Decision-theoretic Machine Learning

Krzysztof Dembczyński and Wojciech Kotłowski

Intelligent Decision Support Systems Laboratory (IDSS)
Poznań University of Technology, Poland

Poznań University of Technology, Summer 2019

Agenda

1. **Introduction to Machine Learning**
2. Binary Classification
3. Bipartite Ranking
4. Multi-Label Classification

Outline

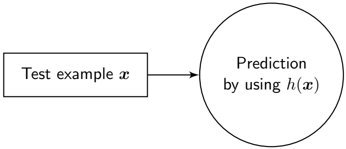
Outline

Supervised learning

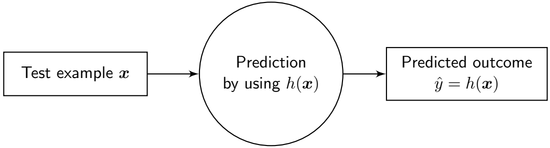
Supervised learning

Test example x

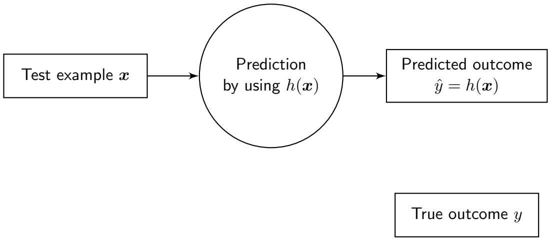
Supervised learning



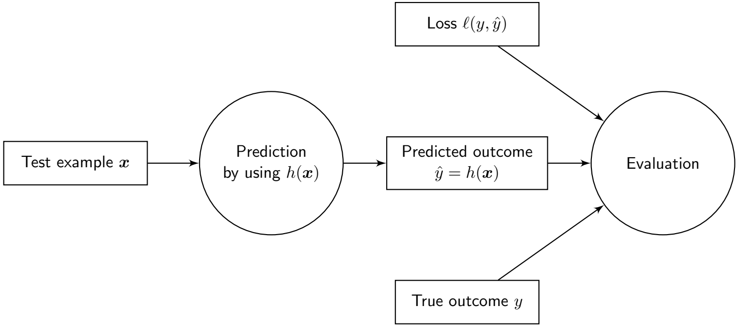
Supervised learning



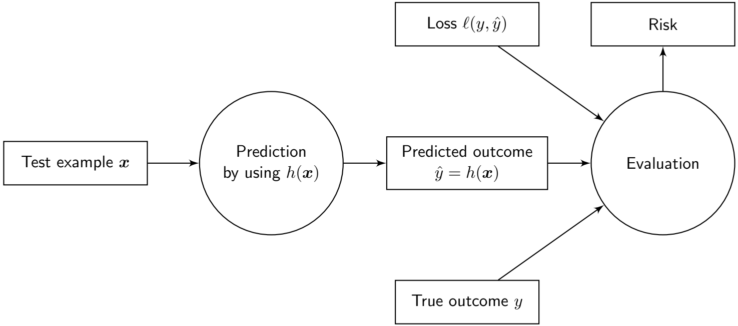
Supervised learning



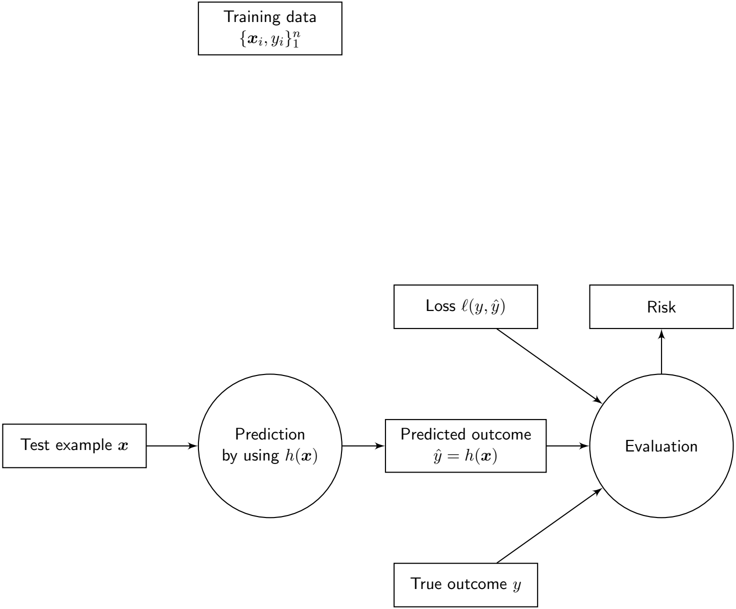
Supervised learning



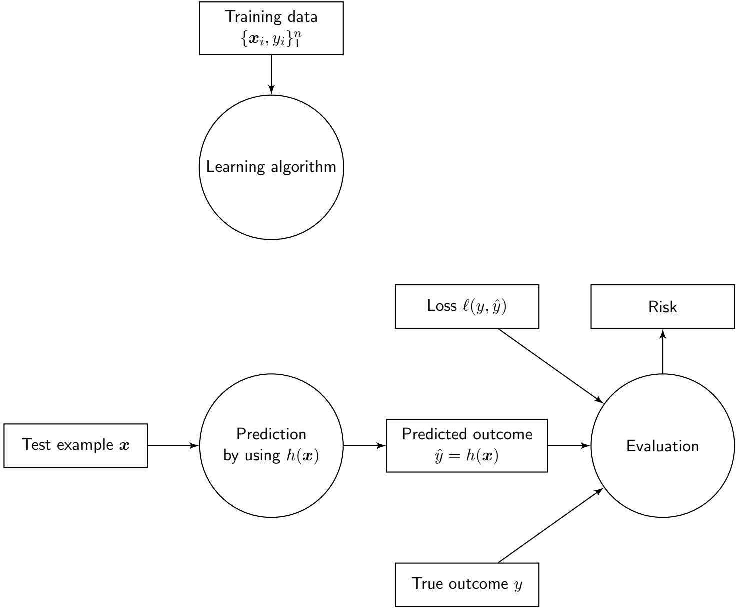
Supervised learning



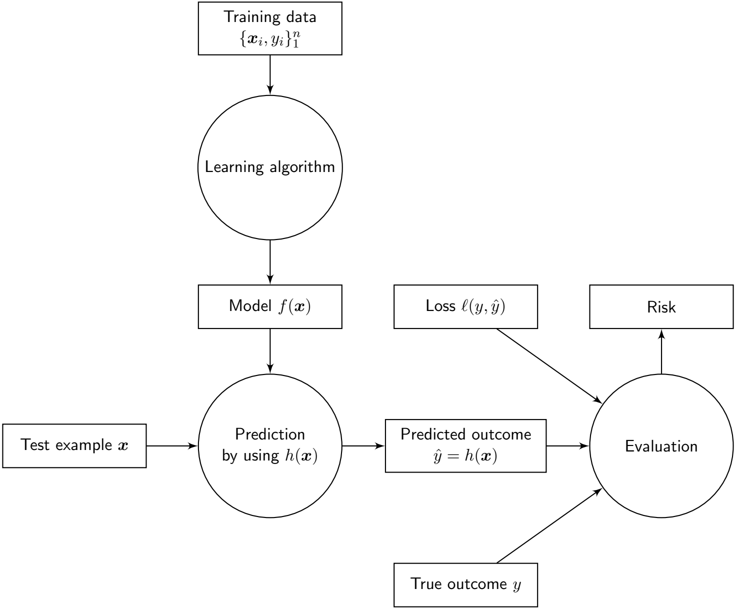
Supervised learning



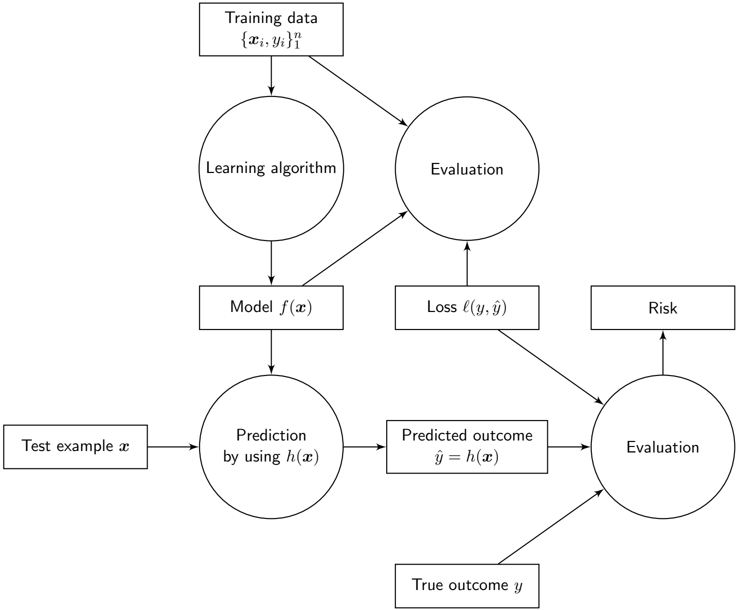
Supervised learning



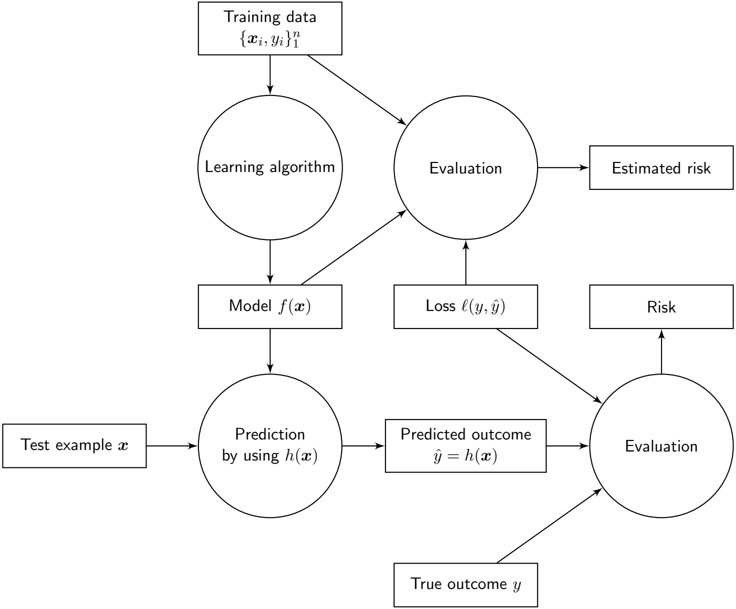
Supervised learning



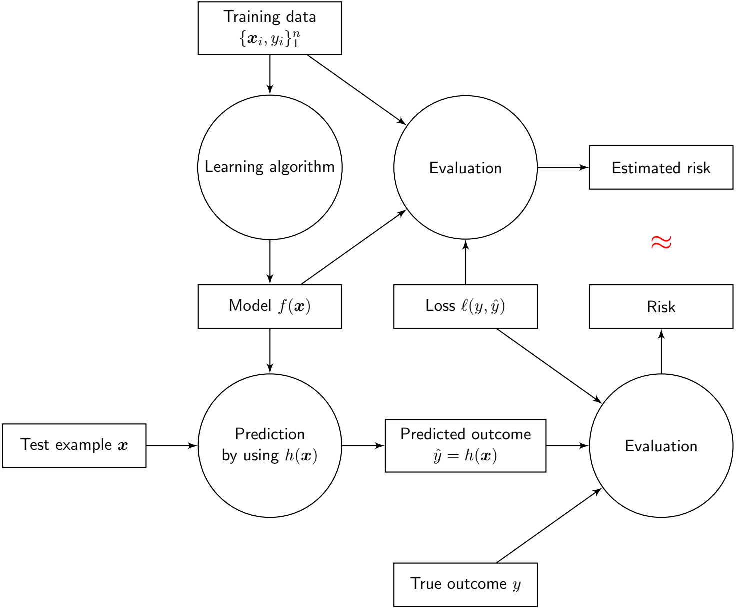
Supervised learning



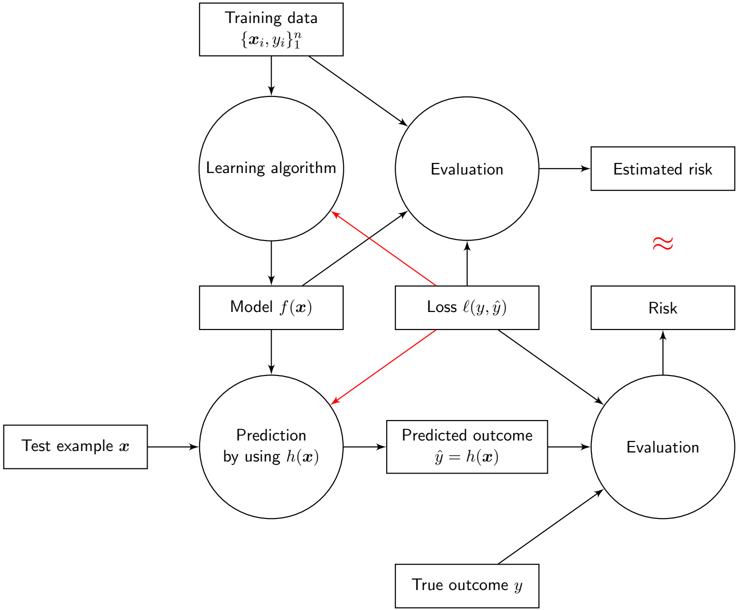
Supervised learning



Supervised learning



Supervised learning



Statistical learning framework

Statistical learning framework

- **Input** $\boldsymbol{x} \in \mathcal{X}$ drawn from a distribution $P(\boldsymbol{x})$.
  - usually a feature vector, $\mathcal{X} \subseteq \mathbb{R}^d$.

