Lecture 15: Kernel perceptron, Neural Networks, SVMs etc

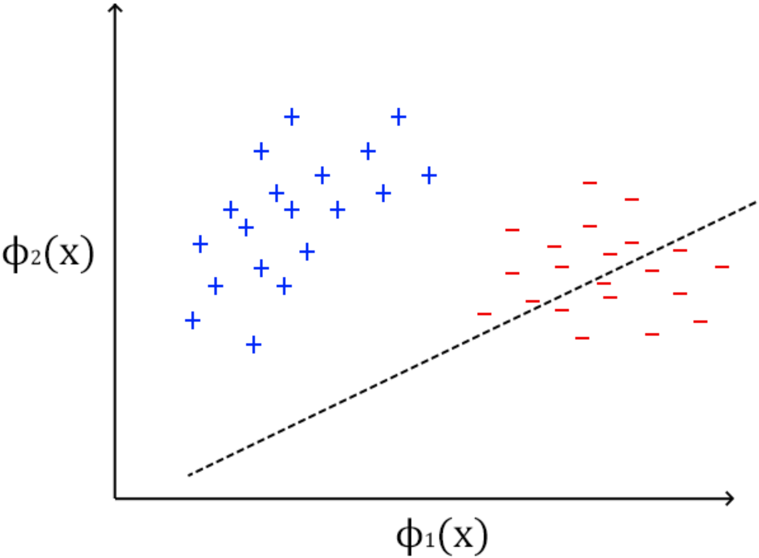
Instructor: Prof. Ganesh Ramakrishnan

Perceptron Update Rule: Basic Idea

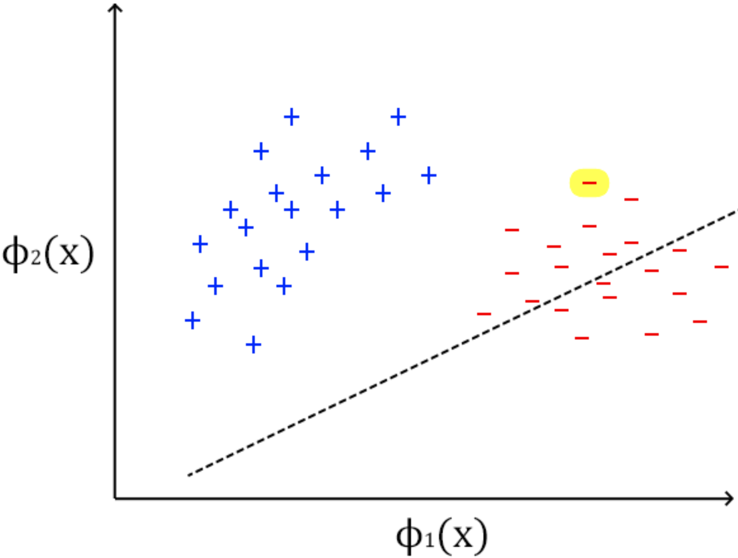
- Perceptron works for two classes ($y = \pm 1$). A point is misclassified if $y\mathbf{w}^T(\phi(\mathbf{x})) < 0$
- Perceptron Algorithm:
  - INITIALIZE: w=ones()
  - REPEAT: for each $< \mathbf{x}, y >$
    - If $y\mathbf{w}^T\Phi(\mathbf{x}) < 0$
    - then, $\mathbf{w} = \mathbf{w} + \eta\phi(\mathbf{x}).y$
    - endif
- **Intuition:**

$$\begin{aligned} y(\mathbf{w}^{(k+1)})^T\phi(\mathbf{x}) &= y\left(\mathbf{w}^k + \eta y\phi^T(\mathbf{x})\right)\phi(\mathbf{x}) \\ &= y(\mathbf{w}^k)^T\phi(\mathbf{x}) + \eta y^2\|\phi(\mathbf{w})\|^2 \\ &> y(\mathbf{w}^k)^T\phi(\mathbf{x}) \end{aligned}$$

