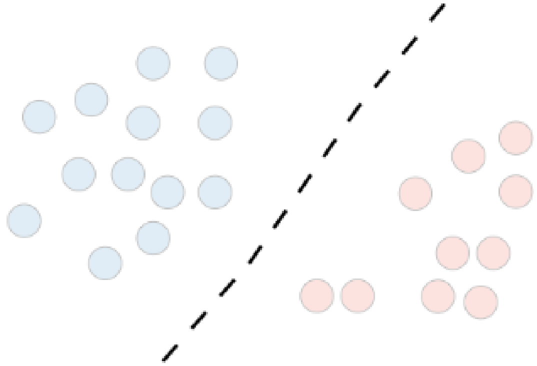
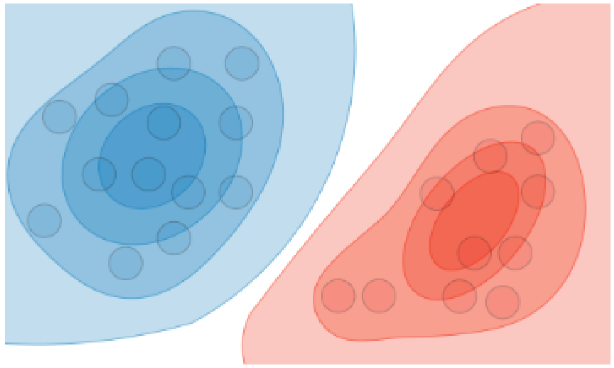
Classifiers

Classification – Lets define the task!

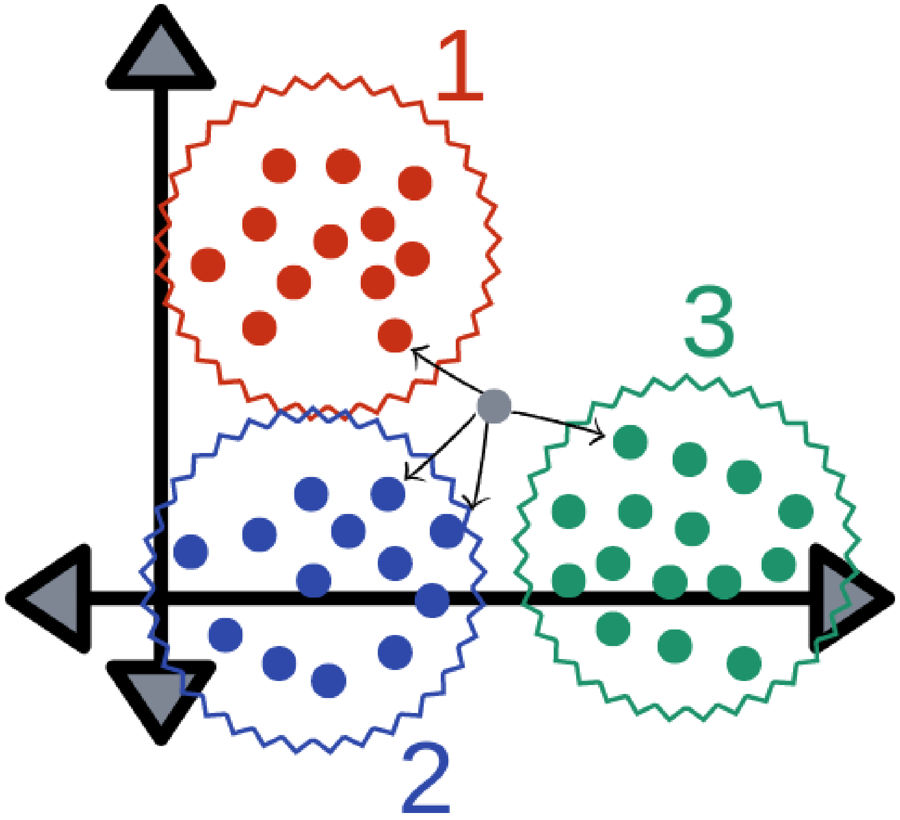
Classification – Lets define the task!