Statistical learning framework

- **Input** $\boldsymbol{x} \in \mathcal{X}$ drawn from a distribution $P(\boldsymbol{x})$.
  - usually a feature vector, $\mathcal{X} \subseteq \mathbb{R}^d$.
- **Outcome** $y \in \mathcal{Y}$ drawn from a distribution $P(y \,|\, \boldsymbol{x})$.
  - target of our prediction: class label, real value, label vector, etc.,
  - alternative view: **example** $(\boldsymbol{x}, y)$ drawn from $P(\boldsymbol{x}, y)$.

Statistical learning framework

- **Input** $\boldsymbol{x} \in \mathcal{X}$ drawn from a distribution $P(\boldsymbol{x})$.
  - usually a feature vector, $\mathcal{X} \subseteq \mathbb{R}^d$.
- **Outcome** $y \in \mathcal{Y}$ drawn from a distribution $P(y \,|\, \boldsymbol{x})$.
  - target of our prediction: class label, real value, label vector, etc.,
  - alternative view: **example** $(\boldsymbol{x}, y)$ drawn from $P(\boldsymbol{x}, y)$.
- **Prediction** $\hat{y} = h(\boldsymbol{x})$ by means of **prediction function** $h \colon \mathcal{X} \to \mathcal{Y}$.
  - h returns prediction $\hat{y} = h(\boldsymbol{x})$ for every input $\boldsymbol{x}$.

Statistical learning framework

- **Input** $\boldsymbol{x} \in \mathcal{X}$ drawn from a distribution $P(\boldsymbol{x})$.
  - usually a feature vector, $\mathcal{X} \subseteq \mathbb{R}^d$.
- **Outcome** $y \in \mathcal{Y}$ drawn from a distribution $P(y \,|\, \boldsymbol{x})$.
  - target of our prediction: class label, real value, label vector, etc.,
  - alternative view: **example** $(\boldsymbol{x}, y)$ drawn from $P(\boldsymbol{x}, y)$.
- **Prediction** $\hat{y} = h(\boldsymbol{x})$ by means of **prediction function** $h \colon \mathcal{X} \to \mathcal{Y}$.
  - h returns prediction $\hat{y} = h(\boldsymbol{x})$ for every input $\boldsymbol{x}$.
- **Loss** of our prediction: $\ell(y, \hat{y})$.
  - $\ell \colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ is a problem-specific **loss function**.

Statistical learning framework

- **Input** $\boldsymbol{x} \in \mathcal{X}$ drawn from a distribution $P(\boldsymbol{x})$.
  - usually a feature vector, $\mathcal{X} \subseteq \mathbb{R}^d$.
- **Outcome** $y \in \mathcal{Y}$ drawn from a distribution $P(y \,|\, \boldsymbol{x})$.
  - target of our prediction: class label, real value, label vector, etc.,
  - alternative view: **example** $(\boldsymbol{x}, y)$ drawn from $P(\boldsymbol{x}, y)$.
- **Prediction** $\hat{y} = h(\boldsymbol{x})$ by means of **prediction function** $h\colon \mathcal{X} \to \mathcal{Y}$.
  - h returns prediction $\hat{y} = h(\boldsymbol{x})$ for every input $\boldsymbol{x}$.
- **Loss** of our prediction: $\ell(y, \hat{y})$.
  - $\ell\colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ is a problem-specific **loss function**.
- **Goal**: find a prediction function with small loss.

Risk

- **Goal**: minimize the **expected** loss over all examples (**risk**):

$$L_\ell(h) = \mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[\ell(y, h(\boldsymbol{x}))\right].$$

Risk

- **Goal**: minimize the **expected** loss over all examples (**risk**):

$$L_\ell(h) = \mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[\ell(y, h(\boldsymbol{x}))\right].$$

- The **optimal** prediction function over all possible functions:

$$h^* = \arg\min_h L(h),$$

sometimes referred to as the **Bayes prediction function**.

Risk

- **Goal**: minimize the **expected** loss over all examples (**risk**):

$$L_\ell(h) = \mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[\ell(y, h(\boldsymbol{x}))\right].$$

- The **optimal** prediction function over all possible functions:

$$h^* = \arg\min_h L(h),$$

  sometimes referred to as the **Bayes prediction function**.

- The smallest achievable risk (**Bayes risk**):

$$L_\ell^* = L_\ell(h^*).$$

Decomposition of risk

$$L_\ell(h) \quad = \quad \mathbb{E}_{(\boldsymbol{x},y)}\left[\ell(y, h(\boldsymbol{x}))\right]$$

Decomposition of risk

$$
\begin{aligned}
L_\ell(h) &= \mathbb{E}_{(\boldsymbol{x},y)}\left[\ell(y,h(\boldsymbol{x}))\right] \\
&= \int_{\mathcal{X}\times\mathcal{Y}} \ell(y,h(\boldsymbol{x}))P(\boldsymbol{x},y)\mathrm{d}\boldsymbol{x}\mathrm{d}y
\end{aligned}
$$

Decomposition of risk

$$
\begin{aligned}
L_\ell(h) &= \mathbb{E}_{(\boldsymbol{x},y)}\left[\ell(y, h(\boldsymbol{x}))\right] \\
&= \int_{\mathcal{X}\times\mathcal{Y}} \ell(y, h(\boldsymbol{x})) P(\boldsymbol{x}, y) \mathrm{d}\boldsymbol{x} \mathrm{d}y \\
&= \int_{\mathcal{X}} \left(\int_{\mathcal{Y}} \ell(y, h(\boldsymbol{x})) P(y \,|\, \boldsymbol{x}) \mathrm{d}y \right) P(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}
\end{aligned}
$$

Decomposition of risk

$$\begin{aligned}
L_\ell(h) &= \mathbb{E}_{(\boldsymbol{x},y)}\left[\ell(y,h(\boldsymbol{x}))\right] \\
&= \int_{\mathcal{X}\times\mathcal{Y}} \ell(y,h(\boldsymbol{x}))P(\boldsymbol{x},y)\mathrm{d}\boldsymbol{x}\mathrm{d}y \\
&= \int_{\mathcal{X}} \left(\int_{\mathcal{Y}} \ell(y,h(\boldsymbol{x}))P(y\,|\,\boldsymbol{x})\mathrm{d}y \right) P(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \\
&= \mathbb{E}_{\boldsymbol{x}}\left[L_\ell(h\,|\,\boldsymbol{x})\right].
\end{aligned}$$

Decomposition of risk

$$
\begin{aligned}
L_\ell(h) &= \mathbb{E}_{(\boldsymbol{x},y)}\left[\ell(y,h(\boldsymbol{x}))\right] \\
&= \int_{\mathcal{X}\times\mathcal{Y}} \ell(y,h(\boldsymbol{x}))P(\boldsymbol{x},y)\mathrm{d}\boldsymbol{x}\mathrm{d}y \\
&= \int_{\mathcal{X}} \left(\int_{\mathcal{Y}} \ell(y,h(\boldsymbol{x}))P(y\,|\,\boldsymbol{x})\mathrm{d}y \right) P(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \\
&= \mathbb{E}_{\boldsymbol{x}}\left[L_\ell(h\,|\,\boldsymbol{x})\right].
\end{aligned}
$$

- $L_\ell(h\,|\,\boldsymbol{x})$ is the **conditional risk** of $\hat{y} = h(\boldsymbol{x})$ at $\boldsymbol{x}$.

Decomposition of risk

$$\begin{aligned} L_\ell(h) &= \mathbb{E}_{(\boldsymbol{x},y)}\left[\ell(y,h(\boldsymbol{x}))\right] \\ &= \int_{\mathcal{X}\times\mathcal{Y}} \ell(y,h(\boldsymbol{x}))P(\boldsymbol{x},y)\mathrm{d}\boldsymbol{x}\mathrm{d}y \\ &= \int_{\mathcal{X}} \left(\int_{\mathcal{Y}} \ell(y,h(\boldsymbol{x}))P(y\,|\,\boldsymbol{x})\mathrm{d}y \right) P(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \\ &= \mathbb{E}_{\boldsymbol{x}}\left[L_\ell(h\,|\,\boldsymbol{x})\right]. \end{aligned}$$

- $L_\ell(h\,|\,\boldsymbol{x})$ is the **conditional risk** of $\hat{y} = h(\boldsymbol{x})$ at $\boldsymbol{x}$.
- Bayes prediction **minimizes the conditional risk** for every $\boldsymbol{x}$:

$$h^*(\boldsymbol{x}) = \arg\min_h L_\ell(h\,|\,\boldsymbol{x}).$$

Example

- Pack of cards: 7 diamonds (red), 5 hearts (red), 5 spades (black), 3 clubs (black).

Example

- Pack of cards: 7 diamonds (red), 5 hearts (red), 5 spades (black), 3 clubs (black).
- Decision = bet (four choices).

Example

- Pack of cards: 7 diamonds (red), 5 hearts (red), 5 spades (black), 3 clubs (black).
- Decision = bet (four choices).
- If you win you get 100$, if you loose you must give 50$.

Making optimal decisions

Example

- Pack of cards: 7 diamonds (red), 5 hearts (red), 5 spades (black), 3 clubs (black).
- Decision = bet (four choices).
- If you win you get 100$, if you loose you must give 50$.
- What is the loss and optimal decision?

Example

- Pack of cards: 7 diamonds (red), 5 hearts (red), 5 spades (black), 3 clubs (black).
- Decision = bet (four choices).
- If you win you get 100$, if you loose you must give 50$.
- What is the loss and optimal decision?
- Suppose we know the card is black. What is the optimal decision now?

Example

- Pack of cards: 7 diamonds (red), 5 hearts (red), 5 spades (black), 3 clubs (black).
- Decision = bet (four choices).
- If you win you get 100$, if you loose you must give 50$.
- What is the loss and optimal decision?
- Suppose we know the card is black. What is the optimal decision now?
- What are the input variables?

Example

- Pack of cards: 7 diamonds (red), 5 hearts (red), 5 spades (black), 3 clubs (black).

Making optimal decisions

Example

- Pack of cards: 7 diamonds (red), 5 hearts (red), 5 spades (black), 3 clubs (black).
- Bet the color:

Making optimal decisions

Example

- Pack of cards: 7 diamonds (red), 5 hearts (red), 5 spades (black), 3 clubs (black).
- Bet the color:
  - if the true color is red and you are correct you win 50, otherwise you loose 100,

Making optimal decisions

Example

- Pack of cards: 7 diamonds (red), 5 hearts (red), 5 spades (black), 3 clubs (black).
- Bet the color:
  - if the true color is red and you are correct you win 50, otherwise you loose 100,
  - if the true color is black and you are correct you win 200, otherwise you loose 100.

Making optimal decisions

Example

- Pack of cards: 7 diamonds (red), 5 hearts (red), 5 spades (black), 3 clubs (black).
- Bet the color:
  - if the true color is red and you are correct you win 50, otherwise you loose 100,
  - if the true color is black and you are correct you win 200, otherwise you loose 100.
- What is the loss and optimal decision now?

Regression

- Prediction of a **real-valued** outcome $y \in \mathbb{R}$.
- Find a prediction function $h(\boldsymbol{x})$ that accurately predicts value of y.
- The most common loss function used is **squared error loss**:

$$\ell_{se}(y, \hat{y}) = (y - \hat{y})^2,$$

where $\hat{y} = h(\boldsymbol{x})$.

Regression

- The conditional risk for squared error loss is :

Regression

- The conditional risk for squared error loss is :

$$L_{se}(h \mid \boldsymbol{x}) = \mathbb{E}_{y|\boldsymbol{x}}\left[(y - \hat{y})^2\right]$$

Regression

- The conditional risk for squared error loss is :

$\mu(\boldsymbol{x}) = \mathbb{E}_{y \,|\, \boldsymbol{x}}[y]$

$$
\begin{aligned}
L_{se}(h \,|\, \boldsymbol{x}) &= \mathbb{E}_{y|\boldsymbol{x}}\left[(y - \hat{y})^2\right] \\
&= \mathbb{E}_{y|\boldsymbol{x}}\left[(y - \mu(\boldsymbol{x}) + \mu(\boldsymbol{x}) - \hat{y})^2\right]
\end{aligned}
$$

- The conditional risk for squared error loss is :

$\mu(\boldsymbol{x}) = \mathbb{E}_{y \mid \boldsymbol{x}}[y]$

$$\begin{aligned} L_{se}(h \mid \boldsymbol{x}) &= \mathbb{E}_{y|\boldsymbol{x}}\left[(y-\hat{y})^2\right] \\ &= \mathbb{E}_{y|\boldsymbol{x}}\left[(y-\mu(\boldsymbol{x})+\mu(\boldsymbol{x})-\hat{y})^2\right] \\ &= \mathbb{E}_{y|\boldsymbol{x}}\Big[(y-\mu(\boldsymbol{x}))^2 + 2\underbrace{(y-\mu(\boldsymbol{x}))}_{=0 \text{ under expectation}}(\mu(\boldsymbol{x})-\hat{y}) + (\mu(\boldsymbol{x})-\hat{y})^2\Big] \end{aligned}$$

- The conditional risk for squared error loss is :

$\mu(\boldsymbol{x}) = \mathbb{E}_{y\,|\,\boldsymbol{x}}[y]$

$$
\begin{aligned}
L_{se}(h \mid \boldsymbol{x}) &= \mathbb{E}_{y|\boldsymbol{x}}\left[(y-\hat{y})^2\right] \\
&= \mathbb{E}_{y|\boldsymbol{x}}\left[(y-\mu(\boldsymbol{x})+\mu(\boldsymbol{x})-\hat{y})^2\right] \\
&= \mathbb{E}_{y|\boldsymbol{x}}\Big[(y-\mu(\boldsymbol{x}))^2 + 2\underbrace{(y-\mu(\boldsymbol{x}))}_{=0 \text{ under expectation}}(\mu(\boldsymbol{x})-\hat{y}) + (\mu(\boldsymbol{x})-\hat{y})^2\Big] \\
&= \underbrace{\mathbb{E}_{y|\boldsymbol{x}}\left[(y-\mu(\boldsymbol{x}))^2\right]}_{\text{independent of } \hat{y}} + (\mu(\boldsymbol{x})-\hat{y})^2.
\end{aligned}
$$

