Introduction

Prof. Dr. Josif Grabocka, Dr. Daniele Cattaneo

Machine Learning Course
Winter Semester 2023/2024

Albert-Ludwigs-Universität Freiburg

grabocka@informatik.uni-freiburg.de, *cattaneo@informatik.uni-freiburg.de*

Overview

Table of Contents

Machine Learning

- Learn to **approximately** solve a task using
- ... past **occurrences** of task instances and *given* solutions
- ... by **maximizing** the quality of the *approximated* solutions.

Figure 1: Face Recognition, Courtesy of `www.nec.com`

Machine Learning is everywhere

- speech recognition, language translation
- search engines, product recommendation
- weather forecast
- biometric verification (face, fingerprint)
- spam email
- computer vision
- ... and thousands of other applications.

Machine Learning: A Task Solver

A task involves:

- a real agent (e.g., a doctor)
- which given a particular context (e.g., patient lab results)
- takes a complex decision (e.g., diagnose a disease)
- in order to achieve a goal (e.g., correct diagnosis).

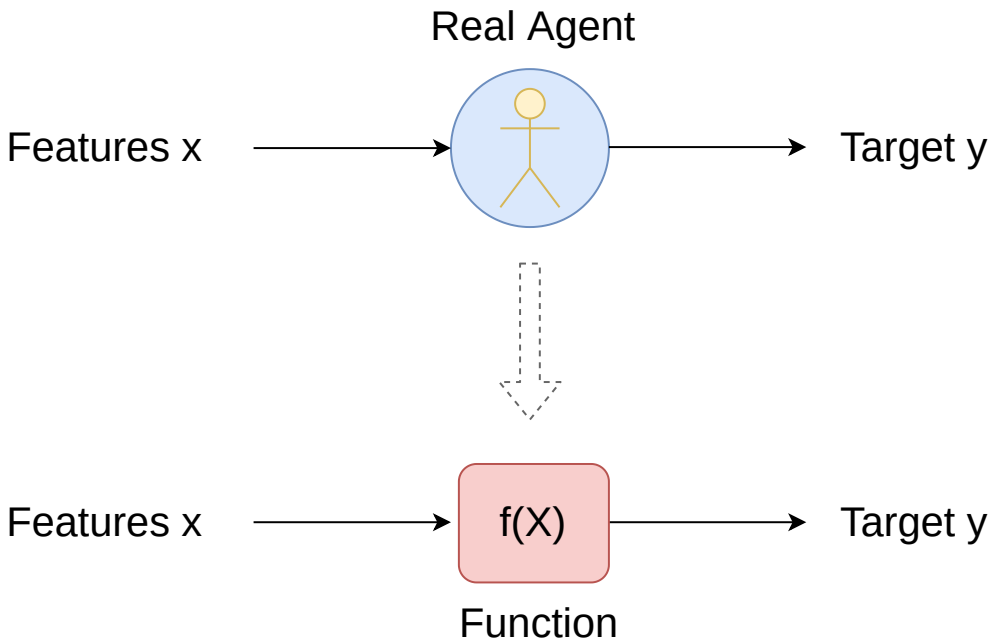
Machine Learning

focuses on approximately solving tasks by learning from

- the past decisions of a real agent and their outcomes
  - e.g., database of patient cases, their lab results and how doctors diagnosed them
- in order to mimic the agent's behavior
  - e.g., learn to diagnose these patient cases in the same way as the doctors did

Modeling Real Agents as Functions

- For an agent whose context are features x and its decision is the target y, ML replaces the agent with a math function f ...



- ... and trains f from observations $\{(x_1, y_1), \ldots, (x_N, y_N)\}$
  - such that $f(x_i) \approx y_i, \quad \forall i \in \{1, \ldots, N\}$

Supervised and Unsupervised Learning

- **Supervised** learning (previous example):
  - Target is given by an expert (ground-truth)
  - *Classification*, *Regression*, *Ranking*
  - The lion's share of this course
- **Unsupervised** learning:
  - Target contains no explicit labels of the context features
  - *Clustering*, *Dimensionality reduction*, *Anomaly/Outlier Detection*

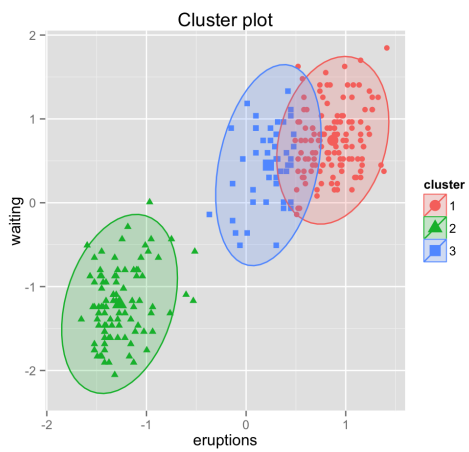


Figure 2: Clustering illustration, Courtesy of `www.sthda.com`

Table of Contents

Supervised Learning Example

- For N existing bank customers and $M = 23$ features, i.e. given $x \in \mathbb{R}^{N \times 23}$ and ground truth $y \in \{0, 1\}^N$

$y_{:}$	Default credit card payment (Yes = 1, No = 0)
$x_{:,1}$	Amount of the given credit (NT dollar)
$x_{:,2}$	Gender (1 = male; 2 = female).
$x_{:,3}$	Education (1=graduate; 2=univ.; 3 = high school; 4 = others).
$x_{:,4}$	Marital status (1 = married; 2 = single; 3 = others).
$x_{:,5}$	Age (year)
$x_{:,6} - x_{:,11}$	Past Delays (-1=duly, . . . , 9=delay of nine months)
$x_{:,12} - x_{:,17}$	Amount of bill statements
$x_{:,18} - x_{:,23}$	Amount of previous payments

Table 1: Yeh, I. C., & Lien, C. H. (2009).

- Goal: Estimate the default of a new $(N+1)$-th customer, i.e. given $x_{N+1,:} \in \mathbb{R}^{23}$, estimate $y_{N+1} = ?$

Estimating the Target Variable

- Given a training data of N recorded instances, composed of
  - features variables $x \in \mathbb{R}^{N \times M}$ and
  - target variable $y \in \mathbb{R}^{N}$.
- Predict the target variable of a future instance $x^{test} \in \mathbb{R}^{M}$?

