Uncertainty Quantification in Machine Learning

From Aleatoric to Epistemic

Eyke Hüllermeier

Institute of Informatics, LMU Munich, Germany

Tutorial at MASCOT-NUM, June 9, 2022

Introduction and motivation

Introduction and motivation

- Machine learning is inseparably connected with **uncertainty**.

Introduction and motivation

- Machine learning is inseparably connected with **uncertainty**.
- **Learning** in the sense of generalising beyond the data seen so far is necessarily based on a process of **induction**.

Introduction and motivation

- Machine learning is inseparably connected with **uncertainty**.
- **Learning** in the sense of generalising beyond the data seen so far is necessarily based on a process of **induction**.
- **Models** induced from data are never provably correct, but hypothetical and therefore uncertain, and the same holds true for the **predictions** produced by a model.

Introduction and motivation

- Machine learning is inseparably connected with **uncertainty**.
- **Learning** in the sense of generalising beyond the data seen so far is necessarily based on a process of **induction**.
- **Models** induced from data are never provably correct, but hypothetical and therefore uncertain, and the same holds true for the **predictions** produced by a model.
- Other sources of uncertainty exist: incorrect **model assumptions** (model misspecification), noisy or imprecise **data**, etc.

Introduction and motivation

- Machine learning is inseparably connected with **uncertainty**.
- **Learning** in the sense of generalising beyond the data seen so far is necessarily based on a process of **induction**.
- **Models** induced from data are never provably correct, but hypothetical and therefore uncertain, and the same holds true for the **predictions** produced by a model.
- Other sources of uncertainty exist: incorrect **model assumptions** (model misspecification), noisy or imprecise **data**, etc.
- **Trustworthy representation** of uncertainty is desirable and should be considered as a key feature of any machine learning method, all the more in safety-critical application domains.

Self-awareness of ML systems

Self-awareness of ML systems

- Many applications require safe and reliable predictions, and hence a certain level of **self-awareness** of ML systems:
  - equip predictions with an appropriate **quantification of uncertainty**,
  - **reject** a decision in cases of high uncertainty (abstention) ,
  - deliver a credible **set-valued prediction** (partial abstention),
  - ...

Driver assistance systems: a safety-critical application

Adversarial examples

There is really but one thing to say about **this** sorry movie It should never have been made The first one one of my favourites An American Werewolf in London is a great movie with a good plot good actors and good FX But this one It stinks to heaven with a cry of helplessness

negative sentiment

There is really but one thing to say about **that** sorry movie It should never have been made The first one one of my favourites An American Werewolf in London is a great movie with a good plot good actors and good FX But this one It stinks to heaven with a cry of helplessness

positive sentiment

Adversarial examples

There is really but one thing to say about **this** sorry movie It should never have been made The first one one of my favourites An American Werewolf in London is a great movie with a good plot good actors and good FX But this one It stinks to heaven with a cry of helplessness

negative sentiment

There is really but one thing to say about **that** sorry movie It should never have been made The first one one of my favourites An American Werewolf in London is a great movie with a good plot good actors and good FX But this one It stinks to heaven with a cry of helplessness

positive sentiment

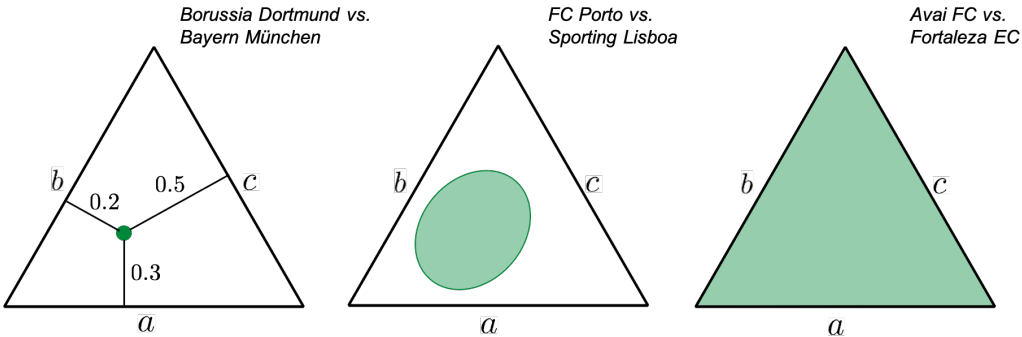
- The **adversarial example** (right) is misclassified by a machine learning model trained on textual data, which changes its prediction due to a change of a single (actually unimportant) word (Sato *et al.*, 2018).

Lack of uncertainty-awareness

Lack of uncertainty-awareness

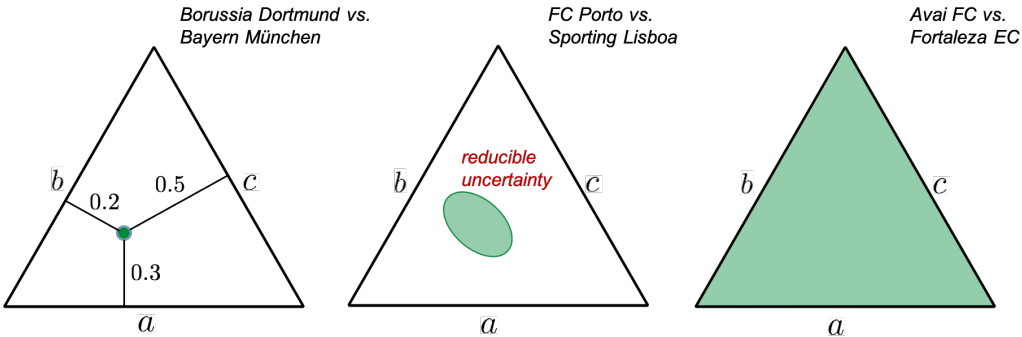
- Predictions by EfficientNet on test images from ImageNet: For the left image, the neural network predicts “typewriter keyboard” with certainty 83.14 %, for the right image “stone wall” with certainty 87.63 %.

Levels of self-awareness and uncertainty representation



Probability distributions $p = (p(a), p(b), p(c))$ on $\Omega = \{a, b, c\}$, for example $\Omega = \{\text{home wins}, \text{draw}, \text{away wins}\}$, as points in a Barycentric coordinate system.

Levels of self-awareness and uncertainty representation



Probability distributions $p = (p(a), p(b), p(c))$ on $\Omega = \{a, b, c\}$, for example $\Omega = \{\text{home wins}, \text{draw}, \text{away wins}\}$, as points in a Barycentric coordinate system.

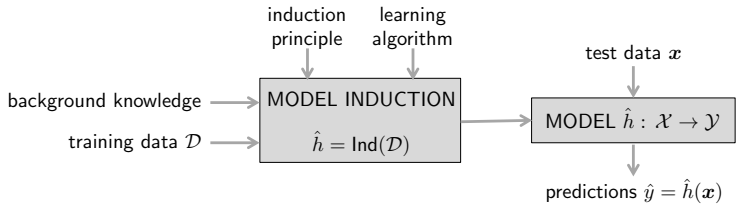
Supervised learning

Supervised learning

- Uncertainty occurs in various facets in machine learning, and different **settings and learning problems** will usually require a different handling from an uncertainty modeling point of view.

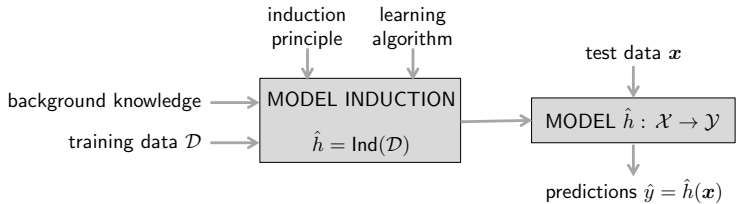
Supervised learning

- Uncertainty occurs in various facets in machine learning, and different **settings and learning problems** will usually require a different handling from an uncertainty modeling point of view.
- Here, we focus on the standard setting of **supervised learning** and **predictive uncertainty**.



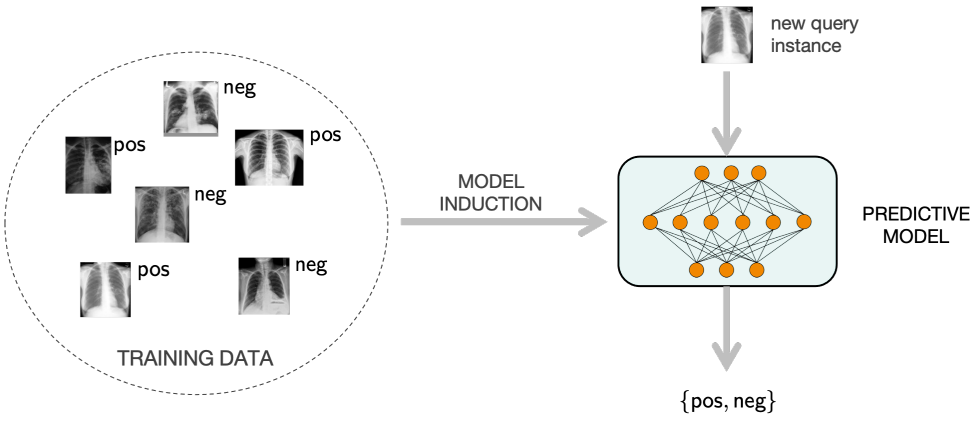
Supervised learning

- Uncertainty occurs in various facets in machine learning, and different **settings and learning problems** will usually require a different handling from an uncertainty modeling point of view.
- Here, we focus on the standard setting of **supervised learning** and **predictive uncertainty**.



- Assuming a probabilistic data generating process $p(\boldsymbol{x}, y) = p(\boldsymbol{x})p(y \mid \boldsymbol{x})$, **probabilistic predictors** (estimating $p(y \mid \boldsymbol{x})$) are natural primitives.

Supervised learning



Supervised learning

Supervised learning

- A learner is given access to a set of (i.i.d.) **training data**

$$\mathcal{D} := \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\} \subset \mathcal{X} \times \mathcal{Y} \ ,$$

  where $\mathcal{X}$ is an instance space and $\mathcal{Y}$ the set of outcomes.

Supervised learning

- A learner is given access to a set of (i.i.d.) **training data**

$$\mathcal{D} := \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\} \subset \mathcal{X} \times \mathcal{Y} \ ,$$

  where $\mathcal{X}$ is an instance space and $\mathcal{Y}$ the set of outcomes.
- Given a **hypothesis space** $\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}$ and a loss function

$$\ell : \mathcal{Y} \times \mathcal{Y} \longrightarrow \mathbb{R} \, ,$$

  the goal of the learner is to induce a hypothesis $h^* \in \mathcal{H}$ with low **risk**

$$R(h) := \int_{\mathcal{X} \times \mathcal{Y}} \ell(h(\boldsymbol{x}), y) \, d \, P(\boldsymbol{x}, y) \ .$$

Classification and regression

Classification and regression

- In the case of **regression**, where the target is numerical ($\mathcal{Y} = \mathbb{R}$), a common loss is the squared error:

$$\ell(h(\boldsymbol{x}), y) = \big(h(\boldsymbol{x}) - y\big)^2$$

Classification and regression

- In the case of **regression**, where the target is numerical ($\mathcal{Y} = \mathbb{R}$), a common loss is the squared error:
$$\ell(h(\boldsymbol{x}), y) = \big(h(\boldsymbol{x}) - y\big)^2$$
- In the case of **classification**, the outcome is categorical ($\mathcal{Y} = [K] \coloneqq \{1, \dots, K\}$), and a common choice is the $0/1$ loss:
$$\ell(h(\boldsymbol{x}), y) = [\![h(\boldsymbol{x}) \neq y]\!]$$
Then, however, a convex, (smooth) upper approximation is typically used as a surrogate for training.

Classification and regression

- In the case of **regression**, where the target is numerical ($\mathcal{Y} = \mathbb{R}$), a common loss is the squared error:
$$\ell(h(\boldsymbol{x}), y) = \big(h(\boldsymbol{x}) - y\big)^2$$
- In the case of **classification**, the outcome is categorical ($\mathcal{Y} = [K] := \{1, \dots, K\}$), and a common choice is the $0/1$ loss:
$$\ell(h(\boldsymbol{x}), y) = [\![h(\boldsymbol{x}) \neq y]\!]$$
Then, however, a convex, (smooth) upper approximation is typically used as a surrogate for training.
- If predictions are **probabilities**
$$h(\boldsymbol{x}) = \hat{\boldsymbol{p}} = (\hat{p}_1, \dots, \hat{p}_K) \in \Delta_K \,,$$
the loss is defined on $\Delta_K \times \mathcal{Y}$; a common example is the logistic loss (log-loss)
$$\ell(\hat{\boldsymbol{p}}, y) = -\log(p_y) \,.$$

Empirical risk minimisation

Empirical risk minimisation

- The learner's choice of a hypothesis is commonly guided by the **empirical risk**

$$R_{emp}(h) := \frac{1}{N} \sum_{i=1}^{N} \ell(h(\boldsymbol{x}_i), y_i) \ .$$

Empirical risk minimisation

- The learner's choice of a hypothesis is commonly guided by the **empirical risk**

$$R_{emp}(h) := \frac{1}{N} \sum_{i=1}^{N} \ell(h(\boldsymbol{x}_i), y_i) \ .$$

- Yet, since $R_{emp}(h)$ is only an estimation of the true risk $R(h)$, the (regularised) **empirical risk minimiser**

$$\hat{h} := \arg\min_{h \in \mathcal{H}} R_{emp}(h) + \Omega(h)$$

will normally not coincide with the true **risk minimiser**

$$h^* := \arg\min_{h \in \mathcal{H}} R(h) \, .$$

Empirical risk minimisation

- The learner's choice of a hypothesis is commonly guided by the **empirical risk**

$$R_{emp}(h) := \frac{1}{N} \sum_{i=1}^{N} \ell(h(\boldsymbol{x}_i), y_i) \ .$$

- Yet, since $R_{emp}(h)$ is only an estimation of the true risk $R(h)$, the (regularised) **empirical risk minimiser**

$$\hat{h} := \arg\min_{h \in \mathcal{H}} R_{emp}(h) + \Omega(h)$$

will normally not coincide with the true **risk minimiser**

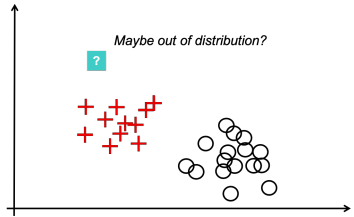
$$h^* := \arg\min_{h \in \mathcal{H}} R(h) \, .$$

- Correspondingly, there remains **uncertainty** regarding h^*, the approximation quality of $\hat{h}$ (in the sense of its proximity to h^*) and its true risk $R(\hat{h})$, as well as **predictive uncertainty** about $\hat{y}_q$ for an individual **query instance** $\boldsymbol{x}_q \in \mathcal{X}$.

Problem setting and assumptions

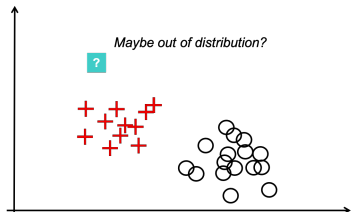
Problem setting and assumptions

- A precise specification of the **problem setting** and **underlying assumptions** is an important prerequisite, not only for providing learning guarantees, but also for **uncertainty quantification**.



Problem setting and assumptions

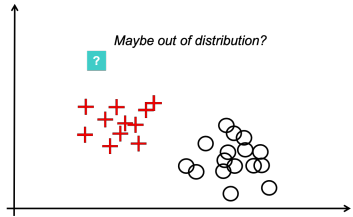
- A precise specification of the **problem setting** and **underlying assumptions** is an important prerequisite, not only for providing learning guarantees, but also for **uncertainty quantification**.



- Here, one might be quite sure about the class of the query under standard assumptions of binary classification, but much less so in a setting of **novelty detection**, where new classes may emerge.

Problem setting and assumptions

- A precise specification of the **problem setting** and **underlying assumptions** is an important prerequisite, not only for providing learning guarantees, but also for **uncertainty quantification**.



- Here, one might be quite sure about the class of the query under standard assumptions of binary classification, but much less so in a setting of **novelty detection**, where new classes may emerge.
- Likewise, assumptions such as **i.i.d. data generation** are really crucial (the past should be representative of the future).

Sources of uncertainty

Sources of uncertainty

- A query instance $\boldsymbol{x}_q$ gives rise to a conditional probability on $\mathcal{Y}$:

$$p(y \mid \boldsymbol{x}_q) = \frac{p(\boldsymbol{x}_q, y)}{p(\boldsymbol{x}_q)}$$

Sources of uncertainty

- A query instance $\boldsymbol{x}_q$ gives rise to a conditional probability on $\mathcal{Y}$:

$$p(y \mid \boldsymbol{x}_q) = \frac{p(\boldsymbol{x}_q, y)}{p(\boldsymbol{x}_q)}$$

- Thus, even given full information in the form of the measure P (and its density p), uncertainty about the actual outcome y remains.

Sources of uncertainty

- A query instance $\boldsymbol{x}_q$ gives rise to a conditional probability on $\mathcal{Y}$:

$$p(y \mid \boldsymbol{x}_q) = \frac{p(\boldsymbol{x}_q, y)}{p(\boldsymbol{x}_q)}$$

- Thus, even given full information in the form of the measure P (and its density p), uncertainty about the actual outcome y remains.
- The best point predictions (minimizing expected loss) are prescribed by the **pointwise Bayes predictor** f^*:

$$f^*(\boldsymbol{x}) := \operatorname*{arg\,min}_{\hat{y} \in \mathcal{Y}} \int_{\mathcal{Y}} \ell(y, \hat{y}) \, dP(y \mid \boldsymbol{x}) \,.$$

Sources of uncertainty

Sources of uncertainty

- The **Bayes predictor** h^* does not necessarily coincide with the pointwise Bayes predictor.

Sources of uncertainty

- The **Bayes predictor** h^* does not necessarily coincide with the pointwise Bayes predictor.
- This discrepancy between h^* and f^* is connected to the uncertainty regarding the **right type of model** to be fit, and hence the choice of the hypothesis space $\mathcal{H}$.

Sources of uncertainty

- The **Bayes predictor** h^* does not necessarily coincide with the pointwise Bayes predictor.
- This discrepancy between h^* and f^* is connected to the uncertainty regarding the **right type of model** to be fit, and hence the choice of the hypothesis space $\mathcal{H}$.
- We shall refer to this uncertainty as **model uncertainty**.

Sources of uncertainty

- The **Bayes predictor** h^* does not necessarily coincide with the pointwise Bayes predictor.
- This discrepancy between h^* and f^* is connected to the uncertainty regarding the **right type of model** to be fit, and hence the choice of the hypothesis space $\mathcal{H}$.
- We shall refer to this uncertainty as **model uncertainty**.
- Due to model uncertainty, one cannot guarantee

$$h^*(\boldsymbol{x}) = f^*(\boldsymbol{x}),$$

or, in the case of probabilistic predictions $p^*(y \mid \boldsymbol{x}) := p(y \mid \boldsymbol{x}, h^*)$, that

$$p^*(\cdot \mid \boldsymbol{x}) = p(\cdot \mid \boldsymbol{x}).$$

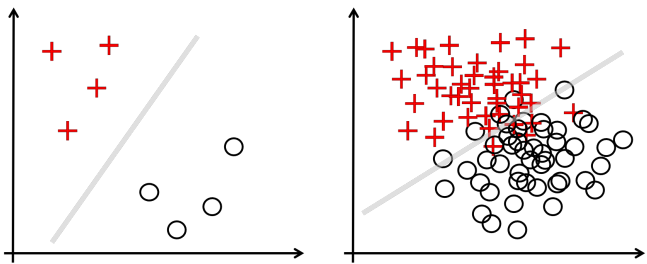
Sources of uncertainty

Sources of uncertainty

- Hypothesis $\hat{h}$ produced by the learner is an estimate of h^*.