Regression

- The conditional risk for squared error loss is :

$\mu(\boldsymbol{x}) = \mathbb{E}_{y \mid \boldsymbol{x}}[y]$

$$
\begin{aligned}
L_{se}(h \mid \boldsymbol{x}) &= \mathbb{E}_{y|\boldsymbol{x}}\left[(y-\hat{y})^2\right] \\
&= \mathbb{E}_{y|\boldsymbol{x}}\left[(y-\mu(\boldsymbol{x})+\mu(\boldsymbol{x})-\hat{y})^2\right] \\
&= \mathbb{E}_{y|\boldsymbol{x}}\Big[(y-\mu(\boldsymbol{x}))^2 + 2\underbrace{(y-\mu(\boldsymbol{x}))}_{=0 \text{ under expectation}}(\mu(\boldsymbol{x})-\hat{y}) + (\mu(\boldsymbol{x})-\hat{y})^2\Big] \\
&= \underbrace{\mathbb{E}_{y|\boldsymbol{x}}\left[(y-\mu(\boldsymbol{x}))^2\right]}_{\text{independent of } \hat{y}} + (\mu(\boldsymbol{x})-\hat{y})^2.
\end{aligned}
$$

- Hence, $h^*(\boldsymbol{x}) = \mu(\boldsymbol{x})$, the **conditional expectation** of y at $\boldsymbol{x}$, and:

$$L_{se}(h^* \mid \boldsymbol{x}) = \mathbb{E}_{y|\boldsymbol{x}}\left[(y-\mu(\boldsymbol{x}))^2\right] = \mathrm{Var}(y|\boldsymbol{x}).$$

- Another loss commonly used in regression is the **absolute error**:

$$\ell_{ae}(y, \hat{y}) = |y - \hat{y}|.$$

- The Bayes classifier for the absolute-error loss is:

$$h^*(\boldsymbol{x}) = \arg\min_h L_{ae}(h \,|\, \boldsymbol{x}) =$$

Regression

- Another loss commonly used in regression is the **absolute error**:

$$\ell_{ae}(y, \hat{y}) = |y - \hat{y}|.$$

- The Bayes classifier for the absolute-error loss is:

$$h^*(\boldsymbol{x}) = \arg\min_h L_{ae}(h \,|\, \boldsymbol{x}) = \mathrm{median}(y|\boldsymbol{x}) \,,$$

i.e., **median** of the conditional distribution of y given $\boldsymbol{x}$.

Regression

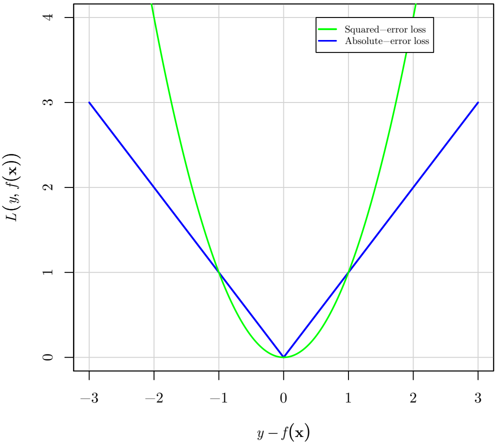


Figure: Loss functions for regression task

Binary Classification

- Prediction of a **binary** outcome $y \in \{-1, 1\}$ (alternatively $y \in \{0, 1\}$).
- Find a prediction function $h(\boldsymbol{x})$ that accurately predicts value of y.
- The most common loss function used is **0/1 loss**:

$$\ell_{0/1}(y, \hat{y}) = \begin{cases} 0, & \text{if } y = \hat{y}\,, \\ 1, & \text{otherwise}\,. \end{cases}$$

Binary Classification

- Define $\eta(\boldsymbol{x}) = P(y = 1|\boldsymbol{x})$.

Binary Classification

- Define $\eta(\boldsymbol{x}) = P(y = 1|\boldsymbol{x})$.
- The conditional 0/1 risk at $\boldsymbol{x}$ is:

- Define $\eta(\boldsymbol{x}) = P(y = 1|\boldsymbol{x})$.
- The conditional 0/1 risk at $\boldsymbol{x}$ is:

$$L_{0/1}(h|\boldsymbol{x}) = \eta(\boldsymbol{x})[\![h(\boldsymbol{x}) = -1]\!] + (1 - \eta(\boldsymbol{x}))[\![h(\boldsymbol{x}) = 1]\!].$$

Binary Classification

- Define $\eta(\boldsymbol{x}) = P(y = 1|\boldsymbol{x})$.
- The conditional 0/1 risk at $\boldsymbol{x}$ is:

$$L_{0/1}(h|\boldsymbol{x}) = \eta(\boldsymbol{x})[\![h(\boldsymbol{x}) = -1]\!] + (1 - \eta(\boldsymbol{x}))[\![h(\boldsymbol{x}) = 1]\!].$$

- The **Bayes classifier**:

- Define $\eta(\boldsymbol{x}) = P(y = 1|\boldsymbol{x})$.
- The conditional 0/1 risk at $\boldsymbol{x}$ is:

$$L_{0/1}(h|\boldsymbol{x}) = \eta(\boldsymbol{x})[\![h(\boldsymbol{x}) = -1]\!] + (1 - \eta(\boldsymbol{x}))[\![h(\boldsymbol{x}) = 1]\!].$$

- The **Bayes classifier**:

$$h^*(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \eta(\boldsymbol{x}) > 1 - \eta(\boldsymbol{x}) \\ -1 & \text{if } \eta(\boldsymbol{x}) < 1 - \eta(\boldsymbol{x}) \end{cases} = \operatorname{sgn}\left(\eta(\boldsymbol{x}) - 1/2\right),$$

and the **Bayes conditional risk**:

- Define $\eta(\boldsymbol{x}) = P(y = 1|\boldsymbol{x})$.
- The conditional 0/1 risk at $\boldsymbol{x}$ is:

$$L_{0/1}(h|\boldsymbol{x}) = \eta(\boldsymbol{x})[\![h(\boldsymbol{x}) = -1]\!] + (1 - \eta(\boldsymbol{x}))[\![h(\boldsymbol{x}) = 1]\!].$$

- The **Bayes classifier**:

$$h^*(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \eta(\boldsymbol{x}) > 1 - \eta(\boldsymbol{x}) \\ -1 & \text{if } \eta(\boldsymbol{x}) < 1 - \eta(\boldsymbol{x}) \end{cases} = \operatorname{sgn}\left(\eta(\boldsymbol{x}) - 1/2\right),$$

and the **Bayes conditional risk**:

$$L_\ell(h^* \,|\, \boldsymbol{x}) = \min\{\eta(\boldsymbol{x}), 1 - \eta(\boldsymbol{x})\}.$$

Multi-class classification

- **Domain** of outcome variable y is a set of labels $\mathcal{Y} = \{1, \ldots, K\}$.
- **Goal**: find a prediction function $h(\boldsymbol{x})$ that for any object $\boldsymbol{x}$ predicts accurately the actual value of y.
- **Loss function**: the most common is 0/1 loss:

$$\ell_{0/1}(y, \hat{y}) = \begin{cases} 0, & \text{if } y = \hat{y}\,, \\ 1, & \text{otherwise}\,. \end{cases}$$

Multi-class classification

- The conditional risk of the 0/1 loss is:

$$L_{0/1}(h \,|\, \boldsymbol{x}) \;\; = \;\; \mathbb{E}_{y|\boldsymbol{x}} \ell_{0/1}(y, h(\boldsymbol{x}))$$

Multi-class classification

- The conditional risk of the 0/1 loss is:

$$\begin{aligned} L_{0/1}(h \,|\, \boldsymbol{x}) &= \mathbb{E}_{y|\boldsymbol{x}} \ell_{0/1}(y, h(\boldsymbol{x})) \\ &= \sum_{k \in \mathcal{Y}} P(y = k|\boldsymbol{x}) \ell_{0/1}(k, h(\boldsymbol{x})) \end{aligned}$$

Multi-class classification

- The conditional risk of the 0/1 loss is:

$$
\begin{aligned}
L_{0/1}(h \,|\, \boldsymbol{x}) &= \mathbb{E}_{y|\boldsymbol{x}} \ell_{0/1}(y, h(\boldsymbol{x})) \\
&= \sum_{k \in \mathcal{Y}} P(y = k|\boldsymbol{x}) \ell_{0/1}(k, h(\boldsymbol{x}))
\end{aligned}
$$

- Therefore, the Bayes classifier is given by:

$$
h^*(\boldsymbol{x}) = \arg\min_{h} L_{0/1}(h \,|\, \boldsymbol{x})
$$

Multi-class classification

- The conditional risk of the 0/1 loss is:

$$
\begin{aligned}
L_{0/1}(h \,|\, \boldsymbol{x}) &= \mathbb{E}_{y|\boldsymbol{x}} \ell_{0/1}(y, h(\boldsymbol{x})) \\
&= \sum_{k \in \mathcal{Y}} P(y = k|\boldsymbol{x}) \ell_{0/1}(k, h(\boldsymbol{x}))
\end{aligned}
$$

- Therefore, the Bayes classifier is given by:

$$
\begin{aligned}
h^*(\boldsymbol{x}) &= \arg\min_h L_{0/1}(h \,|\, \boldsymbol{x}) \\
&= \arg\max_k P(y = k|\boldsymbol{x}) \,,
\end{aligned}
$$

the class with **the largest conditional probability** $P(y|\boldsymbol{x})$.

Multi-class classification

- The conditional risk of the 0/1 loss is:

$$
\begin{aligned}
L_{0/1}(h \,|\, \boldsymbol{x}) &= \mathbb{E}_{y|\boldsymbol{x}} \ell_{0/1}(y, h(\boldsymbol{x})) \\
&= \sum_{k \in \mathcal{Y}} P(y = k|\boldsymbol{x}) \ell_{0/1}(k, h(\boldsymbol{x}))
\end{aligned}
$$

- Therefore, the Bayes classifier is given by:

$$
\begin{aligned}
h^*(\boldsymbol{x}) &= \arg\min_h L_{0/1}(h \,|\, \boldsymbol{x}) \\
&= \arg\max_k P(y = k|\boldsymbol{x}) \,,
\end{aligned}
$$

  the class with **the largest conditional probability** $P(y|\boldsymbol{x})$.

- The Bayes conditional risk:

Multi-class classification

- The conditional risk of the 0/1 loss is:

$$\begin{aligned} L_{0/1}(h \,|\, \boldsymbol{x}) &= \mathbb{E}_{y|\boldsymbol{x}} \ell_{0/1}(y, h(\boldsymbol{x})) \\ &= \sum_{k \in \mathcal{Y}} P(y = k|\boldsymbol{x}) \ell_{0/1}(k, h(\boldsymbol{x})) \end{aligned}$$

- Therefore, the Bayes classifier is given by:

$$\begin{aligned} h^*(\boldsymbol{x}) &= \arg\min_h L_{0/1}(h \,|\, \boldsymbol{x}) \\ &= \arg\max_k P(y = k|\boldsymbol{x}) \,, \end{aligned}$$

  the class with **the largest conditional probability** $P(y|\boldsymbol{x})$.
- The Bayes conditional risk:

$$L_\ell(h^* \,|\, \boldsymbol{x}) = \min\{1 - P(y = k|\boldsymbol{x}) : k \in \mathcal{Y}\}.$$

Deterministic learning framework

- **Input** $\boldsymbol{x} \in \mathcal{X}$ drawn from a distribution $P(\boldsymbol{x})$.
- **Outcome** $y \in \mathcal{Y}$.
- **Unknown target function** $h^*\colon \mathcal{X} \to \mathcal{Y}$, such that $y = h^*(\boldsymbol{x})$.
- **Goal**: discover h^* by observing examples of $(\boldsymbol{x}, y)$.

Deterministic learning framework

- **Input** $\boldsymbol{x} \in \mathcal{X}$ drawn from a distribution $P(\boldsymbol{x})$.
- **Outcome** $y \in \mathcal{Y}$.
- **Unknown target function** $h^*\colon \mathcal{X} \to \mathcal{Y}$, such that $y = h^*(\boldsymbol{x})$.
- **Goal**: discover h^* by observing examples of $(\boldsymbol{x}, y)$.

- This is a **special case** of the statistical framework:
  - What is $P(y|\boldsymbol{x})$?
  - Bayes prediction function?
  - Risk of h^*? (assuming $\ell(y, \hat{y}) = 0$ whenever $y = \hat{y}$)

Deterministic learning framework

- **Input** $\boldsymbol{x} \in \mathcal{X}$ drawn from a distribution $P(\boldsymbol{x})$.
- **Outcome** $y \in \mathcal{Y}$.
- **Unknown target function** $h^*\colon \mathcal{X} \to \mathcal{Y}$, such that $y = h^*(\boldsymbol{x})$.
- **Goal**: discover h^* by observing examples of $(\boldsymbol{x}, y)$.