- Need to have a function $f(x)$ that predicts the target $\hat{y} := f(x)$
  - Known as "**Prediction Model**"

- How to find a good function? Answer:
  - Parametrize through learn-able parameters θ as $f(x, \theta)$
  - Learn parameters θ using the training data

  - *But, according to which criteria should we learn θ?*

Difference to Ground Truth

- The quality of a prediction model $f(x, \theta)$
  - Difference between the estimated target $\hat{y}$ and ground-truth target y
  - Defined as a loss function $\mathcal{L}(y, \hat{y}) : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$
  - The term loss is used for minimization tasks, e.g. regression
- Note: sometimes a maximization of $\mathcal{L}(y, \hat{y})$ is needed

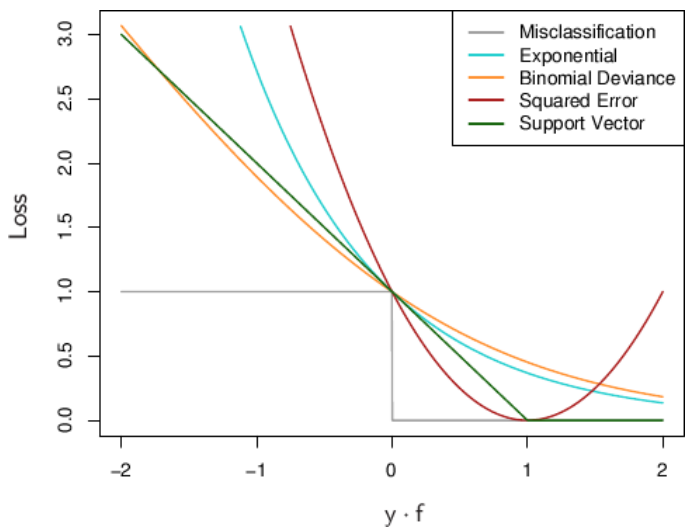


Figure 3: Loss types, (Hastie et al., 2009, The Elements of Statistical Learning)

Archetype of Supervised Learning

- *Data dimensions*: N instances having M features
- *Features*: $x \in \mathbb{R}^{N \times M}$ and *Target*: $y \in \mathbb{R}^N$
- A *prediction model*: having parameters $\theta \in \mathbb{R}^K$ is $f : \mathbb{R}^M \times \mathbb{R}^K \to \mathbb{R}$

$$\hat{y}_n := f(x_n, \theta)$$

- *Loss function*: $\mathcal{L}(y_n, \hat{y}_n) : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$
- *Regularization*: $\Omega(\theta) : \mathbb{R}^K \to \mathbb{R}$
- *Objective function*:

$$\underset{\theta}{\operatorname{argmin}} \sum_{n=1}^{N} \mathcal{L}(y_n, \hat{y}_n) + \Omega(\theta)$$

Prediction Models - I

- Linear Model
  - $\hat{y}_n = \theta_0 + \theta_1 x_{n,1} + \theta_2 x_{n,2} + \cdots + \theta_M x_{n,M} = \theta_0 + \sum_{m=1}^{M} \theta_m x_{n,m}$

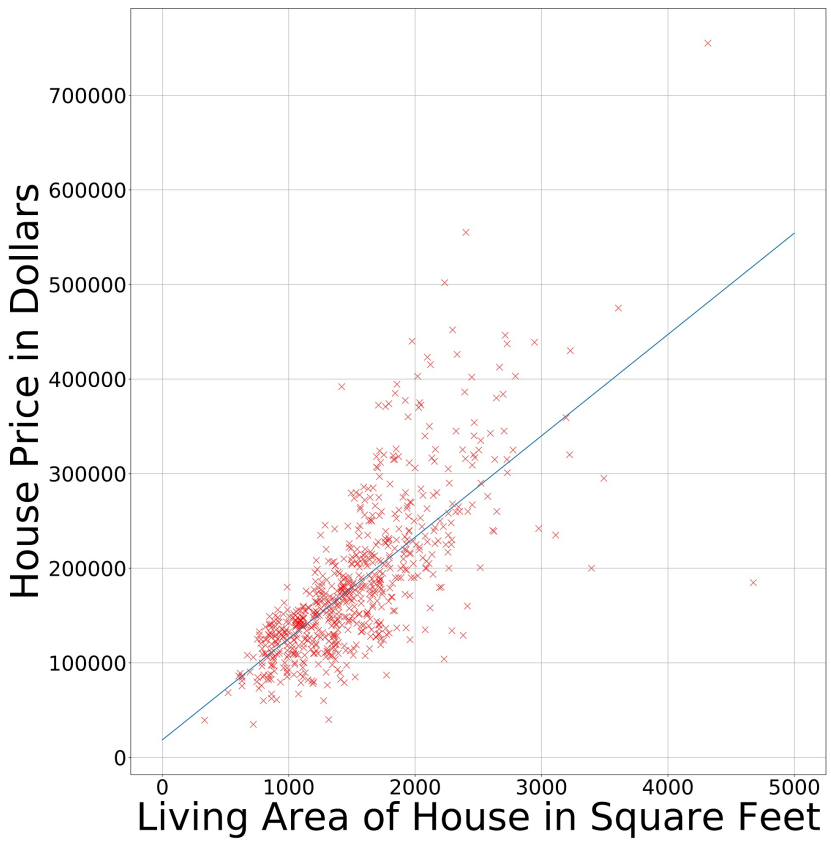


Figure 4: Predicting house prices

Prediction Models - II

- Polynomial Regression
  - $\hat{y}_n = \theta_0 + \sum_{m=1}^{M} \theta_m x_{n,m} + \sum_{m=1}^{M} \sum_{m'=1}^{M} \theta_{m,m'} x_{n,m} x_{n,m'} + \dots$

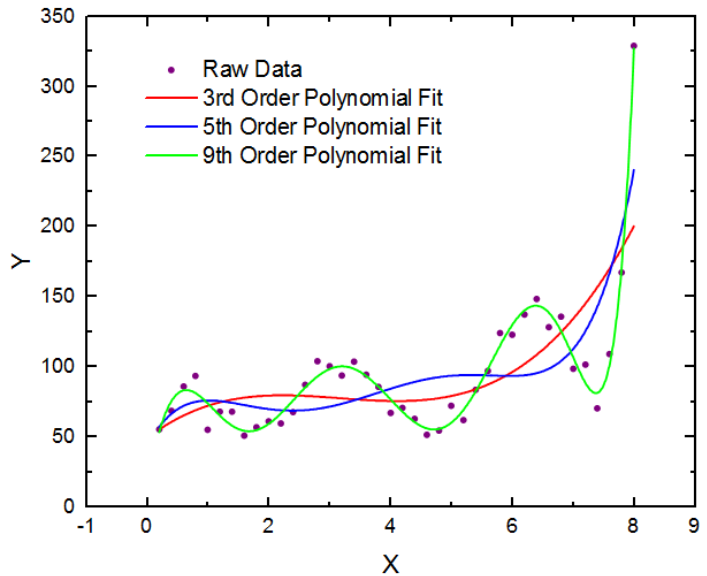


Figure 5: Polynomial regression, Source: `www.originlab.com`

- Decision Trees
- Neural Networks

Decision Tree as a Prediction Model

A prediction model $\hat{y}_n := f(x_n, \theta)$ can be also a tree:

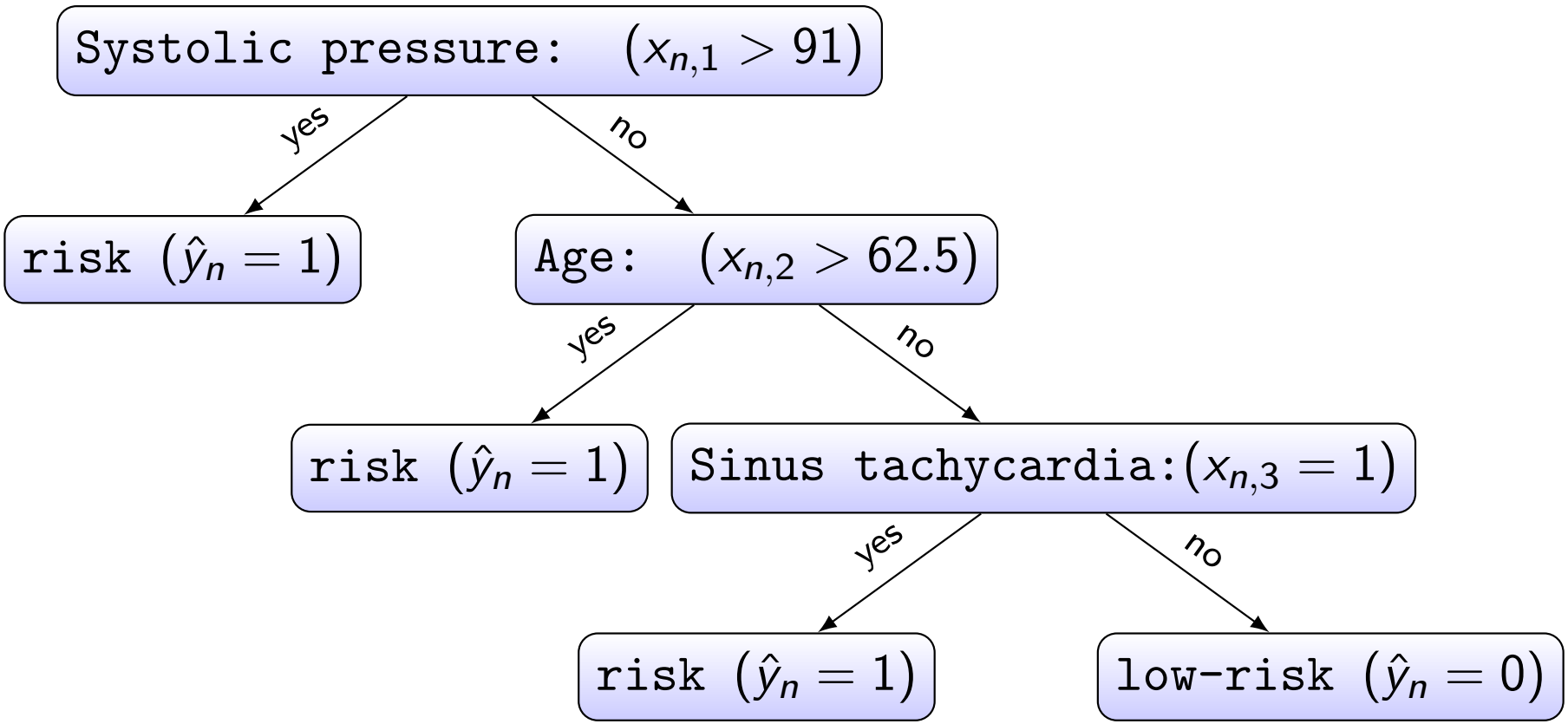
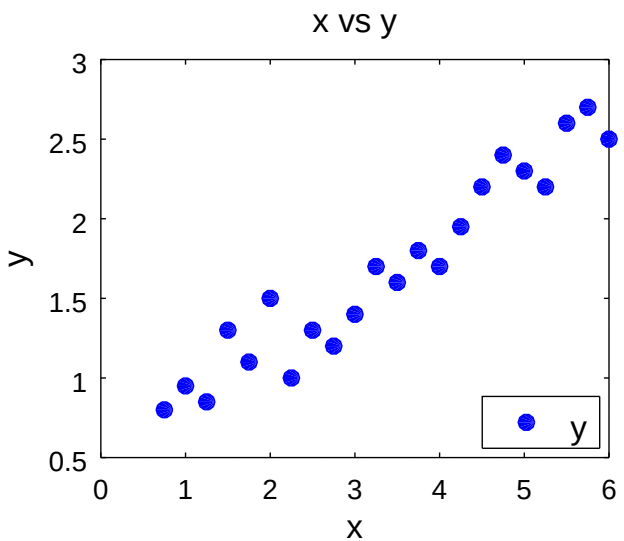
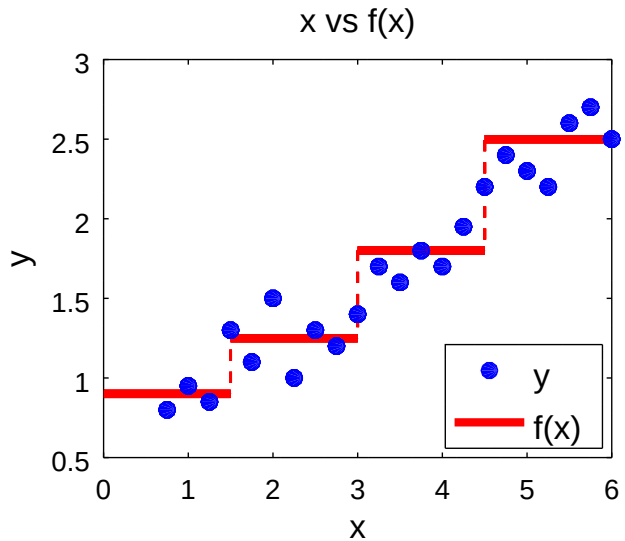
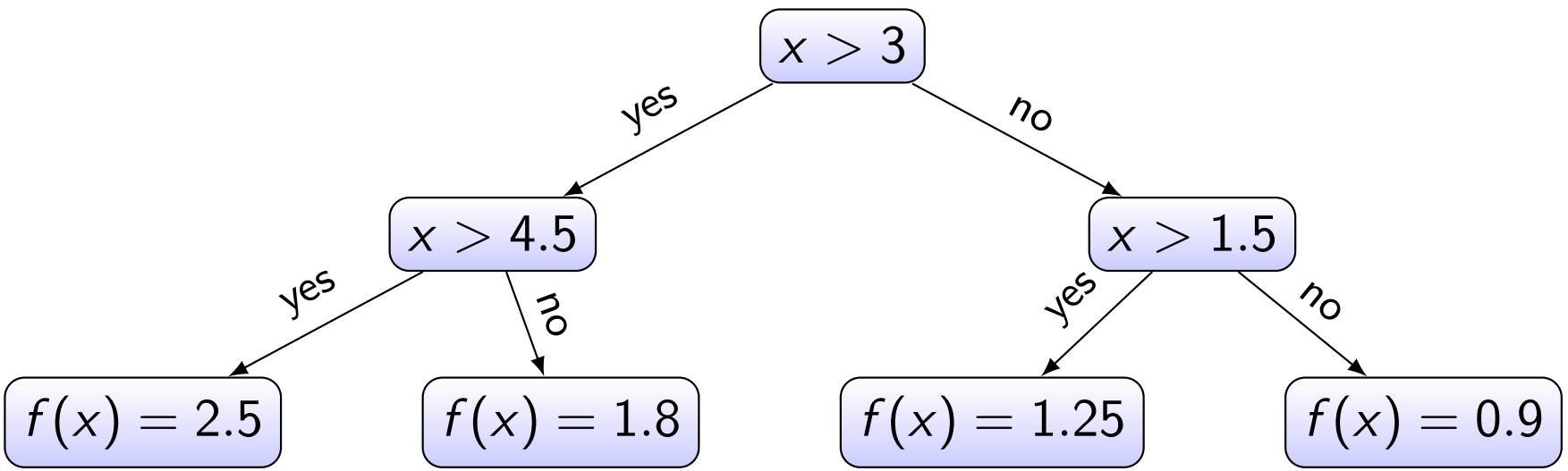