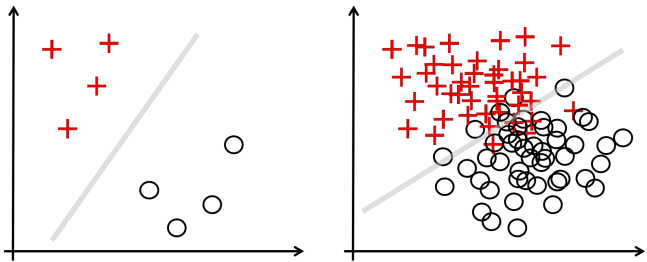
Sources of uncertainty

- Hypothesis $\hat{h}$ produced by the learner is an estimate of h^*.
- The quality of this estimate strongly depends on the quality and the amount of training data.



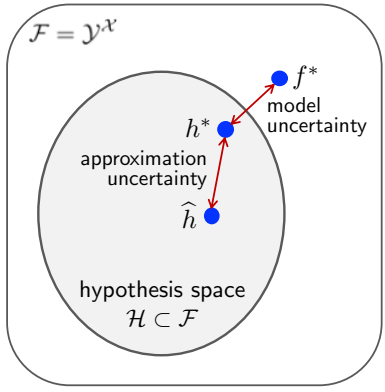
Sources of uncertainty

- Hypothesis $\hat{h}$ produced by the learner is an estimate of h^*.
- The quality of this estimate strongly depends on the quality and the amount of training data.



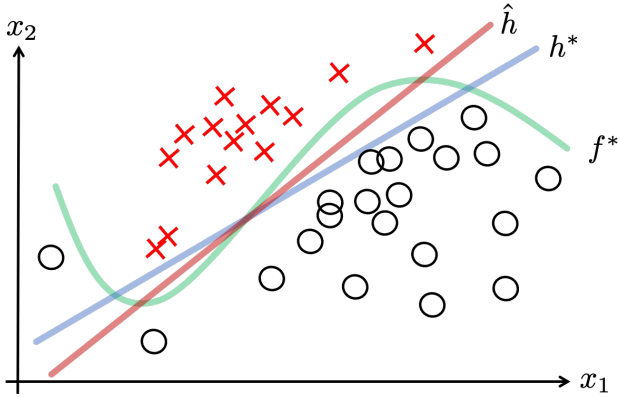
- We refer to the uncertainty about the discrepancy between $\hat{h}$ and h^* as **approximation uncertainty**.

Sources of uncertainty



	point prediction	probability
ground truth	$f^*(\boldsymbol{x})$	$p(\cdot \mid \boldsymbol{x})$
best possible	$h^*(\boldsymbol{x})$	$p^*(\cdot \mid \boldsymbol{x}) = p(\cdot \mid \boldsymbol{x}, h^*)$
induced predictor	$\hat{h}(\boldsymbol{x})$	$\hat{p}(\cdot \mid \boldsymbol{x}) = p(\cdot \mid \boldsymbol{x}, \hat{h})$

Sources of uncertainty



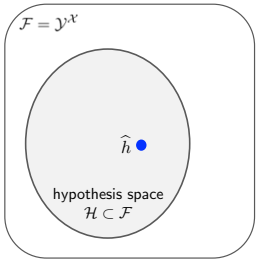
Agenda

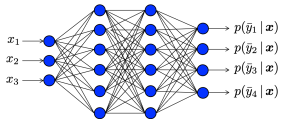
1. Introduction
2. **Training probabilistic predictors**
3. Calibration
4. Set-valued (conformal) prediction
5. Epistemic uncertainty

Probabilistic models

Probabilistic models

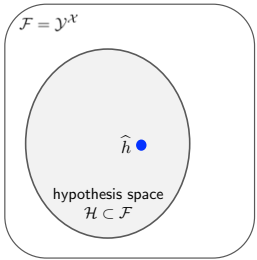
- Probabilistic learners produce a single probabilistic predictor $\hat{h}$:

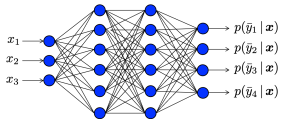




Probabilistic models

- Probabilistic learners produce a single probabilistic predictor $\hat{h}$:

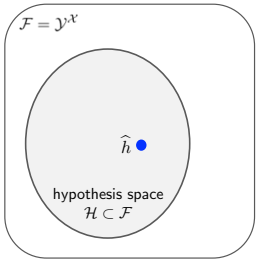


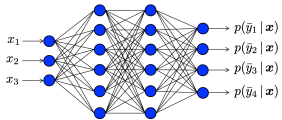


- Captures stochastic nature of dependence $p(y \mid \boldsymbol{x})$ between instances and outcomes, and hence **aleatoric uncertainty**.

Probabilistic models

- Probabilistic learners produce a single probabilistic predictor $\hat{h}$:





- Captures stochastic nature of dependence $p(y \mid \boldsymbol{x})$ between instances and outcomes, and hence **aleatoric uncertainty**.
- Yet, pretends full certainty about this dependence, thereby ignoring approximation and model uncertainty (epistemic uncertainty).

Maximum likelihood

Maximum likelihood

- Suppose that hypotheses h are (uniquely) identified by parameters $\boldsymbol{\theta} \in \Theta$, i.e.,

$$\mathcal{H} = \{h_{\boldsymbol{\theta}} \mid \boldsymbol{\theta} \in \Theta\}.$$

  Learning (model induction) then comes down to **parameter estimation**.

Maximum likelihood

- Suppose that hypotheses h are (uniquely) identified by parameters $\boldsymbol{\theta} \in \Theta$, i.e.,

$$\mathcal{H} = \{h_{\boldsymbol{\theta}} \mid \boldsymbol{\theta} \in \Theta\}.$$

  Learning (model induction) then comes down to **parameter estimation**.

- The **maximum likelihood** (ML) principle suggests to pick the parameter with the highest likelihood:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta} \in \Theta} L(\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta} \in \Theta} P(\mathcal{D} \mid \boldsymbol{\theta})$$

Maximum likelihood

- Suppose that hypotheses h are (uniquely) identified by parameters $\boldsymbol{\theta} \in \Theta$, i.e.,

$$\mathcal{H} = \{h_{\boldsymbol{\theta}} \mid \boldsymbol{\theta} \in \Theta\}.$$

  Learning (model induction) then comes down to **parameter estimation**.

- The **maximum likelihood** (ML) principle suggests to pick the parameter with the highest likelihood:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta} \in \Theta} L(\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta} \in \Theta} P(\mathcal{D} \mid \boldsymbol{\theta})$$

- In general, ML estimation has good statistical properties.

Maximum likelihood

- Suppose that hypotheses h are (uniquely) identified by parameters $\boldsymbol{\theta} \in \Theta$, i.e.,

$$\mathcal{H} = \{h_{\boldsymbol{\theta}} \,|\, \boldsymbol{\theta} \in \Theta\}\,.$$

  Learning (model induction) then comes down to **parameter estimation**.

- The **maximum likelihood** (ML) principle suggests to pick the parameter with the highest likelihood:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}\in\Theta} L(\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}\in\Theta} P(\mathcal{D} \,|\, \boldsymbol{\theta})$$

- In general, ML estimation has good statistical properties.
- Under the i.i.d. assumption, ML estimation comes down to solving

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}\in\Theta} \prod_{i=1}^{N} P\big((\boldsymbol{x}_i, y_i) \,|\, \boldsymbol{\theta}\big) = \arg\max_{\boldsymbol{\theta}\in\Theta} \sum_{i=1}^{N} \log\, P\big((\boldsymbol{x}_i, y_i) \,|\, \boldsymbol{\theta}\big)$$

Logistic regression

Logistic regression

- Suppose $\mathcal{X} = \mathbb{R}^d$, i.e., instances are feature vectors $\boldsymbol{x} = (x_1, \ldots, x_d)$, and $\mathcal{Y} = \{-1, +1\}$ (binary classification).

Logistic regression

- Suppose $\mathcal{X} = \mathbb{R}^d$, i.e., instances are feature vectors $\boldsymbol{x} = (x_1, \ldots, x_d)$, and $\mathcal{Y} = \{-1, +1\}$ (binary classification).
- In **logistic regression**, the probability of the positive class is modeled as follows:

$$p(y = +1 \,|\, \boldsymbol{x}) = \frac{1}{1 + \exp\big(-\langle \boldsymbol{\theta}, \boldsymbol{x} \rangle\big)}$$

Logistic regression

- Suppose $\mathcal{X} = \mathbb{R}^d$, i.e., instances are feature vectors $\boldsymbol{x} = (x_1, \ldots, x_d)$, and $\mathcal{Y} = \{-1, +1\}$ (binary classification).
- In **logistic regression**, the probability of the positive class is modeled as follows:

$$p(y = +1 \,|\, \boldsymbol{x}) = \frac{1}{1 + \exp\big(-\langle \boldsymbol{\theta}, \boldsymbol{x} \rangle\big)}$$

- We can think of the above model as follows: First, an instance $\boldsymbol{x}$ is assigned a (latent) score (expressing the propensity for the positive class)

$$s = s(\boldsymbol{x}) = \big\langle \boldsymbol{\theta}, \boldsymbol{x} \big\rangle \,,$$

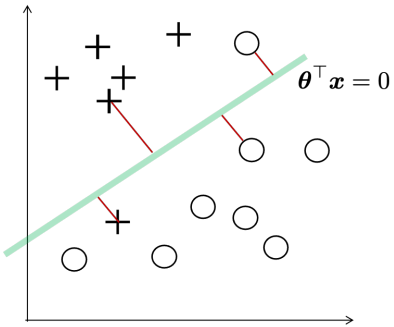
  which is then transformed into a probability via the (logistic) **link function**

$$\phi(s) = \frac{1}{1 + \exp(-s)} \,.$$

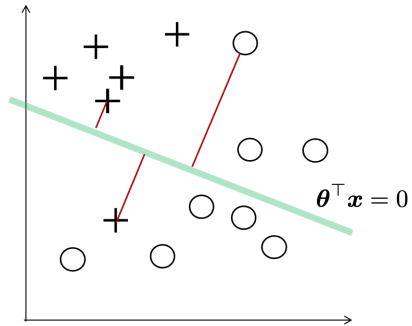
Logistic regression

Logistic regression

- The probability of a label y_i depends on the distance of $\boldsymbol{x}_i$ from the hyperplane defined by $\boldsymbol{\theta}^\top \boldsymbol{x} = 0$.



higher likelihood



lower likelihood

Logistic regression

Logistic regression

- In order to learn $\boldsymbol{\theta}$, we can invoke the ML principle:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta} \in \mathbb{R}^d} \sum_{n=1}^{N} \log p(y_n \,|\, \boldsymbol{x}_n)$$

Logistic regression

- In order to learn $\boldsymbol{\theta}$, we can invoke the ML principle:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta} \in \mathbb{R}^d} \sum_{n=1}^{N} \log p(y_n \,|\, \boldsymbol{x}_n)$$

- This is equivalent to ERM for the **logistic loss** (log-loss) or cross-entropy error

$$\ell(y, s) = \log\left(1 + \exp\left(-y \cdot s\right)\right).$$

Logistic regression

- In order to learn $\boldsymbol{\theta}$, we can invoke the ML principle:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta} \in \mathbb{R}^d} \sum_{n=1}^{N} \log p(y_n \,|\, \boldsymbol{x}_n)$$

- This is equivalent to ERM for the **logistic loss** (log-loss) or cross-entropy error

$$\ell(y, s) = \log\left(1 + \exp\left(-y \cdot s\right)\right).$$

- Probability is only maximised over the y_n, while the $\boldsymbol{x}_n$ are assumed to be fixed.

Logistic regression

- In order to learn $\boldsymbol{\theta}$, we can invoke the ML principle:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta} \in \mathbb{R}^d} \sum_{n=1}^{N} \log p(y_n \,|\, \boldsymbol{x}_n)$$

- This is equivalent to ERM for the **logistic loss** (log-loss) or cross-entropy error

$$\ell(y, s) = \log\left(1 + \exp\left(-y \cdot s\right)\right).$$

- Probability is only maximised over the y_n, while the $\boldsymbol{x}_n$ are assumed to be fixed.
- Logistic regression is an example of **discriminative** learning, i.e., it learns a map

$$h : \boldsymbol{x} \mapsto p(Y \,|\, \boldsymbol{x}).$$

This is to be distinguished from **generative** learning, which essentially means learning the entire data-generating process in the form of the joint distribution $p(\boldsymbol{X}, Y)$.

Bayesian learning

Bayesian learning

- In the Bayesian approach, learning corresponds to turning a **prior distribution** on $\mathcal{H}$ into a **posterior**:

$$p(h \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid h)\, p(h)}{p(\mathcal{D})} \propto p(\mathcal{D} \mid h)\, p(h)$$

Bayesian learning

- In the Bayesian approach, learning corresponds to turning a **prior distribution** on $\mathcal{H}$ into a **posterior**:

$$p(h \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid h)\, p(h)}{p(\mathcal{D})} \propto p(\mathcal{D} \mid h)\, p(h)$$

- The **predictive posterior** distribution on $\mathcal{Y}$ is obtained via **model averaging**:

$$p(y \mid \boldsymbol{x}_q) = \int_{h \in \mathcal{H}} p(y \mid \boldsymbol{x}_q, h)\, d\, P(h \mid \mathcal{D})$$

Bayesian learning

- In the Bayesian approach, learning corresponds to turning a **prior distribution** on $\mathcal{H}$ into a **posterior**:

$$p(h \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid h)\, p(h)}{p(\mathcal{D})} \propto p(\mathcal{D} \mid h)\, p(h)$$

- The **predictive posterior** distribution on $\mathcal{Y}$ is obtained via **model averaging**:

$$p(y \mid \boldsymbol{x}_q) = \int_{h \in \mathcal{H}} p(y \mid \boldsymbol{x}_q, h)\, d\, P(h \mid \mathcal{D})$$

- Bayesian inference is very costly but can be done approximately, for example, using **ensemble methods**.

Bayesian learning

- In the Bayesian approach, learning corresponds to turning a **prior distribution** on $\mathcal{H}$ into a **posterior**:

$$p(h \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid h)\, p(h)}{p(\mathcal{D})} \propto p(\mathcal{D} \mid h)\, p(h)$$

- The **predictive posterior** distribution on $\mathcal{Y}$ is obtained via **model averaging**:

$$p(y \mid \boldsymbol{x}_q) = \int_{h \in \mathcal{H}} p(y \mid \boldsymbol{x}_q, h)\, d\, P(h \mid \mathcal{D})$$

- Bayesian inference is very costly but can be done approximately, for example, using **ensemble methods**.
- An alternative is to commit to the hypothesis with maximum a-posteriori probability (**MAP** inference):

$$h_{MAP} = \arg\max_{h \in \mathcal{H}} p(h \mid \mathcal{D})\,.$$

Agenda

1. Introduction
2. Training probabilistic predictors
3. **Calibration**
4. Set-valued (conformal) prediction
5. Epistemic uncertainty

Proper scoring rules

Proper scoring rules

- The logistic (cross-entropy) loss is an example of a **proper scoring rule**.

Proper scoring rules

- The logistic (cross-entropy) loss is an example of a **proper scoring rule**.
- Consider a probabilistic prediction (for a query $\boldsymbol{x}$) on a set of K classes $\mathcal{Y} = [K]$ in the form of a probability vector

$$\hat{\boldsymbol{p}} = (\hat{p}_1, \ldots, \hat{p}_K) \in \Delta_K \,.$$

Proper scoring rules

- The logistic (cross-entropy) loss is an example of a **proper scoring rule**.
- Consider a probabilistic prediction (for a query $\boldsymbol{x}$) on a set of K classes $\mathcal{Y} = [K]$ in the form of a probability vector

$$\hat{\boldsymbol{p}} = (\hat{p}_1, \ldots, \hat{p}_K) \in \Delta_K \,.$$

- The true distribution (conditioned on $\boldsymbol{x}$) is $\boldsymbol{p} = (p_1, \ldots, p_K)$, i.e., the observed class Y is a random variable $Y \sim \boldsymbol{p}$ with multinomial distribution.

Proper scoring rules

- The logistic (cross-entropy) loss is an example of a **proper scoring rule**.
- Consider a probabilistic prediction (for a query $\boldsymbol{x}$) on a set of K classes $\mathcal{Y} = [K]$ in the form of a probability vector

$$\hat{\boldsymbol{p}} = (\hat{p}_1, \ldots, \hat{p}_K) \in \Delta_K \,.$$

- The true distribution (conditioned on $\boldsymbol{x}$) is $\boldsymbol{p} = (p_1, \ldots, p_K)$, i.e., the observed class Y is a random variable $Y \sim \boldsymbol{p}$ with multinomial distribution.
- We also encode a realisation y of Y in terms of a vector $\boldsymbol{y}$ with entry 1 on position $y = k$ (i.e., the k^{th} outcome was observed), and all other entries 0.

Proper scoring rules

- The logistic (cross-entropy) loss is an example of a **proper scoring rule**.
- Consider a probabilistic prediction (for a query $\boldsymbol{x}$) on a set of K classes $\mathcal{Y} = [K]$ in the form of a probability vector

$$\hat{\boldsymbol{p}} = (\hat{p}_1, \ldots, \hat{p}_K) \in \Delta_K \,.$$

- The true distribution (conditioned on $\boldsymbol{x}$) is $\boldsymbol{p} = (p_1, \ldots, p_K)$, i.e., the observed class Y is a random variable $Y \sim \boldsymbol{p}$ with multinomial distribution.
- We also encode a realisation y of Y in terms of a vector $\boldsymbol{y}$ with entry 1 on position $y = k$ (i.e., the k^{th} outcome was observed), and all other entries 0.
- Expressed in terms of predicted probability, the log-loss can then also be written as

$$\ell(\hat{\boldsymbol{p}}, y) = -\log \hat{p}_y \,.$$

Proper scoring rules

Proper scoring rules

- A loss ℓ is a proper scoring rule if the expected loss minimiser coincides with the true probability $\boldsymbol{p}$:

$$\boldsymbol{p} = \arg\min_{\hat{\boldsymbol{p}}} \mathbb{E}_{Y \sim \boldsymbol{p}} \ell(\hat{\boldsymbol{p}}, Y)$$

Proper scoring rules

- A loss ℓ is a proper scoring rule if the expected loss minimiser coincides with the true probability $\boldsymbol{p}$:

$$\boldsymbol{p} = \arg\min_{\hat{\boldsymbol{p}}} \mathbb{E}_{Y \sim \boldsymbol{p}} \ell(\hat{\boldsymbol{p}}, Y)$$

- A scoring rule is **strictly proper** if the minimiser is unique.

Proper scoring rules

- A loss ℓ is a proper scoring rule if the expected loss minimiser coincides with the true probability $\boldsymbol{p}$:

$$\boldsymbol{p} = \arg\min_{\hat{\boldsymbol{p}}} \mathbb{E}_{Y \sim \boldsymbol{p}} \, \ell(\hat{\boldsymbol{p}}, Y)$$

- A scoring rule is **strictly proper** if the minimiser is unique.

