

# Meaningful Metrics for Probabilistic Predictions

Mark van der Wilk  
Imperial College London

2023-07-18

# Hello

- Associate Professor at Imperial College London
- I work on a wide variety of topics in ML / Probabilistic Inference:
  - Neural Architecture Search (Bayesian Model Selection)
  - Causality
  - Continual Learning
  - Generative Models
  - ...
- Today, I *won't* (directly) talk about my research.

# Metrics

- Main Q: How can we *meaningfully* assess a predictor's performance?
  - Particularly when we consider *probabilistic* predictors
- In *applied* projects, this is where a lot of the thinking happens.
  - I see this in applied collaborations / consultancy projects
  - IMO: The most exciting ML methodology development is happening on the boundary with applications (ML for science!)
    - Protein design, drug discovery, material property prediction

❗ Often: Someone *builds* a system, sees problem instances, and tries to fix these (whack-a-mole).

💡 Specify your problem well, let this *imply* your metric, and *design* your system to optimise metric.



# Metrics in Deep Learning

The motor behind the progress

# Metrics drove Progress in Deep Learning

Common ML workflow:

- Train a model on training data somehow, obtain predictor:

$$\hat{y} = \operatorname{argmax} f_{\theta}(\hat{x})$$

- Evaluate on test set *[Math Processing Error]*
- Make improvements, and repeat

# Measuring Performance in Deep Learning

Focus on optimising this led to meaningful improvements, **because**

- People **agreed** on metrics
- People **understood** the value of improving the metric

💡 Accuracy is a common, easy-to-understand metric. *[Math Processing Error]*

Helpful because:

- In some cases, mistakes can be brought down to near zero.
- *“I know how bad a mistake is, so I know the value of making fewer mistakes.”*

## **Not as Easy for Probabilistic Models!**

Why are metrics harder for Probabilistic Models?

# Probabilities, Losses, Decisions

🚧 Why not just use the test losses that are common in deep learning?

Yesterday you saw how to quantify uncertainty. Output of a model is a *distribution*, e.g. for classification:

$$p_{Y|X,D} (Y = y|X = x, \mathcal{D}) = p_y$$

⚠ This gives us a distribution, rather than a direct class decision!

❓ Q: Given a probabilistic prediction, how do we get a *decision* that we can evaluate with a loss function?



# Decision Theory

We get a result as a consequence of taking an action  $a \in \mathcal{A}$  and observing an outcome of  $X \in \Omega$ .

 Simple axioms for making decisions:

1. You either prefer one result over the other, or you are indifferent.
2. If you prefer A over B and B over C, then you prefer A over C.
3. “Reduction of compound lotteries” (too long, there’s always one...)

 Von Neumann-Morgenstern Utility Theorem states that

- There exists a *utility function*  $u : \Omega \times \mathcal{A} \rightarrow \mathbb{R}$
- That leads to an *expected utility*  $U(a) = \mathbb{E}_X [u(X, a)]$
- Which the selected action maximises  $a = \operatorname{argmax}_{\alpha} U(\alpha)$



# Decision Theory

Simple principle, but explains / implies complex behaviours

- Rational choice theory
  - Economics, psychology, political theory
- Exploration-exploitation
  - RL, experimental design, statistics
- Implies a philosophical stance (ethical?)
  - Utilitarianism / consequentialism

# From Probability to Hard Classification

Where were we?

① Q: Given a probabilistic prediction, how do we get a *decision* that we can evaluate with a loss function?


Let's take *accuracy*


*[Math Processing Error]*

I.e. select the class with the highest predicted probability.

# Losses from Deep Learning

- We got a probabilistic prediction,
- followed decision theory for a common deep learning loss,
- and got the same result as was intuitively obtained in DL...

 Why not just use the test losses that are common in deep learning?

 Given equal point estimates, uncertainty estimates do not influence deep learning evaluation metrics.

Evaluation metrics from deep learning do not benefit from uncertainty!

This is also the case for regression. If you have a squared loss  $\ell(a, y) = (a - y)^2$ , only the mean point estimate influences the action and resulting expected loss. Exercise!

# Where Does Uncertainty Help?

Uncertainty does help in certain tasks!

- Asymmetric losses: If making one error is worse than another.
- Additional actions: Imagine a “send to human” action, if uncertain.

💡 Can show that loss is reduced if given good uncertainty, in these situations.

# Calibration

🚧 Ok, so we want good accuracy. Let's keep that. Can we not *also* want the uncertainties to make sense?

