Linear Classifier and its Learning by Perceptron

Vojtěch Franc

February 24, 2020

Generic linear classifier

Instances of linear classifier

Perceptron algorithm

XEP33SML – Structured Model Learning, Summer 2020

A generic linear classifier

- $\mathcal{X}$ is a set of observations and $\mathcal{Y}$ is a finite set of hidden states
- $\phi\colon \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^n$ input-output feature map embeding $\mathcal{X} \times \mathcal{Y}$ to $\mathbb{R}^n$
- Generic linear classifier $h\colon \mathcal{X} \to \mathcal{Y}$

$$h(x; \boldsymbol{w}) = \operatorname*{Argmax}_{y \in \mathcal{Y}(x)} \langle \boldsymbol{w}, \boldsymbol{\phi}(x, y) \rangle$$

  where $\mathcal{Y}(x) \subseteq \mathcal{Y}$.
- We will usually assume that $\mathcal{Y}(x) = \mathcal{Y}$, $\forall x \in \mathcal{X}$.

Example: two-classes linear classifier

- $\mathcal{X}$ is a set of observations and $\mathcal{Y} = \{+1, -1\}$ is a set of hidden labels
- $\phi \colon \mathcal{X} \to \mathbb{R}^d$ feature map embedding observations from $\mathcal{X}$ to $\mathbb{R}^n$
- Two-class linear classifier $h \colon \mathcal{X} \to \mathcal{Y}$

$$h(x; \boldsymbol{w}, b) = \mathrm{sign}(\langle \boldsymbol{w}, \boldsymbol{\phi}(x) \rangle + b) = \begin{cases} +1 & \text{if} \quad \langle \boldsymbol{w}, \boldsymbol{\phi}(x) \rangle + b \geq 0 \\ -1 & \text{if} \quad \langle \boldsymbol{w}, \boldsymbol{\phi}(x) \rangle + b < 0 \end{cases}$$

- It is equivalent to

$$h(x; \boldsymbol{w}) = \operatorname*{Argmax}_{y \in \{+1,-1\}} y \left(\langle \boldsymbol{w}, \boldsymbol{\phi}(x) \rangle + b \right) = \operatorname*{Argmax}_{y \in \{+1,-1\}} \langle \boldsymbol{w}', \boldsymbol{\phi}(x, y) \rangle$$

for $\phi(x, y) = [y \, \boldsymbol{\phi}(x) \, , y]$ and $\boldsymbol{w}' = [\boldsymbol{w} \, , b]$.

Example: multi-class linear classifier

- $\mathcal{X}$ is a set of observations and $\mathcal{Y} = \{1, \dots, Y\}$ is a set of class labels
- Multi-class linear classifier $h\colon \mathcal{X} \to \mathcal{Y}$

$$h(x; \boldsymbol{w}) = \operatorname*{Argmax}_{y \in \mathcal{Y}} \langle \boldsymbol{w}_y, \boldsymbol{\phi}(x) \rangle$$

where $\boldsymbol{\phi}\colon \mathcal{X} \to \mathbb{R}^d$ is a feature map $\boldsymbol{w} = (\boldsymbol{w}_1, \dots, \boldsymbol{w}_Y) \in \mathbb{R}^{d \cdot Y}$ are parameters.

- We can write the score function as

$$\langle \boldsymbol{w}_y, \boldsymbol{\phi}(x) \rangle = \langle \boldsymbol{w}, \boldsymbol{\phi}(x, y) \rangle$$

where $\boldsymbol{\phi}\colon \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^{d \cdot Y}$ is

$$\boldsymbol{\phi}(x, y) = (\mathbf{0}; \dots; \underbrace{\boldsymbol{\phi}(x)}_{y\text{–th slot}}; \dots; \mathbf{0})$$

Example: sequence classifier

- $\boldsymbol{x} = (x_1, \ldots, x_L) \in \mathcal{X}^L$ sequence of L inputs
- $\boldsymbol{y} = (y_1, \ldots, y_L) \in \mathcal{Y}^L$ sequence of L labels from $\mathcal{Y} = \{A, \ldots, Z\}$

For example:

$$\boldsymbol{x} = (x_1, x_2, x_3, x_4) \quad \boldsymbol{y} = (y_1, y_2, y_3, y_4)$$

