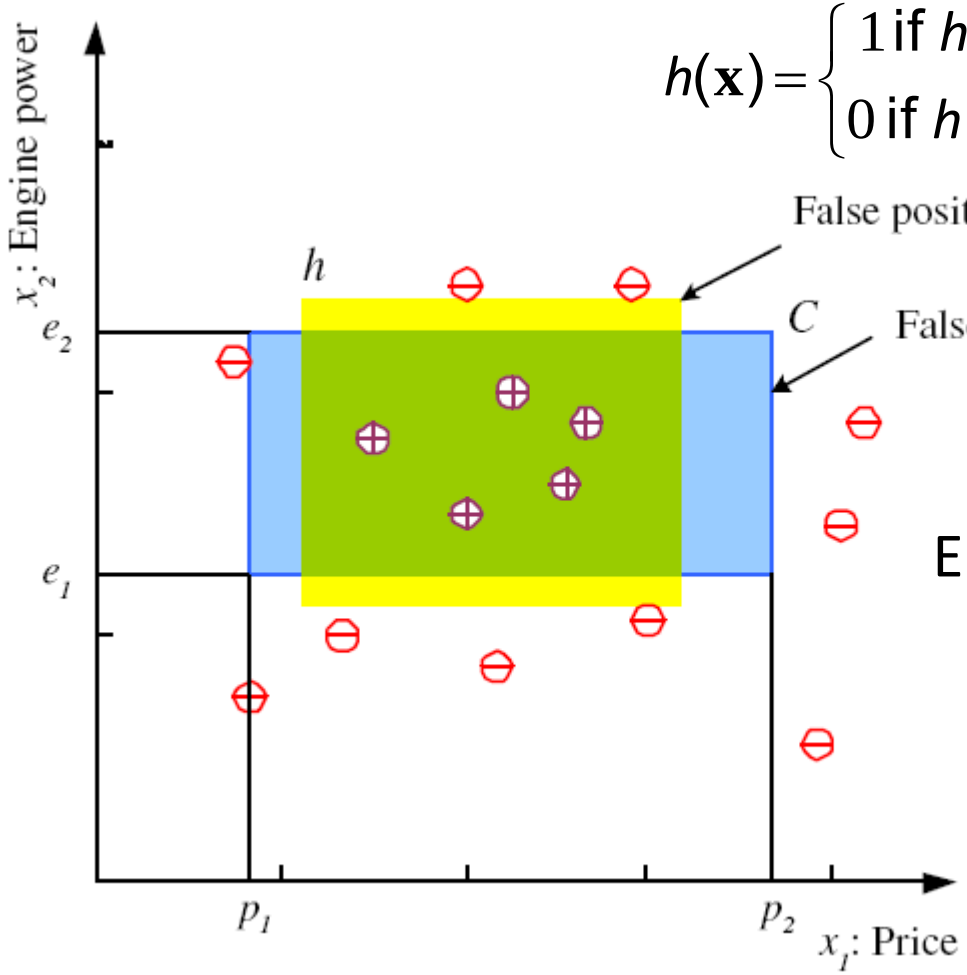
Class C

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



Hypothesis class $\mathcal{H}$

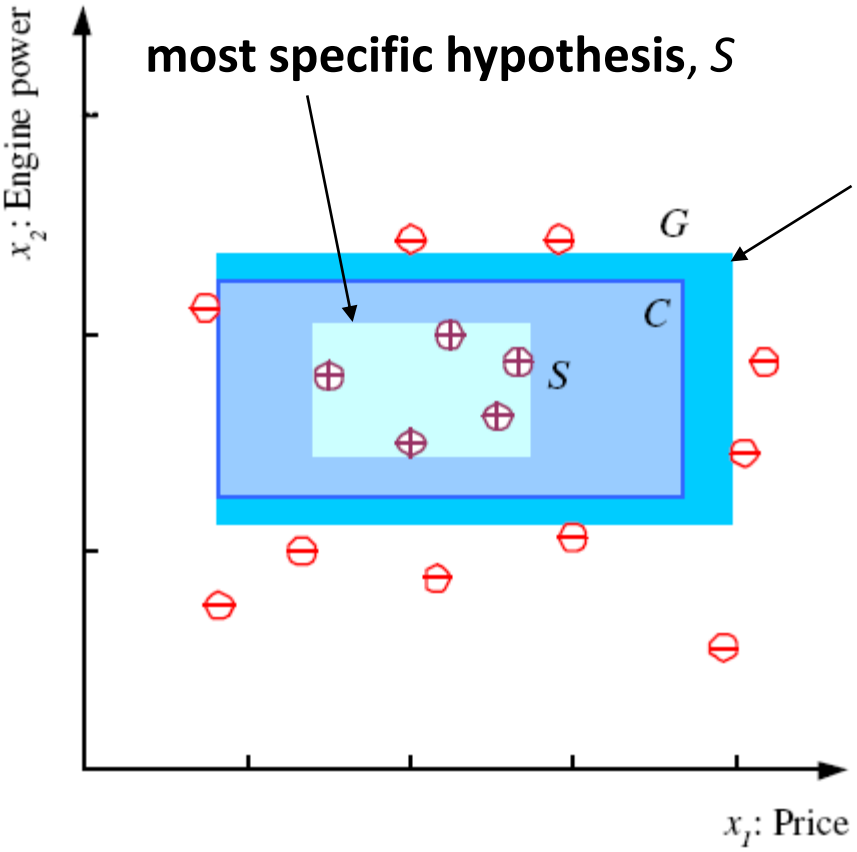
$$h(\mathbf{x}) = \begin{cases} 1 \text{ if } h \text{ says } \mathbf{x} \text{ is positive} \\ 0 \text{ if } h \text{ says } \mathbf{x} \text{ is negative} \end{cases}$$