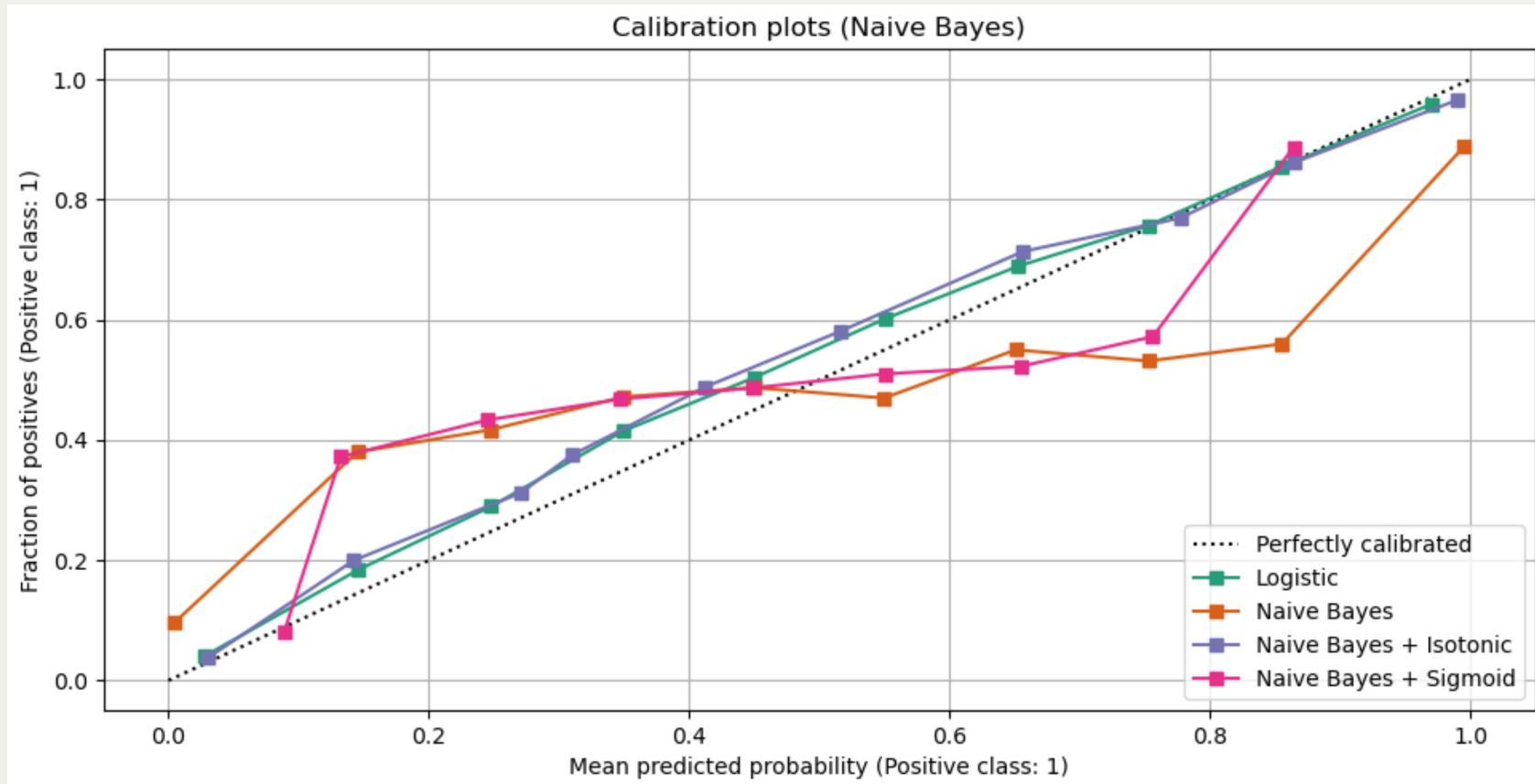
❗ Can we measure the discrepancy between the predicted probabilities  $p_y$  and the frequency with which the outcomes truly appear?

People have introduced *calibration* metrics to additionally measure whether uncertainties make sense.

I.e. if we predict something will happen with  $p = 0.2$ , does it actually occur this frequently?

- Bin predictions by confidence of occurrence of top class
- Measure how often correct class occurs

# Expected Calibration Error



Can plot bins in a *calibration curve*.



# Expected Calibration Error

## 💡 Expected Calibration Error

$$\text{ECE} = \sum_b \frac{n_b}{N} |acc(b) - conf(b)|$$

Perfect calibration: Outcome frequency match predicted probabilities.

⚠ Trivial solutions can obtain perfect calibration! E.g. on MNIST classification, predicting  $p = 0.1$  for all classes, all the time.