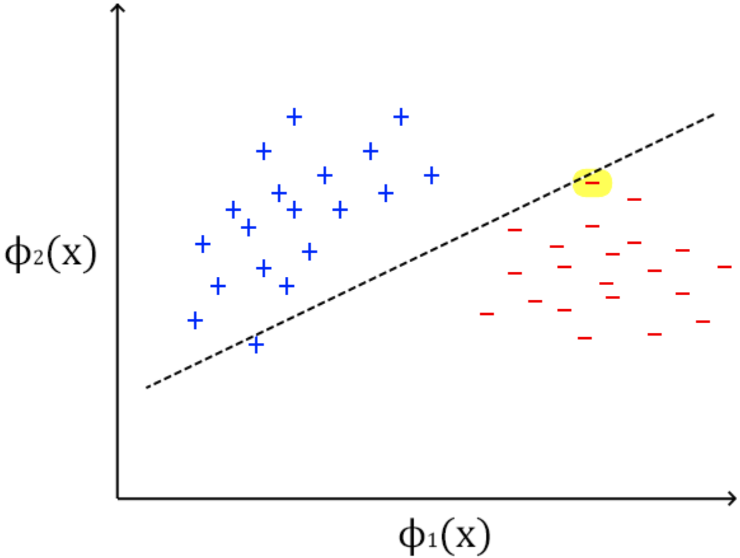
Since $y(\mathbf{w}^k)^T\phi(\mathbf{x}) \leq 0$, we have $y(\mathbf{w}^{(k+1)})^T\phi(\mathbf{x}) > y(\mathbf{w}^k)^T\phi(\mathbf{x}) \Rightarrow$ more hope that this point is classified correctly now.



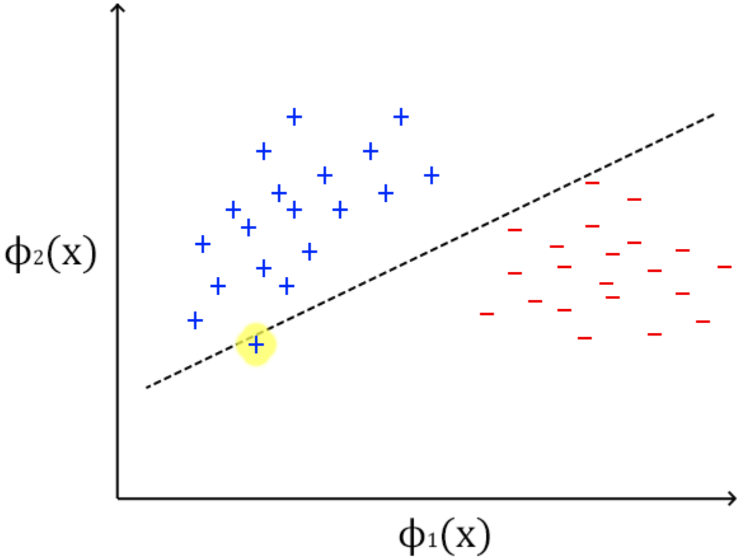
$\phi_2(x)$
$\phi_1(x)$



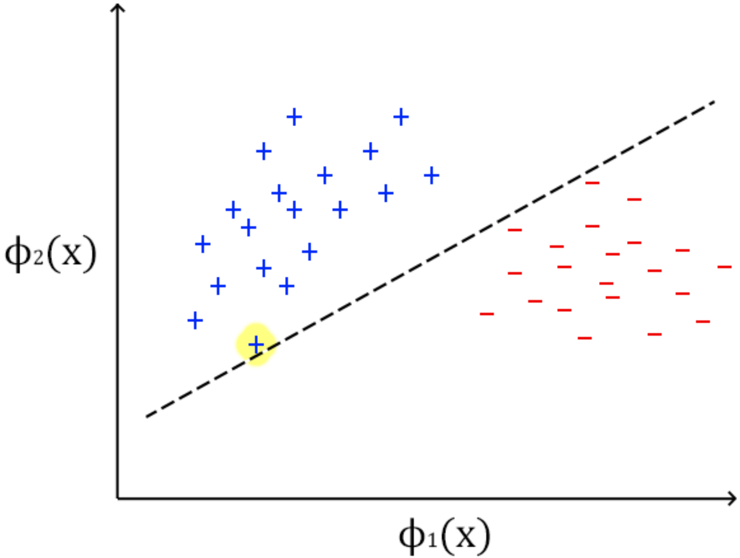
$\phi_2(x)$
$\phi_1(x)$



ϕ2(x)
ϕ1(x)



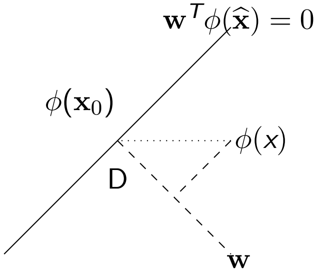
ϕ2(x)
ϕ1(x)