Error of h on $\mathcal{H}$

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

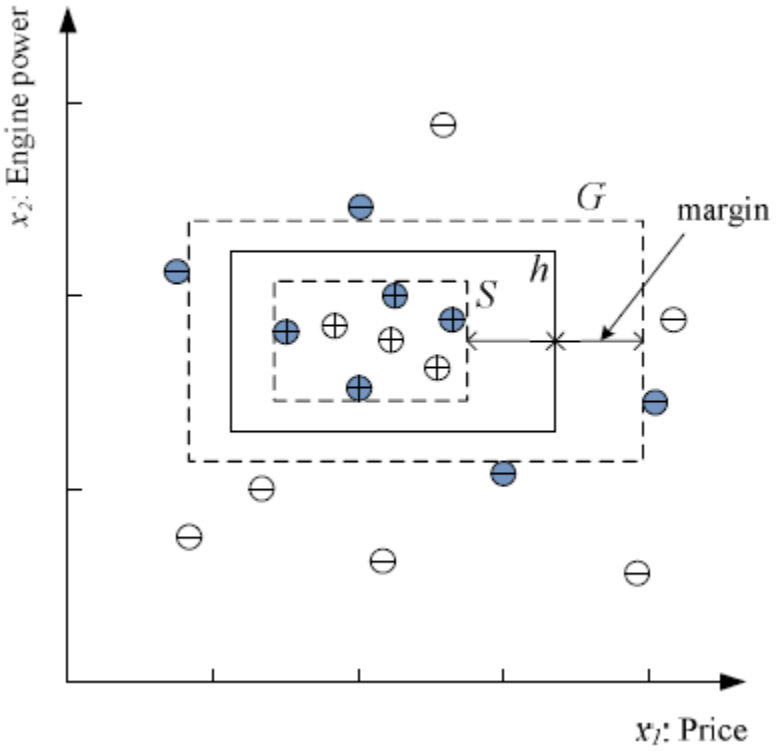
S, G, and the Version Space



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

Margin

- Choose *h* with largest margin (the distance between the boundary and closest instance)