# Metrics in Probabilistic ML

🚧 Ok, so we want good accuracy. Let's keep that. Can we not *also* want the uncertainties to make sense?

- ❗ • Clearly, we prefer better calibration for equal accuracy.
- But how to weigh, if one has better calibration but worse accuracy?

⚠️ Why are metrics harder for Probabilistic Models?

- Interpretable metrics from deep learning are insensitive to uncertainty
- Adding ECE: Multi-metric problems are complicated
- Not intuitively clear how much *value* an improvement in ECE gives!

## One score to rule them all?

Can we find a *single* score that summarises the quality of a probabilistic prediction?

# Goal

Previously, we were looking at a classification setting, with

- Predictive distribution  $p_{Y|x, \mathcal{D}}(y|x, \mathcal{D})$
- Data generating distribution  $\pi_{X,Y}(x, y) = \pi_{Y|X}(y|x)\pi_X(x)$

We want to measure how different  $p_{Y|x, \mathcal{D}}(y|x, \mathcal{D})$  and  $\pi_{Y|X}(y|x)$  are on average over  $\pi_X$ .

We will focus on the simplified problem:

❗ Can we develop a measure of the difference (score) between two distributions that captures *all* ways the distribution can differ?

For our classification problem, we can average this score over  $\pi_X$ , and we will obtain the same properties (exercise!).

# Scoring Rules

- A scoring rule maps a distribution and outcome to some real value:

$$S : \mathbb{P} \times \Omega \rightarrow \mathbb{R}$$

- We observe real data from a distribution  $\pi$
- Averaging score over many test points from  $\pi$  gives  $S : \mathbb{P} \times \mathbb{P} \rightarrow \mathbb{R}$ :

$$S(p, \pi) = \mathbb{E}_{\pi_X} [S(p, X)] = \int S(p, x)p(x)dx$$

💡 A *proper scoring rule* is a rule for which

$$S(p, \pi) \geq S(\pi, \pi) ,$$

i.e. it is *minimised* when you have the true distribution.

<https://mvdw.uk>

# Scoring Rules

Scoring rules care about *all* aspects of the distribution

If you minimise it, you get:

- perfect calibration
- minimal losses on all other loss functions (e.g. accuracy)

Every strictly proper scoring rule also implies a *divergence* measure between distributions.

# Metrics in Probabilistic ML


Situation is complicated: Many metrics to choose from!

Consider classification with predictive distribution

$$p_{Y_i|x_i, \mathcal{D}}(Y_i = y|x_i, \mathcal{D}) = p_y$$

- Log likelihood:  $\log p_y$
- Brier score:  $\sum_o (p_y - \mathbf{1}_{y_o=0})^2$
- Spherical score:  $p_y / \sqrt{\sum_o p_o^2}$

⚠ All proper scoring rule satisfy requirements, but penalise various discrepancies differently! Which one should we choose?

 **One score to rule them all?**

Can we justify using one scoring rule, rather than another?



# Probabilities, Losses, Decisions... Scores?

Previously we saw that we could go from a probabilistic prediction to a decision using decision theory.

Here we saw we could go from a probability distribution to a score.

 In both cases, we had to define losses. Are they related?

 Starting at a decision-making problem, do we get a scoring rule?

 Yes! Strictly proper scoring rule under certain conditions.

*Geometry of Proper Scoring Rules*, A. P. Dawid (2007)

# Most Appropriate Scoring Rule

⚠ So which scoring rule should we choose?

- 💡 • We should define a loss function that relates to our *task*
- The expected utility given optimal actions gives our *score*
- If we define things right, we should be able to interpret and *value* any improvement!

# Example: Kelly Betting

*A New Interpretation of Information Rate*, J. L. Kelly (1956)

- Different levels of risk aversion lead to  $\alpha$ -divergences!
- I don't know the problem that corresponds to the Brier score...

# Thoughts & Conclusions

- Deep Learning test metrics are often not suitable for probabilistic models
- To see the *value* of probabilistic models, we need more complex (realistic!) testing scenarios
- Decision-making problems are the right way to do this
  - they are *meaningful*
  - and naturally lead to other proposed evaluations
- The log-loss corresponds to a natural decision-making problem. Improvements are interpretable.

💡 Opinion: The best way to do *impactful* work in probabilistic modelling, is to show an improvement in a *decision-making* task.

# Further Reading

There is so much knowledge that has been figured out, but which is not in UG curriculums. I didn't properly understand this until *after* my PhD, but it was well worth developing a deeper understanding of this.

I would recommend going through the derivation of the Kelly criterion.

- Kelly Criterion Wikipedia
- Kelly Criterion Paper
- The Geometry of Proper Scoring Rules