Figure 6: San Diego Medical Center

Decision Tree as a Step-wise Function







Neural Network Model

- Composite functions, i.e., functions of functions ...
- A neuron indexed i is a non-linear function $f_i(x, \theta_i)$
- If neuron i is connected to neuron j the model is $f_j(f_i(x, \theta_i), \theta_j)$

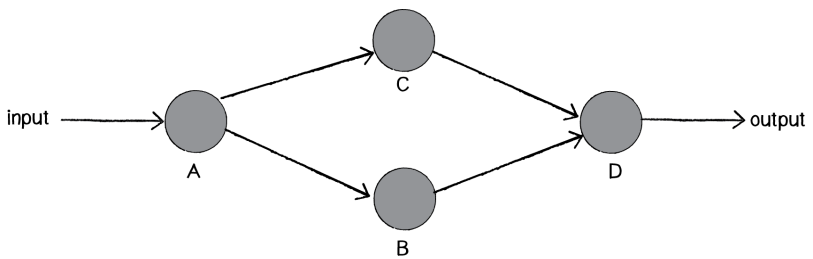


Figure 7: One layer network, Courtesy of Shiffman 2010, The Nature of Code

$$\hat{y}_n := f_D(\theta_0 + \theta_{D1} f_C(f_A(x_n, \theta_A), \theta_C) + \theta_{D2} f_B(f_A(x_n, \theta_A), \theta_B))$$

Loss Functions

- Regression (target is real-valued $y_n \in \mathbb{R}$)
  - Least-squares:

$$\mathcal{L}(y_n, \hat{y}_n) := (y_n - \hat{y}_n)^2$$

  - L1:

$$\mathcal{L}(y_n, \hat{y}_n) := |y_n - \hat{y}_n|$$

- Binary Classification
  - Logistic loss, $y_n \in \{0, 1\}$:

$$\mathcal{L}(y_n, \hat{y}_n) := -y_n \log(\hat{y}_n) - (1 - y_n) \log(1 - \hat{y}_n)$$

  - Hinge loss, $y_n \in \{-1, 1\}$:

$$\mathcal{L}(y_n, \hat{y}_n) := max(0, 1 - y_n \hat{y}_n)$$

Multi-class loss - Softmax

- Re-express targets $y_n \in \{1, \ldots, C\}$ as one-vs-all, i.e.

$$y_{n,c} := \begin{cases} 1 & y_n = C \\ 0 & y_n \neq C \end{cases}$$

- Learn model parameters per class $\theta \in \mathbb{R}^{C \times K}$
- Estimations expressed as probabilities among classes

$$\hat{y}_{n,c} = \frac{e^{f(x_n, \theta_c)}}{\sum_{q=1}^{C} e^{f(x_n, \theta_q)}}$$

- Logloss:

$$\mathcal{L}(y_{n,:}, \hat{y}_{n,:}) := -\sum_{c=1}^{C} y_{n,c} \log(\hat{y}_{n,c})$$

Overfitting, Underfitting

- Underfitting (High model bias): Unable to capture complexity
- Overfitting (High model variance): Capturing noise

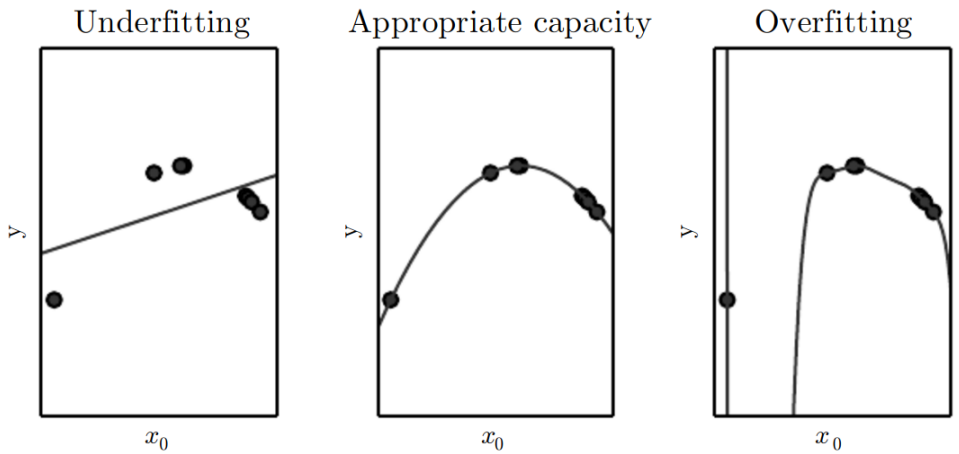


Figure 8: Overfitting, Underfitting, Source: Goodfellow et al., 2016, Deep Learning