- Features
- Classification Function
- Loss
- Update weights

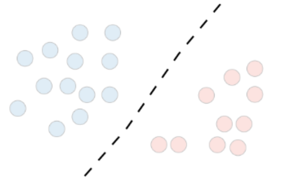
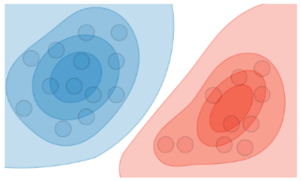
Classification – Lets define the task!

	Discriminative model	Generative model
Goal	Directly estimate $P(y\|x)$	Estimate $P(x\|y)$ to then deduce $P(y\|x)$
What's learned	Decision boundary	Probability distributions of the data
Illustration		
Examples	Regressions, SVMs	GDA, Naive Bayes

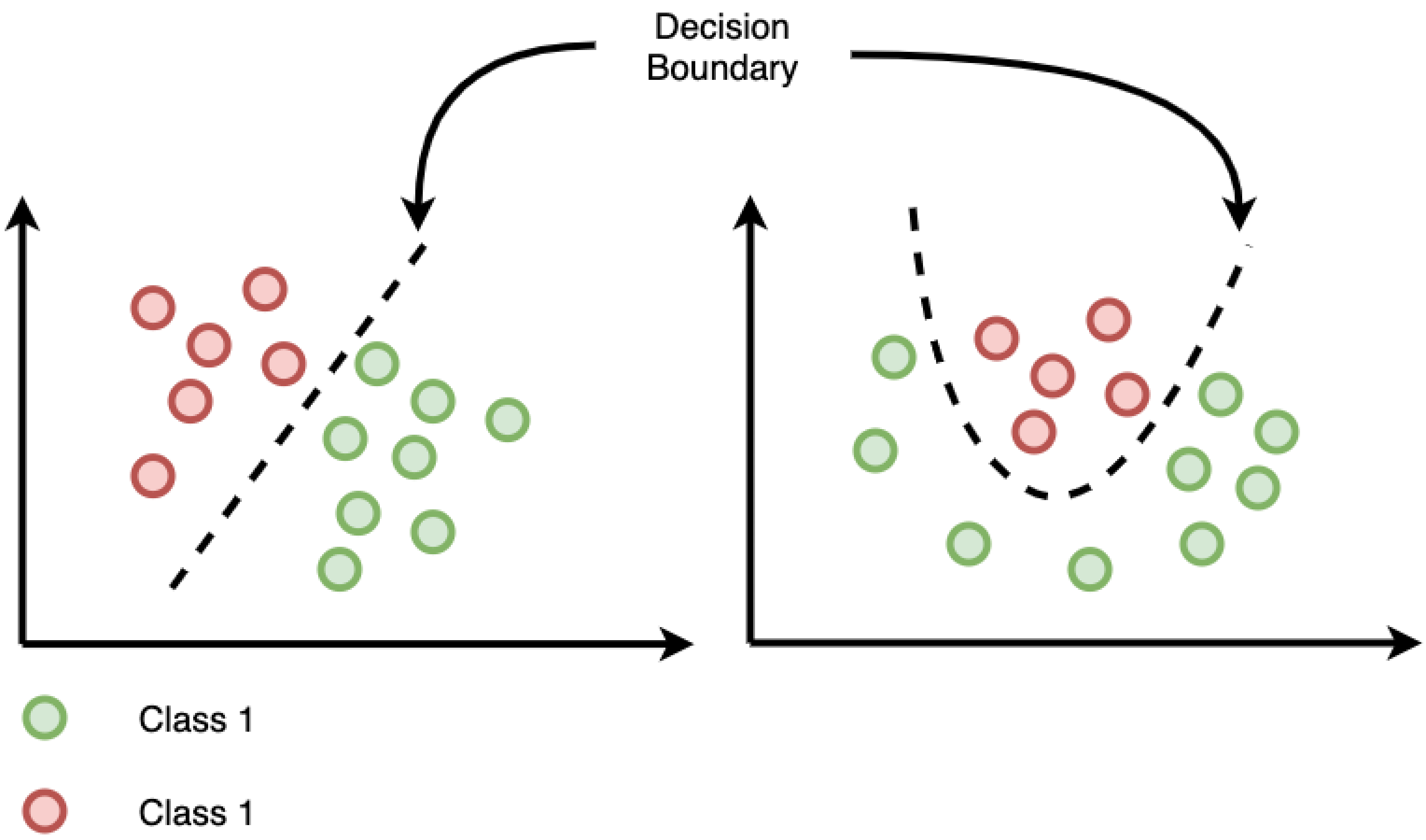
Classification – But I hate weights and parameters?