$\phi_2(x)$
$\phi_1(x)$

Perceptron Update Rule: Error Perspective

- Explicitly account for signed distance of (misclassified) points from the hyperplane $\mathbf{w}^T\phi(\widehat{\mathbf{x}}) = 0$. Consider point $\mathbf{x}_0$ such that $\mathbf{w}^T(\phi(\mathbf{x}_0)) = 0$
- (Signed) Distance from hyperplane is: $\mathbf{w}^T(\phi(\mathbf{x}) - \phi(\mathbf{x}_0)) = \mathbf{w}^T(\phi(\mathbf{x}))$
- Unsigned distance from hyperplane is: $y\mathbf{w}^T(\phi(\mathbf{x}))$ (assumes correct classification)



- If $\mathbf{x}$ is misclassified, the misclassification cost for $\mathbf{x}$ is $-y\mathbf{w}^T(\phi(\mathbf{x}))$

Perceptron Update Rule: Error Minimization

- Perceptron update tries to minimize the error function $E =$ negative of sum of unsigned distances over misclassified examples = sum of misclassification costs

$$E = -\sum_{(\mathbf{x},y)\in\mathcal{M}} y\mathbf{w}^T\phi(\mathbf{x})$$

where $\mathcal{M} \subseteq \mathcal{D}$ is the set of misclassified examples.

- **Gradient Descent (Batch Perceptron) Algorithm** $\nabla_{\mathbf{w}} E = -\sum_{(\mathbf{x},y)\in M} y\phi(\mathbf{x})$

$$\begin{aligned}\mathbf{w}^{(k+1)} &= \mathbf{w}^k - \eta\nabla_{\mathbf{w}} E \\ &= \mathbf{w}^k + \eta\sum_{(\mathbf{x},y)\in\mathcal{M}} y\phi(\mathbf{x})\end{aligned}$$

Perceptron Update Rule: Error Minimization

- Batch update considers all misclassified points simultaneously

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{k} - \eta \nabla_{\mathbf{w}} E$$
$$= \mathbf{w}^{k} + \eta \sum_{(\mathbf{x},y)\in\mathcal{M}} y\phi(\mathbf{x})$$

- Perceptron update $\Rightarrow$ *Stochastic Gradient Descent*:
$\nabla_{\mathbf{w}} E = -\sum_{(\mathbf{x},y)\in\mathcal{M}} y\phi(\mathbf{x}) = -\sum_{(\mathbf{x},y)\in\mathcal{M}} \nabla_{\mathbf{w}} E(\mathbf{x})$ s.t. $E(\mathbf{x}) = -y\mathbf{w}^{T}\phi(\mathbf{x})$

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{k} - \eta \nabla_{w} E(\mathbf{x}) \qquad \text{(for any } (\mathbf{x}, y) \in \mathcal{M})$$
$$= \mathbf{w}^{k} + \eta y \phi(\mathbf{x})$$

Perceptron Update Rule: Further analysis

- **Formally,**:- If $\exists$ an optimal separating hyperplane with parameters $\mathbf{w}^*$ such that,

$$\forall\ (\mathbf{x}, y),\ \ y\phi^T(\mathbf{x})\mathbf{w}^* \geq 0$$

then the perceptron algorithm converges.
Proof:- We want to show that

$$\lim_{k\to\infty} \|\mathbf{w}^{(k+1)} - \rho\mathbf{w}^*\|^2 = 0 \tag{1}$$

(If this happens for some constant ρ, we are fine.)

Perceptron Update Rule: Further analysis

- **Formally,**:- If $\exists$ an optimal separating hyperplane with parameters $\mathbf{w}^*$ such that,

$$\forall\ (\mathbf{x}, y),\ \ y\phi^T(\mathbf{x})\mathbf{w}^* \geq 0$$

  then the perceptron algorithm converges.
  Proof:- We want to show that

$$\lim_{k\to\infty} \|\mathbf{w}^{(k+1)} - \rho\mathbf{w}^*\|^2 = 0 \tag{1}$$

(If this happens for some constant ρ, we are fine.)

$$\|\mathbf{w}^{(k+1)} - \rho\mathbf{w}^*\|^2 = \|\mathbf{w}^k - \rho\mathbf{w}^*\|^2 + \|y\phi(\mathbf{x})\|^2 + 2y(\mathbf{w}^k - \rho\mathbf{w}^*)^T\phi(\mathbf{x}) \tag{2}$$