JOHN

BILL

⋮

DANA

Example: sequence classifier

- $\boldsymbol{x} = (x_1, \ldots, x_L) \in \mathcal{X}^L$ sequence of L images with characters
- $\boldsymbol{y} = (y_1, \ldots, y_L) \in \mathcal{Y}^L$ sequence of L labels from $\mathcal{Y} = \{A, \ldots, Z\}$

For example:

$$JOHN = h\Big(\text{JOHN}; \boldsymbol{w}\Big) = \operatorname*{Argmax}_{\boldsymbol{y} \in \mathcal{Y}^L} \Big\langle \boldsymbol{\phi}\Big(\text{JOHN}, \boldsymbol{y}\Big), \boldsymbol{w} \Big\rangle$$

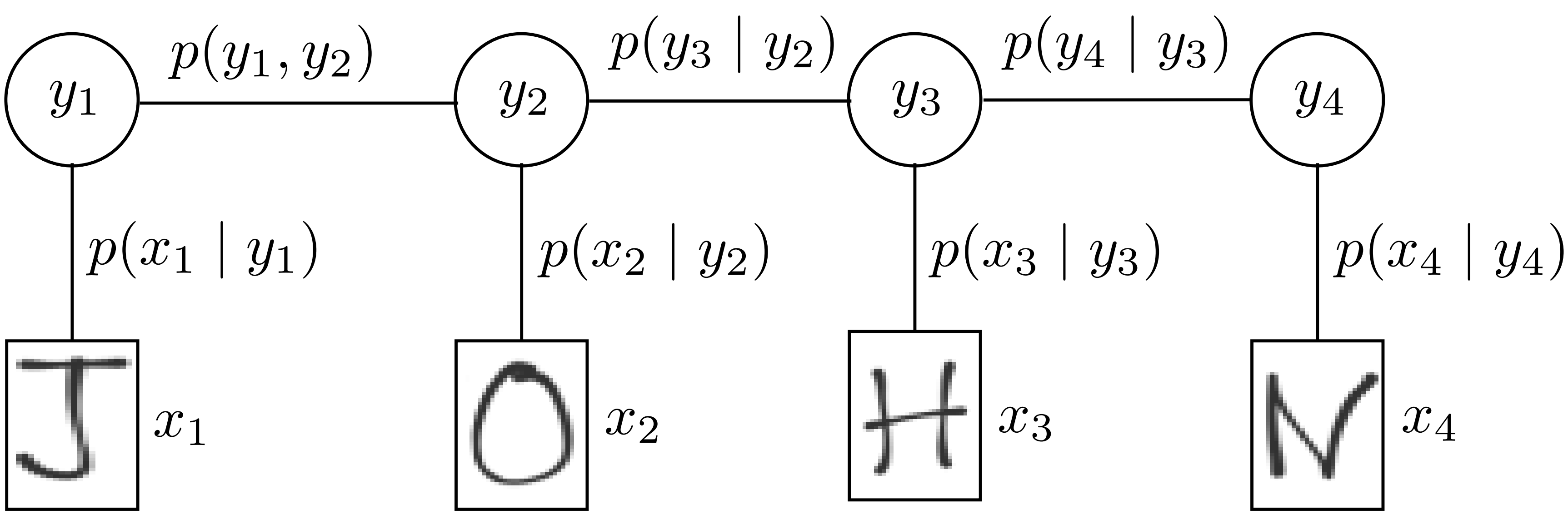
$$\begin{aligned}
\Big\langle \boldsymbol{\phi}\Big(\text{JOHN}, AAAA\Big), \boldsymbol{w} \Big\rangle &= 0.12 \\
\Big\langle \boldsymbol{\phi}\Big(\text{JOHN}, AAAB\Big), \boldsymbol{w} \Big\rangle &= 0.10 \\
&\vdots \\
\Big\langle \boldsymbol{\phi}\Big(\text{JOHN}, JOHN\Big), \boldsymbol{w} \Big\rangle &= 10.12 \\
&\vdots \\
\Big\langle \boldsymbol{\phi}\Big(\text{JOHN}, ZZZZ\Big), \boldsymbol{w} \Big\rangle &= 0.34
\end{aligned}$$

Example: sequence classifier

Hidden Markov Chain model:

- $\boldsymbol{x} = (x_1, \ldots, x_L) \in \mathcal{X}^L$ sequence of L inputs
- $\boldsymbol{y} = (y_1, \ldots, y_L) \in \mathcal{Y}^L$ sequence of L labels from $\mathcal{Y} = \{A, \ldots, Z\}$
- $p(x_i \mid y_i)$ emmission model
- $p(y_i \mid y_{i-1})$ transition model

$$p(\boldsymbol{x}, \boldsymbol{y}) = p(y_1) \prod_{i=2}^{L} p(y_i \mid y_{i-1}) \prod_{i=1}^{L} p(x_i \mid y_i)$$