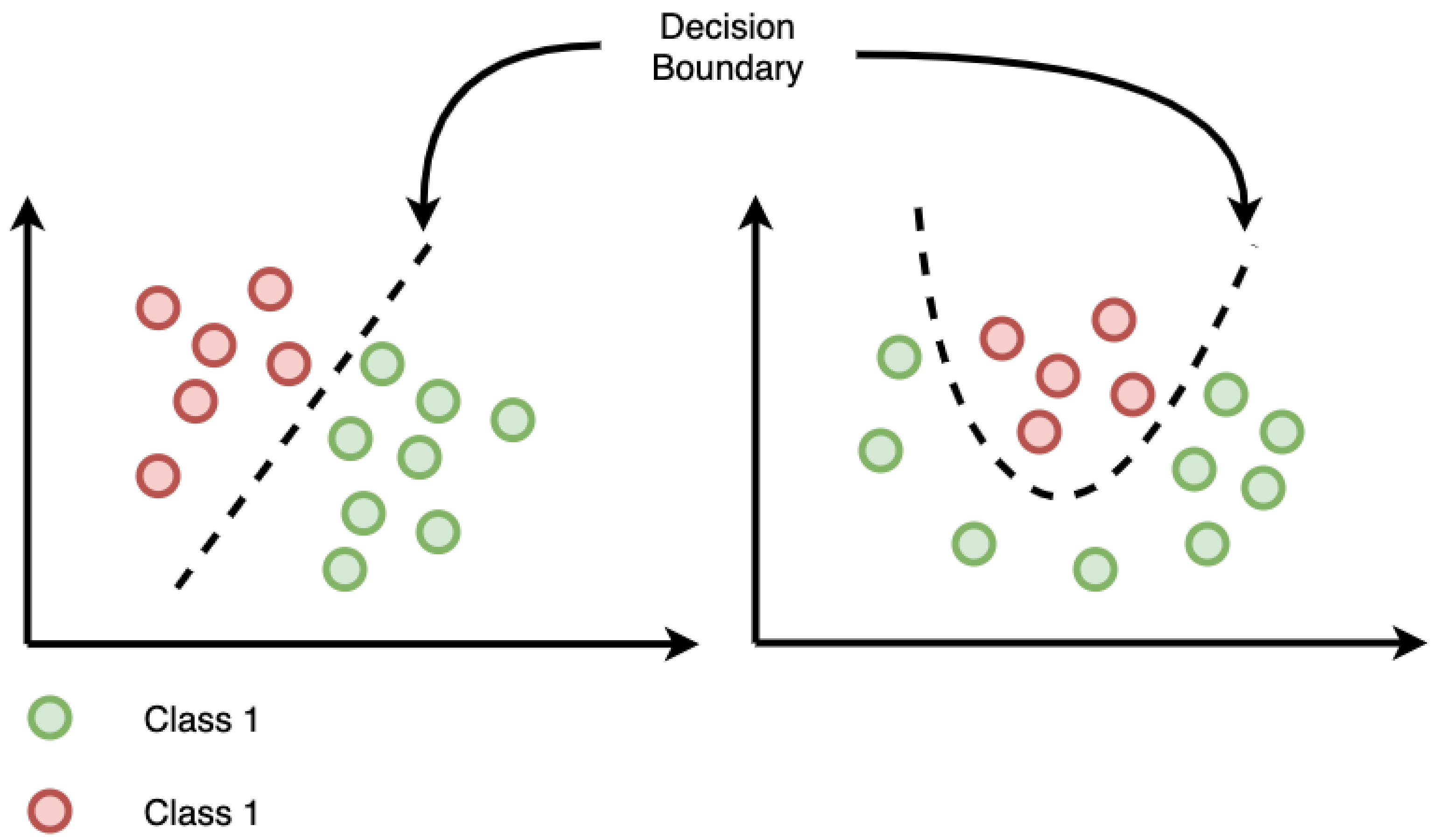
kNN- Non-parametric

	Discriminative model	Generative model
Goal	Directly estimate $P(y\|x)$	Estimate $P(x\|y)$ to then deduce $P(y\|x)$
What's learned	Decision boundary	Probability distributions of the data
Illustration		
Examples	Regressions, SVMs	GDA, Naive Bayes

Decision Boundary



Decision Boundary – what is Linear?



Lets look at some of the classifiers?

Logistic Regression

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}},$$