- Thus, given a query $\boldsymbol{x}$, a learner penalised by a (strictly) proper scoring rule has an incentive to predict the true (conditional) probability $\boldsymbol{p} = p(Y \mid \boldsymbol{x})$.

Proper scoring rules

- A loss ℓ is a proper scoring rule if the expected loss minimiser coincides with the true probability $\boldsymbol{p}$:

$$\boldsymbol{p} = \arg\min_{\hat{\boldsymbol{p}}} \mathbb{E}_{Y \sim \boldsymbol{p}} \ell(\hat{\boldsymbol{p}}, Y)$$

- A scoring rule is **strictly proper** if the minimiser is unique.
- Thus, given a query $\boldsymbol{x}$, a learner penalised by a (strictly) proper scoring rule has an incentive to predict the true (conditional) probability $\boldsymbol{p} = p(Y \mid \boldsymbol{x})$.
- For example, the **Brier score** $\sum_k (\hat{p}_k - y_k)^2$ is strictly proper, because

$$\sum_{k=1}^{K} p_k \cdot \left[(1 - \hat{p}_k)^2 + \sum_{j \neq k} (\hat{p}_j)^2 \right] = \sum_k (\hat{p}_k)^2 + \sum_k p_k (1 - 2\hat{p}_k)$$

  is minimised by $\hat{p}_k = p_k$ for all $k \in [K]$.

Proper scoring rules

Proper scoring rules

- Define the **scoring function** on probability vectors $\hat{\boldsymbol{p}}, \boldsymbol{p}$ as

$$S(\hat{\boldsymbol{p}}, \boldsymbol{p}) := \mathbb{E}_{Y \sim \boldsymbol{p}} S(\hat{\boldsymbol{p}}, Y) = \sum_k S(\hat{\boldsymbol{p}}, y)\, p_k \,,$$

  i.e., as the expected score (under ground-truth $\boldsymbol{p}$).

Proper scoring rules

- Define the **scoring function** on probability vectors $\hat{\boldsymbol{p}}, \boldsymbol{p}$ as

$$S(\hat{\boldsymbol{p}}, \boldsymbol{p}) := \mathbb{E}_{Y \sim \boldsymbol{p}} S(\hat{\boldsymbol{p}}, Y) = \sum_k S(\hat{\boldsymbol{p}}, y)\, p_k \,,$$

  i.e., as the expected score (under ground-truth $\boldsymbol{p}$).

- A scoring rule is proper if the **divergence**

$$d(\hat{\boldsymbol{p}}, \boldsymbol{p}) := S(\hat{\boldsymbol{p}}, \boldsymbol{p}) - S(\boldsymbol{p}, \boldsymbol{p})$$

  is nonnegative, and strictly proper if $d(\hat{\boldsymbol{p}}, \boldsymbol{p}) = 0$ implies $\hat{\boldsymbol{p}} = \boldsymbol{p}$.

Proper scoring rules

- Define the **scoring function** on probability vectors $\hat{\boldsymbol{p}}, \boldsymbol{p}$ as

$$S(\hat{\boldsymbol{p}}, \boldsymbol{p}) := \mathbb{E}_{Y \sim \boldsymbol{p}} S(\hat{\boldsymbol{p}}, Y) = \sum_k S(\hat{\boldsymbol{p}}, y)\, p_k \,,$$

  i.e., as the expected score (under ground-truth $\boldsymbol{p}$).
- A scoring rule is proper if the **divergence**

$$d(\hat{\boldsymbol{p}}, \boldsymbol{p}) := S(\hat{\boldsymbol{p}}, \boldsymbol{p}) - S(\boldsymbol{p}, \boldsymbol{p})$$

  is nonnegative, and strictly proper if $d(\hat{\boldsymbol{p}}, \boldsymbol{p}) = 0$ implies $\hat{\boldsymbol{p}} = \boldsymbol{p}$.
- $e(\boldsymbol{p}) := S(\boldsymbol{p}, \boldsymbol{p})$ is also called **entropy**.

Proper scoring rules

Proper scoring rules

- For the **log-loss**, we obtain a decomposition into KL-divergence and information (Shannon) entropy:

$$d(\hat{\boldsymbol{p}}, \boldsymbol{p}) = D_{KL}(\boldsymbol{p} \| \hat{\boldsymbol{p}}) = \sum_k p_k \cdot \log\left(\frac{p_K}{\hat{p}_k}\right)$$

$$e(\boldsymbol{p}) = -\sum_k p_k \cdot \log(p_k)$$

Proper scoring rules

- For the **log-loss**, we obtain a decomposition into KL-divergence and information (Shannon) entropy:

$$d(\hat{\boldsymbol{p}}, \boldsymbol{p}) = D_{KL}(\boldsymbol{p} \| \hat{\boldsymbol{p}}) = \sum_k p_k \cdot \log\left(\frac{p_K}{\hat{p}_k}\right)$$

$$e(\boldsymbol{p}) = -\sum_k p_k \cdot \log(p_k)$$

- For the **Brier score**, we obtain a decomposition into mean squared difference and Gini index:

$$d(\hat{\boldsymbol{p}}, \boldsymbol{p}) = \sum_k (\hat{p}_k - p_k)^2$$

$$e(\boldsymbol{p}) = \sum_k p_k(1 - p_k)$$

Proper scoring rules

- For the **log-loss**, we obtain a decomposition into KL-divergence and information (Shannon) entropy:

$$d(\hat{\boldsymbol{p}}, \boldsymbol{p}) = D_{KL}(\boldsymbol{p} \| \hat{\boldsymbol{p}}) = \sum_k p_k \cdot \log\left(\frac{p_K}{\hat{p}_k}\right)$$

$$e(\boldsymbol{p}) = -\sum_k p_k \cdot \log(p_k)$$

- For the **Brier score**, we obtain a decomposition into mean squared difference and Gini index:

$$d(\hat{\boldsymbol{p}}, \boldsymbol{p}) = \sum_k (\hat{p}_k - p_k)^2$$

$$e(\boldsymbol{p}) = \sum_k p_k(1 - p_k)$$

- The **0/1 loss** is a proper but not a strictly proper scoring rule.

Proper scoring rules

Proper scoring rules

- The expectation of a strictly proper score can be decomposed as follows:

$$\mathbb{E}_{Y\sim\boldsymbol{p}}\, S(\hat{\boldsymbol{p}}, Y) = S(\hat{\boldsymbol{p}}, \boldsymbol{p}) = d(\hat{\boldsymbol{p}}, \boldsymbol{p}) + S(\boldsymbol{p}, \boldsymbol{p})\,,$$

  where the **entropy** $S(\boldsymbol{p}, \boldsymbol{p})$ is the unavoidable part of the loss (due to the need to predict the realisation of a random variable) and $d(\hat{\boldsymbol{p}}, \boldsymbol{p})$ the extra loss.

Proper scoring rules

- The expectation of a strictly proper score can be decomposed as follows:

$$\mathbb{E}_{Y\sim\boldsymbol{p}}\, S(\hat{\boldsymbol{p}}, Y) = S(\hat{\boldsymbol{p}}, \boldsymbol{p}) = d(\hat{\boldsymbol{p}}, \boldsymbol{p}) + S(\boldsymbol{p}, \boldsymbol{p})\,,$$

  where the **entropy** $S(\boldsymbol{p}, \boldsymbol{p})$ is the unavoidable part of the loss (due to the need to predict the realisation of a random variable) and $d(\hat{\boldsymbol{p}}, \boldsymbol{p})$ the extra loss.

- Another decomposition is

$$\mathbb{E}\, S(\hat{\boldsymbol{p}}, Y) = \mathbb{E}\, d(\hat{\boldsymbol{p}}, \boldsymbol{c}) + \mathbb{E}\, e(\boldsymbol{c})\,,$$

  where the expectation is taken over $\mathcal{X} \times \mathcal{Y}$, and $c_k := p(y = k \mid \hat{\boldsymbol{p}})$, i.e., $\boldsymbol{c}$ is the true class distribution on those instances that receive the same prediction $\hat{\boldsymbol{p}}$.

Proper scoring rules

- The expectation of a strictly proper score can be decomposed as follows:

$$\mathbb{E}_{Y\sim\boldsymbol{p}}\, S(\hat{\boldsymbol{p}}, Y) = S(\hat{\boldsymbol{p}}, \boldsymbol{p}) = d(\hat{\boldsymbol{p}}, \boldsymbol{p}) + S(\boldsymbol{p}, \boldsymbol{p})\,,$$

  where the **entropy** $S(\boldsymbol{p}, \boldsymbol{p})$ is the unavoidable part of the loss (due to the need to predict the realisation of a random variable) and $d(\hat{\boldsymbol{p}}, \boldsymbol{p})$ the extra loss.

- Another decomposition is

$$\mathbb{E}\, S(\hat{\boldsymbol{p}}, Y) = \mathbb{E}\, d(\hat{\boldsymbol{p}}, \boldsymbol{c}) + \mathbb{E}\, e(\boldsymbol{c})\,,$$

  where the expectation is taken over $\mathcal{X} \times \mathcal{Y}$, and $c_k := p(y = k \mid \hat{\boldsymbol{p}})$, i.e., $\boldsymbol{c}$ is the true class distribution on those instances that receive the same prediction $\hat{\boldsymbol{p}}$.

- $\mathbb{E}\, d(\hat{\boldsymbol{p}}, \boldsymbol{c})$ is called **calibration loss** and $\mathbb{E}\, e(\boldsymbol{c})$ **refinement loss**.

Calibration

Calibration

- Many ML methods naturally yield predictions in the form of **scores**

$$s_1, \ldots, s_K \in \mathbb{S} \subseteq \mathbb{R},$$

for the K classes, but these scores (e.g., the distance from a linear hyperplane) are not probabilities.

Calibration

- Many ML methods naturally yield predictions in the form of **scores**

$$s_1, \ldots, s_K \in \mathbb{S} \subseteq \mathbb{R},$$

for the K classes, but these scores (e.g., the distance from a linear hyperplane) are not probabilities.

- Other methods do produce **probability estimates**

$$\boldsymbol{s} = h(\boldsymbol{x}) = (s_1, \ldots, s_K) \in [0, 1]^K,$$

but again, these may not match the true probabilities — they might rather be **pseudo-probabilities** that are not well "calibrated".

Calibration

- Many ML methods naturally yield predictions in the form of **scores**

$$s_1, \ldots, s_K \in \mathbb{S} \subseteq \mathbb{R},$$

  for the K classes, but these scores (e.g., the distance from a linear hyperplane) are not probabilities.

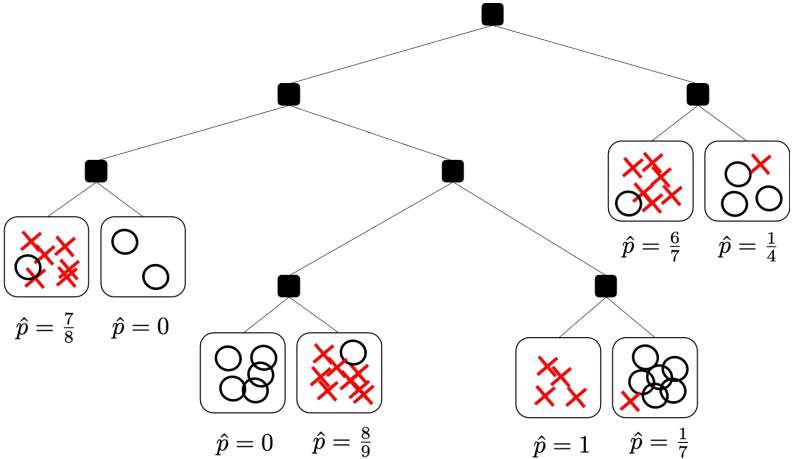
- Other methods do produce **probability estimates**

$$\boldsymbol{s} = h(\boldsymbol{x}) = (s_1, \ldots, s_K) \in [0, 1]^K,$$

  but again, these may not match the true probabilities — they might rather be **pseudo-probabilities** that are not well "calibrated".

- Yet, calibration is a prerequisite for uncertainty-awareness and important for prediction, decision-making, cost-sensitive classification, etc.

Probability estimation with decision trees



Calibration

Calibration

- Consider the **binary case** ($K = 2$) with probabilistic predictions

$$\hat{\boldsymbol{p}} = (\hat{p}_-, \hat{p}_+) = (1 - \hat{p}_+, \hat{p}_+) \in \Delta_2 \,,$$

  where $\hat{p}_+$ denotes the (predicted) probability for the positive class.

Calibration

- Consider the **binary case** ($K = 2$) with probabilistic predictions

$$\hat{\boldsymbol{p}} = (\hat{p}_-, \hat{p}_+) = (1 - \hat{p}_+, \hat{p}_+) \in \Delta_2 \,,$$

  where $\hat{p}_+$ denotes the (predicted) probability for the positive class.
- We say that a probabilistic predictor is **calibrated** if, for all $\alpha \in [0, 1]$,

$$P\big(y = +1 \,|\, \hat{p}_+ = \alpha\big) = \alpha \,.$$

Calibration

- Consider the **binary case** ($K = 2$) with probabilistic predictions

$$\hat{\boldsymbol{p}} = (\hat{p}_-, \hat{p}_+) = (1 - \hat{p}_+, \hat{p}_+) \in \Delta_2 \,,$$

  where $\hat{p}_+$ denotes the (predicted) probability for the positive class.
- We say that a probabilistic predictor is **calibrated** if, for all $\alpha \in [0, 1]$,

$$P\big(y = +1 \,|\, \hat{p}_+ = \alpha\big) = \alpha \,.$$

- Broadly speaking, averaged over all instances $\boldsymbol{x}$ for which the learner predicts $\hat{p}_+ = h(\boldsymbol{x}) = \alpha$, the fraction of positives is indeed α.

Calibration

- Consider the **binary case** ($K = 2$) with probabilistic predictions

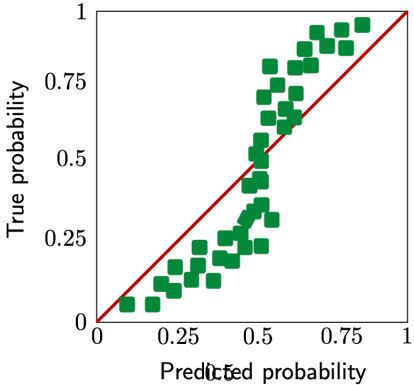
$$\hat{\boldsymbol{p}} = (\hat{p}_-, \hat{p}_+) = (1 - \hat{p}_+, \hat{p}_+) \in \Delta_2 \,,$$

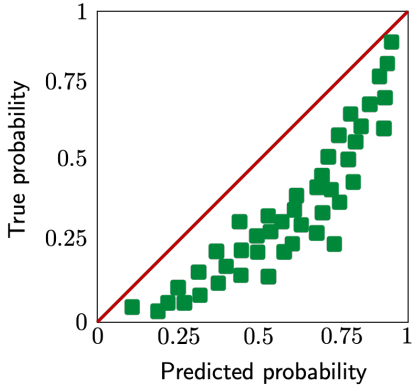
  where $\hat{p}_+$ denotes the (predicted) probability for the positive class.
- We say that a probabilistic predictor is **calibrated** if, for all $\alpha \in [0, 1]$,

$$P\big(y = +1 \,|\, \hat{p}_+ = \alpha\big) = \alpha \,.$$

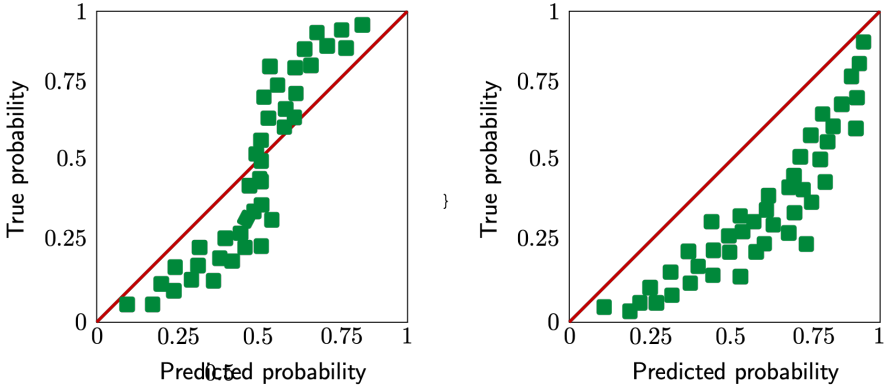
- Broadly speaking, averaged over all instances $\boldsymbol{x}$ for which the learner predicts $\hat{p}_+ = h(\boldsymbol{x}) = \alpha$, the fraction of positives is indeed α.
- In other words, a predicted probability vector is supposed to match empirical frequencies, at least in the long run.

Calibration



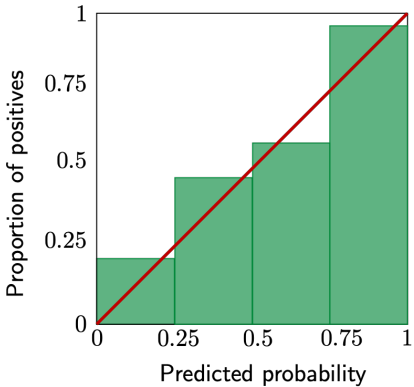


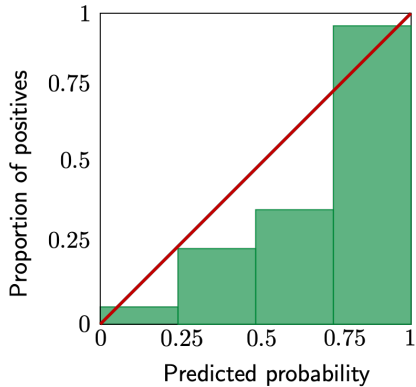
Calibration



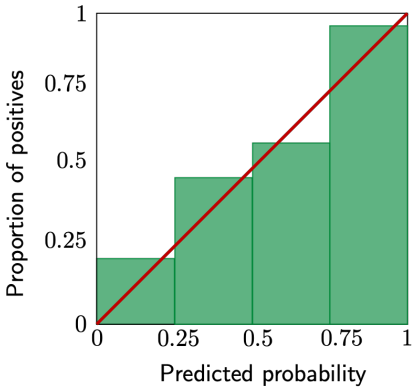
- Examples of miscalibration: bias toward 1/2 (left), systematic underestimation (right).

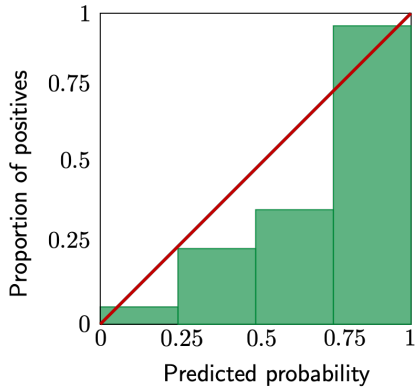
Reliability diagram





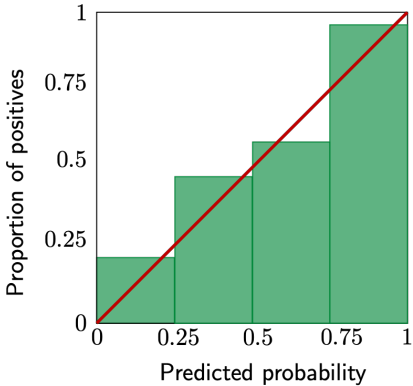
Reliability diagram

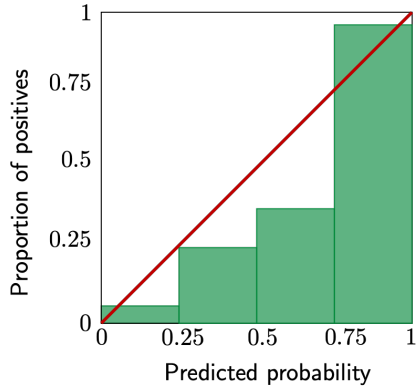




- As ground-truth probabilities are not observed, binning is needed in practice.

