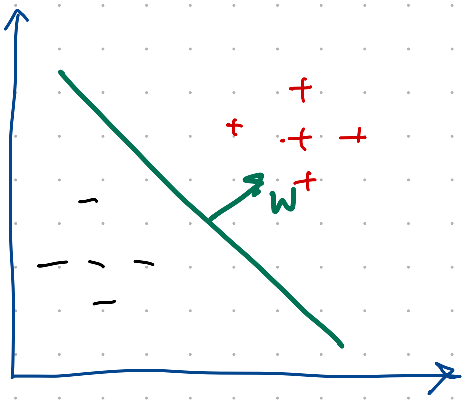
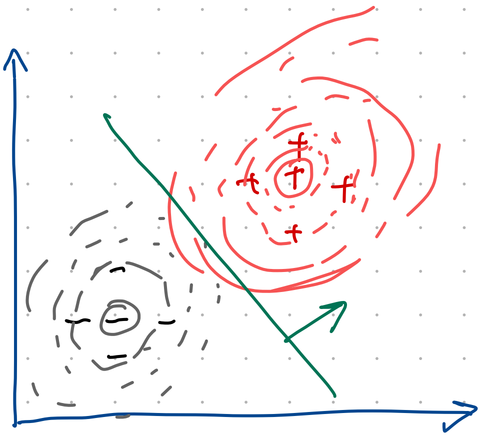
$$\boxed{P(y|x) \propto P(x|y)\,P(y)}$$

Types of ML algorithms:

- Generative algorithms: try to model $P(x|y)$ and $P(y)$.
- Discriminative algorithms: try to model $P(y|x)$ directly.





The perceptron is a discriminative algorithm since it models

$$P(y=+1\,|\,\vec{x}) = \begin{cases} 1 & \text{if } w^T x > 0 \\ 0 & \text{o.t.w.} \end{cases}$$

Naive Bayes is a generative algorithm since it models

$P(\vec{x}\,|\,y=+1)$ with some explicit modeling distribution (e.g. Gaussian).

- From our last lecture, we have shown that if the data is multinomial features, then Naive Bayes is a linear classifier

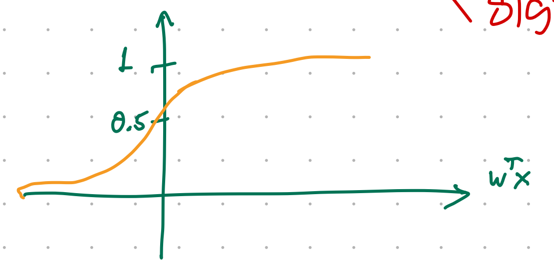
$$h(\vec{x}) = +1 \ \underline{\text{iff}} \ \underbrace{\vec{w}^T\vec{x}+b}_{\vec{w}^T\vec{x}} > 0 \quad \text{for specific vector } \vec{w}$$

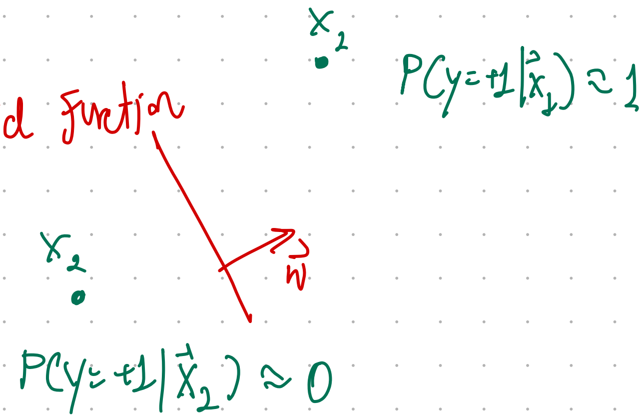
and scalar b and $y \in \{-1, +1\}$

- With the similar derivation for the case of Gaussian naive Bayes, we can also derive

$$P(y|\vec{x}) = \frac{1}{1+e^{-y\vec{w}^T\vec{x}}} \quad \text{for } y \in \{-1, +1\}$$

sigmoid function





*** We have a beatiful closed-form solution for $P(y|\vec{x})$ where the vector $\vec{w}$ can be founded by fitting the parameter regariding Gaussian distribution (our modeling assumption)