Candidate Elimination Algorithm

Initialize the sets S and G, respectively, to the sets of maximally specific and maximally general generalizations that are consistent with the first observed positive training example;

for each *subsequent example ei*

if *ei is a negative example*

- ***retain*** *in S only those generalizations which do not match ei;*
- ***specialize*** *the members of G that match ei, only to the extent required so that they no longer match ei, and only in such ways that each remains more general than some generalization in S;*
- ***remove*** *from G any element that is more specific than some in G*

else if *ei is a positive example*

- ***retain*** *in G only those generalizations that match ei;*
- ***generalize*** *members of S that do not match ei, only to the extent required to allow them to match ei, and only in such ways that each remains more specific than some generalization in G;*
- ***remove*** *from S any element that is more general than some in S*

end

Example: Candidate elimination

- Example

Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

Example: Candidate Elimination

Consider a concept described by three attributes predefined as follows:

sky		temperature		humidity
Sunny	Rainy	Warm	Coo	Normal	Low

and the set of positive and negative training examples:

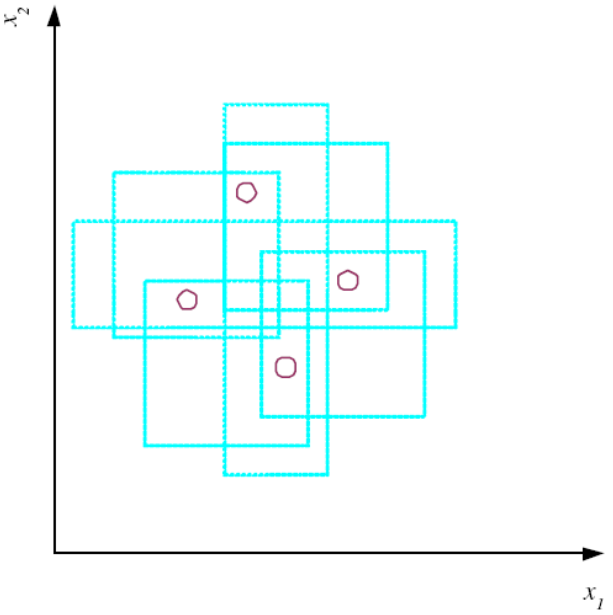
1. (S W N)+)
2. (R C L)-)
3. (S C N)+)
4. (S W L)-)

Example: candidate Elimination

Size	Color	Shape	Class
Big	Red	Circle	-ve
Small	Red	Triangle	-ve
Small	Red	Circle	+v
Big	Blue	Circle	+ve
Small	Blue	Circle	+ve

VC (*Vapnik-Chervonenkis*) Dimension

- N points can be labeled in 2^N ways as +/–
- $\mathcal{H}$ **shatters** N if there exists $h \in \mathcal{H}$ consistent for any of these: $VC(\mathcal{H}) = N$



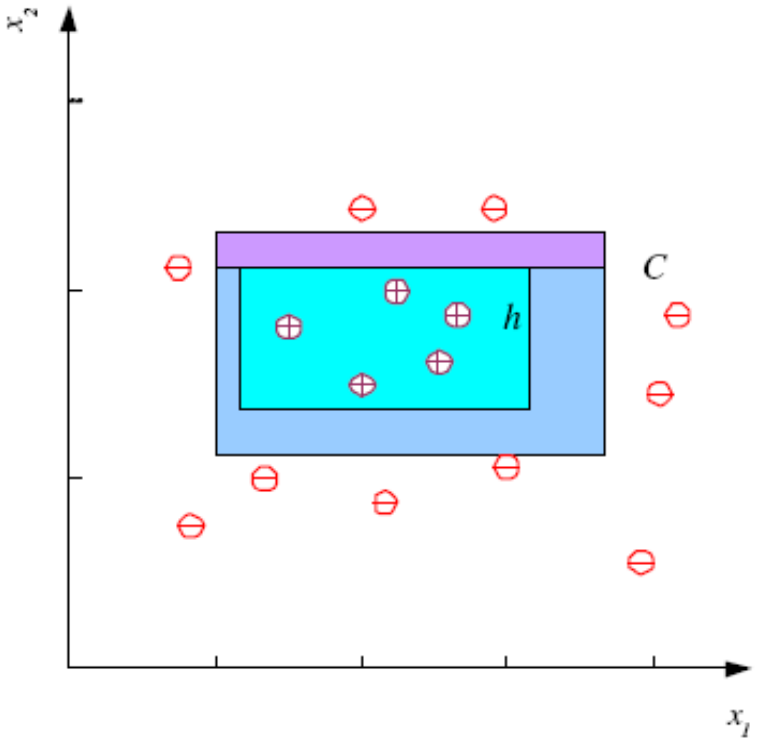
  - *An axis-aligned rectangle shatters 4 points only !*
  - *What about if the points are along one axis ?*
  - *What about 5 points ?*