Regularization

- Fights overfitting to the noise of the measured data

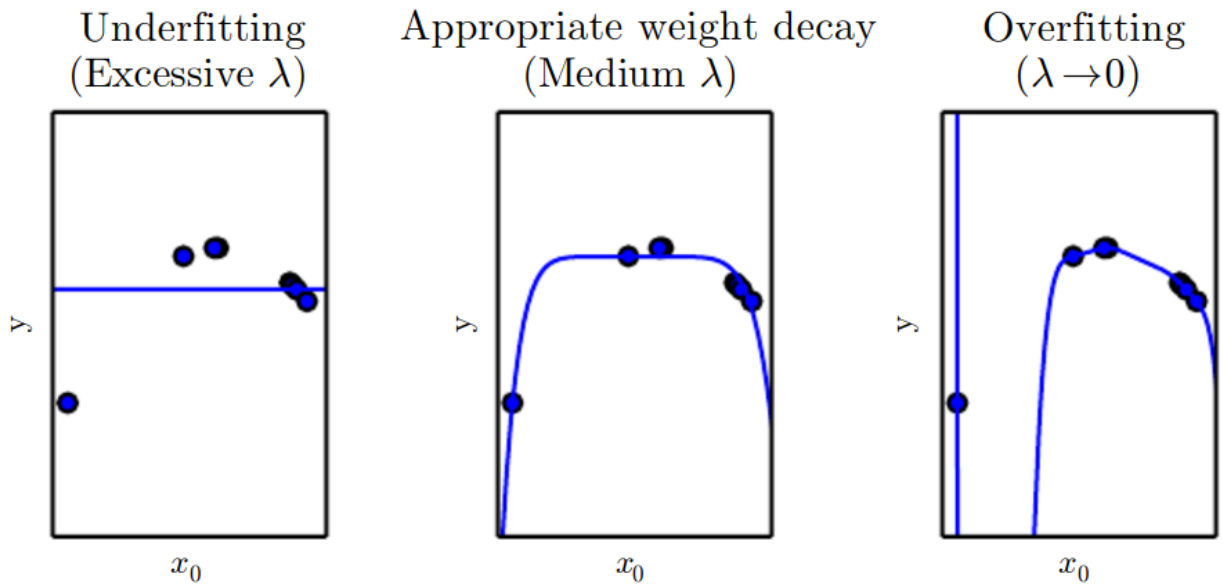


Figure 9: Regularizing a polynomial regression, Source: Goodfellow et al., 2016, Deep Learning

Table of Contents

Probabilistic Interpretation: Generative Model

- Considering a linear model

$$\hat{y} = \theta_0 + \sum_{m=1}^{M} \theta_m x_m$$

- Assume the error in predicting the ground truth y_n is normally distributed

$$\epsilon | x \sim \mathcal{N}(0, \sigma^2)$$

- In other words, the models generates estimations

$$\hat{y} \sim \mathcal{N}\left(\theta_0 + \sum_{m=1}^{M} \theta_m x_m, \sigma^2\right)$$

Maximum Likelihood Estimation

- Let $\hat{p}(y|x,\theta)$ be the probability density function for the target y given features x and parameters θ
- The likelihood of observing the target $y \in \mathbb{R}^N$ is

$$L(\theta) = \prod_{n=1}^{N} \hat{p}(y_n \,|\, x_n, \theta)$$

- What values of θ make our observed target more likely to occur?
- Aim: **Estimate** the θ-s which **maximize** the **likelihood**.

Maximum Likelihood Estimation - II

- Remember

$$\log(a\,b) = \log(a) + \log(b)$$
$$\underset{\theta}{\text{argmax}}\ g(\theta) = \underset{\theta}{\text{argmax}}\ \log(g(\theta))$$

- Taking the logarithm of the likelihood

$$\log \prod_{n=1}^{N} \hat{p}(y_n \,|\, \theta) = \sum_{n=1}^{N} log(\hat{p}(y_n \,|\, \theta))$$

- Assuming $\hat{p}$ is normally distributed we derive the log-likelihood:

$$\log L(\theta) = \sum_{n=1}^{N} \log \left(\frac{1}{\sqrt{2\pi\hat{\sigma}}} e^{-\frac{(y_n - \hat{y}_n)^2}{2\hat{\sigma}^2}} \right)$$

Maximum Likelihood Estimation - III

- Deriving further:

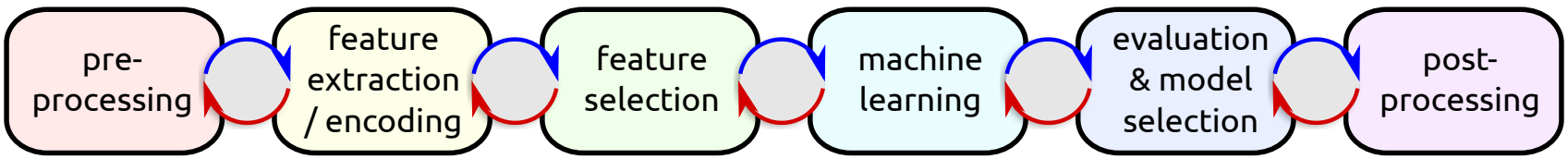
$$\begin{aligned}\log L(\theta) &= \sum_{n=1}^{N} \log\left(\frac{1}{\sqrt{2\pi\hat{\sigma}}} e^{-\frac{(y_n-\hat{y}_n)^2}{2\hat{\sigma}^2}}\right) \\ &= \sum_{n=1}^{N} \log\left(\frac{1}{\sqrt{2\pi\hat{\sigma}}}\right) + \log\left(e^{-\frac{(y_n-\hat{y}_n)^2}{2\hat{\sigma}^2}}\right)\end{aligned}$$