Two brilliant ideas: (1) It is nice to estimate $\vec{w}$ directly

(2) It is nice to set the sigmoid function for modelling $P(y|x)$

Logistic Regression

- Discriminative counter part of naive Bayes.
- The assumption is the following

$$P(y|\vec{x}) = \frac{1}{1+e^{-y(\vec{w}^T\vec{x})}}$$

- Here, we use the sigmoid function to model $P(y|\vec{x})$ where the $\vec{w}$ is the parameter that we need to estimate from data

Estimating $\vec{w}$

- **MLE** : We choose $\vec{w}_{MLE}$ that maximize the conditional likelihood

$$\vec{w}_{MLE} = \arg\max_{\vec{w}} \prod_{i=1}^{n} P(y_i \mid \vec{x}_i, \vec{w})$$

$$= \arg\max_{\vec{w}} \sum_{i=1}^{n} \log\left(\frac{1}{1+e^{-y_i \vec{w}^T \vec{x}_i}}\right)$$

$$= \arg\max_{\vec{w}} -\sum_{i=1}^{n} \log\left(1+e^{-y_i \vec{w}^T \vec{x}_i}\right)$$

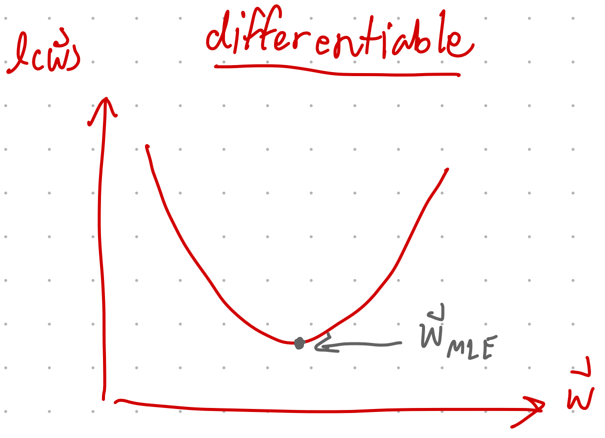
$$= \arg\min_{\vec{w}} \sum_{i=1}^{n} \log\left(1+e^{-y_i \vec{w}^T \vec{x}}\right)$$

- Note that there is no closed-form solution to $\vec{w}_{MLE}$.
- We will use "Gradient Descent" on the negative log likelihood

$$l(\vec{w}) = \sum_{i=1}^{n} \log\left(1+e^{-y_i \vec{w}^T \vec{x}_i}\right).$$

Observation: The function $l(\vec{w})$ is convex, continuous, and differentiable

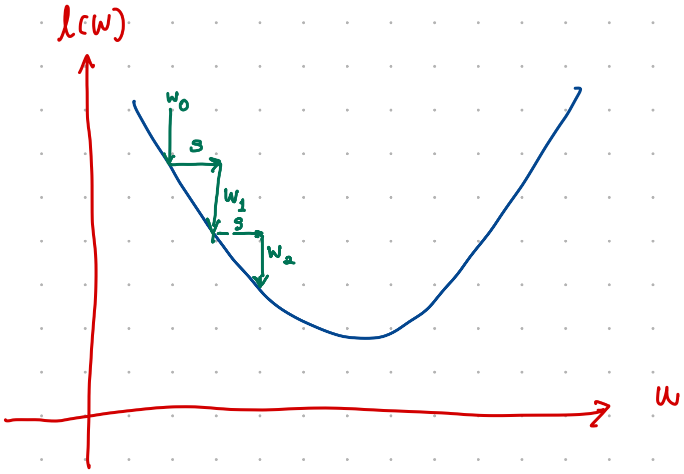
bowl shape



Goal: Find $\vec{w}$ that minimizes $l(\vec{w})$

Hill-climbing algorithm

1. Initialize $\vec{w}_0$
2. While $\|\vec{w}_{t+1} - \vec{w}_t\|_2 \geq \epsilon$
3. $\quad \vec{w}_{t+1} = \vec{w}_t + \vec{s}$



The problem is how to define $\vec{s}$?