Probably Approximately Correct (PAC) Learning

- How many training examples N should we have, such that with probability at least $1 - \delta$, h has error (probablity) at most ε ?

  (Blumer et al., 1989)

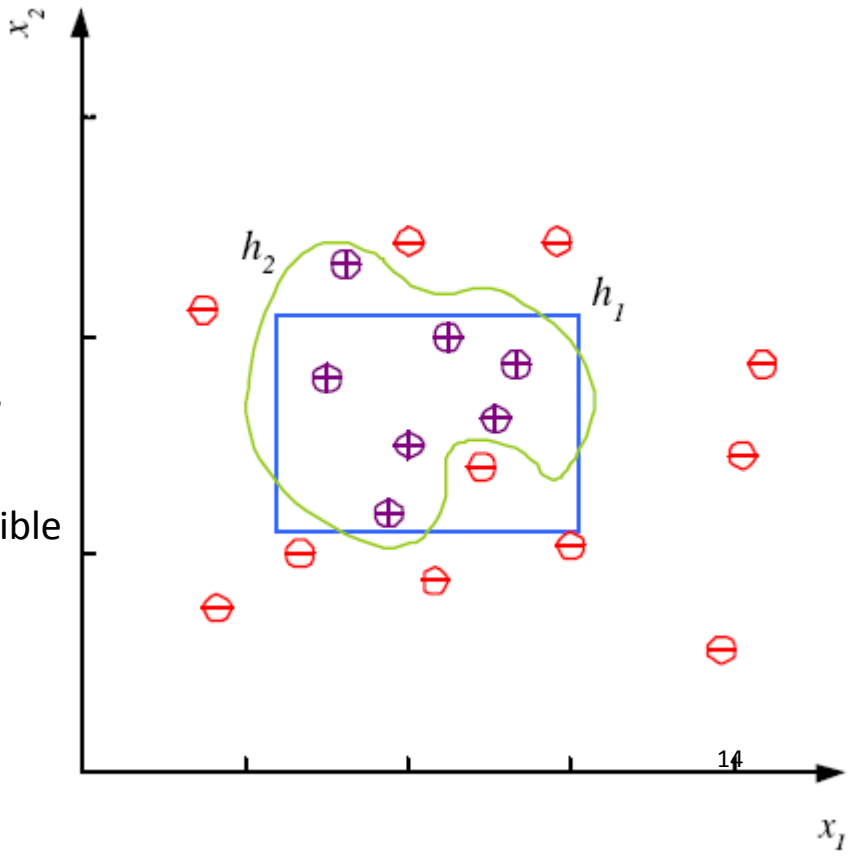
- Each strip is at most $\varepsilon/4$
- Pr that we miss a strip $1 - \varepsilon/4$
- Pr that N instances miss a strip $(1 - \varepsilon/4)^N$
- Pr that N instances miss 4 strips $4(1 - \varepsilon/4)^N$
- $4(1 - \varepsilon/4)^N \leq \delta$ and $(1 - x) \leq \exp(-x)$
- $4\exp(-\varepsilon N/4) \leq \delta$ and $N \geq (4/\varepsilon)\log(4/\delta)$



Noise and Model Complexity

Select simpler model 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)
  - Occam's razor: simpler explanations are more plausible and unnecessary complexity should be shaved off