Reliability diagram





- As ground-truth probabilities are not observed, binning is needed in practice.
- There is a trade-off in the choice of the width of the bins (more data per bin vs. more fine-granular assessment).

Calibration and scaling functions

Calibration and scaling functions

- Different **post-processing** (post-hoc) methods have been proposed for the purpose of calibration, i.e., to construct a **calibration function**

$$C : \mathbb{S} \longrightarrow [0, 1] \,,$$

such that $\hat{p}_+ = C(s)$ is a well-calibrated probability estimate for instances $\boldsymbol{x}$ assigned score $s = h(\boldsymbol{x})$.

Calibration and scaling functions

- Different **post-processing** (post-hoc) methods have been proposed for the purpose of calibration, i.e., to construct a **calibration function**

$$C : \mathbb{S} \longrightarrow [0, 1] \,,$$

  such that $\hat{p}_+ = C(s)$ is a well-calibrated probability estimate for instances $\boldsymbol{x}$ assigned score $s = h(\boldsymbol{x})$.
- NB: To make the distinction between (post-hoc) calibration and scaling, C is often called **scaling function** in the case where $\mathbb{S} \neq [0, 1]$.

Calibration and scaling functions

- Different **post-processing** (post-hoc) methods have been proposed for the purpose of calibration, i.e., to construct a **calibration function**

$$C : \mathbb{S} \longrightarrow [0, 1] \,,$$

  such that $\hat{p}_+ = C(s)$ is a well-calibrated probability estimate for instances $\boldsymbol{x}$ assigned score $s = h(\boldsymbol{x})$.
- NB: To make the distinction between (post-hoc) calibration and scaling, C is often called **scaling function** in the case where $\mathbb{S} \neq [0, 1]$.
- For learning C, a set of **calibration data** is used:

$$\mathcal{D}_{cal} = \big\{(s_1, y_1), \ldots, (s_N, y_N)\big\} \subset \mathbb{S} \times \{0, 1\}$$

Calibration and scaling functions

- Different **post-processing** (post-hoc) methods have been proposed for the purpose of calibration, i.e., to construct a **calibration function**

$$C : \mathbb{S} \longrightarrow [0, 1] \,,$$

  such that $\hat{p}_+ = C(s)$ is a well-calibrated probability estimate for instances $\boldsymbol{x}$ assigned score $s = h(\boldsymbol{x})$.
- NB: To make the distinction between (post-hoc) calibration and scaling, C is often called **scaling function** in the case where $\mathbb{S} \neq [0, 1]$.
- For learning C, a set of **calibration data** is used:

$$\mathcal{D}_{cal} = \big\{(s_1, y_1), \ldots, (s_N, y_N)\big\} \subset \mathbb{S} \times \{0, 1\}$$

- This data should be different from the training data (used to learn the scoring classifier h). Otherwise, there is a risk of introducing a bias.

Empirical binning

Empirical binning

- **Binning** offers a first obvious approach: Partition $\mathbb{S}$ into bins (intervals) $B_1, \ldots, B_M$, and define $C(s) = \hat{p}_{J(s)}$, where $J(s)$ denotes the index of the bin of s (i.e., $s \in B_{J(s)}$).

Empirical binning

- **Binning** offers a first obvious approach: Partition $\mathbb{S}$ into bins (intervals) $B_1, \ldots, B_M$, and define $C(s) = \hat{p}_{J(s)}$, where $J(s)$ denotes the index of the bin of s (i.e., $s \in B_{J(s)}$).
- $\hat{p}_1, \ldots, \hat{p}_M$ are chosen so as to minimise the **estimated calibration error** (ECE)

$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{N} |\hat{p}_m - \bar{p}_m| \,,$$

where

$$\bar{p}_m = \frac{\sum_{n=1}^{N} [\![s_n \in B_m]\!] \, [\![y_n = +1]\!]}{\sum_{n=1}^{N} [\![s_n \in B_m]\!]}$$

is the average proportion of positives in bin B_m.

Empirical binning

- **Binning** offers a first obvious approach: Partition $\mathbb{S}$ into bins (intervals) $B_1, \ldots, B_M$, and define $C(s) = \hat{p}_{J(s)}$, where $J(s)$ denotes the index of the bin of s (i.e., $s \in B_{J(s)}$).
- $\hat{p}_1, \ldots, \hat{p}_M$ are chosen so as to minimise the **estimated calibration error** (ECE)

$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{N} |\hat{p}_m - \bar{p}_m| \,,$$

where

$$\bar{p}_m = \frac{\sum_{n=1}^{N} [\![s_n \in B_m]\!] \, [\![y_n = +1]\!]}{\sum_{n=1}^{N} [\![s_n \in B_m]\!]}$$

is the average proportion of positives in bin B_m.

- Binning is nonparametric and hence flexible, easy to train, and can directly minimise calibration error, albeit at the cost of (increased) groping loss.

Platt scaling

Platt scaling

- Another method is **Platt scaling**, which essentially applies logistic regression to predicted scores $s \in \mathbb{R}$, i.e., it fits a calibration function C such that

$$C_{\alpha,\beta}(s) = \frac{1}{1 + \exp(-\alpha \cdot s - \beta)},$$

  minimising log-loss on $\mathcal{D}_{cal}$ (including regularisation, or cross-validated training).

Platt scaling

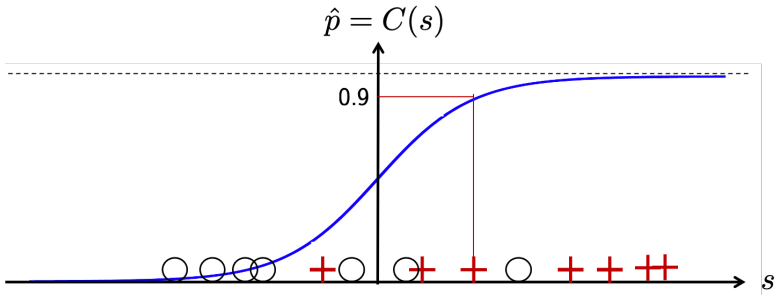
- Another method is **Platt scaling**, which essentially applies logistic regression to predicted scores $s \in \mathbb{R}$, i.e., it fits a calibration function C such that

$$C_{\alpha,\beta}(s) = \frac{1}{1 + \exp(-\alpha \cdot s - \beta)} ,$$

  minimising log-loss on $\mathcal{D}_{cal}$ (including regularisation, or cross-validated training).

- Platt scaling is fast and easy to implement, but restricted to **sigmoidal calibration functions** (pushing scores from the center toward the extremes, hence coming with a risk of over-confidence).

Platt scaling



Beta calibration

Beta calibration

- **Beta calibration** is specifically designed for probabilistic classifiers and fits a function

$$C_{\alpha,\beta,\gamma}(s) = \frac{1}{1 + \exp(-\alpha \cdot \log(s) - \beta \log(1-s) - \gamma)},$$

  again minimising log-loss on $\mathcal{D}_{cal}$.

Beta calibration

- **Beta calibration** is specifically designed for probabilistic classifiers and fits a function

$$C_{\alpha,\beta,\gamma}(s) = \frac{1}{1 + \exp(-\alpha \cdot \log(s) - \beta \log(1-s) - \gamma)},$$

  again minimising log-loss on $\mathcal{D}_{cal}$.

- Although still restricted in a parametric way, Beta calibration is more flexible than Platt scaling and includes inverse sigmoids and the identity map (which helps prevent over-calibration and unnecessary adjustments).

Isotonic regression

Isotonic regression

- **Isotonic regression** combines the nonparametric character of binning with Platt scaling's guarantee of monotonicity.

Isotonic regression

- **Isotonic regression** combines the nonparametric character of binning with Platt scaling's guarantee of monotonicity.
- Isotonic regression minimises

$$\sum_{i=n}^{N} w_n \left(C(s_n) - y_n \right)^2$$

subject to the constraint that C is isotonic: $C(s) \leq C(t)$ for $s < t$.

Isotonic regression

- **Isotonic regression** combines the nonparametric character of binning with Platt scaling's guarantee of monotonicity.
- Isotonic regression minimises

$$\sum_{i=n}^{N} w_n \left(C(s_n) - y_n \right)^2$$

  subject to the constraint that C is isotonic: $C(s) \leq C(t)$ for $s < t$.
- Note that C is evaluated only at a finite number of points; in-between, one may (linearly) interpolate or assume a piecewise constant function.

Isotonic regression

- **Isotonic regression** combines the nonparametric character of binning with Platt scaling's guarantee of monotonicity.
- Isotonic regression minimises

$$\sum_{i=n}^{N} w_n \left(C(s_n) - y_n \right)^2$$

  subject to the constraint that C is isotonic: $C(s) \leq C(t)$ for $s < t$.
- Note that C is evaluated only at a finite number of points; in-between, one may (linearly) interpolate or assume a piecewise constant function.
- Isotonic regression is more expensive in terms of training time and memory consumption.

Pair-adjacent violators algorithm (PAVA)

Pair-adjacent violators algorithm (PAVA)

- Let the scores observed for calibration be sorted (and without ties), such that

$$s_1 < s_2 < \ldots < s_N \,.$$

We then seek values $c_1 \leq c_2 \leq \ldots \leq c_N$ which minimize

$$\sum_{n=1}^{N} w_n (c_n - y_n)^2 \,.$$

Pair-adjacent violators algorithm (PAVA)

- Let the scores observed for calibration be sorted (and without ties), such that

$$s_1 < s_2 < \ldots < s_N \,.$$

  We then seek values $c_1 \leq c_2 \leq \ldots \leq c_N$ which minimize

$$\sum_{n=1}^{N} w_n (c_n - y_n)^2 \,.$$

- **Initialise** one block B_n for each observation (s_n, y_n); the value of the block is $c(B_n) = y_n$ and the width is $w(B_n) = 1$.

Pair-adjacent violators algorithm (PAVA)

- Let the scores observed for calibration be sorted (and without ties), such that

$$s_1 < s_2 < \ldots < s_N \,.$$

  We then seek values $c_1 \leq c_2 \leq \ldots \leq c_N$ which minimize

$$\sum_{n=1}^{N} w_n (c_n - y_n)^2 \,.$$

- **Initialise** one block B_n for each observation (s_n, y_n); the value of the block is $c(B_n) = y_n$ and the width is $w(B_n) = 1$.
- A **merge operation** combines two blocks B' and B'' into a new block B with width $w(B) = w(B') + w(B'')$ and value

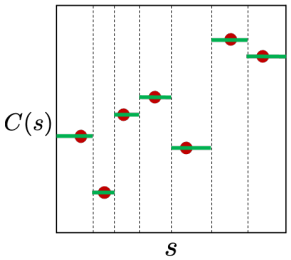
$$c = \frac{w(B')c(B') + w(B'')c(B'')}{w(B') + w(B'')} \,.$$

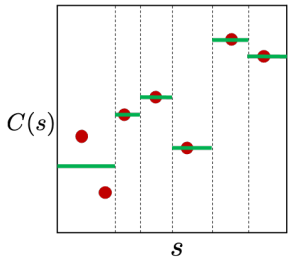
Pair-adjacent violators algorithm (PAVA)

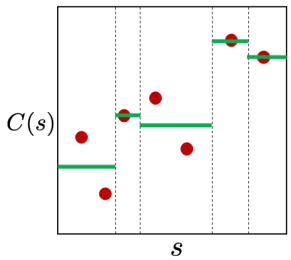
Pair-adjacent violators algorithm (PAVA)

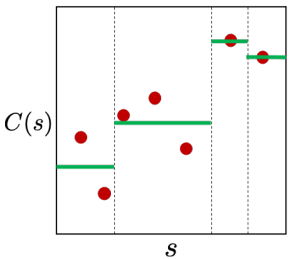
- PAVA iterates the following steps (the description is somewhat simplified to avoid notational overload):
  1. Find the first violating pair, namely, adjacent blocks B_i and B_{i+1} such that $c_i > c_{i+1}$; if there is no such pair, then stop.
  2. Merge B_i and B_{i+1} into a new block B.
  3. If $c(B) < c(B_{i-1})$ for the left neighbor block B_{i-1}, merge also these blocks and continue doing so until no more violations are encountered.
  4. Continue with (1).

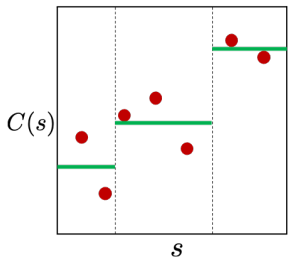
Pair-adjacent violators algorithm (PAVA)







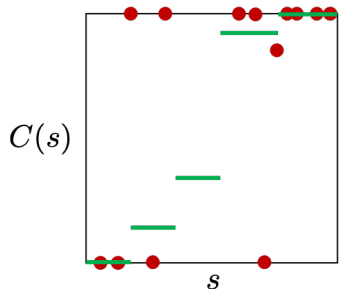




Pair-adjacent violators algorithm (PAVA)

Pair-adjacent violators algorithm (PAVA)

- Note that, in the case of binary classification, the target values are all in $\{0, 1\}$:



Multi-class calibration

Multi-class calibration

- Calibration methods also exist for the **multi-class case** (i.e., classification problems with more than two classes).

Multi-class calibration

- Calibration methods also exist for the **multi-class case** (i.e., classification problems with more than two classes).
- Then, however, the problem becomes conceptually more difficult (and is still a topic of ongoing research).

Multi-class calibration

- Calibration methods also exist for the **multi-class case** (i.e., classification problems with more than two classes).
- Then, however, the problem becomes conceptually more difficult (and is still a topic of ongoing research).
- Some concepts do not immediately generalise, for example isotonic regression (which assumes a ranking on scores, and rankings are inherently bipartite).

Multi-class calibration

- Calibration methods also exist for the **multi-class case** (i.e., classification problems with more than two classes).
- Then, however, the problem becomes conceptually more difficult (and is still a topic of ongoing research).
- Some concepts do not immediately generalise, for example isotonic regression (which assumes a ranking on scores, and rankings are inherently bipartite).
- While essentially coinciding for binary classification, the following definitions of calibration (leading to increasingly difficult problems) can be distinguished for more than two classes:
  - **Confidence calibration:** Calibration of the highest predicted probability
  - **Class-wise calibration:** Calibration of the marginal probabilities
  - **Multi-class calibration:** Calibration of the entire vector of predicted probabilities

Agenda

1. Introduction
2. Training probabilistic predictors
3. Calibration
4. **Set-valued (conformal) prediction**
5. Epistemic uncertainty

Conformal prediction

Conformal prediction

- **Conformal prediction** (Balasubramanian *et al.*, 2014) is a framework for reliable prediction that is rooted in classical frequentist statistics and **hypothesis testing**.

Conformal prediction

- **Conformal prediction** (Balasubramanian *et al.*, 2014) is a framework for reliable prediction that is rooted in classical frequentist statistics and **hypothesis testing**.
- Instead of point predictions, CP makes **set-valued predictions** covering the true outcome with high probability.

Conformal prediction

- **Conformal prediction** (Balasubramanian *et al.*, 2014) is a framework for reliable prediction that is rooted in classical frequentist statistics and **hypothesis testing**.
- Instead of point predictions, CP makes **set-valued predictions** covering the true outcome with high probability.

$\longrightarrow \qquad P\big(\, y \in \{2, 3, 9\} \,\big) \geq 0.9$

Conformal prediction

Conformal prediction

- Given a sequence of training observations

$$(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_N, y_N), (\boldsymbol{x}_{N+1}, \bullet)$$

and a new query $\boldsymbol{x}_{N+1}$ with unknown outcome y_{N+1}, $\bullet$ is hypothetically replaced by each candidate, i.e., the hypothesis $y_{N+1} = y$ is tested for all $y \in \mathcal{Y}$:

$$(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_N, y_N), (\boldsymbol{x}_{N+1}, y)$$

Conformal prediction

- Given a sequence of training observations

$$(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_N, y_N), (\boldsymbol{x}_{N+1}, \bullet)$$

and a new query $\boldsymbol{x}_{N+1}$ with unknown outcome y_{N+1}, $\bullet$ is hypothetically replaced by each candidate, i.e., the hypothesis $y_{N+1} = y$ is tested for all $y \in \mathcal{Y}$:

$$(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_N, y_N), (\boldsymbol{x}_{N+1}, y)$$

- Only those outcomes y for which this **hypothesis can be rejected** at a predefined level of confidence are excluded, while those for which the hypothesis cannot be rejected are collected to form the prediction set or **prediction region** $Y \subseteq \mathcal{Y}$.

Nonconformity function

Nonconformity function

- In conformal prediction, the "strangeness" of a pattern $(\boldsymbol{x}_{N+1}, y)$ is captured in terms of a **nonconformity score**.

Nonconformity function

- In conformal prediction, the "strangeness" of a pattern $(\boldsymbol{x}_{N+1}, y)$ is captured in terms of a **nonconformity score**.
- Moreover, hypothesis testing is done in a **nonparametric way**.

Nonconformity function

- In conformal prediction, the "strangeness" of a pattern $(\boldsymbol{x}_{N+1}, y)$ is captured in terms of a **nonconformity score**.
- Moreover, hypothesis testing is done in a **nonparametric way**.
- Consider any **nonconformity function**

$$f : \mathcal{X} \times \mathcal{Y} \longrightarrow \mathbb{R}$$

that assigns scores $\alpha = f(\boldsymbol{x}, y)$ to input/output tuples.

Nonconformity function

- In conformal prediction, the "strangeness" of a pattern $(\boldsymbol{x}_{N+1}, y)$ is captured in terms of a **nonconformity score**.
- Moreover, hypothesis testing is done in a **nonparametric way**.
- Consider any **nonconformity function**

$$f : \mathcal{X} \times \mathcal{Y} \longrightarrow \mathbb{R}$$

  that assigns scores $\alpha = f(\boldsymbol{x}, y)$ to input/output tuples.
- The higher the score, the more "strange" the pattern $(\boldsymbol{x}, y)$, i.e., the less the data point $(\boldsymbol{x}, y)$ conforms to what one would expect to observe.

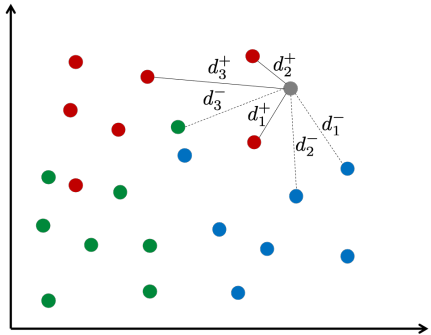
Nonconformity function

Nonconformity function

- Example of a nonconformity score based on nearest neighbors:

$$f(\boldsymbol{x}, y) = \frac{\sum_{i=1}^{k} d_i^+}{\sum_{i=1}^{k} d_i^-},$$

where d_i^+ is the distance from the i^{th} nearest neighbor labeled y, and d_i^- the distance from the i^{th} nearest neighbor labeled differently.