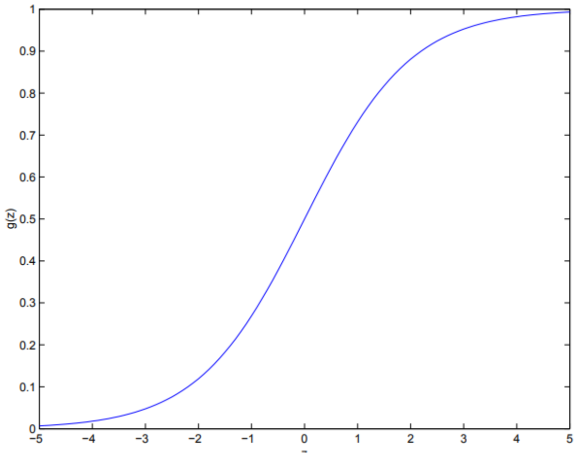
Logistic Regression

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}},$$

What the heck are all these terms?

$$g(z) = \frac{1}{1 + e^{-z}}$$

Logistic / Sigmoid Function



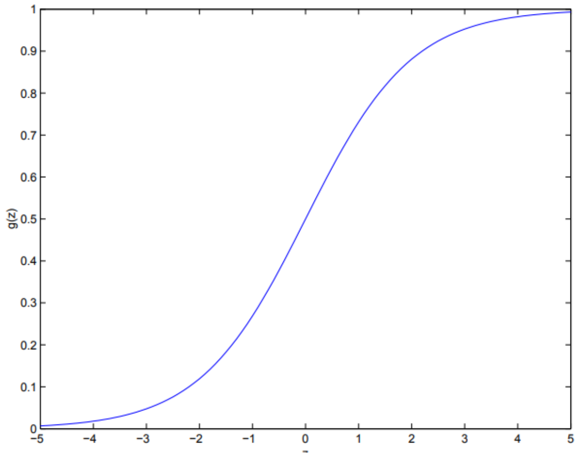
Logistic Regression

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}},$$

What the heck are all these terms?

$$g(z) = \frac{1}{1 + e^{-z}}$$

Logistic / Sigmoid Function



Ouput of sigmod function is a float.

How do you assign class?

Threshold!