Learning Multiple Classes

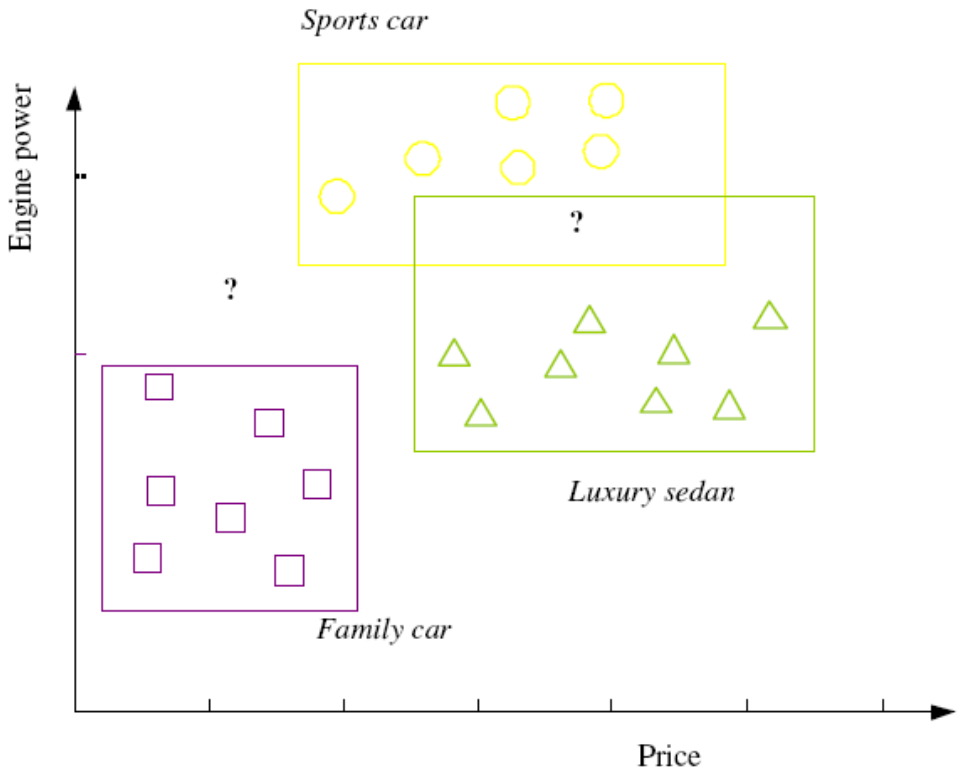
Data: $\mathcal{X}=\{\mathbf{x}^t,r^t\}_{t=1}^N$