Example: sequence classifier

- The MAP estimate from HMC:

$$\hat{\boldsymbol{y}} = \operatorname*{Argmax}_{\boldsymbol{y} \in \mathcal{Y}^L} \left(\log p(y_1) + \sum_{i=2}^{L} \log p(y_i \mid y_{i-1}) + \sum_{i=1}^{L} \log p(x_i \mid y_i) \right)$$

- Let us assume the following parametrization:

$$\begin{aligned} \log p(y_1) &= \langle \boldsymbol{w}, \boldsymbol{\phi}(y_1) \rangle \\ \log p(y_i \mid y_{i-1}) &= \langle \boldsymbol{w}, \boldsymbol{\phi}(y_{i-1}, y_i) \rangle \\ \log p(x_i | y_i) &= \langle \boldsymbol{w}, \boldsymbol{\phi}(x_i, y_i) \rangle \end{aligned}$$

- The MAP estimate becomes a linear classifier:

$$\hat{\boldsymbol{y}} = \operatorname*{Argmax}_{(y_1,\ldots,y_L) \in \mathcal{Y}^L} \left\langle \boldsymbol{w}, \underbrace{\boldsymbol{\phi}(y_1) + \sum_{i=2}^{L} \boldsymbol{\phi}(y_{i-1}, y_i) + \sum_{i=1}^{L} \boldsymbol{\phi}(x_i, y_i)}_{\boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y})} \right\rangle$$

Example: Max-Sum (Markov-Network) classifier

Setting:

- $(\mathcal{V}, \mathcal{E})$ is undirected graph; $\mathcal{V}$ are parts and $\mathcal{E} \subseteq \binom{|\mathcal{V}|}{2}$ are related parts
- $\boldsymbol{x} = (x_v \in \mathcal{X} \mid v \in \mathcal{V}) \in \mathcal{X}^{\mathcal{V}}$ inputs; $\boldsymbol{y} = (y_v \in \mathcal{Y} \mid v \in \mathcal{V}) \in \mathcal{Y}^{\mathcal{V}}$ labels
- $q_v(x, y) = \langle \boldsymbol{w}, \boldsymbol{\phi}_v(x, y) \rangle$
- $g_{vv'}(y, y') = \langle \boldsymbol{w}, \boldsymbol{\phi}_{vv'}(y, y') \rangle$

Linear Max-sum classifier: $h \colon \mathcal{X}^{\mathcal{V}} \to \mathcal{Y}^{\mathcal{V}}$ returns labeling

$$
\begin{aligned}
\hat{\boldsymbol{y}} &= \operatorname*{Argmax}_{\boldsymbol{y} \in \mathcal{Y}^{\mathcal{V}}} \left(\sum_{v \in \mathcal{V}} g_v(x_v, y_v) + \sum_{(v,v') \in \mathcal{E}} g_{vv'}(y_v, y_{v'}) \right) \\
&= \operatorname*{Argmax}_{\boldsymbol{y} \in \mathcal{Y}^{\mathcal{V}}} \left\langle \boldsymbol{w}, \underbrace{\sum_{v \in \mathcal{V}} \boldsymbol{\phi}(x_v, y_v) + \sum_{(v,v') \in \mathcal{E}} \boldsymbol{\phi}(y_v, y_{v'})}_{\boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y})} \right\rangle
\end{aligned}
$$

Learning by Emprical Risk Minimization

- $\ell\colon \mathcal{Y}\times\mathcal{Y}\to[0,\infty)$ loss function; we assume $\ell(y,y')=0$ iff $y=y'$.
- Find parameters $\boldsymbol{w}$ of $h(x;\boldsymbol{w})$ which minimize the expected risk

$$R(\boldsymbol{w}) = \mathbb{E}_{(x,y)\sim p}\Big(\ell(y,h(x;\boldsymbol{w}))\Big)$$

- The Empirical Risk Minimization principle leads to solving

$$\boldsymbol{w}^* \in \operatorname*{Argmin}_{\boldsymbol{w}\in\mathbb{R}^n} R_{\mathcal{T}^m}(\boldsymbol{w})$$

where the empirical risk is

$$R_{\mathcal{T}^m}(\boldsymbol{w}) = \frac{1}{m}\sum_{i=1}^{m}\ell(y^i,h(x^i;\boldsymbol{w}))$$

and $\mathcal{T}^m=\{(x^i,y^i)\in(\mathcal{X}\times\mathcal{Y})\mid i=1,\ldots,m\}$ are training examples drawn from i.i.d. with distribution $p(x,y)$.