Logistic Regression

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}},$$

Maximise likelihood

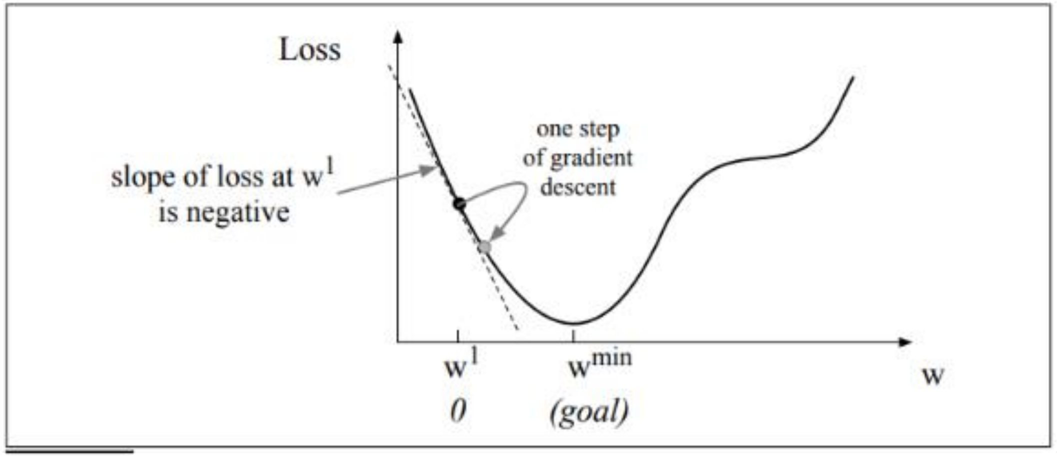
$$p(y|x) = \hat{y}^y \, (1-\hat{y})^{1-y}$$

$$\begin{aligned} \log p(y|x) &= \log \left[\hat{y}^y \, (1-\hat{y})^{1-y}\right] \\ &= y \log \hat{y} + (1-y) \log(1-\hat{y}) \end{aligned}$$

Minimize likelihood

$$L_{\mathrm{CE}}(\hat{y}, y) = -\log p(y|x) = -[y \log \hat{y} + (1-y) \log(1-\hat{y})]$$

$$L_{\mathrm{CE}}(\hat{y}, y) = -[y \log \sigma(\mathbf{w} \cdot \mathbf{x} + b) + (1-y) \log(1 - \sigma(\mathbf{w} \cdot \mathbf{x} + b))]$$



Logistic Regression

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}},$$

Maximise likelihood

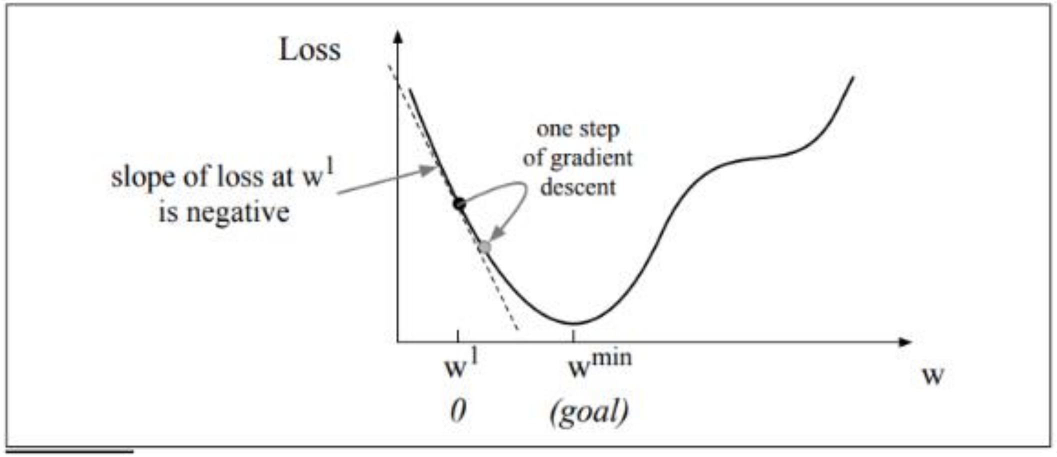
$$p(y|x) = \hat{y}^y (1-\hat{y})^{1-y}$$

$$\begin{aligned}\log p(y|x) &= \log\left[\hat{y}^y (1-\hat{y})^{1-y}\right] \\ &= y\log\hat{y} + (1-y)\log(1-\hat{y})\end{aligned}$$