Conformal prediction

Conformal prediction

- Applying this function to the sequence of observations, with a specific (though hypothetical) choice of $y = y_{N+1}$, yields a sequence of scores

$$\alpha_1, \alpha_2, \ldots, \alpha_N, \alpha_{N+1},$$

where $\alpha_i = f(\boldsymbol{x}_i, y_i)$.

Conformal prediction

- Applying this function to the sequence of observations, with a specific (though hypothetical) choice of $y = y_{N+1}$, yields a sequence of scores

$$\alpha_1, \alpha_2, \ldots, \alpha_N, \alpha_{N+1},$$

  where $\alpha_i = f(\boldsymbol{x}_i, y_i)$.

- Denote by σ the permutation of $\{1, \ldots, N+1\}$ that sorts the scores in increasing order, i.e., such that

$$\alpha_{\sigma(1)} \leq \ldots \leq \alpha_{\sigma(N+1)}.$$

Conformal prediction

- Applying this function to the sequence of observations, with a specific (though hypothetical) choice of $y = y_{N+1}$, yields a sequence of scores

$$\alpha_1, \alpha_2, \ldots, \alpha_N, \alpha_{N+1},$$

  where $\alpha_i = f(\boldsymbol{x}_i, y_i)$.
- Denote by σ the permutation of $\{1, \ldots, N+1\}$ that sorts the scores in increasing order, i.e., such that

$$\alpha_{\sigma(1)} \leq \ldots \leq \alpha_{\sigma(N+1)}.$$

- Under the assumption that the hypothetical choice of y_{N+1} is in agreement with the true data-generating process, and that this process has the property of **exchangeability**, every permutation σ has the same probability of occurrence.

Conformal prediction

Conformal prediction

- Consequently, the probability that α_{N+1} is among the $\epsilon\,\%$ highest nonconformity scores should be low.

Conformal prediction

- Consequently, the probability that α_{N+1} is among the $\epsilon\,\%$ highest nonconformity scores should be low.
- This notion can be captured by the p-values associated with the candidate y, defined as

$$p(y) := \frac{\#\{i \mid \alpha_i \geq \alpha_{N+1}\}}{N+1}$$

Conformal prediction

- Consequently, the probability that α_{N+1} is among the $\epsilon\,\%$ highest nonconformity scores should be low.
- This notion can be captured by the p-values associated with the candidate y, defined as

$$p(y) := \frac{\#\{i \,|\, \alpha_i \geq \alpha_{N+1}\}}{N+1}$$

- According to what we said, the probability that $p(y) < \epsilon$ (i.e., α_{N+1} is among the $\epsilon\,\%$ highest α-values) is upper-bounded by ϵ.

Conformal prediction

- Consequently, the probability that α_{N+1} is among the $\epsilon\,\%$ highest nonconformity scores should be low.
- This notion can be captured by the p-values associated with the candidate y, defined as

$$p(y) := \frac{\#\{i \,|\, \alpha_i \geq \alpha_{N+1}\}}{N+1}$$

- According to what we said, the probability that $p(y) < \epsilon$ (i.e., α_{N+1} is among the $\epsilon\,\%$ highest α-values) is upper-bounded by ϵ.
- Thus, the hypothesis $y_{N+1} = y$ can be rejected for those candidates y for which $p(y) < \epsilon$.

Conformal prediction

Conformal prediction

- By construction, the set-valued prediction $Y = Y(\boldsymbol{x}_{n+1})$ is guaranteed to cover the true outcome y_{N+1} with a pre-specified probability of $1 - \epsilon$ (for example 95 %).

Conformal prediction

- By construction, the set-valued prediction $Y = Y(\boldsymbol{x}_{n+1})$ is guaranteed to cover the true outcome y_{N+1} with a pre-specified probability of $1 - \epsilon$ (for example 95 %).
- The error bounds are valid by construction, regardless of the nonconformity function.

Conformal prediction

- By construction, the set-valued prediction $Y = Y(\boldsymbol{x}_{n+1})$ is guaranteed to cover the true outcome y_{N+1} with a pre-specified probability of $1 - \epsilon$ (for example 95 %).
- The error bounds are valid by construction, regardless of the nonconformity function.
- However, the choice of this function has an important influence on the **efficiency** of conformal prediction, that is, the **size of prediction regions**: The more suitably the nonconformity function f is chosen, the smaller these sets will be.

Remarks

Remarks

- The above validity property is also called **marginal coverage**. The randomisation is over the entire data generation and prediction procedure; thus, the coverage of $1 - \epsilon$ is neither guaranteed
  - for a fixed sequence on previous data (coverage can be higher or lower),
  - conditioned on the query $\boldsymbol{x}_q$.

Remarks

- The above validity property is also called **marginal coverage**. The randomisation is over the entire data generation and prediction procedure; thus, the coverage of $1 - \epsilon$ is neither guaranteed
  - for a fixed sequence on previous data (coverage can be higher or lower),
  - conditioned on the query $\boldsymbol{x}_q$.
- The above is a **transductive** version of CP, but meanwhile, there are also **inductive** versions (making use of a single training and a single calibration data set).

Remarks

- The above validity property is also called **marginal coverage**. The randomisation is over the entire data generation and prediction procedure; thus, the coverage of $1 - \epsilon$ is neither guaranteed
  - for a fixed sequence on previous data (coverage can be higher or lower),
  - conditioned on the query $\boldsymbol{x}_q$.
- The above is a **transductive** version of CP, but meanwhile, there are also **inductive** versions (making use of a single training and a single calibration data set).
- There are various other extensions of CP, also for **conditional coverage**. Besides, instead of controlling coverage, there are variants for controlling more general notions of **risk** (Angelopoulos *et al.*, 2021).

Remarks

- The above validity property is also called **marginal coverage**. The randomisation is over the entire data generation and prediction procedure; thus, the coverage of $1 - \epsilon$ is neither guaranteed
  - for a fixed sequence on previous data (coverage can be higher or lower),
  - conditioned on the query $\boldsymbol{x}_q$.
- The above is a **transductive** version of CP, but meanwhile, there are also **inductive** versions (making use of a single training and a single calibration data set).
- There are various other extensions of CP, also for **conditional coverage**. Besides, instead of controlling coverage, there are variants for controlling more general notions of **risk** (Angelopoulos *et al.*, 2021).
- Uncertainty quantification with conformal prediction is
  - agnostic to the underlying model,
  - agnostic to the underlying data distribution (i.e., distribution-free),
  - valid for the finite sample case (not only asymptotically).

Agenda

1. Introduction
2. Training probabilistic predictors
3. Calibration
4. Set-valued (conformal) prediction
5. **Epistemic uncertainty**

Aleatoric versus epistemic uncertainty

Aleatoric versus epistemic uncertainty

- **Aleatoric** (aka statistical) uncertainty refers to the notion of **randomness**, that is, the variability in the outcome of an experiment which is due to inherently random effects.

Aleatoric versus epistemic uncertainty

- **Aleatoric** (aka statistical) uncertainty refers to the notion of **randomness**, that is, the variability in the outcome of an experiment which is due to inherently random effects.
- **Epistemic** (aka systematic) uncertainty refers to uncertainty caused by a **lack of knowledge**, i.e., to the epistemic state of the agent.

Aleatoric versus epistemic uncertainty

- **Aleatoric** (aka statistical) uncertainty refers to the notion of **randomness**, that is, the variability in the outcome of an experiment which is due to inherently random effects.
- **Epistemic** (aka systematic) uncertainty refers to uncertainty caused by a **lack of knowledge**, i.e., to the epistemic state of the agent.
- As opposed to aleatoric uncertainty, epistemic uncertainty can in principle be reduced on the basis of additional information.

Aleatoric versus epistemic uncertainty

H

$1/2$

T

$1/2$

H

$1/2$

T

$1/2$

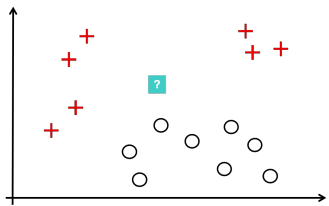
“Not knowing the chance of mutually exclusive events and knowing the chance to be equal are two quite different states of knowledge"

Ronald Fisher (1890-1962)

Aleatoric versus epistemic uncertainty in ML

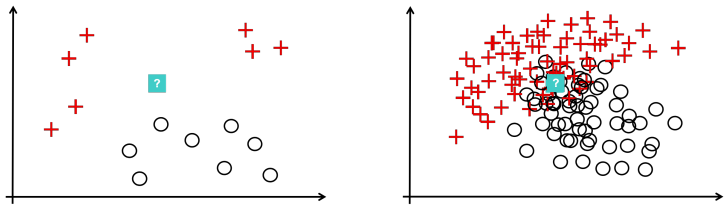
Aleatoric versus epistemic uncertainty in ML

- Both types of uncertainty also play an important role in ML, where the learner's state of knowledge strongly depends on the amount of data seen so far ...



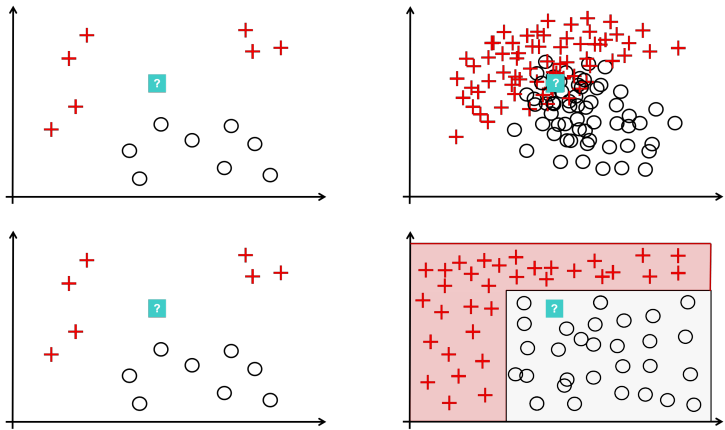
Aleatoric versus epistemic uncertainty in ML

- Both types of uncertainty also play an important role in ML, where the learner's state of knowledge strongly depends on the amount of data seen so far ...



Aleatoric versus epistemic uncertainty in ML

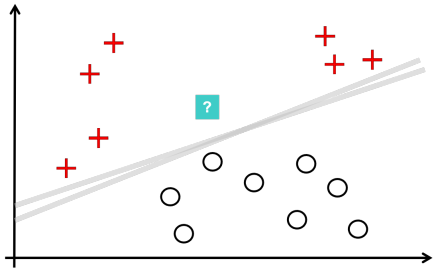
- Both types of uncertainty also play an important role in ML, where the learner's state of knowledge strongly depends on the amount of data seen so far ...



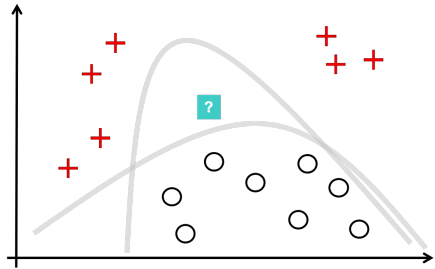
Aleatoric versus epistemic uncertainty in ML

Aleatoric versus epistemic uncertainty in ML

- ... but also on the underlying model assumptions:



strong prior



weaker prior

Uncertainty-aware learning

Uncertainty-aware learning

- **Uncertainty representation:** How should the learner represent its (model, predictive) uncertainty, i.e., which mathematical formalisms should be used?

Uncertainty-aware learning

- **Uncertainty representation:** How should the learner represent its (model, predictive) uncertainty, i.e., which mathematical formalisms should be used?
- **Learning and inference:** How to make the learner accomplish the task? How to make sure that the uncertainty representation is accurate (and why to trust it more than the actual prediction)?

Uncertainty-aware learning

- **Uncertainty representation:** How should the learner represent its (model, predictive) uncertainty, i.e., which mathematical formalisms should be used?
- **Learning and inference:** How to make the learner accomplish the task? How to make sure that the uncertainty representation is accurate (and why to trust it more than the actual prediction)?
- **Uncertainty quantification:** How to quantify the learner's uncertainty in terms of numbers? How to measure and disentangle the different types of uncertainty (aleatoric, epistemic, total)?

Aleatoric versus epistemic uncertainty in ML

Aleatoric versus epistemic uncertainty in ML

- The distinction between aleatoric and epistemic uncertainty can be very difficult: Is the data-generating process completely random or only very complicated?

Aleatoric versus epistemic uncertainty in ML

- The distinction between aleatoric and epistemic uncertainty can be very difficult: Is the data-generating process completely random or only very complicated?
- **Predict the next number**: 116, 304, 194, 341, 224, 654, 609, 625, 533, 91, 205, 35, 527, 611, 128, 235, 348, 912, 582, 52, 672, 20, 856, 904, 628, 273, 615, 105, 610, 862, 384, 705, 73, 794, 775, 156, **??**

Aleatoric versus epistemic uncertainty in ML

- The distinction between aleatoric and epistemic uncertainty can be very difficult: Is the data-generating process completely random or only very complicated?
- **Predict the next number**: 116, 304, 194, 341, 224, 654, 609, 625, 533, 91, 205, 35, 527, 611, 128, 235, 348, 912, 582, 52, 672, 20, 856, 904, 628, 273, 615, 105, 610, 862, 384, 705, 73, 794, 775, 156, **??**

$$x \leftarrow x \times 237 \bmod 971$$

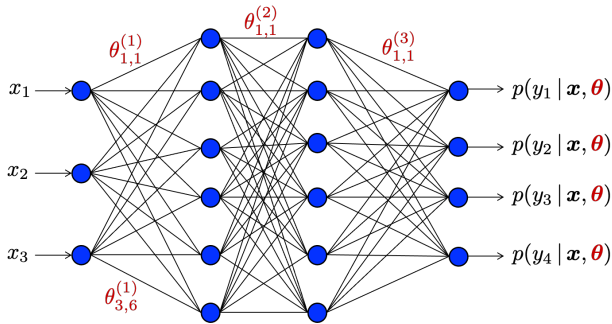
Uncertainty quantification

Uncertainty quantification

- In the case of neural networks, where hypotheses $h = h_{\boldsymbol{\theta}}$ are identified by network weights $\boldsymbol{\theta}$, epistemic uncertainty essentially corresponds to uncertainty about these weights.

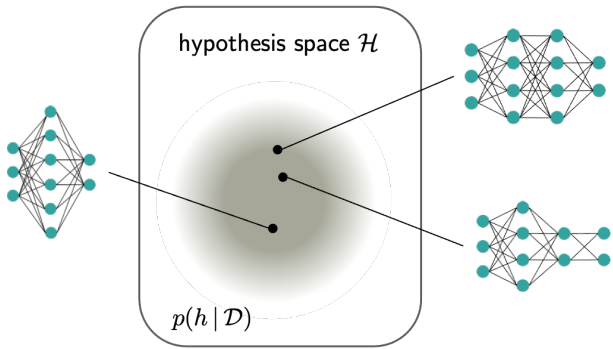
Uncertainty quantification

- In the case of neural networks, where hypotheses $h = h_{\boldsymbol{\theta}}$ are identified by network weights $\boldsymbol{\theta}$, epistemic uncertainty essentially corresponds to uncertainty about these weights.
- Fixed weights $\boldsymbol{\theta}$ lead to a fixed probability $p(\cdot \mid \boldsymbol{x}, \boldsymbol{\theta})$.



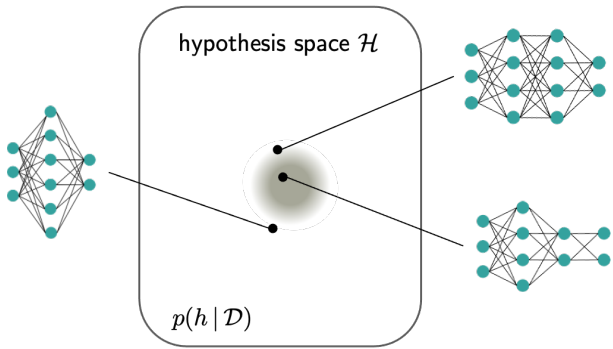
The Bayesian approach

- A Bayesian learner maintains a probability distribution over the hypothesis space (probabilistic predictors).
- The less concentrated that distribution, the higher the learner's epistemic uncertainty.

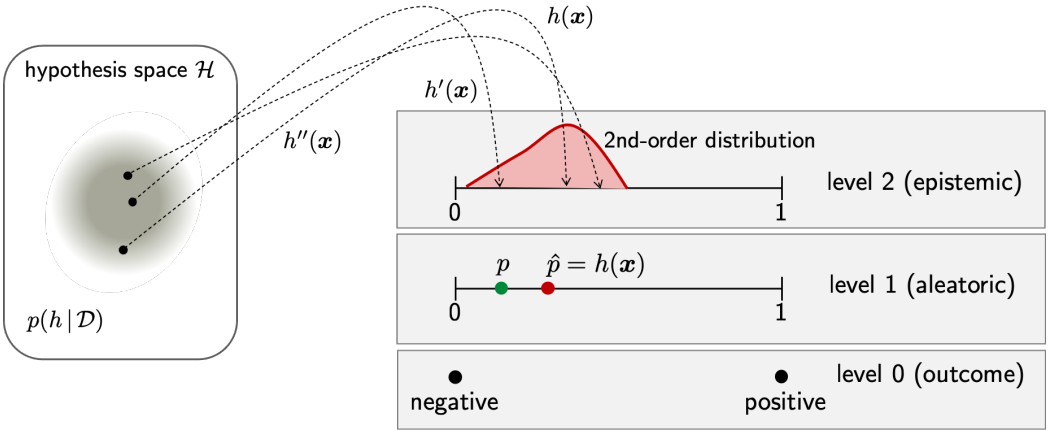


The Bayesian approach

- A Bayesian learner maintains a probability distribution over the hypothesis space (probabilistic predictors).
- The less concentrated that distribution, the higher the learner's epistemic uncertainty.



Posterior predictive distribution



Uncertainty quantification

Uncertainty quantification

- How to measure uncertainty, i.e., quantify the amount of uncertainty contained in a prediction?

Uncertainty quantification

- How to measure uncertainty, i.e., quantify the amount of uncertainty contained in a prediction?
- A well-known uncertainty measure is the **Shannon entropy**, which, in the case of discrete probability $p : \mathcal{Y} \longrightarrow [0, 1]$, is given by

$$H\big[\, Y \,\big] = H\big[\, p \,\big] = -\sum_{y \in \mathcal{Y}} p(y) \log_2 p(y) \,.$$

Uncertainty quantification

- How to measure uncertainty, i.e., quantify the amount of uncertainty contained in a prediction?
- A well-known uncertainty measure is the **Shannon entropy**, which, in the case of discrete probability $p : \mathcal{Y} \longrightarrow [0, 1]$, is given by

$$H\big[\, Y \,\big] = H\big[\, p \,\big] = -\sum_{y \in \mathcal{Y}} p(y) \log_2 p(y) \,.$$

- What we seek is a **decomposition**

$$\underbrace{\mathrm{TU}(\boldsymbol{x})}_{\text{total uncertainty}} = \underbrace{\mathrm{AU}(\boldsymbol{x})}_{\text{aleatoric uncertainty}} + \underbrace{\mathrm{EU}(\boldsymbol{x})}_{\text{epistemic uncertainty}}$$

Uncertainty quantification

Uncertainty quantification

- One idea is to model **epistemic uncertainty** as **mutual information** between outcomes and hypotheses (Depeweg *et al.*, 2018):

$$\underbrace{H[Y]}_{\text{total uncertainty}} = \underbrace{I(Y;\Theta)}_{\text{epistemic}} + \underbrace{H[Y \mid \Theta]}_{\text{aleatoric}}$$

Uncertainty quantification

- One idea is to model **epistemic uncertainty** as **mutual information** between outcomes and hypotheses (Depeweg *et al.*, 2018):

$$\underbrace{H[Y]}_{\text{total uncertainty}} = \underbrace{I(Y;\Theta)}_{\text{epistemic}} + \underbrace{H[Y \mid \Theta]}_{\text{aleatoric}}$$

- Intuitively, epistemic uncertainty thus captures the amount of information about the model parameters $\boldsymbol{\theta}$ that would be gained through knowledge of the true outcome y.