Learning linear classifier from separable examples

- A correctly classified example (x^i, y^i), that is,

$$y^i = h(x^i; \boldsymbol{w}) = \operatorname*{Argmax}_{y \in \mathcal{Y}} \langle \boldsymbol{w}, \boldsymbol{\phi}(x^i, y) \rangle$$

implies

$$\langle \boldsymbol{\phi}(x^i, y^i), \boldsymbol{w} \rangle > \langle \boldsymbol{\phi}(x^i, y), \boldsymbol{w} \rangle \,, \qquad \forall y \in \mathcal{Y} \setminus \{y^i\}$$

Definition 1. *The examples $\mathcal{T}^m = \{(x^i, y^i) \in (\mathcal{X} \times \mathcal{Y}) \mid i = 1, \ldots, m\}$ are linearly separable w.r.t. joint feature map $\boldsymbol{\phi}\colon \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^n$ if there exists $\boldsymbol{w} \in \mathbb{R}^n$ such that*

$$\langle \boldsymbol{\phi}(x^i, y^i), \boldsymbol{w} \rangle > \langle \boldsymbol{\phi}(x^i, y), \boldsymbol{w} \rangle \,, \qquad \forall i \in \{1, \ldots, m\}, y \in \mathcal{Y} \setminus \{y^i\}$$

Example: sequence classifier

$$\mathcal{T}^m = \{(\text{JOHN}, JOHN), (\text{BILL}, BILL), \cdots\}$$

$$\left\{\begin{array}{lcl}
\left\langle \boldsymbol{\phi}\big(\text{JOHN}, JOHN\big), \boldsymbol{w} \right\rangle & > & \left\langle \boldsymbol{\phi}\big(\text{JOHN}, AAAA\big), \boldsymbol{w} \right\rangle \\
\left\langle \boldsymbol{\phi}\big(\text{JOHN}, JOHN\big), \boldsymbol{w} \right\rangle & > & \left\langle \boldsymbol{\phi}\big(\text{JOHN}, AAAB\big), \boldsymbol{w} \right\rangle \\
 & \vdots & \\
\left\langle \boldsymbol{\phi}\big(\text{JOHN}, JOHN\big), \boldsymbol{w} \right\rangle & > & \left\langle \boldsymbol{\phi}\big(\text{JOHN}, ZZZZ\big), \boldsymbol{w} \right\rangle
\end{array}\right\} \quad 26^4 - 1 \text{ inequalities}$$

$$\left\{\begin{array}{lcl}
\left\langle \boldsymbol{\phi}\big(\text{BILL}, BILL\big), \boldsymbol{w} \right\rangle & > & \left\langle \boldsymbol{\phi}\big(\text{BILL}, AAAA\big), \boldsymbol{w} \right\rangle \\
\left\langle \boldsymbol{\phi}\big(\text{BILL}, BILL\big), \boldsymbol{w} \right\rangle & > & \left\langle \boldsymbol{\phi}\big(\text{BILL}, AAAB\big), \boldsymbol{w} \right\rangle \\
 & \vdots & \\
\left\langle \boldsymbol{\phi}\big(\text{BILL}, BILL\big), \boldsymbol{w} \right\rangle & > & \left\langle \boldsymbol{\phi}\big(\text{JOHN}, ZZZZ\big), \boldsymbol{w} \right\rangle
\end{array}\right\} \quad 26^4 - 1 \text{ inequalities}$$

Example: noise-free setting

- $x \in \mathcal{X}$ is randomly generated according to some $p(x)$
- $y \in \mathcal{Y}$ are labels ($\mathcal{Y}$ is finite) generated from

$$p(y \mid x) = [\![h^*(x) = y]\!]$$

where $h^* \colon \mathcal{X} \to \mathcal{Y}$ is a function.