Minimize likelihood

$$L_{\mathrm{CE}}(\hat{y},y) = -\log p(y|x) = -[y\log\hat{y} + (1-y)\log(1-\hat{y})]$$

$$L_{\mathrm{CE}}(\hat{y},y) = -[y\log\sigma(\mathbf{w}\cdot\mathbf{x}+b) + (1-y)\log(1-\sigma(\mathbf{w}\cdot\mathbf{x}+b))]$$



$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \frac{1}{m}\sum_{i=1}^{m} L_{\mathrm{CE}}(f(x^{(i)};\theta), y^{(i)})$$

$$w^{t+1} = w^t - \eta\frac{d}{dw}L(f(x;w),y)$$

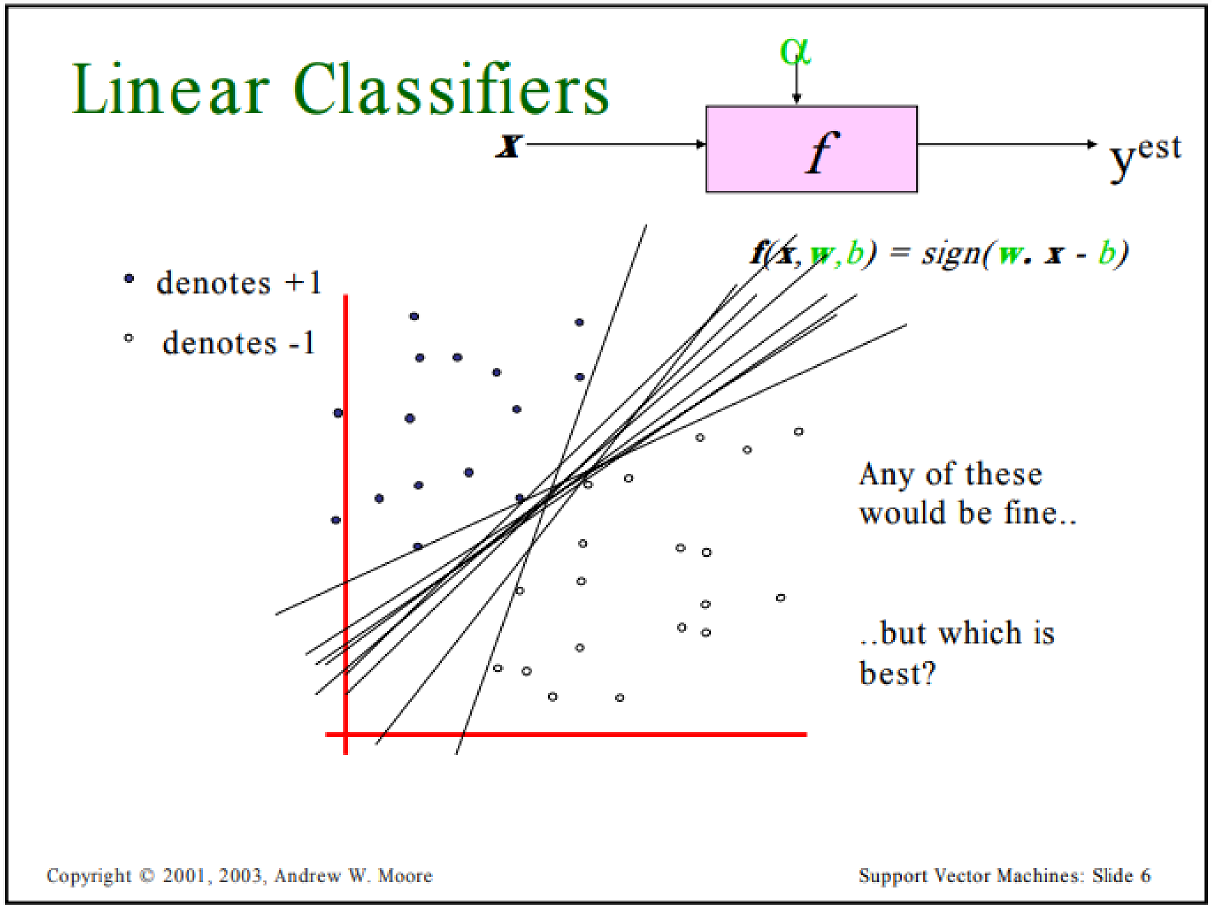
Stochastic Gradient Descent

function STOCHASTIC GRADIENT DESCENT($L()$, $f()$, x, y) **returns** θ
    # where: L is the loss function
    #   f is a function parameterized by θ
    #   x is the set of training inputs $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$
    #   y is the set of training outputs (labels) $y^{(1)}, y^{(2)}, \ldots, y^{(m)}$

$\theta \leftarrow 0$
repeat til done   # see caption
  For each training tuple $(x^{(i)}, y^{(i)})$ (in random order)