Uncertainty quantification

- One idea is to model **epistemic uncertainty** as **mutual information** between outcomes and hypotheses (Depeweg *et al.*, 2018):

$$\underbrace{H[Y]}_{\text{total uncertainty}} = \underbrace{I(Y;\Theta)}_{\text{epistemic}} + \underbrace{H[Y \mid \Theta]}_{\text{aleatoric}}$$

- Intuitively, epistemic uncertainty thus captures the amount of information about the model parameters $\boldsymbol{\theta}$ that would be gained through knowledge of the true outcome y.
- **Total uncertainty** = entropy of the predictive posterior distribution, in the case of discrete $\mathcal{Y}$ given by

$$\mathrm{TU}(\boldsymbol{x}) = H\big[\, p(y \mid \boldsymbol{x}) \,\big] = -\sum_{y \in \mathcal{Y}} p(y \mid \boldsymbol{x}) \log_2 p(y \mid \boldsymbol{x}) \,.$$

Uncertainty quantification

Uncertainty quantification

- This uncertainty also includes the (epistemic) uncertainty about the network weights $\boldsymbol{\theta}$, but fixing a set of weights, i.e., considering a distribution $p(y \mid \boldsymbol{x}, \boldsymbol{\theta})$, removes the epistemic uncertainty.

Uncertainty quantification

- This uncertainty also includes the (epistemic) uncertainty about the network weights $\boldsymbol{\theta}$, but fixing a set of weights, i.e., considering a distribution $p(y \mid \boldsymbol{x}, \boldsymbol{\theta})$, removes the epistemic uncertainty.
- Therefore, the expectation over the entropies of these distributions,

$$\mathbb{E}_{p(\boldsymbol{\theta} \mid \mathcal{D})} H\big[p(y \mid \boldsymbol{x}, \boldsymbol{\theta})\big] = \\ = -\int p(\boldsymbol{\theta} \mid \mathcal{D}) \left(\sum_{y \in \mathcal{Y}} p(y \mid \boldsymbol{x}, \boldsymbol{\theta}) \log_2 p(y \mid \boldsymbol{x}, \boldsymbol{\theta}) \right) d\,\boldsymbol{\theta} \ ,$$

  is a measure of the **aleatoric uncertainty** (conditional entropy).

Uncertainty quantification

- This uncertainty also includes the (epistemic) uncertainty about the network weights $\boldsymbol{\theta}$, but fixing a set of weights, i.e., considering a distribution $p(y \mid \boldsymbol{x}, \boldsymbol{\theta})$, removes the epistemic uncertainty.
- Therefore, the expectation over the entropies of these distributions,

$$\mathbb{E}_{p(\boldsymbol{\theta} \mid \mathcal{D})} H\big[p(y \mid \boldsymbol{x}, \boldsymbol{\theta})\big] = \\ = -\int p(\boldsymbol{\theta} \mid \mathcal{D}) \left(\sum_{y \in \mathcal{Y}} p(y \mid \boldsymbol{x}, \boldsymbol{\theta}) \log_2 p(y \mid \boldsymbol{x}, \boldsymbol{\theta}) \right) d\,\boldsymbol{\theta} \ ,$$

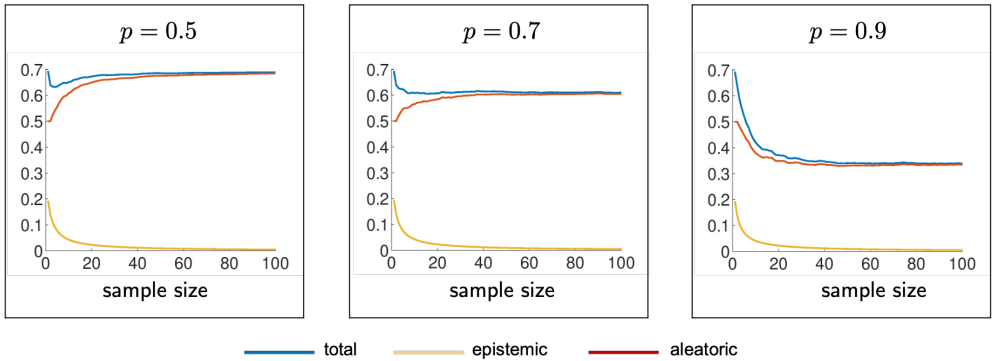
  is a measure of the **aleatoric uncertainty** (conditional entropy).
- Finally, the **epistemic uncertainty** is obtained as the difference

$$\mathrm{EU}(\boldsymbol{x}) := H\big[p(y \mid \boldsymbol{x})\big] - \mathbb{E}_{p(\boldsymbol{\theta} \mid \mathcal{D})} H\big[p(y \mid \boldsymbol{x}, \boldsymbol{\theta})\big] \ ,$$

  which equals the **mutual information** between y and $\boldsymbol{\theta}$.

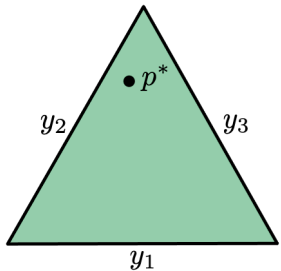
Example: coin flipping

- Tossing a coin with bias p, task is to predict the next outcome, hypothesis space equipped with Dirichlet distribution



Remarks

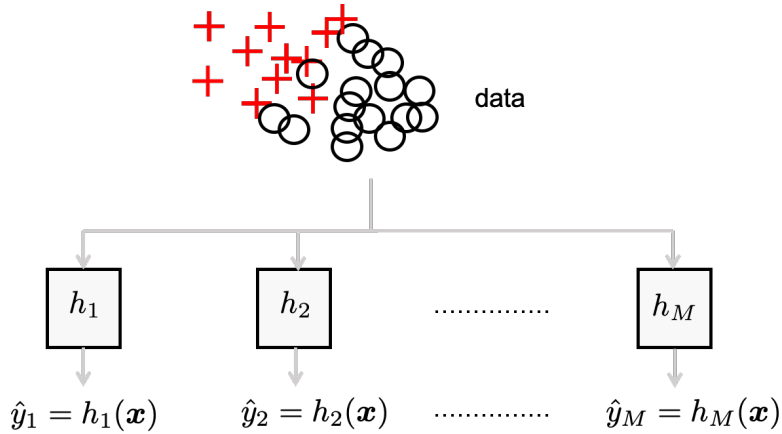
- Is a uniform distribution on $\Delta = \mathbb{P}(\mathcal{Y})$ (set of all distributions p on $\mathcal{Y}$) an adequate representation of **complete ignorance** (full epistemic uncertainty)?
- **Averaging** the (conditional) entropies over all p is meaningful only if all p are indeed **known** to be equally likely (aleatoric uncertainty is always $1/2$).
- But this is certainly not the case, as only one p^* can be the ground truth.



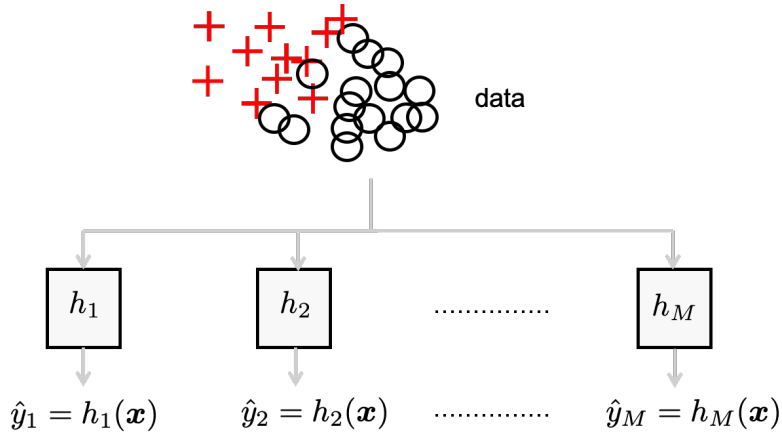
Remarks

- One may also question the **additive decomposition** $\mathrm{TU} = \mathrm{AU} + \mathrm{EU}$ itself.
- In the beginning, total uncertainty should be full ($\mathrm{TU} = 1$), and so should epistemic uncertainty ($\mathrm{EU} = 1$) — but this implies $\mathrm{AU} = 0$.
- This suggests a role of AU as a **lower bound** on (the true) aleatoric uncertainty.
- Indeed, epistemic uncertainty partially comprises aleatoric uncertainty (high EU implies high uncertainty about AU, showing interaction between both).

Ensemble methods for uncertainty quantification

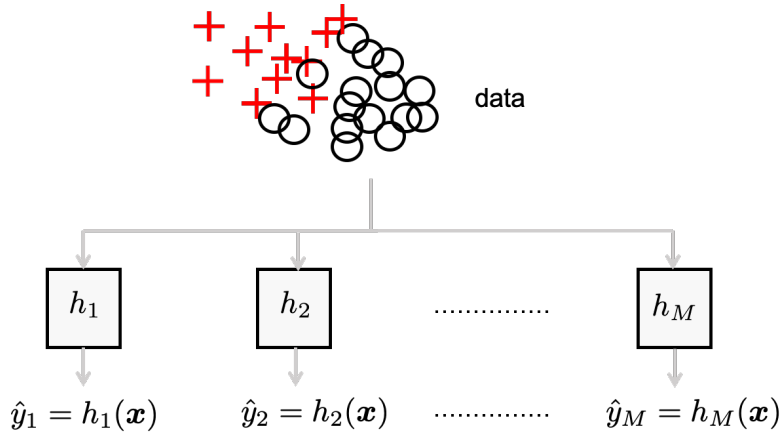


Ensemble methods for uncertainty quantification



- Ensemble can be seen as an approximation of a distribution.

Ensemble methods for uncertainty quantification



- Ensemble can be seen as an approximation of a distribution.
- Intuitively, diversity is an indicator of epistemic uncertainty.

Bayesian agents: Ensemble-based approximation

Bayesian agents: Ensemble-based approximation

- Based on an ensemble of hypotheses $h_1, \ldots, h_M$, producing respective predictions $p_1, \ldots, p_M$, an approximation of conditional entropy can be obtained by

$$\mathrm{AU}(\boldsymbol{x}) \coloneqq -\frac{1}{M} \sum_{i=1}^{M} \sum_{y \in \mathcal{Y}} p_i(y \mid \boldsymbol{x}) \log_2 p_i(y \mid \boldsymbol{x}) \,,$$

  an approximation of total uncertainty (Shannon entropy) by

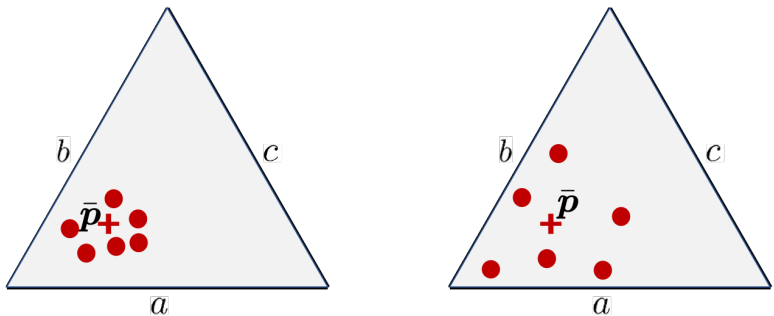
$$\mathrm{TU}(\boldsymbol{x}) \coloneqq -\sum_{y \in \mathcal{Y}} \underbrace{\left(\frac{1}{M} \sum_{i=1}^{M} p_i(y \mid \boldsymbol{x}) \right)}_{\bar{p}(y \mid \boldsymbol{x})} \log_2 \underbrace{\left(\frac{1}{M} \sum_{i=1}^{M} p_i(y \mid \boldsymbol{x}) \right)}_{\bar{p}(y \mid \boldsymbol{x})} ,$$

  and an approximation of epistemic uncertainty (mutual information) by the difference.

Bayesian agents: Ensemble-based approximation

Bayesian agents: Ensemble-based approximation

- Epistemic uncertainty thus defined is equivalent to **Jensen-Shannon divergence** of the distributions $p_i(y \mid \boldsymbol{x})$, $i = 1, \ldots, M$



low divergence

high divergence

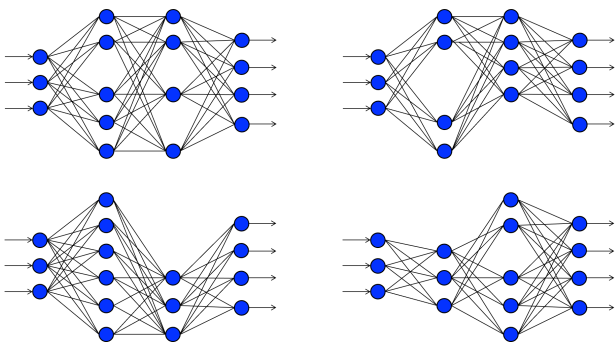
Bayesian agents: Ensemble-based approximation

Bayesian agents: Ensemble-based approximation

- For neural networks, it has been shown that techniques such as **Dropout** (Gal and Ghahramani, 2016) and **DropConnect** (Mobiny *et al.*, 2021) can be interpreted as (implicit) ensemble methods, and can hence be used to implement this approach.

Bayesian agents: Ensemble-based approximation

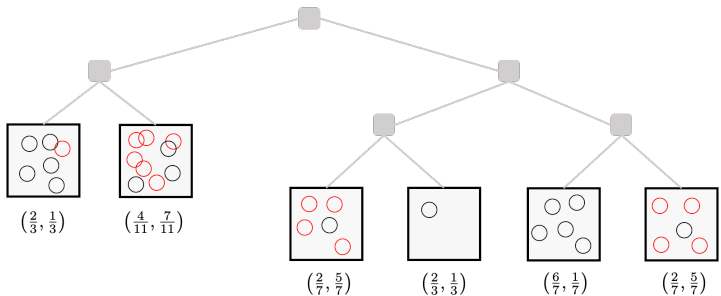
- For neural networks, it has been shown that techniques such as **Dropout** (Gal and Ghahramani, 2016) and **DropConnect** (Mobiny *et al.*, 2021) can be interpreted as (implicit) ensemble methods, and can hence be used to implement this approach.
- Of course, any other ensemble technique could be used as well.



Bayesian agents: Ensemble-based approximation

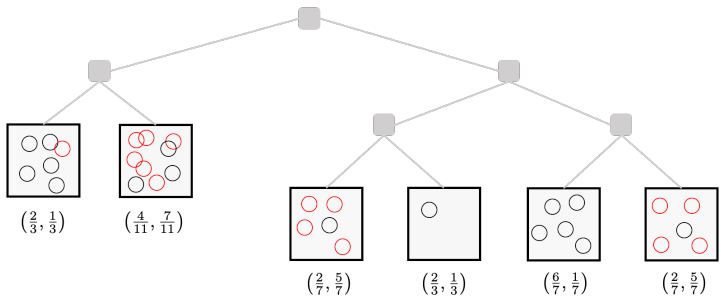
Bayesian agents: Ensemble-based approximation

- We proposed an implementation based on **Random Forests**, using decision trees that predict probabilities in terms of (Laplace-corrected) relative frequencies (Shaker and Hüllermeier, 2020).



Bayesian agents: Ensemble-based approximation

- We proposed an implementation based on **Random Forests**, using decision trees that predict probabilities in terms of (Laplace-corrected) relative frequencies (Shaker and Hüllermeier, 2020).



- Empirically, there are no significant performance differences between neural networks and random forests.

Direct (epistemic) uncertainty prediction

- Consider a (level-1) loss for **probabilistic predictions**:

$$\ell_1 : \mathbb{P}(\mathcal{Y}) \times \mathcal{Y} \longrightarrow \mathbb{R}$$

- Recall that ERM yields good (unbiased) predictors if ℓ_1 is a (strictly) **proper scoring rule**, which incentivises the learner to predict the true $p(y \mid \boldsymbol{x})$.
- **Question**: Can we do the same on the **epistemic level**, i.e., training a predictor

$$h : \mathcal{X} \longrightarrow \mathbb{P}(\mathbb{P}(\mathcal{Y}))$$

by minimising a **level-2 loss**

$$\ell_2 : \mathbb{P}(\mathbb{P}(\mathcal{Y})) \times \mathcal{Y} \longrightarrow \mathbb{R}\,,$$

such that the predictor represents its epistemic uncertainty in a faithful way?

The Dirichlet distribution

- A **Dirichlet distribution** $\mathrm{Dir}(\alpha)$ is specified by means of $K \geq 2$ positive real-valued parameters, i.e., a vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K) \in \mathbb{R}_+^K$.

The Dirichlet distribution

- A **Dirichlet distribution** $\mathrm{Dir}(\alpha)$ is specified by means of $K \geq 2$ positive real-valued parameters, i.e., a vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K) \in \mathbb{R}_+^K$.
- The probability density function is defined on the K simplex

$$\Delta_K = \left\{ \boldsymbol{\theta} = (\theta_1, \ldots, \theta_K)^\top \,|\, \theta_1, \ldots, \theta_K \geq 0, \sum_{k=1}^{K} \theta_k = 1 \right\}$$

and given as follows:

$$p(\boldsymbol{\theta} \,|\, \boldsymbol{\alpha}) = p(\theta_1, \ldots, \theta_K \,|\, \boldsymbol{\alpha}) = \frac{1}{\mathbb{B}(\boldsymbol{\alpha})} \prod_{k=1}^{K} \theta_k^{\alpha_k - 1},$$

where the normalisation constant is the multivariate Beta function.

The Dirichlet distribution

- A **Dirichlet distribution** $\mathrm{Dir}(\alpha)$ is specified by means of $K \geq 2$ positive real-valued parameters, i.e., a vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K) \in \mathbb{R}_+^K$.
- The probability density function is defined on the K simplex

$$\Delta_K = \left\{ \boldsymbol{\theta} = (\theta_1, \ldots, \theta_K)^\top \,|\, \theta_1, \ldots, \theta_K \geq 0, \sum_{k=1}^{K} \theta_k = 1 \right\}$$

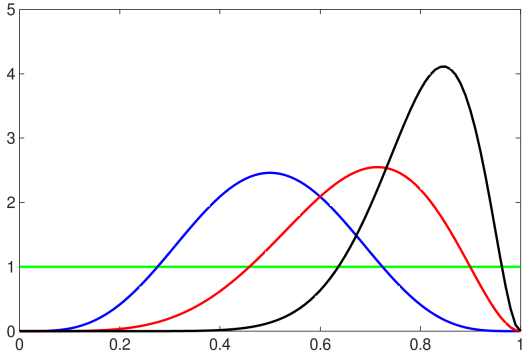
  and given as follows:

$$p(\boldsymbol{\theta} \,|\, \boldsymbol{\alpha}) = p(\theta_1, \ldots, \theta_K \,|\, \boldsymbol{\alpha}) = \frac{1}{\mathbb{B}(\boldsymbol{\alpha})} \prod_{k=1}^{K} \theta_k^{\alpha_k - 1},$$

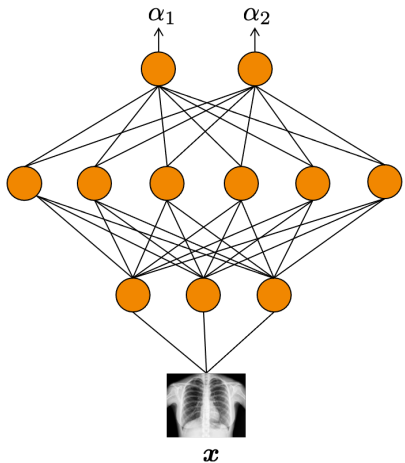
  where the normalisation constant is the multivariate Beta function.
- In Bayesian statistics, the Dirichlet distribution is commonly used as the conjugate prior of the **multinomial distribution**.