- This is a **special case** of the statistical framework:
  - What is $P(y|\boldsymbol{x})$?
    - $P(y|\boldsymbol{x})$ is a **degenerate** distribution for every $\boldsymbol{x}$.
  - Bayes prediction function?
  - Risk of h^*? (assuming $\ell(y, \hat{y}) = 0$ whenever $y = \hat{y}$)

Deterministic learning framework

- **Input** $\boldsymbol{x} \in \mathcal{X}$ drawn from a distribution $P(\boldsymbol{x})$.
- **Outcome** $y \in \mathcal{Y}$.
- **Unknown target function** $h^*\colon \mathcal{X} \to \mathcal{Y}$, such that $y = h^*(\boldsymbol{x})$.
- **Goal**: discover h^* by observing examples of $(\boldsymbol{x}, y)$.

- This is a **special case** of the statistical framework:
  - What is $P(y|\boldsymbol{x})$?
    - $P(y|\boldsymbol{x})$ is a **degenerate** distribution for every $\boldsymbol{x}$.
  - Bayes prediction function?
    - h^*
  - Risk of h^*? (assuming $\ell(y, \hat{y}) = 0$ whenever $y = \hat{y}$)

Deterministic learning framework

- **Input** $\boldsymbol{x} \in \mathcal{X}$ drawn from a distribution $P(\boldsymbol{x})$.
- **Outcome** $y \in \mathcal{Y}$.
- **Unknown target function** $h^*\colon \mathcal{X} \to \mathcal{Y}$, such that $y = h^*(\boldsymbol{x})$.
- **Goal**: discover h^* by observing examples of $(\boldsymbol{x}, y)$.

- This is a **special case** of the statistical framework:
  - What is $P(y|\boldsymbol{x})$?
    - $P(y|\boldsymbol{x})$ is a **degenerate** distribution for every $\boldsymbol{x}$.
  - Bayes prediction function?
    - h^*
  - Risk of h^*? (assuming $\ell(y, \hat{y}) = 0$ whenever $y = \hat{y}$)
    - h^* has **zero risk**.

Deterministic learning framework

- **Input** $\boldsymbol{x} \in \mathcal{X}$ drawn from a distribution $P(\boldsymbol{x})$.
- **Outcome** $y \in \mathcal{Y}$.
- **Unknown target function** $h^*\colon \mathcal{X} \to \mathcal{Y}$, such that $y = h^*(\boldsymbol{x})$.
- **Goal**: discover h^* by observing examples of $(\boldsymbol{x}, y)$.

- This is a **special case** of the statistical framework:
  - What is $P(y|\boldsymbol{x})$?
    - $P(y|\boldsymbol{x})$ is a **degenerate** distribution for every $\boldsymbol{x}$.
  - Bayes prediction function?
    - h^*
  - Risk of h^*? (assuming $\ell(y, \hat{y}) = 0$ whenever $y = \hat{y}$)
    - h^* has **zero risk**.
  - Unrealistic scenario in real life.

Outline

Learning

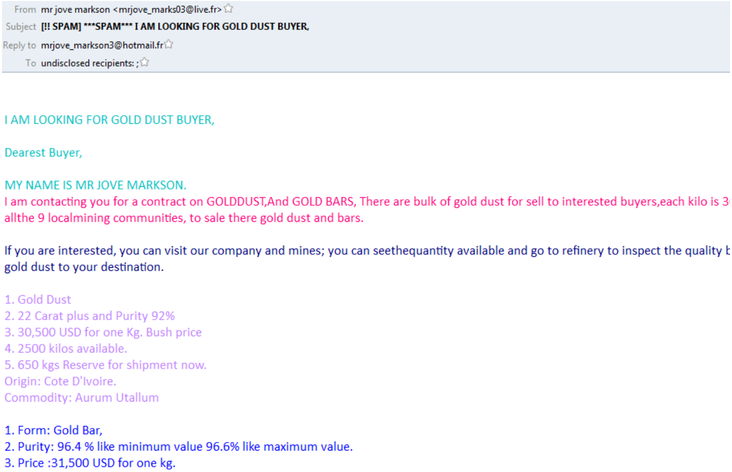
- Distribution $P(\boldsymbol{x}, y)$ is unknown **unknown**.
- Therefore, Bayes classifier h^* is also **unknown**.
- Instead, we have access to n independent and identically distributed (i.i.d) **training examples** (**sample**):

$$\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)\}.$$

- **Learning**: use training data to find a good **approximation** of h^*.

Spam filtering

- Problem: Predict whether a given email is spam or not.
- An object to be classified: an email.
- There are two possible responses (classes): spam, not spam.


From mr jove markson <mrjove_marks03@live.fr>
Subject **[!! SPAM] ***SPAM*** I AM LOOKING FOR GOLD DUST BUYER,**
Reply to mrjove_markson3@hotmail.fr
To undisclosed recipients: ;

I AM LOOKING FOR GOLD DUST BUYER,

Dearest Buyer,

MY NAME IS MR JOVE MARKSON.
I am contacting you for a contract on GOLDDUST,And GOLD BARS, There are bulk of gold dust for sell to interested buyers,each kilo is 3
allthe 9 localmining communities, to sale there gold dust and bars.

If you are interested, you can visit our company and mines; you can seethequantity available and go to refinery to inspect the quality b
gold dust to your destination.

1. Gold Dust
2. 22 Carat plus and Purity 92%
3. 30,500 USD for one Kg. Bush price
4. 2500 kilos available.
5. 650 kgs Reserve for shipment now.
Origin: Cote D'Ivoire.
Commodity: Aurum Utallum

1. Form: Gold Bar,
2. Purity: 96.4 % like minimum value 96.6% like maximum value.
3. Price :31,500 USD for one kg.

Spam filtering

Example

- Representation of an email through (meaningful) features:

Spam filtering

Example

- Representation of an email through (meaningful) features:
  - length of subject
  - length of email body,
  - use of colors,
  - domain,
  - words in subject,
  - words in body.

length of subject	length of body	use of colors	domain	gold	price	USD	...	machine	learning	spam?
7	240	1	live.fr	1	1	1	...	0	0	1
2	150	0	poznan.pl	0	0	0	...	1	1	0
2	250	0	tibco.com	0	1	1	...	1	1	0
4	120	1	r-project.org	0	1	0	...	0	0	?

Learning

- Four types of datasets:
  - **training** data: historical emails,
  - **validation** data: a subset of historical emails used for tuning learning algorithms
  - **test** data: a subset of historical emails used for estimating the risk,
  - **new incoming** data to be classified: new incoming emails.

Different learning paradigms

Different learning paradigms

- **Generative learning**

Different learning paradigms

- **Generative learning**
  - Follow a data generating process
  - Learn a model of the joint distribution $P(\boldsymbol{x}, y)$ and then use the Bayes theorem to obtain $P(y \,|\, \boldsymbol{x})$.
  - Make the final prediction by computing the optimal decision based on $P(y \,|\, \boldsymbol{x})$ with respect to a given $\ell(y, \hat{y})$.

Different learning paradigms

- **Generative learning**
  - Follow a data generating process
  - Learn a model of the joint distribution $P(\boldsymbol{x}, y)$ and then use the Bayes theorem to obtain $P(y \,|\, \boldsymbol{x})$.
  - Make the final prediction by computing the optimal decision based on $P(y \,|\, \boldsymbol{x})$ with respect to a given $\ell(y, \hat{y})$.
- **Discriminative learning**

Different learning paradigms

- **Generative learning**
  - Follow a data generating process
  - Learn a model of the joint distribution $P(\boldsymbol{x}, y)$ and then use the Bayes theorem to obtain $P(y \mid \boldsymbol{x})$.
  - Make the final prediction by computing the optimal decision based on $P(y \mid \boldsymbol{x})$ with respect to a given $\ell(y, \hat{y})$.
- **Discriminative learning**
  - Approximate $h^*(\boldsymbol{x})$ which is a direct map from $\boldsymbol{x}$ to y or
  - Model the conditional probability $P(y \mid \boldsymbol{x})$ directly, and
  - Make the final prediction by computing the optimal decision based on $P(y \mid \boldsymbol{x})$ with respect to a given $\ell(y, \hat{y})$.

Different learning paradigms

- **Generative learning**
  - Follow a data generating process
  - Learn a model of the joint distribution $P(\boldsymbol{x}, y)$ and then use the Bayes theorem to obtain $P(y \,|\, \boldsymbol{x})$.
  - Make the final prediction by computing the optimal decision based on $P(y \,|\, \boldsymbol{x})$ with respect to a given $\ell(y, \hat{y})$.
- **Discriminative learning**
  - Approximate $h^*(\boldsymbol{x})$ which is a direct map from $\boldsymbol{x}$ to y or
  - Model the conditional probability $P(y \,|\, \boldsymbol{x})$ directly, and
  - Make the final prediction by computing the optimal decision based on $P(y \,|\, \boldsymbol{x})$ with respect to a given $\ell(y, \hat{y})$.
- **Two phases** of the learning models: learning and prediction (inference).

Different learning paradigms

- Various principles on how to learn:
  - Empirical risk minimization,
  - Maximum likelihood principle,
  - Bayes approach,
  - Minimum description length,
  - ...

Empirical Risk Minimization (ERM)

- Choose a prediction function $\widehat{h}$ which minimizes the loss on the training data within some **restricted** class of functions $\mathcal{H}$.

$$\widehat{h} = \arg\min_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, h(\boldsymbol{x}_i)).$$

- The average loss on the training data is called **empirical risk** $\widehat{L}_\ell(h)$.

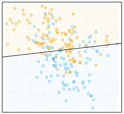
1

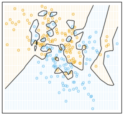
Empirical Risk Minimization (ERM)

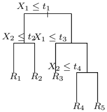
- Choose a prediction function $\widehat{h}$ which minimizes the loss on the training data within some **restricted** class of functions $\mathcal{H}$.

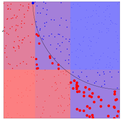
$$\widehat{h} = \arg\min_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, h(\boldsymbol{x}_i)).$$

- The average loss on the training data is called **empirical risk** $\widehat{L}_\ell(h)$.
- $\mathcal{H}$ can be: linear functions, polynomials, trees of a given depth, rules, linear combinations of trees, etc.[1]









[1] T. Hastie, R. Tibshirani, and J.H. Friedman. *Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, second edition, 2009

Outline

Selected learning methods

- Almost no-learning methods: histogram-based classifier, nearest neighbors
- Generative methods: naive Bayes
- Linear methods and their generalizations: linear regression, logistic regression, perceptron, support vector machines, AdaBoost, neural networks.

Almost no-learning methods

- Based on empirical distribution and direct application of the Bayes rule to a local estimate of $P(y \,|\, \boldsymbol{x})$.

Almost no-learning methods

- Based on empirical distribution and direct application of the Bayes rule to a local estimate of $P(y \mid \boldsymbol{x})$.
- The simplest approach estimates conditional probabilities $P(y|\boldsymbol{x})$ for any $\boldsymbol{x}$ from training data:
  - Based on *group-bys* and simple counting.
  - Needs a lot of data to get reasonable estimates!!!
  - Data should be discrete/nominal or we need to discretize numerical data before.

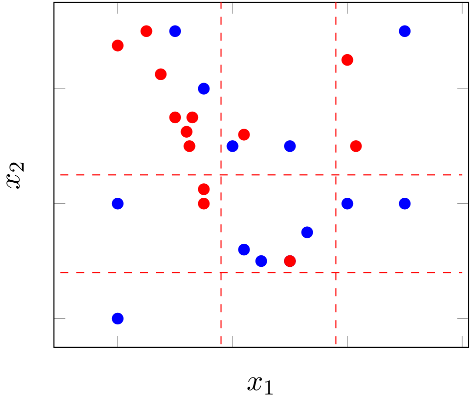
Example

gold	price	spam?
1	1	1
1	1	1
1	1	1
1	1	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	1
0	0	1
0	1	0
0	1	1

$$
\begin{aligned}
P(y=1|\text{gold}=1 \wedge \text{price}=1) &= 0.75 \\
P(y=0|\text{gold}=1 \wedge \text{price}=1) &= 0.25 \\
P(y=1|\text{gold}=0 \wedge \text{price}=0) &= 0.33 \\
P(y=0|\text{gold}=0 \wedge \text{price}=0) &= 0.66 \\
P(y=1|\text{gold}=0 \wedge \text{price}=1) &= 0.5 \\
P(y=0|\text{gold}=0 \wedge \text{price}=1) &= 0.5 \\
P(y=1|\text{gold}=1 \wedge \text{price}=0) &= ? \\
P(y=0|\text{gold}=1 \wedge \text{price}=0) &= ?
\end{aligned}
$$

Histogram-based methods

- Build a multidimensional grid and estimate the conditional probability in each element of the grid,
- Plug the estimates to the Bayes classifier for a given $\ell(y, \hat{y})$ to obtain prediction.



Histogram-based methods

- The predictive performance depends on the grid resolution, dimensionality of data and the size of training data.

Histogram-based methods

- The predictive performance depends on the grid resolution, dimensionality of data and the size of training data.
- With some tricks can be efficiently implemented.

Histogram-based methods

- The predictive performance depends on the grid resolution, dimensionality of data and the size of training data.
- With some tricks can be efficiently implemented.
- Piecewise-constant prediction for a given region.

Histogram-based methods

- The predictive performance depends on the grid resolution, dimensionality of data and the size of training data.
- With some tricks can be efficiently implemented.
- Piecewise-constant prediction for a given region.
- Computation of the estimates in the region: well-know statistical problem, maximum likelihood estimates, regularization.

Histogram-based methods

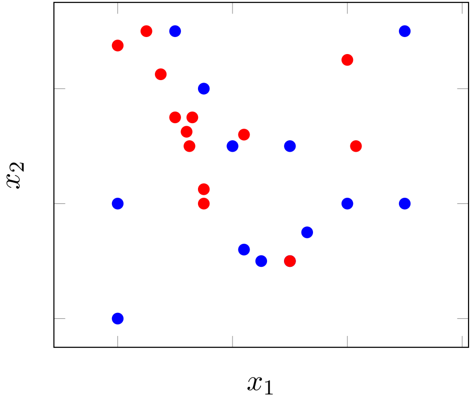
- The predictive performance depends on the grid resolution, dimensionality of data and the size of training data.
- With some tricks can be efficiently implemented.
- Piecewise-constant prediction for a given region.
- Computation of the estimates in the region: well-know statistical problem, maximum likelihood estimates, regularization.
- The grid can be given as a domain knowledge, simple discretization, or random splits.