    1. Optional (for reporting):   # How are we doing on this tuple?
      Compute $\hat{y}^{(i)} = f(x^{(i)}; \theta)$ # What is our estimated output $\hat{y}$?
      Compute the loss $L(\hat{y}^{(i)}, y^{(i)})$ # How far off is $\hat{y}^{(i)}$ from the true output $y^{(i)}$?
    2. $g \leftarrow \nabla_\theta L(f(x^{(i)}; \theta), y^{(i)})$ # How should we move θ to maximize loss?
    3. $\theta \leftarrow \theta - \eta\, g$ # Go the other way instead
return θ

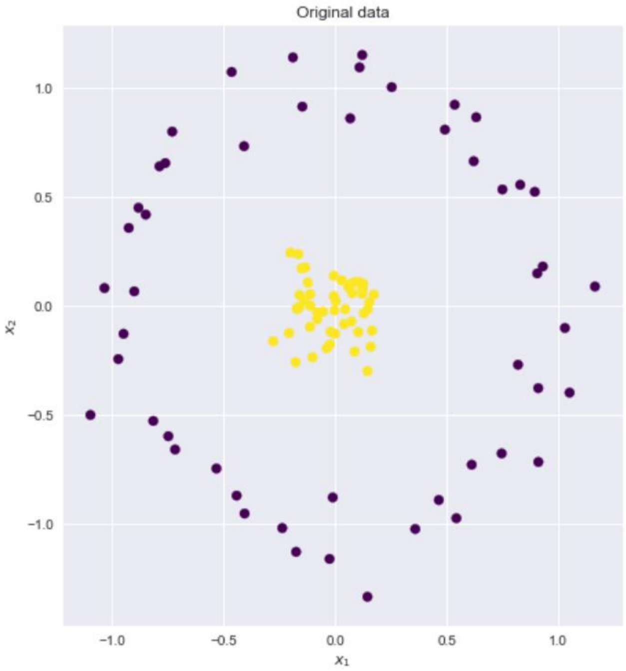
Figure 5.6 The stochastic gradient descent algorithm. Step 1 (computing the loss) is used mainly to report how well we are doing on the current tuple; we don't need to compute the loss in order to compute the gradient. The algorithm can terminate when it converges (or when the gradient norm $< \epsilon$), or when progress halts (for example when the loss starts going up on a held-out set).