The Dirichlet distribution

- Dirichlet distribution with parameters $\boldsymbol{\alpha} = (\alpha_1, \alpha_2) = (1, 1)$, $(5, 5)$, $(3, 6)$, $(3, 12)$.



Predicting a Dirichlet distribution



Direct epistemic uncertainty prediction

- Several authors have proposed the minimisation of an empirical loss of the form

$$L = \frac{1}{N} \sum\nolimits_{n=1}^{N} \ell_2\big(Q^{(n)}, y^{(n)}\big) = \frac{1}{N} \sum\nolimits_{n=1}^{N} \mathbb{E}_{\boldsymbol{\theta} \sim Q}\, \ell_1 \left(\boldsymbol{\theta}, y^{(n)}\right) ,$$

  where $Q^{(n)}$ is the level-2 prediction for the instance $\boldsymbol{x}^{(n)}$.

Direct epistemic uncertainty prediction

- Several authors have proposed the minimisation of an empirical loss of the form

$$L = \frac{1}{N} \sum_{n=1}^{N} \ell_2\big(Q^{(n)}, y^{(n)}\big) = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{\boldsymbol{\theta} \sim Q}\, \ell_1 \left(\boldsymbol{\theta}, y^{(n)}\right) ,$$

  where $Q^{(n)}$ is the level-2 prediction for the instance $\boldsymbol{x}^{(n)}$.

- Thus, an individual prediction Q is penalised in terms of the **expected** level-1 loss, with the expectation taken over the realisations of $\boldsymbol{\theta}$.

Direct epistemic uncertainty prediction

- Several authors have proposed the minimisation of an empirical loss of the form

$$L = \frac{1}{N} \sum_{n=1}^{N} \ell_2\big(Q^{(n)}, y^{(n)}\big) = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{\boldsymbol{\theta} \sim Q}\, \ell_1 \left(\boldsymbol{\theta}, y^{(n)}\right) ,$$

  where $Q^{(n)}$ is the level-2 prediction for the instance $\boldsymbol{x}^{(n)}$.

- Thus, an individual prediction Q is penalised in terms of the **expected** level-1 loss, with the expectation taken over the realisations of $\boldsymbol{\theta}$.

- Examples of level-1 loss include cross entropy (Charpentier *et al.*, 2020) and Brier score (Sensoy *et al.*, 2018).

Direct epistemic uncertainty prediction

- Several authors have proposed the minimisation of an empirical loss of the form

$$L = \frac{1}{N}\sum_{n=1}^{N} \ell_2\big(Q^{(n)}, y^{(n)}\big) = \frac{1}{N}\sum_{n=1}^{N} \mathbb{E}_{\boldsymbol{\theta}\sim Q}\, \ell_1\left(\boldsymbol{\theta}, y^{(n)}\right) ,$$

  where $Q^{(n)}$ is the level-2 prediction for the instance $\boldsymbol{x}^{(n)}$.

- Thus, an individual prediction Q is penalised in terms of the **expected** level-1 loss, with the expectation taken over the realisations of $\boldsymbol{\theta}$.

- Examples of level-1 loss include cross entropy (Charpentier *et al.*, 2020) and Brier score (Sensoy *et al.*, 2018).

- Besides, a regularised version has been proposed:

$$L = \frac{1}{N}\sum_{n=1}^{N} \underbrace{\mathbb{E}_{\boldsymbol{\theta}\sim Q}\, \ell_1\left(\boldsymbol{\theta}, y\right) + \lambda\, d_{KL}\left(Q^{(n)}, Q_0\right)}_{\ell_2\left(Q^{(n)}, y^{(n)}\right)}$$

A negative result

- Informally, we define a level-2 loss function ℓ_2 as **appropriate** if the following holds for the empirical loss minimiser

$$Q^{(N)} = \arg\min_{Q} \frac{1}{N} \sum_{n=1}^{N} \ell_2 \left(Q, y^{(n)} \right)$$

on any i.i.d. observational data sequence $y^{(1)}, y^{(2)}, \ldots$ with $y^{(i)} \sim \boldsymbol{\theta}^*$:

(A1) The learner's uncertainty gradually decreases (in expectation) with increasing sample size N, in terms of a suitable uncertainty measure U.

(A2) In the limit $N \to \infty$, all epistemic uncertainty disappears and $Q^{(N)} \to \delta_{\boldsymbol{\theta}^*}$.

A negative result

- Informally, we define a level-2 loss function ℓ_2 as **appropriate** if the following holds for the empirical loss minimiser

$$Q^{(N)} = \arg\min_{Q} \frac{1}{N} \sum_{n=1}^{N} \ell_2 \left(Q, y^{(n)} \right)$$

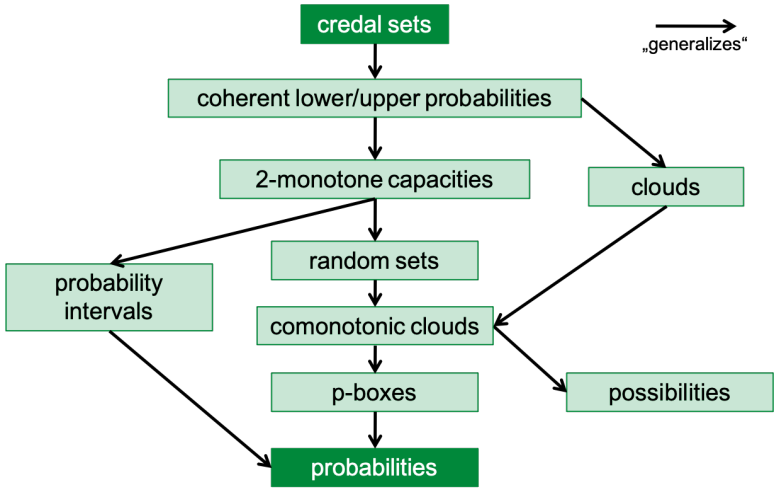
on any i.i.d. observational data sequence $y^{(1)}, y^{(2)}, \ldots$ with $y^{(i)} \sim \boldsymbol{\theta}^*$:

(A1) The learner's uncertainty gradually decreases (in expectation) with increasing sample size N, in terms of a suitable uncertainty measure U.

(A2) In the limit $N \to \infty$, all epistemic uncertainty disappears and $Q^{(N)} \to \delta_{\boldsymbol{\theta}^*}$.

- Bengs *et al.* (2022) formally prove that a loss minimisation approach using a level-2 loss as specified above does not lead to an appropriate level-2 loss.

Generalised uncertainty calculi



Credal uncertainty representation

Probabilistic learner	Bayesian learner	Credal learner
$h : \mathcal{X} \longrightarrow \mathbb{P}(\mathcal{Y})$	$h : \mathcal{X} \longrightarrow \mathbb{P}(\mathbb{P}(\mathcal{Y}))$	$h : \mathcal{X} \longrightarrow 2^{\mathbb{P}(\mathcal{Y})}$

ENSEMBLE-BASED UNCERTAINTY QUANTIFICATION:
BAYESIAN VERSUS CREDAL INFERENCE

A PREPRINT

Mohammad Hossein Shaker
Department of Computer Science
Paderborn University
Paderborn, Germany
mhshaker@mail.upb.de

Eyke Hüllermeier
Institute of Informatics
University of Munich (LMU)
Munich, Germany
eyke@lmu.de

December 13, 2021

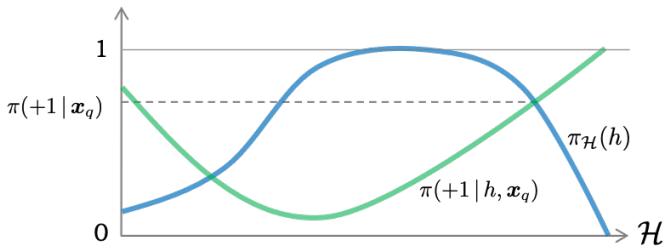
Possibilistic uncertainty representation

Possibilistic uncertainty representation

- Senge *et al.* (2014) formalise the following principle in a mathematical way: Given a query $\boldsymbol{x}_q$, an outcome $y \in \mathcal{Y}$ is a **plausible** candidate if there exists a **plausible** hypothesis $h \in \mathcal{H}$ (compatible with the data) such that y is **strongly supported** by h (in the sense that $p(y \mid \boldsymbol{x}_q, h)$ is high).

Possibilistic uncertainty representation

- Senge *et al.* (2014) formalise the following principle in a mathematical way: Given a query $\boldsymbol{x}_q$, an outcome $y \in \mathcal{Y}$ is a **plausible** candidate if there exists a **plausible** hypothesis $h \in \mathcal{H}$ (compatible with the data) such that y is **strongly supported** by h (in the sense that $p(y \mid \boldsymbol{x}_q, h)$ is high).
- Thus, they induce a graded (fuzzy) set of plausible candidate hypotheses instead of a single one, and consider the plausible outcomes under these candidates.



Summary and Outlook

Summary and Outlook

- **Uncertainty** is of major importance in ML and attracting more and more attention, also due to practical applications.

Summary and Outlook

- **Uncertainty** is of major importance in ML and attracting more and more attention, also due to practical applications.
- We highlighted the benefits of distinguishing between different types of uncertainty, notably **aleatoric and epistemic uncertainty**.

Summary and Outlook

- **Uncertainty** is of major importance in ML and attracting more and more attention, also due to practical applications.
- We highlighted the benefits of distinguishing between different types of uncertainty, notably **aleatoric and epistemic uncertainty**.
- Currently, **uncertainty quantification** for ML is developing very dynamically.

Summary and Outlook

- **Uncertainty** is of major importance in ML and attracting more and more attention, also due to practical applications.
- We highlighted the benefits of distinguishing between different types of uncertainty, notably **aleatoric and epistemic uncertainty**.
- Currently, **uncertainty quantification** for ML is developing very dynamically.
- Most approaches so far neglect **model uncertainty**, assuming instead that the model is correctly specified, altough **model misspecification** is a common problem in practice.

Summary and Outlook

- **Uncertainty** is of major importance in ML and attracting more and more attention, also due to practical applications.
- We highlighted the benefits of distinguishing between different types of uncertainty, notably **aleatoric and epistemic uncertainty**.
- Currently, **uncertainty quantification** for ML is developing very dynamically.
- Most approaches so far neglect **model uncertainty**, assuming instead that the model is correctly specified, altough **model misspecification** is a common problem in practice.
- Related to this is the **"closed world" assumption**, which is often violated in practice, e.g., in the case of OOD data.

Summary and Outlook

- **Uncertainty** is of major importance in ML and attracting more and more attention, also due to practical applications.
- We highlighted the benefits of distinguishing between different types of uncertainty, notably **aleatoric and epistemic uncertainty**.
- Currently, **uncertainty quantification** for ML is developing very dynamically.
- Most approaches so far neglect **model uncertainty**, assuming instead that the model is correctly specified, altough **model misspecification** is a common problem in practice.
- Related to this is the **"closed world" assumption**, which is often violated in practice, e.g., in the case of OOD data.
- Usefulness of **generalized uncertainty calculi**?

More on this topic ...

Open Access | Published: 08 March 2021

Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods

Eyke Hüllermeier ✉ & Willem Waegeman

Machine Learning **110**, 457–506 (2021) | Cite this article

23k Accesses | **73** Citations | **2** Altmetric | Metrics

Abstract

The notion of uncertainty is of major importance in machine learning and constitutes a key element of machine learning methodology. In line with the statistical tradition, uncertainty has long been perceived as almost synonymous with standard probability and probabilistic predictions. Yet, due to the steadily increasing relevance of machine learning for practical applications and related issues such as safety requirements, new problems and challenges have recently been identified by machine learning scholars, and these problems may call for new methodological developments. In particular, this includes the importance of distinguishing between (at least) two different types of uncertainty, often referred to as *aleatoric* and *epistemic*. In this paper, we provide an introduction to the topic of uncertainty in machine learning as well as an overview of attempts so far at handling uncertainty in general and formalizing this distinction in particular.

References

A.N. Angelopoulos, S. Bates, E.J. Candès, M.I. Jordan, and L. Lei. Learn then test: Calibrating predictive algorithms to achieve risk control. *CoRR*, abs/2110.01052, 2021.

V. Balasubramanian, S.S. Ho, and V. Vovk, editors. *Conformal Prediction for Reliable Machine Learning: Theory, Adaptations and Applications*. Morgan Kaufmann, 2014.

V. Bengs, E. Hüllermeier, and W. Waegeman. On the difficulty of epistemic uncertainty quantification in machine learning: The case of direct uncertainty estimation through loss minimisation. *arXiv:2203.06102*, 2022.

B. Charpentier, D. Zügner, and S. Günnemann. Posterior network: Uncertainty estimation without OOD samples via density-based pseudo-counts. In *Proc. NeurIPS, Neural Information Processing Systems*, 2020.

S. Depeweg, J.M. Hernandez-Lobato, F. Doshi-Velez, and S. Udluft. Decomposition of uncertainty in Bayesian deep learning for efficient and risk-sensitive learning. In *Proc. ICML, 35th Int. Conf. on Machine Learning*, Stockholm, Sweden, 2018.

Y. Gal and Z. Ghahramani. Bayesian convolutional neural networks with Bernoulli approximate variational inference. In *Proc. of the ICLR Workshop Track*, 2016.

A. Mobiny, H.V. Nguyen, S. Moulik, N. Garg, and C.C. Wu. DropConnect is effective in modeling uncertainty of Bayesian networks. *Scientific Reports*, 11(5458), 2021.

M. Sato, J. Suzuki, H. Shindo, and Y. Matsumoto. Interpretable adversarial perturbation in input embedding space for text. In *Proc. IJCAI*, pages 4323–4330, Stockholm, Sweden, 2018.

R. Senge, S. Bösner, K. Dembczynski, J. Haasenritter, O. Hirsch, N. Donner-Banzhoff, and E. Hüllermeier. Reliable classification: Learning classifiers that distinguish aleatoric and epistemic uncertainty. *Information Sciences*, 255:16–29, 2014.

M. Sensoy, L. Kaplan, and M. Kandemir. Evidential deep learning to quantify classification uncertainty. In *Proc. NeurIPS, 32nd Conf. on Neural Information Processing Systems*, Montreal, Canada, 2018.

M.H. Shaker and E. Hüllermeier. Aleatoric and epistemic uncertainty with random forests. In *Proc. IDA, 18th Int. Symposium on Intelligent Data Analysis*, pages 444–456, Konstanz, Germany, 2020. Springer.

Epistemic uncertainty sampling

Open Access | Published: 18 June 2021

How to measure uncertainty in uncertainty sampling for active learning

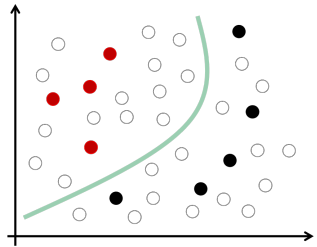
Vu-Linh Nguyen, Mohammad Hossein Shaker & Eyke Hüllermeier

Machine Learning (2021) | Cite this article

1033 Accesses | **1** Altmetric | Metrics

Abstract

Various strategies for active learning have been proposed in the machine learning literature. In uncertainty sampling, which is among the most popular approaches, the active learner sequentially queries the label of those instances for which its current prediction is maximally uncertain. The predictions as well as the measures used to quantify the degree of uncertainty, such as entropy, are traditionally of a probabilistic nature. Yet, alternative approaches to capturing uncertainty in machine learning, alongside with corresponding uncertainty measures, have been proposed in recent years. In particular, some of these measures seek to distinguish different sources and to separate different types of uncertainty, such as the reducible (epistemic) and the irreducible (aleatoric) part of the total uncertainty in a prediction. The goal of this paper is to elaborate on the usefulness of such measures for uncertainty sampling, and to compare their performance in active learning. To this end, we instantiate uncertainty sampling with different measures, analyze the properties of the sampling strategies thus obtained, and compare them in an experimental study.



Sampling in epistemically uncertain regions of the instance space is potentially more useful than sampling in aleatorically uncertain regions ...

Excursus: Hypothesis testing

Excursus: Hypothesis testing

- The goal in hypothesis testing is to reject a **null hypothesis** H_0 as being unlikely in light of the data, and hence to provide evidence in favor of the **alternative hypothesis** $H_1 = \neg H_0$.

Excursus: Hypothesis testing

- The goal in hypothesis testing is to reject a **null hypothesis** H_0 as being unlikely in light of the data, and hence to provide evidence in favor of the **alternative hypothesis** $H_1 = \neg H_0$.
- To this end, a suitable test statistic $T = T(\mathcal{D})$ is defined, so that the distribution of T is known under H_0.

Excursus: Hypothesis testing

- The goal in hypothesis testing is to reject a **null hypothesis** H_0 as being unlikely in light of the data, and hence to provide evidence in favor of the **alternative hypothesis** $H_1 = \neg H_0$.
- To this end, a suitable test statistic $T = T(\mathcal{D})$ is defined, so that the distribution of T is known under H_0.
- Suppose the distribution is concentrated in a "normal" range, so that T takes values in $A = [t_l, t_u] \subset \mathbb{R}$ with high probability $1 - \delta$, where δ is the **significance level**.

Excursus: Hypothesis testing

- The goal in hypothesis testing is to reject a **null hypothesis** H_0 as being unlikely in light of the data, and hence to provide evidence in favor of the **alternative hypothesis** $H_1 = \neg H_0$.
- To this end, a suitable test statistic $T = T(\mathcal{D})$ is defined, so that the distribution of T is known under H_0.
- Suppose the distribution is concentrated in a "normal" range, so that T takes values in $A = [t_l, t_u] \subset \mathbb{R}$ with high probability $1 - \delta$, where δ is the **significance level**.
- That is, assuming H_0, it is unlikely to observe values $T \notin A$.