- For convergence of perceptron, we need L.H.S. to be less than R.H.S. at every step, although by some small but non-zero value (with $\theta \neq 0$)

$$\|\mathbf{w}^{(k+1)} - \rho\mathbf{w}^*\|^2 \leq \|\mathbf{w}^k - \rho\mathbf{w}^*\|^2 - \theta^2 \tag{3}$$

Perceptron Update Rule: Further analysis

- Need that $\|\mathbf{w}^{(k+1)} - \rho\mathbf{w}^*\|^2$ reduces by atleast θ^2 at every iteration.

$$\|\mathbf{w}^{(k+1)} - \rho\mathbf{w}^*\|^2 \leq \|\mathbf{w}^k - \rho\mathbf{w}^*\|^2 - \theta^2 \quad (4)$$

- Based on (2) and (4), we need to find θ such that,

Perceptron Update Rule: Further analysis

- Need that $\|\mathbf{w}^{(k+1)} - \rho\mathbf{w}^*\|^2$ reduces by atleast θ^2 at every iteration.

$$\|\mathbf{w}^{(k+1)} - \rho\mathbf{w}^*\|^2 \leq \|\mathbf{w}^k - \rho\mathbf{w}^*\|^2 - \theta^2 \tag{4}$$

- Based on (2) and (4), we need to find θ such that,

$$\|\phi(\mathbf{x})\|^2 + 2y(\mathbf{w}^k - \rho\mathbf{w}^*)^T\phi(\mathbf{x}) \leq -\theta^2$$

($\|y\phi(\mathbf{x})\|^2 = \|\phi(\mathbf{x})\|^2$ since $y = \pm 1$)

- The number of iterations would be: $O\left(\frac{\|\mathbf{w}^{(0)} - \rho\mathbf{w}^*\|^2}{\theta^2}\right)$
- Tutorial 6, Problem 4 is concerning the number of iterations. But first we will discuss how convergence holds in the first place!

Perceptron Update Rule: Further analysis

- **Observations**:-
  1. $y(\mathbf{w}^k)^T\phi(\mathbf{x}) < 0$ ($\because$ $\mathbf{x}$ was misclassified)
  2. $\Gamma^2 = \max_{\mathbf{x}\in\mathcal{D}} \|\phi(\mathbf{x})\|^2$
  3. $\delta = \max_{\mathbf{x}\in\mathcal{D}} -2y{\mathbf{w}^*}^T\phi(\mathbf{x})$
- Here, negative margin $\delta = -2y{\mathbf{w}^*}^T\phi(\widehat{\mathbf{x}})$ is the negative of unsigned distance of closest point $\widehat{\mathbf{x}}$ from separating hyperplane : $\widehat{\mathbf{x}} = \operatorname{argmax}_{\mathbf{x}\in\mathcal{D}} -2y{\mathbf{w}^*}^T\phi(\mathbf{x}) = \operatorname{argmin}_{\mathbf{x}\in\mathcal{D}} y{\mathbf{w}^*}^T\phi(x)$
- Since the data is linearly separable,

Perceptron Update Rule: Further analysis

- **Observations**:-
  1. $y(\mathbf{w}^k)^T\phi(\mathbf{x}) < 0$ ($\because$ $\mathbf{x}$ was misclassified)
  2. $\Gamma^2 = \max\limits_{\mathbf{x}\in\mathcal{D}} \|\phi(\mathbf{x})\|^2$
  3. $\delta = \max\limits_{\mathbf{x}\in\mathcal{D}} -2y{\mathbf{w}^*}^T\phi(\mathbf{x})$
- Here, negative margin $\delta = -2y{\mathbf{w}^*}^T\phi(\widehat{\mathbf{x}})$ is the negative of unsigned distance of closest point $\widehat{\mathbf{x}}$ from separating hyperplane : $\widehat{\mathbf{x}} = \operatorname*{argmax}\limits_{\mathbf{x}\in\mathcal{D}} -2y{\mathbf{w}^*}^T\phi(\mathbf{x}) = \operatorname*{argmin}\limits_{\mathbf{x}\in\mathcal{D}} y{\mathbf{w}^*}^T\phi(x)$
- Since the data is linearly separable, $\widehat{y}{\mathbf{w}^*}^T\phi(\widehat{\mathbf{x}}) \geq 0$, so, $\delta \leq 0$. Consequently:

$$0 \leq \|\mathbf{w}^{(k+1)} - \rho\mathbf{w}^*\|^2 < \|\mathbf{w}^k - \rho\mathbf{w}^*\|^2 + \Gamma^2 + \rho\delta$$