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

- *Model it as a K two-class problem*
  - *Results into k hypothesis*

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

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

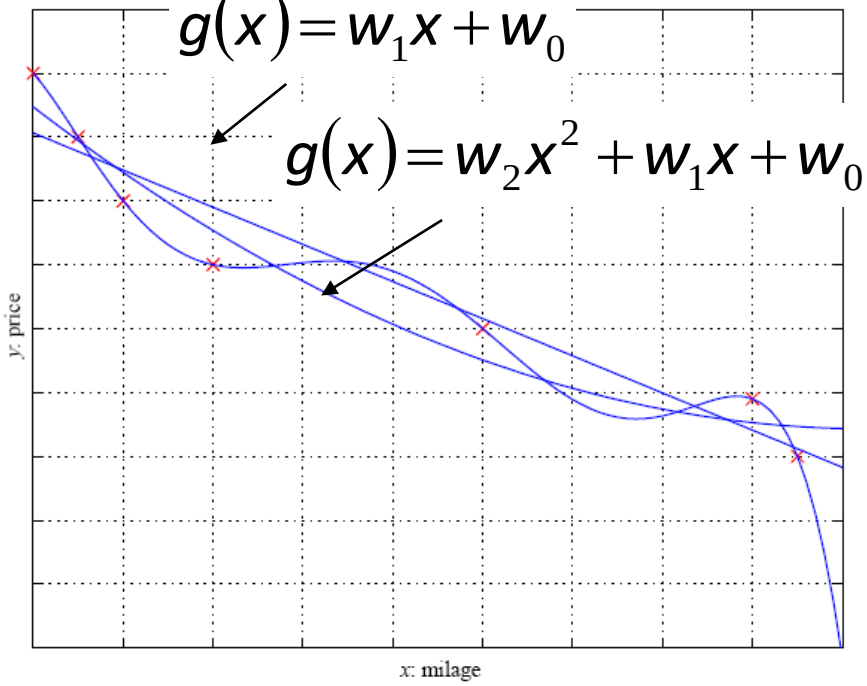


Regression

Data: $\mathcal{X} = \left\{x^t, r^t\right\}_{t=1}^{N}$

$r^t \in \Re$

- *Actual fn* $r^t = f\left(x^t\right) + \varepsilon$



- *Loss function*

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

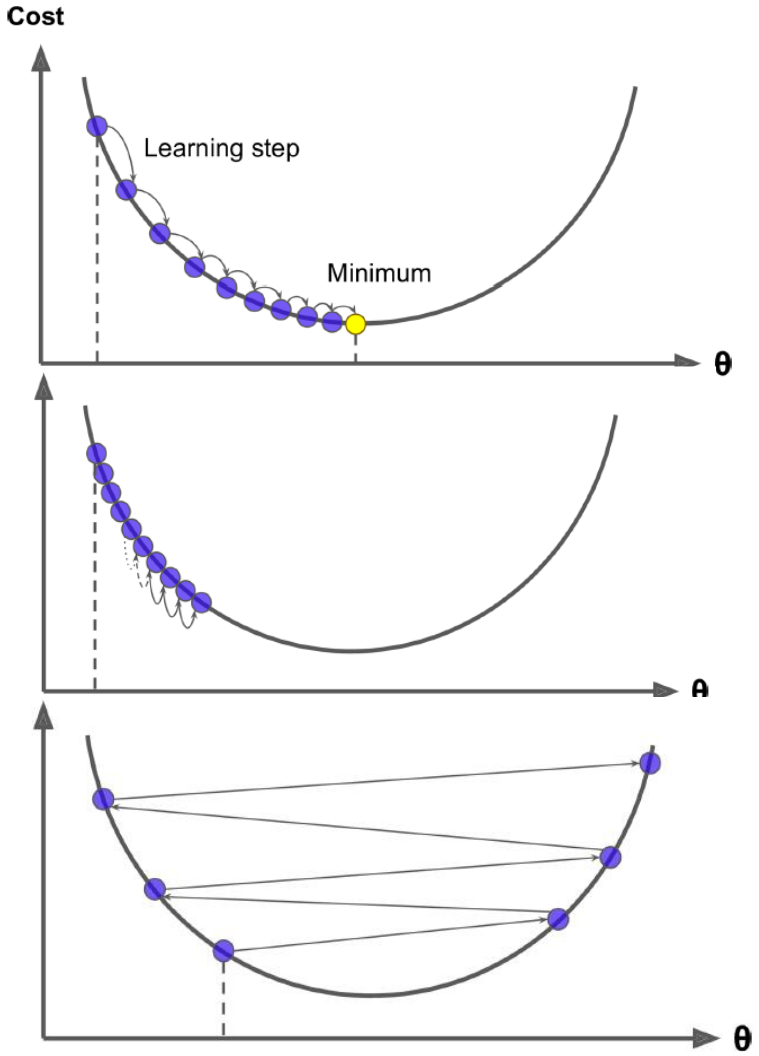
- *Ex:* $E(w_1, w_0 \mid \mathcal{X}) = \frac{1}{N}\sum_{t=1}^{N}\left[r^t - \left(w_1 x^t + w_0\right)\right]^2$

Solution to regression problem

- Close form solution: $W = (X^T X)^{-1}. X^T y$
  - Worst case time complexity: $O(n^3)$
    - n: number of features
- Gradient Descent: Tweak parameters interatively to minimize a cost function
  - Batch Gradient Descent:
  - Stochastic Gradient Descent
    - ***Programming Note:*** `SGDRegressor` in `sklearn.linear_model`
  - **Programming note:**