- Omitting the constant term above with respect to the parameters θ:

$$\begin{aligned}\underset{\theta}{\text{argmax}} \log L(\theta) &\approx \underset{\theta}{\text{argmax}} \frac{1}{2\hat{\sigma}^2} \sum_{n=1}^{N} -\left(y_n - \left(\theta_0 + \sum_{m=1}^{M} \theta_m x_m\right)\right)^2 \\ &\approx \underset{\theta}{\text{argmin}} \sum_{n=1}^{N} \left(y_n - \left(\theta_0 + \sum_{m=1}^{M} \theta_m x_m\right)\right)^2\end{aligned}$$

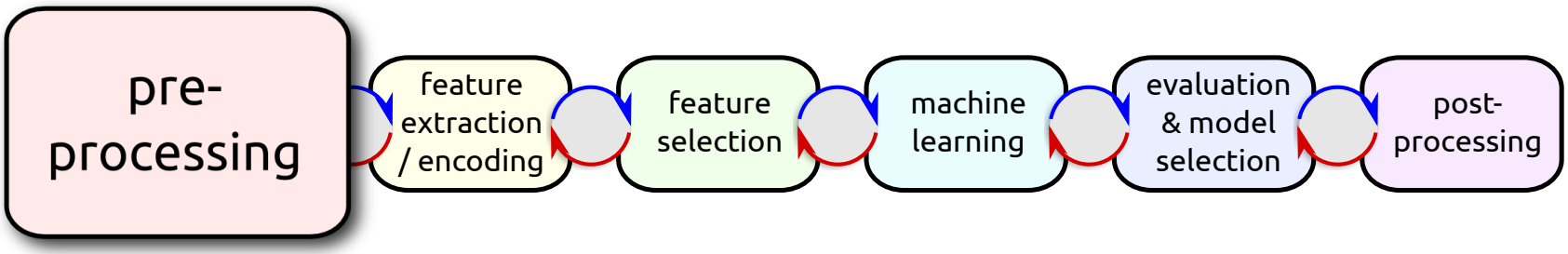
Table of Contents

The Machine Learning Design Cycle



- Real ML applications have many different steps, including:
  - Converting the data into targets and features (x,y)
  - Modeling your task as a ML problem
  - Evaluating and deploying your solution

ML Design Cycle: Preprocessing



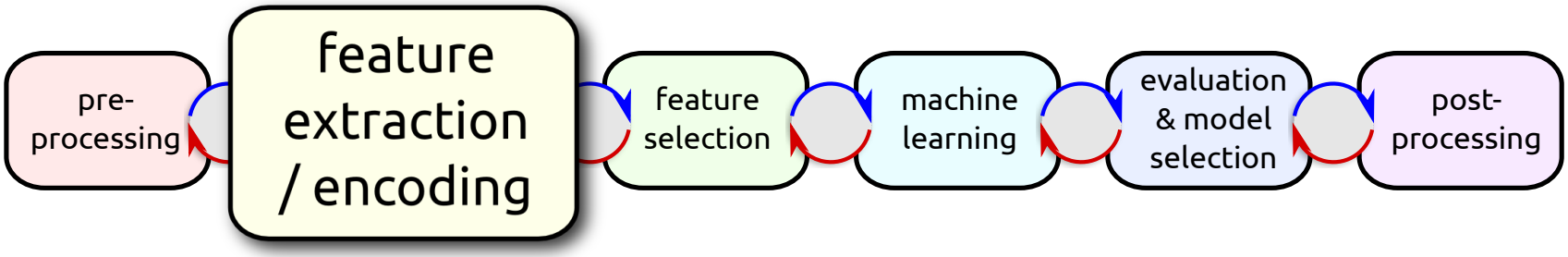
Collecting the data

- Design sensor systems, design questionaires
- Invest in labeling, get legal approval to use data, etc.

Cleaning the data

- Standardization
- Missing values
- Outliers
- Imbalanced target

ML Design Cycle: Feature Extraction & Encoding



Extract features:

- ... from a human cell, a text, a sound, a brainwave, an image

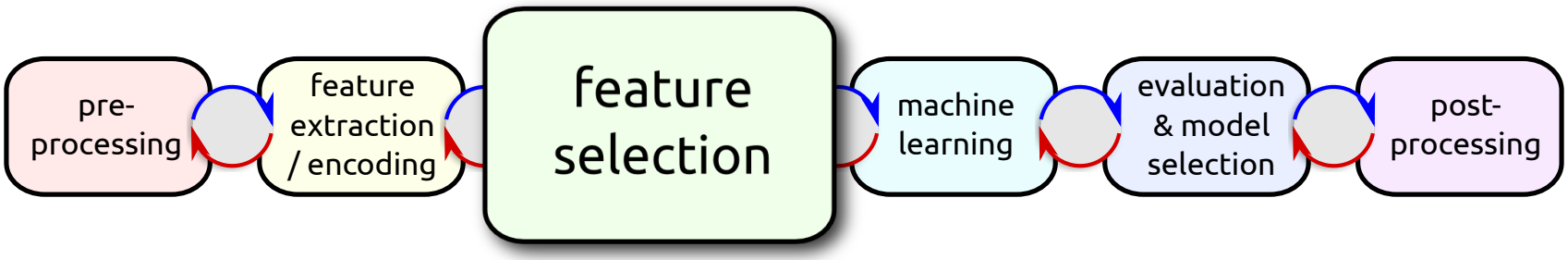
Engineer features:

- E.g., body mass index: BMI $=$ weight$/$ (height^2)

Encode features:

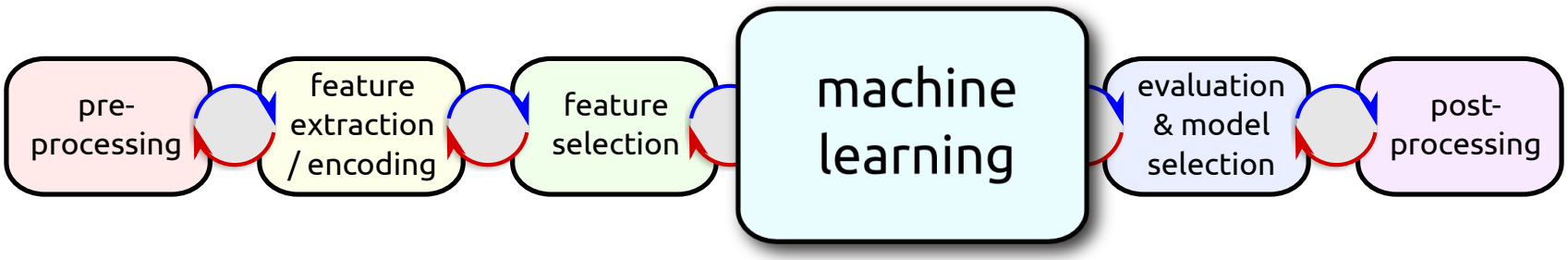
- One-hot encoding of categorical features