Histogram-based methods

- The predictive performance depends on the grid resolution, dimensionality of data and the size of training data.
- With some tricks can be efficiently implemented.
- Piecewise-constant prediction for a given region.
- Computation of the estimates in the region: well-know statistical problem, maximum likelihood estimates, regularization.
- The grid can be given as a domain knowledge, simple discretization, or random splits.
- One can use more intelligent methods to obtain a grid, for example, supervised discretization or supervised recursive splitting like in decision trees.

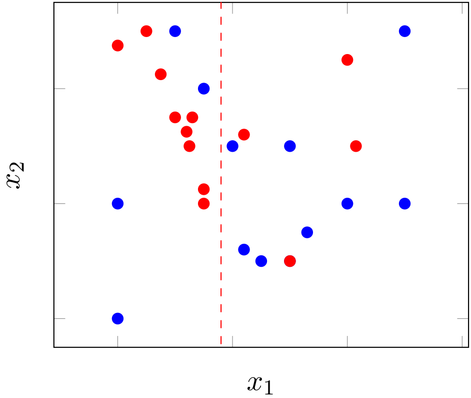
Decision trees

- Recursively make a partition of the feature space (in a smart way),
- Compute the optimal decision in each region.



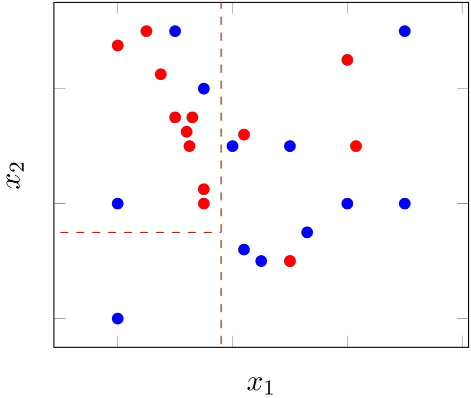
Decision trees

- Recursively make a partition of the feature space (in a smart way),
- Compute the optimal decision in each region.



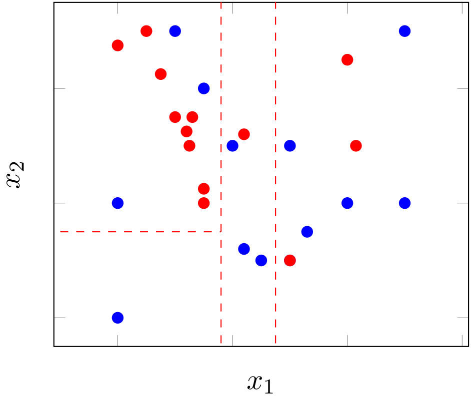
Decision trees

- Recursively make a partition of the feature space (in a smart way),
- Compute the optimal decision in each region.



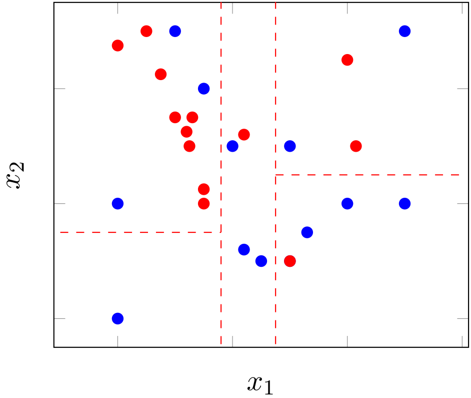
Decision trees

- Recursively make a partition of the feature space (in a smart way),
- Compute the optimal decision in each region.

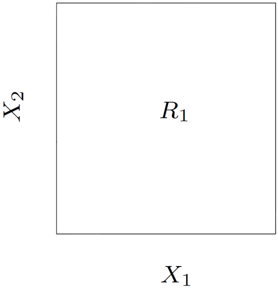


Decision trees

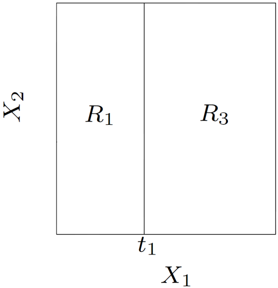
- Recursively make a partition of the feature space (in a smart way),
- Compute the optimal decision in each region.

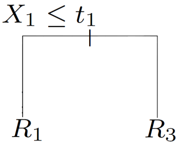


Decision trees

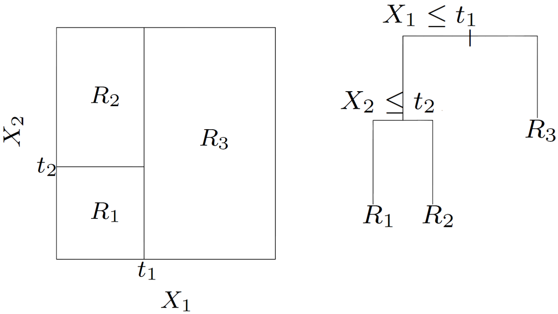


Decision trees

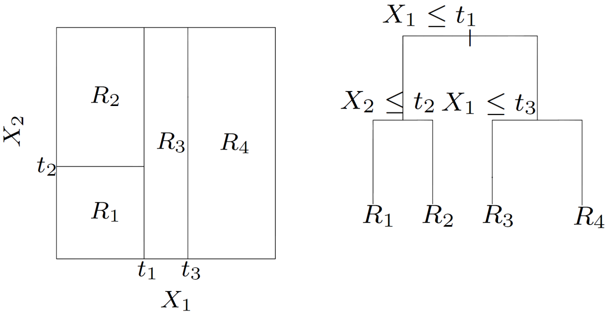




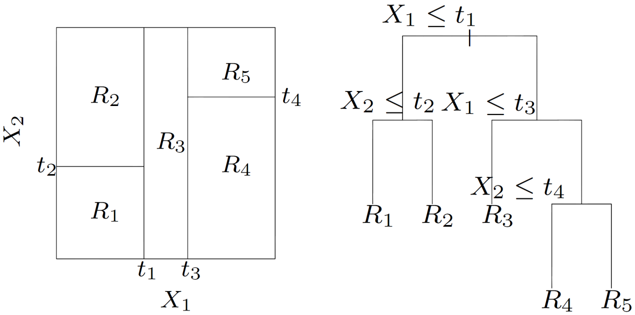
Decision trees



Decision trees



Decision trees



Decision trees

- The learning method seeks an optimal tree shape (e.g. feature space partition) by minimizing the empirical risk (usually expressed in terms of a surrogate loss).

Decision trees

- The learning method seeks an optimal tree shape (e.g. feature space partition) by minimizing the empirical risk (usually expressed in terms of a surrogate loss).

Question

How to design the splitting criterion for the squared-error loss and 0/1 loss?

Decision trees

- The learning method seeks an optimal tree shape (e.g. feature space partition) by minimizing the empirical risk (usually expressed in terms of a surrogate loss).

Question

How to design the splitting criterion for the squared-error loss and 0/1 loss?

- Greedy methods used for constructing a tree.

Decision trees

- The learning method seeks an optimal tree shape (e.g. feature space partition) by minimizing the empirical risk (usually expressed in terms of a surrogate loss).

Question

How to design the splitting criterion for the squared-error loss and 0/1 loss?

- Greedy methods used for constructing a tree.
- The resulting model can be easily interpreted.

- The learning method seeks an optimal tree shape (e.g. feature space partition) by minimizing the empirical risk (usually expressed in terms of a surrogate loss).

Question

How to design the splitting criterion for the squared-error loss and 0/1 loss?

- Greedy methods used for constructing a tree.
- The resulting model can be easily interpreted.
- The most influential splits are close to the root (like in the 20-question game).

Decision trees

- The learning method seeks an optimal tree shape (e.g. feature space partition) by minimizing the empirical risk (usually expressed in terms of a surrogate loss).

Question

How to design the splitting criterion for the squared-error loss and 0/1 loss?

- Greedy methods used for constructing a tree.
- The resulting model can be easily interpreted.
- The most influential splits are close to the root (like in the 20-question game).
- Learning and prediction is very efficient.

Decision trees

- The learning method seeks an optimal tree shape (e.g. feature space partition) by minimizing the empirical risk (usually expressed in terms of a surrogate loss).

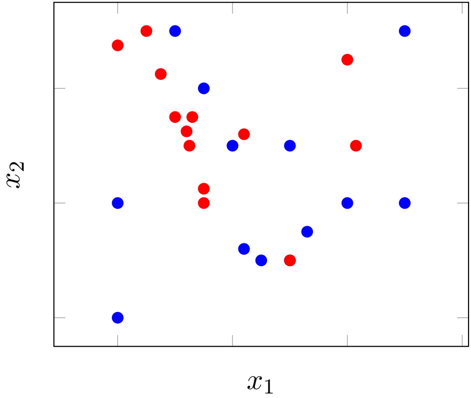
Question

How to design the splitting criterion for the squared-error loss and 0/1 loss?

- Greedy methods used for constructing a tree.
- The resulting model can be easily interpreted.
- The most influential splits are close to the root (like in the 20-question game).
- Learning and prediction is very efficient.
- Estimation of the decision in each leaf – the same problem like in histogram-based methods.

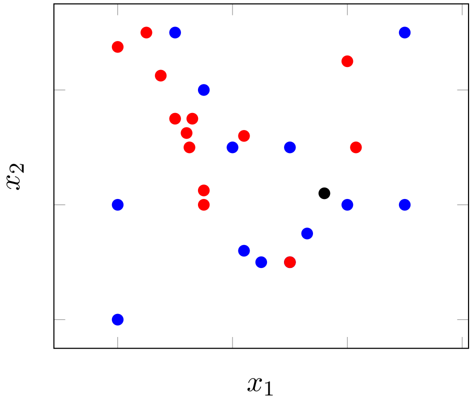
Nearest neighbor methods

- Find k-nearest neighbors of the test example according to a given metric,
- Estimate the Bayes classifier based on the neighborhood.



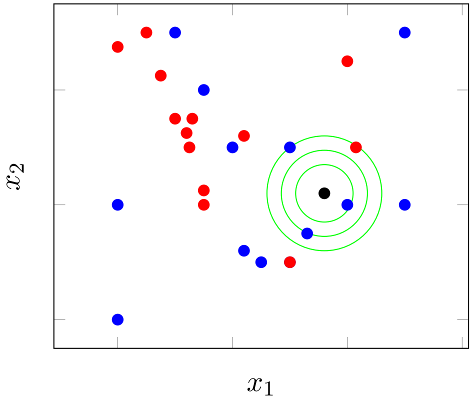
Nearest neighbor methods

- Find k-nearest neighbors of the test example according to a given metric,
- Estimate the Bayes classifier based on the neighborhood.



Nearest neighbor methods

- Find k-nearest neighbors of the test example according to a given metric,
- Estimate the Bayes classifier based on the neighborhood.



Nearest neighbor methods

- Prediction for a test example is computed based on nearest training examples – there is no learning.

Nearest neighbor methods

- Prediction for a test example is computed based on nearest training examples – there is no learning.
- The same principles for computing the prediction as in histogram-based and tree classifiers.

Nearest neighbor methods

- Prediction for a test example is computed based on nearest training examples – there is no learning.
- The same principles for computing the prediction as in histogram-based and tree classifiers.
- Training set can be used for tunning k and finding a metric.

Nearest neighbor methods

- Prediction for a test example is computed based on nearest training examples – there is no learning.
- The same principles for computing the prediction as in histogram-based and tree classifiers.
- Training set can be used for tunning k and finding a metric.
- Specialized data structures for efficient search of nearest neighbors.

Nearest neighbor methods

- Prediction for a test example is computed based on nearest training examples – there is no learning.
- The same principles for computing the prediction as in histogram-based and tree classifiers.
- Training set can be used for tunning k and finding a metric.
- Specialized data structures for efficient search of nearest neighbors.
- Reduction of training data: prototypes, feature selection, dimensionality reduction by PCA or similar methods.

Nearest neighbor methods

- Prediction for a test example is computed based on nearest training examples – there is no learning.
- The same principles for computing the prediction as in histogram-based and tree classifiers.
- Training set can be used for tunning k and finding a metric.
- Specialized data structures for efficient search of nearest neighbors.
- Reduction of training data: prototypes, feature selection, dimensionality reduction by PCA or similar methods.
- Approximate nearest neighbors.

Naive Bayes

- Generative methods rely on the Bayes theorem:

$$P(y = k|\boldsymbol{x}) = \frac{P(\boldsymbol{x}|y = k)P(y = k)}{P(\boldsymbol{x})}$$

  where $P(\boldsymbol{x}|y = k)$ is the density function $f_k(\boldsymbol{x})$ (for example, multivariate Gaussian distribution), and $P(\boldsymbol{x})$ is given by:

Naive Bayes

- Generative methods rely on the Bayes theorem:

$$P(y = k|\boldsymbol{x}) = \frac{P(\boldsymbol{x}|y = k)P(y = k)}{P(\boldsymbol{x})}$$

where $P(\boldsymbol{x}|y = k)$ is the density function $f_k(\boldsymbol{x})$ (for example, multivariate Gaussian distribution), and $P(\boldsymbol{x})$ is given by:

$$P(\boldsymbol{x}) = \sum_j P(\boldsymbol{x}|y = j)P(y = j)$$

from the law of total probability.