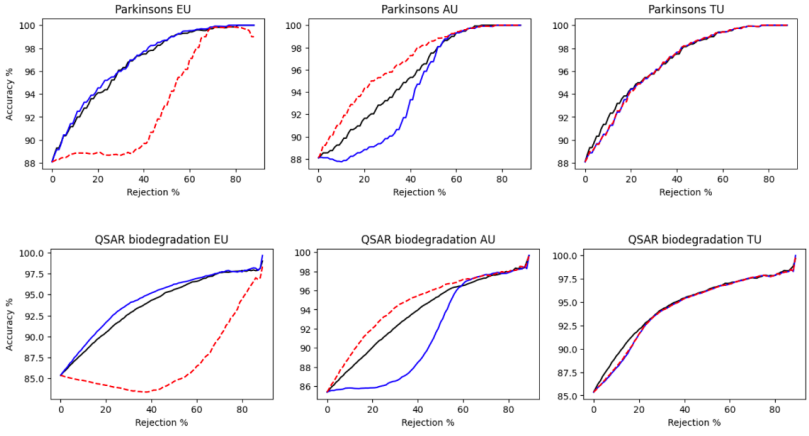
Excursus: Hypothesis testing

- The goal in hypothesis testing is to reject a **null hypothesis** H_0 as being unlikely in light of the data, and hence to provide evidence in favor of the **alternative hypothesis** $H_1 = \neg H_0$.
- To this end, a suitable test statistic $T = T(\mathcal{D})$ is defined, so that the distribution of T is known under H_0.
- Suppose the distribution is concentrated in a "normal" range, so that T takes values in $A = [t_l, t_u] \subset \mathbb{R}$ with high probability $1 - \delta$, where δ is the **significance level**.
- That is, assuming H_0, it is unlikely to observe values $T \notin A$.
- If this nevertheless happens, i.e., $T \in R = \mathbb{R} \setminus A$, then H_0 is rejected and H_1 accepted.

Evaluation: accuracy-rejection curves

Evaluation: accuracy-rejection curves

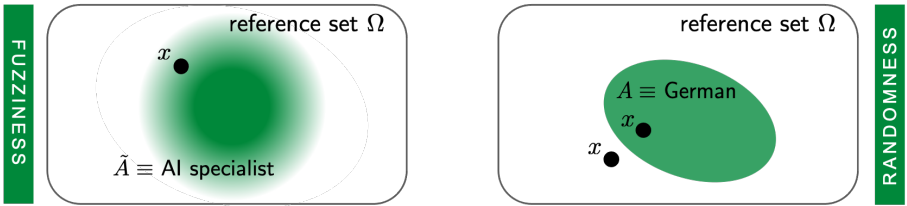
- Reject test instances for which (total, aleatoric, epistemic) uncertainty exceeds a certain threshold, measure accuracy on the remaining ones.



Facets of uncertainty

Facets of uncertainty

- Randomness, imprecision, inconsistency, ambiguity, vagueness, fuzziness, ...



"$x \in \tilde{A}$" uncertain due to non-sharp boundaries of $\tilde{A}$

$$\tilde{A}(x) = 0.7$$

"$x \in A$" for $A \subseteq \Omega$ uncertain due to lack of knowledge about x

$$P(A) = 0.7$$

"Fuzziness is orthogonal to probability theory – it focuses on the ambiguity of describing events, rather than the uncertainty about the occurrence or non-occurrence of events." | Judea Pearl (2000)

Excursus: Hypothesis testing

Excursus: Hypothesis testing

- Example: You want to convict a coin as being biased (unfair).

Excursus: Hypothesis testing

- Example: You want to convict a coin as being biased (unfair).
- $H_0 : \theta = \frac{1}{2}$, $H_1 : \theta \neq \frac{1}{2}$

Excursus: Hypothesis testing

- Example: You want to convict a coin as being biased (unfair).
- $H_0 : \theta = \frac{1}{2}$, $H_1 : \theta \neq \frac{1}{2}$
- $\mathcal{D} = \{x_1, \ldots, x_N\}$, outcomes of N coin flips

Excursus: Hypothesis testing

- Example: You want to convict a coin as being biased (unfair).
- $H_0 : \theta = \frac{1}{2}$, $H_1 : \theta \neq \frac{1}{2}$
- $\mathcal{D} = \{x_1, \ldots, x_N\}$, outcomes of N coin flips
- $T =$ number of heads in $\mathcal{D}$.

Excursus: Hypothesis testing

- Example: You want to convict a coin as being biased (unfair).
- $H_0 : \theta = \frac{1}{2}$, $H_1 : \theta \neq \frac{1}{2}$
- $\mathcal{D} = \{x_1, \ldots, x_N\}$, outcomes of N coin flips
- $T =$ number of heads in $\mathcal{D}$.
- Distribution under H_0 is binomial:

$$P\left(T = k \,|\, \theta = \frac{1}{2}\right) = B(k, 1/2) = \binom{N}{k} \left(\frac{1}{2}\right)^k \left(\frac{1}{2}\right)^{N-k} = \binom{N}{k} \left(\frac{1}{2}\right)^N$$

Excursus: Hypothesis testing

- Example: You want to convict a coin as being biased (unfair).
- $H_0 : \theta = \frac{1}{2}$, $H_1 : \theta \neq \frac{1}{2}$
- $\mathcal{D} = \{x_1, \ldots, x_N\}$, outcomes of N coin flips
- $T =$ number of heads in $\mathcal{D}$.
- Distribution under H_0 is binomial:

$$P\left(T = k \,|\, \theta = \frac{1}{2}\right) = B(k, 1/2) = \binom{N}{k} \left(\frac{1}{2}\right)^k \left(\frac{1}{2}\right)^{N-k} = \binom{N}{k} \left(\frac{1}{2}\right)^N$$

- Probability is highest around $k = N/2$ and decreases toward both ends (very high or very low number of heads).

Excursus: Hypothesis testing

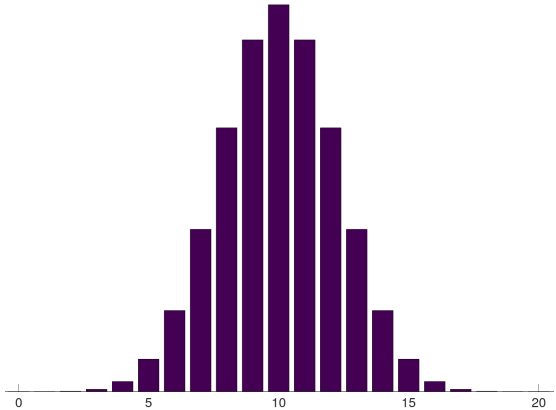
- Example: You want to convict a coin as being biased (unfair).
- $H_0 : \theta = \frac{1}{2}$, $H_1 : \theta \neq \frac{1}{2}$
- $\mathcal{D} = \{x_1, \ldots, x_N\}$, outcomes of N coin flips
- $T =$ number of heads in $\mathcal{D}$.
- Distribution under H_0 is binomial:

$$P\left(T = k \,|\, \theta = \frac{1}{2}\right) = B(k, 1/2) = \binom{N}{k} \left(\frac{1}{2}\right)^k \left(\frac{1}{2}\right)^{N-k} = \binom{N}{k} \left(\frac{1}{2}\right)^N$$

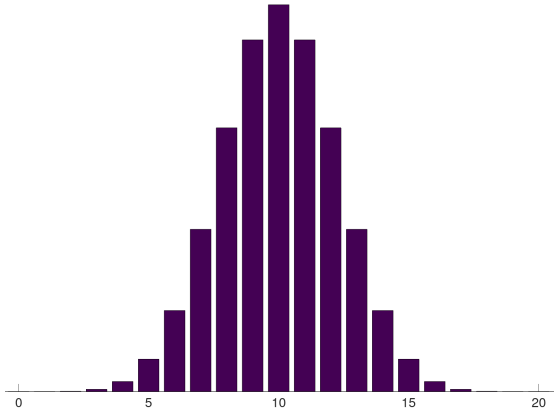
- Probability is highest around $k = N/2$ and decreases toward both ends (very high or very low number of heads).
- Take $R = \{0, 1, \ldots, r\} \cup \{N - r, \ldots, N\}$, with r the largest number such that

$$P(T \in R) = \sum_{k=0}^{r} B(k, 1/2) + \sum_{k=N-r}^{N} B(k, 1/2) \leq \delta\,.$$

Excursus: Hypothesis testing



Excursus: Hypothesis testing



- For $R = \{0, 1, \ldots, 5\} \cup \{15, \ldots, 20\}$,

$$P(T \in R) = 0.041389 \leq 0.05$$

Excursus: Hypothesis testing

Excursus: Hypothesis testing

- Note that test decisions might be **wrong**.

Excursus: Hypothesis testing

- Note that test decisions might be **wrong**.
- **Type-I error:** rejecting H_0 although it is true — the probability of this error is bounded by δ by construction.

Excursus: Hypothesis testing

- Note that test decisions might be **wrong**.
- **Type-I error:** rejecting H_0 although it is true — the probability of this error is bounded by δ by construction.
- **Type-II error:** not rejecting H_0 although it is false — the probability of this error depends on the (then unknown) ground truth.

Excursus: Hypothesis testing

- Note that test decisions might be **wrong**.
- **Type-I error:** rejecting H_0 although it is true — the probability of this error is bounded by δ by construction.
- **Type-II error:** not rejecting H_0 although it is false — the probability of this error depends on the (then unknown) ground truth.
- Also note that **nothing** can be concluded from the test in case H_0 is not rejected.

Excursus: Hypothesis testing

- Note that test decisions might be **wrong**.
- **Type-I error:** rejecting H_0 although it is true — the probability of this error is bounded by δ by construction.
- **Type-II error:** not rejecting H_0 although it is false — the probability of this error depends on the (then unknown) ground truth.
- Also note that **nothing** can be concluded from the test in case H_0 is not rejected.
- In particular, not rejecting H_0 does not mean one can "accept" it:

$$P\left(\theta = \frac{1}{2} \mid T \in \{6, \ldots, 14\}\right) = ??$$

Exchangeability

Exchangeability

- A (finite) **sequence of random variables** $X_1, X_2, \ldots, X_N$ is exchangeable, if the following holds for any permutation $\sigma : [N] \longrightarrow [N]$: the joint probability distribution of the random variables is the same as the joint distribution of the permuted sequence $X_{\sigma(1)}, X_{\sigma(2)}, \ldots, X_{\sigma(N)}$.

Exchangeability

- A (finite) **sequence of random variables** $X_1, X_2, \ldots, X_N$ is exchangeable, if the following holds for any permutation $\sigma : [N] \longrightarrow [N]$: the joint probability distribution of the random variables is the same as the joint distribution of the permuted sequence $X_{\sigma(1)}, X_{\sigma(2)}, \ldots, X_{\sigma(N)}$.
- This means that the cumulative distribution function

$$F_{X_1,\ldots,X_N} : \mathbb{R}^N \longrightarrow [0,1]$$

is **symmetric** in its arguments.

Exchangeability

- A (finite) **sequence of random variables** $X_1, X_2, \ldots, X_N$ is exchangeable, if the following holds for any permutation $\sigma : [N] \longrightarrow [N]$: the joint probability distribution of the random variables is the same as the joint distribution of the permuted sequence $X_{\sigma(1)}, X_{\sigma(2)}, \ldots, X_{\sigma(N)}$.
- This means that the cumulative distribution function

$$F_{X_1,\ldots,X_N} : \mathbb{R}^N \longrightarrow [0, 1]$$

  is **symmetric** in its arguments.
- Exchangeability is weaker than the i.i.d. assumption (the latter implies the former but not the other way around).

Inductive conformal prediction

Inductive conformal prediction

- Conformal prediction as outlined above realises **transductive inference**, i.e., inference directly targeting a query instance (known by the learner) without inducing a general model beforehand.

Inductive conformal prediction

- Conformal prediction as outlined above realises **transductive inference**, i.e., inference directly targeting a query instance (known by the learner) without inducing a general model beforehand.
- This is computationally costly, as it possibly requires refitting a model to the data in each iteration.

Inductive conformal prediction

- Conformal prediction as outlined above realises **transductive inference**, i.e., inference directly targeting a query instance (known by the learner) without inducing a general model beforehand.
- This is computationally costly, as it possibly requires refitting a model to the data in each iteration.
- **Inductive conformal prediction** (ICP) is an alternative that is computationally less expensive.

Inductive conformal prediction

- Conformal prediction as outlined above realises **transductive inference**, i.e., inference directly targeting a query instance (known by the learner) without inducing a general model beforehand.
- This is computationally costly, as it possibly requires refitting a model to the data in each iteration.
- **Inductive conformal prediction** (ICP) is an alternative that is computationally less expensive.
- In the **split-conformal prediction** variant, ICP splits the training data $\mathcal{D}$ into
  - proper training data $\mathcal{D}_T$ of size $N - M$,
  - calibration data $\mathcal{D}_C = \left\{(\boldsymbol{x}_j, y_j)\right\}_{j=1}^{M}$ of size $M < N$.

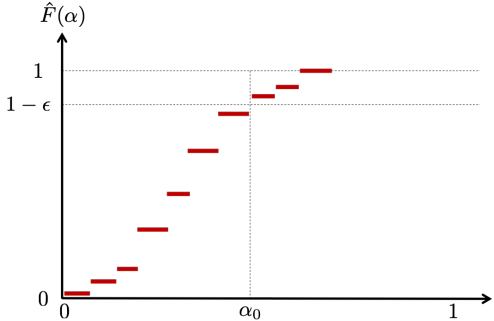
Inductive conformal prediction

Inductive conformal prediction

- Imagine, for example, a probabilistic classifier h is trained on the proper training data, and nonconformity of $(\boldsymbol{x}, y)$ is defined as $\alpha = 1 - p(y)$, where $p = h(\boldsymbol{x})$.

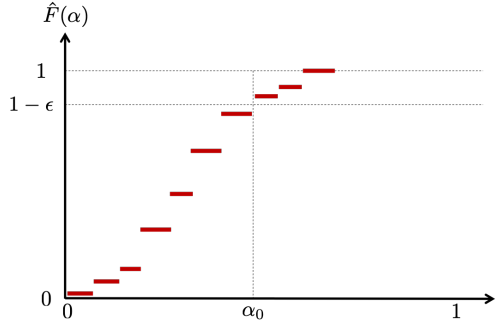
Inductive conformal prediction

- Imagine, for example, a probabilistic classifier h is trained on the proper training data, and nonconformity of $(\boldsymbol{x}, y)$ is defined as $\alpha = 1 - p(y)$, where $p = h(\boldsymbol{x})$.
- The scores on the calibration data define an empirical CDF ($\hat{F}(\alpha) =$ probability (relative frequency) of nonconformity scores $\leq \alpha$):



Inductive conformal prediction

- Imagine, for example, a probabilistic classifier h is trained on the proper training data, and nonconformity of $(\boldsymbol{x}, y)$ is defined as $\alpha = 1 - p(y)$, where $p = h(\boldsymbol{x})$.
- The scores on the calibration data define an empirical CDF ($\hat{F}(\alpha) =$ probability (relative frequency) of nonconformity scores $\leq \alpha$):



- With high probability, nonconformity of a "real" data point is $\leq \alpha_0$.

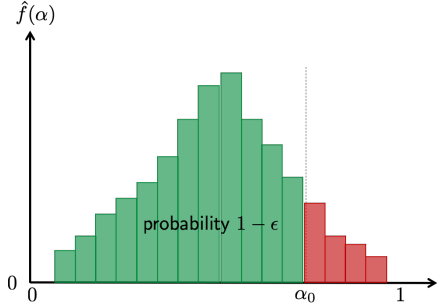
Inductive conformal prediction

Inductive conformal prediction

- Imagine, for example, a probabilistic classifier h is trained on the proper training data, and nonconformity of $(\boldsymbol{x}, y)$ is defined as $\alpha = 1 - p(y)$, where $p = h(\boldsymbol{x})$.

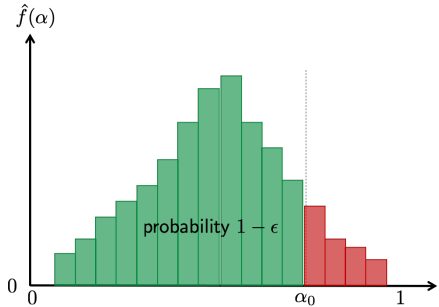
Inductive conformal prediction

- Imagine, for example, a probabilistic classifier h is trained on the proper training data, and nonconformity of $(\boldsymbol{x}, y)$ is defined as $\alpha = 1 - p(y)$, where $p = h(\boldsymbol{x})$.
- The scores on the calibration data define an empirical PDF ($\hat{f}(\alpha) =$ probability (relative frequency) of nonconformity scores α):



Inductive conformal prediction

- Imagine, for example, a probabilistic classifier h is trained on the proper training data, and nonconformity of $(\boldsymbol{x}, y)$ is defined as $\alpha = 1 - p(y)$, where $p = h(\boldsymbol{x})$.
- The scores on the calibration data define an empirical PDF ($\hat{f}(\alpha) =$ probability (relative frequency) of nonconformity scores α):



- With high probability, nonconformity of a "real" data point is $\leq \alpha_0$.

Inductive conformal prediction

Inductive conformal prediction

- A predictive model h is trained on the proper training set $\mathcal{D}_T$.

Inductive conformal prediction

- A predictive model h is trained on the proper training set $\mathcal{D}_T$.
- Using this model, the nonconformity scores $\alpha_1, \ldots, \alpha_M$ are computed for the examples in the calibration set — for example, if h predicts probabilities, scores could be defined as $\alpha_j = f(\boldsymbol{x}_j, y_j) = 1 - p(y_j \mid h, \boldsymbol{x}_j)$.

Inductive conformal prediction

- A predictive model h is trained on the proper training set $\mathcal{D}_T$.
- Using this model, the nonconformity scores $\alpha_1, \ldots, \alpha_M$ are computed for the examples in the calibration set — for example, if h predicts probabilities, scores could be defined as $\alpha_j = f(\boldsymbol{x}_j, y_j) = 1 - p(y_j \mid h, \boldsymbol{x}_j)$.
- Define $\hat{q}$ to be the $\lceil (M+1)(1-\epsilon) \rceil / M$ empirical quantile of $\alpha_1, \ldots, \alpha_M$.

Inductive conformal prediction

- A predictive model h is trained on the proper training set $\mathcal{D}_T$.
- Using this model, the nonconformity scores $\alpha_1, \ldots, \alpha_M$ are computed for the examples in the calibration set — for example, if h predicts probabilities, scores could be defined as $\alpha_j = f(\boldsymbol{x}_j, y_j) = 1 - p(y_j \mid h, \boldsymbol{x}_j)$.
- Define $\hat{q}$ to be the $\lceil (M+1)(1-\epsilon) \rceil / M$ empirical quantile of $\alpha_1, \ldots, \alpha_M$.
- For a query $\boldsymbol{x}_q \in \mathcal{X}$, construct the prediction set as follows:

$$C(\boldsymbol{x}_q) = \left\{ y \in \mathcal{Y} \mid f(\boldsymbol{x}_q, y) \leq \hat{q} \right\}$$

Inductive conformal prediction

- A predictive model h is trained on the proper training set $\mathcal{D}_T$.
- Using this model, the nonconformity scores $\alpha_1, \ldots, \alpha_M$ are computed for the examples in the calibration set — for example, if h predicts probabilities, scores could be defined as $\alpha_j = f(\boldsymbol{x}_j, y_j) = 1 - p(y_j \mid h, \boldsymbol{x}_j)$.
- Define $\hat{q}$ to be the $\lceil (M+1)(1-\epsilon) \rceil / M$ empirical quantile of $\alpha_1, \ldots, \alpha_M$.
- For a query $\boldsymbol{x}_q \in \mathcal{X}$, construct the prediction set as follows:

$$C(\boldsymbol{x}_q) = \left\{ y \in \mathcal{Y} \mid f(\boldsymbol{x}_q, y) \leq \hat{q} \right\}$$

- Assuming $\mathcal{D}$ to be an i.i.d. sample, and $(\boldsymbol{x}_q, y_q)$ sampled from the same distribution, the following estimate is provably correct:

$$1 - \epsilon \leq P\big(y_q \in Y(\boldsymbol{x}_q)\big) \leq 1 - \epsilon + \frac{1}{M+1}$$