Perceptron Update Rule: Further analysis

- Since, ${\mathbf{w}^*}^T \phi(\widehat{\mathbf{x}}) \geq 0$, so, $\delta \leq 0$. Consequently:

$$0 \leq \|\mathbf{w}^{(k+1)} - \rho \mathbf{w}^*\|^2 < \|\mathbf{w}^k - \rho \mathbf{w}^*\|^2 + \Gamma^2 + \rho\delta$$

Taking,

Perceptron Update Rule: Further analysis

- Since, $\mathbf{w}^{*T}\phi(\widehat{\mathbf{x}}) \geq 0$, so, $\delta \leq 0$. Consequently:

$$0 \leq \|\mathbf{w}^{(k+1)} - \rho\mathbf{w}^*\|^2 < \|\mathbf{w}^k - \rho\mathbf{w}^*\|^2 + \Gamma^2 + \rho\delta$$

  Taking, $\rho = \dfrac{2\Gamma^2}{-\delta}$,

$$0 \leq \|\mathbf{w}^{(k+1)} - \rho\mathbf{w}^*\|^2 \leq \|\mathbf{w}^k - \rho\mathbf{w}^*\|^2 - \Gamma^2$$

- Hence, we got, $\Gamma^2 = \theta^2$, that we were looking for in eq.(3).
  $\therefore \|\mathbf{w}^{(k+1)} - \rho\mathbf{w}^*\|^2$ decreases by atleast Γ^2 at every iteration.
- Summarily: $\mathbf{w}^k$ converges to $\rho\mathbf{w}^*$ by making a minimum θ^2 decrement at each step.
- Thus, for $k \to \infty$, $\|\mathbf{w}^k - \rho\mathbf{w}^*\| \to 0$. This proves convergence.

Perceptron Update Rule: Further analysis

- A statement on number of iterations for convergence:
  If $||\mathbf{w}^*|| = 1$ and if there exists $\delta > 0$ such that for all $i = 1, \ldots, n$, $y_i(\mathbf{w}^*)^T\phi(\mathbf{x}_i) \geq \delta$ and $||\phi(\mathbf{x}_i)||^2 \leq \Gamma^2$ then the perceptron algorithm will make atmost $\frac{\Gamma^2}{\delta^2}$ errors (that is take atmost $\frac{\Gamma^2}{\delta^2}$ iterations to converge)

Non-linear perceptron?

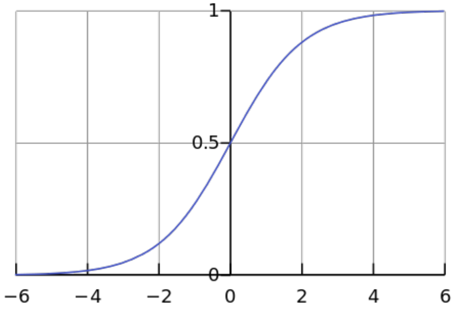
- Kernelized perceptron:

Non-linear perceptron?

- Kernelized perceptron: $f(\mathbf{x}) = sign\left(\sum_i \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b\right)$
  - INITIALIZE: α=zeroes()
  - REPEAT: for $< \mathbf{x}_i, y_i >$
    - If $sign\left(\sum_j \alpha_j y_j K(\mathbf{x}_j, \mathbf{x}_j) + b\right) \neq y_i$
    - then, $\alpha_i = \alpha_i + 1$
    - endif
- Neural Networks: Cascade of layers of perceptrons giving you non-linearity. But before that, we will discuss the specific sigmoidal percentron used most often in Neural Networks

Sigmoidal (perceptron) Classifier

1. **(Binary) Logistic Regression**, abbreviated as **LR** is a single node perceptron-like classifier, but with....
   - $sign\left((\mathbf{w}^*)^T\phi(x)\right)$ replaced by $g\left((\mathbf{w}^*)^T\phi(\mathbf{x})\right)$ where $g(s)$ is sigmoid function: $g(s) = \frac{1}{1+e^{-s}}$
2. $g\left((\mathbf{w}^*)^T\phi(\mathbf{x})\right) = \frac{1}{1+e^{-(\mathbf{w}^*)^T\phi(\mathbf{x})}} \in [0,1]$ can be intepreted as $Pr(y=1|\mathbf{x})$
   - Then $Pr(y=0|x) = ?$