Learning

- The main algorithms:
  - Linear and quadratic discriminant analysis that use Gaussian densities,
  - General nonparametric density estimates for each class density,
  - Naive Bayes model that assumes that each of the class densities are products of marginal densities, i.e., the features are conditionally independent in each class.

Naive Bayes

- The naive Bayes model assumes that given a class $y = k$, the features $\boldsymbol{x} = (x_1, x_2, \ldots, x_m)$ are independent:

$$P(\boldsymbol{x}|y) =$$

Naive Bayes

- The naive Bayes model assumes that given a class $y = k$, the features $\boldsymbol{x} = (x_1, x_2, \ldots, x_m)$ are independent:

$$P(\boldsymbol{x}|y) = \prod_{j=1}^{m} P(x_j|y).$$

Naive Bayes

- The naive Bayes model assumes that given a class $y = k$, the features $\boldsymbol{x} = (x_1, x_2, \ldots, x_m)$ are independent:

$$P(\boldsymbol{x}|y) = \prod_{j=1}^{m} P(x_j|y).$$

- The model takes the following form:

$$P(y = k|\boldsymbol{x}) =$$

Naive Bayes

- The naive Bayes model assumes that given a class $y = k$, the features $\boldsymbol{x} = (x_1, x_2, \ldots, x_m)$ are independent:

$$P(\boldsymbol{x}|y) = \prod_{j=1}^{m} P(x_j|y).$$

- The model takes the following form:

$$P(y = k|\boldsymbol{x}) = \frac{P(y = k) \prod_{j=1}^{m} P(x_j|y = k)}{\sum_{k'} P(y = k') \prod_{j=1}^{m} P(x_j|y = k')}$$

Naive Bayes

- The naive Bayes model assumes that given a class $y = k$, the features $\boldsymbol{x} = (x_1, x_2, \ldots, x_m)$ are independent:

$$P(\boldsymbol{x}|y) = \prod_{j=1}^{m} P(x_j|y).$$

- The model takes the following form:

$$P(y = k|\boldsymbol{x}) = \frac{P(y = k) \prod_{j=1}^{m} P(x_j|y = k)}{\sum_{k'} P(y = k') \prod_{j=1}^{m} P(x_j|y = k')}$$

- The individual class-conditional marginal densities f_{jk} can each be estimated separately using univariate Gaussian distributions:

$$N(\mathbb{E}(x_j|y = k), \mathrm{Var}(x_j|y = k))$$

- If a component x_j of $\boldsymbol{x}$ is discrete, then an appropriate histogram estimate can be used.

Naive Bayes

Example

gold	price	spam?
1	1	1
1	1	1
1	1	1
1	1	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	1
0	0	1
0	1	0
0	1	1

$$P(y = 1) =$$

Example

gold	price	spam?
1	1	1
1	1	1
1	1	1
1	1	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	1
0	0	1
0	1	0
0	1	1

$P(y = 1) = 0.5$

$P(y = 0) =$

Naive Bayes

Example

gold	price	spam?
1	1	1
1	1	1
1	1	1
1	1	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	1
0	0	1
0	1	0
0	1	1

$P(y=1) = 0.5$

$P(y=0) = 0.5$

$P(\text{gold}=1|Y=1) =$

Naive Bayes

Example

gold	price	spam?
1	1	1
1	1	1
1	1	1
1	1	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	1
0	0	1
0	1	0
0	1	1

$P(y = 1) = 0.5$

$P(y = 0) = 0.5$

$P(\text{gold} = 1|Y = 1) = 0.5$

$P(\text{gold} = 0|Y = 1) =$

Naive Bayes

Example

gold	price	spam?
1	1	1
1	1	1
1	1	1
1	1	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	1
0	0	1
0	1	0
0	1	1

$$P(y = 1) = 0.5$$
$$P(y = 0) = 0.5$$
$$P(\text{gold} = 1 | Y = 1) = 0.5$$
$$P(\text{gold} = 0 | Y = 1) = 0.5$$
$$P(\text{gold} = 1 | Y = 0) =$$

Naive Bayes

Example

gold	price	spam?
1	1	1
1	1	1
1	1	1
1	1	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	1
0	0	1
0	1	0
0	1	1

$$P(y = 1) = 0.5$$
$$P(y = 0) = 0.5$$
$$P(\text{gold} = 1|Y = 1) = 0.5$$
$$P(\text{gold} = 0|Y = 1) = 0.5$$
$$P(\text{gold} = 1|Y = 0) = 0.17$$
$$P(\text{gold} = 0|Y = 0) =$$

Naive Bayes

Example

gold	price	spam?
1	1	1
1	1	1
1	1	1
1	1	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	1
0	0	1
0	1	0
0	1	1

$$P(y = 1) = 0.5$$
$$P(y = 0) = 0.5$$
$$P(\text{gold} = 1|Y = 1) = 0.5$$
$$P(\text{gold} = 0|Y = 1) = 0.5$$
$$P(\text{gold} = 1|Y = 0) = 0.17$$
$$P(\text{gold} = 0|Y = 0) = 0.83$$
$$P(\text{price} = 1|Y = 1) =$$

Naive Bayes

Example

gold	price	spam?
1	1	1
1	1	1
1	1	1
1	1	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	1
0	0	1
0	1	0
0	1	1

$$P(y = 1) = 0.5$$
$$P(y = 0) = 0.5$$
$$P(\text{gold} = 1|Y = 1) = 0.5$$
$$P(\text{gold} = 0|Y = 1) = 0.5$$
$$P(\text{gold} = 1|Y = 0) = 0.17$$
$$P(\text{gold} = 0|Y = 0) = 0.83$$
$$P(\text{price} = 1|Y = 1) = 0.66$$
$$P(\text{price} = 0|Y = 1) =$$

Naive Bayes

Example

gold	price	spam?
1	1	1
1	1	1
1	1	1
1	1	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	1
0	0	1
0	1	0
0	1	1

$P(y = 1) = 0.5$

$P(y = 0) = 0.5$

$P(\text{gold} = 1|Y = 1) = 0.5$

$P(\text{gold} = 0|Y = 1) = 0.5$

$P(\text{gold} = 1|Y = 0) = 0.17$

$P(\text{gold} = 0|Y = 0) = 0.83$

$P(\text{price} = 1|Y = 1) = 0.66$

$P(\text{price} = 0|Y = 1) = 0.33$

$P(\text{price} = 1|Y = 0) =$

Naive Bayes

Example

gold	price	spam?
1	1	1
1	1	1
1	1	1
1	1	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	1
0	0	1
0	1	0
0	1	1

$$
\begin{aligned}
&P(y=1) = 0.5 \\
&P(y=0) = 0.5 \\
&P(\text{gold}=1|Y=1) = 0.5 \\
&P(\text{gold}=0|Y=1) = 0.5 \\
&P(\text{gold}=1|Y=0) = 0.17 \\
&P(\text{gold}=0|Y=0) = 0.83 \\
&P(\text{price}=1|Y=1) = 0.66 \\
&P(\text{price}=0|Y=1) = 0.33 \\
&P(\text{price}=1|Y=0) = 0.33 \\
&P(\text{price}=0|Y=0) =
\end{aligned}
$$

Example

gold	price	spam?
1	1	1
1	1	1
1	1	1
1	1	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	1
0	0	1
0	1	0
0	1	1

$$P(y = 1) = 0.5$$
$$P(y = 0) = 0.5$$
$$P(\text{gold} = 1|Y = 1) = 0.5$$
$$P(\text{gold} = 0|Y = 1) = 0.5$$
$$P(\text{gold} = 1|Y = 0) = 0.17$$
$$P(\text{gold} = 0|Y = 0) = 0.83$$
$$P(\text{price} = 1|Y = 1) = 0.66$$
$$P(\text{price} = 0|Y = 1) = 0.33$$
$$P(\text{price} = 1|Y = 0) = 0.33$$
$$P(\text{price} = 0|Y = 0) = 0.66$$

We can, for example, compute:

$$P(y = 1|\text{gold} = 1 \wedge \text{price} = 0) =$$

Example

gold	price	spam?
1	1	1
1	1	1
1	1	1
1	1	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	1
0	0	1
0	1	0
0	1	1

$$P(y=1) = 0.5$$
$$P(y=0) = 0.5$$
$$P(\text{gold}=1|Y=1) = 0.5$$
$$P(\text{gold}=0|Y=1) = 0.5$$
$$P(\text{gold}=1|Y=0) = 0.17$$
$$P(\text{gold}=0|Y=0) = 0.83$$
$$P(\text{price}=1|Y=1) = 0.66$$
$$P(\text{price}=0|Y=1) = 0.33$$
$$P(\text{price}=1|Y=0) = 0.33$$
$$P(\text{price}=0|Y=0) = 0.66$$

We can, for example, compute:

$$P(y=1|\text{gold}=1 \wedge \text{price}=0) = \frac{0.5 \times 0.33 \times 0.5}{0.1386} = \frac{0.825}{0.1386} = 0.595$$
$$P(y=0|\text{gold}=1 \wedge \text{price}=0) =$$

Naive Bayes

Example

gold	price	spam?
1	1	1
1	1	1
1	1	1
1	1	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	1
0	0	1
0	1	0
0	1	1

$$P(y=1) = 0.5$$
$$P(y=0) = 0.5$$
$$P(\text{gold}=1|Y=1) = 0.5$$
$$P(\text{gold}=0|Y=1) = 0.5$$
$$P(\text{gold}=1|Y=0) = 0.17$$
$$P(\text{gold}=0|Y=0) = 0.83$$
$$P(\text{price}=1|Y=1) = 0.66$$
$$P(\text{price}=0|Y=1) = 0.33$$
$$P(\text{price}=1|Y=0) = 0.33$$
$$P(\text{price}=0|Y=0) = 0.66$$

We can, for example, compute:

$$P(y=1|\text{gold}=1 \wedge \text{price}=0) = \frac{0.5 \times 0.33 \times 0.5}{0.1386} = \frac{0.825}{0.1386} = 0.595$$
$$P(y=0|\text{gold}=1 \wedge \text{price}=0) = 1 - 0.595 = 0.405$$

Naive Bayes

Naive Bayes

- If the independence assumption is not valid, then the model can provide very bad predictions.

Naive Bayes

- If the independence assumption is not valid, then the model can provide very bad predictions.
- In many applications, however, this assumption seems to be at least partially satisfied, for example, in text classification.

Naive Bayes

- If the independence assumption is not valid, then the model can provide very bad predictions.
- In many applications, however, this assumption seems to be at least partially satisfied, for example, in text classification.
- Training is very efficient: one pass over training data to collect all necessary statistics.

Naive Bayes

- If the independence assumption is not valid, then the model can provide very bad predictions.
- In many applications, however, this assumption seems to be at least partially satisfied, for example, in text classification.
- Training is very efficient: one pass over training data to collect all necessary statistics.
- Prediction is linear in number of features.

Naive Bayes

- If the independence assumption is not valid, then the model can provide very bad predictions.
- In many applications, however, this assumption seems to be at least partially satisfied, for example, in text classification.
- Training is very efficient: one pass over training data to collect all necessary statistics.
- Prediction is linear in number of features.
- Some tricks to improve quality of computed statistics: Laplace correction and similar.

Naive Bayes

- If the independence assumption is not valid, then the model can provide very bad predictions.
- In many applications, however, this assumption seems to be at least partially satisfied, for example, in text classification.
- Training is very efficient: one pass over training data to collect all necessary statistics.
- Prediction is linear in number of features.
- Some tricks to improve quality of computed statistics: Laplace correction and similar.

Question

Is Naive Bayes a linear classifier? **Prove** under which conditions it is true.

Linear models

Linear models

- Consider a linear model of the form:

$$f(\boldsymbol{x}) = w_0 + \sum_{j=1}^{m} w_j x_j \, ,$$

where $\boldsymbol{w} = (w_0, w_1, \ldots, w_m)$ are the parameters of the model and $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ is a feature vector describing an example.

Linear models

- Consider a linear model of the form:

$$f(\boldsymbol{x}) = w_0 + \sum_{j=1}^{m} w_j x_j \,,$$

  where $\boldsymbol{w} = (w_0, w_1, \ldots, w_m)$ are the parameters of the model and $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ is a feature vector describing an example.
- It is often convenient to use vector notation:

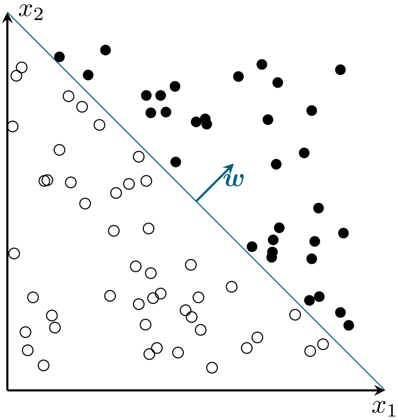
$$f(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x} \,,$$

  where $\boldsymbol{x} = (1, x_1, x_2, \ldots, x_n)$ has an additional 1 at the first position.

Linear models

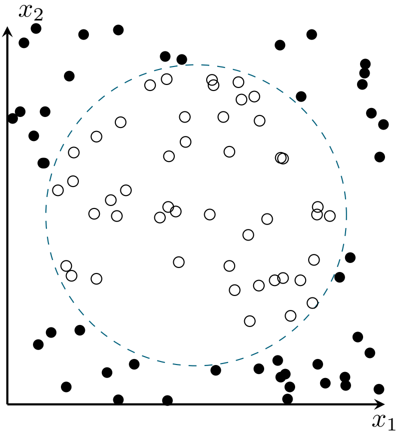
- Linear models constitute a very general class of models:
  - Basic transformations and expansion of original features,
  - Kernel trick (SVM),
  - Linear combination of weak classifiers (AdaBoost),
  - The fundamental component of the neural networks.