Gradient Descent

- Tweak parameters
  - To optimize a cost function

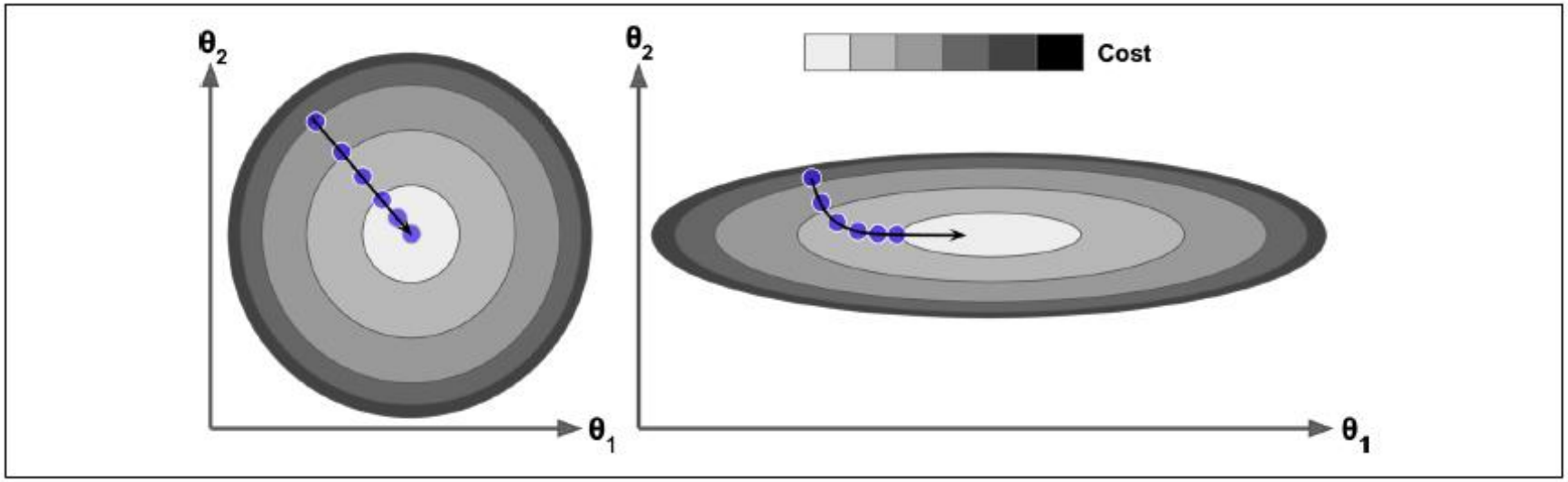


- Learning rate too small

- Learning rate too large

Pitfalls ?