Trick : Taylor's Approximation

- If the norm $\|s\|_2$ is small (i.e. $\vec{w}+\vec{s}$ is close to $\vec{w}$), then the following holds

$$\boxed{\ell(\vec{w}+\vec{s}) \approx \ell(\vec{w}) + \nabla \ell(\vec{w})^T \vec{s}}$$

Gradient Descent : First-order approximation

- We assume that the function ℓ around $\vec{w}$ is linear.
- We wish to take step $\vec{s}$ where $\ell(\vec{w}) > \ell(\vec{w}+\vec{s})$.
- In gradient descent, we set

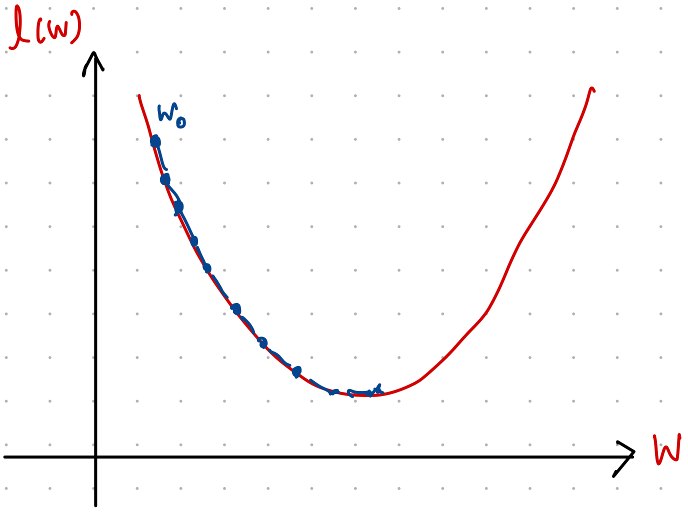
$$\boxed{\vec{s} = -\alpha \nabla \ell(\vec{w})}$$

where $\alpha > 0$ is the learning rate.

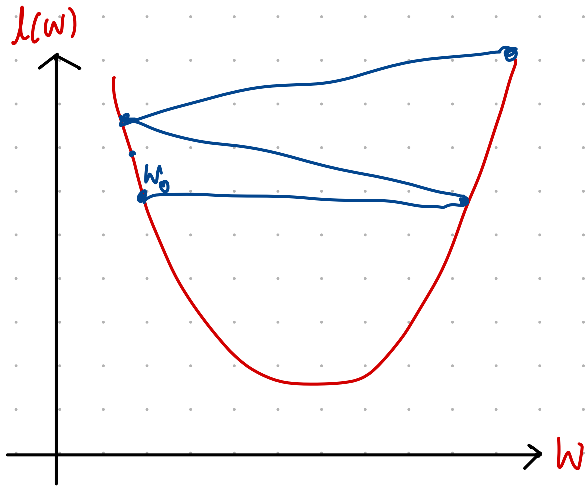
$$\Rightarrow \quad \ell(\vec{w}+\vec{s}) = \underbrace{\ell(\vec{w} + (-\alpha \nabla \ell(\vec{w})))}_{\text{after one update}} \approx \ell(\vec{w}) - \alpha \underbrace{(\nabla \ell(\vec{w}))^T (\nabla \ell(\vec{w}))}_{\|\nabla \ell(\vec{w})\|_2 > 0}$$

$< \ell(\vec{w})$

Setting the learning rate α



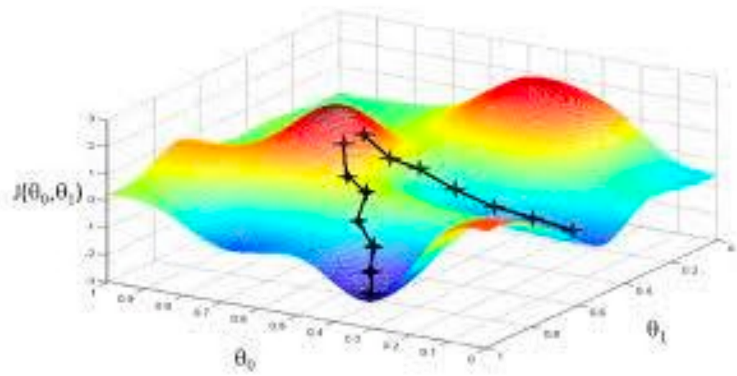
If α is too small,
the algorithm will converge
very slowly.



If α is too large,
the algorithm will never
converge

- A safe choice is to set $\alpha = \frac{t_0}{t_1}$, which guarantees that it will eventually become small enough to converge (for any $t_0 > 0$).