Linear models – transformations of features



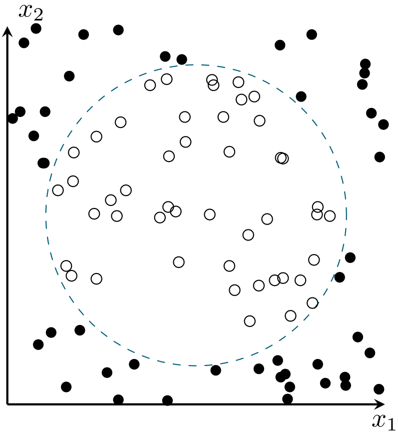
- Ideal case of perfect linear separation.

Linear models – transformations of features



- What if the data is not even close to linear?
  Give up on linear classifier...?

Linear models – transformations of features

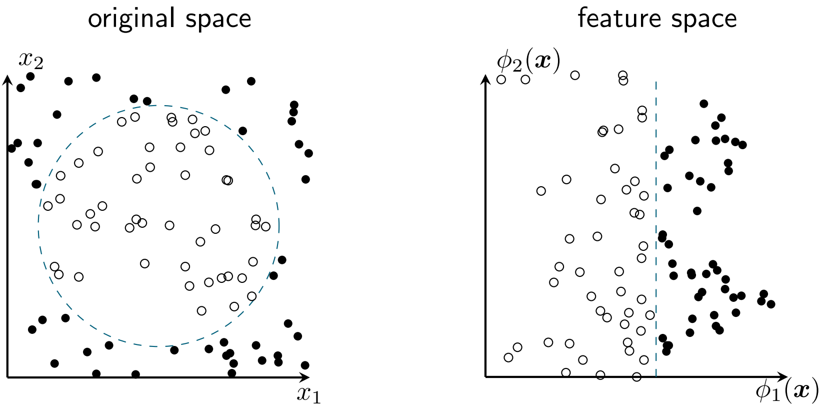


- What if the data is not even close to linear?
  Give up on linear classifier...?
- Or better, simply **invent new features**.

Linear models – transformations of features

Embed instances into a feature space:

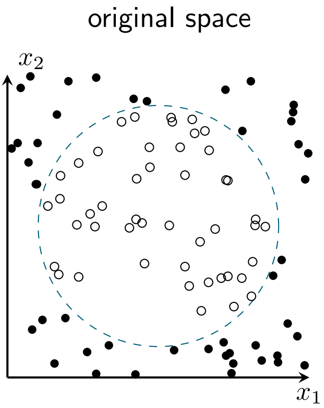
$$\boldsymbol{\phi}\colon \mathbb{R}^m \to \mathbb{R}^N$$

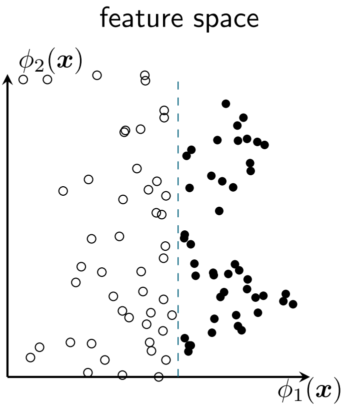


Linear models – transformations of features

Embed instances into a feature space:

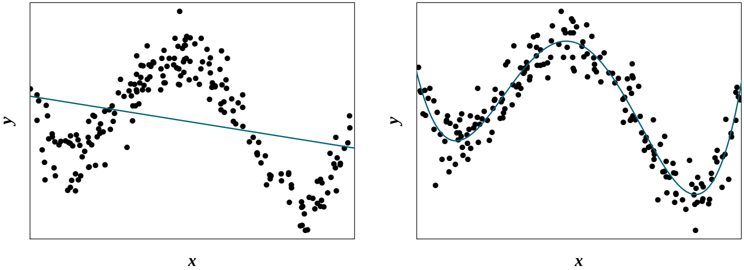
$$\boldsymbol{\phi}\colon \mathbb{R}^m \to \mathbb{R}^N$$



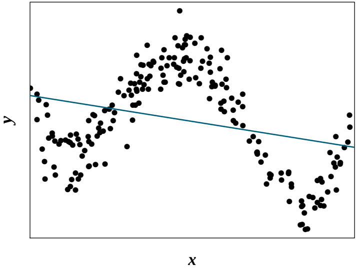


- $\boldsymbol{\phi}(x_1, x_2) = \left(\sqrt{x_1^2 + x_2^2}\ ,\ \arctan \frac{x_2}{x_1}\right)$.

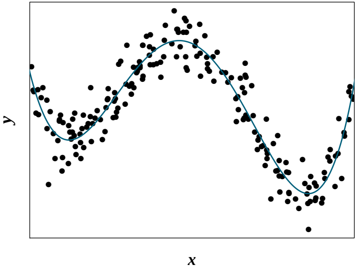
Linear models – feature expansions



Linear models – feature expansions

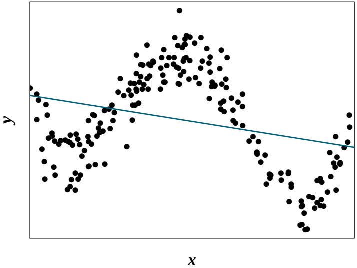


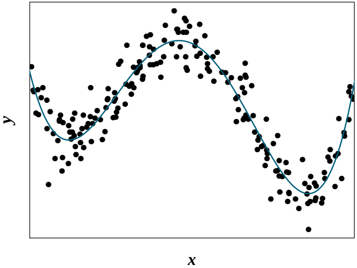
prediction: $f(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x}$



prediction: $f(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x}$

Linear models – feature expansions





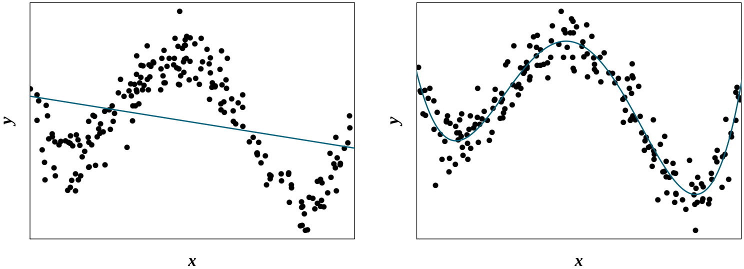
prediction: $f(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x}$

features: $\boldsymbol{x} = (1, x)$

prediction: $f(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x}$

features: $\boldsymbol{x} = (1, x, x^2, x^3, x^4)$

Linear models – feature expansions



prediction: $f(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x}$

features: $\boldsymbol{x} = (1, x)$

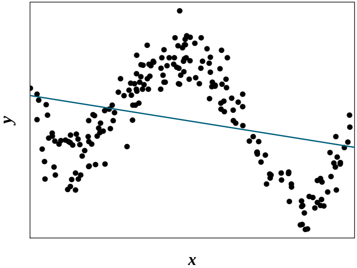
$f(x) = w_0 + w_1 x$

prediction: $f(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x}$

features: $\boldsymbol{x} = (1, x, x^2, x^3, x^4)$

$f(x) = w_0 + \sum_{i=1}^{4} w_i x_i$

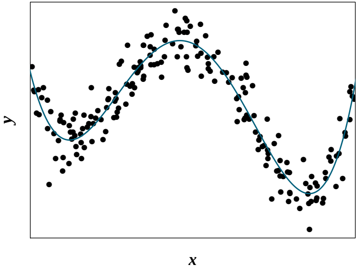
Linear models – feature expansions



prediction: $f(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x}$

features: $\boldsymbol{x} = (1, x)$

$f(x) = w_0 + w_1 x$



prediction: $f(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x}$

features: $\boldsymbol{x} = (1, x, x^2, x^3, x^4)$

$f(x) = w_0 + \sum_{i=1}^{4} w_i x_i$

- $\boldsymbol{\phi}(x) = \big(1, x, x^2, x^3, x^4\big)$.
- Both models are **linear** in features space!

Linear models – the kernel trick[2]

- If $\boldsymbol{w}$ is a linear combination of the training instances:

$$\boldsymbol{w} = \sum_{i=1}^{n} c_i \boldsymbol{x}_i,$$

then:

$$\boldsymbol{w} \cdot \boldsymbol{x} = \underbrace{\left(\sum_{i=1}^{n} c_i \, \boldsymbol{x}_i \right)}_{\boldsymbol{w}} \cdot \, \boldsymbol{x} = \sum_{i=1}^{n} c_i \, (\boldsymbol{x}_i \cdot \boldsymbol{x}).$$

[2] Bernhard E. Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *Conference on Learning Theory (COLT)*, pages 144–152, 1992

Linear models – the kernel trick[2]

- If $\boldsymbol{w}$ is a linear combination of the training instances:

$$\boldsymbol{w} = \sum_{i=1}^{n} c_i \boldsymbol{x}_i,$$

then:

$$\boldsymbol{w} \cdot \boldsymbol{x} = \underbrace{\left(\sum_{i=1}^{n} c_i \, \boldsymbol{x}_i \right)}_{\boldsymbol{w}} \cdot \; \boldsymbol{x} = \sum_{i=1}^{n} c_i \, (\boldsymbol{x}_i \cdot \boldsymbol{x}).$$

- After embedding $\boldsymbol{x} \mapsto \boldsymbol{\phi}(\boldsymbol{x})$:

$$\boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}) = \underbrace{\left(\sum_{i=1}^{n} c_i \, \boldsymbol{\phi}(\boldsymbol{x}_i) \right)}_{\boldsymbol{w}} \cdot \; \boldsymbol{\phi}(\boldsymbol{x}) = \sum_{i=1}^{n} c_i \left(\underbrace{\boldsymbol{\phi}(\boldsymbol{x}_i) \cdot \boldsymbol{\phi}(\boldsymbol{x})}_{K(\boldsymbol{x}_i, \boldsymbol{x})} \right)$$

[2] Bernhard E. Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *Conference on Learning Theory (COLT)*, pages 144–152, 1992

Linear models – the kernel trick[2]

- If $\boldsymbol{w}$ is a linear combination of the training instances:

$$\boldsymbol{w} = \sum_{i=1}^{n} c_i \boldsymbol{x}_i,$$

dot product

then:

$$\boldsymbol{w} \cdot \boldsymbol{x} = \underbrace{\left(\sum_{i=1}^{n} c_i \, \boldsymbol{x}_i \right)}_{\boldsymbol{w}} \cdot \, \boldsymbol{x} = \sum_{i=1}^{n} c_i \, (\boldsymbol{x}_i \cdot \boldsymbol{x}).$$

kernel function

- After embedding $\boldsymbol{x} \mapsto \boldsymbol{\phi}(\boldsymbol{x})$:

$$\boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}) = \underbrace{\left(\sum_{i=1}^{n} c_i \, \boldsymbol{\phi}(\boldsymbol{x}_i) \right)}_{\boldsymbol{w}} \cdot \, \boldsymbol{\phi}(\boldsymbol{x}) = \sum_{i=1}^{n} c_i \left(\underbrace{\boldsymbol{\phi}(\boldsymbol{x}_i) \cdot \boldsymbol{\phi}(\boldsymbol{x})}_{K(\boldsymbol{x}_i, \boldsymbol{x})} \right)$$

[2] Bernhard E. Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *Conference on Learning Theory (COLT)*, pages 144–152, 1992

Linear models – the kernel trick[3]

primal form

$$f(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x})$$

N parameters
(feature space dim.)

kernelized form

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} c_i K(\boldsymbol{x}_i, \boldsymbol{x})$$

n parameters
(num. of instances)

[3] Bernhard E. Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *Conference on Learning Theory (COLT)*, pages 144–152, 1992

Fitting linear models

Fitting linear models

- We fit parameters $\boldsymbol{w}$ of a linear model using training data

$$\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)\}$$

where $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{im})$ is a feature vector of the i-th training example.

Fitting linear models

- We fit parameters $\boldsymbol{w}$ of a linear model using training data

$$\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)\}$$

  where $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{im})$ is a feature vector of the i-th training example.
- We use loss function $\ell(y, f(\boldsymbol{x}))$ to guide the learning process.

Linear regression

Linear regression

- Let $f(\boldsymbol{x})$ be a linear function of the input variables:

$$f(\boldsymbol{x}) = w_0 + \sum_{j=1}^{m} w_j x_j \;=\; \boldsymbol{w} \cdot \boldsymbol{x}.$$

Linear regression

- Let $f(\boldsymbol{x})$ be a linear function of the input variables:

$$f(\boldsymbol{x}) = w_0 + \sum_{j=1}^{m} w_j x_j \; = \; \boldsymbol{w} \cdot \boldsymbol{x}.$$

- We minimize the squared error loss:

$$\ell_{\mathrm{sq}}(y, f(\boldsymbol{x})) = (y - f(\boldsymbol{x}))^2 \,.$$

Linear regression

- Let $f(\boldsymbol{x})$ be a linear function of the input variables:

$$f(\boldsymbol{x}) = w_0 + \sum_{j=1}^{m} w_j x_j \; = \; \boldsymbol{w} \cdot \boldsymbol{x}.$$

- We minimize the squared error loss:

$$\ell_{\mathrm{sq}}(y, f(\boldsymbol{x})) = (y - f(\boldsymbol{x}))^2 \, .$$

- Minimizing squared error loss is equivalent to estimating:

$$\mathbb{E}(y|\boldsymbol{x}) = w_0 + \sum_{j=1}^{n} w_j x_j \; = \; \boldsymbol{w} \cdot \boldsymbol{x},$$

the **conditional mean value**.