Logistic Regression: The Sigmoidal (perceptron) Classifier

1. Estimator $\widehat{\mathbf{w}}$ is a function of the dataset
$\mathcal{D} = \left\{(\phi(\mathbf{x}^{(1)}, y^{(1)}), (\phi(\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\phi(\mathbf{x}^{(m)}, y^{(m)}))\right\}$
   - Estimator $\widehat{\mathbf{w}}$ is meant to approximate the parameter $\mathbf{w}$.
2. Maximum Likelihood Estimator: Estimator $\widehat{\mathbf{w}}$ that maximizes the likelihood $L(\mathcal{D}; \mathbf{w})$ of the data $\mathcal{D}$.
   - Assumes that all the instances $(\phi(\mathbf{x}^{(1)}, y^{(1)}), (\phi(\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\phi(\mathbf{x}^{(m)}, y^{(m)}))$ in $\mathcal{D}$ are all independent and identically distributed (iid)
   - Thus, Likelihood is the probability of $\mathcal{D}$ under iid assumption: $\hat{\mathbf{w}} = \underset{\mathbf{w}}{\text{argmax}}\ L(\mathcal{D}, \mathbf{w}) =$

Logistic Regression: The Sigmoidal (perceptron) Classifier

1. Estimator $\widehat{\mathbf{w}}$ is a function of the dataset $\mathcal{D} = \left\{(\phi(\mathbf{x}^{(1)}, y^{(1)}), (\phi(\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\phi(\mathbf{x}^{(m)}, y^{(m)}))\right\}$
   - Estimator $\widehat{\mathbf{w}}$ is meant to approximate the parameter $\mathbf{w}$.
2. Maximum Likelihood Estimator: Estimator $\widehat{\mathbf{w}}$ that maximizes the likelihood $L(\mathcal{D}; \mathbf{w})$ of the data $\mathcal{D}$.
   - Assumes that all the instances $(\phi(\mathbf{x}^{(1)}, y^{(1)}), (\phi(\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\phi(\mathbf{x}^{(m)}, y^{(m)}))$ in $\mathcal{D}$ are all independent and identically distributed (iid)
   - Thus, Likelihood is the probability of $\mathcal{D}$ under iid assumption: $\hat{\mathbf{w}} = \underset{\mathbf{w}}{\text{argmax}}\ L(\mathcal{D}, \mathbf{w}) =$ $\text{argmax}_{\mathbf{w}} \prod_{i=1}^{m} p(y^{(i)} | \phi(\mathbf{x}^{(i)})) = \text{argmax}_{\mathbf{w}} \prod_{i=1}^{m} \left(\frac{1}{1+e^{-(w)^T \phi(x^{(i)})}}\right)^{y^{(i)}} \left(\frac{e^{-(w)^T \phi(x^{(i)})}}{1+e^{-(w)^T \phi(x^{(i)})}}\right)^{1-y^{(i)}}$

Training LR

1. Thus, Maximum Likelihood Estimator for $\mathbf{w}$ is

$$\hat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmax}}\; L(\mathcal{D}, w) = \underset{\mathbf{w}}{\operatorname{argmax}} \prod_{i=1}^{m} p(y^{(i)} | \phi(\mathbf{x}^{(i)}))$$

$$= \underset{\mathbf{w}}{\operatorname{argmax}} \prod_{i=1}^{m} \left(\frac{1}{1 + e^{-\mathbf{w}^T \phi(\mathbf{x}^{(i)})}} \right)^{y^{(i)}} \left(\frac{e^{-\mathbf{w}^T \phi(\mathbf{x}^{(i)})}}{1 + e^{-\mathbf{w}^T \phi(\mathbf{x}^{(i)})}} \right)^{1-y^{(i)}}$$

$$= \underset{\mathbf{w}}{\operatorname{argmax}}\; \prod_{i=1}^{m} \left(f_{\mathbf{w}} \left(\mathbf{x}^{(i)} \right) \right)^{y^{(i)}} \left(1 - f_{\mathbf{w}} \left(\mathbf{x}^{(i)} \right) \right)^{1-y^{(i)}}$$

2. Maximizing the likelihood $\Pr(\mathcal{D}; \mathbf{w})$ w.r.t $\mathbf{w}$, is the same as minimizing the negative log-likelihood $E(\mathbf{w}) = -\frac{1}{m} \log \Pr(\mathcal{D}; \mathbf{w})$ w.r.t $\mathbf{w}$.
   - Derive the expression for $E(\mathbf{w})$.
   - $E(\mathbf{w})$ is called the cross-entropy loss function