ML Design Cycle: Feature Selection



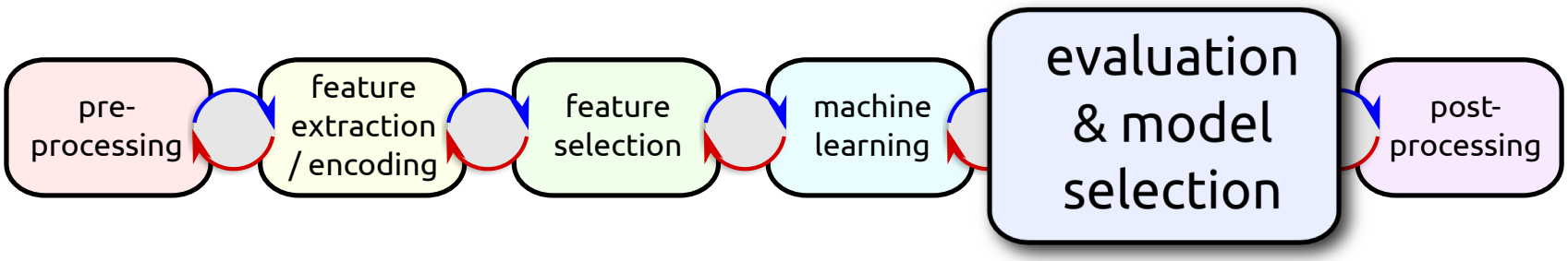
- E.g. recommendation system datasets have millions of features
- certain methods incorporate 'automatic' feature selection, others do not scale in terms of the number of features

ML Design Cycle: 'Machine Learning'



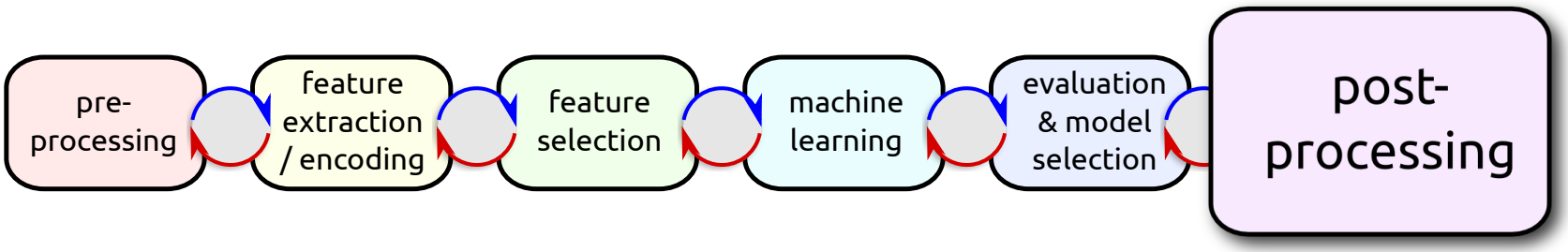
- Models: 'core' of machine learning
- In practice features are as important as models

ML Design Cycle: Evaluation and Model Selection 1/3



- Tuning the settings of models (hyper-parameters)
- ... ensures generalization.

The Machine Learning Design Cycle



- Ensure fairness / avoid algorithmic bias
  - E.g., model should not discriminate based on race, gender, religion, sexual orientation, ...
- Some models don't yield proper probabilities
  - Transform model outputs to probabilities

Table of Contents

Nearest Neighbors

- Predict the target $\hat{y} \in \mathcal{Y}$ of $x \in \mathcal{X}$
  - by aggregating the targets of the k neighbors of x with the
  - smallest distance $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$, e.g., $d(x, z) = ||x - z||_p$

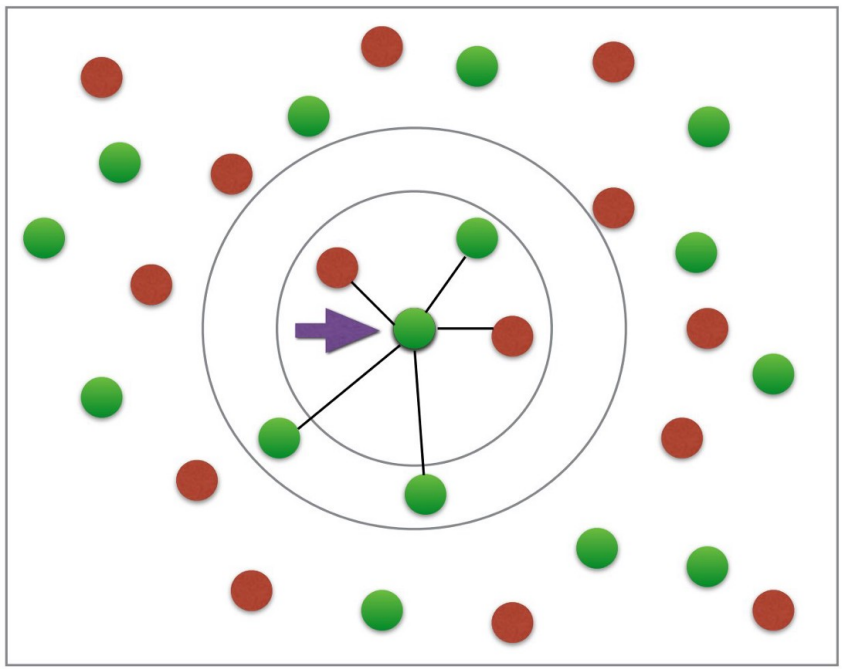


Figure 10: $\mathcal{X} = \mathbb{R}^2, \mathcal{Y} = \{red, green\}$, Credit: datacamp.com

Model

- Let our dataset be $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \ldots (x_N, y_N)\}$
- The set of the k nearest neighbors of x is $\mathcal{S}(x) \subset \mathcal{D}$:

$$\mathcal{S}(x) := \underset{\mathcal{S}(x) \subset \mathcal{D},\ |\mathcal{S}(x)| = k}{\operatorname{argmin}} \sum_{(x', y') \in \mathcal{S}(x)} d\left(x', x\right)$$