Gradient Descent: Impact of scaling



With Scaling

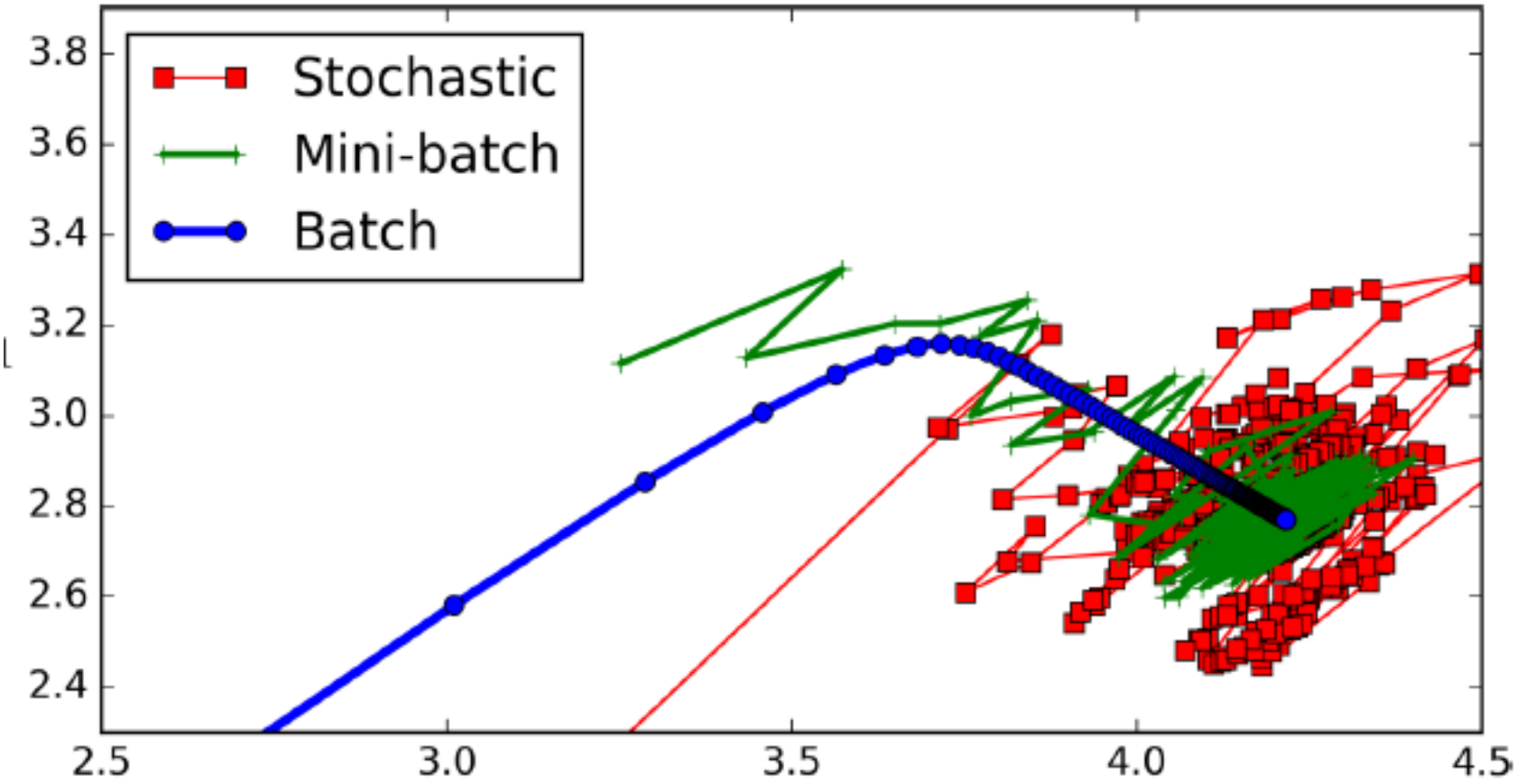
Without Scaling

Practice: Ensure all parameters have similar scale

Gradient Descent: Batch, Stochastic and Mini batch

- Batch: Update all by using matrix operation
  - $\nabla_W(E(W)) = \begin{pmatrix} \frac{\partial}{\partial w_0} E(W) \\ \frac{\partial}{\partial w_2} E(W) \\ . \\ . \\ \frac{\partial}{\partial w_N} E(W) \end{pmatrix} = \frac{2}{N} X^T.(X.W - r)$
- Gradient Descent step $W^{next} = W^{current} - \eta \nabla_W(E(W))$
  - η: Learning rate
- Stochastic: Pick a random instance and compute gradient on that single instance
- Mini Batch: Pick a random set of instances and computer gradient on that set

Batch vs Gradient vs Mini



Polynomial regression

- Add powers of each features and then train a linear model
- **Programming Note:** Use Scikitlearn's `PolynomialFeatures` to add powers of features, then use `LinearRegression()`

Model Selection & Generalization

- 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):
  - Complexity of $\mathcal{H}$, $c(\mathcal{H})$,
  - Training set size, N,
  - Generalization error, E, on new data
  - As N , $E\downarrow$
  - As $c(\mathcal{H})$, first $E\downarrow$ and then E

Cross-Validation: General Practice

- To estimate generalization error, we need data unseen during training. We split the data as
  - Training set (50%)
  - Validation set (25%)
    - Used to select the best model
  - Test (publication) set (25%)
    - Test the best model on this set
- Fit the best hypothesis, then select the one that fits most accurately (on the validation set).
- Resampling when there is few data