Support Vector Machines – Switch to other PDF



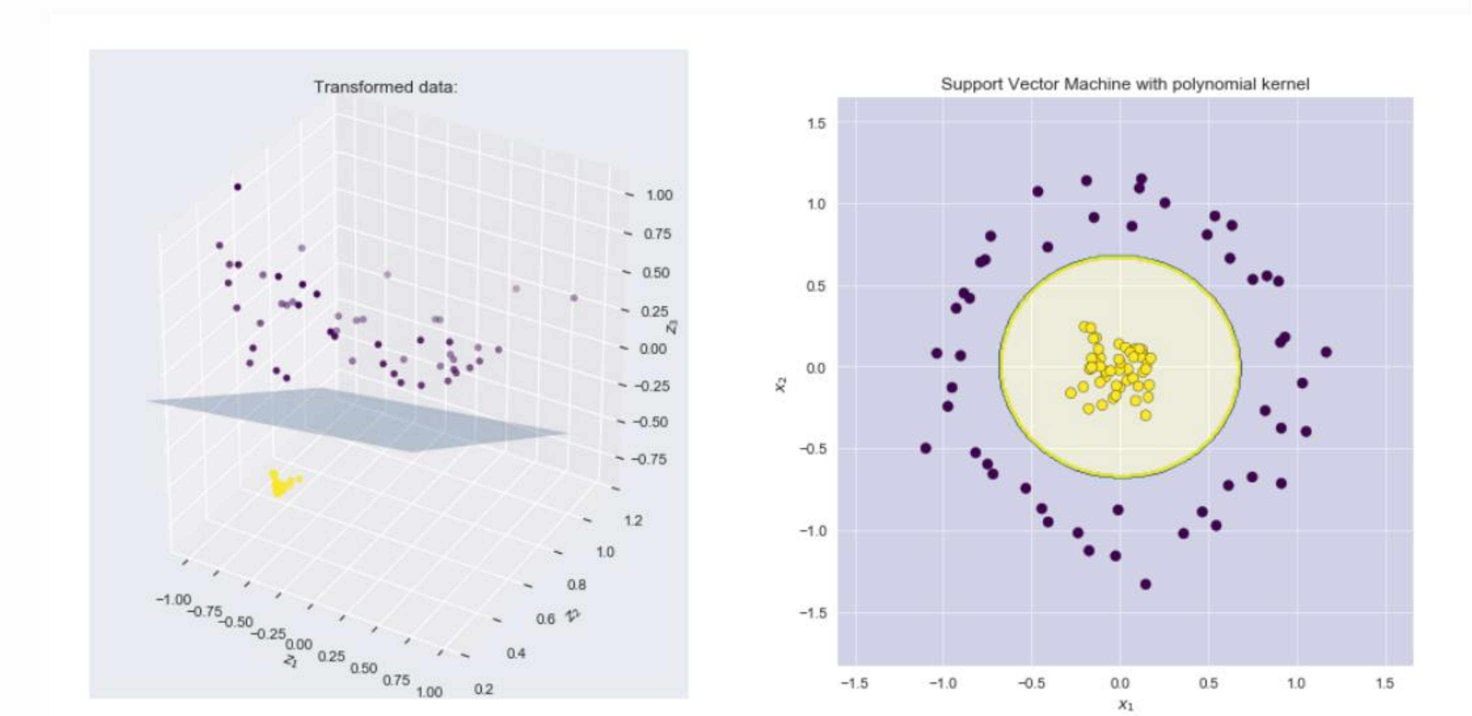
SVM

- Margin – better generalizability!
- Hard Margin , Soft Margin
- What if the decision boundary is non-linear?



SVM

- Margin – better generalizability!
- Hard Margin , Soft Margin
- What if the decision boundary is non-linear?



Ref : https://xavierbourretsicotte.github.io/Kernel_feature_map.html

How do I generate the higher dimension features?

$$\begin{aligned}
\phi(\mathbf{p})^T\phi(\mathbf{q}) &= [p_1^2, p_2^2, \sqrt{2}p_1p_2]^T[q_1^2, q_2^2, \sqrt{2}q_1q_2] \\
&= p_1^2q_1^2 + p_2^2q_2^2 + 2p_1p_2q_1q_2 \\
&= (p_1q_1 + p_2q_2)^2 \\
&= (\mathbf{p}^T\mathbf{q})^2 = \kappa(\mathbf{p}, \mathbf{q})
\end{aligned}$$

References

- Classifiers – Stanford CS229 Notes
- SVM
  - Series of Blogs by Alexandre KOWALCZYK
  - On Kernels, Blog by Xavier Bourret Sicotte
  - Andrew Moore's Tutorial on SVM