- The prediction of the nearest neighbor model $f(x)$ is:

$$f(x) := \text{aggregate}\left(\pi_y\left(\mathcal{S}(x)\right)\right)$$

Predictions

Targets of the k nearest neighbors of x are $y_x^{(\text{nearest})} := \pi_y\left(\mathcal{S}(x)\right)$

- For a continuous target the aggregation is:

$$f(x) := \frac{1}{k} \sum_{y' \in y_x^{(\text{nearest})}} y'$$

- for nominal targets (class c) the aggregation is:

$$f(x)_c := \frac{1}{k} \left| y' \in y_x^{(\text{nearest})} \mid y' = c \right|$$

Naive Bayes - Bayes Rule

Assume $x \in \mathcal{X}, y \in \mathcal{Y}$:

$$P(y \mid x) = P(y)\frac{P(x \mid y)}{P(x)}$$
$$= P(y)\frac{P(x \mid y)}{\sum_{y' \in \mathcal{Y}} P(x \mid y')P(y')}$$

Naive Bayes - Bayes Rule

Assume $x \in \mathcal{X}, y \in \mathcal{Y}$:

$$P(y \mid x) = P(y)\frac{P(x \mid y)}{P(x)}$$
$$= P(y)\frac{P(x \mid y)}{\sum_{y' \in \mathcal{Y}} P(x \mid y')P(y')}$$

- Let $y :=$ success $\in \{yes, no\}$ and $x :=$ plan $\in \{yes, no\}$

$$P(\text{success} = yes \mid \text{plan} = no)$$
$$= P(\text{success} = yes)\frac{P(\text{plan} = no \mid \text{success} = yes)}{P(\text{plan} = no)}$$

Bayes Rule - COVID-19 Example

- Let $y := \text{covid} \in \{pos, neg\}$ and $x := \text{fever} \in \{high, low\}$
- Compute $P(\text{covid} = pos \,|\, \text{fever} = low) = ?$

$$\implies P(\text{covid} = pos) \frac{P(\text{fever}{=}low \,|\, \text{covid}{=}pos)}{P(\text{fever}{=}low)}$$
$$\approx 0.43 \cdot \frac{0.33}{0.57} \approx 0.25$$

$$P(\text{fever} = low)$$
$$= P(\text{fever} = low \,|\, \text{covid} = pos) P(\text{covid} = pos)$$
$$+ P(\text{fever} = low \,|\, \text{covid} = neg) P(\text{covid} = neg)$$
$$= 0.33 \cdot 0.43 + 0.75 \cdot 0.57 = 0.57$$

Table 2: COVID-19 data

covid	**fever**
pos	low
neg	low
neg	high
pos	high
neg	low
neg	low
pos	high

Multiple conditional variables

$$P(y \mid x_1, x_2 \ldots, x_M) = P(y) \frac{P(x_1, x_2 \ldots, x_M \mid y)}{P(x_1, x_2 \ldots, x_M)}$$

Multiple conditional variables

$$P(y \mid x_1, x_2 \ldots, x_M) = P(y)\frac{P(x_1, x_2 \ldots, x_M \mid y)}{P(x_1, x_2 \ldots, x_M)}$$

- Let $y :=$ covid $\in \{pos, neg\}$ and $x_1 :=$ fever $\in \{high, low\}$, $x_2 :=$ cough $\in \{yes, no\}$, $x_3 :=$ headache $\in \{yes, no\}$

covid	**fever**	**cough**	**headache**
pos	low	yes	no
neg	low	yes	yes
neg	high	no	no
pos	high	yes	yes
neg	low	no	yes
neg	low	yes	no
pos	high	no	yes

$$P(\text{covid} = pos \mid \text{fever} = high, \text{cough} = yes, \text{headache} = no) = ?$$

Naive Bayes

Assume $x_1, x_2 \ldots, x_M$ are all independent given y:

$$\begin{aligned} P(y \mid x_1, x_2 \ldots, x_M) &= P(y) \frac{P(x_1, x_2 \ldots, x_M \mid y)}{P(x_1, x_2 \ldots, x_M)} \\ &= P(y) \frac{P(x_1|y)P(x_2|y) \ldots P(x_M|y)}{P(x_1, x_2 \ldots, x_M)} \end{aligned}$$

If the goal is only to predict y we can drop the denominator:

$$P(y \mid x_1, x_2 \ldots, x_M) \propto P(y)P(x_1|y)P(x_2|y) \ldots P(x_M|y)$$

COVID-19 Example

$$P(\text{covid} = ? \mid \text{fever} = \textit{high}, \text{cough} = \textit{yes}, \text{headache} = \textit{no})$$

covid	fever	cough	headache
pos	low	yes	no
neg	low	yes	yes
neg	high	no	no
pos	high	yes	yes
neg	low	no	yes
neg	low	yes	no
pos	high	no	yes

$$P(\text{covid} = \textit{pos}) = 0.43$$
$$P(\text{covid} = \textit{neg}) = 0.57$$
$$P(\text{fever} = \textit{high} \mid \text{covid} = \textit{pos}) = 0.67$$
$$P(\text{fever} = \textit{high} \mid \text{covid} = \textit{neg}) = 0.25$$
$$P(\text{cough} = \textit{yes} \mid \text{covid} = \textit{pos}) = 0.67$$
$$P(\text{cough} = \textit{yes} \mid \text{covid} = \textit{neg}) = 0.5$$
$$P(\text{headache} = \textit{no} \mid \text{covid} = \textit{pos}) = 0.33$$
$$P(\text{headache} = \textit{no} \mid \text{covid} = \textit{neg}) = 0.5$$

$$\begin{aligned}
&P(\text{covid} = \textit{pos} \mid \text{fever} = \textit{high}, \text{cough} = \textit{yes}, \text{headache} = \textit{no}) \\
&\propto P(\text{covid} = \textit{pos}) \cdot P(\text{fever} = \textit{high} \mid \text{covid} = \textit{pos}) \\
&\quad \cdot P(\text{cough} = \textit{yes} \mid \text{covid} = \textit{pos}) \cdot P(\text{headache} = \textit{no} \mid \text{covid} = \textit{pos}) \\
&= 0.43 \cdot 0.67 \cdot 0.67 \cdot 0.33 = 0.064 \\
&P(\text{covid} = \textit{neg} \mid \text{fever} = \textit{high}, \text{cough} = \textit{yes}, \text{headache} = \textit{no}) \\
&\propto 0.57 \cdot 0.25 \cdot 0.5 \cdot 0.5 = 0.036
\end{aligned}$$