<u>Remark:</u> If the function is not converge, it may converge at local optimum.



$$P(y=1|\vec{x}) = \frac{P(\vec{x}|y=+1) \times P(y=+1)}{P(\vec{x})}$$

$$= \frac{P(\vec{x}|y=+1) \cancel{\times P(y=+1)}}{P(\vec{x}|y=+1)\cancel{P(y=+1)} + \frac{P(\vec{x}|y=-1)\,P(y=-1)}{P(\vec{x}|y=+1)\,P(y=+1)}}$$

$$= \frac{1}{1 + e^{\log\left(\frac{P(\vec{x}|y=-1)\,P(y=-1)}{P(\vec{x}|y=+1)\,P(y=+1)}\right)}}$$

$$= \frac{1}{1 + e^{\left[\log\left(\frac{P(\vec{x}|y=-1)}{P(\vec{x}|y=+1)}\right) + \log\left(\frac{P(y=-1)}{P(y=+1)}\right)\right]}}$$

$$= \frac{1}{1 + e^{\left[\log\left(\underbrace{\frac{\overset{\pi}{\cancel{P(y=-1)}}}{P(y=+1)}}_{1-\pi}\right) + \sum_{i=1}^{d}\log\left(\frac{P(x_i|y=-1)}{P(x_i|y=+1)}\right)\right]}} = \frac{1}{1 + e^{\left(\log\left(\frac{\pi}{1-\pi}\right) + \sum_{i=1}^{d}\log\left(\frac{P(x_i|y=-1)}{P(x_i|y=+1)}\right)\right)}}$$

$$\sum_i \log\left(\frac{P(x_i|y=-1)}{P(x_i|y=+1)}\right) = \sum_{i=1}^{d} \log \frac{\cancel{\frac{1}{\sqrt{2\pi}\sigma_i^2}} \exp\left(\frac{-(x_i-\mu_{i,-1})^2}{2\sigma_i^2}\right)}{\cancel{\frac{1}{\sqrt{2\pi}\sigma_i^2}} \exp\left(-\frac{(x_i-\mu_{i,+1})^2}{2\sigma_i^2}\right)}$$

$$= \sum_{i=1}^{d} \cancel{\log}\left(\cancel{\exp}\left(\frac{-(x_i-\mu_{i,-1})^2 + (x_i-\mu_{i,+1})^2}{2\sigma_i^2}\right)\right)$$

$$= \sum_{i=1}^{d} \left(\frac{\cancel{x_i^2} - 2x_i\mu_{i,+1} + (\mu_{i,+1})^2 - \cancel{x_i^2} + 2x_i\mu_{i,-1} - (\mu_{i,-1})^2}{2\sigma_i^2}\right)$$

$$= \sum_{i=1}^{d} \left(\frac{2x_i(\mu_{i,-1} - \mu_{i,+1}) + (\mu_{i,+1})^2 - (\mu_{i,-1})^2}{2\sigma_i^2}\right)$$

$$= \sum_{i=1}^{d} \left(\frac{(\mu_{i,-1} - \mu_{i,+1})}{\sigma_i^2} x_i + \frac{(\mu_{i,+1})^2 - (\mu_{i,-1})^2}{2\sigma_i^2}\right)$$

$$= \sum_{i=1}^{d} \underbrace{\frac{\mu_{i,-1} - \mu_{i,+1}}{\sigma_i^2}}_{w_d} x_i + \underbrace{\boxed{\sum_{i=1}^{d} \frac{(\mu_{i,+1})^2 - (\mu_{i,-1})^2}{2\sigma_i^2}}}_{y}$$

$$= \vec{w}^T \vec{x} + y$$

$$P(y=+1 \mid \vec{x}) = \frac{1}{1+\exp\left(\log\left(\frac{\pi}{1-\pi}\right) + \vec{w}^T\vec{x} + y\right)}$$

$$= \frac{1}{1+\exp\left(\vec{w}^T\vec{x} + \underbrace{\log\left(\frac{\pi}{1-\pi}\right) + y}_{b}\right)}$$

$$= \frac{1}{1+\exp(\vec{w}^T\vec{x}+b)} = \frac{1}{1+e^{(w^Tx+b)}}$$