- The task of a learning algorithm is to estimate

$$\boldsymbol{w} = (w_0, w_1, \ldots, w_m)$$

by solving the following optimization problem:

$$\begin{aligned}
\widehat{\boldsymbol{w}} &= \arg\min_{\boldsymbol{w}} \sum_{i=1}^{n} \ell_{\text{sq}}(y_i, w_0 + \sum_{j=1}^{m} w_j x_{ij}) \\
&= \arg\min_{\boldsymbol{w}} \sum_{i=1}^{n} (y_i - w_0 - \sum_{j=1}^{m} w_j x_{ij})^2 \\
&= \arg\min_{\boldsymbol{w}} \sum_{i=1}^{n} (y_i - \boldsymbol{w} \cdot \boldsymbol{x}_i)^2.
\end{aligned}$$

Linear regression

- The task of a learning algorithm is to estimate

$$\boldsymbol{w} = (w_0, w_1, \dots, w_m)$$

by solving the following optimization problem:

$$\begin{aligned}
\widehat{\boldsymbol{w}} &= \arg\min_{\boldsymbol{w}} \sum_{i=1}^{n} \ell_{\mathrm{sq}}(y_i, w_0 + \sum_{j=1}^{m} w_j x_{ij}) \\
&= \arg\min_{\boldsymbol{w}} \sum_{i=1}^{n} (y_i - w_0 - \sum_{j=1}^{m} w_j x_{ij})^2 \\
&= \arg\min_{\boldsymbol{w}} \sum_{i=1}^{n} (y_i - \boldsymbol{w} \cdot \boldsymbol{x}_i)^2.
\end{aligned}$$

- Let us solve this problem in a simple one-dimension case $(m = 1)$ ...

Linear regression

- Define:

$$\widehat{L}(w_0, w_1) = \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i)^2.$$

Linear regression

- Define:
$$\widehat{L}(w_0, w_1) = \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i)^2.$$

- We take derivative of $\widehat{L}$ with respect to w_0 and equate it to zero:

$$\frac{\partial \widehat{L}}{\partial w_0} = 0 \iff -2 \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i) \quad = \quad 0$$

Linear regression

- Define:

$$\widehat{L}(w_0, w_1) = \sum_{i=1}^{n}(y_i - w_0 - w_1 x_i)^2.$$

- We take derivative of $\widehat{L}$ with respect to w_0 and equate it to zero:

$$\frac{\partial \widehat{L}}{\partial w_0} = 0 \iff -2\sum_{i=1}^{n}(y_i - w_0 - w_1 x_i) = 0$$

$$n w_0 = \sum_{i=1}^{n} y_i - w_1 \sum_{i=1}^{n} x_i$$

- Define:

$$\widehat{L}(w_0, w_1) = \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i)^2.$$

- We take derivative of $\widehat{L}$ with respect to w_0 and equate it to zero:

$$\frac{\partial \widehat{L}}{\partial w_0} = 0 \iff -2 \sum_{i=1}^{n} (y_i - w_0 - w_1 x_i) = 0$$

$$n w_0 = \sum_{i=1}^{n} y_i - w_1 \sum_{i=1}^{n} x_i$$

$$w_0 = \bar{y} - w_1 \bar{x}\,,$$

where:

$$\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i \quad \bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

Linear regression

- In the next step we take derivative of $\widehat{L}$ with respect to w_1 and equate it to zero:

$$\frac{\partial \widehat{L}}{\partial w_1} = 0 \iff -2\sum_{i=1}^{n}(y_i - w_0 - w_1 x_i)x_i \quad = \quad 0$$

- In the next step we take derivative of $\widehat{L}$ with respect to w_1 and equate it to zero:

$$\begin{aligned}\frac{\partial \widehat{L}}{\partial w_1} = 0 \iff -2\sum_{i=1}^{n}(y_i - w_0 - w_1 x_i)x_i &= 0 \\ \sum_{i=1}^{n} y_i x_i - w_0 \sum_{i=1}^{n} x_i - w_1 \sum_{i=1}^{n} x_i^2 &= 0\end{aligned}$$

- In the next step we take derivative of $\widehat{L}$ with respect to w_1 and equate it to zero:

$$
\begin{aligned}
\frac{\partial \widehat{L}}{\partial w_1} = 0 \iff -2\sum_{i=1}^{n}(y_i - w_0 - w_1 x_i)x_i &= 0 \\
\sum_{i=1}^{n} y_i x_i - w_0 \sum_{i=1}^{n} x_i - w_1 \sum_{i=1}^{n} x_i^2 &= 0 \\
(w_0 = \bar{y} - w_1\bar{x}) \qquad \sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x}) - w_1 \sum_{i=1}^{n}(x_i - \bar{x})^2 &= 0
\end{aligned}
$$

- In the next step we take derivative of $\widehat{L}$ with respect to w_1 and equate it to zero:

$$
\begin{aligned}
\frac{\partial \widehat{L}}{\partial w_1} = 0 \iff -2\sum_{i=1}^{n}(y_i - w_0 - w_1 x_i)x_i &= 0 \\
\sum_{i=1}^{n} y_i x_i - w_0 \sum_{i=1}^{n} x_i - w_1 \sum_{i=1}^{n} x_i^2 &= 0 \\
(w_0 = \bar{y} - w_1\bar{x}) \qquad \sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x}) - w_1 \sum_{i=1}^{n}(x_i - \bar{x})^2 &= 0
\end{aligned}
$$

so that we get:

$$
w_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}
$$

- The solution for one-dimensional problem is:

$$\begin{aligned}\widehat{w}_1 &= \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}, \\ \widehat{w}_0 &= \bar{y}_1 - \widehat{w}_1 \bar{x}.\end{aligned}$$

- The final model is given by:

$$f(\boldsymbol{x}) = \widehat{w}_0 + \widehat{w}_1 x$$

Linear regression – general case

- The criterion to be minimized:

$$\widehat{L}(\boldsymbol{w}) = \sum_{i=1}^{n} (y_i - \boldsymbol{w} \cdot \boldsymbol{x}_i)^2.$$

Linear regression – general case

- The criterion to be minimized:

$$\widehat{L}(\boldsymbol{w}) = \sum_{i=1}^{n} (y_i - \boldsymbol{w} \cdot \boldsymbol{x}_i)^2.$$

- Differentiating with respect to $\boldsymbol{w}$ and setting the gradient to 0:

$$\frac{\partial \widehat{L}}{\partial \boldsymbol{w}} = \mathbf{0} \iff 2 \sum_{i=1}^{n} (y_i - \boldsymbol{w} \cdot \boldsymbol{x}_i) \boldsymbol{x}_i \quad = \quad \mathbf{0}$$

Linear regression – general case

- The criterion to be minimized:

$$\widehat{L}(\boldsymbol{w}) = \sum_{i=1}^{n} (y_i - \boldsymbol{w} \cdot \boldsymbol{x}_i)^2.$$

- Differentiating with respect to $\boldsymbol{w}$ and setting the gradient to 0:

$$\begin{aligned} \frac{\partial \widehat{L}}{\partial \boldsymbol{w}} = \mathbf{0} \iff 2\sum_{i=1}^{n} (y_i - \boldsymbol{w} \cdot \boldsymbol{x}_i)\boldsymbol{x}_i &= \mathbf{0} \\ \sum_{i=1}^{n} y_i \boldsymbol{x}_i - \left(\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top \right) \boldsymbol{w} &= \mathbf{0} \end{aligned}$$

Linear regression – general case

- The criterion to be minimized:

$$\widehat{L}(\boldsymbol{w}) = \sum_{i=1}^{n} (y_i - \boldsymbol{w} \cdot \boldsymbol{x}_i)^2.$$

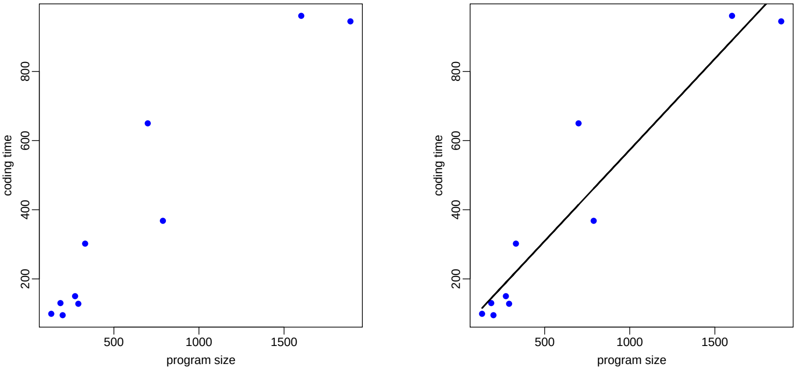
- Differentiating with respect to $\boldsymbol{w}$ and setting the gradient to 0:

$$\begin{aligned} \frac{\partial \widehat{L}}{\partial \boldsymbol{w}} = \boldsymbol{0} \iff 2\sum_{i=1}^{n} (y_i - \boldsymbol{w} \cdot \boldsymbol{x}_i)\boldsymbol{x}_i &= \boldsymbol{0} \\ \sum_{i=1}^{n} y_i \boldsymbol{x}_i - \left(\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top\right) \boldsymbol{w} &= \boldsymbol{0} \end{aligned}$$

- Assuming $\sum_i \boldsymbol{x}_i \boldsymbol{x}_i^\top$ is nonsingular, the solution is:

$$\widehat{\boldsymbol{w}} = \left(\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top\right)^{-1} \left(\sum_{i=1}^{n} y_i \boldsymbol{x}_i\right).$$

Linear regression – Example



Linear regression

Linear regression

- Very efficient method for a small or moderate number of features.

Linear regression

- Very efficient method for a small or moderate number of features.
- For large number of features different learning algorithms should be used.

Linear regression

- Very efficient method for a small or moderate number of features.
- For large number of features different learning algorithms should be used.
- Statistical properties of linear regression are very well-studied – very mature statistical procedure.

Linear regression

- Very efficient method for a small or moderate number of features.
- For large number of features different learning algorithms should be used.
- Statistical properties of linear regression are very well-studied – very mature statistical procedure.
- Can also be used for binary classification – quite popular in large scale problems.

Linear models for binary classification

- Direct optimization of $\ell_{0/1}(y, f(\boldsymbol{x}))$ is hard as this loss is neither convex nor differentiable.

- Direct optimization of $\ell_{0/1}(y, f(\boldsymbol{x}))$ is hard as this loss is neither convex nor differentiable.
- We can solve binary classification by using the so-called surrogate loss functions ℓ_{s}:

$$\widehat{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} \sum_{i=1}^{n} \ell_{\mathrm{s}}(y_i, \boldsymbol{w} \cdot \boldsymbol{x}_i)$$

- Direct optimization of $\ell_{0/1}(y, f(\boldsymbol{x}))$ is hard as this loss is neither convex nor differentiable.
- We can solve binary classification by using the so-called surrogate loss functions ℓ_{s}:

$$\widehat{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} \sum_{i=1}^{n} \ell_{\mathrm{s}}(y_i, \boldsymbol{w} \cdot \boldsymbol{x}_i)$$

- The surrogate losses should be characterized by desired statistical and computational properties such as convergence to the optimal solution of the 0/1 loss, smoothness and convexity.

Outline

Empirical risk minimization

- A wide spectrum of learning algorithms can be given in a general form of surrogate loss minimization:

$$\widehat{f} = \arg\min_{f \in \mathcal{F}} \sum_{i=1}^{n} \ell_s(y_i, f(\boldsymbol{x}))$$

Empirical risk minimization

- A wide spectrum of learning algorithms can be given in a general form of surrogate loss minimization:

$$\widehat{f} = \arg\min_{f \in \mathcal{F}} \sum_{i=1}^{n} \ell_s(y_i, f(\boldsymbol{x}))$$

- The differences between algorithms: form of the surrogate loss, model class, optimization procedure.

Empirical risk minimization

- A wide spectrum of learning algorithms can be given in a general form of surrogate loss minimization:

$$\widehat{f} = \arg\min_{f \in \mathcal{F}} \sum_{i=1}^{n} \ell_s(y_i, f(\boldsymbol{x}))$$

- The differences between algorithms: form of the surrogate loss, model class, optimization procedure.
- This general form allows to compare and analyze learning algorithms.

Problems to be discussed

- Surrogate losses and learning algorithms for linear models.

Problems to be discussed

- Surrogate losses and learning algorithms for linear models.
- Is learning possible?

Problems to be discussed

- Surrogate losses and learning algorithms for linear models.
- Is learning possible?
- Can learning converge to an optimal classifier?

Problems to be discussed

- Surrogate losses and learning algorithms for linear models.
- Is learning possible?
- Can learning converge to an optimal classifier?
- How to solve complex problems such as ranking or multi-label classification?

Summary

- Statistical decision theory for supervised learning.

Summary

- Statistical decision theory for supervised learning.
- Two phases: learning and prediction.

Summary

- Statistical decision theory for supervised learning.
- Two phases: learning and prediction.
- A wide spectrum of learning methods:

Summary

- Statistical decision theory for supervised learning.
- Two phases: learning and prediction.
- A wide spectrum of learning methods:
  - Histogram-based classifiers,

Summary

- Statistical decision theory for supervised learning.
- Two phases: learning and prediction.
- A wide spectrum of learning methods:
  - Histogram-based classifiers,
  - Decision trees,

Summary

- Statistical decision theory for supervised learning.
- Two phases: learning and prediction.
- A wide spectrum of learning methods:
  - Histogram-based classifiers,
  - Decision trees,
  - Nearest neighbors,

Summary

- Statistical decision theory for supervised learning.
- Two phases: learning and prediction.
- A wide spectrum of learning methods:
  - Histogram-based classifiers,
  - Decision trees,
  - Nearest neighbors,
  - Naive Bayes,

Summary

- Statistical decision theory for supervised learning.
- Two phases: learning and prediction.
- A wide spectrum of learning methods:
  - Histogram-based classifiers,
  - Decision trees,
  - Nearest neighbors,
  - Naive Bayes,
  - Linear models.