- Under assumption that $h^*(x) = \operatorname{argmax}_{y \in \mathcal{Y}} \langle \boldsymbol{w}, \boldsymbol{\phi}(x, y) \rangle$ the examples

$$\mathcal{T}^m = \{(x^i, y^i) \in \mathcal{X} \times \mathcal{Y} \mid i = 1, \ldots, m\}$$

generated from $p(x, y) = p(x)\, p(y \mid x)$ are linearly separable.

(Generic) Perceptron algorithm

- **Task:** given a set of points $\{\boldsymbol{a}^i \in \mathbb{R}^n \mid i = 1, 2, \ldots, l\}$ we want to find $\boldsymbol{w} \in \mathbb{R}^n$ such that

$$\langle \boldsymbol{w}, \boldsymbol{a}^i \rangle > 0\,, \qquad \forall i \in \{1, 2, \ldots, l\} \tag{1}$$

- **Perceptron:**
  1. $\boldsymbol{w} \leftarrow \mathbf{0}$
  2. Find a violating $\langle \boldsymbol{w}, \boldsymbol{a}^i \rangle \leq 0$, $i \in \{1, 2, \ldots, l\}$
  3. If there is no violating inequality return $\boldsymbol{w}$ otherwise update

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \boldsymbol{a}^i$$

  and go to step 2.

Convergence of Perceptron

Theorem 1. *For any linearly separable points* $\{\boldsymbol{a}^i \in \mathbb{R}^n \mid i = 1, 2, \ldots l\}$, *the Perceptron algorithm terminates in*

$$\frac{A^2}{\gamma^2}$$

steps at most where

$$A = \max_{i=1,\ldots,l} \|\boldsymbol{a}^i\|_2 \qquad \textit{and} \qquad \gamma = \max_{\|\boldsymbol{w}\|=1} \min_{i=1,\ldots,l} \frac{\langle \boldsymbol{w}, \boldsymbol{a}^i \rangle}{\|\boldsymbol{w}\|_2}$$

- Note that the upper bound $\frac{A^2}{\gamma^2}$ does not depend on the number of points l.

Structured Output Perceptron

- Learning $h(x;\boldsymbol{w}) = \mathrm{Argmax}_{y\in\mathcal{Y}}\langle \boldsymbol{w}, \boldsymbol{\phi}(x,y)\rangle$ from examples $\mathcal{T}^m = \{(x^i,y^i)\in(\mathcal{X}\times\mathcal{Y}) \mid i=1,\ldots,m\}$ leads to solving

$$\langle \boldsymbol{\phi}(x^i,y^i) - \boldsymbol{\phi}(x^i,y), \boldsymbol{w}\rangle > 0 \,, \qquad \forall i\in\{1,\ldots,m\}, y\in\mathcal{Y}\setminus\{y^i\}$$

- **Algorithm:**

1. $\boldsymbol{w} \leftarrow \boldsymbol{0}$
2. Find a misclassified example $(x^i,y^i)\in\mathcal{T}^m$ such that

$$y^i \neq \hat{y}^i = \mathop{\mathrm{Argmax}}_{y\in\mathcal{Y}}\langle \boldsymbol{w}, \boldsymbol{\phi}(x^i,y)\rangle \qquad \text{prediction problem}$$

3. If there is no misclassified example return $\boldsymbol{w}$ otherwise update

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \boldsymbol{\phi}(x^i,y^i) - \boldsymbol{\phi}(x^i,\hat{y}^i) \qquad \text{parameter update}$$

   and go to step 2.

Convergence of Structured Output Perceptron

- By Theorem 1 we have a guarantee that for linearly separable training set $\{(x^i, y^i) \in \mathcal{X} \times \mathcal{Y} \mid i = 1, 2, \ldots, m\}$ the SO-Perceptron terminates after at most $\frac{A^2}{\gamma^2}$ iterations where

$$A = \max_{\substack{i=1,2,\ldots,m \\ y \in \mathcal{Y} \setminus \{y^i\}}} \|\boldsymbol{\phi}(x^i, y^i) - \boldsymbol{\phi}(x^i, y)\| \leq 2 \max_{x \in \mathcal{X}, y \in \mathcal{Y}} \|\boldsymbol{\phi}(x, y)\|$$

and

$$\gamma = \max_{\|\boldsymbol{w}\|=1} \min_{\substack{i=1,2,\ldots,m \\ y \in \mathcal{Y} \setminus \{y^i\}}} \frac{\langle \boldsymbol{w}, \boldsymbol{\phi}(x^i, y^i) - \boldsymbol{\phi}(x^i, y) \rangle}{\|\boldsymbol{w}\|_2}$$